

## SIMATIC

## Programmer avec STEP 7

### Manuel

Avant-propos	
Sommaire	
Introduction sur le produit et installation	<b>1</b>
Installation	<b>2</b>
Conception de la solution d'automatisation	<b>3</b>
Principes de conception d'une structure de programme	<b>4</b>
Démarrage et utilisation du programme	<b>5</b>
Création et édition du projet	<b>6</b>
Edition de projets avec des versions différentes de STEP 7	<b>7</b>
Définition de mnémoniques	<b>8</b>
Création de blocs et de bibliothèques	<b>9</b>
Création de blocs de code	<b>10</b>
Création de blocs de données	<b>11</b>
Paramétrage de blocs de données	<b>12</b>
Création de sources LIST	<b>13</b>
Affichage des données de référence	<b>14</b>
Vérifier la cohérence des blocs et horodatage comme propriété de bloc	<b>15</b>
Configuration de messages	<b>16</b>
Contrôle-commande de variables	<b>17</b>
Etablissement d'une liaison en ligne et choix de la CPU	<b>18</b>
Chargement	<b>19</b>
Test avec des tables de variables	<b>20</b>
Test avec la visualisation d'état du programme	<b>21</b>
Test avec le programme de simulation (logiciel optionnel)	<b>22</b>
Diagnostic	<b>23</b>
Impression et archivage	<b>24</b>
Utilisation des systèmes d'automatisation M7	<b>25</b>
Astuces et conseils	<b>26</b>
Annexe	<b>A</b>
Index	

Ce manuel est livré avec la documentation référencée :  
**6ES7810-4CA08-8CW0**

## Consignes de sécurité

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité et pour éviter des dommages matériels. Les avertissements servant à votre sécurité personnelle sont accompagnés d'un triangle de danger, les avertissements concernant uniquement des dommages matériels sont dépourvus de ce triangle. Les avertissements sont représentés ci-après par ordre décroissant de niveau de risque.



### **Danger**

signifie que la non-application des mesures de sécurité appropriées entraîne la mort ou des blessures graves.



### **Attention**

signifie que la non-application des mesures de sécurité appropriées peut entraîner la mort ou des blessures graves.



### **Prudence**

signifie que la non-application des mesures de sécurité appropriées peut entraîner des blessures légères.

### **Prudence**

signifie que la non-application des mesures de sécurité appropriées peut entraîner un dommage matériel.

### **Important**

signifie que le non-respect de l'avertissement correspondant peut entraîner l'apparition d'un événement ou d'un état indésirable.

En présence de plusieurs niveaux de risque, c'est toujours l'avertissement correspondant au niveau le plus élevé qui est reproduit. Si un avertissement avec triangle de danger prévient des risques de dommages corporels, le même avertissement peut aussi contenir un avis de mise en garde contre des dommages matériels.

## Personnel qualifié

L'installation et l'exploitation de l'appareil/du système concerné ne sont autorisées qu'en liaison avec la présente documentation. La mise en service et l'exploitation d'un appareil/système ne doivent être effectuées que par des personnes qualifiées. Au sens des consignes de sécurité figurant dans cette documentation, les personnes qualifiées sont des personnes qui sont habilitées à mettre en service, à mettre à la terre et à identifier des appareils, systèmes et circuits en conformité avec les normes de sécurité.

## Utilisation conforme à la destination

Tenez compte des points suivants :



### **Attention**

L'appareil/le système ne doit être utilisé que pour les applications spécifiées dans le catalogue ou dans la description technique, et uniquement en liaison avec des appareils et composants recommandés ou agréés par Siemens s'ils ne sont pas de Siemens.

Le transport, le stockage, le montage, la mise en service ainsi que l'utilisation et la maintenance Le fonctionnement correct et sûr du produit implique son transport, stockage, montage et mise en service selon les règles de l'art ainsi qu'une utilisation et maintenance soigneuses.

## Marque de fabrique

Toutes les désignations repérées par ® sont des marques déposées de Siemens AG. Les autres désignations dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfreindre les droits de leurs propriétaires respectifs.

## Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent document avec le matériel et le logiciel qui y sont décrits. Ne pouvant toutefois exclure toute divergence, nous ne pouvons pas nous porter garants de la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition.

# Avant-propos

## Objet de ce manuel

Ce manuel vous procure une vue d'ensemble sur la programmation avec **STEP 7**. Il a pour but de vous assister lors de l'installation et du démarrage du logiciel. Il explique la démarche de création de programmes et décrit les différents éléments d'un programme utilisateur.

Ce manuel s'adresse aux personnes chargées de réaliser des tâches d'automatisation avec le logiciel STEP 7 et mettant en œuvre des systèmes d'automatisation SIMATIC S7.

Nous vous recommandons de vous familiariser tout d'abord avec les exemples du manuel "Getting Started de STEP 7". Ils représentent une approche simple de la thématique traitée plus en profondeur dans le manuel "Programmer avec STEP 7".

## Connaissances fondamentales requises

La compréhension du manuel requiert des connaissances générales dans le domaine de la technique d'automatisation.

Nous supposons en outre des connaissances dans l'utilisation d'ordinateurs ou autres équipements (par exemple consoles de programmation) analogues au PC et des systèmes d'exploitation MS Windows 2000 Professional, MS Windows XP Professional ou MS Windows Server 2003.

## Domaine de validité du manuel

Le présent manuel est valable pour le logiciel STEP 7 V5.4.

Vous trouvez des informations relatives aux Servicepack paraissant après la publication de ce manuel dans :

- le fichier "Lisezmoi.wri",
- l'aide en ligne actualisée de STEP 7.

Le thème "Nouveautés ?" de l'aide en ligne vous permet une approche aisée ainsi qu'une bonne vue d'ensemble sur les innovations dans STEP 7.

## Documentation de STEP 7

Ce manuel fait partie de la documentation "STEP 7 Connaissances fondamentales". Le tableau suivant présente la documentation de STEP 7 :

Manuel	Objet	Numéro de référence
STEP 7 Connaissances fondamentales avec <ul style="list-style-type: none"> <li>• STEP 7 Getting Started</li> <li>• Programmer avec STEP 7</li> <li>• Configuration matérielle et communication dans STEP 7</li> <li>• STEP 7 Pour une transition facile de S5 à S7</li> </ul>	Connaissances fondamentales pour le personnel technique. Décrit la marche à suivre pour réaliser des tâches d'automatisation avec STEP 7 et S7-300/400.	6ES7810-4CA08-8CW0
STEP 7 Connaissances de référence avec <ul style="list-style-type: none"> <li>• Langages CONT/LOG/LIST pour SIMATIC S7-300/400</li> <li>• Logiciel système pour SIMATIC S7-300/400 Fonctions standard et fonctions système Volume 1 et Volume 2</li> </ul>	Connaissances de référence. Décrit les langages de programmation CONT, LOG et LIST de même que les fonctions standard et les fonctions système en complément des connaissances fondamentales de STEP 7.	6ES7810-4CA08-8CW1

Aides en ligne	Objet	Numéro de référence
Aide de STEP 7	Connaissances fondamentales pour la programmation ainsi que pour la configuration du matériel avec STEP 7, sous forme d'aide en ligne.	Fait partie du logiciel STEP 7
Aides de référence de LIST/CONT/LOG Aide de référence sur les SFB/SFC Aide de référence sur les blocs d'organisation	Connaissances de référence contextuelles	Fait partie du logiciel STEP 7

## Aide en ligne

En complément au manuel, l'aide en ligne intégrée au logiciel vous offre une assistance détaillée lors de l'utilisation du logiciel.

Ce système d'aide est intégré au logiciel grâce à plusieurs interfaces :

- Le menu d'aide ? propose plusieurs commandes : **Rubriques d'aide** ouvre le sommaire de l'aide de STEP 7.
- **Utiliser l'aide** fournit des instructions détaillées sur l'utilisation de l'aide en ligne.
- L'aide contextuelle donne des informations sur le contexte actuel, par exemple sur une boîte de dialogue ouverte ou sur une fenêtre active. Vous l'appellez en cliquant sur le bouton "Aide" ou en appuyant sur la touche F1.
- La barre d'état constitue une autre forme d'aide contextuelle. Lorsque le curseur est positionné sur une commande, elle en affiche une description succincte.
- Une description succincte des boutons de la barre d'outils s'affiche également lorsque le curseur y est positionné quelques instants.

Si vous préférez consulter les informations de l'aide en ligne sur papier, vous avez la possibilité d'imprimer des rubriques d'aide individuelles, des livres ou l'ensemble de l'aide.

Ce manuel tout comme les manuels "Configuration matérielle et communication dans STEP 7", "Modifications de l'installation en fonctionnement au moyen de CiR" et "Système d'automatisation S7-400H - Systèmes à haute disponibilité" sont extraits de l'aide de STEP 7 basée sur HTML. Si vous désirez des instructions plus détaillées, référez vous à l'aide de STEP 7. En raison de la structure similaire entre le manuel et l'aide en ligne, le passage de l'un à l'autre est aisé.

Après l'installation de STEP 7, vous trouvez les manuels électroniques sous **Démarrer > SIMATIC > Documentation**.

## Assistance supplémentaire

Si des questions sont restées sans réponse dans ce manuel, veuillez vous adresser à votre interlocuteur Siemens dans la filiale ou l'agence de votre région.

Vous trouvez votre interlocuteur sous :

<http://www.siemens.com/automation/partner>

Vous trouvez un fil rouge pour la recherche de documentations techniques sur les produits et systèmes SIMATIC à l'adresse suivante sur Internet :

<http://www.siemens.com/simatic-tech-doku-portal>

Le catalogue en ligne et le système de commande en ligne se trouvent à l'adresse :

<http://mall.automation.siemens.com/>

## Centre de formation SIMATIC

Nous proposons des cours de formation pour vous faciliter l'apprentissage des automates programmables SIMATIC S7. Veuillez vous adresser à votre centre de formation régional ou au centre principal à D 90327 Nuremberg.

Téléphone : +49 (911) 895-3200.

Internet: <http://www.sitrain.com>

## Technical Support

Vous pouvez joindre le support technique pour tous les produits A&D

- Via le formulaire Web de demande d'assistance (Support Request)  
<http://www.siemens.com/automation/support-request>
- Téléphone : + 49 180 5050 222
- Télécopie : + 49 180 5050 223

Vous trouvez plus d'informations concernant notre Technical Support sur Internet à l'adresse suivante :

<http://www.siemens.com/automation/service>

## Service & Support sur Internet

En plus de la documentation offerte, vous trouvez la totalité de notre savoir-faire en ligne sur Internet à l'adresse suivante :

<http://www.siemens.com/automation/service&support>

Vous y trouvez :

- le bulletin d'informations qui vous fournit constamment les dernières informations sur le produit,
- les documents dont vous avez besoin à l'aide de la fonction de recherche du Service & Support,
- le forum où utilisateurs et spécialistes peuvent échanger informations,
- votre interlocuteur Automation & Drives sur place,
- des informations sur le service après-vente, les réparations, les pièces de rechange à la rubrique "Service

# Sommaire

<b>1</b>	<b>Introduction sur le produit et installation .....</b>	<b>1-1</b>
1.1	Guide de STEP 7 .....	1-1
1.2	Logiciel de base STEP 7 .....	1-6
1.3	Nouveautés dans la version 5.4 de STEP 7 .....	1-10
1.4	Possibilités d'extension du logiciel de base STEP 7 .....	1-12
1.4.1	Applications techniques .....	1-13
1.4.2	Logiciels exécutables .....	1-15
1.4.3	Interface homme/machine .....	1-17
<b>2</b>	<b>Installation .....</b>	<b>2-1</b>
2.1	Automation License Manager .....	2-1
2.1.1	Licence d'utilisation avec Automation License Manager .....	2-1
2.1.2	Installation de Automation License Manager .....	2-4
2.1.3	Règles d'utilisation des License Keys (clé de licence) .....	2-5
2.2	Installation de STEP 7 .....	2-6
2.2.1	Marche à suivre pour l'installation de STEP 7 .....	2-7
2.2.2	Paramétrage de l'interface PG/PC .....	2-10
2.3	Désinstallation de STEP 7 .....	2-11
<b>3</b>	<b>Conception de la solution d'automatisation .....</b>	<b>3-1</b>
3.1	Conception d'une solution d'automatisation .....	3-1
3.2	Subdivision du processus en tâches et zones .....	3-2
3.3	Description des différentes zones fonctionnelles .....	3-4
3.4	Liste des entrées, sorties et entrées/sorties .....	3-6
3.5	Création d'un diagramme d'entrées/sorties pour les moteurs .....	3-7
3.6	Création d'un diagramme d'entrées/sorties pour les soupapes .....	3-8
3.7	Définition des exigences en matière de sécurité .....	3-9
3.8	Description des éléments de signalisation et de commande requis .....	3-10
3.9	Création du schéma de configuration .....	3-11
<b>4</b>	<b>Principes de conception d'une structure de programme .....</b>	<b>4-1</b>
4.1	Programmes dans une CPU .....	4-1
4.2	Blocs dans le programme utilisateur .....	4-2
4.2.1	Blocs d'organisation et structure du programme .....	4-3
4.2.2	Hiérarchie d'appel dans le programme utilisateur .....	4-10
4.2.3	Catégories de blocs .....	4-12
4.2.3.1	Bloc d'organisation pour le traitement de programme cyclique (OB1) .....	4-12
4.2.3.2	Fonctions (FC) .....	4-17
4.2.3.3	Blocs fonctionnels (FB) .....	4-19
4.2.3.4	Blocs de données d'instance .....	4-22
4.2.3.5	Blocs de données globaux (DB) .....	4-25
4.2.3.6	Blocs fonctionnels système (SFB) et fonctions système (SFC) .....	4-26
4.2.4	Blocs d'organisation pour le traitement de programme déclenché par alarme .....	4-28
4.2.4.1	Blocs d'organisation pour l'alarme horaire (OB10 à OB17) .....	4-29
4.2.4.2	Blocs d'organisation pour l'alarme temporisée (OB20 à OB23) .....	4-31
4.2.4.3	Blocs d'organisation pour l'alarme cyclique (OB30 à OB38) .....	4-32

4.2.4.4	Blocs d'organisation pour l'alarme de processus (OB40 à OB47) .....	4-34
4.2.4.5	Blocs d'organisation pour la mise en route (OB100 / OB101 / OB102) .....	4-35
4.2.4.6	Bloc d'organisation pour l'exécution du programme en arrière-plan (OB90) .....	4-37
4.2.4.7	Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122).....	4-39
<b>5</b>	<b>Démarrage et utilisation du programme .....</b>	<b>5-1</b>
5.1	Démarrage de STEP 7 .....	5-1
5.2	Démarrage de STEP 7 avec des paramètres initiaux prédéfinis.....	5-2
5.3	Appel des fonctions d'aide .....	5-4
5.4	Objets et hiérarchie d'objets .....	5-5
5.4.1	Objet Projet .....	5-6
5.4.2	Objet Bibliothèque.....	5-7
5.4.3	Objet Station .....	5-8
5.4.4	Objet Module programmable .....	5-10
5.4.5	Objet Programme S7/M7 .....	5-12
5.4.6	Objet Dossier Blocs.....	5-14
5.4.7	Objet Dossier Sources .....	5-17
5.4.8	Programme S7/M7 sans station ni CPU .....	5-18
5.5	Interface utilisateur et manipulation .....	5-19
5.5.1	Concept d'utilisation.....	5-19
5.5.2	Structure de la fenêtre .....	5-20
5.5.3	Éléments dans les boîtes de dialogue .....	5-21
5.5.4	Création et manipulation d'objets .....	5-22
5.5.5	Sélection d'objets dans les boîtes de dialogue.....	5-28
5.5.6	Historique des sessions.....	5-29
5.5.7	Modification de la disposition des fenêtres.....	5-29
5.5.8	Enregistrement et restauration de la disposition des fenêtres .....	5-30
5.6	Utilisation du clavier .....	5-31
5.6.1	Combinaisons de touches pour les commandes de menu.....	5-31
5.6.2	Combinaisons de touches pour le déplacement du curseur .....	5-33
5.6.3	Combinaisons de touches pour la sélection de texte .....	5-35
5.6.4	Combinaisons de touches pour accéder à l'aide en ligne .....	5-35
5.6.5	Combinaisons de touches pour la bascule entre les différents types de fenêtres ...	5-36
<b>6</b>	<b>Création et édition du projet .....</b>	<b>6-1</b>
6.1	Structure du projet .....	6-1
6.2	Informations sur la protection d'accès .....	6-3
6.3	Informations sur le journal des modifications .....	6-4
6.4	Utilisation de jeux de caractères de langues étrangères.....	6-5
6.5	Paramétrage de la langue de Windows.....	6-8
6.6	Création d'un projet.....	6-9
6.6.1	Création d'un projet.....	6-9
6.6.2	Insertion de stations.....	6-11
6.6.3	Insertion d'un programme S7/M7 .....	6-12
6.7	Édition d'un projet .....	6-14
6.7.1	Recherche des logiciels requis pour un projet.....	6-15
6.8	Gestion multilingue des textes.....	6-15
6.8.1	Types de textes à gestion multilingue.....	6-17
6.8.2	Structure du fichier d'exportation .....	6-18
6.8.3	Gestion de textes utilisateur dont la police de langue n'est pas installée .....	6-19
6.8.4	Informations sur le fichier-journal.....	6-20
6.8.5	Optimisation du modèle à traduire.....	6-21
6.8.6	Optimisation de la traduction .....	6-22



6.9	Carte mémoire micro (MMC) en tant que support de données .....	6-23
6.9.1	Ce qu'il faut savoir sur les cartes mémoire micro (MMC) .....	6-23
6.9.2	Utilisation d'une micro-carte mémoire (MMC) comme support de données .....	6-24
6.9.3	Fichier carte mémoire .....	6-24
6.9.4	Enregistrement des données du projet sur une micro-carte mémoire (MMC) .....	6-25
<b>7</b>	<b>Edition de projets avec des versions différentes de STEP 7 .....</b>	<b>7-1</b>
7.1	Edition de projets et bibliothèques de la version 2 .....	7-1
7.2	Extension d'esclaves DP créés avec des versions antérieures de STEP 7 .....	7-1
7.3	Editer des configurations actuelles avec des versions antérieures de STEP 7 .....	7-3
7.4	Utiliser les configurations PC SIMATIC à partir des versions précédentes.....	7-4
7.5	Représentation de modules ayant été configurés avec des versions plus récentes de STEP 7 ou avec un progiciel optionnel .....	7-6
<b>8</b>	<b>Définition de mnémoniques .....</b>	<b>8-1</b>
8.1	Adressage absolu et adressage symbolique .....	8-1
8.2	Mnémoniques globaux et mnémoniques locaux .....	8-3
8.3	Représentation des mnémoniques globaux et des mnémoniques locaux .....	8-4
8.4	Définition de la priorité de l'opérande (symbolique/absolu).....	8-5
8.5	Table des mnémoniques pour mnémoniques globaux.....	8-8
8.5.1	Structure et éléments de la table des mnémoniques .....	8-8
8.5.2	Opérandes et types de données autorisés dans la table des mnémoniques.....	8-10
8.5.3	Mnémoniques incomplets ou non univoques dans la table des mnémoniques .....	8-11
8.6	Possibilités de saisie de mnémoniques globaux .....	8-12
8.6.1	Remarques générales sur la saisie de mnémoniques.....	8-12
8.6.2	Saisie de mnémoniques globaux individuels dans des boîtes de dialogue.....	8-13
8.6.3	Saisie de plusieurs mnémoniques globaux dans la table des mnémoniques .....	8-14
8.6.4	Majuscules/minuscules pour les mnémoniques .....	8-15
8.6.5	Exportation et importation de tables de mnémoniques .....	8-17
8.6.6	Formats de fichier pour l'importation/exportation d'une table des mnémoniques ....	8-17
8.6.7	Edition de zones dans des tables de mnémoniques .....	8-20
<b>9</b>	<b>Création de blocs et de bibliothèques .....</b>	<b>9-1</b>
9.1	Choix de la méthode de création .....	9-1
9.2	Choix du langage de programmation.....	9-2
9.2.1	Langage de programmation CONT (schéma à contacts).....	9-4
9.2.2	Langage de programmation LOG (logigramme).....	9-5
9.2.3	Langage de programmation LIST (liste d'instructions) .....	9-6
9.2.4	Langage de programmation SCL.....	9-7
9.2.5	Langage de programmation GRAPH (commande séquentielle) .....	9-8
9.2.6	Langage de programmation HiGraph (graphe d'état).....	9-9
9.2.7	Langage de programmation CFC .....	9-11
9.3	Ce qu'il faut savoir pour créer des blocs.....	9-12
9.3.1	Dossier Blocs .....	9-12
9.3.2	Types de données utilisateur (UDT).....	9-13
9.3.3	Attributs de bloc .....	9-14
9.3.4	Affichage de la longueur des blocs.....	9-16
9.3.5	Comparaison de blocs .....	9-17
9.3.6	Réassignation .....	9-20
9.3.7	Attributs pour blocs et pour paramètres.....	9-20
9.4	Utilisation de bibliothèques .....	9-21
9.4.1	Structure hiérarchique des bibliothèques .....	9-23
9.4.2	Présentation des bibliothèques standard .....	9-23

<b>10</b>	<b>Création de blocs de code .....</b>	<b>10-1</b>
10.1	Principes de la création de blocs de code .....	10-1
10.1.1	Structure de la fenêtre de l'éditeur de programmes .....	10-1
10.1.2	Marche à suivre pour la création de blocs de code .....	10-3
10.1.3	Présélections pour l'éditeur de programmes CONT/LOG/LIST .....	10-4
10.1.4	Droits d'accès aux blocs ou aux sources.....	10-4
10.1.5	Instructions dans la vue d'ensemble des éléments de programme.....	10-5
10.2	Edition de la déclaration des variables .....	10-6
10.2.1	Utilisation de la déclaration des variables dans les blocs de code.....	10-6
10.2.2	Relation entre la vue de détail des variables et la section des instructions .....	10-7
10.2.3	Structure de la fenêtre de déclaration des variables .....	10-8
10.3	Multi-instances dans la déclaration des variables .....	10-9
10.3.1	Utilisation de multi-instances .....	10-9
10.3.2	Règles pour la formation de multi-instances .....	10-10
10.3.3	Saisie de multi-instances dans la fenêtre de déclaration des variables .....	10-10
10.4	Remarques générales sur la saisie d'instructions et de commentaires.....	10-11
10.4.1	Structure de la section des instructions .....	10-11
10.4.2	Marche à suivre pour la saisie d'instructions .....	10-12
10.4.3	Saisie de mnémoniques globaux dans un programme .....	10-13
10.4.4	Titres et commentaires de blocs et de réseaux.....	10-14
10.4.5	Saisie de commentaires de blocs et de réseaux .....	10-15
10.4.6	Utilisation de modèles de réseau.....	10-16
10.4.7	Fonction de recherche d'erreurs dans la section des instructions.....	10-17
10.5	Edition d'instructions CONT dans la section des instructions .....	10-18
10.5.1	Paramètres pour le langage de programmation CONT .....	10-18
10.5.2	Règles pour la saisie d'instructions CONT .....	10-18
10.5.3	Branchements interdits en CONT .....	10-21
10.6	Edition d'instructions LOG dans la section des instructions .....	10-22
10.6.1	Paramètres pour le langage de programmation LOG .....	10-22
10.6.2	Règles pour la saisie d'instructions LOG .....	10-23
10.7	Edition d'instructions LIST dans la section des instructions .....	10-25
10.7.1	Paramètres pour le langage de programmation LIST .....	10-25
10.7.2	Règles pour la saisie d'instructions LIST .....	10-25
10.8	Actualisation d'appels de blocs.....	10-26
10.8.1	Modifications d'interfaces .....	10-27
10.9	Enregistrement de blocs de code .....	10-28
<b>11</b>	<b>Création de blocs de données .....</b>	<b>11-1</b>
11.1	Principes de la création des blocs de données .....	11-1
11.2	Vue des déclarations de blocs de données.....	11-2
11.3	Vue des données de blocs de données .....	11-3
11.4	Saisie et enregistrement des blocs de données .....	11-4
11.4.1	Saisie de la structure de données de blocs de données globaux .....	11-4
11.4.2	Saisie / affichage de la structure de données de blocs de données associés à un FB (DB d'instance).....	11-5
11.4.3	Saisie de la structure de types de données utilisateur (UDT) .....	11-6
11.4.4	Saisie / affichage de la structure de blocs de données associés à un UDT.....	11-7
11.4.5	Modification de valeurs dans la vue des données.....	11-8
11.4.6	Réinitialisation de valeurs en leur substituant leur valeur initiale .....	11-8
11.4.7	Enregistrement de blocs de données .....	11-9
<b>12</b>	<b>Paramétrage de blocs de données.....</b>	<b>12-1</b>
12.1	Paramétrage de fonctions technologiques .....	12-2

<b>13</b>	<b>Création de sources LIST</b> .....	<b>13-1</b>
13.1	Principes de la programmation dans des sources LIST .....	13-1
13.2	Règles pour la programmation dans une source LIST .....	13-2
13.2.1	Règles pour la saisie d'instructions dans une source LIST .....	13-2
13.2.2	Règles pour la déclaration de variables dans une source LIST .....	13-3
13.2.3	Règles pour l'ordre des blocs dans une source LIST .....	13-4
13.2.4	Règles pour la définition d'attributs système dans une source LIST .....	13-4
13.2.5	Règles pour la définition de propriétés de bloc dans une source LIST .....	13-5
13.2.6	Propriétés de bloc autorisées pour chaque type de bloc .....	13-7
13.3	Structure des blocs dans une source LIST .....	13-8
13.3.1	Structure des blocs de code dans une source LIST .....	13-8
13.3.2	Structure des blocs de données dans une source LIST .....	13-9
13.3.3	Structure des types de données utilisateur dans une source LIST .....	13-9
13.4	Syntaxe et formats pour les blocs dans une source LIST .....	13-10
13.4.1	Tableau du format pour les OB .....	13-10
13.4.2	Tableau du format pour les FB .....	13-11
13.4.3	Tableau du format pour les FC .....	13-12
13.4.4	Tableau du format pour les DB .....	13-13
13.5	Création d'une source LIST .....	13-14
13.5.1	Création d'une source LIST .....	13-14
13.5.2	Edition d'une source S7 .....	13-14
13.5.3	Définition de la mise en page du texte source .....	13-15
13.5.4	Insertion de modèles de blocs dans une source LIST .....	13-15
13.5.5	Insertion du contenu d'autres source LIST .....	13-15
13.5.6	Insertion du code source de blocs existant dans une source LIST .....	13-16
13.5.7	Insertion d'une source externe .....	13-16
13.5.8	Génération d'une source LIST à partir de blocs .....	13-17
13.5.9	Importation d'une source .....	13-17
13.5.10	Exportation d'une source .....	13-18
13.6	Enregistrement, compilation et vérification d'une source LIST .....	13-19
13.6.1	Enregistrement d'une source LIST .....	13-19
13.6.2	Vérification de la cohérence d'une source LIST .....	13-19
13.6.3	Recherche d'erreurs dans une source LIST .....	13-19
13.6.4	Compilation d'une source LIST .....	13-20
13.7	Exemples de sources LIST .....	13-21
13.7.1	Exemples de déclarations de variables dans une source LIST .....	13-21
13.7.2	Exemple d'OB dans une source LIST .....	13-22
13.7.3	Exemple de FC dans une source LIST .....	13-23
13.7.4	Exemple de FB dans une source LIST .....	13-25
13.7.5	Exemples de DB dans une source LIST .....	13-27
13.7.6	Exemple d'UDT dans une source LIST .....	13-28
<b>14</b>	<b>Affichage des données de référence</b> .....	<b>14-1</b>
14.1	Présentation des données de référence possibles .....	14-1
14.1.1	Liste des références croisées .....	14-2
14.1.2	Structure du programme .....	14-3
14.1.3	Tableau d'affectation .....	14-5
14.1.4	Opérandes libres .....	14-7
14.1.5	Mnémoniques manquants .....	14-8
14.1.6	Affichage d'informations sur le bloc pour CONT, LOG, LIST .....	14-8

14.2	Utilisation des données de référence .....	14-9
14.2.1	Affichage des données de référence .....	14-9
14.2.2	Affichage de listes dans des fenêtres supplémentaires .....	14-9
14.2.3	Création et affichage de données de référence .....	14-10
14.2.4	Positionnement rapide sur les occurrences dans le programme .....	14-11
14.2.5	Exemple de recherche d'occurrences .....	14-12
<b>15</b>	<b>Vérifier la cohérence des blocs et horodatage comme propriété de bloc .....</b>	<b>15-1</b>
15.1	Vérifier la cohérence des blocs.....	15-1
15.2	Horodatage comme propriété de bloc et conflits d'horodatage .....	15-3
15.3	Horodatage dans les blocs de code .....	15-4
15.4	Horodatage dans les blocs de données globaux.....	15-5
15.5	Horodatage dans les blocs de données d'instance .....	15-5
15.6	Horodatage dans les UDT et DB repris d'UDT .....	15-6
15.7	Correction des interfaces dans une FC, un FB ou un UDT .....	15-6
15.8	Comment éviter des erreurs lors de l'appel de blocs .....	15-7
<b>16</b>	<b>Configuration de messages .....</b>	<b>16-1</b>
16.1	Concept de signalisation.....	16-1
16.1.1	Quels procédés de signalisation existe-t-il ? .....	16-1
16.1.2	Sélection du procédé de signalisation .....	16-3
16.1.3	Composants SIMATIC .....	16-5
16.1.4	Éléments constituant d'un message .....	16-5
16.1.5	Quels blocs de signalisation existe-t-il ?.....	16-6
16.1.6	Paramètres formels, attributs système et blocs de signalisation.....	16-8
16.1.7	Modèle de message et messages .....	16-9
16.1.8	Création d'une source LIST à partir de blocs de signalisation .....	16-10
16.1.9	Attribution de numéros de message .....	16-10
16.1.10	Différences entre l'attribution de numéros de message pour tout le projet et celle pour la CPU .....	16-11
16.1.11	Possibilités de modification de l'attribution des numéros de message d'un projet.	16-11
16.2	Configuration de messages pour tout le projet .....	16-12
16.2.1	Attribution de numéros de message pour tout le projet.....	16-12
16.2.2	Affectation et édition de messages sur bloc .....	16-12
16.2.2.1	Création de messages sur bloc (pour tout le projet).....	16-13
16.2.2.2	Édition de messages sur bloc (pour tout le projet) .....	16-16
16.2.2.3	Configuration des messages PCS7 (pour tout le projet) .....	16-16
16.2.3	Affectation et édition de messages sur mnémonique.....	16-18
16.2.3.1	Affectation et édition de messages sur mnémonique (pour tout le projet) .....	16-18
16.2.4	Création et édition de messages de diagnostic personnalisés .....	16-19
16.3	Configuration de messages pour la CPU .....	16-20
16.3.1	Attribution de numéros de message pour la CPU .....	16-20
16.3.2	Affectation et édition de messages sur bloc .....	16-20
16.3.2.1	Création de messages sur bloc (pour la CPU).....	16-20
16.3.2.2	Édition de messages sur bloc (pour la CPU).....	16-24
16.3.2.3	Configuration des messages PCS7 (pour la CPU).....	16-24
16.3.3	Affectation et édition de messages sur mnémonique.....	16-26
16.3.3.1	Affectation et édition de messages sur mnémonique (pour la CPU).....	16-26
16.3.4	Création et édition de messages de diagnostic personnalisés .....	16-27
16.4	Conseils pour l'utilisation de messages.....	16-28
16.4.1	Insertion d'une variable dans un message .....	16-28
16.4.2	Incorporer dans des messages des textes tirés de bibliothèques.....	16-30
16.4.3	Effacer des variables additionnelles .....	16-30
16.5	Traduction et édition de textes destinés à l'utilisateur .....	16-31
16.5.1	Traduction et édition de textes personnalisés .....	16-31

16.6	Edition et traduction de bibliothèques de textes .....	16-33
16.6.1	Bibliothèques de texte utilisateur .....	16-33
16.6.2	Création de bibliothèques de textes utilisateur .....	16-33
16.6.3	Edition de bibliothèques de textes utilisateur .....	16-34
16.6.4	Bibliothèques de textes système .....	16-34
16.6.5	Traduction de bibliothèques de textes .....	16-35
16.7	Transfert des données de configuration dans le système cible .....	16-37
16.8	Affichage des messages de CPU et des messages de diagnostic personnalisés .....	16-38
16.8.1	Configuration des messages de CPU.....	16-41
16.8.2	Affichage des messages de CPU enregistrés .....	16-41
16.9	Configuration de la signalisation d'erreurs système .....	16-42
16.9.1	Composants pris en charge et fonctionnalités.....	16-44
16.9.2	Paramétrage de la signalisation d'erreurs système.....	16-47
16.9.3	Génération de blocs pour la signalisation d'erreurs système .....	16-48
16.9.4	FB, DB générés .....	16-48
16.9.5	Création de textes de message dans une autre langue dans la 'Signalisation d'erreurs système' .....	16-50
<b>17</b>	<b>Contrôle-commande de variables .....</b>	<b>17-1</b>
17.1	Configuration de variables pour le contrôle-commande .....	17-1
17.2	Configuration d'attributs de contrôle-commande avec LIST, CONT, LOG.....	17-3
17.3	Configuration des attributs de contrôle-commande au moyen de la table des mnémoniques .....	17-4
17.4	Modification des attributs de contrôle-commande avec CFC .....	17-5
17.5	Transfert des données de configuration dans le système cible de contrôle-commande .....	17-6
<b>18</b>	<b>Etablissement d'une liaison en ligne et choix de la CPU.....</b>	<b>18-1</b>
18.1	Etablissement de liaisons en ligne.....	18-1
18.1.1	Etablissement d'une liaison en ligne depuis la fenêtre "Partenaires accessibles" ...	18-2
18.1.2	Etablissement d'une liaison en ligne depuis la fenêtre en ligne du projet .....	18-3
18.1.3	Accès en ligne aux systèmes cibles dans le multiprojet.....	18-4
18.1.4	Protection par mot de passe contre l'accès aux systèmes cible .....	18-6
18.1.5	Remarque sur l'actualisation du contenu de la fenêtre .....	18-7
18.2	Affichage et modification de l'état de fonctionnement .....	18-8
18.3	Affichage et réglage de l'heure et de la date .....	18-9
18.3.1	Horloges CPU avec réglage des zones horaires et heure d'été/heure d'hiver.....	18-9
18.4	Mise à jour du microprogramme .....	18-10
18.4.1	Mise à jour en ligne du microprogramme de modules.....	18-10
<b>19</b>	<b>Chargement .....</b>	<b>19-1</b>
19.1	Chargement dans le système cible depuis la PG.....	19-1
19.1.1	Conditions préalables au chargement .....	19-1
19.1.2	Différence entre l'enregistrement et le chargement de blocs .....	19-2
19.1.3	Mémoire de chargement et mémoire de travail dans la CPU .....	19-3
19.1.4	Possibilités de chargement selon la mémoire de chargement .....	19-4
19.1.5	Chargement du programme dans la CPU S7 .....	19-5
19.1.5.1	Chargement dans la gestion du projet.....	19-5
19.1.5.2	Chargement hors gestion du projet .....	19-6
19.1.5.3	Chargement de blocs dans le système cible .....	19-6
19.1.5.4	Enregistrement de blocs chargés dans la mémoire intégrée EPROM .....	19-7
19.1.5.5	Chargement via des cartes mémoire EPROM .....	19-7

19.2	Compilation et chargement de plusieurs objets depuis la PG .....	19-9
19.2.1	Conditions et remarques pour le chargement .....	19-9
19.2.2	Compilation et chargement d'objets .....	19-11
19.3	Chargement depuis le système cible dans la PG .....	19-13
19.3.1	Chargement d'une station dans la PG .....	19-14
19.3.2	Chargement de blocs depuis la CPU S7 .....	19-15
19.3.3	Edition de blocs chargés dans votre PG/PC .....	19-16
19.3.3.1	Edition de blocs chargés lorsque le programme utilisateur se trouve dans votre PG/PC .....	19-17
19.3.3.2	Edition de blocs chargés lorsque le programme utilisateur ne se trouve pas dans votre PG/PC .....	19-17
19.4	Effacement sur le système cible .....	19-18
19.4.1	Effacement de la mémoire de chargement/travail et effacement général de la CPU.....	19-18
19.4.2	Effacement de blocs S7 sur le système cible .....	19-19
19.5	Compression de la mémoire utilisateur (RAM).....	19-20
19.5.1	Intervalles dans la mémoire utilisateur (RAM).....	19-20
19.5.2	Compression du contenu de la mémoire d'une CPU S7 .....	19-21
<b>20</b>	<b>Test avec des tables de variables .....</b>	<b>20-1</b>
20.1	Introduction au test avec des tables de variables.....	20-1
20.2	Marche à suivre pour la visualisation et le forçage avec des tables de variables....	20-2
20.3	Edition et enregistrement de tables de variables.....	20-3
20.3.1	Création et ouverture d'une table de variables .....	20-3
20.3.2	Copie ou déplacement de tables de variables.....	20-3
20.3.3	Enregistrement d'une table de variables .....	20-4
20.4	Saisie de variables dans des tables de variables.....	20-4
20.4.1	Insertion d'opérandes ou de mnémoniques dans une table de variables .....	20-4
20.4.2	Insertion d'une plage d'opérandes continue dans une table de variables.....	20-6
20.4.3	Insertion de valeurs de forçage .....	20-6
20.4.4	Limites supérieures pour la saisie de temporisations.....	20-7
20.4.5	Limites supérieures pour la saisie de compteurs .....	20-8
20.4.6	Insertion de lignes de commentaire.....	20-8
20.4.7	Exemples .....	20-9
20.4.7.1	Exemple de saisie d'opérandes dans une table de variables.....	20-9
20.4.7.3	Exemples de saisie de valeurs de forçage/forçage permanent.....	20-11
20.5	Etablissement d'une liaison à la CPU.....	20-13
20.6	Visualisation de variables .....	20-14
20.6.1	Introduction à la visualisation de variables .....	20-14
20.6.2	Définition du déclenchement pour la visualisation de variables .....	20-14
20.7	Forçage de variables .....	20-16
20.7.1	Introduction au forçage de variables.....	20-16
20.7.2	Définition du déclenchement pour le forçage de variables.....	20-17
20.8	Forçage permanent de variables .....	20-19
20.8.1	Mesures de sécurité pour le forçage permanent de variables .....	20-19
20.8.2	Introduction au forçage permanent de variables .....	20-20
20.8.3	Différences entre forçage de variables et forçage permanent de variables .....	20-22

<b>21</b>	<b>Test avec la visualisation d'état du programme .....</b>	<b>21-1</b>
21.1	Affichage dans la visualisation d'état de programme .....	21-3
21.2	Informations sur le test en mode pas à pas et sur les points d'arrêt .....	21-4
21.3	Informations sur l'état de fonctionnement "Attente".....	21-6
21.4	Etat du programme de blocs de données.....	21-7
21.5	Définition de l'affichage de l'état du programme.....	21-8
21.6	Définition du mode de fonctionnement pour le test .....	21-9
<b>22</b>	<b>Test avec le programme de simulation (logiciel optionnel).....</b>	<b>22-1</b>
22.1	Test avec le programme de simulation S7-PLCSIM (logiciel optionnel) .....	22-1
<b>23</b>	<b>Diagnostic .....</b>	<b>23-1</b>
23.1	Diagnostic du matériel et recherche d'erreurs .....	23-1
23.2	Icônes de diagnostic dans la vue en ligne .....	23-3
23.3	Diagnostic du matériel : vue rapide .....	23-5
23.3.1	Appel de la vue rapide .....	23-5
23.3.2	Fonctions d'information de la vue rapide .....	23-5
23.4	Diagnostic du matériel : vue du diagnostic .....	23-6
23.4.1	Appel de la vue de diagnostic de HW Config .....	23-6
23.4.2	Fonctions d'information de la vue du diagnostic.....	23-8
23.5	Etat du module.....	23-9
23.5.1	Possibilités d'appel de l'état du module .....	23-9
23.5.2	Fonctions d'information de l'état du module .....	23-10
23.5.3	Volume d'informations selon le type de module dans l'état du module.....	23-12
23.5.4	Affichage de l'état du module d'appareils de terrain PA et d'esclaves DP placés derrière un Y-Link.....	23-14
23.6	Diagnostic à l'état de fonctionnement STOP .....	23-16
23.6.1	Marche à suivre pour déterminer la cause d'un passage à l'état d'arrêt .....	23-16
23.6.2	Contenu des piles à l'état d'arrêt .....	23-16
23.7	Contrôle des temps de cycle pour éviter les erreurs d'horloge .....	23-18
23.8	Transmission d'informations de diagnostic.....	23-19
23.8.1	Liste d'état système (SZL) .....	23-20
23.8.2	Envoi de vos propres messages de diagnostic .....	23-22
23.8.3	Fonctions de diagnostic .....	23-23
23.9	Mesures à prendre dans le programme pour traiter les erreurs.....	23-24
23.9.1	Exploitation du paramètre de sortie RET_VAL.....	23-25
23.9.2	OB d'erreur en réaction à la détection d'une erreur.....	23-26
23.9.3	Insertion de valeurs de remplacement en cas d'erreur détectée.....	23-31
23.9.4	Erreur de redondance de périphérie (OB70).....	23-33
23.9.5	Erreur de redondance de CPU (OB72).....	23-34
23.9.6	Erreur de temps (OB80) .....	23-35
23.9.7	Erreur d'alimentation (OB81) .....	23-36
23.9.8	Alarme de diagnostic (OB82).....	23-37
23.9.9	Alarme de débrogage/enfichage (OB83).....	23-38
23.9.10	Erreur matérielle CPU (OB84) .....	23-39
23.9.11	Erreur d'exécution du programme (OB85) .....	23-40
23.9.12	Défaillance d'unité (OB86).....	23-41
23.9.13	Erreur de communication (OB87).....	23-42
23.9.14	Erreur de programmation (OB121).....	23-43
23.9.15	Erreur d'accès à la périphérie (OB122) .....	23-44

<b>24</b>	<b>Impression et archivage</b> .....	<b>24-1</b>
24.1	Impression de la documentation du projet.....	24-1
24.1.1	Marche à suivre pour l'impression .....	24-2
24.1.2	Fonctions d'impression .....	24-2
24.1.3	Particularités pour l'impression de l'arborescence des objets .....	24-3
24.2	Archivage de projets et de bibliothèques.....	24-4
24.2.1	Possibilités d'enregistrement / archivage .....	24-5
24.2.2	Conditions requises pour l'archivage.....	24-6
24.2.3	Marche à suivre pour l'archivage/le désarchivage.....	24-6
<b>25</b>	<b>Utilisation des systèmes d'automatisation M7</b> .....	<b>25-1</b>
25.1	Marche à suivre pour les systèmes M7 .....	25-1
25.2	Logiciel optionnel pour la programmation M7.....	25-3
25.3	Systèmes d'exploitation pour M7-300/400.....	25-6
<b>26</b>	<b>Astuces et conseils</b> .....	<b>26-1</b>
26.1	Remplacement de modules dans la table de configuration.....	26-1
26.2	Projets comportant un grand nombre de stations en réseau .....	26-1
26.3	Réorganisation.....	26-2
26.4	Edition de mnémoniques dans plusieurs réseaux .....	26-2
26.5	Test à l'aide de la table des variables.....	26-3
26.6	Forçage de variables depuis l'éditeur de programmes.....	26-4
26.7	Mémoire virtuelle.....	26-5
<b>A</b>	<b>Annexe</b> .....	<b>A-1</b>
A.1	Etats de fonctionnement .....	A-1
A.1.1	Etats de fonctionnement et changement d'état de fonctionnement.....	A-1
A.1.2	Etat de fonctionnement "Arrêt" (STOP) .....	A-4
A.1.3	Mode de mise en route .....	A-5
A.1.4	Etat de fonctionnement "Marche" (RUN).....	A-13
A.1.5	Etat de fonctionnement "Attente" .....	A-14
A.2	Zones de mémoire des CPU S7 .....	A-15
A.2.1	Organisation des zones de mémoire .....	A-15
A.2.2	Mémoire de chargement et mémoire de travail .....	A-16
A.2.3	Mémoire système.....	A-19
A.2.3.1	Utilisation des zones de mémoire système .....	A-19
A.2.3.2	Mémoire image des entrées/sorties.....	A-21
A.2.3.3	Pile des données locales .....	A-25
A.2.3.4	Pile des interruptions .....	A-26
A.2.3.5	Pile des blocs .....	A-27
A.2.3.6	Mémoire tampon de diagnostic.....	A-28
A.2.3.7	Exploitation de la mémoire tampon de diagnostic .....	A-28
A.2.3.8	Zones de mémoire rémanentes des CPU S7-300.....	A-30
A.2.3.9	Zones de mémoire rémanentes des CPU S7-400.....	A-31
A.2.3.10	Objets mémoire configurables dans la mémoire de travail.....	A-32
A.3	Types de données et de paramètre.....	A-33
A.3.1	Introduction aux types de données et de paramètre .....	A-33
A.3.2	Types de données simples .....	A-34
A.3.2.1	Format du type de données INT (entiers de 16 bits) .....	A-35
A.3.2.2	Format du type de données DINT (nombres entiers de 32 bits) .....	A-36
A.3.2.3	Format du type de données REAL (nombres à virgule flottante) .....	A-37
A.3.2.4	Format des types de données WORD et DWORD pour les nombres décimaux codés binaire .....	A-41
A.3.2.5	Format du type de données S5TIME (durée SIMATIC) .....	A-42



A.3.3	Types de données complexes .....	A-43
A.3.3.1	Format du type de données DATE_AND_TIME (date et heure) .....	A-44
A.3.3.2	Utilisation de types de données complexes .....	A-46
A.3.3.3	Utilisation de tableaux pour l'accès aux données.....	A-47
A.3.3.4	Utilisation de structures pour l'accès aux données.....	A-50
A.3.3.5	Utilisation de types de données utilisateur pour l'accès aux données .....	A-52
A.3.4	Types de paramètre.....	A-54
A.3.4.1	Format des types de paramètre BLOCK, COUNTER et TIMER .....	A-55
A.3.4.2	Format du type de données Paramètre POINTER.....	A-55
A.3.4.3	Utilisation du type de données Paramètre POINTER.....	A-56
A.3.4.4	Bloc pour modifier le pointeur .....	A-57
A.3.4.5	Format du type de données Paramètre ANY .....	A-60
A.3.4.6	Utilisation du type de données Paramètre ANY .....	A-63
A.3.4.7	Affectation de types de données aux données locales de blocs de code .....	A-66
A.3.4.8	Types de données autorisés pour la transmission de paramètres .....	A-68
A.3.4.9	Transmission au paramètre IN_OUT d'un FB .....	A-73
A.4	Utilisation d'anciens projets .....	A-74
A.4.1	Conversion d'un ancien projet de version 1.....	A-74
A.4.2	Conversion d'un ancien projet de version 2.....	A-75
A.4.3	Remarque sur les projets STEP 7 de version V2.1 avec communication par données globales .....	A-76
A.4.4	Esclaves DP avec fichiers GSD manquants ou erronés .....	A-76
A.5	Exemples de programmes.....	A-77
A.5.1	Exemples de projets et de programmes.....	A-77
A.5.2	Exemple de programme pour un processus de mélange industriel .....	A-79
A.5.2.1	Définition de blocs de code.....	A-82
A.5.2.2	Affectation de mnémoniques .....	A-83
A.5.2.3	Création du bloc fonctionnel pour le moteur.....	A-86
A.5.2.4	Création de la fonction pour les soupapes .....	A-90
A.5.2.5	Création de l'OB1.....	A-92
A.5.3	Exemple d'utilisation d'alarmes horaires.....	A-99
A.5.3.1	Structure de l'alarme horaire du programme utilisateur .....	A-99
A.5.3.2	FC12 .....	A-101
A.5.3.3	OB10.....	A-103
A.5.3.4	OB1 et OB80.....	A-106
A.5.4	Exemple d'utilisation d'alarmes temporisées.....	A-108
A.5.4.1	Structure de l'alarme temporisée du programme utilisateur .....	A-108
A.5.4.2	OB20.....	A-110
A.5.4.3	OB1 .....	A-112
A.5.4.4	Exemple de masquage et de démasquage d'événements d'erreurs synchrones..	A-115
A.5.4.5	Exemple d'inhibition et de validation d'événements d'alarme et d'événements asynchrones (SFC 39 et 40).....	A-120
A.5.4.6	Exemple de traitement différé d'événements d'alarme et d'événements asynchrones (SFC 41 et 42) .....	A-121
A.6	Accès aux zones de données du processus et de la périphérie .....	A-122
A.6.1	Accès à la zone de données du processus .....	A-122
A.6.2	Accès à la zone de données de périphérie .....	A-124
A.7	Définition du comportement en fonctionnement.....	A-126
A.7.1	Modification du comportement et des propriétés des modules .....	A-127
A.7.2	Mise à jour hors ligne du microprogramme (du système d'exploitation) de modules .....	A-129
A.7.3	Avantage des fonctions d'horodatage .....	A-130
A.7.4	Utilisation de mémentos de cadence et de temporisations .....	A-132
<b>Index</b>	.....	<b>Index-1</b>



# 1 Introduction sur le produit et installation

## 1.1 Guide de STEP 7

### Qu'est-ce que STEP 7 ?

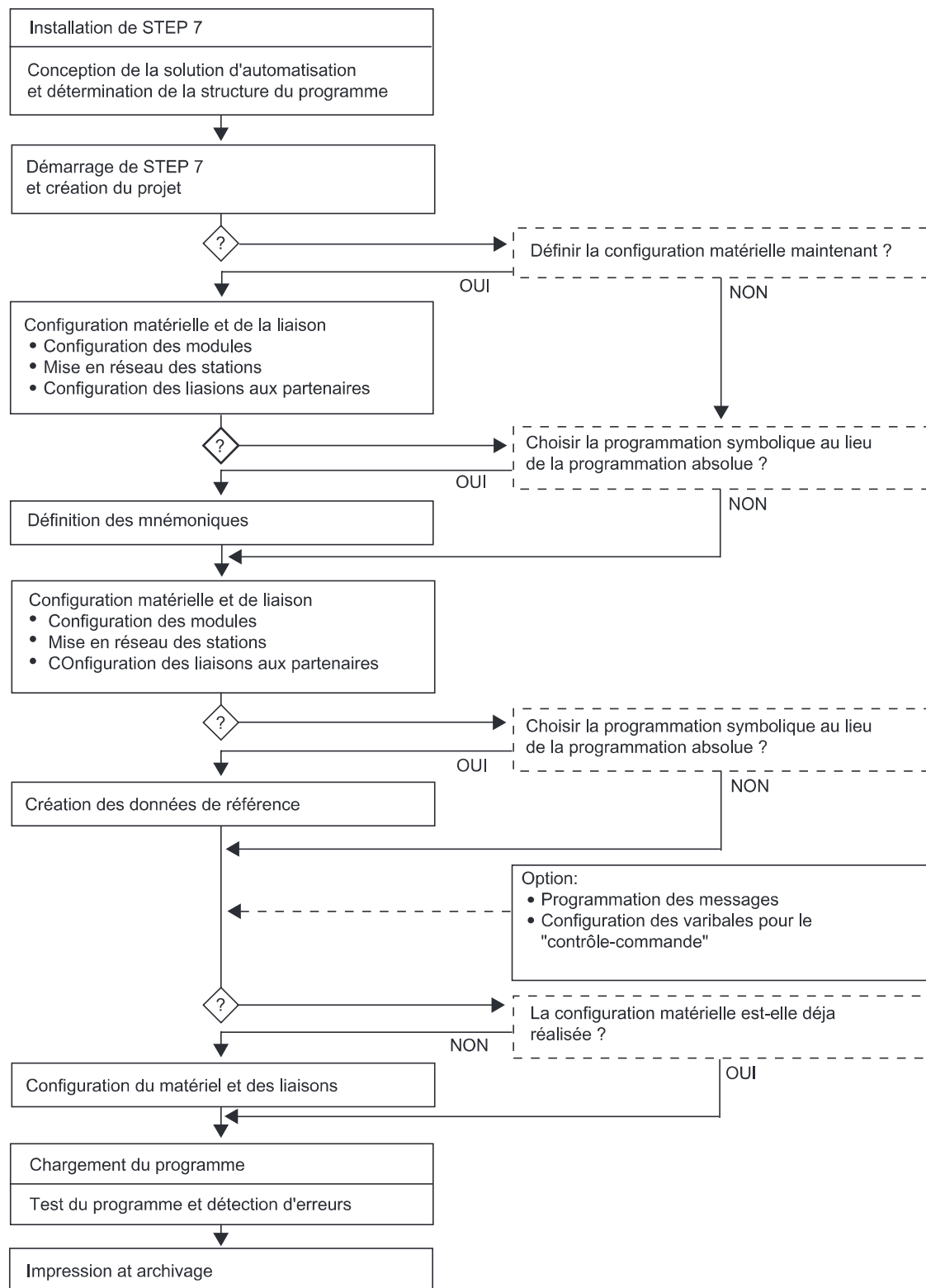
STEP 7 est le progiciel de base pour la configuration et la programmation de systèmes d'automatisation SIMATIC. Il fait partie de l'industrie logicielle SIMATIC. Le progiciel de base STEP 7 existe en plusieurs versions :

- STEP 7-Micro/DOS et STEP 7-Micro/Win pour des applications autonomes simples sur SIMATIC S7 - 200.
- STEP 7 pour des applications sur SIMATIC S7-300/400, SIMATIC M7-300/400 et SIMATIC C7 présentant des fonctionnalités supplémentaires :
  - Possibilité d'extension grâce aux applications proposées par l'industrie logicielle SIMATIC (voir aussi Possibilités d'extension du logiciel de base STEP 7)
  - Possibilité de paramétrage de modules fonctionnels et de modules de communication
  - Forçage et fonctionnement multiprocesseur
  - Communication par données globales
  - Transfert de données commandé par événement à l'aide de blocs de communication et de blocs fonctionnels
  - Configuration de liaisons

STEP 7 fait l'objet du présent manuel d'utilisation, STEP 7-Micro étant décrit dans la documentation "STEP 7-Micro/DOS".

### Tâches fondamentales

La mise en place d'une solution d'automatisation avec STEP 7 nécessite la réalisation de tâches fondamentales. La figure suivante indique les tâches à exécuter dans la plupart des projets et les classe selon la marche à suivre. Ce guide renvoie aux chapitres respectifs, vous permettant ainsi de vous déplacer dans le manuel selon la tâche que vous avez à réaliser.



## Solutions de rechange

Comme le montre la figure précédente, vous pouvez procéder de deux manières différentes :

- Vous pouvez configurer le matériel en premier lieu, puis programmer les blocs.
- Mais vous pouvez aussi programmer d'abord les blocs sans avoir à configurer auparavant le matériel. Ceci est particulièrement recommandé pour les tâches de maintenance. En effet, vous avez ainsi la possibilité d'intégrer des blocs programmés dans un projet existant.

## Brève description des diverses étapes :

- **Installation de STEP 7 et des "License Keys"**  
Pour une première utilisation, vous devez installer STEP 7 et transférer les "License Keys" depuis la disquette sur le disque dur (voir aussi Installation de STEP 7 et Autorisation).
- **Conception de la solution d'automatisation**  
Avant d'utiliser STEP 7, vous devez planifier votre solution d'automatisation depuis la division du processus en tâches individuelles jusqu'à la réalisation d'un schéma de configuration (voir aussi Conception d'une solution d'automatisation).
- **Conception de la structure du programme**  
En utilisant les blocs mis à votre disposition par STEP 7, vous transposez les tâches décrites lors de la conception de votre solution d'automatisation en structure de programme (voir aussi Blocs dans le programme utilisateur).
- **Démarrage de STEP 7**  
Vous démarrez STEP 7 depuis l'interface utilisateur de Windows (voir aussi Démarrage de STEP 7).
- **Définition de la structure du projet**  
Un projet peut être comparé à un dossier dans lequel toutes les données sont organisées de manière hiérarchique et sont toujours disponibles. Dès lors que vous avez créé un projet, toutes les tâches suivantes y seront exécutées (voir aussi Structure du projet).
- **Création de la station**  
En créant la station, vous définissez l'automate programmable : p.ex. SIMATIC 300, SIMATIC 400, SIMATIC S5 (voir aussi Insertion de stations).
- **Configuration matérielle**  
Dans une table de configuration, vous définissez les modules que vous allez mettre en oeuvre dans votre solution d'automatisation ainsi que les adresses permettant d'y accéder depuis le programme utilisateur. Vous pouvez en outre y paramétrer les caractéristiques des modules (voir aussi Manipulations de base pour la configuration matérielle).
- **Configuration de réseaux et de liaisons de communication**  
La condition requise à l'établissement d'une communication est l'existence d'un réseau préalablement configuré. Vous devez à cet effet créer les réseaux auxiliaires nécessaires à votre solution d'automatisation, définir leurs propriétés et pour les stations mises en réseau, les caractéristiques de connexion au réseau ainsi que, le cas échéant, les liaisons de communication requises (voir aussi Marche à suivre pour la configuration d'un sous-réseau).
- **Définition de mnémoniques**  
Dans une table des mnémoniques, vous pouvez remplacer des adresses par des mnémoniques locaux ou globaux de désignation plus évocatrice afin de les utiliser dans votre programme (voir aussi Création d'une table des mnémoniques)

- **Création du programme**  
En utilisant l'un des langages de programmation mis à votre disposition, vous créez un programme affecté ou non à un module, que vous enregistrez sous forme de blocs, de sources ou de diagrammes (voir aussi Marche à suivre pour la création de blocs de code et Principes de la programmation dans les sources LIST).
- **S7 uniquement : création et exploitation de données de référence**  
Vous pouvez utiliser ces données de référence afin de vous faciliter le test et la modification de votre programme utilisateur (voir aussi Affichage des données de référence existantes).
- **Configuration de messages**  
Créez par exemple des messages sur bloc avec leurs textes et attributs. En utilisant le programme de transfert, vous transférez ensuite les données de configuration de messages dans la base de données du système de contrôle-commande (p.ex. SIMATIC WinCC, SIMATIC ProTool) (voir aussi Configuration de messages).
- **Configuration de variables de contrôle-commande**  
Vous définissez une fois pour toutes les variables de contrôle-commande dans STEP 7 et leur affectez les attributs souhaités. En utilisant le programme de transfert, vous transférez les variables de contrôle-commande créées dans la base de données du système de contrôle-commande WinCC (voir aussi Configuration de variables de contrôle-commande).
- **Chargement de programmes dans le système cible**  
S7 uniquement : une fois la configuration, le paramétrage et la création du programme terminés, vous pouvez transférer votre programme utilisateur complet ou des blocs individuels dans le système cible (module programmable de votre solution matérielle). La CPU contient déjà le système d'exploitation (voir aussi Conditions préalables au chargement).  
M7 uniquement : parmi différents systèmes d'exploitation, vous sélectionnez celui qui s'adapte à votre solution d'automatisation et le transférez seul ou avec le programme utilisateur sur le support de données souhaité du système cible M7.
- **Test de programmes**  
S7 uniquement : pour effectuer un test, vous avez la possibilité d'afficher les valeurs de variables depuis votre programme utilisateur ou depuis une CPU, d'affecter des valeurs à ces variables et de créer une table des variables que vous souhaitez afficher ou forcer (voir aussi Introduction au test avec des tables de variables).  
M7 uniquement : test du programme utilisateur à l'aide d'un programme de débogage en langage évolué.
- **Surveillance du fonctionnement, diagnostic du matériel**  
Vous déterminez les causes du défaut d'un module en affichant des informations en ligne relatives à ce module. Vous déterminez les causes d'un défaut dans le déroulement d'un programme utilisateur à l'aide de la mémoire tampon de diagnostic et du contenu des piles. Vous pouvez en outre vérifier si un programme utilisateur est exécutable sur une CPU donnée (voir aussi Diagnostic du matériel et affichage de l'état du module).
- **Documentation de l'installation**  
Après avoir créé un projet ou une installation, il est conseillé de documenter les données de configuration de manière claire afin de faciliter le traitement ultérieur du projet de même que les tâches de maintenance (voir aussi Impression de la documentation du projet).  
DOCPRO, l'application optionnelle de création et de gestion de documentation d'installations permet la structuration des données de configuration, la présentation sous forme de dossiers des schémas de l'installation et l'impression dans une présentation homogène.

## **Extension du manuel avec des thèmes particuliers**

Différents thèmes spéciaux peuvent représenter un intérêt pour vous lors de la réalisation d'une solution d'automatisation :

- Fonctionnement multiprocesseur - synchrone de plusieurs CPU (voir aussi Mode multiprocesseur - fonctionnement synchrone de plusieurs CPU)
- Travail de plusieurs personnes sur un même projet (voir aussi Edition de projets par plusieurs personnes)
- Utilisation de systèmes M7 (voir aussi Marche à suivre pour les systèmes M7)

## 1.2 Logiciel de base STEP 7

### Normes en vigueur

Les langages de programmation SIMATIC intégrés à STEP 7 répondent à la norme DIN EN 6.1131-3.

Le progiciel de base s'exécute sous les systèmes d'exploitation MS Windows 2000 Professional (que par la suite nous appellerons Windows 2000) ainsi que MS Windows XP Professional (que par la suite nous appellerons Windows XP) ainsi que MS Windows Server 2003 et s'adapte à son organisation graphique orientée objet.

### Fonctions du logiciel de base

Le logiciel de base vous assiste dans toutes les phases du processus de création de vos solutions d'automatisation, comme par exemple :

- la création et la gestion de projets,
- la configuration et le paramétrage du matériel et de la communication,
- la gestion des mnémoniques,
- la création de programmes, par exemple pour les systèmes cible S7,
- le chargement de programmes dans des systèmes cible,
- le test de l'installation d'automatisation,
- le diagnostic lors de perturbations de l'installation.

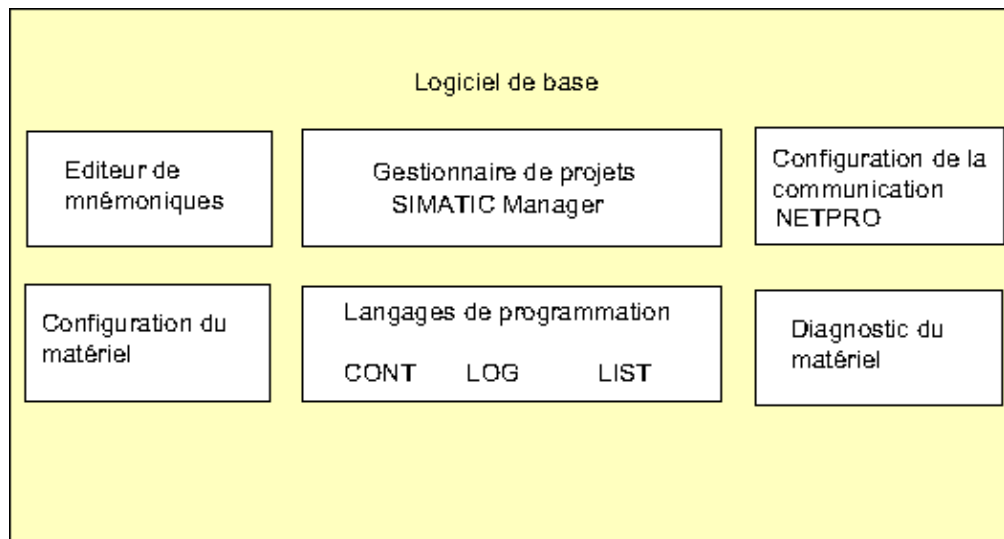
La conception de l'interface utilisateur du logiciel STEP 7 répond aux connaissances ergonomiques modernes et son apprentissage est très facile.

La documentation du logiciel STEP 7 met à votre disposition tous les informations nécessaires en ligne, dans l'aide en ligne et dans des manuels électroniques de format PDF.



## Applications disponibles

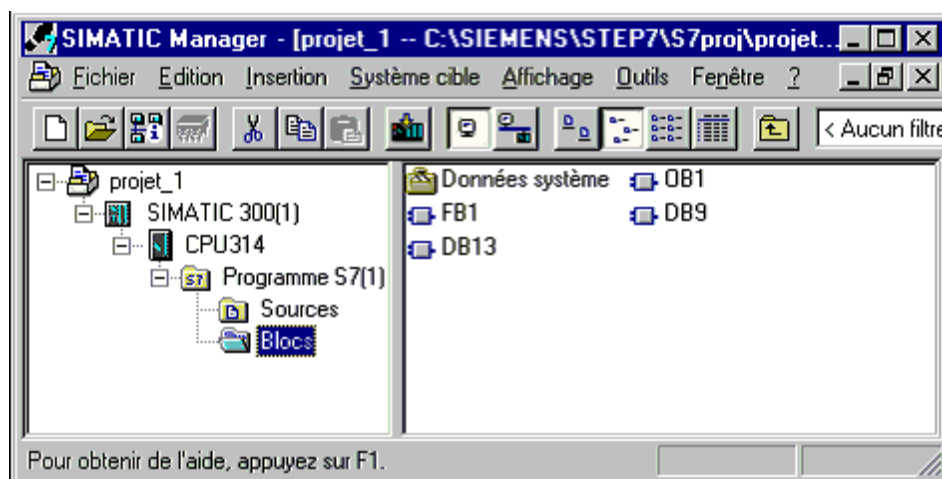
Le logiciel de base STEP 7 met à votre disposition différentes applications :



Il n'est pas nécessaire d'appeler séparément chaque application, car elles sont démarrées automatiquement lorsque vous sélectionnez une fonction correspondante ou ouvrez un objet.

## Gestionnaire de projets SIMATIC

Le gestionnaire de projets SIMATIC gère toutes les données relatives à un projet d'automatisation – quel que soit le système cible (S7/M7/C7) sur lequel elles ont été créées. Le gestionnaire de projets SIMATIC démarre automatiquement les applications requises pour le traitement des données sélectionnées.



## Editeur de mnémoniques

L'éditeur de mnémoniques vous permet de gérer toutes les variables globales. Vous disposez des fonctions suivantes :

- définition de désignations symboliques et de commentaires pour les signaux du processus (entrées/sorties), mémentos et blocs,
- fonctions de tri,
- importation/exportation avec d'autres programmes Windows.

La table des mnémoniques qui en résulte est mise à disposition de toutes les applications. La modification de l'un des paramètres d'un mnémonique est de ce fait reconnue automatiquement par toutes les applications.

## Diagnostic du matériel

Le diagnostic du matériel fournit un aperçu de l'état du système d'automatisation. Dans une représentation d'ensemble, un symbole permet de préciser pour chaque module, s'il est défaillant ou pas. Un double clic sur le module défaillant permet d'afficher des informations détaillées sur le défaut. Les informations disponibles dépendent des différents modules :

- affichage d'informations générales sur le module (p.ex. numéro de commande, version, désignation) et son état (p.ex. défaillant),
- affichage d'erreurs sur les modules (p.ex. erreur de voie) de la périphérie centrale et des esclaves DP,
- affichage des messages de la mémoire tampon de diagnostic.

Pour les CPU, des informations supplémentaires s'affichent :

- causes de défaillance dans le déroulement d'un programme utilisateur
- durée de cycle (le plus long, le plus court et dernier),
- possibilités et charge de la communication MPI,
- performances (nombre d'entrées/sorties, de mémentos, de compteurs, de temporisations et de blocs possibles).

## Langages de programmation

Les langages de programmation CONT, LIST et LOG pour S7-300/400 font partie intégrante du logiciel de base.

- Le schéma à contacts (CONT) est un langage de programmation graphique. La syntaxe des instructions fait penser aux schémas de circuits. CONT permet de suivre facilement le trajet du courant entre les barres d'alimentation en passant par les contacts, les éléments complexes et les bobines.
- La liste d'instructions (LIST) est un langage de programmation textuel proche de la machine. Dans un programme LIST, les différentes instructions correspondent, dans une large mesure, aux étapes par lesquelles la CPU traite le programme. Pour faciliter la programmation, LIST a été complété par quelques structures de langage évolué (comme, par exemple, des paramètres de blocs et accès structurés aux données).
- Le logigramme (LOG) est un langage de programmation graphique qui utilise les boîtes de l'algèbre de Boole pour représenter les opérations logiques. Les fonctions complexes, comme par exemple les fonctions mathématiques, peuvent être représentées directement combinées avec les boîtes logiques.

Vous pouvez vous procurer d'autres langages de programmation sous forme de logiciels optionnels.

## Configuration matérielle

Vous utilisez cette application pour configurer et paramétrer le matériel d'un projet d'automatisation. Vous disposez des fonctions suivantes :

- Pour configurer le système d'automatisation, vous sélectionnez des châssis (Racks) dans un catalogue électronique et affectez les modules sélectionnés aux emplacements souhaités dans les racks.
- La configuration de la périphérie décentralisée est identique à celle de la périphérie centralisée. La périphérie voie par voie est également possible.
- Pour le paramétrage de la CPU, des menus vous permettent de définir des caractéristiques telles que le comportement à la mise en route et la surveillance du temps de cycle. Le fonctionnement multiprocesseur est possible. Les données saisies sont enregistrées dans des blocs de données système.
- Pour le paramétrage des modules, des boîtes de dialogue vous permettent de définir tous les paramètres modifiables. Les réglages à l'aide de commutateurs DIP s'avèrent inutiles. Le paramétrage des modules est réalisé automatiquement au démarrage de la CPU. L'avantage suivant en résulte. Le remplacement d'un module est ainsi possible sans nouveau paramétrage.
- Le paramétrage de modules fonctionnels (FM) et de processeurs de communication (CP) s'effectue de manière identique à celui des autres modules dans la configuration matérielle. A cet effet, des boîtes de dialogues ainsi que des règles spécifiques aux modules sont ainsi mises à disposition pour chaque FM et CP (fournies dans le logiciel fonctionnel du FM/CP). Dans les boîtes de dialogue, le système ne propose que des saisies possibles, ce qui empêche les entrées erronées.

## NetPro

NetPro permet un transfert de données cyclique déclenché par temporisation via MPI avec :

- choix des participants à la communication,
- saisie de la source et de la destination des données dans un tableau ; la génération de tous les blocs à charger (SDB) et leur transfert complet dans toutes les CPU s'effectuent automatiquement.

En outre, un transfert de données déclenché par événement est possible avec :

- la définition des liaisons de communication,
- le choix des blocs de communication/ blocs fonctionnels dans la bibliothèque des blocs intégrée,
- le paramétrage des blocs de communication/ blocs fonctionnels sélectionnés dans le langage de programmation habituel.

## 1.3 Nouveautés dans la version 5.4 de STEP 7

Les thèmes suivants ont été actualisés :

- SIMATIC Manager
- Configuration et diagnostic du matériel
- Configuration de réseaux et de liaisons
- Bibliothèques standard
- Signalisation d'erreurs système

### SIMATIC Manager

- A partir de STEP 7 V5.4, vous disposez de deux formats d'affichage pour la date et l'heure. Vous pouvez choisir entre l'affichage dans la langue respective de STEP 7 et la norme ISO 8601. Vous réalisez le paramétrage dans SIMATIC Manager, dans l'onglet "Date et heure" de la boîte de dialogue "Paramètres".
- A partir de STEP 7 V5.4, vous pouvez afficher les heures de modules dans l'heure locale de votre PG/PC. Vous réalisez le paramétrage dans SIMATIC Manager, dans l'onglet "Date et heure" de la boîte de dialogue "Paramètres".
- A partir de STEP 7 V5.4, vous avez la possibilité de définir une protection d'accès en attribuant un mot de passe aux projets ou aux bibliothèques. Ceci suppose cependant que SIMATIC Logon V1.3 SP1, que par la suite nous appellerons SIMATIC Logon, soit installé (voir Informations sur la protection d'accès).
- A partir de STEP 7 V5.4, vous avez la possibilité de tenir un journal des modifications après avoir défini une protection d'accès pour les projets et bibliothèques. Ce journal des modifications permet de consigner des actions en ligne, telles que le "Chargement", les "Modifications de l'état de fonctionnement" ou un "Effacement général". Ceci suppose cependant que SIMATIC Logon V1.3 SP1, que par la suite nous appellerons SIMATIC Logon, soit installé (voir Informations sur le journal des modifications).

### Configuration et diagnostic du matériel

- Le procédé "Information et maintenance" est pris en charge pour permettre la lecture ou l'écriture de données d'identification sur le module. Cette fonction est également disponible dans SIMATIC Manager (voir Identification et maintenance (I&M)).
- L'écriture de données d'identification est également possible pour les connexions PROFIBUS DP en mode redondant (via "Partenaires accessibles"). Le coupleur (IM) doit prendre en charge cette fonction.
- Il est possible d'importer et d'exporter des données CAX. Il est ainsi possible d'échanger de données communes entre STEP 7 et p. ex. des systèmes d'ingénierie CAD ou CAE (voir Exportation et importation de données CAX)
- Vous avez également la possibilité d'actualiser le firmware de coupleurs PROFIBUS DP (IM) en mode redondant, dans la mesure où les IM sont prévus à cet effet. Chacun des IM redondants mis en œuvre est en mesure de transmettre automatiquement le firmware actualisé à l'IM redondant mis en œuvre via le bus de fond de panier actif.
- Pour la fonction "Redondance logicielle", il est également possible de copier et d'insérer de manière redondante des PA-Link avec des esclaves PA sous-jacents (voir Configuration de la redondance logicielle).

- Les applications d'édition d'objets dans HW Config peuvent être démarrées avec la commande de menu **Edition > Ouvrir l'objet** (voir Ouverture d'objets dans HW Config).
- Pour les IO Device PROFINET, vous pouvez configurer un délai de réponse (voir Configuration du délai de réponse).
- A partir de STEP 7 V5.4, vous pouvez afficher l'heure des modules dans l'heure locale de votre PG/PC.

### Configuration de réseaux et de liaisons

- PROFINET IO avec la communication IRT (Isochronous Realtime) est pris en charge. Il est ainsi également possible de configurer des temps de cycle du bus courts et de même longueur pour PROFINET IO (voir Introduction : Isochronous Realtime Ethernet).
- Manipulation plus aisée lors de l'insertion d'IO Devices copiés dans une autre station. Si les adresses IP sont déjà attribuées, vous pouvez définir le comportement lors de l'insertion (conserver les adresses ou en faire attribuer de nouvelles).
- Comme pour les esclaves PROFIBUS DP, vous pouvez à présent paramétrer la surveillance de réponse pour les IO Devices PROFINET : comme propriété de l'objet de l'IO Device dans l'onglet "Cycle IO".
- Si vous utilisez des composants optiques pour PROFIBUS DP : dans les configurations avec un anneau optique, vous pouvez spécifier les OLM utilisés. Le calcul des paramètres du bus est ainsi plus précis et, si vous utilisez des composants plus performant, le temps de cycle du bus plus court.

### Bibliothèques standard

- La bibliothèque standard "Communication Blocks" a été étendue avec les blocs FB 67 et FB 68 pour la communication TCP/IP ouverte.
- La bibliothèque standard "Communication Blocks" a été étendue avec les blocs FB 20, FB 21, FB 22 et FB 23 pour l'accès cyclique aux données utiles conformément à PNO.
- A partir de STEP 7 V5.4, il existe outre la bibliothèque valables pour les modules Redundant IO (V1) une nouvelle bibliothèque de blocs Redundant IO CGP (channel granular peripheral devices). Elle prend en charge la redondance pour les différentes voies de modules. Vous trouverez des informations complémentaires dans l'aide contextuelle du bloc ou dans le fichier Lisezmoi de STEP 7. Vous trouverez une liste actuelle des modules pris en charge sous forme de FAQ sous <http://support.automation.siemens.com/>.

### Signalisation d'erreurs système

- A partir de V5.4, un bloc de données (DB 125) est pris en charge pour PROFIBUS, avec lequel il est possible d'éditer des événements de diagnostic de manière graphique sur un appareil IHM.

## 1.4 Possibilités d'extension du logiciel de base STEP 7

L'extension du logiciel de base peut être réalisée à l'aide de logiciels optionnels, regroupés dans les trois catégories de logiciels suivantes :

- Applications techniques  
elles comportent des langages de programmation évolués et des logiciels à orientation technologique.
- Logiciels exécutables  
ils englobent des logiciels exécutables directement utilisables dans le processus de production.
- Interfaces homme/machine (Human Machine Interfaces ; HMI)  
elles désignent des logiciels spécifiques au contrôle-commande.

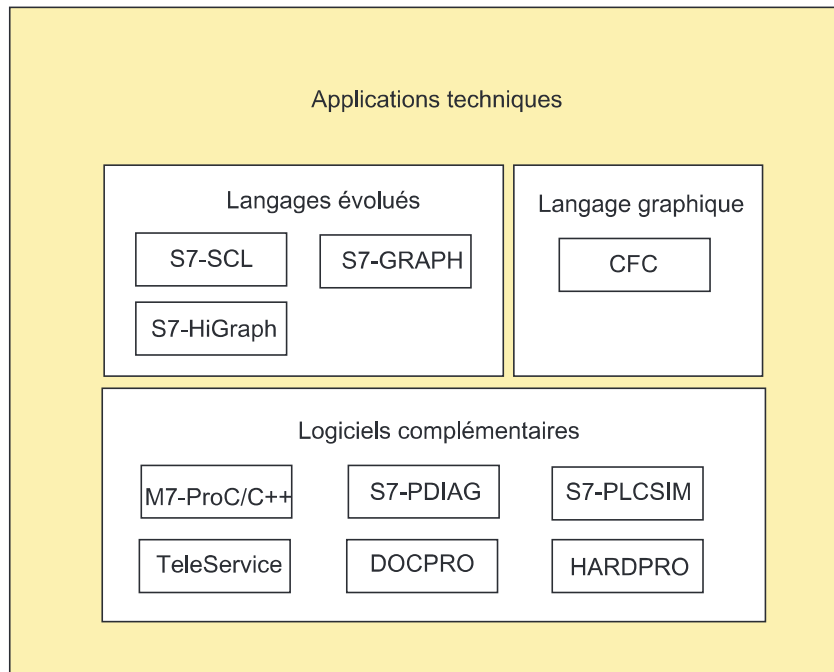
Le tableau suivant indique les logiciels optionnels pouvant être mis en oeuvre selon le système d'automatisation utilisé :

	STEP 7		
	S7-300 S7-400	M7-300 M7-400	C7-620
Applications techniques			
• Borland C/C++		o	
• CFC	+ <sup>1)</sup>	+	+ <sup>2)</sup>
• DOCPRO	+	+ <sup>3)</sup>	+
• HARDPRO	+		
• M7-ProC/C++		o	
• S7-GRAPH	+ <sup>1)</sup>		+ <sup>2)</sup>
• S7-HiGraph	+		+
• S7-PDIAG	+		
• S7-PLCSIM	+		+
• S7-SCL	+		+
• TeleService	+	+	+
Logiciels exécutables			
• Fuzzy Control	+		+
• M7-DDE-Server		+	
• M7-SYS RT		o	
• Modular PID Control	+		+
• PC-DDE-Server	+		
• PRODAVE MPI	+		
• Standard PID Control	+		+
Interface homme/machine			
• ProAgent			
• SIMATIC ProTool			
• SIMATIC ProTool/Lite			o
• SIMATIC WinCC			
o = indispensable + = optionnel <sup>1)</sup> = recommandé à partir de S7-400 <sup>2)</sup> = non recommandé pour C7-620 <sup>3)</sup> = pas pour les programmes C			

### 1.4.1 Applications techniques

Les applications techniques sont des applications orientées tâche pouvant être mises en oeuvre en tant qu'extension du logiciel de base. Elles englobent :

- les langages évolués pour le programmeur,
- le langage graphique pour l'ingénieur en technologie,
- des logiciels complémentaires pour le diagnostic, la simulation, la maintenance à distance, la documentation de l'installation, etc.



## Langages évolués

Vous disposez des logiciels de langage optionnels suivants pour la programmation des automates programmables SIMATIC S7-300/400.

- GRAPH est un langage de programmation permettant la description aisée de commandes séquentielles (programmation de graphes séquentiels). Le déroulement du processus y est subdivisé en étapes. Celles-ci contiennent en particulier des actions pour la commande des sorties. Le passage d'une étape à la suivante est soumis à des conditions de transition.
- HiGraph est un langage de programmation permettant la description aisée de processus asynchrones non séquentiels sous forme de graphes d'état. A cet effet, l'installation est subdivisée en unités fonctionnelles pouvant prendre différents états. Ces unités fonctionnelles peuvent se synchroniser par l'échange de messages.
- SCL est un langage évolué textuel conforme à la norme DIN EN 61131-3. Il comporte des éléments de langage que l'on trouve également sous une forme similaire dans les langages de programmation Pascal et C. SCL convient donc particulièrement aux utilisateurs déjà habitués à se servir d'un langage de programmation évolué. Vous pouvez, par exemple, faire appel à SCL pour programmer des fonctions très complexes ou se répétant souvent.

## Langage graphique

CFC pour S7 et M7 est un langage de programmation permettant l'interconnexion graphique de fonctions existantes. Ces fonctions couvrent un large éventail allant de combinaisons logiques simples à des régulations et commandes complexes. Un grand nombre de ces fonctions est disponible sous la forme de blocs dans une bibliothèque. La programmation se fait en copiant des blocs sur un diagramme et en reliant les connecteurs de blocs par des lignes.

## Logiciels complémentaires

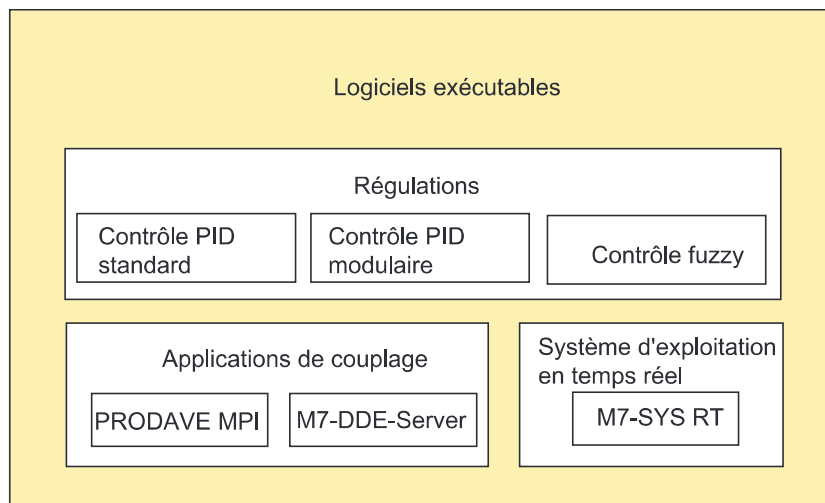
- Borland C++ (pour M7 uniquement) contient l'environnement de développement Borland.
- Avec DOCPRO, vous pouvez organiser toutes les données de configuration créées avec STEP 7 dans un dossier des schémas de l'installation. Ceci facilite la gestion des données de configuration et garantit la conformité aux normes lors de la préparation à l'impression.
- HARDPRO est le système de configuration matérielle pour S7-300 qui assiste l'utilisateur dans son énorme tâche de configuration de solutions d'automatisation complexes.
- M7-ProC/C++ (pour M7 uniquement) permet d'intégrer l'environnement de développement Borland pour les langages de programmation C et C++ à l'environnement de développement STEP 7.
- PLCSIM (pour S7 uniquement) permet de simuler des automates programmables S7 connectés à votre outil de développement (PG/PC) à des fins de test.
- PDIAG (pour S7 uniquement) permet la configuration homogène du diagnostic du processus pour SIMATIC S7-300/400. Le diagnostic du processus permet de détecter des états erronés hors du système d'automatisation (p.ex. position finale non atteinte).
- Téléservice offre la possibilité de programmer et d'effectuer la maintenance de systèmes d'automatisation S7 et M7 depuis la PG ou le PC via le réseau téléphonique.



## 1.4.2 Logiciels exécutable

Il s'agit de solutions logicielles finies programmées pouvant être appelées dans le programme utilisateur. Les logiciels exécutable sont directement intégrés dans la solution d'automatisation. Ils englobent :

- des régulations pour SIMATIC S7. Des exemples en sont les régulations standard, modulaire et Fuzzy,
- des applications de couplage des systèmes d'automatisation avec des applications Windows,
- un système de fonctionnement en temps réel pour SIMATIC M7.



### Régulations pour SIMATIC S7

- Le contrôle PID standard permet l'intégration de régulateurs à action continue, de régulateurs à impulsion et de régulateurs incrémentiels dans le programme utilisateur. L'application de paramétrage à laquelle la définition du régulateur est intégrée permet le paramétrage rapide et le réglage optimal du régulateur.
- Le contrôle PID modulaire est mis en oeuvre lorsqu'un régulateur PID simple ne permet pas la résolution de la tâche d'automatisation. La mise en circuit des blocs fonctionnels standard fournis permet de réaliser quasiment toutes les structures techniques de régulation.
- Le contrôle Fuzzy permet de créer des systèmes Fuzzy. Ces systèmes sont mis en oeuvre lorsque des processus ne peuvent pas ou peuvent difficilement être décrits mathématiquement, lorsque le déroulement de mécanismes et de processus est imprévisible, lorsque des comportements non linéaires surviennent alors que l'on dispose d'une connaissance acquise par expérience du processus.

### **Applications de couplage**

- PRODAVE MPI est une palette d'outils permettant l'échange de données du processus entre SIMATIC S7, SIMATIC M7 et SIMATIC C7. Elle réalise de manière autonome l'échange de données via l'interface MPI.
- Le serveur M7-DDE ( 19>Dynamic **D**ata **E**xchange) permet de relier des applications Windows à des variables du processus dans SIMATIC M7, sans qu'une programmation supplémentaire ne soit nécessaire.

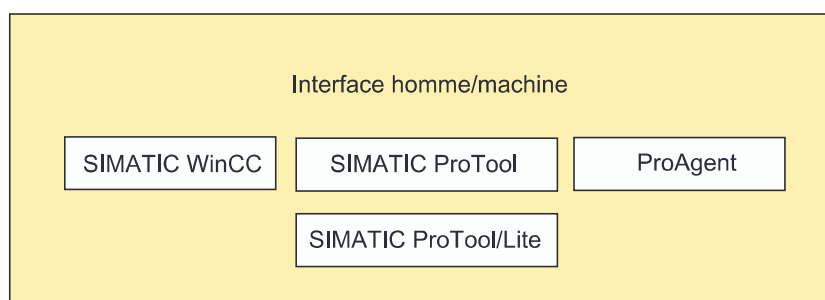
### **Systeme d'exploitation en temps réel**

- M7-SYS RT contient le système d'exploitation M7 RMOS 32 et des programmes système. Il est indispensable à l'utilisation des progiciels M7-ProC/C++ et CFC pour SIMATIC M7.

### 1.4.3 Interface homme/machine

Les interfaces homme/machine sont des logiciels spécifiques au contrôle-commande dans SIMATIC.

- Le système de visualisation du processus SIMATIC WinCC est un système de base indépendant des branches et technologies d'utilisation qui comporte toutes les fonctions indispensables au contrôle-commande.
- SIMATIC ProTool et SIMATIC ProTool/Lite sont des applications modernes permettant la configuration des visuels SIMATIC et des appareils compacts SIMATIC C7.
- ProAgent permet un diagnostic du processus précis et rapide dans les installations et machines en fournissant des informations relatives à la localisation et à la cause des erreurs.





## 2 Installation

### 2.1 Automation License Manager

#### 2.1.1 Licence d'utilisation avec Automation License Manager

##### Automation Licence Manager

Pour pouvoir utiliser le logiciel de programmation STEP 7, une "License Key" (clé de licence) spécifique au produit (licence d'utilisation) est nécessaire. A partir de la version V5.3 de STEP 7, celle-ci est installée par Automation License Manager.

Automation License Manager est un produit logiciel de Siemens AG. Il permet de gérer des "License Keys" (clés de licence, représentants techniques de licences) dans l'ensemble du système.

Automation License Manager est disponible :

- sur le support d'installation du produit logiciel correspondant pour lequel une "License Key" est requise ou
- sur un support d'installation distinct, ainsi que
- sur les pages Internet de A&D Customer Support de Siemens AG, sous forme de produit à télécharger.

Automation License Manager dispose d'une aide en ligne contextuelle intégrée, que vous pouvez appeler après l'installation en appuyant sur la touche F1 ou en choisissant la commande de menu ? > **Aide de License Manager**. Vous y trouverez des informations détaillées sur la fonctionnalité et l'utilisation de Automation License Manager.

##### Licences

Des licences sont nécessaires pour l'utilisation des progiciels de STEP 7 protégés par des droits de licence. Une licence est accordée comme droit d'utilisation de produits. Les représentants de ce droit sont :

- CoL (**C**ertificate **o**f **L**icense) et
- License Key (clé de licence)

## Certificate of License (CoL)

Le "Certificate of License" contenu dans la livraison des produits respectifs constitue la preuve juridique du droit d'utilisation. Seul le propriétaire du CoL ou les personnes mandatées sont autorisées à utiliser le produit respectif.

## License Keys (clés de licence)

La "License Key" (clé de licence) est le représentant technique d'une licence (marque de licence électronique).

SIEMENS AG attribue une "License Key" (clé de licence) pour tout logiciel protégé par des droits de licence. Le logiciel correspondant ne peut être utilisé conformément aux conditions de licence et d'utilisation liées à la "Licence Key" (clé de licence) correspondante que si la présence de la "License Key" (clé de licence) valide est constatée sur un ordinateur, après démarrage du logiciel.

---

### Nota

- Vous avez également la possibilité d'utiliser le logiciel de base de STEP 7 sans "License Key" (clé de licence) pour en découvrir l'interface utilisateur et les fonctionnalités.
  - L'utilisation sans restrictions, conforme aux déclarations de droits de licence, n'est cependant autorisée et possible que lorsque la "License Key" (clé de licence) est installée.
  - Si vous n'avez **pas** installé la "License Key" (clé de licence), vous êtes sollicité à intervalles réguliers de réaliser l'installation.
- 

Les "License Keys" (clé de licence) sont disponibles sur les supports de données suivants et peuvent être transférées d'un support à l'autre :

- disquettes "License Key" (clé de licence),
- disques durs locaux et
- disques durs d'ordinateurs dans le réseau.

Si vous avez installé des produits logiciels pour lesquels vous ne disposez pas de "License Key" (clé de licence), vous pouvez encore commander après-coup les licences requises.

Des informations supplémentaires sur l'utilisation des "License Keys" (clés de licence) sont fournies dans l'aide en ligne de Automation License Manager.

## Types de licences

Pour les produits logiciels de Siemens AG, il existe les types suivants de licences orientées application. Le comportement du logiciel est déterminé par les diverses "License Keys" (clés de licence) correspondant à ces types de licences. Le type d'utilisation est défini par le "Certificate of License" respectif.

Type de licence	Description
Single License	L'utilisation du logiciel est autorisée pour une durée illimitée sur un ordinateur quelconque.
Floating License	L'utilisation du logiciel est autorisée pour une durée illimitée sur un réseau (utilisation "à distance").
Trial License	L'utilisation du logiciel est restreinte : <ul style="list-style-type: none"> <li>à une validité de 14 jours au maximum,</li> <li>à un nombre de jours défini à partir de la première utilisation,</li> <li>à l'utilisation pour test et validation (à l'exclusion de toute responsabilité).</li> </ul>
Rental License	L'utilisation du logiciel est restreinte : <ul style="list-style-type: none"> <li>à une validité de 50 jours au maximum,</li> <li>à un nombre d'heures défini d'utilisation.</li> </ul>
Upgrade License	Une mise à niveau peut être nécessaire en raison d'exigences spécifiques de l'état du système : <ul style="list-style-type: none"> <li>Une licence de mise à niveau permet de transformer une licence d'une "ancienne" version x en une licence de version &gt;x+....</li> <li>Une mise à niveau peut p. ex. être requise en raison d'extension des capacités.</li> </ul>

## 2.1.2 Installation de Automation License Manager

L'installation de Automation License Manager est réalisée par un Setup MSI. Le logiciel d'installation de Automation License Manager se trouve sur le CD du produit STEP 7.

Vous pouvez installer Automation License Manager avec STEP 7 ou ultérieurement.

---

### Nota

- Des informations détaillées sur la procédure d'installation de Automation License Manager sont données dans le fichier Lisezmoi.wri actuel.
  - Tous les informations relatives à la fonctionnalité et à l'utilisation des "License Keys" (clés de licence) de Automation License Manager sont données dans l'aide en ligne.
- 

## Installation ultérieure de "License Keys" (clés de licence)

Si vous démarrez le logiciel STEP 7 et qu'aucune "License Key" (clé de licence) n'est présente, vous obtenez un message correspondant.

---

### Nota

- Vous avez également la possibilité d'utiliser le logiciel de base de STEP 7 sans "License Key" (clé de licence) pour en découvrir l'interface utilisateur et les fonctionnalités.
  - L'utilisation sans restrictions, conforme aux déclarations de droits de licence, n'est cependant autorisée et possible que lorsque la "License Key" (clé de licence) est installée.
  - Si vous n'avez **pas** installé la "License Key" (clé de licence), vous êtes sollicité à intervalles réguliers de réaliser l'installation.
- 

Pour installer les "License Keys" (clé de licence) ultérieurement, vous avez les possibilités suivantes :

- installez les "License Keys" (clé de licence) depuis les disquettes,
- installez les "License Keys" (clé de licence) en les téléchargeant depuis Internet (une commande préalable est requise),
- utilisez des "Floating License Keys" (clé de licence) disponibles sur Internet.

Vous trouverez des informations détaillées sur la procédure dans l'aide en ligne contextuelle de Automation License Manager, que vous pouvez appeler après l'installation en appuyant sur la touche F1 ou en choisissant la commande de menu ? > **Aide de License Manager**.

---

### Nota

- Les "License Keys" (clés de licence) sont uniquement fonctionnelles sous Windows 2000/XP/Server 2003 si elles se trouvent sur un disque dur possédant des droits d'accès en écriture.
  - Les "Floating Licenses" peuvent également être utilisées via un réseau, c'est-à-dire "à distance".
-



### 2.1.3 Règles d'utilisation des License Keys (clé de licence)

---



#### **Avertissement**

Tenez compte des indications d'utilisation des "License Keys" (clés de licence) décrites dans l'aide en ligne de Automation License Manager et dans le fichier Lisezmoi.wri sur le CD-ROM de STEP 7. Le non-respect de ces règles risque d'entraîner la perte irréversible des "License Keys" (clés de licence).

---

Vous pouvez appeler l'aide en ligne contextuelle de Automation License Manager en appuyant sur la touche F1 ou en choisissant la commande de menu ? > **Aide de License Manager**.

Cette aide en ligne contient toutes les informations relatives à la fonctionnalité et à l'utilisation des "License Keys".

## 2.2 Installation de STEP 7

STEP 7 contient un programme SETUP qui exécute l'installation automatiquement. Des messages s'affichant à l'écran vous guident étape par étape tout au long de la procédure d'installation. Vous l'appellez via la procédure d'installation de logiciel standard sous Windows 2000/XP/Server 2003.

Les phases principales de l'installation sont :

- la copie des données dans votre outil de développement,
- l'installation des pilotes pour EPROM et communication,
- l'installation des "License Keys" (clés de licence) (si souhaitée).

---

### Nota

Les consoles de programmation Siemens sont livrées avec, sur leur disque dur, le logiciel STEP 7 installable.

---

### Conditions d'installation

- Système d'exploitation  
Windows 2000, Windows XP, Windows Server 2003 de Microsoft.
- Matériel de base  
Ordinateur personnel (PC) ou console de programmation (PG) avec :
  - processeur pentium (600 MHz),
  - mémoire vive : 256 Mo au minimum,
  - moniteur couleur, clavier et souris pris en charge par Microsoft Windows.

Une console de programmation (PG) est un ordinateur personnel compact tout spécialement conçu pour être utilisé dans un environnement industriel. Elle est équipée en série de tous les programmes nécessaires à la programmation de systèmes d'automatisation SIMATIC.

- Mémoire requise  
Pour l'espace mémoire nécessaire sur le disque dur voir le fichier LISEZMOI.WRI.
- Interface MPI (optionnelle)  
L'interface multipoint MPI entre l'outil de développement (console de programmation ou ordinateur personnel) et le système cible n'est nécessaire que si vous voulez communiquer, sous STEP 7 et via MPI, avec le système cible.  
A cet effet, vous devez utiliser :
  - un adaptateur PC USB qui est relié à l'interface de communication de votre console ou
  - une carte MPI (p. ex. CP 5611) installée dans votre console.

L'interface MPI est déjà intégrée à certaines consoles de programmation.

- Programmeur d'EPROM externe (optionnel)  
Un programmeur d'EPROM externe n'est nécessaire, lorsque vous utilisez un PC, que si vous voulez programmer des EPROM.

---

**Nota**

Veillez également tenir compte des remarques sur l'installation de STEP 7 figurant dans le fichier LISEZMOI.WRI ainsi que de la "Liste de compatibilité des logiciels SIMATIC avec les versions du logiciel de base STEP 7".

Vous trouverez le fichier Lisezmoi en cliquant dans la barre des tâches sur **Démarrer > Simatic > Informations**.

La liste de compatibilité se trouve dans **Démarrer > Simatic > Documentation**.

---

## 2.2.1 Marche à suivre pour l'installation de STEP 7

### Préparatifs

Vous devez lancer le système d'exploitation (Windows 2000, XP ou Server 2003) avant de commencer l'installation.

- Un support de données externe est inutile si le logiciel STEP 7 installable se trouve déjà sur le disque dur de la PG.
- Pour effectuer l'installation à partir du CD-ROM, insérez le CD-ROM dans le lecteur de CD-ROM de votre PC.

### Lancement du programme d'installation

Procédez comme suit pour lancer l'installation :

1. Insérez le CD-ROM et lancez le Setup en cliquant sur "setup.exe".
2. Suivez étape par étape les instructions affichées par le programme d'installation.

Ce programme vous guide pas à pas tout au long de la procédure d'installation. Vous avez toujours la possibilité de revenir à l'étape précédente ou d'aller à l'étape suivante.

Pendant l'installation, des questions vous sont posées ou des options proposées dans des boîtes de dialogue. Tenez compte des indications ci-après qui vous permettront de répondre plus rapidement et aisément aux dialogues.

### **Version de STEP 7 déjà installée...**

Si le programme d'installation constate qu'une version de STEP 7 se trouve déjà sur l'outil de développement, un message vous le signale et vous avez les possibilités suivantes :

- interrompre l'installation pour, ensuite, désinstaller l'ancienne version de STEP7 sous Windows puis relancer l'installation ou
- poursuivre l'installation et substituer ainsi la nouvelle version à l'ancienne.

Une maintenance correcte du logiciel exigerait que vous désinstalliez toute version antérieure existante avant de procéder à une nouvelle installation. L'écrasement pur et simple d'une ancienne version présente, en outre, l'inconvénient qu'une désinstallation ultérieure n'effacerait pas les parties éventuellement encore existantes d'une installation précédente.

### **Choisir son installation**

Trois variantes d'installation au choix sont possibles :

- Standard : Installation de STEP 7 sur votre ordinateur avec tous ses composants. La boîte de dialogue suivante vous permettra de modifier le choix de la langue.
- Compacte : Installation de STEP 7 sur votre ordinateur avec le minimum de composants nécessaires. La boîte de dialogue suivante vous permettra de modifier le choix de la langue.
- Personnalisée : La boîte de dialogue suivante affiche tous les composants pouvant être installés. Vous pouvez choisir parmi ceux-ci les composants que vous désirez installer.

### **Numéro d'identification**

Un numéro d'identification vous est demandé durant l'installation. Entrez ce dernier. Vous le trouverez sur le certificat du logiciel ou sur la disquette "License Key" (clé de licence) correspondante.

### **Installation de "License Keys"**

Lors de l'installation, le programme vérifie si une "License Key" (clé de licence) correspondante existe sur le disque dur. Si aucune "License Key" (clé de licence) valide n'est reconnue, un message vous avertit que vous ne pouvez utiliser le logiciel qu'en présence d'une "License Key" (clé de licence). Vous pouvez, si vous le désirez, installer immédiatement les "License Keys" ou bien poursuivre l'installation de STEP 7 et installer les "License Keys" ultérieurement. Dans le premier cas, insérez la disquette "License Key" (clé de licence) fournie lorsque le message correspondant vous y invite.

### **Paramétrage de l'interface PG/PC**

Une boîte de dialogue de paramétrage de l'interface PG/PC s'affiche pendant l'installation. Lisez à cet effet "Paramétrage de l'interface PG/PC".

## Paramétrage de cartes à mémoire

Une boîte de dialogue relative au paramétrage de cartes à mémoire apparaît pendant l'installation.

- Vous n'avez pas besoin de pilote EPROM si vous n'utilisez pas de cartes mémoire. Choisissez alors l'option "Aucun".
- Sinon, choisissez l'option correspondant à votre PG.
- Si vous vous servez d'un ordinateur personnel, vous pouvez choisir un pilote pour programmeur d'EPROM externe. Vous devez alors également indiquer l'interface à laquelle ce programmeur est connecté (par exemple LPT1).

Vous pouvez modifier les paramètres choisis après l'installation en appelant le programme "Paramétrage de cartes mémoire" dans le groupe de programmes STEP 7 ou dans le panneau de configuration.

## Système de fichiers flash

Vous pouvez préciser, dans la boîte de dialogue de paramétrage de cartes à mémoire, s'il faut installer un système de fichiers flash.

Ce système est, par exemple, nécessaire si dans SIMATIC M7 vous voulez écrire ou effacer des fichiers individuels sur une carte mémoire EPROM sans en modifier le contenu restant .

Choisissez l'option d'installation du système de fichiers flash si vous voulez utiliser cette fonction et disposez d'une console de programmation (PG 720/740/760, Field PG et Power PG) ou d'un programmeur d'EPROM appropriés.

## Erreurs pendant l'installation

Les erreurs suivantes entraînent l'interruption de l'installation :

- Si une erreur d'initialisation se produit immédiatement après le démarrage du SETUP, vous avez certainement lancé l'installation dans un environnement autre que Windows.
- L'espace mémoire est insuffisant : selon l'option d'installation choisie, vous avez besoin d'environ 100 Mo d'espace libre sur votre disque dur pour le logiciel de base.
- CD défectueux: adressez-vous à votre agence Siemens si vous constatez qu'un CD est défectueux.
- Erreur de manipulation : recommencez l'installation en observant rigoureusement les instructions.

## Fin de l'installation

Un message s'affiche à l'écran pour vous signaler que l'installation a réussi.

Si l'installation a entraîné l'actualisation des fichiers système, vous êtes invité à relancer Windows. Une fois Windows redémarré (démarrage à chaud), vous pouvez cliquer sur l'icône du SIMATIC Manager pour lancer l'interface utilisateur de STEP7.

Une installation sans erreur s'achève par la création d'un groupe de programmes pour STEP 7.

## 2.2.2 Paramétrage de l'interface PG/PC

Le paramétrage que vous réalisez ici vous permet de définir la communication entre PG/PC et système d'automatisation. Une boîte de dialogue de paramétrage de l'interface PG/PC s'affiche pendant l'installation. Vous pouvez également afficher cette boîte de dialogue après l'installation en appelant le programme "Paramétrage de l'interface PG/PC". Ce programme vous permet de modifier les jeux de paramètres après coup, indépendamment d'une quelconque installation.

### Principe

L'utilisation d'une interface nécessite :

- des paramétrages dans le système d'exploitation,
- un jeu de paramètres adéquat.

Lorsque vous utilisez un PC avec une carte MPI ou des processeurs de communication (CP), vous devez vérifier l'affectation des interruptions et des adresses dans le "Panneau de configuration" de Windows, pour vous assurer de l'absence de conflits d'interruptions ou de recoupement de plages d'adresses.

Le composant ISA (carte MPI ISA) n'est plus pris en charge sous Windows 2000, Windows XP et Windows Server 2003 et son installation ne vous est pas proposée.

Des jeux de paramètres prédéfinis vous sont proposés dans la boîte de dialogue afin de simplifier le paramétrage de l'interface PG/PC.

### Paramétrage de l'interface PG/PC

Procédez de la manière suivante (une description plus détaillée est donnée dans l'aide en ligne) :

1. Dans le "Panneau de configuration" de Windows, effectuez un double clic sur "Paramétrage de l'interface PG/PC".
2. Sélectionnez "S7ONLINE" comme "Entrée de l'application".
3. Sélectionnez le jeu de paramètres souhaité dans la liste "Jeux de paramètres utilisés". Si le jeu de paramètres souhaité ne figure pas dans la liste proposée, vous devez d'abord installer un module ou un protocole en cliquant sur le bouton "Sélectionner". Le jeu de paramètres est alors automatiquement créé. Dans les systèmes Plug&Play, les CP Plug&Play (CP 5611 et CP 5511) ne peuvent pas être installés manuellement. Ils sont automatiquement intégrés au "Paramétrage de l'interface PG/PC" lorsque vous les branchez dans votre PG/PC.
  - Si vous sélectionnez une interface **avec détection automatique des paramètres de bus**, (par exemple un CP 5611 (Auto)), vous pouvez connecter la PG ou le PC au réseau MPI ou PROFIBUS sans devoir sélectionner ces paramètres de bus. Pour des vitesses de transmission inférieures à 187,5 kBit/s, des temps d'attente allant jusqu'à une minute ne sont pas exclus.  
**Condition pour la détection automatique** : les maîtres connectés au bus répartissent les paramètres de bus de manière cyclique ; tous les nouveaux composants MPI le font ; la répartition cyclique des paramètres de bus ne doit pas être désactivée pour les sous-réseaux PROFIBUS (paramétrage par défaut du réseau PROFIBUS).

- Lorsque vous sélectionnez une interface **sans détection automatique des paramètres de bus**, vous pouvez afficher les paramètres afin de les adapter au sous-réseau.

Des modifications sont également indispensables en cas de conflit avec d'autres paramètres (par exemple, affectation d'interruptions ou d'adresses). Dans ce cas, effectuez les modifications requises en utilisant la fonction d'ajout de nouveau matériel et le panneau de configuration de Windows (voir ci-après).



#### **Avertissement**

Ne supprimez en aucun cas le jeu de paramètres "TCP/IP" éventuellement présent !

Vous risqueriez de perturber l'exécution des autres applications.

---

### **Contrôle de l'affectation des interruptions et adresses**

Lorsque vous utilisez un PC avec carte MPI, vous devez absolument vérifier si l'interruption et la plage d'adresses prédéfinies sont libres.

## **2.3 Désinstallation de STEP 7**

Utilisez la procédure de désinstallation courante sous Windows :

1. Lancez, sous Windows, le dialogue d'installation de logiciel en effectuant un double clic sur l'icône "Ajout/Suppression de programmes" dans le "Panneau de configuration".
2. Sélectionnez l'entrée STEP 7 dans la liste affichée des logiciels installés. Cliquez sur le bouton de suppression du logiciel.
3. Si des boîtes de dialogue de suppression de fichiers autorisés apparaissent, cliquez sur le bouton "Non" en cas de doute.



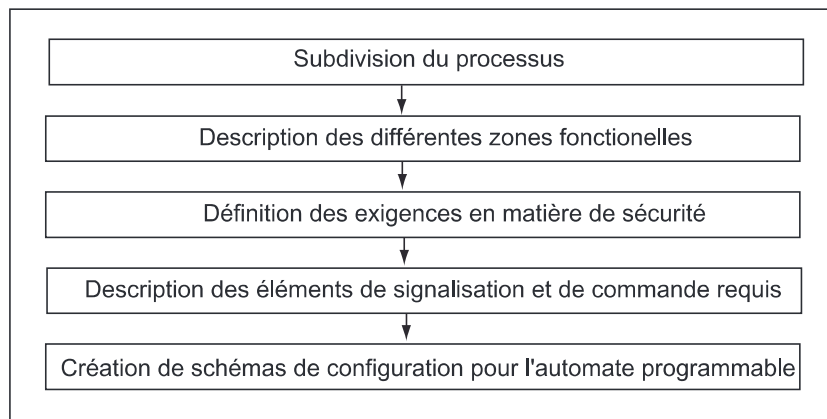


## 3 Conception de la solution d'automatisation

### 3.1 Conception d'une solution d'automatisation

Ce chapitre donne des informations pour l'exécution des tâches fondamentales nécessaires à la planification d'une solution d'automatisation pour un automate programmable (AP). Un exemple d'automatisation de processus de mélange industriel vous explique comment procéder étape par étape.

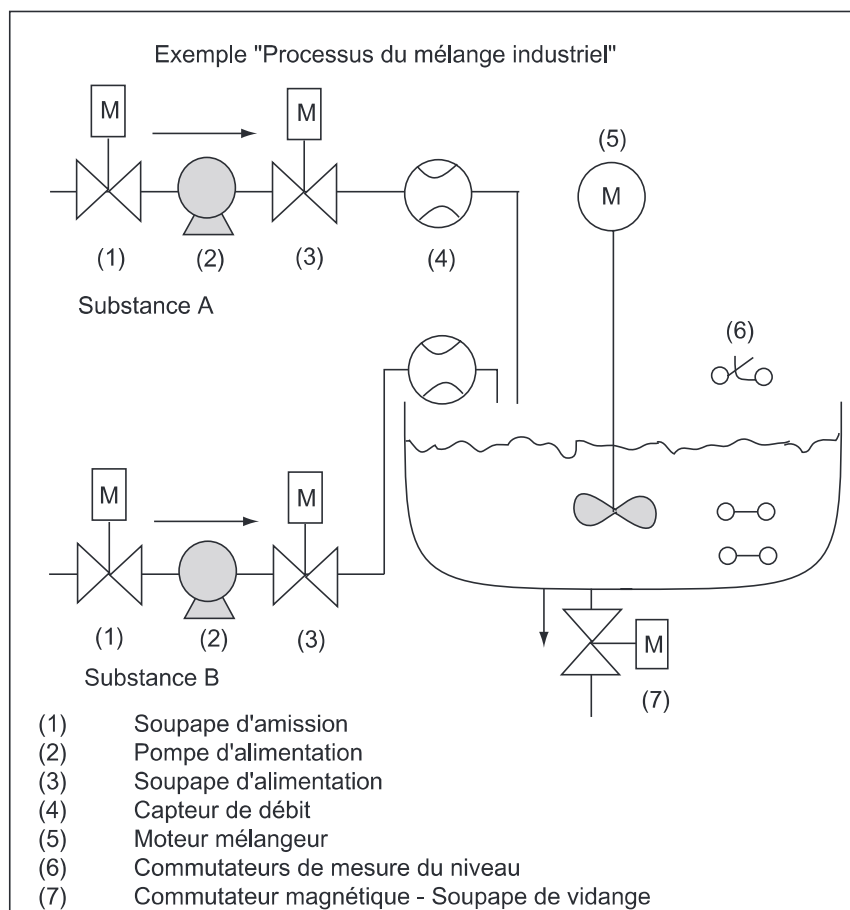
Il existe de nombreuses méthodes pour concevoir une solution d'automatisation. La figure ci-après montre la marche à suivre fondamentale que vous pouvez appliquer à tout projet.



### 3.2 Subdivision du processus en tâches et zones

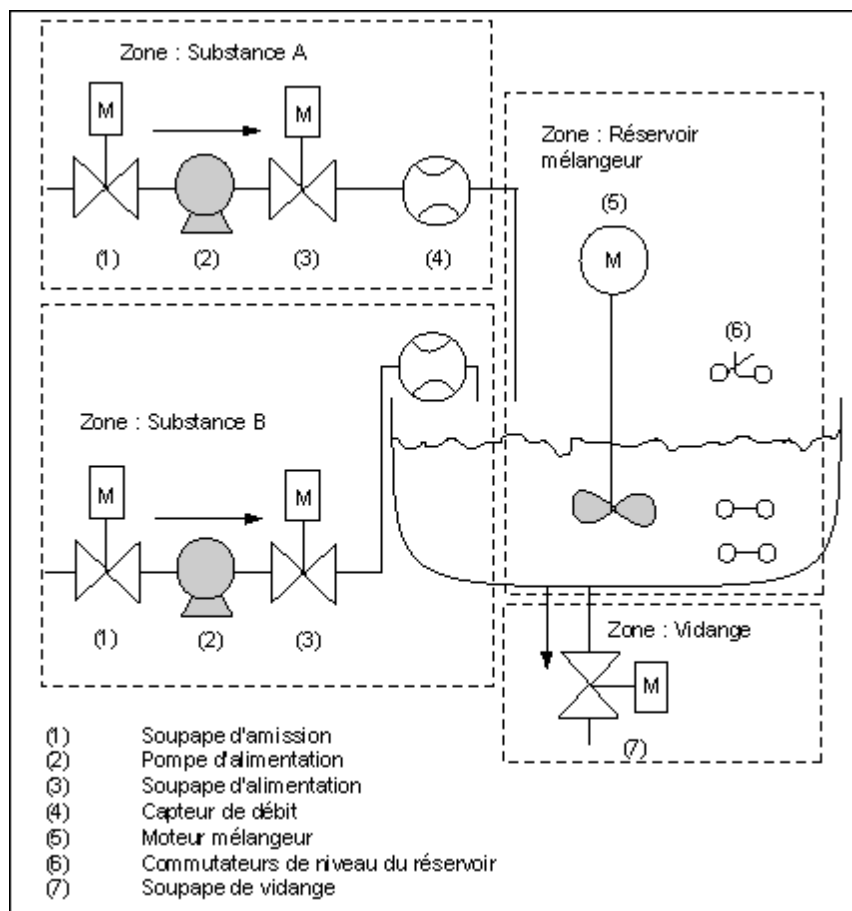
Un processus d'automatisation est constitué de différentes tâches. Il est possible de définir même le processus le plus complexe en déterminant des zones cohérentes au sein du processus et en subdivisant ces dernières en tâches partielles plus petites.

L'exemple suivant vous montre, en se basant sur un processus de mélange industriel, comment structurer un processus en zones fonctionnelles et en tâches individuelles.



## Identification des zones du processus

Une fois le processus à commander défini, décomposez le projet en groupes ou zones apparentées.



Comme chaque zone est à son tour subdivisée en tâches plus petites, les tâches nécessaires pour commander la partie correspondante du processus ne sont pas très complexes.

Dans notre exemple de mélangeur industriel, nous pouvons identifier quatre zones (voir le tableau suivant). La zone pour la substance A nécessite le même équipement que la zone pour la substance B.

Zone fonctionnelle	Équipement associé
Substance A	Pompe d'alimentation pour la substance A Soupape d'admission pour la substance A Soupape d'alimentation pour la substance A Capteur de débit pour la substance A
Substance B	Pompe d'alimentation pour la substance B Soupape d'admission pour la substance B Soupape d'alimentation pour la substance B Capteur de débit pour la substance B
Réservoir mélangeur	Moteur mélangeur Commutateurs de mesure du niveau
Vidange	Soupape de vidange

### 3.3 Description des différentes zones fonctionnelles

Lorsque vous décrivez chaque zone et chaque tâche dans votre processus, vous définissez non seulement le fonctionnement de chaque zone, mais également les différents éléments commandant cette zone, à savoir :

- les entrées et sorties logiques, mécaniques et électriques pour chaque tâche,
- les verrouillages et les relations de dépendance entre les différentes tâches.

Notre exemple de processus de mélange industriel fait appel à des pompes, des moteurs et des soupapes. Il faut décrire chacun de ces éléments précisément afin d'identifier leurs caractéristiques de fonctionnement et le type des verrouillages nécessaires pendant l'exploitation. Les tableaux suivants fournissent des modèles de description de l'équipement utilisé dans le mélangeur industriel pris en exemple. Vous pouvez également vous servir de ces descriptions pour vous procurer l'équipement nécessaire.

<b>Substances A et B : moteurs des pompes d'alimentation</b>
Les pompes d'alimentation amènent les substances A et B au réservoir mélangeur. <ul style="list-style-type: none"> <li>• Débit : 400 l par minute</li> <li>• Puissance : 100 KW pour 1200 tours/min</li> </ul>
Le démarrage et l'arrêt des pompes sont commandés à partir d'un poste d'opération situé à proximité du réservoir mélangeur. Le nombre de démarrages est comptabilisé à des fins de maintenance. Il est possible de remettre à zéro le compteur et l'indicateur à l'aide d'un même bouton-poussoir.
Les conditions de validation sont les suivantes : <ul style="list-style-type: none"> <li>• Le réservoir mélangeur n'est pas plein.</li> <li>• La soupape de vidange du réservoir mélangeur est fermée.</li> <li>• L'arrêt d'urgence n'est pas activé.</li> </ul>
Les conditions d'arrêt sont les suivantes : <ul style="list-style-type: none"> <li>• Le capteur de débit ne signale pas de débit 7 s après le déclenchement du moteur des pompes.</li> <li>• Le capteur de débit ne signale plus de débit pendant le fonctionnement.</li> </ul>

<b>Substances A et B : soupapes d'admission et d'alimentation</b>
Les soupapes d'admission et d'alimentation pour les substances A et B permettent ou empêchent l'arrivée des substances dans le réservoir mélangeur. Ces soupapes comportent un commutateur magnétique avec rappel à ressort. <ul style="list-style-type: none"> <li>• La soupape est ouverte lorsque le commutateur magnétique est activé.</li> <li>• La soupape est fermée lorsque le commutateur magnétique est désactivé.</li> </ul>
Les soupapes d'admission et d'alimentation sont commandées par le programme utilisateur.
La condition de validation est la suivante : <ul style="list-style-type: none"> <li>• Le moteur de la pompe d'alimentation fonctionne pendant une seconde au moins.</li> </ul>
Les conditions d'arrêt sont les suivantes : <ul style="list-style-type: none"> <li>• Le capteur de débit ne signale pas de débit.</li> </ul>

### **Moteur mélangeur**

Le moteur mélangeur mélange les substances A et B dans le réservoir mélangeur.

- Puissance : 100 KW pour 1200 tours/min

1. Le démarrage et l'arrêt du moteur mélangeur sont commandés à partir d'un poste d'opération situé à proximité du réservoir mélangeur. Le nombre de démarrages est comptabilisé à des fins de maintenance. Il est possible de remettre à zéro le compteur et l'indicateur à l'aide d'un même bouton-poussoir.

Les conditions de validation sont les suivantes :

- Le capteur de niveau n'indique pas "Réservoir en dessous du minimum".
- La soupape de vidange du réservoir mélangeur est fermée.
- L'arrêt d'urgence n'est pas activé.

Les conditions d'arrêt sont les suivantes :

- Le capteur de débit ne signale pas que le régime nominal est atteint au plus tard 10 secondes après le déclenchement du moteur des pompes.

### **Soupape de vidange**

La soupape de vidange permet de vidanger le mélange (par gravitation) afin de l'amener à l'étape suivante du processus. Cette soupape comporte un commutateur magnétique avec rappel à ressort.

- Si le commutateur magnétique est activé, la soupape de vidange est ouverte.
- Si le commutateur magnétique est désactivé, la soupape de vidange est fermée.

L'ouverture et la fermeture de la soupape de vidange sont commandées à partir du poste d'opération.

La soupape de vidange peut être ouverte dans les conditions suivantes :

- Le moteur mélangeur est à l'arrêt.
- Le capteur de niveau ne signale pas "Réservoir vide".
- L'arrêt d'urgence n'est pas activé.

La condition d'arrêt est la suivante :

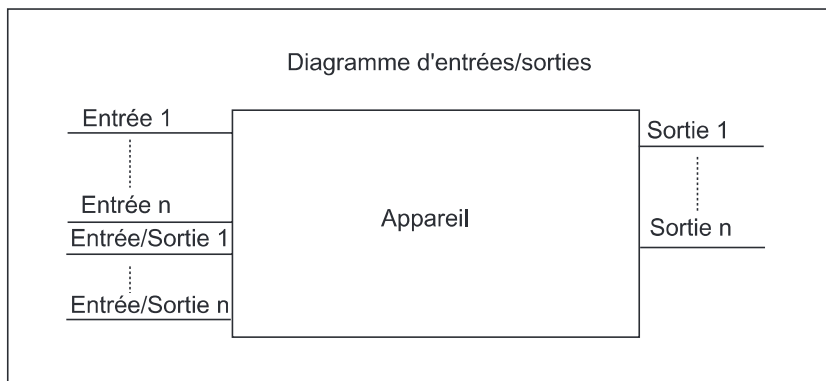
- Le capteur de niveau signale "Réservoir vide".

### **Commutateurs de mesure du niveau**

Les commutateurs de niveau informent sur le niveau dans le réservoir et servent également au verrouillage des pompes d'alimentation et du moteur mélangeur.

### 3.4 Liste des entrées, sorties et entrées/sorties

Après avoir décrit physiquement chaque appareil à commander, vous devez créer des diagrammes d'entrées/sorties pour chaque appareil ou zone.



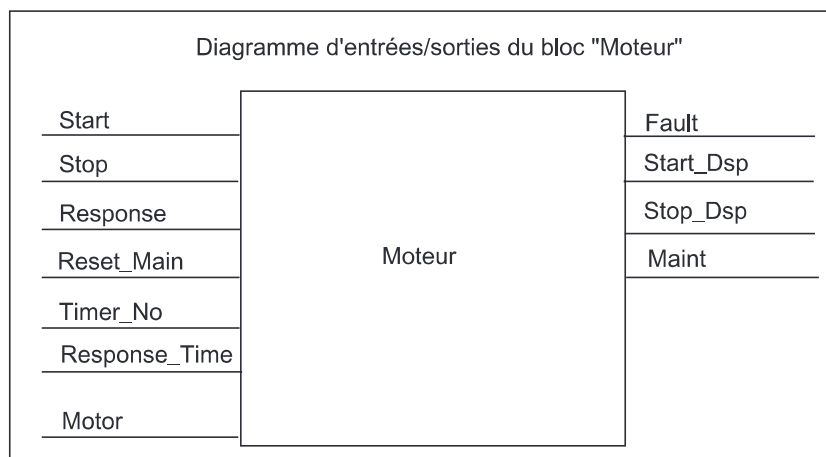
Ces diagrammes correspondent aux blocs de code à programmer.

### 3.5 Création d'un diagramme d'entrées/sorties pour les moteurs

Notre exemple de processus de mélange industriel fait appel à deux pompes d'alimentation et un moteur mélangeur. La commande des différents moteurs se fait via un bloc "Moteur" identique pour les trois appareils. Ce bloc requiert six entrées : deux entrées pour le démarrage et l'arrêt du moteur, une entrée pour la remise à zéro de l'indicateur de maintenance, une entrée pour le signal en retour du moteur (moteur en marche/moteur arrêté), une entrée pour l'intervalle de temps durant lequel doit parvenir le signal en retour et une entrée pour le numéro de la temporisation utilisée pour mesurer le temps.

Ce bloc de code nécessite en outre quatre sorties : deux sorties pour l'indication de l'état de fonctionnement du moteur, une sortie pour la signalisation d'erreurs et une sortie indiquant qu'il faut effectuer la maintenance du moteur.

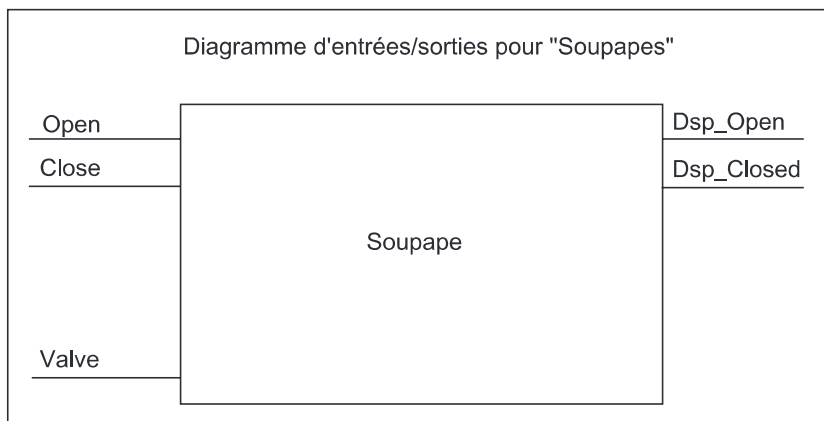
Ce bloc comporte également un paramètre d'entrée/sortie qui sert à commander le moteur mais est aussi traité et modifié dans le programme du bloc "Moteur".



### 3.6 Création d'un diagramme d'entrées/sorties pour les soupapes

La commande des différentes soupapes se fait via un bloc "Soupape" identique pour toutes les soupapes utilisées. Ce bloc de code comporte deux entrées: une entrée pour l'ouverture et une entrée pour la fermeture de la soupape. Il nécessite en outre deux sorties: une sortie signale que la soupape est ouverte et l'autre que la soupape est fermée.

Ce bloc comporte également un paramètre d'entrée/sortie qui sert à commander la soupape mais est aussi traité et modifié dans le programme du bloc "Soupape".





## 3.7 Définition des exigences en matière de sécurité

Choisissez les éléments nécessaires pour garantir la sécurité du processus, en accord avec les exigences légales et la ligne suivie par votre entreprise. Précisez, dans votre description, les influences qu'exercent ces éléments de sécurité sur les zones de votre processus.

### Définition des exigences en matière de sécurité

Déterminez les appareils qui nécessitent, pour des raisons de sécurité, des circuits câblés. Ces circuits de sécurité fonctionnent, par définition, indépendamment de l'automate programmable (bien qu'ils disposent, en général, d'une interface d'entrée/sortie pour assurer la coordination avec le programme utilisateur). En principe, vous configurez une matrice pour relier chaque actionneur à une zone d'arrêt d'urgence propre. Cette matrice constitue alors la base pour les schémas des circuits de sécurité.

Procédez comme suit pour concevoir les dispositifs de sécurité :

- Identifiez les verrouillages logiques et mécaniques ou électriques entre les différentes parties de l'automatisme.
- Concevez les circuits permettant de commander manuellement en cas d'urgence les appareils utilisés dans le processus.
- Déterminez les autres exigences relatives à la sécurité assurant un déroulement sûr du processus.

### Création d'un circuit de sécurité

Le mélangeur industriel pris comme processus-exemple fait appel à la logique suivante pour son circuit de sécurité :

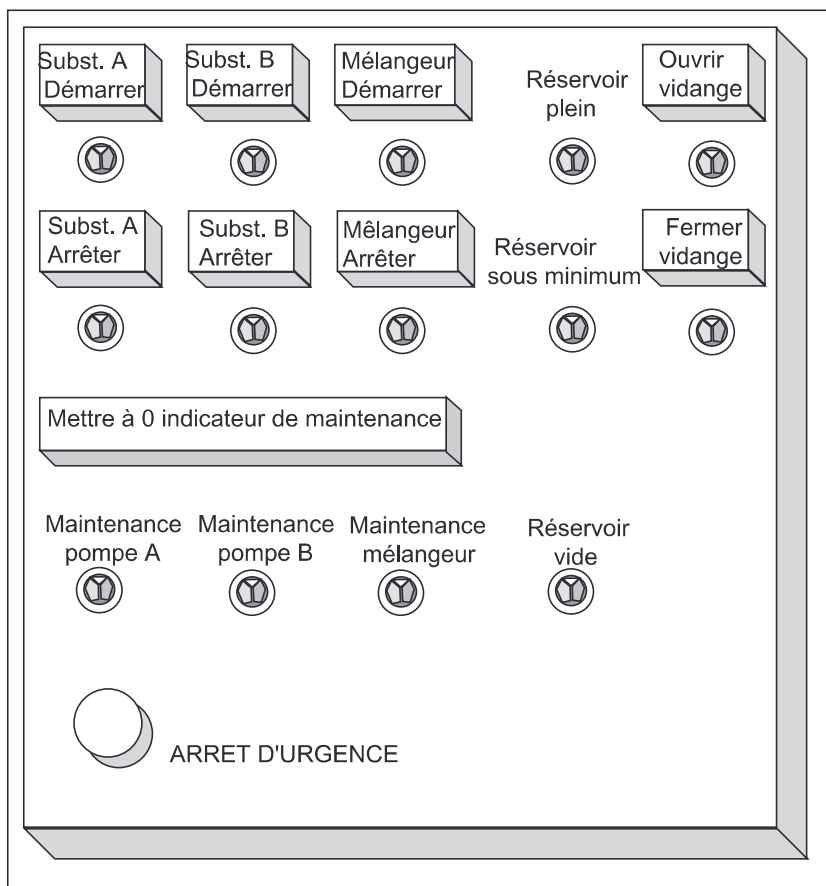
- Un commutateur d'arrêt d'urgence arrête les appareils suivants indépendamment de l'automate programmable :
  - Pompe d'alimentation pour la substance A
  - Pompe d'alimentation pour la substance B
  - Moteur mélangeur
  - Soupapes.
- Ce commutateur d'arrêt d'urgence est situé sur le poste d'opération.
- Une entrée de l'automatisme reflète l'état du commutateur d'arrêt d'urgence.

### 3.8 Description des éléments de signalisation et de commande requis

Tout processus nécessite un système de contrôle et de commande permettant à l'homme d'intervenir dans le processus. La mise au point de ce poste d'opération fait aussi partie des spécifications de conception.

#### Définition d'un poste d'opération

Dans notre exemple de mélangeur industriel, chaque appareil démarre ou s'arrête par l'intermédiaire d'un commutateur situé sur le poste d'opération. Ce poste comporte des indicateurs montrant l'état de fonctionnement (voir la figure suivante).



Il dispose également de lampes de signalisation pour les appareils devant faire l'objet d'une maintenance après un nombre donné de démarrages et d'un bouton d'arrêt d'urgence arrêtant immédiatement le processus. Sur le poste d'opération se trouve également un bouton de remise à zéro pour l'indicateur de maintenance des trois moteurs. Il vous permet d'éteindre les lampes de signalisation de maintenance pour les moteurs ayant besoin d'une maintenance et de mettre à zéro les valeurs correspondantes des compteurs pour l'intervalle entre les maintenances.

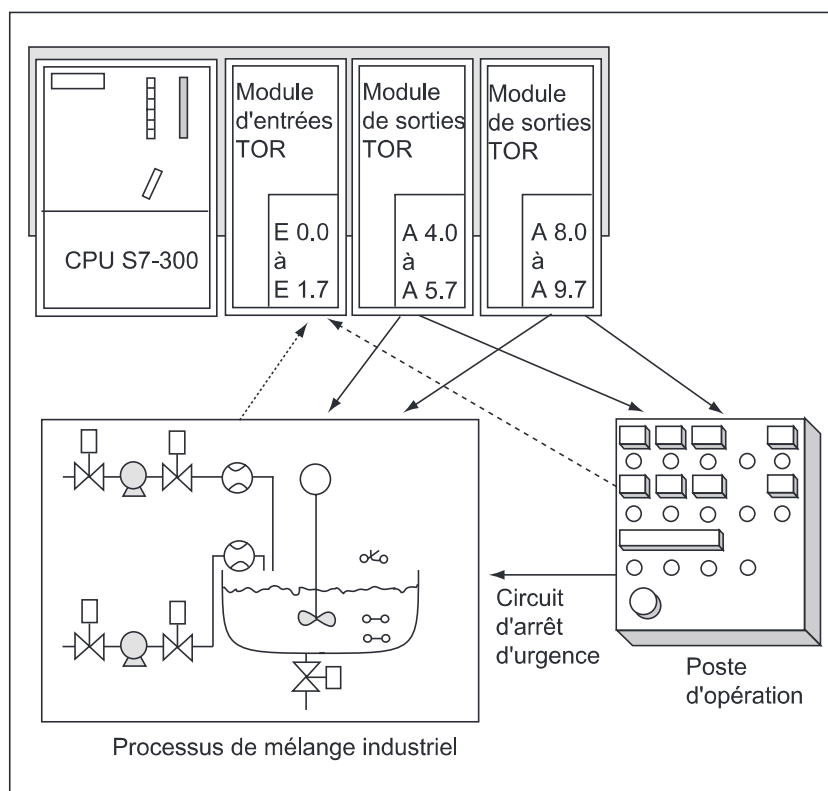
### 3.9 Création du schéma de configuration

Déterminez, une fois les exigences de conception documentées, l'équipement de commande nécessaire pour ce projet.

En décidant des modules qui seront utilisés, vous déterminez la structure de votre automate programmable. Créez un schéma de configuration dans lequel vous spécifiez les points suivants :

- type de CPU,
- nombre et type des modules de signaux,
- configuration des entrées et sorties physiques.

La figure suivante montre l'exemple d'une configuration S7 pour le mélangeur industriel.





## 4 Principes de conception d'une structure de programme

### 4.1 Programmes dans une CPU

Deux programmes différents s'exécutent dans une CPU :

- le système d'exploitation et
- le programme utilisateur.

#### Systeme d'exploitation

Le système d'exploitation, contenu dans chaque CPU, organise toutes les fonctions et procédures dans la CPU qui ne sont pas liées à une tâche d'automatisation spécifique. Ses tâches sont les suivantes :

- le déroulement du démarrage à chaud et du redémarrage,
- l'actualisation de la mémoire image des entrées et l'émission de la mémoire image des sorties,
- l'appel du programme utilisateur,
- l'enregistrement des alarmes et l'appel des OB d'alarme,
- la détection et le traitement d'erreurs,
- la gestion des zones de mémoire,
- la communication avec des consoles de programmation et d'autres partenaires de communication.

La modification des paramètres par défaut du système d'exploitation permet d'influer sur le comportement de la CPU dans des domaines précis.

#### Programme utilisateur

Vous devez créer votre programme utilisateur et le charger dans la CPU. Il contient toutes les fonctions nécessaires au traitement de votre tâche d'automatisation spécifique. Il doit entre autres :

- déterminer les conditions pour le démarrage à chaud et le redémarrage de la CPU (par exemple, initialiser des signaux),
- traiter des données du processus (par exemple, combiner des signaux binaires, lire et exploiter des valeurs analogiques, définir des signaux binaires pour la sortie, écrire des valeurs analogiques),
- réagir aux alarmes,
- traiter les perturbations dans le déroulement normal du programme.

## 4.2 Blocs dans le programme utilisateur

Le logiciel de programmation STEP 7 vous permet de structurer votre programme utilisateur, c'est-à-dire de le subdiviser en différentes parties autonomes. Il en résulte les avantages suivants :

- écrire des programmes importants mais clairs,
- standardiser certaines parties du programme,
- simplifier l'organisation du programme,
- modifier facilement le programme,
- simplifier le test du programme, car vous pouvez l'exécuter section par section,
- faciliter la mise en service.

Dans notre exemple de processus de mélange industriel, vous avez appris à subdiviser votre processus d'automatisation en différentes tâches. Les parties d'un programme utilisateur structuré correspondent à ces différentes tâches ; il s'agit des blocs du programme.

### Types de bloc

Vous pouvez utiliser différents types de bloc dans un programme utilisateur S7 :

Bloc	Brève description de la fonction	Pour plus de détails, voir
Blocs d'organisation (OB)	Les OB déterminent la structure du programme utilisateur.	Blocs d'organisation et structure du programme
Blocs fonctionnels système (SFB) et fonctions système (SFC)	Les SFB et SFC sont intégrés à la CPU S7 et vous permettent de réaliser quelques fonctions systèmes importantes.	Blocs fonctionnels système (SFB) et fonctions système (SFC)
Blocs fonctionnels (FB)	Les FB sont des blocs avec "mémoire" que vous programmez vous-même.	Blocs fonctionnels (FB)
Fonctions (FC)	Les FC contiennent des routines de programmes pour les fonctions fréquemment utilisées.	Fonctions (FC)
Blocs de données d'instance (DB d'instance)	Les DB d'instance sont affectés au bloc FB/SFB appelé. Ils sont générés automatiquement lors de la compilation.	Blocs de données d'instance
Blocs de données (DB)	Les DB sont des zones de données dans lesquelles l'on enregistre les données utilisateur. Outre les données affectées respectivement à un bloc fonctionnel, vous pouvez définir des données globales utilisables par tous les blocs.	Blocs de données globaux (DB)

Les OB, FB, SFB, FC et SFC contiennent des parties de programme et sont de ce fait également désignés comme blocs de code. Le nombre de blocs autorisés par type de bloc ainsi que la longueur maximale de chaque bloc dépendent de la CPU.

### 4.2.1 Blocs d'organisation et structure du programme

Les blocs d'organisation (OB) constituent l'interface entre le système d'exploitation et le programme utilisateur. Ils sont appelés par le système d'exploitation et gèrent le traitement de programme cyclique et déclenché par alarme, ainsi que le comportement à la mise en route de l'automate programmable et le traitement des erreurs. Vous pouvez programmer les blocs d'organisation et déterminer ainsi le comportement de la CPU.

#### Priorité des blocs d'organisation

Les blocs d'organisation définissent l'ordre (événements de déclenchement) dans lequel les différentes parties du programme sont traitées. L'exécution d'un OB peut être interrompue par l'appel d'un autre OB. Cette interruption se fait selon la priorité : les OB de priorité plus élevée interrompent les OB de priorité plus faible. La priorité la plus faible est celle de l'OB d'arrière-plan.

#### Types d'alarme et classes de priorité

On appelle alarmes les événements qui déclenchent l'appel d'un OB donné. Le tableau suivant présente les types d'alarme pour STEP 7 et la priorité des blocs d'organisation associés. Tous les blocs d'organisation indiqués et toutes leurs classes de priorité ne sont pas contenus dans toutes les CPU S7 (voir le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" ainsi que le manuel de référence "Systèmes d'automatisation S7-400, M7-400 - Caractéristiques des modules").

Type d'alarme	Bloc d'organisation	Classe de priorité (prédéfinie)	Pour plus de détails, voir
Cycle libre	OB1	1	Bloc d'organisation pour le traitement de programme cyclique (OB1)
Alarmes horaires	OB10 à OB17	2	Blocs d'organisation pour l'alarme horaire (OB10 à OB17)
Alarmes temporisées	OB20	3	Blocs d'organisation pour l'alarme temporisée (OB20 à OB23)
	OB21	4	
	OB22	5	
	OB23	6	
Alarmes cycliques	OB30	7	Blocs d'organisation pour l'alarme cyclique (OB30 à OB38)
	OB31	8	
	OB32	9	
	OB33	10	
	OB34	11	
	OB35	12	
	OB36	13	
	OB37	14	
	OB38	15	

Type d'alarme	Bloc d'organisation	Classe de priorité (prédéfinie)	Pour plus de détails, voir
Alarmes de processus	OB40 OB41 OB42 OB43 OB44 OB45 OB46 OB47	16 17 18 19 20 21 22 23	Blocs d'organisation pour l'alarme de processus (OB40 à OB47)
Alarme DPV1	OB 55 OB 56 OB 57	2 2 2	Programmation d'appareils DPV1
Alarme multiprocesseur	OB60 multiprocesseur	25	Mode multiprocesseur - fonctionnement synchrone de plusieurs CPU
Alarmes de synchronisme d'horloge	OB 61 OB 62 OB 63 OB 64	25	Configuration de temps de réaction du processus courts et de durée identique sur PROFIBUS DP
Erreur de redondance	OB70 Erreur de redondance de périphérie (uniquement dans les systèmes H) OB72 Erreur de redondance de CPU (uniquement dans les systèmes H)	25 28	Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)
Erreurs asynchrones	OB80 Erreur de temps OB81 Erreur d'alimentation OB82 Alarme de diagnostic OB83 Alarme de débrogage/enfichage OB84 Erreur matérielle CPU OB85 Erreur d'exécution du programme OB86 Défaillance d'unité OB87 Erreur de communication	25 (ou 28 si l'OB d'erreur asynchrone figure dans le programme de mise en route)	Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)
Cycle en arrière-plan	OB90	29 <sup>1</sup>	Bloc d'organisation pour l'exécution du programme en arrière-plan (OB90)
Mise en route	OB100 Démarrage à chaud OB101 Redémarrage OB102 Démarrage à froid	27 27 27	Blocs d'organisation pour la mise en route (OB100/OB101/OB102)



Type d'alarme	Bloc d'organisation	Classe de priorité (prédéfinie)	Pour plus de détails, voir
Erreurs synchrones	OB121 Erreur de programmation OB122 Erreur d'accès à la périphérie	Priorité de l'OB à l'origine de l'erreur	Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)

1) A la classe de priorité 29 correspond la priorité 0.29. La priorité du cycle en arrière-plan et donc inférieure à celle du cycle libre.

## Modification de la priorité

STEP 7 permet de paramétrer les alarmes. Le paramétrage vous permet par exemple de désactiver des OB d'alarme ou de modifier des classes de priorité dans les blocs de paramètres : alarmes horaires, alarmes temporisées, alarmes cycliques et alarmes de processus.

Vous ne pouvez pas modifier la priorité des blocs d'organisation dans les CPU S7-300.

Dans les CPU S7-400 (ainsi que la CPU 318), vous pouvez modifier la priorité des blocs d'organisation suivants avec STEP 7 :

- OB10 à OB47,
- OB70 à OB72 (uniquement les CPU H) et OB81 à OB87 à l'état de marche (RUN).

Classes de priorité autorisée :

- les classes de priorité 2 à 23 pour les OB10 à OB47,
- les classes de priorité 2 à 28 pour les OB70 à OB72 ainsi que
- les classes de priorité 24 à 26 pour les OB81 à OB87 ; ces classes de priorité ont été étendues pour les CPU à partir de mi 2001 environ (version du Firmware 3.0) : pour les OB 81 à OB 84, de même que pour les OB 86 et OB 87, vous pouvez paramétrer les classes de priorité 2 à 26.

Vous pouvez affecter la même priorité à plusieurs OB. Les OB de priorité identique sont traités dans l'ordre d'apparition de leurs événements déclencheurs.

Les OB d'erreur déclenchés en cas d'erreurs synchrones sont traités selon la même classe de priorité que celle du bloc en cours d'exécution lors de la détection de l'erreur.

## Données locales

Vous pouvez déclarer des données locales temporaires lors de la création de blocs de code (OB, FC, FB). La zone de données locales disponible dans la CPU est partagée entre les différentes classes de priorité.

Dans les CPU S7-400, vous pouvez, avec STEP 7, modifier le nombre de données locales par classe de priorité dans le bloc de paramètres "Classes de priorité".

## Informations de déclenchement d'un OB

Chaque bloc d'organisation dispose d'informations de déclenchement de 20 octets de données locales que le système d'exploitation transmet lors du lancement d'un OB. Ces informations précisent l'événement ayant déclenché l'OB, la date et l'heure du déclenchement de l'OB, les erreurs apparues et les événements de diagnostic.

Les informations de déclenchement de l'OB40 d'alarme de processus contiennent, par exemple, l'adresse du module ayant généré l'alarme.

## OB d'alarme désactivés

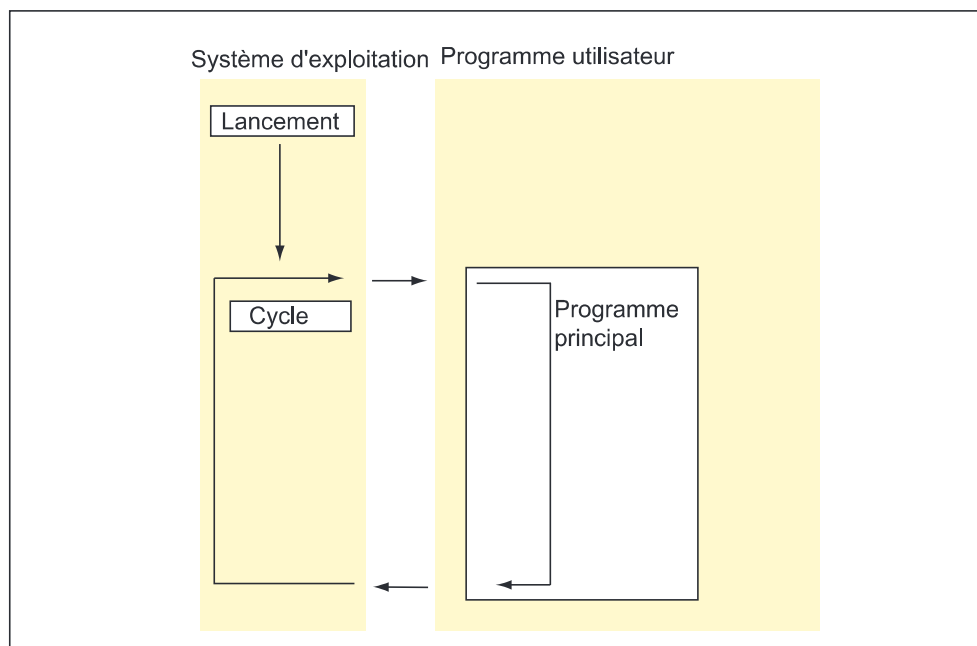
Si vous choisissez la classe de priorité 0 ou affectez moins de vingt octets de données locales à une classe de priorité, l'OB d'alarme correspondant est désactivé. Les OB d'alarme désactivés :

- ne peuvent pas être copiés et insérés dans le programme utilisateur à l'état de fonctionnement "Marche" (RUN),
- peuvent certes être copiés et insérés dans le programme utilisateur à l'état "Arrêt" (STOP), mais entraînent lors du démarrage à chaud de la CPU l'interruption de la mise en route et génèrent une entrée dans la mémoire tampon de diagnostic.

La désactivation des OB d'alarme inutiles augmente la zone de données locales libre disponible qui peut donc servir à la sauvegarde de données temporaires dans d'autres classes de priorité.

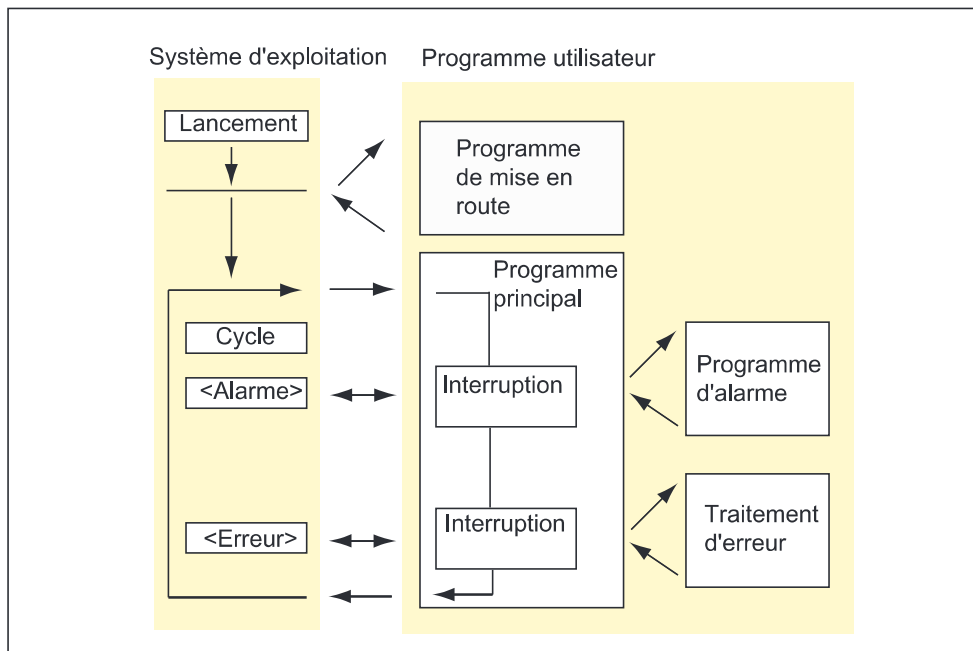
## Traitement de programme cyclique

Le traitement de programme cyclique constitue le traitement normal pour les automates programmables. Ceci signifie que le système d'exploitation parcourt une boucle de programme (le cycle) et appelle le bloc d'organisation OB1 dans le programme principal une fois par boucle. Le programme utilisateur dans le bloc OB1 est donc exécuté cycliquement.



### Traitement de programme déclenché par événement

Le traitement de programme cyclique peut être interrompu par des événements déclencheurs précis : les alarmes. En présence d'un tel événement, le bloc en cours d'exécution est interrompu à la fin de l'instruction et le bloc d'organisation associé à l'événement déclencheur est traité. Le traitement du programme cyclique reprend ensuite au point d'interruption.

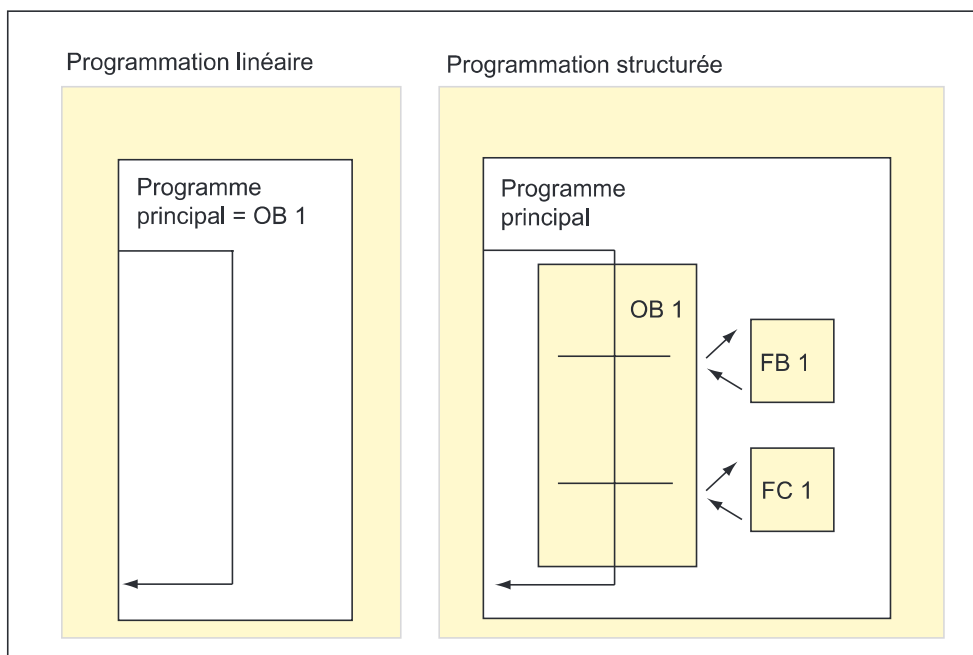


Vous avez ainsi la possibilité de ne traiter qu'en cas de besoin les parties du programme utilisateur qui ne doivent pas l'être cycliquement. Vous pouvez subdiviser votre programme en parties que vous répartissez dans différents blocs d'organisation. Il est ainsi recommandé d'utiliser un OB qui sera déclenché sur événement en réaction à un signal se présentant peu souvent (par exemple un signal de capteur indiquant qu'une cuve est pleine). L'apparition de l'événement déclenche alors le traitement de cette partie de programme.

## Programmation linéaire ou structurée

Vous pouvez écrire votre programme utilisateur complet dans l'OB1 (programmation linéaire). Cela n'est toutefois recommandé que pour des programmes simples s'exécutant sur des CPU S7-300 avec une mémoire peu importante.

Les automatismes complexes seront mieux traités si vous les subdivisez en parties plus petites qui correspondent aux fonctions technologiques du processus d'automatisation ou qui peuvent être utilisées plusieurs fois. Dans le programme utilisateur, ces tâches partielles sont représentées par des parties de programme correspondantes : les blocs (programmation structurée).



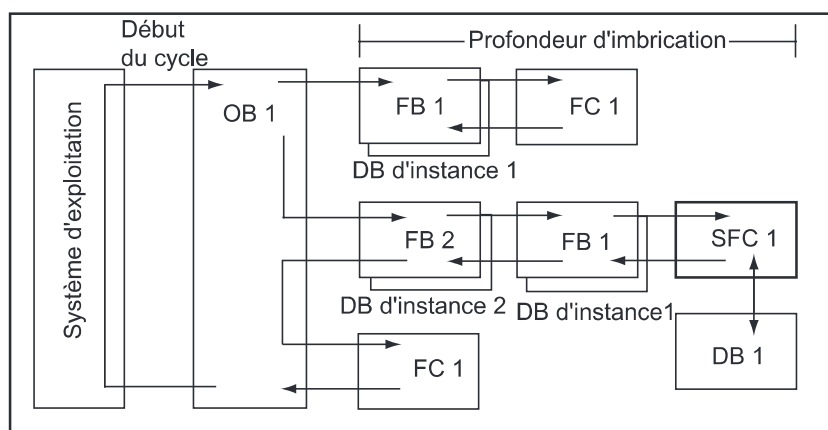
## 4.2.2 Hiérarchie d'appel dans le programme utilisateur

Pour faire fonctionner le programme utilisateur, vous devez appeler les blocs qui le composent. C'est ce que vous réalisez à l'aide d'opérations STEP 7 spéciales, les appels de blocs que vous ne pouvez programmer et démarrer que dans des blocs de code.

### Ordre et profondeur d'imbrication

On appelle hiérarchie d'appel l'ordre et l'imbrication des appels de blocs. Le niveau de profondeur autorisé pour les imbrications dépend de la CPU.

L'exemple de la figure suivante illustre l'ordre et l'imbrication des appels de blocs dans un cycle de traitement.



Règles relatives à l'ordre de création des blocs :

- Vous créez les blocs de haut en bas, ce qui signifie que vous commencez par la rangée de blocs supérieure.
- Tout bloc appelé doit déjà exister, ce qui signifie que dans une rangée de blocs, le sens de création est de droite à gauche.
- En dernier, vous créez l'OB1.

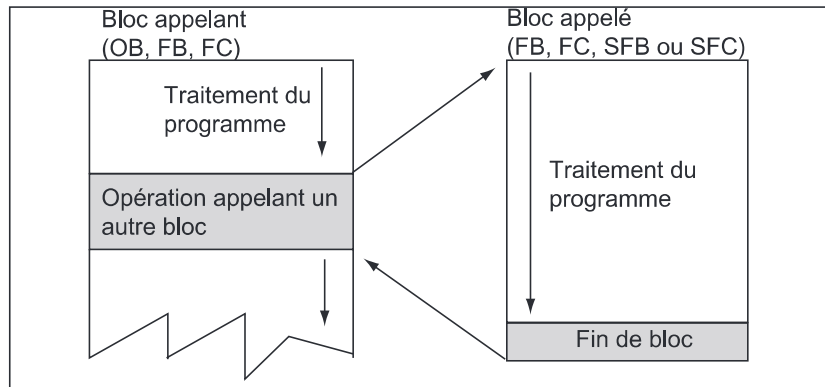
La transposition de ces règles signifie l'ordre de création suivant dans l'exemple considéré :  
 FC1 > FB1 + DB d'instance 1 > DB1 > SFC1 > FB2 + DB d'instance 2 > OB1

#### Nota

Lorsque la profondeur d'imbrication est trop grande, la pile des données locales risque de déborder (voir aussi Pile des données locales).

## Appels de blocs

La figure suivante montre comment s'exécute un appel de bloc au sein d'un programme utilisateur : le programme appelle le deuxième bloc dont les opérations sont alors traitées dans leur intégralité. Une fois le bloc appelé achevé, le traitement se poursuit avec l'opération suivant l'appel de bloc dans le bloc appelant.



Avant de programmer un bloc, vous devez déterminer les données que le programme doit traiter : vous déclarez les variables du bloc.

### Nota

- Les paramètres OUT doivent être décrits à chaque appel de bloc.
- Le système d'exploitation remet à zéro les instances du SFB3 "TP" lors d'un démarrage à froid. Pour initialiser des instances de ce SFB après le démarrage à chaud, vous devez les appeler avec PT = 0 ms dans l'OB100. C'est ce que vous pouvez par exemple obtenir avec une routine d'initialisation dans les blocs contenant des instances de ce SFB.

## 4.2.3 Catégories de blocs

### 4.2.3.1 Bloc d'organisation pour le traitement de programme cyclique (OB1)

Le traitement de programme cyclique constitue le traitement normal pour les automates programmables. Le système d'exploitation appelle l'OB1 cycliquement et déclenche ainsi le traitement cyclique du programme utilisateur.

#### Déroulement du traitement de programme cyclique

Le tableau suivant montre les phases du traitement de programme cyclique :

Phase	Déroulement dans les CPU jusqu'à 10/98	Déroulement dans les CPU à partir de 10/98
1	Le système d'exploitation démarre la surveillance du temps de cycle.	Le système d'exploitation démarre la surveillance du temps de cycle.
1er 2	La CPU lit l'état des entrées dans les modules d'entrées et met à jour la mémoire image des entrées.	Elle écrit ensuite les valeurs de la mémoire image des sorties dans les modules de sorties.
2e 3	Puis, elle traite le programme utilisateur et exécute les opérations indiquées dans le programme.	La CPU lit l'état des entrées dans les modules d'entrées et met à jour la mémoire image des entrées.
3e 4	Elle écrit ensuite les valeurs de la mémoire image des sorties dans les modules de sorties.	Puis, elle traite le programme utilisateur et exécute les opérations indiquées dans le programme.
4e 5	A la fin d'un cycle, le système d'exploitation exécute les travaux en attente, par exemple le chargement et l'effacement de blocs ou la réception et l'émission de données globales.	A la fin d'un cycle, le système d'exploitation exécute les travaux en attente, par exemple le chargement et l'effacement de blocs ou la réception et l'émission de données globales.
5e 6	La CPU revient alors au début du cycle et démarre à nouveau la surveillance du temps de cycle.	La CPU revient alors au début du cycle et démarre à nouveau la surveillance du temps de cycle.

#### Mémoires image du processus

Pour disposer d'une image cohérente des signaux du processus pendant la durée du traitement de programme cyclique, la CPU n'accède pas directement aux modules de signaux lors de l'utilisation des zones d'opérandes Entrées (E) et Sorties (A), mais à une zone de mémoire interne de la CPU qui contient une image des entrées et sorties.

#### Programmation du traitement de programme cyclique

Pour programmer le traitement cyclique, vous écrivez votre programme utilisateur avec STEP 7 dans l'OB1 et les blocs qui y sont appelés.

Le traitement de programme cyclique commence dès que le programme de mise en route s'est achevé sans erreur.



## Possibilités d'interruption

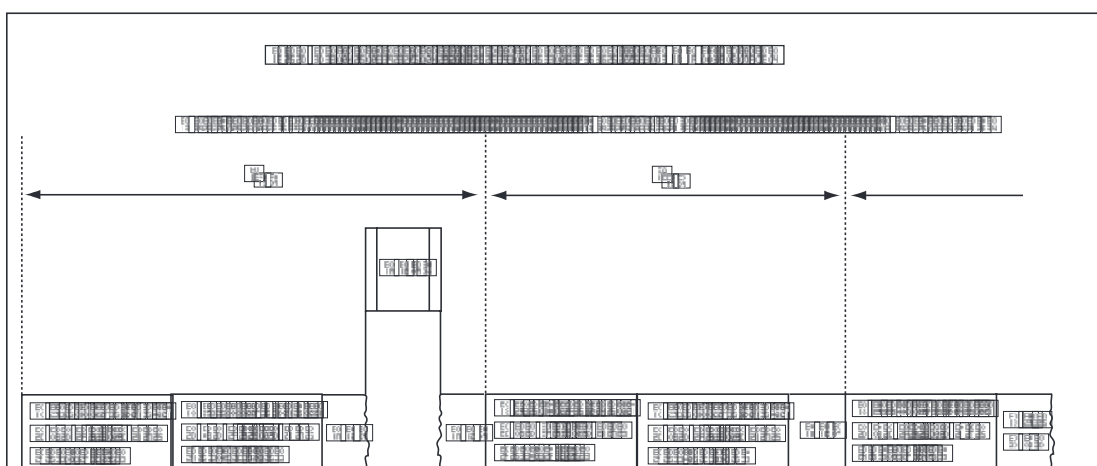
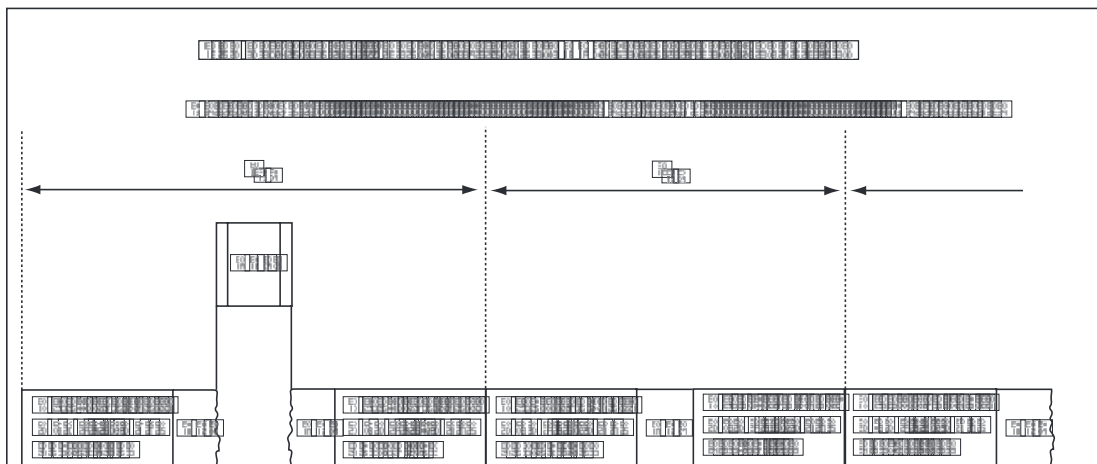
Le traitement de programme cyclique peut être interrompu par :

- une alarme,
- une commande STOP (commutateur de mode, commande de menu depuis la PG, SFC46 STP, SFB20 STOP),
- une coupure de tension secteur,
- l'apparition d'une erreur de matériel ou de programme.

## Temps de cycle

Le temps de cycle est le temps dont a besoin le système d'exploitation pour le traitement du programme cyclique ainsi que de toutes les parties de programme interrompant ce cycle (par exemple, traitement des autres blocs d'organisation) et des activités du système (par exemple, mise à jour de la mémoire image). Ce temps est contrôlé.

Ce temps ( $T_c$ ) n'est pas identique à chaque cycle. Les figures suivantes indiquent différents temps de cycle ( $TZ1 \neq TZ2$ ) pour les anciennes et les CPU jusqu'à 10/98 et les CPU à partir de 10/98.



L'OB1 est interrompu par une alarme horaire dans le cycle en cours.

### Temps de surveillance du cycle

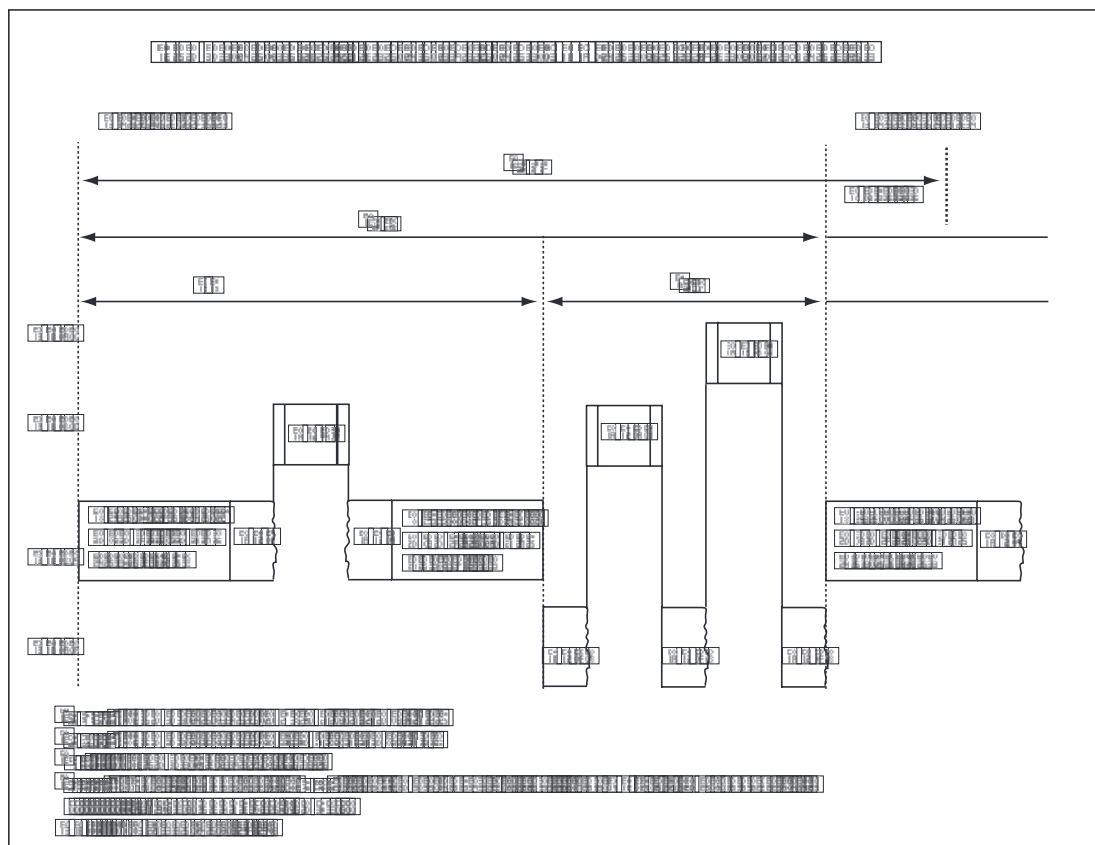
STEP 7 vous permet de modifier le temps de surveillance du cycle pris par défaut. A l'expiration de ce temps, soit la CPU passe à l'état de fonctionnement "Arrêt", soit l'OB80 dans lequel vous pouvez définir comment la CPU doit réagir à cette erreur de temps est appelé.

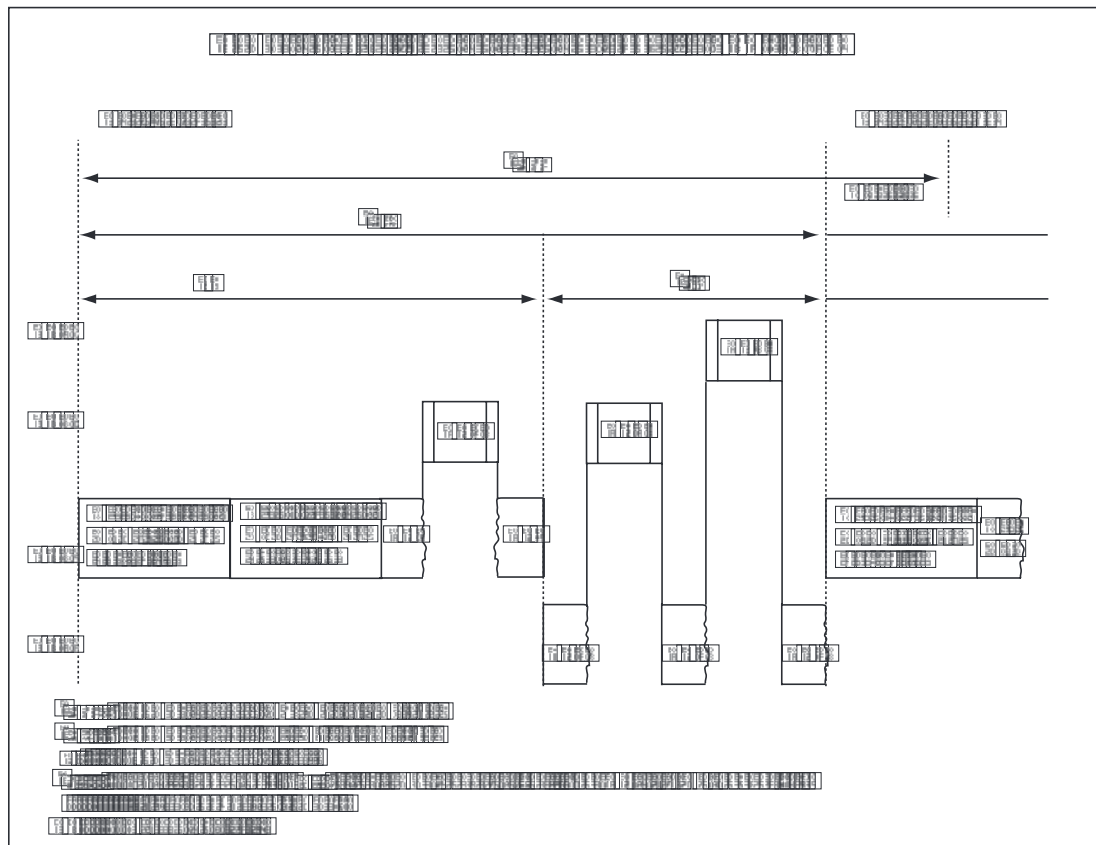
### Temps de cycle minimal

STEP 7 vous permet de définir un temps de cycle minimal pour les CPU S7-400 et pour la CPU 318. Ceci est recommandé :

- si l'intervalle de temps séparant deux exécutions de l'OB1 (cycle libre) doit rester constant
- afin d'éviter une actualisation trop fréquente des mémoires image lorsque le temps de cycle est très court.

Les figures suivantes illustrent la fonction du temps de surveillance du cycle dans le déroulement du programme pour les CPU jusqu'à 10/98 et les CPU à partir de 10/98.





### Mise à jour de la mémoire image

La mémoire image du processus est automatiquement mise à jour lors du traitement de programme cyclique de la CPU. Vous pouvez désactiver cette mise à jour pour les CPU S7-400 et pour la CPU 318 :

- si vous voulez au lieu de cela accéder directement à la périphérie ou
- si vous voulez actualiser une ou plusieurs mémoires image des entrées et des sorties à un autre moment à l'aide des fonctions système SFC26 UPDAT\_PI et SFC27 UPDAT\_PO.

### Charge du cycle due à la communication

Le paramètre de CPU "Charge du cycle due à la communication" vous permet de commander dans une certaine mesure la durée des processus de communication, qui allongent toujours le temps de cycle. On appelle processus de communication, par exemple, la transmission de données à une autre CPU via MPI ou le chargement de blocs déclenché via PG.

Ce paramètre n'a presque pas d'influence sur les fonctions de test avec la PG qui peuvent pourtant allonger considérablement le temps de cycle. C'est dans le mode processus qu'on peut limiter le temps disponible pour les fonctions de test (seulement avec S7-300).

## Comment le paramètre agit-il ?

Le système d'exploitation de la CPU met continuellement à la disposition de la communication le pourcentage configuré de la puissance de traitement totale de la CPU (technique des tranches de temps). Lorsque cette puissance de traitement n'est pas nécessaire à la communication, elle est disponible pour le reste du traitement.

## Effet sur le temps de cycle réel

Sans événements asynchrones supplémentaires, le temps de cycle de l'OB1 s'allonge d'un facteur calculable par la formule suivante :

$$\frac{100}{100 - \text{"Charge du cycle due à la communication (\%)\"}}$$

Exemple 1 (pas d'événements asynchrones en plus)

Une charge du cycle par la communication fixée à 50 % peut doubler le temps de cycle de l'OB1.

En même temps, le temps de cycle de l'OB1 est influencé aussi par des événements asynchrones (tels qu'alarmes de processus ou alarmes cycliques). Le temps de cycle étant allongé par la partie réservée à la communication, il se produira - statistiquement parlant - plus d'événements asynchrones dans un cycle d'OB1, ce qui allonge encore ce dernier. Cet allongement dépend du nombre d'événements survenant par cycle d'OB1 et de la durée de traitement d'un événement.

Exemple 2 (compte tenu des événements asynchrones supplémentaires)

Une durée d'exécution de l'OB1 de 500 ms avec une charge de communication de 50% peut donner un temps de cycle réel allant jusqu'à 1000 ms (à condition que la CPU ait toujours assez de tâches de communication à traiter). Une alarme cyclique intervenant toutes les 100 ms avec une durée d'exécution de 20 ms allongerait de  $5 \times 20 \text{ ms} = 100 \text{ ms}$  au total un cycle sans charge de communication, ce qui donnerait un temps de cycle réel de 600 ms. Puisqu'une alarme cyclique interrompt aussi la communication, elle allongera le temps de cycle de  $10 \times 20 \text{ ms}$  avec une charge de communication de 50%, C'est-à-dire que le temps de cycle réel sera dans ce cas non pas de 1000 ms mais de 1200 ms.

---

### Nota

- Lorsque vous modifiez la valeur attribuée au paramètre "Charge du cycle due à la communication", vérifiez-en les effets dans le fonctionnement de l'installation.
  - Tenez compte de la charge due à la communication lorsque vous fixez le temps de cycle minimal, pour éviter les erreurs de temps.
-

## Recommandations

- Autant que possible, adoptez la valeur par défaut.
- Augmentez cette valeur seulement si la CPU est employée surtout à des fins de communication et que le programme utilisateur n'est pas à durée critique.
- Dans tous les autres cas, bornez-vous à réduire cette valeur !
- Passez en mode processus (seulement avec S7-300) et limitez le temps requis à cet endroit pour les fonctions de test.

### 4.2.3.2 Fonctions (FC)

Les fonctions font partie des blocs que vous programmez vous-même. Une fonction est un bloc de code sans mémoire. Les variables temporaires d'une fonction sont sauvegardées dans la pile des données locales. Ces données sont perdues à l'achèvement de la fonction. Les fonctions peuvent faire appel à des blocs de données globaux pour la sauvegarde des données.

Comme une fonction ne dispose pas de mémoire associée, vous devez toujours indiquer des paramètres effectifs pour elle. Vous ne pouvez pas affecter de valeur initiale aux données locales d'une FC.

### Domaine d'application

Une fonction contient un programme qui est exécuté quand cette fonction est appelée par un autre bloc de code. Vous pouvez faire appel à des fonctions pour :

- renvoyer une valeur de fonction au bloc appelant (exemple : fonctions mathématiques),
- exécuter une fonction technologique (exemple : commande individuelle avec combinaison binaire).

### Affectation de paramètres effectifs aux paramètres formels

Un paramètre formel sert de paramètre générique au paramètre "réel", le paramètre effectif. Les paramètres effectifs remplacent les paramètres formels lors de l'appel d'une FC. Vous devez toujours affecter des paramètres effectifs aux paramètres formels d'une FC (par exemple, le paramètre effectif "E3.6" au paramètre formel "Démarrage"). Les paramètres d'entrée, de sortie et d'entrée/sortie utilisés par la FC sont sauvegardés comme pointeurs désignant les paramètres effectifs du bloc de code qui a appelé la fonction.

## Différence importante entre les paramètres de sortie des FC et des FB

Dans les blocs fonctionnels (FB), la copie du paramètre actuel figurant dans le DB d'instance est utilisée lors de l'accès aux paramètres. Si un paramètre d'entrée n'est pas transmis ou si un paramètre de sortie n'est pas affecté dans le bloc lors de l'appel d'un FB, ce sont les anciennes valeurs encore disponibles dans le DB d'instance qui sont utilisées (DB d'instance = mémoire du FB).

Les fonctions (FC) ne disposent pas de mémoire (FC). Contrairement au FB, l'affectation des paramètres formels n'est de ce fait pas optionnelle, mais indispensable. L'accès aux paramètres de la FC s'effectue via des adresses (pointeur interzone). Si un opérande de la zone de données (bloc de données) ou une variable locale du bloc appelant sont utilisés comme paramètre actuel, une copie de ce paramètre actuel est enregistrée temporairement dans les données locales du bloc appelant, lors de la transmission de paramètres.

---

### Important

Si dans un tel cas, un paramètre OUTPUT n'est pas affecté dans une FC, les valeurs fournies risquent d'être aléatoires !

La zone mise à disposition dans les données locales du bloc appelant pour la copie n'est pas affectée en raison de l'absence d'affectation au paramètre OUTPUT et reste inchangée. C'est la valeur encore inscrite dans cette zone qui sera fournie, puisque les données locales ne disposent pas automatiquement d'une valeur par défaut, p. ex. 0.

---

Tenez compte des points suivants :

- Si possible, initialisez le paramètre OUTPUT.
- Les instructions de mise à 1 et de remise à 0 dépendent du résultat logique. Si ces instructions déterminent la valeur d'un paramètre OUTPUT, aucune valeur n'est fournie lorsque RLG = 0.
- Veillez à ce que les paramètres OUTPUT soient affectés dans tous les cas — quels que soient les chemins de programme possibles dans le bloc. En particulier, tenez compte des instructions de saut et de la sortie ENO dans CONT et LOG. N'oubliez pas non plus BEB et l'effet des instructions MCR.

---

### Nota

Pour les paramètres OUTPUT d'un FB ou les paramètres INOUT d'une FC et d'un FB, les valeurs fournies ne sont pas aléatoires, car en l'absence d'affectation du paramètre, c'est l'ancienne valeur de sortie ou l'ancienne valeur d'entrée qui restent conservées comme valeur de sortie. Dans ce cas, il est également recommandé de tenir compte des remarques précédentes, afin de ne pas continuer à utiliser les "anciennes" valeurs.

---

### 4.2.3.3 Blocs fonctionnels (FB)

Les blocs fonctionnels font partie des blocs que vous programmez vous-même. Un bloc fonctionnel est un bloc avec rémanence. Un bloc de données d'instance lui est associé qui en constitue la mémoire. Les paramètres transmis au FB ainsi que les variables statiques sont sauvegardés dans le bloc de données d'instance. Les variables temporaires sont rangées dans la pile des données locales.

Les données sauvegardées dans le bloc de données d'instance ne sont pas perdues à l'achèvement du traitement du FB. En revanche, les données sauvegardées dans la pile des données locales le sont.

#### Nota

Afin d'éviter des erreurs d'utilisation de FB, veuillez lire le paragraphe Types de données autorisés pour la transmission de paramètres en annexe.

### Domaine d'application

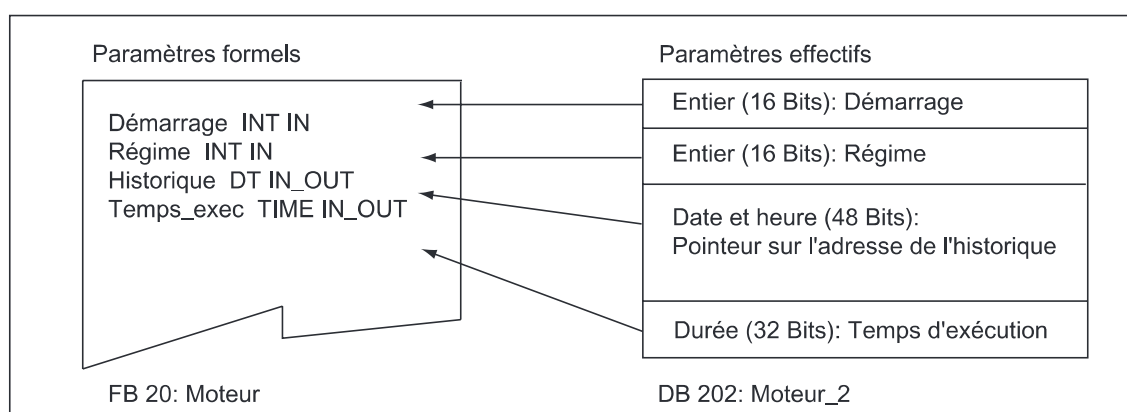
Un bloc fonctionnel contient un programme qui est exécuté quand ce bloc fonctionnel est appelé par un autre bloc de code. Les blocs fonctionnels facilitent la programmation de fonctions complexes souvent utilisées.

### FB et DB d'instance

Un bloc de données d'instance est associé à chaque appel de bloc fonctionnel transmettant des paramètres.

En appelant plusieurs instances d'un FB, vous pouvez piloter plusieurs appareils avec un seul bloc fonctionnel. Un FB pour un type de moteur peut, par exemple, commander différents moteurs en utilisant des données d'instance différentes pour les différents moteurs. Il est possible de ranger les données pour chaque moteur (régime, accélération, cumul des temps de fonctionnement, etc.) dans un ou plusieurs DB d'instance.

La figure suivante montre les paramètres formels d'un FB qui utilise les paramètres effectifs sauvegardés dans le DB d'instance.



### Variables de type de données FB

Si votre programme utilisateur est organisé de telle manière que, dans un FB, soient appelés d'autres blocs fonctionnels existant déjà, vous pouvez déclarer les FB appelés comme variables statiques de type de données FB dans la table de déclaration des variables du FB appelant. Vous obtenez ainsi une imbrication des variables et la concentration des données d'instance dans un bloc de données d'instance (multi-instance).

### Affectation de paramètres effectifs aux paramètres formels

Il n'est, en général, pas obligatoire dans STEP 7 d'affecter des paramètres effectifs aux paramètres formels d'un FB. Des paramètres effectifs doivent toutefois être affectés :

- pour un paramètre d'entrée/sortie de type de données complexe (par exemple, STRING, ARRAY ou DATE\_AND\_TIME),
- pour tous les types de paramètre (par exemple, TIMER, COUNTER ou POINTER).

STEP 7 associe les paramètres effectifs aux paramètres formels de la manière suivante :

- Lorsque vous indiquez des paramètres effectifs dans l'instruction d'appel, les opérations du FB utilisent les paramètres effectifs ainsi mis à disposition.
- Lorsque vous n'indiquez pas de paramètres effectifs dans l'instruction d'appel, les opérations du FB utilisent les valeurs contenues dans le DB d'instance.

Le tableau ci-après montre à quelles variables du FB il faut affecter des paramètres effectifs.

Variables		Type de données	
	Type de données simple	Type de données complexe	Type de données paramètre
Entrée	Paramètres facultatifs	Paramètres facultatifs	Paramètres effectifs obligatoires
Sortie	Paramètres facultatifs	Paramètres facultatifs	Paramètres effectifs obligatoires
Entrée/sortie	Paramètres facultatifs	Paramètres effectifs obligatoires	–



### Affectation de valeurs initiales aux paramètres formels

Vous pouvez préciser des valeurs initiales pour les paramètres formels dans la section de déclaration du FB. Ces valeurs sont reprises dans le bloc de données d'instance associé au FB.

Si vous n'affectez pas de paramètres effectifs aux paramètres formels dans l'instruction d'appel, STEP 7 utilise les valeurs sauvegardées dans le DB d'instance. Il peut alors s'agir de valeurs initiales que vous avez saisies dans la table de déclaration des variables du FB.

Le tableau suivant présente les variables pour lesquelles vous pouvez indiquer une valeur initiale. Comme les données temporaires ne sont pas sauvegardées après le traitement du bloc, vous ne pouvez pas leur affecter de valeur.

	Type de données		
Variables	Type de données simple	Type de données complexe	Type de données paramètre
Entrée	Valeur initiale autorisée	Valeur initiale autorisée	–
Sortie	Valeur initiale autorisée	Valeur initiale autorisée	–
Entrée/sortie	Valeur initiale autorisée	–	–
Statique	Valeur initiale autorisée	Valeur initiale autorisée	–
Temporaire	–	–	–

#### 4.2.3.4 Blocs de données d'instance

Un bloc de données d'instance est associé à chaque appel de bloc fonctionnel transmettant des paramètres. Ce bloc de données d'instance contient les paramètres effectifs et les données statiques du FB. Les variables déclarées dans le FB déterminent la structure du bloc de données d'instance. On appelle instance l'appel d'un bloc fonctionnel. Si, par exemple, un bloc fonctionnel est appelé cinq fois dans le programme utilisateur S7, il existe cinq instances de ce bloc.

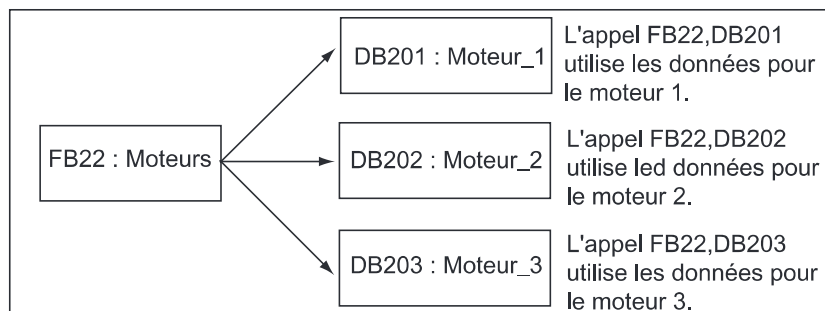
#### Création d'un DB d'instance

Le bloc fonctionnel correspondant à un DB d'instance doit exister avant que vous ne créiez ce DB d'instance. Vous indiquez le numéro de ce FB lors de la création du bloc de données d'instance.

#### Un DB d'instance pour chaque instance

Si vous associez plusieurs blocs de données d'instance à un bloc fonctionnel commandant un moteur, vous pourrez utiliser ce FB pour piloter différents moteurs.

Vous rangez les différentes données pour chaque moteur (comme, par exemple, régime, temps d'accélération, durée totale de fonctionnement) dans les différents blocs de données. Selon le DB associé au FB lors de l'appel, un autre moteur est commandé. Ainsi, un seul bloc fonctionnel est nécessaire pour plusieurs moteurs (voir la figure ci-après).

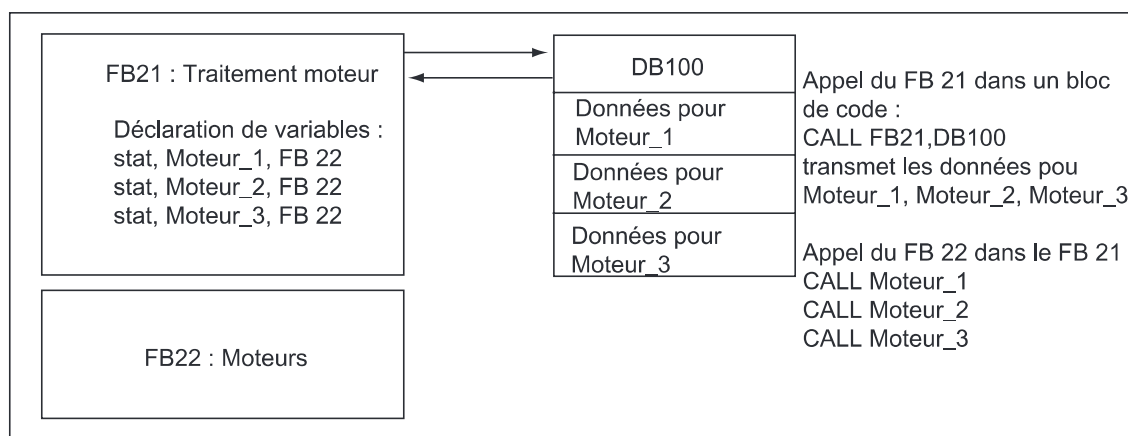


### Un DB d'instance pour plusieurs instances d'un FB (multi-instances)

Vous pouvez transmettre à un bloc fonctionnel les données d'instance pour différents moteurs dans le même DB d'instance. Pour ce faire, vous devez appeler les commandes de moteur dans un autre FB dans la section de déclaration duquel vous déclarez des variables statiques de type de données FB pour les différentes instances.

Utiliser un seul DB d'instance pour plusieurs instances d'un FB vous permet de gagner de l'espace mémoire et d'optimiser l'utilisation des blocs de données.

Dans la figure suivante par exemple, le FB appelant est le FB21 "Traitement moteur", les variables sont de type de données FB22 et les instances sont identifiées par Moteur\_1, Moteur\_2 et Moteur\_3.



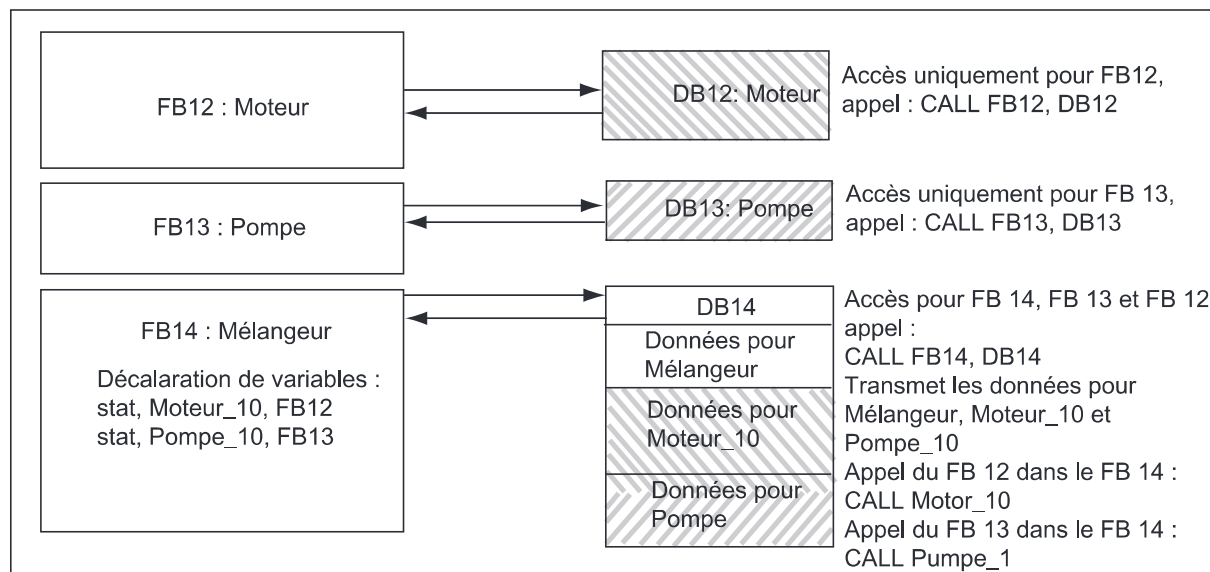
Le FB22 ne requiert pas de DB d'instance en propre dans cet exemple, car ses données d'instance sont contenues dans le DB d'instance du FB appelant.

### Un DB d'instance pour plusieurs instances de différents FB (multi-instances)

Vous pouvez appeler, dans un bloc fonctionnel, des instances d'autres FB déjà créés. Vous pouvez associer les données d'instance nécessaires pour cela au bloc de données d'instance du FB appelant et n'avez ainsi pas besoin de blocs de données supplémentaires pour les FB appelés.

Pour ces multi-instances dans un DB d'instance, vous devez déclarer, dans la section de déclaration du FB appelant, des variables statiques avec le type de données du FB appelé pour les différentes instances. L'appel à l'intérieur du FB se fait alors sans indication de DB d'instance, mais uniquement via le nom de la variable.

Dans l'exemple de la figure, les données d'instance associées sont sauvegardées ensemble dans un seul DB d'instance.



#### 4.2.3.5 Blocs de données globaux (DB)

Contrairement aux blocs de code, les blocs de données ne contiennent pas d'instructions STEP 7. Ils servent à l'enregistrement de données utilisateur : ils contiennent des données variables que le programme utilisateur utilise. Les blocs de données globaux servent à l'enregistrement de données utilisateur pouvant être utilisées par tous les autres blocs.

La taille des DB peut varier. Vous trouverez la taille maximale autorisée dans les descriptions de CPU /70/ et /101/.

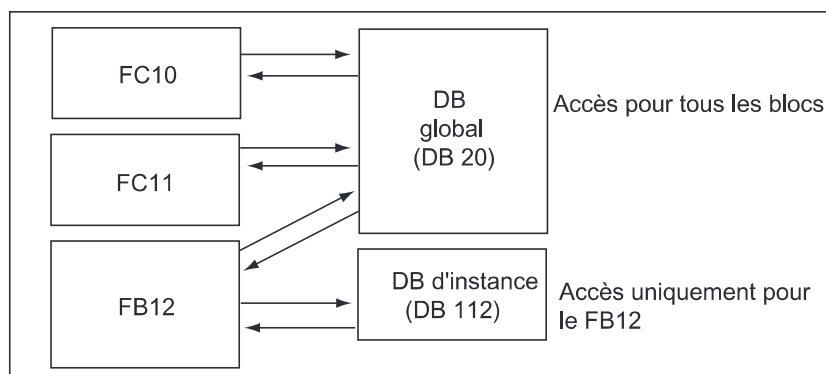
C'est vous qui définissez l'organisation des blocs de données globaux.

#### DB globaux dans le programme utilisateur

Lorsqu'il est appelé, un bloc de code (FC, FB ou OB) peut occuper temporairement de l'espace mémoire dans la zone des données locales (pile L). En plus de cette zone de données locales, ce bloc de code peut ouvrir une autre zone de mémoire sous la forme d'un DB. Contrairement aux données dans la zone des données locales, les données du DB ne sont pas effacées à la fermeture du DB ou à la fin du traitement du bloc de code correspondant.

Tout FB, FC ou OB peut lire les données contenues dans un DB global ou écrire des données dans un DB global. Ces données sont conservées dans le blocs de données même lorsqu'on quitte le DB.

Il est possible d'ouvrir simultanément un DB global et un DB d'instance. La figure ci-après présente les différents accès aux blocs de données.



#### 4.2.3.6 Blocs fonctionnels système (SFB) et fonctions système (SFC)

##### Blocs déjà programmés

Il n'est pas nécessaire que vous programmiez vous-même chaque fonction. En effet, les CPU S7 vous proposent des blocs tout prêts que vous pouvez appeler à partir du programme utilisateur.

De plus amples informations à ce sujet sont données dans l'aide de référence sur les blocs système et fonctions système (voir Sauts dans les descriptions de langage, aides sur les blocs, attributs système).

##### Blocs fonctionnels système

Un bloc fonctionnel système (SFB) est un bloc fonctionnel intégré à la CPU S7. Comme les SFB font partie du système d'exploitation, ils ne sont pas chargés en tant que partie du programme. Comme les FB, les SFB sont des blocs avec mémoire. Vous devez donc également créer pour les SFB des blocs de données d'instance que vous chargez dans la CPU en tant que partie du programme.

Les CPU S7 proposent des SFB :

- pour la communication via des liaisons configurées,
- pour des fonctions spéciales intégrées (par exemple, SFB29 HS\_COUNT dans la CPU 312 IFM et la CPU 314 IFM).

##### Fonctions système

Une fonction système (SFC) est une fonction préprogrammée et intégrée dans la CPU S7. Vous pouvez appeler les SFC à partir de votre programme. Comme ces fonctions font partie du système d'exploitation, elles ne sont pas chargées en tant que partie du programme. Comme les FC, les SFC constituent des blocs sans mémoire.

Les CPU S7 proposent des fonctions système pour :

- des fonctions de copie et de blocs,
- le contrôle du programme,
- la gestion de l'horloge et du compteur d'heures de fonctionnement,
- le transfert d'enregistrements logiques,
- le transfert, en mode de fonctionnement multiprocesseur, d'événements d'une CPU à toutes les CPU enfichées,
- la gestion des alarmes horaires et temporisées,
- la gestion des événements d'erreur synchrone, des événements d'alarme et des événements d'erreur asynchrone,
- l'information sur les données système statiques et dynamiques, p. ex. le diagnostic,
- la mise à jour de la mémoire image du processus et le traitement de champ binaire,
- l'adressage de modules,
- la périphérie décentralisée,

- la communication par données globales,
- la communication via des liaisons non configurées,
- la création de messages relatifs aux blocs.

### **Informations supplémentaires**

De plus amples informations sur les SFB et SFC sont données dans le manuel de référence "Logiciel système pour SIMATIC S7-300/400 - Fonctions standard et fonctions système". Les SFB et SFC disponibles sont précisés dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" ou le manuel de référence "Systèmes d'automatisation S7-400, M7-400 - Caractéristiques des modules".

#### 4.2.4 Blocs d'organisation pour le traitement de programme déclenché par alarme

En mettant à votre disposition des OB d'alarme, les CPU S7 vous donnent la possibilité :

- de déclencher le traitement de parties de programme par horloge,
- de réagir de manière optimale aux signaux externes du processus.

Il est inutile que le programme utilisateur cyclique teste constamment si des événements d'alarme sont apparus. En effet, en cas d'alarme, le système d'exploitation fait en sorte que soit traitée la partie du programme utilisateur figurant dans l'OB d'alarme et qui détermine comment l'automate programmable doit réagir à cette alarme.

#### Types d'alarme et applications

Le tableau suivant montre comment utiliser les types d'alarme.

Type d'alarme	OB d'alarme	Exemples d'application
Alarme horaire	OB10 à OB17	Calcul du débit d'un processus de mélange à la fin de la journée de travail
Alarme temporisée	OB20 à OB23	Commande d'un ventilateur devant fonctionner encore 20 s après l'arrêt d'un moteur avant d'être lui-même arrêté
Alarme cyclique	OB30 à OB38	Echantillonnage d'un niveau de signal pour une installation de régulation
Alarme de processus	OB40 à OB47	Signaler que le niveau maximal d'une cuve est atteint



#### 4.2.4.1 Blocs d'organisation pour l'alarme horaire (OB10 à OB17)

Les CPU S7 mettent à votre disposition des OB d'alarme horaire pouvant être traités à une date donnée ou à des intervalles de temps définis.

Les alarmes horaires peuvent être déclenchées :

- une seule fois à un moment donné (indication de temps absolue avec date),
- périodiquement avec indication du commencement et de la fréquence de répétition (par exemple, toutes les minutes, toutes les heures, tous les jours).

#### Règles d'utilisation des alarmes horaires

Les alarmes horaires ne peuvent être traitées que si une alarme horaire a été paramétrée et qu'un bloc d'organisation correspondant est contenu dans le programme utilisateur. Si tel n'est pas le cas, un message d'erreur est inscrit dans la mémoire tampon de diagnostic et un traitement d'erreur asynchrone est exécuté (OB80, voir "Blocs d'organisation pour le traitement d'erreur (OB70 à OB87 / OB121 à OB122)").

Les alarmes horaires périodiques doivent correspondre à une date réelle. Il n'est, par exemple, pas possible de réitérer chaque mois un OB10 ayant le 31 janvier comme point de départ. Dans ce cas, l'OB ne serait déclenché que pour les mois ayant effectivement 31 jours (pas en février, avril, juin, etc.).

Une alarme horaire activée au cours de la mise en route (démarrage à chaud ou redémarrage) n'est traitée qu'à la fin de la mise en route.

Il n'est pas possible de déclencher les OB d'alarme horaire désactivés par paramétrage. La CPU détecte dans ce cas une erreur de programmation et passe à l'état de fonctionnement "Arrêt" (STOP).

Après un démarrage à chaud, il faut à nouveau activer les alarmes horaires générées, par exemple à l'aide de la SFC30 ACT\_TINT dans le programme de mise en route.

#### Déclenchement de l'alarme horaire

La CPU ne peut déclencher une alarme horaire qu'une fois que vous avez généré puis activé cette dernière. Il existe trois types de déclenchement :

- déclenchement automatique de l'alarme horaire par paramétrage avec STEP 7 (bloc de paramètres "Alarmes horaires"),
- génération et activation de l'alarme horaire via la SFC28 SET\_TINT et la SFC30 ACT\_TINT à partir du programme utilisateur,
- génération de l'alarme horaire par paramétrage avec STEP 7 et activation de l'alarme horaire via la SFC30 ACT\_TINT à partir du programme utilisateur.

#### Interrogation de l'alarme horaire

Pour savoir si des alarmes horaires ont été générées et à quel moment, vous pouvez :

- appeler la SFC31 QRY\_TINT
- ou demander la liste partielle "Etat d'alarme" de la liste d'état système.

### Désactivation de l'alarme horaire

Vous pouvez désactiver des alarmes horaires non encore traitées à l'aide de la SFC29 CAN\_TINT. Il est possible de générer à nouveau des alarmes horaires désactivées via la SFC28 SET\_TINT et de les activer avec la SFC30 ACT\_TINT.

### Priorité des OB d'alarme horaire

Les huit OB d'alarme horaire sont prédéfinis avec la même classe de priorité (2) et sont donc traités dans l'ordre d'apparition de leurs événements déclencheurs. Il est possible de modifier la classe de priorité par paramétrage.

### Changement de l'heure réglée

Il est possible de modifier l'heure réglée comme suit :

- Un maître d'heure synchronise l'heure pour le maître et les esclaves.
- L'heure est redéfinie dans le programme utilisateur via la SFC0 SET\_CLK.

### Comportement en cas de changement d'heure

Le tableau suivant montre comment les alarmes horaires se comportent après modification de l'heure.

Si	alors
une ou plusieurs alarmes horaires ont été sautées en raison de l'avancement de l'heure,	l'OB80 est démarré avec inscription dans ses informations de déclenchement des alarmes horaires sautées.
vous avez désactivé dans l'OB80 les alarmes horaires sautées,	les alarmes horaires sautées ne sont pas rattrapées.
vous n'avez pas désactivé dans l'OB80 les alarmes horaires sautées,	la première alarme horaire sautée est reprise, mais il n'est pas tenu compte des suivantes.
des alarmes horaires déjà traitées sont à nouveau en attente en raison du retardement de l'heure,	ces alarmes horaires sont traitées une nouvelle fois <b>dans le cas des CPU S7-300</b> ne sont pas traitées une nouvelle fois <b>dans le cas des CPU S7-400 et de la CPU 318.</b>

#### 4.2.4.2 Blocs d'organisation pour l'alarme temporisée (OB20 à OB23)

Les CPU S7 mettent à votre disposition des OB d'alarme temporisée grâce auxquels vous pouvez programmer l'exécution retardée de certaines parties de votre programme utilisateur.

##### Règles d'utilisation des alarmes temporisées

Les alarmes temporisées ne peuvent être traitées que si un bloc d'organisation correspondant est contenu dans le programme CPU. Si tel n'est pas le cas, un message d'erreur est inscrit dans la mémoire tampon de diagnostic et un traitement d'erreur asynchrone est réalisé (OB80, voir Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)).

Il n'est pas possible de déclencher les OB d'alarme temporisée désactivés par paramétrage. La CPU détecte dans ce cas une erreur de programmation et passe à l'état de fonctionnement "Arrêt" (STOP).

Les alarmes temporisées sont déclenchées lorsque le temps de retard précisé dans la SFC32 SRT\_DINT a expiré.

##### Déclenchement de l'alarme temporisée

Pour démarrer une alarme temporisée, vous devez fixer dans la SFC32 le temps de retard à l'expiration duquel l'OB d'alarme temporisée correspondant doit être appelé. La durée maximale autorisée pour le temps de retard est donnée dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" et le manuel de référence "Systèmes d'automatisation S7-400, M7-400 - Caractéristiques des modules".

##### Priorité des OB d'alarme temporisée

Par défaut, les OB d'alarme temporisée ont les classes de priorité 3 à 6. Vous pouvez les modifier par paramétrage.

#### 4.2.4.3 Blocs d'organisation pour l'alarme cyclique (OB30 à OB38)

Les CPU S7 mettent à votre disposition des OB d'alarme cyclique qui interrompent le traitement de programme cyclique à intervalles précis.

Les alarmes cycliques sont déclenchées à des intervalles de temps précis. Le moment de déclenchement de la période est le passage de l'état de fonctionnement "Arrêt" (STOP) à l'état "Marche" (RUN).

#### Règles d'utilisation des alarmes cycliques

Veillez, lorsque vous choisissez la période, à ce qu'il reste suffisamment de temps pour le traitement des alarmes cycliques entre les événements de déclenchement des différentes alarmes cycliques.

Il n'est pas possible de déclencher les OB d'alarme cyclique désactivés par paramétrage. La CPU détecte dans ce cas une erreur de programmation et passe à l'état de fonctionnement "Arrêt" (STOP).

#### Déclenchement de l'alarme cyclique

Pour déclencher une alarme cyclique, vous devez préciser via STEP 7 une période dans le bloc de paramètres "Alarmes cycliques". Cette période est toujours un multiple entier de la période de base de 1 ms.

Période = n ? période de base 1 ms

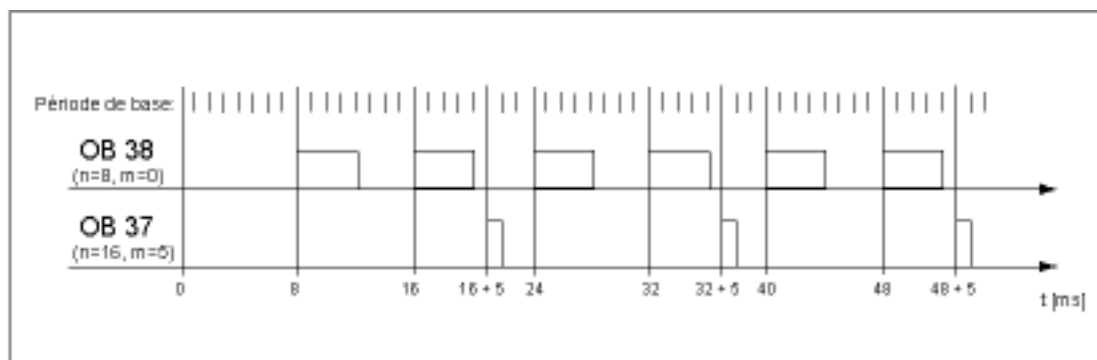
Les neuf OB d'alarme cyclique disponibles ont des périodes prédéfinies (voir tableau suivant). La période par défaut entre en vigueur lorsque l'OB d'alarme cyclique qui lui est associé est chargé. Vous pouvez toutefois modifier par paramétrage les valeurs prédéfinies. La limite supérieure est donnée dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" et le manuel de référence "Systèmes d'automatisation S7-400, M7-400 - Caractéristiques des modules.

#### Décalage de phase pour les alarmes cycliques

Afin d'éviter que les alarmes cycliques de différents OB d'alarme cyclique ne reçoivent une demande de déclenchement au même moment provoquant ainsi éventuellement une erreur de temps (dépassement du temps de cycle), vous pouvez préciser un décalage de phase. Ce décalage de phase assure que le traitement d'une alarme cyclique est décalé d'une durée donnée après écoulement de la période.

Décalage de phase = m ? période de base (avec  $0 \leq m < n$ )

La figure suivante montre le traitement d'un OB d'alarme cyclique avec décalage de phase (OB37) comparé à une alarme cyclique sans décalage de phase (OB38).



### Priorité des OB d'alarme cyclique

Le tableau suivant montre les périodes et classes de priorité prédéfinies des OB d'alarme cyclique. Vous pouvez modifier les périodes et classes de priorité par paramétrage.

OB d'alarme cyclique	Période en ms	Classe de priorité
OB30	5000	7
OB31	2000	8
OB32	1000	9
OB33	500	10
OB34	200	11
OB35	100	12
OB36	50	13
OB37	20	14
OB38	10	15

#### 4.2.4.4 Blocs d'organisation pour l'alarme de processus (OB40 à OB47)

Les CPU S7 mettent à votre disposition des OB d'alarme de processus qui réagissent à des signaux provenant des modules (modules de signaux SM, processeurs de communication CP, modules de fonction FM). STEP 7 vous permet de définir quel signal doit déclencher l'OB pour les modules TOR et analogiques paramétrables. Pour les CP et les FM, vous utiliserez à cet effet les dialogues de paramétrage correspondants.

Les alarmes de processus sont déclenchées lorsqu'un module de signaux pouvant générer des alarmes de processus, avec validation d'alarme de processus paramétrée, transmet un signal de processus reçu à la CPU ou lorsqu'un module de fonction de la CPU signale une alarme.

#### Règles d'utilisation d'alarmes du processus

Les alarmes de processus ne peuvent être traitées que si un bloc d'organisation correspondant est contenu dans le programme CPU. Si tel n'est pas le cas, un message d'erreur est inscrit dans la mémoire tampon de diagnostic et un traitement d'erreur asynchrone est réalisé (voir Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)).

Il n'est pas possible de déclencher les OB d'alarme de processus désactivés par paramétrage. La CPU détecte dans ce cas une erreur de programmation et passe à l'état de fonctionnement "Arrêt" (STOP).

#### Paramétrage de modules de signaux pouvant générer des alarmes de processus

Chaque voie d'un module de signaux pouvant générer des alarmes de processus peut déclencher une alarme de processus. Aussi devez-vous définir à l'aide de STEP 7 dans les jeux de paramètres de ces modules :

- ce qui doit déclencher une alarme de processus,
- quel OB d'alarme de processus doit être traité (l'OB40 est prévu par défaut pour le traitement de toutes les alarmes de processus).

Vous activez avec STEP 7 la génération d'alarmes de processus des modules de fonction. Vous affectez d'autres paramètres dans les dialogues de paramétrage de ces modules.

#### Priorité des OB d'alarme de processus

Par défaut, les OB d'alarme de processus ont les classes de priorité 16 à 23. Vous pouvez les modifier par paramétrage.

#### 4.2.4.5 Blocs d'organisation pour la mise en route (OB100 / OB101 / OB102)

##### Modes de mise en route

On distingue entre les modes de mise en route suivants :

- redémarrage (n'existe pas pour les S7-300 et S7-400H),
- démarrage à chaud,
- démarrage à froid.

Le tableau suivant indique l'OB respectivement appelé par le système d'exploitation.

Mode de mise en route	OB associé
Redémarrage	OB101
Démarrage à chaud	OB100
Démarrage à froid	OB102

##### Evénements de déclenchement pour les OB de mise en route

La CPU exécute une mise en route

- après mise sous tension
- lorsque vous actionnez le commutateur de mode de fonctionnement à partir de STOP "RUN"/"RUN-P"
- après sollicitation par une fonction de communication
- après synchronisation en mode multiprocesseur
- dans un système H, après couplage (uniquement sur la CPU de réserve)

L'OB de mise en route correspondant (OB100, OB101 ou OB102) est appelé selon l'événement de déclenchement, la CPU mise en oeuvre ainsi que les paramètres sélectionnés pour cette dernière.

##### Programme de mise en route

Vous pouvez déterminer les conditions supplémentaires pour le comportement de mise en route de votre CPU (valeurs d'initialisation pour "Marche", valeurs de mise en route pour les modules de périphérie) en écrivant votre programme de mise en route dans les blocs d'organisation OB100 pour le démarrage à chaud, OB101 pour le redémarrage ou OB102 pour le démarrage à froid.

La longueur du programme de mise en route est indifférente : son exécution n'est pas limitée en durée, car la surveillance du temps de cycle n'est pas active. L'exécution commandée par horloge ou par alarme n'est pas possible dans le programme de mise en route. Pendant cette dernière, toutes les sorties TOR prennent l'état de signal 0.

### **Mode de mise en route après une mise en route manuelle**

Pour les CPU S7-300, seuls les démarrages manuel (démarrages à chaud) ou à froid (uniquement CPU 318-2) sont possibles.

Pour certaines CPU S7-400, vous pouvez exécuter un redémarrage manuel (démarrage à chaud) ou un démarrage à froid avec le commutateur de mode de fonctionnement et le commutateur de mode de mise en route (CRST/WRST) si cela a été paramétré ainsi avec STEP 7. Le démarrage manuel (démarrage à chaud) est possible sans paramétrage.

### **Mode de mise en route après une mise en route automatique**

Seul le démarrage à chaud est possible après la mise sous tension pour les CPU S7-300.

Pour les CPU S7-400, vous pouvez déterminer si une mise en route automatique après mise sous tension entraîne un démarrage (démarrage à chaud) ou un redémarrage.

### **Effacement de la mémoire image**

En cas de redémarrage d'une CPU S7-400, la mémoire image des sorties est effacée par défaut après l'exécution du reste du cycle. Vous pouvez toutefois désactiver l'effacement de la mémoire image si le programme utilisateur doit, après le redémarrage, continuer à utiliser les valeurs en vigueur avant le redémarrage.

### **Contrôle des modules : configuration prévue-configuration sur site**

Vous pouvez demander par paramétrage que soit vérifié, avant la mise en route, si tous les modules figurant dans la table de configuration sont réellement enfichés et si leur type est correct.

Lorsque le contrôle des modules est activé, la mise en route n'est pas exécutée si une différence entre la configuration prévue et la configuration réelle est mise en évidence.

### **Temps de surveillance**

Vous pouvez paramétrer les temps de surveillance suivants pour garantir une mise en route sans erreur de l'automate programmable :

- le temps maximal autorisé pour la transmission des paramètres aux modules,
- le temps maximal autorisé pour le message Prêt des modules après la mise sous tension,
- pour les CPU S7-400, le temps d'interruption maximal pendant lequel un redémarrage est encore autorisé.

La CPU passe à l'état "Arrêt" à l'expiration des temps de surveillance ou seul un démarrage est alors possible.



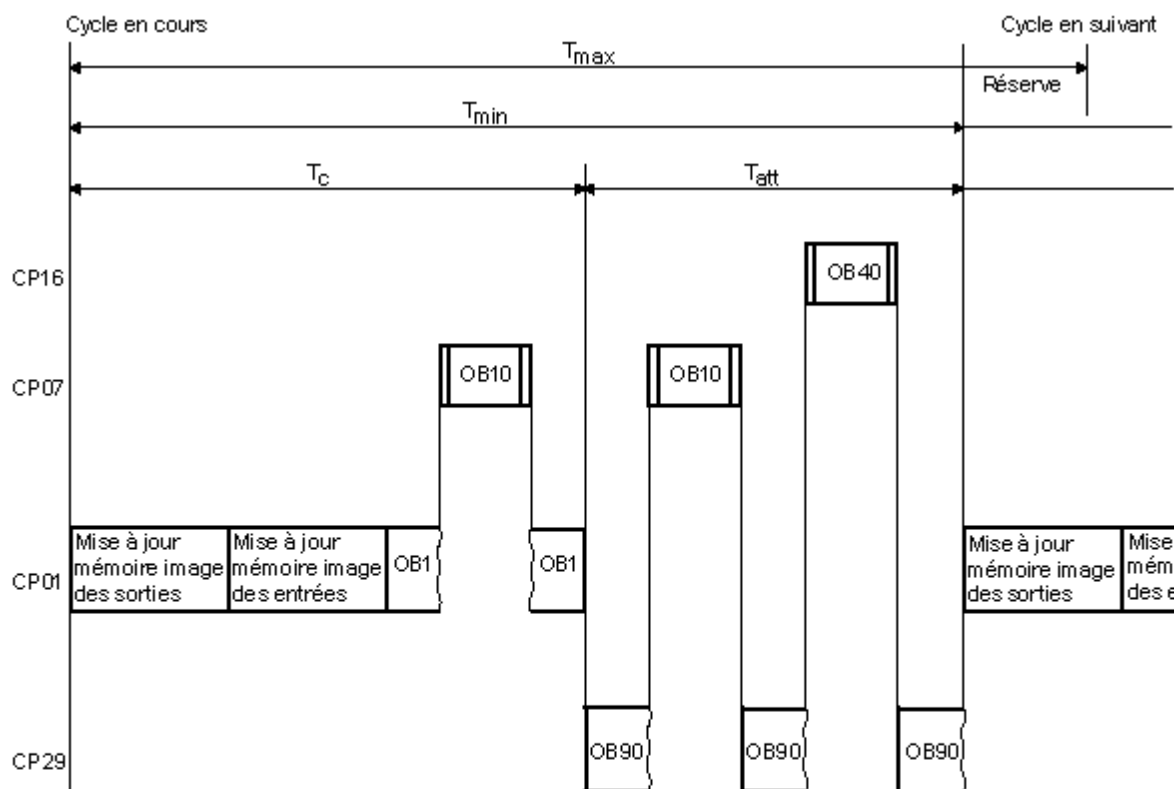
#### 4.2.4.6 Bloc d'organisation pour l'exécution du programme en arrière-plan (OB90)

Si vous avez paramétré un temps de cycle minimal avec STEP 7 et que celui-ci s'avère supérieur au temps de cycle effectif, la CPU dispose de temps d'exécution à la fin du programme cyclique. Ce temps restant sert à l'exécution de l'OB d'arrière-plan. Si l'OB90 n'existe pas dans votre CPU, cette dernière attend que le temps de cycle minimal paramétré soit expiré. L'OB90 vous permet donc d'exécuter des processus à durée non critique et ainsi d'éviter des temps d'attente.

#### Priorité de l'OB d'arrière-plan

L'OB d'arrière-plan a la classe de priorité 29 qui correspond à la priorité 0.29. C'est donc l'OB à la priorité la plus faible et vous ne pouvez pas modifier sa classe de priorité par paramétrage.

La figure suivante montre un exemple d'exécution du cycle en arrière-plan, du cycle libre et de l'OB10 (pour les CPU à partir de 10/98).



- $T_{max}$  = Temps de cycle maximal paramétrable
- $T_{min}$  = Temps de cycle minimal paramétrable
- $T_c$  = Temps de cycle effectif
- $T_{att}$  = La différence entre  $T_{min}$  et le temps de cycle effectif. Les OB d'alarme peuvent être traités pendant ce temps.
- CP = Classe de priorité

## **Programmation de l'OB90**

Le système d'exploitation de la CPU ne contrôle pas le temps d'exécution de l'OB90 ; vous pouvez donc y programmer des boucles de longueur indifférente. Veillez à la cohérence des données que vous utilisez dans le programme en arrière-plan en tenant compte des points suivants lors de leur programmation :

- les événements de remise à zéro de l'OB90 (voir manuel de référence "Logiciel système pour SIMATIC S7-300/400 - Fonctions standard et fonctions système"),
- la mise à jour de la mémoire image du processus qui est asynchrone par rapport à l'OB90.

#### 4.2.4.7 Blocs d'organisation pour le traitement d'erreurs (OB70 à OB87 / OB121 à OB122)

##### Types d'erreur

Les erreurs que les CPU S7 détectent et auxquelles elles peuvent réagir à l'aide de blocs d'organisation sont classables en deux catégories :

- Erreurs synchrones : ces erreurs peuvent être associées à une partie précise du programme utilisateur. L'erreur apparaît pendant le traitement d'une opération précise. Si l'OB d'erreur synchrone correspondant n'est pas chargé, la CPU passe à l'état "Arrêt" (STOP) à l'apparition d'une telle erreur.
- Erreurs asynchrones : ces erreurs ne peuvent pas être directement associées au programme utilisateur traité. Il s'agit d'erreurs de classe de priorité, d'erreurs dans l'automate programmable (par exemple, module défaillant) ou d'erreurs de redondance. Si l'OB d'erreur asynchrone correspondant n'est pas chargé, la CPU passe à l'état "Arrêt" (STOP) à l'apparition d'une telle erreur (exceptions : OB70, OB72, OB81, OB87).

Le tableau ci-après montre les types d'erreur pouvant survenir, classés selon la catégorie des OB d'erreur.

Erreurs asynchrones / erreurs de redondance	Erreurs synchrones
OB70 Erreur de redondance de périphérie (seulement dans les CPU H)	OB121 Erreur de programmation (ex. : DB non chargé)
OB72 Erreur de redondance de CPU (seulement dans les CPU H, ex. : défaillance d'une CPU)	OB122 Erreur d'accès à la périphérie (ex. : accès à un module d'entrées/sorties inexistant)
OB80 Erreur de temps (ex. : dépassement du temps de cycle)	
OB81 Erreur d'alimentation (ex. : pile défaillante)	
OB82 Alarme de diagnostic (ex. : court-circuit dans le module d'entrées)	
OB83 Alarme de débrochage/enfichage (ex. : débrochage d'un module d'entrées)	
OB84 Erreur matérielle CPU (erreur à l'interface avec le réseau MPI)	
OB85 Erreur d'exécution du programme (ex. : OB non chargé)	
OB86 Défaillance d'unité	
OB87 Erreur de communication (ex. : mauvaise ID de télégramme pour communication par données globales)	

### Utilisation des OB pour erreurs synchrones

Les erreurs synchrones apparaissent pendant le traitement d'une opération précise. A l'apparition de ces erreurs, le système d'exploitation génère une entrée dans la pile des interruptions et déclenche l'OB pour erreurs synchrones.

Les OB d'erreur appelés par des erreurs synchrones sont traités en tant que partie du programme avec la même classe de priorité que le bloc en cours d'exécution lors de la détection de l'erreur. Les détails sur l'erreur ayant déclenché l'appel de l'OB sont donnés dans les informations de déclenchement de l'OB. Vous pouvez vous servir de cette information pour réagir à l'erreur, puis reprendre l'exécution de votre programme (par exemple, indiquer une valeur de remplacement dans l'OB122 via la SFC44 RPL\_VAL pour une erreur d'accès à un module d'entrées analogiques). Ainsi, toutefois, la pile L de cette classe de priorité doit en plus prendre en charge les données locales des OB d'erreur.

Dans les CPU S7-400, un nouvel OB d'erreur synchrone peut être lancé à partir d'un OB d'erreur synchrone. Cela n'est pas possible dans les CPU S7-300.

### Utilisation des OB pour erreurs asynchrones

Lorsque le système d'exploitation de la CPU détecte une erreur asynchrone, il déclenche l'OB d'erreur correspondant (OB70 à OB72 et OB80 à OB87). Les OB pour erreurs asynchrones ont la priorité la plus élevée comme défaut : ils ne peuvent pas être interrompus par d'autres OB, lorsque tous les OB d'erreurs synchrones ont la même priorité. Si plusieurs erreurs asynchrones de la même priorité apparaissent simultanément, les OB d'erreur correspondants sont traités dans l'ordre d'apparition des erreurs.

### Masquage d'événements de déclenchement

Des fonctions système vous permettent de masquer, d'ajourner ou d'inhiber les événements de déclenchement pour quelques OB d'erreur. De plus amples informations à ce sujet et sur les blocs d'organisation en particulier sont données dans le manuel de référence "Logiciel système pour SIMATIC S7-300/400 - Fonctions standard et fonctions système".

Type de l'OB d'erreur	SFC	Fonction de la SFC
OB d'erreur synchrone	SFC36 MSK_FLT	Masquer des événements d'erreur synchrone individuels. Les événements d'erreur masqués ne déclenchent aucun OB d'erreur et n'entraînent pas la réaction programmée.
	SFC37 DMSK_FLT	Démasquer des événements d'erreur synchrone
OB d'erreur asynchrone	SFC39 DIS_IRT	Inhiber globalement des événements d'erreur asynchrone et d'alarme. Les événements d'erreur inhibés ne déclenchent d'OB d'erreur dans aucun des cycles de CPU suivants et n'entraînent pas la réaction programmée.
	SFC40 EN_IRT	Valider des événements d'erreur asynchrone et d'alarme
	SFC41 DIS_AIRT	Ajourner les événements d'erreur asynchrone et d'alarme prioritaires jusqu'à la fin de l'OB
	SFC42 EN_AIRT	Valider les événements d'erreur asynchrone et d'alarme prioritaires

#### Nota

Pour ignorer des alarmes, il est plus efficace de les inhiber au moyen de SFC à la mise en route que de charger un OB vide (contenant BE).

# 5 Démarrage et utilisation du programme

## 5.1 Démarrage de STEP 7



Une fois Windows démarré, vous trouverez dans l'interface Windows une icône pour SIMATIC Manager qui permet d'accéder au logiciel STEP 7.

Vous démarrez rapidement STEP 7 en effectuant un double clic sur l'icône "SIMATIC Manager". La fenêtre de SIMATIC Manager s'ouvre alors. De là, vous pouvez accéder à toutes les fonctions que vous avez installées aussi bien du logiciel de base que des logiciels optionnels.

L'autre méthode consiste à lancer SIMATIC Manager via le bouton "Démarrer" dans la barre des tâches du système d'exploitation (sous "Simatic").

---

### Nota

Vous trouverez plus d'informations sur les manipulations et options standard de Windows dans votre guide de l'utilisateur Windows ou dans l'aide en ligne de votre système d'exploitation Windows.

---

## SIMATIC Manager

SIMATIC Manager constitue l'interface d'accès à la configuration et à la programmation. Vous pouvez :

- créer des projets,
- configurer et paramétrer le matériel,
- configurer le fonctionnement en réseau du matériel,
- programmer des blocs,
- tester et mettre en œuvre vos programmes.

L'accès aux fonctions se fonde sur les objets et s'apprend facilement et intuitivement.

Avec SIMATIC Manager, vous pouvez travailler :

- hors ligne, c'est-à-dire sans qu'un automate soit raccordé ou
- en ligne, c'est-à-dire avec un automate raccordé.

Tenez compte, dans ce dernier cas, des remarques relatives à la sécurité.

## Pour poursuivre

Vous créez des solutions d'automatisation sous la forme de "projets". Vous vous faciliterez la tâche en vous familiarisant tout d'abord avec :

- l'interface utilisateur,
- quelques manipulations fondamentales,
- l'aide en ligne.

## 5.2 Démarrage de STEP 7 avec des paramètres initiaux prédéfinis

A partir de STEP 7 V5, vous pouvez créer plusieurs icônes de SIMATIC Manager et indiquer pour chacune d'elles des paramètres initiaux dans la ligne cible. Vous pouvez ainsi faire en sorte que SIMATIC Manager se positionne sur l'objet décrit par ces paramètres. Ceci vous permet de parvenir immédiatement à une position donnée dans un projet, par simple double clic.

En appelant **s7tgotpx.exe**, vous pouvez spécifier les paramètres initiaux suivants :

**/e** <chemin physique complet du projet>

**/o** <chemin logique de l'objet sur lequel se positionner>

**/h** <ObjektID>

**/onl**

Avec le paramètre initial **/onl**, le projet est ouvert en ligne et le chemin indiqué est appelé.

**/off**

Avec le paramètre initial **/off**, le projet est ouvert hors ligne et le chemin indiqué est appelé.

**/keep**

Lorsque SIMATIC Manager est ouvert, le paramètre initial **/keep** ouvre le nouveau projet indiqué dans la ligne de commande en plus des projets déjà affichés. Lorsque SIMATIC Manager n'est pas encore ouvert, le nouveau projet à ouvrir sera ouvert en plus des projets figurant dans la mémoire de session de SIMATIC Manager.

Si ce paramètre initial n'est pas indiqué, les projets ouverts sont d'abord fermés, la mémoire de session ignorée et seul le projet transmis est ouvert.

Voici comment déterminer simplement les paramètres requis.

### Détermination des paramètres par copier/coller

Procédez de la manière suivante :

1. Sur votre bureau, créez un nouveau raccourci pour le fichier s7tgotpx.exe. Ce fichier se trouve dans votre répertoire d'installation, sous S7bin.
2. Affichez la boîte de dialogue des propriétés.
3. Sélectionnez l'onglet "Raccourci". Complétez la zone de saisie "Cible" de la manière suivante :
4. Sélectionnez l'objet souhaité dans SIMATIC Manager.
5. Copiez l'objet sélectionné dans le presse-papiers à l'aide de la combinaison de touches CTRL+ALT+C.
6. Positionnez le curseur à la fin de la zone de saisie "Cible" dans la page d'onglet "Raccourci".
7. Collez le contenu du presse-papiers à l'aide de la combinaison de touches CTRL+V.
8. Quittez la boîte de dialogue par "OK".

### Exemple de saisie de paramètres :

```
/e F:\SIEMENS\STEP7\S7proj\MyConfig\MyConfig.s7p /keep
```

```
/o "1,8:MyConfig\SIMATIC 400(1)\CPU416-1\Programme S7(1)\Blocs\FB1"
```

```
/h T00112001;129;T00116001;1;T00116101;16° /keep
```

### Remarque concernant la structure du chemin du projet

Le chemin du projet correspond au chemin physique dans le système de fichiers.

Le chemin logique complet est structuré de la manière suivante :

- [Identification visible, Identification en ligne]:Nom du projet\{Nom de l'objet\}\* \ Nom de l'objet
- Exemple /o "1,8:MyConfig\SIMATIC 400(1)\CPU416-1\Programme S7(1)\Blocs\FB1"

### Remarque concernant la structure du chemin logique

Le chemin logique complet ainsi que l'ID d'objet ne peuvent être déterminés que par copie/collage. Il est toutefois également possible de spécifier le chemin lisible par l'utilisateur, c'est-à-dire pour l'exemple précédent :

/o "MyConfig\SIMATIC 400(1)\CPU416-1\Programme S7(1)\Blocs\FB1".

Avec les paramètres /onl ou /off, l'utilisateur peut préciser s'il s'agit du chemin pour la fenêtre en ligne ou hors ligne. La saisie de ce paramètre s'avère inutile lorsque vous procédez par copie/collage.

Important : Lorsque le chemin contient des caractères d'espacement, il doit être indiqué entre guillemets.

## 5.3 Appel des fonctions d'aide

### Aide en ligne

L'aide en ligne vous propose des informations à l'endroit où vous en avez besoin. Vous pouvez ainsi aisément trouver des renseignements précis sans devoir consulter des manuels. L'aide en ligne se compose des éléments suivants :

- **Rubriques d'aide** : offre différents accès à l'affichage d'informations d'aide.
- **Aide contextuelle** (touche F1) : fournit des informations sur l'objet sélectionné ou encore sur la boîte de dialogue ou la fenêtre actives.
- **Introduction** : donne un bref aperçu sur l'utilisation, les caractéristiques fondamentales et les fonctions d'une application.
- **Mise en route** : résume les premières opérations que vous devez exécuter pour obtenir votre premier succès.
- **Utiliser l'aide** : décrit les possibilités dont vous disposez pour trouver certaines informations dans l'aide.
- **A propos de** : donne des informations sur la version en cours de l'application.

Le menu d'aide "?" vous permet également d'accéder, à partir de chaque fenêtre, à des rubriques en rapport avec la situation en cours.

### Appel de l'aide en ligne

Vous pouvez appeler l'aide en ligne de différentes manières :

- Choisissez une commande du menu d'aide "?" dans la barre des menus.
- Cliquez sur le bouton "Aide" dans une boîte de dialogue. L'aide correspondant à la boîte de dialogue apparaît alors.
- Dans une fenêtre ou une boîte de dialogue, positionnez le pointeur de la souris sur le thème à propos duquel vous avez besoin d'aide et appuyez sur la touche F1, ou choisissez la commande ? > **Aide contextuelle**.
- Servez-vous du curseur "point d'interrogation" de Windows.

On appelle ces trois dernières catégories l'aide en ligne contextuelle.

### Appel de l'aide abrégée

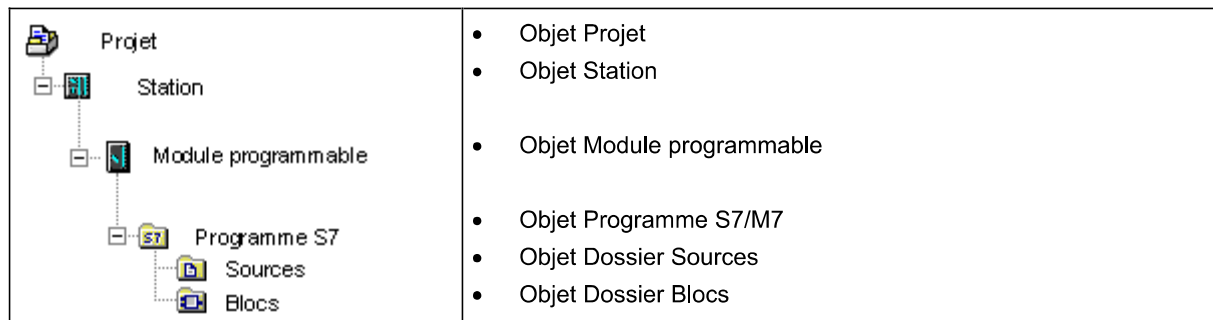
Une aide abrégée s'affiche pour les boutons de la barre d'outils lorsque vous y positionnez le curseur et l'y laissez un court instant.



## 5.4 Objets et hiérarchie d'objets

Dans SIMATIC Manager, la hiérarchie d'objets pour les projets et bibliothèques est similaire à la structure des répertoires comportant des dossiers et fichiers dans l'explorateur de Windows.

La figure suivante donne un exemple de hiérarchie d'objets.



Les objets servent :

- de supports de propriétés,
- de dossiers,
- de supports de fonctions (par exemple, pour le démarrage d'une application précise).

### Objets comme supports de propriétés

Les objets peuvent servir de supports aussi bien pour des fonctions que pour des propriétés (par exemple, paramètres). Une fois un objet sélectionné, vous pouvez :

- l'éditer à l'aide de la commande **Edition > Ouvrir l'objet**.
- ouvrir une boîte de dialogue avec la commande **Edition > Propriétés de l'objet**, dans laquelle vous effectuez les paramétrages propres à l'objet.

Un dossier peut également constituer un support de propriétés.

### Objets comme dossiers

Un dossier peut contenir d'autres dossiers ou des objets. Ceux-ci s'affichent lorsque vous ouvrez le dossier.

### Objets comme supports de fonctions

Lorsque vous ouvrez un objet, une fenêtre dans laquelle vous pouvez traiter l'objet apparaît.


Un objet est soit un dossier, soit un support de fonctions. Les stations constituent toutefois une exception : elles sont à la fois dossiers (pour modules programmables) et supports de fonctions (pour la configuration matérielle).


- Lorsque vous effectuez un double clic sur une station, les objets qu'elle contient sont visualisés, à savoir les modules programmables et la configuration de station (station comme dossier).
- Lorsque vous ouvrez une station avec la commande **Edition > Ouvrir l'objet**, vous pouvez la configurer et la paramétrer (station comme support d'une fonction). Cette commande a la même fonction qu'un double clic sur l'objet "Matériel".





### 5.4.1 Objet Projet

Le projet représente l'ensemble des données et programmes d'une solution d'automatisation et se trouve à la tête d'une hiérarchie d'objets.

#### Position dans la vue du projet


	<ul style="list-style-type: none"> <li>• <b>Objet Projet</b></li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	--


Icône	Dossier d'objets	Sélection de fonctions importantes
	Projet	<ul style="list-style-type: none"> <li>• Création d'un projet</li> <li>• Archivage de projets et de bibliothèques</li> <li>• Gestion multilingue des textes</li> <li>• Recherche des logiciels optionnels requis pour un projet</li> <li>• Impression de la documentation du projet</li> <li>• Réorganisation</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Insertion d'un objet OS</li> <li>• Edition de projets par plusieurs personnes</li> <li>• Conversion d'un ancien projet de version 1</li> <li>• Conversion d'un ancien projet de version 2</li> <li>• Paramétrage de l'interface PG/PC</li> </ul>



Icône	Objets dans le niveau de projet	Sélection de fonctions importantes
	Station : station SIMATIC 300 station SIMATIC 400	<ul style="list-style-type: none"> <li>• Insertion de stations</li> <li>• Les stations sont à la fois des objets (niveau du projet) et des dossiers d'objets (niveau de la station). Pour d'autres fonctions, reportez-vous à Objet Station</li> </ul>
	Programme S7	<ul style="list-style-type: none"> <li>• Programme S7/M7 sans station ni CPU</li> </ul>
	Programme M7	<ul style="list-style-type: none"> <li>• Les programmes S7/M7 sont à la fois des objets (niveau du projet) et des dossiers d'objets (niveau du programme). Pour d'autres fonctions, reportez-vous à Objet Programme S7/M7.</li> </ul>
	Réseau pour le démarrage de l'application de configuration de réseaux et pour la sélection des paramètres de réseau	<ul style="list-style-type: none"> <li>• Propriétés des sous-réseaux et des participants à la communication</li> <li>• Présentation : communication par données globales</li> <li>• Configuration de la communication par données globales</li> </ul>

## 5.4.2 Objet Bibliothèque

Une bibliothèque peut contenir des programmes S7 ou M7 et sert à stocker des blocs. Elle se trouve à la tête d'une hiérarchie d'objets.

	<ul style="list-style-type: none"> <li>• <b>Objet Bibliothèque</b></li> <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	--


Icône	Dossier d'objets	Sélection de fonctions importantes
	Bibliothèque	<ul style="list-style-type: none"> <li>• Présentation des bibliothèques standard</li> <li>• Utilisation de bibliothèques</li> <li>• Archivage de projets et de bibliothèques</li> </ul>




Icône	Objets dans le niveau de la bibliothèque	Sélection de fonctions importantes
	Programme S7	<ul style="list-style-type: none"> <li>• Insertion d'un programme S7/M7</li> <li>• Les programmes S7/M7 sont à la fois des objets (niveau du projet) et des dossiers d'objets (niveau du programme). Pour d'autres fonctions, reportez-vous à Objet Programme S7/M7.</li> </ul>
	Programme M7	



### 5.4.3 Objet Station

Une station SIMATIC 300/400 représente une configuration matérielle S7 comportant un ou plusieurs modules programmables.

#### Position dans la vue du projet

	<ul style="list-style-type: none"> <li>• Objet Projet</li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	---

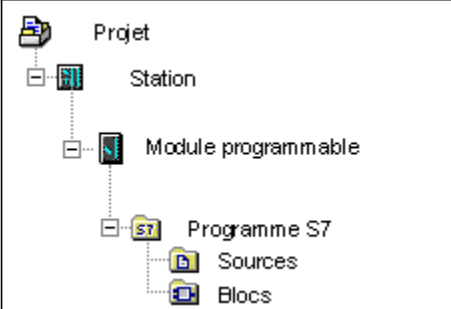
Icône	Dossier d'objets	Sélection de fonctions importantes
	Station	<ul style="list-style-type: none"> <li>• Insertion de stations</li> <li>• Chargement d'une station dans la PG</li> <li>• Chargement d'une configuration dans un système cible</li> <li>• Chargement d'une configuration depuis une station dans la PG</li> <li>• Affichage des messages de CPU et des messages de diagnostic personnalisés</li> <li>• Configuration de la signalisation d'erreurs système</li> <li>• Diagnostic du matériel et affichage de l'état du module</li> <li>• Affichage et modification de l'état de fonctionnement</li> <li>• Affichage et réglage de l'heure et de la date</li> <li>• Effacement de la mémoire de chargement/travail et effacement général de la CPU</li> </ul>
	Station SIMATIC PC (non affectée)	<ul style="list-style-type: none"> <li>• Création et paramétrage de stations SIMATIC PC</li> <li>• Configuration de liaisons pour une station SIMATIC PC</li> <li>• Chargement dans une station SIMATIC PC</li> </ul>
	Station SIMATIC PC (affectée)	<ul style="list-style-type: none"> <li>• Mise en valeur dans la vue de réseau d'une station SIMATIC PC configurée</li> </ul>


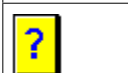
Icône	Objets dans le niveau de la station	Sélection de fonctions importantes
	Matériel	<ul style="list-style-type: none"><li>• Manipulations de base pour la configuration matérielle</li><li>• Marche à suivre pour la configuration d'une station</li><li>• Configuration et paramétrage d'une installation centralisée</li><li>• Marche à suivre pour la configuration d'un réseau maître DP</li><li>• Configuration du mode multiprocesseur</li></ul>
	Module programmable	<ul style="list-style-type: none"><li>• Les modules programmables sont à la fois des objets (niveau de la station) et des dossiers d'objets (niveau des modules programmables). Pour d'autres fonctions, reportez-vous à Objet Module programmable.</li></ul>



### 5.4.4 Objet Module programmable

Un module programmable représente les données de paramétrage d'un module programmable (CPUxxx, FMxxx, CPxxx). Les données système de modules ne possédant pas de mémoire rémanente (par exemple CP441) sont chargés via la CPU de la station. Aucun objet "Données système" n'est de ce fait affecté à de tels modules qui n'apparaissent pas dans la hiérarchie du projet.

#### Position dans la vue du projet

	<ul style="list-style-type: none"> <li>• Objet Projet</li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	---


Icône	Dossier d'objets	Sélection de fonctions importantes
	Module programmable	<ul style="list-style-type: none"> <li>• Configuration et paramétrage d'une installation centralisée</li> <li>• Affichage des messages de CPU et des messages de diagnostic personnalisés</li> <li>• Configuration de la signalisation d'erreurs système</li> <li>• Diagnostic du matériel et affichage de l'état du module</li> <li>• Chargement via des cartes mémoire EPROM</li> <li>• Protection par mot de passe contre l'accès aux systèmes cibles</li> <li>• Affichage de la fenêtre de forçage permanent</li> <li>• Affichage et modification de l'état de fonctionnement</li> <li>• Affichage et réglage de l'heure et de la date</li> <li>• Définition du comportement en fonctionnement</li> <li>• Effacement de la mémoire de chargement/travail et effacement général de la CPU</li> <li>• Icônes de diagnostic dans la vue en ligne</li> <li>• Organisation des zones de mémoire</li> <li>• Enregistrement de blocs chargés sur la mémoire intégrée EPROM</li> <li>• Actualisation du système d'exploitation sur le système cible</li> </ul>
	Objet suppléant pour un module programmable	<ul style="list-style-type: none"> <li>• Représentation de modules ayant été configurés avec des versions plus récentes de STEP 7</li> </ul>




Icône	Objets dans le niveau "Module programmable"	Sélection de fonctions importantes
	<p>Programmes :</p> <p>Programme S7</p> <p>Programme M7</p> <p>Programme</p>	<ul style="list-style-type: none"> <li>• Insertion d'un programme S7/M7</li> <li>• Les programmes S7/M7 sont à la fois des objets (niveau du projet) et des dossiers d'objets (niveau du programme). Pour d'autres fonctions, reportez-vous à Objet Programme S7/M7</li> </ul>
	<p>Liaisons pour la définition de liaisons dans le réseau</p>	<ul style="list-style-type: none"> <li>• Mise en réseau de stations au sein d'un projet</li> <li>• Types de liaison pour des partenaires dans le même projet</li> <li>• Informations sur les divers types de liaison</li> <li>• Saisie d'une nouvelle liaison</li> <li>• Configuration de liaisons pour les modules d'une station SIMATIC</li> </ul>

### 5.4.5 Objet Programme S7/M7





Un programme (S7/M7) est un dossier contenant les logiciels pour les modules CPU S7/M7 et les logiciels pour les modules autres que les CPU (par exemple modules CP ou FM programmables).

#### Position dans la vue du projet

	<ul style="list-style-type: none"> <li>• Objet Projet</li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	---

Icône	Dossier d'objets	Sélection de fonctions importantes
	Programme S7	<ul style="list-style-type: none"> <li>• Insertion d'un programme S7/M7</li> <li>• Définition de la priorité de l'opérande (symbolique/absolu)</li> <li>• Marche à suivre pour la création de blocs de code</li> <li>• Attribution de numéros de message</li> <li>• Création et édition de messages de diagnostic personnalisés (pour tout le projet)</li> <li>• Création et édition de messages de diagnostic personnalisés (pour la CPU)</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Gestion multilingue de textes</li> <li>• Affichage des messages de CPU et des messages de diagnostic personnalisés</li> <li>• Mesures à prendre dans le programme pour traiter les erreurs</li> </ul>
	Programme M7	<ul style="list-style-type: none"> <li>• Procédure pour les systèmes M7</li> </ul>
	Programme	<ul style="list-style-type: none"> <li>• Création du logiciel dans le projet (principe)</li> </ul>



Icône	Objets dans le niveau du programme	Sélection de fonctions importantes
	Dossier Sources	<ul style="list-style-type: none"> <li>• Pour d'autres fonctions, reportez-vous à Objet Dossier Sources</li> </ul>
	Dossier Blocs	<ul style="list-style-type: none"> <li>• Pour d'autres fonctions, reportez-vous à Objet Dossier Blocs</li> </ul>
	Dossier Bibliothèques de texte	<ul style="list-style-type: none"> <li>• Bibliothèques de texte utilisateur</li> </ul>
	Table des mnémoniques pour l'affectation de mnémoniques à des signaux et autres variables	<ul style="list-style-type: none"> <li>• Adressage absolu et adressage symbolique</li> <li>• Structure et éléments de la table des mnémoniques</li> <li>• Possibilités de saisie de mnémoniques globaux</li> <li>• Remarques générales sur la saisie de mnémoniques</li> <li>• Affectation et édition de messages sur mnémoniques (pour tout le projet)</li> <li>• Affectation et édition de messages sur mnémoniques (pour la CPU)</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Configuration des attributs de contrôle-commande au moyen de la table des mnémoniques</li> <li>• Edition de l'attribut de communication</li> <li>• Exportation et importation de tables de mnémoniques</li> </ul>

### 5.4.6 Object Dossier Blocs

Le dossier Blocs d'une vue hors ligne peut contenir : des blocs de code (OB, FB, FC, SFB, SFC), des blocs de données (DB), des types de données utilisateur (UDT) et des tables de variables. L'objet Données système représente les blocs de données système.





Le dossier Blocs d'une vue en ligne contient les éléments de programme exécutables, chargés de manière résidente dans le système cible.








#### Position dans la vue du projet

	<ul style="list-style-type: none"> <li>• Objet Projet</li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
--	---

Icône	Dossier d'objets	Sélection de fonctions importantes
	Blocs	<ul style="list-style-type: none"> <li>• Chargement dans la gestion du projet</li> <li>• Chargement hors gestion du projet</li> <li>• Vue synoptique des données de référence possibles</li> <li>• Réassignation</li> <li>• Comparaison de blocs</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Sauts dans la description des langages, aide sur les blocs, attributs système</li> </ul>

Icône	Objets dans le dossier Blocs	Sélection de fonctions importantes
	Généralités sur les blocs	<ul style="list-style-type: none"> <li>• Marche à suivre pour la création de blocs de code</li> <li>• Création de blocs</li> <li>• Principes de la programmation dans les sources LIST</li> <li>• Comparaison de blocs</li> </ul>
	OB (Blocs d'organisation)	Fonctions supplémentaires : <ul style="list-style-type: none"> <li>• Introduction aux types de données et de paramètres</li> <li>• Conditions préalables au chargement</li> <li>• Test avec la visualisation d'état du programme</li> <li>• Informations sur le test en mode pas à pas et sur les points d'arrêt</li> <li>• Réassignation</li> <li>• Aide sur les blocs</li> </ul>

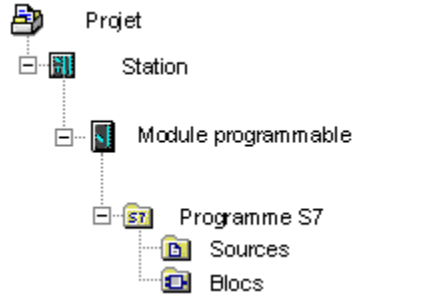
Icône	Objets dans le dossier Blocs	Sélection de fonctions importantes
	FC (Fonctions)	Fonctions supplémentaires : <ul style="list-style-type: none"> <li>• Introduction aux types de données et de paramètres</li> <li>• Conditions préalables au chargement</li> <li>• Test avec la visualisation d'état du programme</li> <li>• Informations sur le test en mode pas à pas et sur les points d'arrêt</li> <li>• Réassignation</li> <li>• Attributs pour blocs et pour paramètres</li> </ul>
	FB (Blocs fonctionnels)	Fonctions supplémentaires : <ul style="list-style-type: none"> <li>• Introduction aux types de données et de paramètres</li> <li>• Utilisation de multi-instances</li> <li>• Conditions préalables au chargement</li> <li>• Test avec la visualisation d'état du programme</li> <li>• Informations sur le test en mode pas à pas et sur les points d'arrêt</li> <li>• Réassignation</li> <li>• Attributs pour blocs et pour paramètres</li> <li>• Affectation et édition de messages sur bloc (pour tout le projet)</li> <li>• Création de messages sur bloc (pour la CPU)</li> <li>• Configuration de messages PCS7 (pour tout le projet)</li> <li>• Configuration de messages PCS7 (pour la CPU)</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Affectation d'attributs C+C aux paramètres FB</li> </ul>
	UDT (Types de données utilisateur)	<ul style="list-style-type: none"> <li>• Création de blocs</li> <li>• Principes de la programmation dans des sources LIST</li> <li>• Introduction aux types de données et de paramètres</li> <li>• Utilisation de types de données utilisateur pour l'accès aux données</li> <li>• Attributs pour blocs et pour paramètres</li> </ul>
	DB (Blocs de données)	<ul style="list-style-type: none"> <li>• Vue des données de blocs de données</li> <li>• Vue des déclarations de blocs de données</li> <li>• Conditions préalables au chargement</li> <li>• Etat du programme de blocs de données</li> <li>• Introduction aux types de données et de paramètres</li> <li>• Utilisation de multi-instances</li> <li>• Attributs pour blocs et pour paramètres</li> <li>• Affectation et édition de messages sur blocs (pour tout le projet -uniquement DB d'instance)</li> <li>• Edition de messages sur bloc (pour la CPU - uniquement DB d'instance)</li> <li>• Configuration de messages PCS7 (pour tout le projet - uniquement DB d'instance)</li> <li>• Configuration de messages PCS7 (pour la CPU - uniquement DB d'instance)</li> <li>• Traduction et édition de textes destinés à l'utilisateur (uniquement DB d'instance)</li> </ul>


Icône	Objets dans le dossier Blocs	Sélection de fonctions importantes
	SFC (Fonctions système)	<ul style="list-style-type: none"> <li>• Conditions préalables au chargement</li> <li>• Attributs pour blocs et pour paramètres</li> <li>• Aide sur les blocs</li> </ul>
	SFB (Blocs fonctionnels système)	<ul style="list-style-type: none"> <li>• Conditions préalables au chargement</li> <li>• Attributs pour blocs et pour paramètres</li> <li>• Affectation et édition de messages sur bloc (pour tout le projet)</li> <li>• Création de messages sur bloc (pour la CPU)</li> <li>• Configuration de messages PCS7 (pour tout le projet)</li> <li>• Configuration de messages PCS7 (pour la CPU)</li> <li>• Traduction et édition de textes destinés à l'utilisateur</li> <li>• Aide sur les blocs</li> </ul>
	Bloc avec protection KNOW HOW	<ul style="list-style-type: none"> <li>• Règles pour la définition des propriétés de bloc dans les sources LIST</li> <li>• Propriétés de bloc</li> </ul>
	Bloc avec fonction de diagnostic	De plus amples informations sont fournies dans la documentation du progiciel optionnel S7-PDIAG.
	Bloc créé avec le langage de programmation LOG F/CONT F/LIST F/DB F	De plus amples informations sont fournies dans la documentation du progiciel optionnel S7 Distributed Safety.
	Table de variables	<ul style="list-style-type: none"> <li>• Marche à suivre pour la visualisation et le forçage avec des tables de variables</li> <li>• Introduction au test avec des tables de variables</li> <li>• Introduction à la visualisation de variables</li> <li>• Introduction au forçage de variables</li> <li>• Introduction au forçage permanent de variables</li> </ul>
	Données système (SDB)	<p>Les SDB ne sont édités que de manière indirecte à l'aide de fonctions de :</p> <ul style="list-style-type: none"> <li>• Introduction à la configuration du matériel</li> <li>• Propriétés des sous-réseaux et des participants à la communication</li> <li>• Présentation : communication par données globales</li> <li>• Affectation et édition de messages sur mnémoniques</li> <li>• Conditions préalables au chargement</li> </ul>



## 5.4.7 Objet Dossier Sources

Un dossier Sources contient les programmes source sous forme de texte.

### Position dans la vue du projet

	<ul style="list-style-type: none"> <li>• Objet Projet</li> <li>• Objet Station</li>   <li>• Objet Module programmable</li>   <li>• Objet Programme S7/M7</li> <li>• Objet Dossier Sources</li> <li>• Objet Dossier Blocs</li> </ul>
---	---

Icône	Dossier d'objets	Sélection de fonctions importantes
	Dossier Sources	<ul style="list-style-type: none"> <li>• Principes de la programmation dans les sources LIST</li> <li>• Exportation d'une source</li> <li>• Importation d'une source</li> </ul>

Icône	Objets dans le dossier Sources	Sélection de fonctions importantes
	Source (p. ex. source LIST)	<ul style="list-style-type: none"> <li>• Principes de la programmation dans les sources LIST</li> <li>• Création d'une source LIST</li> <li>• Insertion de modèles de blocs dans une source LIST</li> <li>• Insertion du code source de blocs existants dans des sources LIST</li> <li>• Vérification de la cohérence d'une source LIST</li> <li>• Compilation d'une source LIST</li> <li>• Génération d'une source LIST à partir de blocs</li> <li>• Exportation d'une source</li> <li>• Importation d'une source</li> </ul>
	Modèle de réseau	<ul style="list-style-type: none"> <li>• Utilisation de modèles de réseau</li> </ul>

### 5.4.8 Programme S7/M7 sans station ni CPU

Vous avez la possibilité de créer des programmes sans avoir préalablement configuré une station SIMATIC. Dans un premier temps, vous pouvez ainsi programmer indépendamment du module à programmer et de ses paramètres.

#### Création du programme S7/M7

1. Ouvrez le projet correspondant en choisissant la commande **Fichier > Ouvrir** ou activez la fenêtre du projet.
2. Sélectionnez le projet dans la fenêtre du projet de la vue hors ligne.
3. Selon le système cible auquel le programme créé est destiné, choisissez la commande correspondante :  
**Insertion > Programme > Programme S7**, si votre programme doit ultérieurement être exécuté sur un automate programmable SIMATIC S7.  
**Insertion > Programme > Programme M7**, si votre programme doit ultérieurement être exécuté sur un système d'automatisation SIMATIC M7.

Le programme S7/M7 est inséré est placé directement sous le projet dans la fenêtre du projet. Il contient un dossier pour les blocs ainsi qu'une table des mnémoniques vide. Vous pouvez à présent créer et programmer des blocs.

#### Affectation à un module programmable

Des programmes insérés indépendamment d'un module peuvent être affectés ultérieurement à un module dans la fenêtre du projet. Il suffit de copier ou déplacer ces programmes par glisser-lâcher sur l'icône du module.

#### Insertion dans une bibliothèque

Lorsque le programme est destiné au système cible SIMATIC S7 et doit être utilisé de manière multiple tel un "regroupement de logiciel", vous pouvez également l'insérer dans une bibliothèque. Pour le test, les programmes doivent cependant se trouver directement sous un projet afin que vous puissiez établir une liaison au système cible.

#### Accès à un système cible

Sélectionnez la vue en ligne du projet. Dans la boîte de dialogue des propriétés du programme, vous pouvez effectuer le paramétrage des adresses.

---

#### Nota

Lorsque vous supprimez des stations ou modules programmables, le système vous demande si vous souhaitez également supprimer le programme qu'ils contiennent. Si vous répondez par non, le programme est directement accroché sous le projet en tant que programme sans station.

---

## **5.5 Interface utilisateur et manipulation**

### **5.5.1 Concept d'utilisation**

#### **But : utilisation simple**

L'interface utilisateur graphique doit permettre une approche aussi intuitive que possible. Vous y trouvez donc des objets que vous connaissez de par votre environnement de travail quotidien, comme les stations, les modules, les programmes et les blocs.

Les actions que vous exécutez lorsque vous utilisez STEP 7 comprennent la création, la sélection et la manipulation de tels objets.

#### **Différences par rapport au concept orienté application**

Selon le concept d'utilisation traditionnel, qui est orienté application, il fallait d'abord trouver quelle était l'application nécessaire à la résolution d'un travail précis et appeler ensuite cette application.

Selon le concept orienté objets, il s'agit de savoir quel objet doit être traité, puis d'ouvrir et de traiter cet objet.

Ainsi, avec ce concept, il n'est plus nécessaire de connaître une syntaxe de commande particulière. Les objets sont représentés, sur l'interface utilisateur, par des icônes que vous pouvez ouvrir par commandes de menu ou clics de la souris.

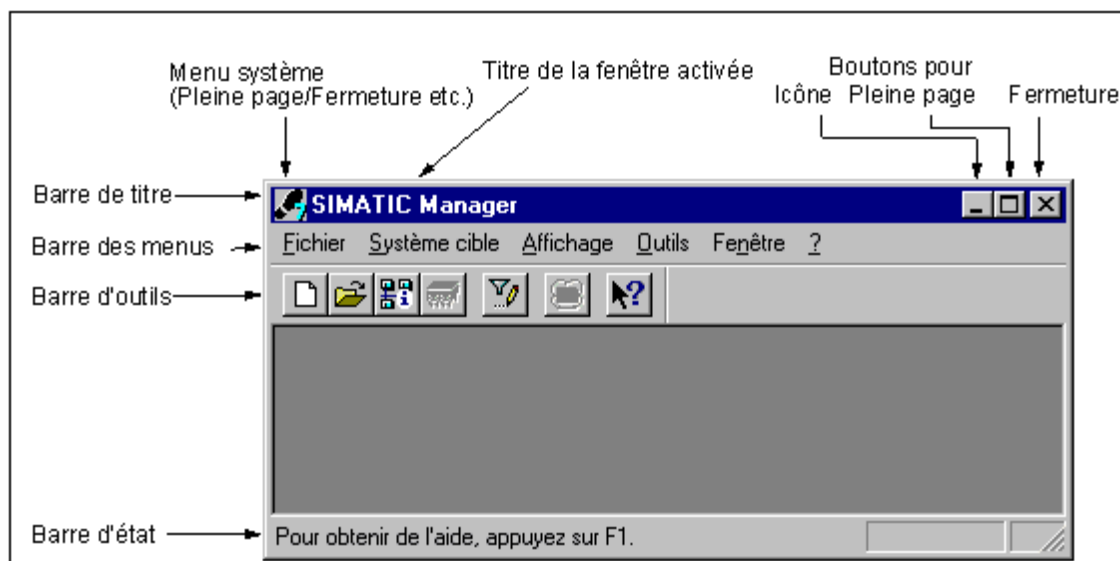
A l'ouverture d'un objet, le composant logiciel approprié est automatiquement appelé pour afficher ou traiter le contenu de l'objet.

#### **Pages suivantes ...**

Les pages suivantes présentent les opérations fondamentales pour le traitement des objets. Familiarisez-vous dès maintenant avec ces opérations fondamentales qui vous seront toujours indispensables par la suite.

## 5.5.2 Structure de la fenêtre

Les composants standard d'une fenêtre sont présentés dans la figure suivante.



### Barre de titre et barre des menus

La barre de titre et la barre des menus se situent toujours au bord supérieur de la fenêtre. La barre de titre contient le titre de la fenêtre et les boutons permettant d'influer sur la fenêtre. La barre des menus contient tous les menus disponibles dans la fenêtre.

### Barre d'outils

La barre d'outils contient des boutons vous permettant d'accéder rapidement par clic de la souris aux commandes de menu disponibles qui sont le plus souvent utilisées. Une information succincte sur la fonction de chaque bouton s'affiche lorsque vous positionnez le curseur pendant un court instant sur le bouton, une information supplémentaire s'affiche dans la barre d'état.

Quand l'accès à un bouton n'est pas possible dans la configuration en cours, celui-ci est estompé.

### Barre d'état

La barre d'état affiche des informations contextuelles.



### 5.5.3 Éléments dans les boîtes de dialogue

#### Saisie dans les boîtes de dialogue

Vous pouvez entrer dans les boîtes de dialogue les informations nécessaires pour l'exécution de certains travaux. La figure suivante présente, à l'aide d'un exemple, les composants les plus courants des boîtes de dialogue.



#### Zones de listes, zones de combinaisons

Certaines zones de texte sont suivies d'une flèche vers le bas. Cette flèche signifie qu'il existe d'autres choix pour la zone correspondante. Cliquez sur la flèche pour ouvrir une zone de liste ou de combinaison. Si vous cliquez alors sur une entrée, cette entrée sera automatiquement reprise dans la zone de texte.

#### Boîtes de dialogue à onglets

Le contenu de certaines boîtes de dialogue est organisé par pages afin d'assurer une meilleure vue d'ensemble (voir la figure suivante).



Le nom des différentes pages est inscrit dans des onglets apparaissant en haut de la boîte de dialogue. Pour amener une page d'onglet au premier plan, il suffit de cliquer sur l'onglet.

### 5.5.4 Création et manipulation d'objets

Quelques opérations de base sont identiques pour tous les objets. Nous allons d'abord les résumer, puis nous les considérerons comme acquises lorsque nous décrirons les procédés dans les chapitres suivants de ce guide.

La séquence normale de manipulation des objets est :

- création de l'objet,
- sélection de l'objet,
- exécution d'actions sur l'objet (par exemple, copie, suppression).

#### Chemin de nouveaux projets/ nouvelles bibliothèques

Les nouveaux projets utilisateur, nouvelles bibliothèques et nouveaux multiprojets sont enregistrés par défaut dans le répertoire "\Siemens\Step7\S7proj". Si vous souhaitez choisir un autre répertoire, vous devriez en définir le chemin avant la première création de nouveaux projets utilisateur, de nouvelles bibliothèques ou de nouveaux multiprojets. Choisissez à cet effet la commande **Outils > Paramètres**. Dans la page d'onglet "Général" de la boîte de dialogue affichée, vous pouvez spécifier le chemin de création des nouveaux projets et des nouvelles bibliothèques.

#### Création d'objets

L'assistant de STEP 7 "Nouveau projet" vous aidera à créer un nouveau projet et à insérer des objets. Pour l'appeler, choisissez la commande **Fichier > Assistant "Nouveau projet"**. Dans les boîtes de dialogue qui s'affichent, vous pouvez définir la structure de votre projet, puis le faire créer par l'assistant.

Si vous préférez ne pas avoir recours à l'assistant, vous pouvez créer des projets et des bibliothèques à l'aide de la commande **Fichier > Nouveau**. Ces objets constituent la tête d'une bjets. Vous pouvez créer tous les autres objets - à condition bien sûr qu'ils ne le soient pas automatiquement - à l'aide des commandes du menu "Insertion". Les modules d'une station SIMATIC représentent une exception, car ils sont créés uniquement dans le cadre de la configuration matérielle et par l'assistant "Nouveau projet".

## Ouverture d'objets

Vous pouvez ouvrir un objet dans la vue de détail de plusieurs manières :

- effectuez un double clic sur l'icône de l'objet ou
- sélectionnez l'objet puis la commande **Edition > Ouvrir l'objet**. Ceci ne s'applique qu'à des objets qui ne sont pas des dossiers.

Une fois un objet ouvert, vous pouvez créer ou modifier son contenu.

Lorsque vous ouvrez un objet de ce second type, son contenu est représenté pour traitement par un composant logiciel approprié dans une nouvelle fenêtre. Vous ne pouvez pas modifier des objets dont le contenu est déjà utilisé à un autre endroit.

---

### Nota

Exception : Les stations représentent les dossiers des modules programmables (par double clic) et de la configuration de la station. Lorsque vous effectuez un double clic sur l'objet "Matériel", l'application de configuration du matériel démarre. Le même résultat s'obtient par sélection de la station puis activation de la commande **Edition > Ouvrir l'objet**.

---

## Constitution d'une hiérarchie d'objets

Faites-vous établir la hiérarchie des objets par l'assistant "Nouveau projet". Lorsque vous ouvrez un dossier, les objets qu'il contient déjà s'affichent à l'écran. Le menu "Insertion" vous permet alors de créer d'autres sous-objets, par exemple d'autres stations dans un projet. Ce menu ne met à votre disposition que les commandes d'insertion de ceux des objets qui sont autorisés dans le dossier en cours.

## Définition des propriétés d'un objet

Les propriétés sont des données de l'objet qui déterminent son comportement. La boîte de dialogue de définition des propriétés d'un objet s'affiche automatiquement quand vous créez un objet et qu'il faut en définir les propriétés. Mais il est aussi possible de modifier les propriétés après coup.

La commande **Edition > Propriétés de l'objet** appelle une boîte de dialogue permettant de lire ou de définir les propriétés pour l'objet choisi.

La commande **Edition > Propriétés spécifiques de l'objet** appelle des boîtes de dialogue permettant de saisir les données requises pour le contrôle-commande ainsi que pour la configuration des messages.

Par exemple, pour pouvoir appeler les propriétés spécifiques d'un bloc pour le contrôle-commande, il faut avoir défini ce bloc comme bloc de contrôle-commande, c'est-à-dire avoir écrit l'attribut système "s7\_m\_c" avec la valeur "true" dans la page d'onglet "Attributs" des propriétés du bloc.

---

**Nota**

- Vous ne pouvez ni afficher, ni modifier les propriétés du dossier "Données système" et de l'objet "Matériel".
  - Vous ne pouvez pas écrire dans les boîtes de dialogue des propriétés d'un projet protégé en écriture. Dans ce cas, les zones de saisie sont estompées.
  - Lorsque vous affichez les propriétés de modules programmables, vous ne pouvez pas éditer les paramètres affichés pour des raisons de cohérence. Pour éditer les paramètres, vous devez ouvrir l'application "Configuration du matériel".
  - Lorsque vous modifiez les paramètres d'objets sur l'outil de développement (p. ex. les données de paramétrage d'un module), ils ne sont pas immédiatement actifs sur le système cible. En effet, les blocs de données système dans lesquels ces paramètres sont enregistrés doivent se trouver sur le système cible.
  - Lorsque vous chargez un programme utilisateur complet, les blocs de données système sont automatiquement chargés. Si après avoir chargé un programme, vous effectuez des modifications de paramétrage, vous pouvez recharger l'objet "Données système", afin d'amener les nouveaux paramètres dans le système cible.
  - Il est vivement recommandé d'éditer les dossiers uniquement dans STEP 7, car leur structure physique peut être différente que celle que vous voyez dans SIMATIC Manager.
- 

**Couper, coller ou copier**

Vous pouvez couper, coller et copier la plupart des objets comme vous le faites sous Windows. Les commandes correspondantes appartiennent au menu "Edition".

Vous pouvez également copier des objets à l'aide de la fonction "glisser-lâcher" (Drag&Drop). Si vous pointez sur une destination incorrecte, le curseur se change en signe d'interdiction.

Lorsque vous copiez un objet, toute la hiérarchie en aval de cet objet est également copiée. Cela permet d'utiliser à nouveau des composants conçus pour une autre solution d'automatisation.

---

**Nota**

Il n'est pas possible de copier la table des liaisons dans le dossier "Liaisons". Lorsque vous copiez des listes de textes destinés à l'utilisateur, veillez à ce que seules soient reprises les langues qui sont installées dans l'objet cible.

---

La marche à suivre pour la copie est décrite étape par étape sous Copie d'objets.

## Renommer des objets

SIMATIC Manager attribue des noms standardisés à certains objets nouvellement collés. Ces noms sont généralement formés à partir du type de l'objet et (lorsque plusieurs objets de ce type sont créés dans le même dossier) d'un numéro en cours.

Ainsi, par exemple, le premier programme S7 est nommé "Programme S7 (1)", le deuxième "Programme S7 (2)". La table des mnémonique, quant à elle, s'appelle simplement "Mnémoniques" puisqu'elle n'existe qu'une seule fois dans chaque dossier de niveau hiérarchique supérieur.

Vous avez la possibilité de renommer la plupart des objets afin de leur attribuer une désignation plus explicite.

Pour les projets, les noms de répertoire du chemin ne doivent pas excéder 8 caractères. Sinon, des problèmes risqueraient de se poser lors de l'archivage et de l'utilisation de "C pour M7" (compilateur Borland).

Les noms d'objets modifiables peuvent directement être édités ou modifiés à l'aide des propriétés d'objet.

### **Edition directe :**

- Dans la fenêtre du projet, cliquez deux fois lentement sur le nom d'un objet sélectionné. Un cadre apparaît autour du texte. Vous pouvez alors éditer ce texte via le clavier.

### **Modification à l'aide des propriétés de l'objet :**

- Dans la fenêtre du projet, sélectionnez l'objet souhaité et choisissez la commande de menu **Edition > Renommer**. Un cadre s'affiche autour du textee. Vous pouvez à présent éditer la désignation via le clavier.

Lorsqu'il n'est pas permis de modifier un nom d'objet, la zone de saisie est représentée en gris dans la boîte de dialogue, le nom en cours est affiché et la saisie est impossible.

---

### **Nota**

Si, durant l'édition, vous déplacez le curseur hors de la zone du nom pour effectuer une autre action (par exemple sélectionner une commande), l'édition est interrompue. S'il est valide, le nom modifié est repris.

---

La marche à suivre pour renommer un objet est décrite étape par étape sous Renommer un objet.

## Déplacer un objet

SIMATIC Manager vous permet de déplacer des objets d'un dossier à un autre, même si ce dernier se trouve dans un autre projet. Lorsque vous déplacez un dossier, tout son contenu est également déplacé.

---

### Nota

Il n'est pas possible de déplacer les objets suivants :

- Liaisons
  - Blocs de données système (SDB) dans la vue en ligne
  - Fonctions système (SFC) et blocs fonctionnels système (SFB) dans la vue en ligne
- 

Le marche à suivre pour le déplacement est décrite étape par étape dans Déplacement d'objets.

## Trier des objets

Dans l'affichage de détail (commande **Affichage > Détails**), vous pouvez trier les objets d'après leurs attributs. Cliquez à cet effet sur l'en-tête de colonne correspondant à l'attribut souhaité. Un nouveau clic inverse l'ordre de tri. Les blocs sont triés d'après le numéro qui leur est attribué, par exemple FB1, FB2, FB11, FB12, FB21, FC1.

### Ordre de tri prédéfini (tri par défaut) :

Lorsque vous ouvrez un projet pour la première fois, les objets s'affichent d'après un ordre de tri prédéfini dans l'affichage "Détails". Exemples :

- Les blocs sont affichés dans l'ordre "Données système, OB, FB, FC, DB, UDT, VAT, SFB, SFC".
- Dans les projets s'affichent d'abord toutes les stations puis les programmes S7.

Le critère de tri prédéfini pour l'affichage de détail ne correspond donc pas à un classement alphabétique croissant ou décroissant.

### Restauration du tri par défaut :

Après avoir effectué un tri, par exemple par clic sur l'en-tête de colonne "Nom de l'objet", vous pouvez à nouveau restaurer l'ordre prédéfini en procédant de la manière suivante :

- cliquez sur l'en-tête de colonne "Type" dans l'affichage de détail,
- quittez puis ouvrez à nouveau le projet.

## Suppression d'objets

Vous pouvez supprimer aussi bien des dossiers que des objets. Lorsque vous supprimez un dossier, tous les objets qu'il contient le sont également.

Il n'est pas possible d'annuler une opération de suppression. Si vous n'êtes pas certain de ne plus avoir besoin d'un objet, il est préférable d'archiver précédemment l'ensemble du projet.

---

### Nota

Il n'est pas possible de supprimer les objets suivants :

- Liaisons
  - Blocs de données système (SDB) dans la vue en ligne
  - Fonctions système (SFC) et blocs fonctionnels système (SFB) dans la vue en ligne
- 

La marche à suivre pour la suppression est décrite étape par étape dans Suppression d'objets.

### 5.5.5 Sélection d'objets dans les boîtes de dialogue

La sélection d'objets dans une boîte de dialogue est une opération que vous devez effectuer à différentes étapes.

#### Appel d'une boîte de dialogue

Une boîte de dialogue est, par exemple, appelée dans la configuration matérielle par des commandes telles que **Station > Nouvelle.../Ouvrir...** ; la fenêtre d'accès "SIMATIC Manager" constitue une exception.

#### Composition d'une boîte de dialogue

Une boîte de dialogue offre les possibilités de sélection présentées par la figure suivante.

**Entrée :** ici, vous sélectionnez le type de l'objet dans lequel vous souhaitez débiter la recherche (p. ex. "Projet", "Bibliothèque", mais aussi des entrées qui permettent l'accès à des lecteurs ou des systèmes d'automatisation connectés).

**Affichage :** ici vous sélectionnez la vue standard ou la vue technologique.

**En ligne/Hors ligne :** pour l'entrée "Projet" exclusivement, vous pouvez commuter entre la vue hors ligne (choix dans les données de projet sur votre P.G./P.C) et la vue en ligne (choix dans les données de projet sur le système d'automatisation connecté).

**Parcourir :** cliquez sur ce bouton pour rechercher des objets qui ne font pas partie de la liste.

**Nom :** les objets connus du type indiqué dans "Entrée" sont affichés dans une zone de liste. Vous pouvez sélectionner un nom dans la liste ou le saisir directement via le clavier.

**Type d'objet :** ici vous pouvez indiquer un critère de filtrage pour la liste d'objets. Vous pouvez ainsi restreindre le nombre d'objets affichés.

**Nom de l'objet :** lorsque vous sélectionnez un objet, son nom est entré ici. Vous pouvez également saisir directement le nom souhaité.

**Vue du projet :** affiche l'arborescence des objets qui eux-mêmes peuvent contenir d'autres objets.

**Vue détaillée :** affiche le contenu de l'objet sélectionné à gauche.



## 5.5.6 Historique des sessions

SIMATIC Manager est en mesure de mémoriser le contenu des fenêtres, c'est-à-dire les projets et bibliothèques ouverts, ainsi que la disposition des fenêtres.

- La commande **Outils > Paramètres** vous permet de définir si le contenu et la disposition des fenêtres doivent être enregistrés en fin de la session. Dans ce cas, ils seront restaurés au début de la session suivante. Dans les projets ouverts, le curseur se positionne sur le dernier dossier ouvert.
- La commande **Fenêtre > Enregistrer la disposition** enregistre le contenu et la disposition actuels de la fenêtre.
- La commande **Fenêtre > Restaurer la disposition** restaure le contenu et la disposition de la fenêtre précédemment enregistrés à l'aide de la commande **Fenêtre > Enregistrer la disposition**. Le curseur se positionne sur le dernier dossier ouvert.

---

### Nota

Le contenu de la fenêtre de projets en ligne, celui de la fenêtre "Partenaires accessibles" et celui de la fenêtre "Carte mémoire S7" ne sont pas enregistrés.

Les mots de passe éventuellement saisis pour l'accès aux systèmes cible (S7- 300/S7-400) ne sont pas enregistrés au-delà de la fin de la session.

---

## 5.5.7 Modification de la disposition des fenêtres

Pour disposer toutes les fenêtres affichées de manière à ce qu'elles se chevauchent, vous avez les possibilités suivantes :

- Choisissez la commande **Fenêtre > Disposition > Cascade**.
- Appuyez sur les touches MAJ + F5.

Pour disposer toutes les fenêtres les unes sous les autres, choisissez la commande **Fenêtre > Disposition > Mosaique verticale**.

Pour disposer toutes les fenêtres les unes à côté des autres, choisissez la commande **Fenêtre > Disposition > Mosaique horizontale**.

## 5.5.8 Enregistrement et restauration de la disposition des fenêtres

Les applications de STEP 7 offrent la possibilité d'enregistrer la disposition actuelle des fenêtres afin de pouvoir la restaurer à tout moment. Ce paramétrage peut être réalisé à l'aide de la commande **Outils > Paramètres**, dans l'onglet "Général".

### Informations sauvegardées

Lorsque vous effectuez l'enregistrement de la disposition des fenêtres, les informations suivantes sont sauvegardées :

- position de la fenêtre principale,
- projets et bibliothèques ouverts ainsi que position des fenêtres correspondantes,
- éventuellement ordre de superposition des fenêtres.

---

#### Nota

Le contenu des fenêtres de projets en ligne, celui de la fenêtre "Partenaires accessibles" et celui de la fenêtre "Carte mémoire S7" ne sont pas enregistrés.

---

### Enregistrement de la disposition des fenêtres

Pour enregistrer la disposition actuelle des fenêtres, choisissez la commande **Fenêtre > Enregistrer la disposition**.

### Restauration de la disposition des fenêtres

Pour restaurer une disposition des fenêtres enregistrée, choisissez la commande **Fenêtre > Restaurer la disposition**.

---

#### Nota

Lors de la restauration d'une fenêtre, seule la partie de la hiérarchie contenant l'objet qui était sélectionné lors de l'enregistrement sera représentée en détail.

---

## 5.6 Utilisation du clavier

Désignation internationale des touches	Désignation française des touches
HOME	ORIGINE
END	FIN
PAGE-UP	PG.PREC
PAGE-DOWN	PG.SUIV
CTRL	CTRL
ENTER	ENTREE
DEL	SUPPR
INSERT	INSER

### 5.6.1 Combinaisons de touches pour les commandes de menu

Vous pouvez déclencher chaque commande en tapant la combinaison de la touche ALT avec la lettre soulignée appropriée.

Appuyez successivement sur les touches suivantes :

- touche ALT,
- lettre soulignée dans le menu désiré (par exemple, ALT, F pour le menu Fichier si le menu Fichier figure dans la barre des menus). Le menu s'ouvre.
- lettre soulignée dans la commande désirée (par exemple N pour la commande Nouveau). S'il s'agit d'une commande comportant des sous-menus, ceux-ci s'ouvriront. Continuez à taper les lettres soulignées, jusqu'à ce que la commande souhaitée soit complète.

La commande est déclenchée une fois la dernière lettre de la combinaison de touches saisie.

Exemples :

Commandes	Touches
Fichier > Archiver	ALT, F, A
Fichier > Ouvrir	ALT, F, O

## Equivalences clavier

Fonction	Equivalence
Nouveau (menu Fichier)	CTRL + N
Ouvrir (menu Fichier)	CTRL + O
Enregistrer sous (menu Fichier)	CTRL + S
Imprimer > Liste d'objets (menu Fichier)	CTRL + P
Imprimer > Objet (menu Fichier)	CTRL + ALT + P
Quitter (menu Fichier)	ALT + F4
Couper (menu Edition)	CTRL + X
Copier (menu Edition)	CTRL + C
Coller (menu Edition)	CTRL + V
Effacer (menu Edition)	SUPPR
Sélectionner tout (menu Edition)	CTRL + A
Renommer (menu Edition)	F2
Propriétés de l'objet (menu Edition)	ALT + ENTREE
Ouvrir un objet (menu Edition)	CTRL + ALT + O
Compiler (menu Edition)	CTRL+B
Charger (menu Système cible)	CTRL + L
Diagnostic/Paramètres > Etat du module (menu Système cible)	CTRL+D
Diagnostic/Paramètres > Etat de fonctionnement (menu Système cible)	CTRL+I
Actualiser (menu Affichage)	F5
Pour actualiser l'affichage d'état des CPU visibles dans la vue en ligne	CTRL + F5
Paramètres (menu Outils)	CTRL + ALT + E
Données de référence > afficher (menu Outils)	CTRL + ALT + R
Disposition > Cascade (menu Fenêtre)	MAJ + F5
Disposition > Mosaïque horizontale (menu Fenêtre)	MAJ + F2
Disposition > Mosaïque verticale (menu Fenêtre)	MAJ + F3
Aide contextuelle (menu d'aide ?)	F1 (S'il y a un contexte, par exemple si une commande est sélectionnée, la rubrique d'aide correspondante s'affiche. Sinon, c'est le sommaire de l'aide qui s'affiche.)

## 5.6.2 Combinaisons de touches pour le déplacement du curseur

### Déplacement du curseur dans la barre des menus ou dans un menu contextuel

Fonction	Touches
Activer la barre des menus	F10
Atteindre le menu contextuel	MAJ + F10
Au menu dont le nom contient le caractère souligné X	ALT + X
Choisir une commande subordonnée	Lettre soulignée dans le nom de commande
Déplacement d'un menu vers la gauche	Flèche vers la gauche
Déplacement d'un menu vers la droite	Flèche vers la droite
Déplacement d'un menu vers le haut	Flèche vers le haut
Déplacement d'un menu vers le bas	Flèche vers le bas
Activer la commande sélectionnée	ENTREE
Quitter le menu ou revenir au texte	ECHAP

### Déplacement du curseur lors de l'édition de texte

Fonction	Touches
Une ligne vers le haut ou un caractère vers la gauche dans un texte d'une seule ligne	Flèche vers le haut
Une ligne vers le bas ou un caractère vers la droite dans un texte d'une seule ligne	Flèche vers le bas
Un caractère vers la droite	Flèche vers la droite
Un caractère vers la gauche	Flèche vers la gauche
Un mot vers la droite	CTRL + Flèche vers la droite
Un mot vers la gauche	CTRL + Flèche vers la gauche
Au début de la ligne	ORIGINE
A la fin de la ligne	FIN
Une page d'écran vers le haut	PAGE PRECEDENTE
Une page d'écran vers le bas	PAGE SUIVANTE
Au début du texte	CTRL + ORIGINE
A la fin du texte	CTRL + FIN

## Déplacement du curseur lors de l'édition de tables

Fonction	Touches
Une cellule vers le haut	Flèche vers le haut
Une cellule vers le bas	Flèche vers le bas
Un caractère ou une cellule vers la gauche	Flèche vers la gauche
Un caractère ou une cellule vers la droite	Flèche vers la droite
Au début de la ligne	CTRL + Flèche vers la gauche
A la fin de la ligne	CTRL + Flèche vers la droite
Au début de la cellule	ORIGINE
A la fin de la cellule	FIN
Une page d'écran vers le haut	PAGE PRECEDENTE
Une page d'écran vers le bas	PAGE SUIVANTE
Au début de la table	CTRL + ORIGINE
A la fin de la table	CTRL + FIN
Uniquement dans la table des mnémoniques : à la colonne mnémonique	MAJ + ORIGINE
Uniquement dans la table des mnémoniques : à la colonne commentaire	MAJ + FIN

## Déplacement du curseur dans les boîtes de dialogue

Fonction	Touches
Au champ de saisie suivant (de gauche à droite et de haut en bas)	TAB
Au champ de saisie précédent	MAJ + TAB
Au champ de saisie dont le nom contient le caractère souligné X	ALT + X
Sélectionner dans une liste de choix	TOUCHES DE DIRECTION
Ouvrir une liste de choix	ALT + Flèche vers le bas
Sélectionner un objet ou en annuler la sélection	ESPACE
Confirmer la saisie et fermer la boîte de dialogue (bouton "OK")	ENTREE
Fermer la boîte de dialogue sans enregistrer les choix (bouton "Annuler").	ECHAP

### 5.6.3 Combinaisons de touches pour la sélection de texte

Fonction	Touches
Un caractère vers la droite	MAJ + flèche vers la droite
Un caractère vers la gauche	MAJ + flèche vers la gauche
Jusqu'au début de la ligne	MAJ + ORIGINE
Jusqu'à la fin de la ligne	MAJ + FIN
Une ligne dans la table	MAJ + BARRE D'ESPACEMENT
Une ligne de texte vers le haut	MAJ + flèche vers le haut
Une ligne de texte vers le bas	MAJ + flèche vers le bas
Une page d'écran vers le haut	MAJ + PG.PREC
Une page d'écran vers le bas	MAJ + PG.SUIV
Le texte jusqu'au début du fichier	CTRL + MAJ + ORIGINE
Le texte jusqu'à la fin du fichier	CTRL + MAJ + FIN

### 5.6.4 Combinaisons de touches pour accéder à l'aide en ligne

Fonction	Touches
Ouvrir l'aide	F1 (S'il y a un contexte, par exemple si une commande est sélectionnée, la rubrique d'aide correspondante s'affiche. Sinon, c'est le sommaire de l'aide qui s'affiche.)
Activer le bouton "Point d'interrogation" afin d'obtenir une aide contextuelle	MAJ + F1
Fermer la fenêtre d'aide et revenir à celle de l'application	ALT + F4

### 5.6.5 Combinaisons de touches pour la bascule entre les différents types de fenêtres

Fonction	Touches
Bascule d'un volet à un autre	F6
Bascule au volet précédent, en l'absence d'une fenêtre ancrée	MAJ + F6
Bascule entre la fenêtre du document et la fenêtre ancrée au document (par exemple fenêtre de déclaration des variables) En l'absence d'une fenêtre ancrée, la bascule s'effectue au volet précédent.	MAJ + F6
Bascule entre des fenêtres de document	CTRL + F6
Bascule à la fenêtre de document précédente	MAJ + CTRL + F6
Bascule entre des fenêtres autres que de documents (fenêtres d'applications et fenêtre qui y sont ancrées ; lors du retour à une fenêtre d'application, la bascule s'effectue à la dernière fenêtre de document active)	ALT + F6
Bascule à la fenêtre autre que de document précédente	MAJ + ALT + F6
Fermer la fenêtre active	CTRL + F4



## 6 Création et édition du projet

### 6.1 Structure du projet

Un projet permet de regrouper l'ensemble des programmes et données nécessaires à réaliser une tâche d'automatisation. Ces données englobent en particulier :

- les données de configuration pour la configuration matérielle et les données de paramétrage pour les modules,
- les données de configuration pour la communication par réseau et
- les programmes pour modules programmables.

La tâche principale dans la réalisation d'un projet, consiste à préparer ces données et à effectuer la programmation.

Dans un projet, les données sont enregistrées sous forme d'objets. Les objets sont organisés à l'intérieur d'un projet selon une structure arborescente (hiérarchie du projet). Dans la fenêtre du projet, cette structure hiérarchique est représentée de la même manière que dans l'Explorateur Windows. Seules les icônes des objets sont différentes.

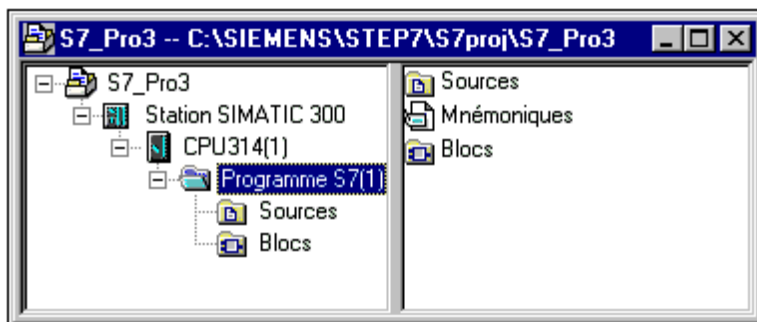
Le sommet de la hiérarchie se compose comme suit :

1. Niveau : projet
2. Niveau : sous-réseaux, stations ou programmes S7/M7
3. Niveau : dépend de l'objet correspondant du niveau 2.

## Fenêtre de projet

La fenêtre de projet est partagée en deux volets. Le volet gauche représente l'arborescence du projet. Le volet droit affiche le contenu de l'objet sélectionné dans le volet gauche dans le mode d'affichage sélectionné (grandes icônes, petites icônes, liste ou détails) .

Pour afficher l'arborescence complète du projet, cliquez sur les cases affichant le signe "Plus" dans la partie gauche de la fenêtre. Vous obtenez alors une représentation similaire à celle de la figure suivante.



L'objet "S7\_Pro1" se trouve ici à la tête de la hiérarchie d'objet, comme icône pour l'ensemble du projet. Il peut être utilisé pour afficher les propriétés du projet et sert de dossier aux réseaux (pour la configuration de réseaux), stations (pour la configuration matérielle) ainsi qu'aux programmes S7 ou M7 (pour la création du logiciel). Lorsque vous sélectionnez l'icône du projet, les objets que contient ce dernier sont affichés dans le volet droit de la fenêtre de projet. Les objets à la tête d'une telle hiérarchie (projets mais aussi bibliothèques) constituent les points de départ dans les boîtes de dialogue pour la sélection d'objets.

## Vue du projet

Une fenêtre du projet vous permet d'afficher la structure du projet relative à l'ensemble des données sur l'outil de développement dans la vue hors ligne, une autre fenêtre du projet vous permettant d'afficher les données correspondantes sur le système cible dans la vue en ligne.

Vous pouvez également sélectionner la vue du gestionnaire de station, lorsque le logiciel optionnel est installé.

---

### Nota

La configuration du matériel et des réseaux ne peut être réalisée que dans l'affichage hors ligne.

---

## 6.2 Informations sur la protection d'accès

A partir de STEP 7 V5.4, vous avez la possibilité de mettre en place une protection d'accès pour les projets et bibliothèques en attribuant un mot de passe au projet. Cette fonctionnalité suppose cependant que SIMATIC Logon est installé.

Vous pouvez en outre activer, désactiver et afficher un journal des modifications.

Si vous avez installé SIMATIC Logon sur votre ordinateur, les commandes de menu suivantes s'affichent de manière dynamique à l'écran dans SIMATIC Manager, afin que vous puissiez gérer la protection d'accès à un projet ou à une bibliothèque :

- Protection d'accès, Activer
- Protection d'accès, Désactiver
- Protection d'accès, Gérer les utilisateurs
- Protection d'accès, Synchroniser dans le multiprojet
- Suppression de la protection d'accès et du journal des modifications

Vous activez la protection d'accès dans SIMATIC Manager, en choisissant la commande de menu **Outils > Protection d'accès, Activer**. Une boîte de dialogue s'ouvre alors, dans laquelle vous attribuez un mot de passe pour le projet. Le projet ou la bibliothèque correspondante peuvent alors uniquement être édités après saisie du mot de passe du projet.

La commande de menu **Supprimer la protection d'accès et le journal des modifications** vous permet de supprimer à la fois la protection d'accès et le journal des modifications d'un projet ou d'une bibliothèques protégés par un mot de passe. Une fois la protection d'accès désactivée, vous pouvez éditer des projets de nouveau également avec une version de STEP 7 antérieure à V5.4.

---

### Nota

- Pour pouvoir activer ou désactiver la protection d'accès, vous devez être authentifié en tant qu'administrateur du projet dans SIMATIC Logon.
  - Lors de la première activation de la protection d'accès, le format du projet est modifié. Un message vous indique que le projet modifié ne peut plus être édité avec des versions plus anciennes de STEP 7.
  - Grâce à la fonction **Outils -> Protection d'accès -> Supprimer la protection d'accès et le journal des modifications**, vous pouvez de nouveau éditer le projet ou la bibliothèque avec une version de STEP 7 inférieure à la version V5.4. Vous perdez cependant l'information sur les utilisateurs ayant accès à ce projet ou à cette bibliothèque ainsi que tous les journaux des modifications.
  - L'utilisateur actuellement connecté est affiché dans la barre d'état de SIMATIC Manager.
  - L'utilisateur du projet actuellement connecté, qui active la protection d'accès, est entré en tant qu'administrateur du projet et doit attribuer le mot de passe du projet.
  - Pour pouvoir ouvrir un projet avec protection d'accès, vous devez être authentifié en tant qu'administrateur du projet ou éditeur du projet dans SIMATIC Logon et la protection d'accès doit être activée ou vous connaissez le mot de passe du projet.
-

## 6.3 Informations sur le journal des modifications

A partir de STEP 7 V5.4, après avoir mis en place une protection d'accès pour les projets et bibliothèques, vous avez la possibilité de tenir un journal des modifications consignnant les actions en ligne.

### Exemples

- Activation / Désactivation / Configuration de la protection d'accès et journal des modifications
- Ouverture / Fermeture de projets et de bibliothèques
- Chargement dans le système cible (données système)
- Opérations sélectionnées pour le chargement et la copie de blocs
- Actions de modification de l'état de fonctionnement
- Effacement général

Vous pouvez afficher le journal des modifications et lui affecter des explications relatives à la modification ainsi que des commentaires. Cette fonctionnalité suppose toutefois que vous avez installé SIMATIC Logon.

Vous activez le journal des modifications dans SIMATIC Manager, en choisissant la commande de menu **Outils > Journal des modifications, Activer**. Vous pouvez ensuite l'afficher ou le désactiver en choisissant la commande de menu correspondante.

Selon l'objet que vous avez sélectionné dans l'arborescence du projet (p. ex. le dossier Projet ou la station subordonnée), le journal des modifications correspondant s'affiche.

---

### Nota

- Grâce à la fonction **Outils > Protection d'accès > Supprimer la protection d'accès et le journal des modifications**, vous pouvez de nouveau éditer le projet ou la bibliothèque avec une version de STEP 7 inférieure à la version V5.4. Vous perdez cependant l'information sur les utilisateurs ayant accès à ce projet ou à cette bibliothèque ainsi que tous les journaux des modifications.
  - Pour pouvoir exécuter cette fonction, vous devez être authentifié en tant d'administrateur du projet dans SIMATIC Logon Admin Tool et la protection d'accès doit être activée pour ce projet.
-

## 6.4 Utilisation de jeux de caractères de langues étrangères

A partir de STEP 7 V5.3 SP2, vous pouvez entrer, dans les projets et bibliothèques, des textes dans une langue étrangère qui ne correspond pas à la langue paramétrée dans STEP 7. A cet effet, vous devez sélectionner les paramètres de langue de Windows correspondants dans le panneau de configuration du système d'exploitation. Ainsi, il est p. ex. possible d'utiliser STEP 7 dans la langue anglaise sur un Windows chinois, tout en saisissant des textes en chinois.

Il faut distinguer les possibilités suivantes de paramétrage de la langue :

### Paramètres de langue de Windows

Vous sélectionnez les paramètres de langue dans le panneau de configuration de Windows. Les textes du système d'exploitation s'affichent dans la langue sélectionnée et vous pouvez entrer des textes dans des jeux de caractères de langues étrangères.

### Langue du projet

La langue du projet correspond à la langue paramétrée dans le panneau de configuration de Windows lors de la création d'un projet. Il n'est pas possible de modifier la langue du projet. Le paramétrage "Indépendant de la langue" permet toutefois d'ouvrir un projet sur des ordinateurs possédant un autre paramétrage de langue de Windows. Avant de modifier la langue du projet en choisissant "Indépendant de la langue", veuillez vous assurer que seuls des caractères du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f) aient été utilisés lors de la saisie de textes dans le projet.

Avec la commande de menu **Edition > Propriétés de l'objet**, vous pouvez afficher la langue dans laquelle un projet ou une bibliothèque ont été créés. Dans la même boîte de dialogue, vous pouvez également sélectionner l'option "Peut être ouvert sous tous les paramètres linguistiques de Windows (indépendant de la langue)".

Lorsque vous copiez un projet avec la commande de menu **Enregistrer sous** et que la langue du projet ne correspond pas au paramètre linguistique actuel de Windows, vous pouvez changer la langue dans le projet copié en choisissant la langue actuelle paramétrée dans Windows. Ceci s'avère utile lorsque vous souhaitez p. ex. créer des variantes d'un projet spécifiques à la langue. Il est recommandé que le projet maître contienne exclusivement des caractères du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f), de sorte à pouvoir garantir qu'aucune corruption de données ne se produise lors de l'édition du projet dans la langue respective.

### Langue de STEP 7

Vous paramétrez la langue de STEP 7 dans SIMATIC Manager, en choisissant la commande de menu **Outils > Paramètres**. Les éléments de l'interface, les commandes de menu, les boîtes de dialogue et les messages d'erreur de STEP 7 s'affichent dans cette langue.

Si vous utilisez une autre langue de Windows que le français, l'anglais, l'allemand, l'italien ou l'espagnol, vous pouvez vous assurer que l'interface utilisateur de STEP 7 est représentée correctement en choisissant l'anglais comme langue de STEP 7.

## Règles

Si vous avez l'intention d'éditer vos projets ou bibliothèques sur des ordinateurs possédant des paramètres de langue différents, tenez compte des règles suivantes afin d'éviter des incompatibilités ou les altérations de données lors de l'utilisation de jeux de caractères de langues étrangères :

- Installez STEP 7 uniquement dans des répertoires contenant des caractères du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f).
- Utilisez exclusivement des noms et des chemins de projets contenant des caractères du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f). Si vous utilisez p. ex. des trémas allemands ou des caractères cyrilliques ou chinois, le projet peut uniquement être ouvert sur des ordinateurs sur lesquels les paramètres de langue correspondants de Windows ont été sélectionnés.
- Dans les multiprojets, utilisez exclusivement des projets et des bibliothèques possédant la même langue de projet ou caractérisés comme indépendants de la langue. Le multiprojet proprement dit est indépendant de la langue.
- Créez toujours des bibliothèques indépendantes de la langue afin de garantir leur utilisation sur des ordinateurs avec différents paramètres de langue de Windows. Pour l'attribution des noms aux objets des bibliothèques, les commentaires, les noms symboliques, etc., utilisez uniquement des caractères ASCII (0x2a - 0x7f) afin de pouvoir utiliser les bibliothèques sans problèmes.
- Lors de l'importation/exportation de configurations matérielles ou de tables des mnémoniques, veillez à importer/exporter exclusivement des fichiers avec des langues compatibles.
- Dans les noms des attributs définis par l'utilisateur, utilisez exclusivement des caractères du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f).
- Si, dans une source LIST, vous utilisez des caractères qui ne font pas partie du jeu de caractères anglais (caractères ASCII 0x2a - 0x7f) pour les entrées des propriétés de bloc TITLE, AUTHOR, FAMILY ou NAME, mettez simplement ces entrées entre guillemets.

---

**Nota**

- Si vous modifiez ou copiez des projets ou des bibliothèques ayant été créés sur un ordinateur qui, bien qu'il soit caractérisé comme indépendant de la langue en ce qui concerne les paramètres de langue de Windows, n'est toutefois pas compatible avec l'ordinateur actuel, des corruptions de données risquent de se produire si dans le projet ou la bibliothèque des caractères absents du jeu de caractères anglais (caractères ASCII 0x2a – 0x7f) ont été utilisés.  
Avant l'édition de projets ou de bibliothèques "étrangers", vérifiez par conséquent si les paramètres de langue de Windows sur votre ordinateur correspondent à la langue de projet.
  - Si vous exportez des configurations matérielles ou des tables de mnémoniques qui doivent être importées avec un autre paramètre de langue de Windows, assurez-vous que seuls des caractères ASCII (0x2a - 0x7f) ont été utilisés et pas de caractères spécifiques à la langue, tels que p. ex. des trémas allemands, des caractères japonais ou cyrilliques.
  - Les configurations matérielles ou tables des mnémoniques exportées contenant des caractères spécifiques à la langue, tels que p. ex. des trémas allemands, des caractères japonais ou cyrilliques, doivent uniquement être importées avec le paramètre de langue de Windows avec lequel elles ont été exportées. Lorsque p. ex. vous importez d'anciennes tables de mnémoniques contenant probablement des caractères spécifiques à la langue, vérifiez soigneusement le résultat : les mnémoniques doivent être univoques, ne pas contenir de point d'interrogation et être plausibles.
  - Dans les tables de mnémoniques contenant des caractères spéciaux qui ne sont pas définis pour le paramètre de langue respectif de Windows, les points d'interrogation risquent d'entraîner des modifications du nom des mnémoniques et des erreurs lors du tri d'après le nom et le commentaire.
  - Lors de l'adressage symbolique, vous devez écrire les noms symboliques entre guillemets ("`<nom symbolique>`").
- 

**Marche à suivre**

Afin de pouvoir saisir dans les projets et bibliothèques, des textes dans des jeux de caractères de langues étrangères, procédez de la manière suivante :

1. Paramétrez la langue souhaitée dans le panneau de configuration de Windows.
2. Créez un projet.
3. Saisissez les textes souhaités dans la langue étrangère.

Dans le cas de projets et de bibliothèques dont la création est antérieure à STEP 7 V5.3 SP2, la langue du projet est "non encore définie". En choisissant la commande de menu **Edition > Propriétés de l'objet** pour ces projets, vous pouvez sélectionner comme langue du projet, la langue de Windows actuellement paramétrée. Assurez-vous auparavant qu'aucun caractère non défini dans le paramètre de langue actuel de Windows n'ait été utilisé.

## 6.5 Paramétrage de la langue de Windows

Pour paramétrer la langue de Windows, procédez de la manière suivante :

### Paramétrage de la langue sous Windows XP et Windows Server 2003 :

1. Paramétrez la langue d'affichage souhaitée dans le **Panneau de configuration > Options régionales et linguistiques > Options avancées > Langue pour les programmes non Unicode**.
2. Paramétrez la langue d'entrée dans le **Panneau de configuration > Options régionales et linguistiques > Langues > Détails (langue d'entrée par défaut)**.
3. Paramétrez la langue souhaitée dans le **Panneau de configuration > Options régionales et linguistiques > Options régionales (Standards et formats)**.

Ce n'est qu'après avoir effectué tous les paramétrages que vous pouvez entrer afficher correctement les textes dans la langue souhaitée.

### Paramétrage de la langue sous Windows 2000 :

1. Paramétrez la langue d'affichage souhaitée dans le **Panneau de configuration > Options régionales > Général > Vos paramètres régionaux (emplacement)**.
2. Paramétrez la langue d'affichage souhaitée dans le **Panneau de configuration > Options régionales > Général > Valeurs par défaut**.
3. Paramétrez la langue d'entrée dans le **Panneau de configuration > Options régionales > Paramètres régionaux d'entrée**.

Ce n'est qu'après avoir effectué tous les paramétrages que vous pouvez entrer afficher correctement les textes dans la langue souhaitée.



## 6.6 Création d'un projet

### 6.6.1 Création d'un projet

Pour réaliser votre tâche d'automatisation au sein d'un gestionnaire de projets, vous devez d'abord créer un nouveau projet. Il sera créé dans le répertoire que vous avez sélectionné pour les projets, lorsque vous avez choisi la commande **Outils > Paramètres** et l'onglet "Général".

#### Nota

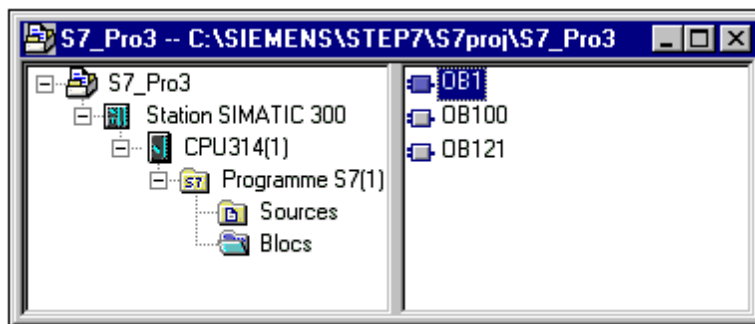
SIMATIC Manager vous permet d'attribuer des noms dont le nombre de caractères est supérieur à 8. Le nom du répertoire du projet est tronqué après 8 caractères. Les noms de projets doivent de ce fait se distinguer dans leur 8 premiers caractères. Aucune différenciation n'est faite entre les majuscules et minuscules.

La marche à suivre pour créer un projet est décrite étape par étape dans Création manuelle d'un projet ou Création d'un projet à l'aide de l'assistant .

### Création d'un projet à l'aide de l'assistant

Le plus simple pour créer un nouveau projet, c'est d'avoir recours à l'assistant "Nouveau projet". Pour l'appeler, choisissez la commande **Fichier > Assistant "Nouveau projet"**. Il vous posera les questions nécessaires dans des boîtes de dialogue et créera le projet pour vous. Outre la station, la CPU, les dossiers Programmes, Sources et Blocs ainsi que l'OB1, vous pouvez déjà y sélectionner les OB de traitement d'erreurs et d'alarmes.

La figure suivante illustre un projet créé à l'aide de l'assistant.



### Création manuelle d'un projet

Vous avez également la possibilité de créer un nouveau projet en choisissant la commande **Fichier > Nouveau** dans SIMATIC Manager. Ce projet contient déjà l'objet "Sous-réseaux MPI".

## Différentes façons de poursuivre

Vous disposez d'une grande liberté d'action pour la suite du traitement de votre projet. Une fois votre projet créé, vous pouvez par exemple poursuivre votre travail en

- configurant d'abord le matériel, puis en créant le logiciel correspondant ou
- en créant d'abord le logiciel indépendamment d'un matériel configuré.

### Solution 1 : commencer par configurer le matériel

Si vous souhaitez commencer par la configuration matérielle, procédez comme décrit dans le volume 2 du manuel "Configuration matérielle avec STEP 7". Après la configuration, les programmes S7 ou M7 requis pour la création du logiciel seront déjà insérés. Poursuivez ensuite en insérant les objets requis pour la création du programme. Ecrivez ensuite le logiciel destiné aux modules programmables.

### Solution 2 : commencer par écrire le logiciel

Vous pouvez, même sans configuration matérielle préalable, créer le logiciel et procéder à la configuration ultérieurement. Pour saisir des programmes, il n'est pas nécessaire que la configuration matérielle d'une station soit fixée.

Procédez de la manière suivante :

1. Insérez les dossiers du logiciel requis Programme S7/M7 sans station ni CPU dans votre projet.  
Vous décidez uniquement si le dossier doit contenir des programmes pour du matériel S7 ou pour du matériel M7.
2. Ecrivez ensuite le logiciel destiné aux modules programmables.
3. Configurez le matériel.
4. Après avoir configuré le matériel, affectez le programme M7 ou S7 à une CPU.

## 6.6.2 Insertion de stations

Dans un projet, la station représente la configuration matérielle de l'automate programmable et contient les données pour la configuration et le paramétrage des divers modules.

Les nouveaux projets créés par l'assistant "Nouveau projet" contiennent déjà une station. Vous pouvez également créer la station en choisissant la commande **Insertion > Station**.

Vous pouvez choisir les stations suivantes :

- station SIMATIC 300,
- station SIMATIC 400,
- station SIMATIC H,
- station SIMATIC PC,
- PC/PG,
- SIMATIC S5,
- autre station, c'est-à-dire non SIMATIC S7/M7, SIMATIC S5.

Les stations sont alors insérées avec une désignation par défaut (par exemple, station SIMATIC300 (1), station SIMATIC300 (2), etc.). Vous pouvez remplacer ces désignations par un nom plus évocateur.

La marche à suivre pour l'insertion est décrite étape par étape dans Insertion d'une station.

### Réalisation de la configuration matérielle

Dans la configuration matérielle, vous utilisez un catalogue des modules pour définir la CPU et tous les modules contenus dans sa commande. Vous démarrez la configuration matérielle par double clic sur la station.

Une fois que vous avez sauvegardé et quitté la configuration matérielle, un programme S7 ou M7 est automatiquement créé comme dossier du logiciel ainsi qu'une table de liaisons (objet "Liaisons"), et ce pour chaque module programmable que vous avez créé lors de la configuration. Les projets créés par l'assistant "Nouveau projet" contiennent déjà ces objets.

La marche à suivre pour la configuration est décrite étape par étape dans Configuration du matériel, des informations détaillées sont données dans Marche à suivre pour la configuration d'une station.

### Création de la table des liaisons

Une table de liaisons vide (objet "Liaisons") est automatiquement créée pour chaque module programmable. Elle est utilisée pour la définition de liaisons de communication entre modules programmables au sein d'un réseau. A son ouverture apparaît une fenêtre contenant une table pour la définition de liaisons entre modules programmables.

Des informations détaillées sont données dans Mise en réseau de stations au sein d'un projet.

### Étapes suivantes

Après avoir réalisé la configuration matérielle, vous pouvez créer le logiciel pour vos modules programmables (voir aussi Insertion d'un programme S7/M7).

### 6.6.3 Insertion d'un programme S7/M7

Le logiciel destiné aux modules programmables est stocké dans des dossiers d'objets. Pour les modules SIMATIC S7, un tel dossier d'objets s'appelle "ProgrammeS7", pour les modules SIMATIC M7, "ProgrammeM7".

La figure suivante montre l'exemple d'un programme S7 dans un module programmable d'une station SIMATIC 300.



#### Composants déjà créés

Un programme S7/M7 est automatiquement généré comme dossier du logiciel pour chaque module programmable.

Dans un programme S7 figurent déjà :

- une table des mnémoniques (objet "Mnémoniques"),
- un dossier "Blocs" pour les blocs, contenant le premier bloc,
- un dossier "Sources" pour des programmes source.

Dans un programme M7 figurent déjà :

- une table des mnémoniques (objet "Mnémoniques"),
- un dossier "Blocs".

#### Création de blocs S7

Si vous souhaitez écrire des programmes LIST, CONT ou LOG, vous sélectionnez l'objet "Blocs" déjà créé et choisissez ensuite la commande **Insertion > Bloc S7**. Le menu suivant vous permet de choisir le type de bloc : par exemple, bloc de données, type de données utilisateur (UDT), fonction, bloc fonctionnel, fonction, bloc d'organisation, table des variables (VAT).

Vous saisissez votre programme LIST, CONT ou LOG dans le bloc (vide) qui s'ouvre alors. De plus amples informations sont données dans Marche à suivre pour la création de blocs de code ainsi que dans les manuels traitant des langages LIST, CONT et LOG.

---

#### Nota

L'objet Données système (SDB), que vous trouverez éventuellement dans des programmes utilisateur, est créé par le système. Vous pouvez l'ouvrir, mais vous ne pouvez pas en modifier le contenu pour des raisons de cohérence. Il sert à modifier la configuration après le chargement d'un programme et à charger ces modifications dans le système cible.

---

## Blocs tirés de bibliothèques standard

Pour créer vos programmes utilisateur, vous pouvez aussi utiliser des blocs tirés des bibliothèques standard faisant partie du logiciel. Vous accédez aux bibliothèques par la commande **Fichier > Ouvrir**. L'aide en ligne vous donnera des renseignements complémentaires sur l'utilisation des bibliothèques standard ainsi que sur la création de vos propres bibliothèques sous Utilisation de bibliothèques.

## Création de sources et diagrammes CFC

Si vous souhaitez créer une source dans un langage de programmation donné ou un diagramme CFC, vous sélectionnez l'objet "Sources" ou "Diagrammes" dans le programme S7 et activez ensuite la commande **Insertion > Logiciel S7**. Le menu suivant vous permet de choisir la source correspondant au langage de programmation. Vous pouvez saisir le programme une fois la source vide ouverte. De plus amples informations sont données dans Principes de la programmation dans les sources LIST.

## Création de programmes pour M7

Si vous souhaitez créer des programmes pour le système d'exploitation RMOS d'un module programmable de la gamme M7, vous sélectionnez le programme M7 et choisissez la commande **Insertion > Logiciel M7**. Le menu suivant vous permet de choisir l'objet correspondant au langage de programmation ou au système d'exploitation. Une fois l'objet créé ouvert, vous parvenez à l'environnement de développement correspondant.

## Création de table des mnémoniques

Une table des mnémoniques vide (objet "Mnémoniques") est automatiquement générée lors de la création d'un programme S7 ou M7. Son ouverture entraîne également celle de la fenêtre de l'éditeur de mnémoniques et l'affichage de la table des mnémoniques qu'elle contient. De plus amples informations sont données dans Saisie de plusieurs mnémoniques globaux dans la table des mnémoniques.

## Insertion de sources externes

Vous pouvez créer et éditer des fichiers source avec des éditeurs ASCII quelconques. Il est ensuite possible d'importer ces fichiers dans un projet et de les compiler en blocs individuels.

Les blocs créés lors de la compilation d'une source importée sont placés dans le dossier Blocs.

De plus amples informations sont données dans Insertion d'une source externe.

## 6.7 Edition d'un projet

### Ouverture d'un projet

Pour ouvrir un projet, choisissez d'abord la commande **Fichier > Ouvrir**. Sélectionnez ensuite un projet dans les boîtes de dialogue suivantes. La fenêtre de projet s'ouvre alors.

---

#### Nota

Si ce projet ne figure pas dans la liste de projets proposée, cliquez sur le bouton "Parcourir". Dans la boîte de dialogue correspondante, vous pouvez chercher d'autres projets et reporter les projets trouvés dans la liste des projets. Vous pouvez modifier les entrées dans la liste de projets en choisissant la commande **Fichier > Gérer**.

---

### Copie d'un projet

Vous copiez un projet en l'enregistrant sous un autre nom via la commande **Fichier > Enregistrer sous**.

Vous copiez les éléments de projet comme les stations, programmes, blocs etc. en choisissant la commande **Edition > Copier**.

La marche à suivre pour copier un projet est décrite étape par étape dans Copie d'un projet et Copie d'un élément de projet.

### Suppression d'un projet

Vous supprimez un projet en choisissant la commande **Fichier > Supprimer**.

Vous supprimez des éléments de projet comme les stations, programmes, blocs etc. en choisissant la commande **Edition > Effacer**.

La marche à suivre pour supprimer un projet est décrite en détail dans Suppression d'un projet et Suppression d'un élément de projet.

### 6.7.1 Recherche des logiciels requis pour un projet

Lorsque vous éditez un projet contenant des objets créés avec un autre logiciel, vous avez besoin de ce dernier.

Quel que soit l'outil de développement avec lequel vous éditez vos multiprojets, projets ou bibliothèques, STEP 7 affiche les logiciels requis ainsi que leur version.

Les informations relatives aux logiciels sont complètes lorsque :

- Le projet (ou tous les projets dans le multiprojet) ou la bibliothèque ont été créés avec STEP 7 à partir de la version V5.2.
- Vous avez recherché les logiciels requis pour un projet. Sélectionnez à cet effet le projet dans SIMATIC Manager, puis choisissez la commande de menu **Edition > Propriétés de l'objet**. Dans la boîte de dialogue qui s'ouvre, sélectionnez l'onglet "Logiciels requis". Les informations affichées dans cette boîte de dialogue précisent si vous devez rechercher les logiciels requis pour le projet.

## 6.8 Gestion multilingue des textes

STEP 7 permet d'exporter des textes stockés dans un projet en une seule langue, pour les faire traduire, puis les réimporter et les afficher dans la langue de traduction.

Les types de textes suivants autorisent cette gestion multilingue.

- Titres et commentaires :
  - titres de bloc et commentaires de bloc,
  - titres de réseau et commentaires de réseau,
  - commentaires de ligne dans les programmes LIST,
  - commentaires tirés des tables de mnémoniques, des tables de déclaration de variables, des types de données utilisateur et des blocs de données,
  - commentaires, noms d'état et noms de transition dans les programmes HiGraph,
  - extensions des noms d'étape et des commentaires d'étape dans les programmes GRAPH.
- Textes affichés :
  - textes de message générés par STEP 7, GRAPH, HiGraph, PDIAG ou ProTool
  - bibliothèques de textes système
  - bibliothèques de textes utilisateur
  - textes personnalisés
  - textes utilisateur

### Exportation

L'exportation est effectuée pour tous les blocs et tables de mnémoniques se trouvant sous l'objet sélectionné. Un fichier d'exportation est généré pour chaque type de texte. Il contient une colonne pour la langue source et une pour la langue cible. Il est interdit de modifier les textes dans la langue source.

## Importation

L'importation consiste à adopter dans l'objet sélectionné le contenu des colonnes de la langue cible (colonne droite). Seules sont adoptés les traductions dont les textes source (textes exportés) correspondent avec une entrée existante dans dans la colonne "Langue source".

## Changement de langue

Vous pouvez choisir ici toutes les langues que vous avez indiquées lors de l'importation dans le projet sélectionné. Le changement de langue pour "Titre et commentaires" s'applique uniquement à l'objet sélectionné. Celui pour "Textes" s'applique toujours à l'ensemble du projet.

## Effacer langue

Lorsque vous effacez une langue, tous les textes traduits dans cette langue sont effacés dans la base de données interne.

Il est recommandé de toujours sélectionner une langue de référence dans le projet. Il peut par exemple s'agir de votre langue nationale. N'effacez jamais cette langue. Lors de l'exportation et de l'importation, indiquez toujours cette langue de référence comme langue source. Choisissez la langue cible que vous souhaitez.

## Réorganiser

Lors de la réorganisation, la langue est remplacée par la langue actuellement paramétrée. Il s'agit de la langue que vous avez sélectionnée comme "Langue des nouveaux blocs" dans les paramètres de traitement de commentaires. La réorganisation concerne uniquement le titre et les commentaires.

## Traitement des commentaires

Vous pouvez sélectionner la manière dont les commentaires des blocs être traités dans des projets à gestion multilingue de textes.

## Marche à suivre





### 6.8.1 Types de textes à gestion multilingue

A l'exportation, un fichier est créé pour chaque type de texte. Le nom de ce fichier est celui du type de texte et son extension est le format d'exportation (TypeTexte.Format : par ex. SymbolComment.CSV ou SymbolComment.XLS). Les fichiers ne respectant pas les conventions de nom ne peuvent être utilisés comme source ou comme cible.

Les textes traduisibles au sein d'un projet sont classés en types de texte suivants :

Type de texte	Signification
BlockTitle	Titre de bloc
BlockComment	Commentaire de bloc
NetworkTitle	Titre de réseau
NetworkComment	Commentaire de réseau
LineComment	Ligne de commentaire dans LIST
InterfaceComment	Commentaire de Var_Section (table de déclaration dans les blocs de code) Commentaire d'UDT (type de données utilisateur) Commentaire de bloc de données
SymbolComment	Commentaire de mnémonique
S7UserTexts	Textes saisis par l'utilisateur pouvant être affichés sur des visuels
S7SystemTextLibrary	Textes de bibliothèques de textes système pouvant être intégrés à des messages, actualisés de manière dynamique par rapport au temps de cycle et affichés sur la PG ou sur d'autres visuels
S7UserTextLibrary	Textes de bibliothèques de textes utilisateur pouvant être intégrés à des messages, actualisés de manière dynamique par rapport au temps de cycle et affichés sur la PG ou sur d'autres visuels
HiGraphStateName	<b>S7-HiGraph</b> Nom d'état
HiGraphStateComment	Commentaire d'état
HiGraphTansitionName	Nom de transition
HiGraphTransitionComment	Commentaire de transition
S7GraphStateName	<b>S7-GRAPH</b> Extension de nom d'étape
S7GraphStateComment	Commentaire d'étape

Il peut encore exister d'autres types de textes spécifiques à une application issus d'éditeurs d'autres progiciels optionnels (p. ex. Protocol, WinCC,.....). Ils ne sont pas décrits ici.

## 6.8.2 Structure du fichier d'exportation

Le fichier d'exportation est toujours construit de la manière suivante.

Exemple :

<code>\$_Langages</code>		
<code>7(1) Français (France)</code>	<code>9(1) Anglais (USA)</code>	
<code>\$_Typ(NetworkTitle)</code>		
<code>//Longueur max. du texte : 64 caractères</code>		
<code>//\$_Exportation le 16.11.2005 13:14:34</code>		
Première chaîne de caractères à traduire	Traduction	<code>test\Programme S7(1)\Blocs\OB1</code>
Seconde chaîne de caractères	Traduction	<code>test\Programme S7(1)\Blocs\OB1</code>
Chaîne de caractères qui ne doit pas être affichée dans la traduction	<code>\$_hide</code>	<code>test\Programme S7(1)\Blocs\OB1</code>

Langue source

Langue cible

Occurrence

Voici les règles fondamentales à connaître :

- Il est interdit de modifier, d'écraser ou d'effacer :
  - les champs précédés d'un "\$\_" (ce sont des mots-clé),
  - les numéros indiquant la langue (dans l'exemple ci-dessus, 7(1) pour la langue source Français (France) et 9(1) pour la langue cible Anglais).
- Un fichier contient toujours des textes du même type. Dans notre exemple, il s'agit du type de texte Titre de reseau (`$_Typ(NetworkTitle)`). Les règles pour le traducteur qui édite ce fichier sont rangées dans le texte introductif du fichier d'exportation.
- Les informations complémentaires sur les textes ou commentaires doivent toujours figurer avant la définition du type (`$_Typ...`) ou après la dernière colonne.

### Nota

Si la colonne de la langue cible contient seulement "512(32) \$\_Undefined", c'est qu'il n'en a pas été indiqué lors de l'exportation. Pour plus de clarté, vous pouvez remplacer cette chaîne par la langue cible, "9(1) Anglais (USA)" par exemple. Dans ce cas, il faudra contrôler la langue cible proposée à l'importation et la choisir de nouveau au besoin.

Vous pouvez masquer des textes que vous ne souhaitez pas afficher dans la langue cible en utilisant le mot-clé `$_hide`. Ceci ne s'applique ni aux commentaires de variables (`InterfaceComment`), ni aux commentaires de mnémoniques (`SymbolComment`).

## Format du fichier d'exportation

Vous pouvez choisir le format dans lequel vous souhaitez stocker les fichiers d'exportation.

Si vous choisissez le format CSV, sachez que lors de leur édition avec EXCEL, ce dernier n'ouvre un fichier CSV correctement que par la commande du menu Fichier. **L'ouverture d'un fichier CSV par double-clic dans l'Explorateur le rend fréquemment inutilisable.**

L'édition de fichiers CSV avec EXCEL sera plus facile si vous avez recours au procédé suivant :

1. Ouvrir les fichiers d'exportation avec EXCEL.
2. Enregistrer les fichiers en tant que fichiers XLS.
3. Traduire les textes dans les fichiers XLS.
4. Enregistrer les fichiers XLS en format CSV avec EXCEL.

---

### Nota

Il est interdit de renommer les fichiers d'exportation.

---

### 6.8.3 Gestion de textes utilisateur dont la police de langue n'est pas installée

Vous avez également la possibilité d'exporter, de compiler, d'importer puis d'enregistrer dans votre projet, des textes utilisateur dont la police de langue n'est pas installée dans votre système d'exploitation.

Cependant, vous pouvez uniquement afficher ces textes sur un ordinateur sur lequel la police de langue correspondante est installée.

Pour traduire des textes utilisateurs en russe, p. ex., procédez de la manière suivante lorsque la police cyrillique n'est pas installée :

1. Exportez les textes utilisateur à traduire avec comme langue source l'allemand et comme langue cible le russe.
2. Envoyez le fichier d'exportation au traducteur sur l'ordinateur duquel la police cyrillique est installée.
3. Importez les fichiers d'exportation traduits.  
**Résultat** : le projet existe à présent dans plusieurs langues (allemand et russe) sur votre ordinateur.
4. Enregistrez le projet global et envoyez-le au client final qui aura la possibilité d'afficher les textes russes sur son ordinateur sur lequel la police cyrillique est installée.

## 6.8.4 Informations sur le fichier-journal

Les messages d'erreur et avertissements résultant de la gestion multilingue des textes sont édités dans un fichier-journal de format TXT. Ce fichier se trouve dans le même répertoire que les fichiers d'exportation.

Les messages sont généralement implicites. Voici toutefois quelques explications complémentaires :



### Avertissement

Le texte 'xyz' existe déjà dans le fichier 'xyz'. Les autres occurrences du texte seront ignorées.

### Explication

Un texte est utilisé en tant que clé pour la traduction, quelle que soit sa langue. Lorsqu'un texte identique est utilisé dans plusieurs langues ou alors plusieurs fois dans une même langue, il ne peut plus être affecté de manière univoque et n'est de ce fait pas traduit.

Exemple :

\$ _Languages	
7(1) Deutsch (Deutschland)	9(1) Englisch (USA)
kein	none
keine	none
keiner	none

Langue source                      Langue cible

Ceci est valable uniquement pour les titres et les commentaires.

### Solution

Renommez les textes concernés dans le fichier de texte exporté (dans l'exemple, il faut utiliser un seul terme au lieu des 3 termes allemands différents) et importez de nouveau les textes.

### 6.8.5 Optimisation du modèle à traduire

Vous pouvez simplifier le "matériau source" de la traduction en unifiant les termes ou expressions ayant le même sens.

#### Exemple

Avant la simplification (fichier d'exportation)

\$_Langues		
7(1) Français (France)	7 (1) Français (France)	
\$_Typ(SymbolComment)		
ValidAuto		
Validation automatique		
Validation auto		

Langue source

Langue cible

Réduction à une même expression :

\$_Langues		
7(1) Français (France)	7 (1) Français (France)	
\$_Typ(SymbolComment)		
Valid auto	Validation auto	
Validation automatique	Validation auto	
Validation auto	Validation auto	

Langue source

Langue cible

Après la simplification (c'est-à-dire après importation suivie d'une exportation) :

\$_Langues		
7(1) Français (France)	7 (1) Français (France)	
\$_Typ(SymbolComment)		
Validation auto	Validation auto	

Langue source

Langue cible

### 6.8.6 Optimisation de la traduction

Pour les projets dont la structure et les textes sont analogues à ceux d'un projet précédent, vous pouvez optimiser l'opération de traduction.

Le procédé décrit ci-après est recommandé en particulier pour les projets que vous avez créés par copie et modification ultérieure.

#### Condition requise

Il y a déjà une cible d'exportation.

#### Marche à suivre

1. Copiez les fichiers d'exportation dans le répertoire du nouveau projet à traduire.
2. Ouvrez le nouveau projet et exportez les textes (commande **Outils > Gestion multilingue des textes > Exporter**)  
La cible d'exportation existant déjà, un dialogue vous demande s'il faut la compléter ou la remplacer.
3. Cliquez sur le bouton "Compléter".
4. Faites traduire les fichiers d'exportation (seuls les nouveaux textes sont à traduire).
5. Importez pour finir les fichiers traduits.

## 6.9 Carte mémoire micro (MMC) en tant que support de données

### 6.9.1 Ce qu'il faut savoir sur les cartes mémoire micro (MMC)

Les cartes mémoire micro (MMC) sont des cartes enfichables caractérisées par leur forme extrêmement compacte que vous pouvez utiliser, par exemple, pour une CPU 31xC ou un IM 151/CPU (ET 200S).

Le nouveau concept mémoire des cartes MMC est décrit ci-après.

#### Contenu de la carte MMC

La carte MMC assure aussi bien la fonction de mémoire de chargement que la fonction de support de données.

#### Carte mémoire micro (MMC) en tant que mémoire de chargement

Les cartes MMC contiennent la **mémoire de chargement** complète d'une CPU adaptée pour les cartes MMC. Celle-ci contient le programme avec les blocs (OB, DB, FC, ...) de même que la configuration matérielle. Ces contenus ont une influence sur le mode de fonctionnement de la CPU. Lorsque la carte mémoire micro est utilisée comme mémoire de chargement, il est possible de transférer des blocs, de même que la configuration matérielle à l'aide de fonctions de chargement (p. ex. **Charger** dans CPU). Les blocs chargés dans la CPU sont immédiatement actifs, la configuration matérielle l'étant après le démarrage de la CPU.

#### Comportement après un effacement général

Après un effacement général, les blocs se trouvent toujours sur la carte MMC.

#### Chargement et effacement

Vous pouvez écraser les blocs sur la carte MMC.

Vous pouvez effacer les blocs sur la carte MMC.

Vous ne pouvez pas restaurer les blocs écrasés ou effacés.

#### Accès aux blocs de données sur la carte MMC

Vous avez la possibilité de traiter les blocs de données et leur contenu sur la carte MMC, ce qui s'avère tout particulièrement utile pour de grandes quantités de données ou pour des données rarement utilisées dans le programme utilisateur. Vous disposez à cet effet des nouvelles fonctions système :

- SFC 82 : Création d'un bloc de données dans la mémoire de chargement
- SFC 83 : Lecture d'un bloc de données dans la mémoire de chargement
- SFC 84 : Ecriture dans un bloc de données dans la mémoire de chargement

## Carte MMC et protection par mot de passe

Si une CPU équipée d'une carte mémoire micro (MMC) (p. ex. une CPU de la famille 300-C) est protégée par mot de passe, ce dernier est également demandé dans SIMATIC Manager, à l'ouverture de cette carte mémoire micro sur la PG ou le PC.

## Affichage de l'occupation mémoire avec STEP 7

Dans la boîte de dialogue de l'état du module (onglet "Mémoire"), l'affichage de l'occupation de la mémoire de chargement regroupe l'EPROM et la RAM.

Sur la carte MMC, le comportement des blocs correspond à 100% à celui de l'EPROM.

### 6.9.2 Utilisation d'une micro-carte mémoire (MMC) comme support de données

Avec STEP 7, vous pouvez utiliser la micro-carte mémoire (MMC) SIMATIC comme un support de données externe tout à fait normal.

Dans la mesure où vous avez choisi une micro-carte mémoire (MMC) de taille suffisante, vous pouvez y transférer toutes les données visibles dans l'Explorateur de fichiers du système d'exploitation

Vous pouvez ainsi mettre à disposition de vos collaborateurs tous les dessins, instructions d'utilisation, descriptions fonctionnelles, etc. supplémentaires de votre installation.

### 6.9.3 Fichier carte mémoire

Les fichiers carte mémoire (\*.wld) sont créés pour :

- l'automate logiciel (Software PLC) **WinLC** (WinAC Base et WinAC RTX) et
- les cartes CPU enfichées dans le PC (SlotPLCs) **CPU 41x-2 PCI** (WinAC Slot 412 et WinAC Slot 416).

Vous pouvez enregistrer les blocs et données système pour WinLC ou la CPU 41x-2 PCI dans un fichier carte mémoire comme vous le faites avec une carte mémoire S7. Le contenu de ces fichiers équivaut à celui d'une carte mémoire correspondante de CPU S7.

Vous avez la possibilité de charger ce fichier dans la mémoire de chargement d'un WinLC ou de la CPU 41x-2 PCI à l'aide du panneau de commande. Ceci correspond au chargement du programme utilisateur avec STEP 7.

Avec les CPU 41x-2 PCI, ce fichier peut être chargé automatiquement lors du démarrage du système d'exploitation du PC, lorsque la CPU 41x-2 PCI ne possède pas de mémoire de sauvegarde et fonctionne uniquement avec une carte RAM (fonction "Autoload").

Windows considère les fichiers carte mémoire comme des fichiers "classiques" que vous pouvez déplacer, effacer ou copier sur un support de données avec l'Explorateur.

De plus amples informations sont fournies dans la documentation correspondante relative aux produits WinAC.



#### **6.9.4 Enregistrement des données du projet sur une micro-carte mémoire (MMC)**

Avec STEP 7, vous pouvez enregistrer les données de votre projet STEP 7 et d'autres données quelconques (p. ex. fichiers WORD ou Excel) sur une micro-carte mémoire (MMC) SIMATIC d'une CPU adaptée ou de votre PG/PC. L'avantage réside dans le fait que vous pouvez ainsi également accéder aux données du projet avec des consoles de programmation sur lesquelles le projet n'est pas enregistré.

##### **Conditions**

L'enregistrement de données du projet sur une micro-carte mémoire MMC est uniquement possible si celle-ci est enfichée dans le logement correspondant de la CPU adaptée ou de votre PG/PC et si une liaison en ligne est établie.

La taille de la micro-carte mémoire doit être suffisante pour les données à enregistrer.

##### **Données pouvant être enregistrées sur une micro-carte mémoire (MMC)**

Dans la mesure où vous avez choisi une micro-carte mémoire (MMC) de taille suffisante, vous pouvez y transférer toutes les données visibles dans l'Explorateur de fichiers du système d'exploitation, comme p. ex. :

- données de projet STEP 7 complètes,
- configurations de stations,
- tables des mnémoniques,
- blocs et sources,
- texte multilingue,
- autres données quelconques, comme p. ex. des fichiers WORD ou Excel.



## 7 Edition de projets avec des versions différentes de STEP 7

### 7.1 Edition de projets et bibliothèques de la version 2

La version V5.2 de STEP 7 **ne prend plus** en charge la **modification de projets et de bibliothèques de la version 2**. Si vous éditez des projets ou bibliothèques de cette version, des incohérences risquent de se produire qui rendraient impossible l'édition avec des versions plus anciennes de STEP 7.

Pour continuer à éditer des projets ou des bibliothèques de la version 2, il faut avoir recours à une version de STEP 7 antérieure à V5.1.

### 7.2 Extension d'esclaves DP créés avec des versions antérieures de STEP 7

#### Constellations pouvant résulter de l'addition de nouveaux fichiers GSD

Il est possible d'ajouter de nouveaux esclaves DP au catalogue du matériel de HW Config en installant de nouveaux fichiers GSD. Une fois l'installation effectuée, ils sont disponibles dans le dossier "Autres appareils de terrain".

Vous ne pouvez plus modifier ou étendre comme d'habitude la configuration d'un esclave DP modulaire lorsque

- il a été configuré avec une version antérieure de STEP 7 et que
- il n'a pas été représenté dans le catalogue du matériel par un fichier GSD, mais par un fichier de type et que
- un nouveau fichier GSD a écrasé l'ancienne installation.

#### Solution

Si vous voulez utiliser l'esclave DP **avec de nouveaux modules** qui sont décrits dans le fichier GSD :

- Effacez l'esclave DP et configurez-le de nouveau - il ne sera plus décrit alors par le fichier de type, mais entièrement par le fichier GSD.

Si vous comptez utiliser l'esclave DP **sans les nouveaux modules** qui ne sont décrits que dans le fichier GSD :

- Sélectionnez le dossier "Autres APPAREILS DE TERRAIN/Esclaves DP PROFIBUS compatibles" sous PROFIBUS DP dans la fenêtre "Catalogue du matériel". C'est là que STEP 7 range les "anciens" fichiers de type lorsqu'ils sont remplacés par de nouveaux fichiers GSD. Vous y trouverez les modules permettant l'extension de l'esclave DP déjà configuré.

### **Constellation après un remplacement des fichiers de type par des fichiers GSD dans STEP 7 V5.1 Servicepack 4**

A partir de STEP 7 V5.1, Servicepack 4, les fichiers de types ont été presque tous remplacés par des fichiers GSD ou mis à jour. Le remplacement n'est effectué que pour les profils de catalogue livrés par STEP 7 mais pas pour les profils de catalogue que vous avez vous-même créés.

Les esclaves DP dont les propriétés ne sont plus définies dans les fichiers de type mais dans les fichiers GSD se trouvent toujours au même endroit dans le catalogue du matériel.

Les "anciens" fichiers de type n'ont pas été effacés mais se trouvent à un autre endroit du catalogue. Ils se trouvent sous "**Autres appareils de terrain\Esclaves Profibus-DP compatibles\...**".

### **Extension d'une configuration DP existante avec STEP 7 à partir de V5.1, Servicepack 4**

Si vous souhaitez étendre un esclave DP modulaire dans un projet créé avec une version de STEP 7 antérieure à V5.1 SP4, vous ne pouvez pas utiliser les blocs ou les modules à l'endroit habituel dans le catalogue du matériel. Utilisez l'esclave DP sous "**Autres APPAREILS DE TERRAIN\Esclaves Profibus-DP compatibles\...**".

### **Edition d'une configuration DP avec une version de STEP 7 antérieure à V5.1 SP4**

Si vous configurez un esclave DP "mis à jour" avec STEP 7 à partir de V5.1 Servicepack 4 et éditez ensuite le projet avec une version antérieure (version de STEP 7 antérieure à V5.1, SP4), l'esclave DP n'est pas éditable car le fichier GSD est inconnu à la version de STEP 7.

Solution : Vous pouvez installer le fichier GSD requis dans l'ancienne version de STEP 7. Enregistrez pour cela le fichier GSD dans le projet. STEP 7 utilisera le nouveau fichier GSD lors de l'édition ultérieure du projet avec la version actuelle de STEP 7.

## 7.3 Editer des configurations actuelles avec des versions antérieures de STEP 7

### Configuration de l'échange de données direct

Configuration de l'échange de données direct avec un maître DP **sans** réseau maître DP :

- impossible avec STEP 7 V5, Servicepack 2 (ou plus ancien),
- possible avec STEP 7 V5.0 à partir de Servicepack 3 et avec STEP 7 V5.1.

Si vous enregistrez un maître DP sans réseau maître DP avec des affectations configurées pour l'échange de données direct, puis continuez à éditer ce projet avec une version antérieure de STEP 7 V5 (STEP 7 V5.0, SP2 (ou antérieure)), les effets suivants peuvent se produire :

- Un réseau maître DP s'affiche avec des esclaves DP utilisés par STEP 7 pour stocker de manière interne les affectations pour l'échange direct. Ces esclaves DP ne font pas partie du réseau maître DP indiqué.
- Vous ne pouvez connecter à ce maître DP ni un nouveau réseau maître ni un réseau maître orphelin.

### Liaison en ligne à la CPU via l'interface PROFIBUS DP

Configuration de l'interface PROFIBUS DP **sans** réseau maître DP :

- STEP 7 V5.0, Servicepack 2 (ou antérieur) : une liaison à la CPU via cette interface n'est pas possible.
- A partir de STEP 7 V5.0 à partir de Servicepack 3 : la compilation génère des données système pour l'interface PROFIBUS DP ; une liaison à la CPU via cette interface est possible après le chargement.

## 7.4 Utiliser les configurations PC SIMATIC à partir des versions précédentes

### Configurations PC des projets de STEP 7 V5.1 (jusqu'à SP 1)

A partir de la version STEP 7 V5.1, Servicepack 2, vous pouvez charger des liaisons dans la station PC, tel que c'est le cas pour une station S7-300 ou S7-400 (sans passer par un fichier de configuration). Cependant, un fichier de configuration est toujours créé lors de l'enregistrement et de la compilation afin de pouvoir transmettre, également par ce moyen, la configuration sur la station PC cible.

Par conséquent, les fichiers de configuration qui viennent d'être créés contiennent des informations qui ne pouvaient pas être interprétées par les «anciennes» stations PC. STEP 7 s'adapte automatiquement à cet état:

- Lorsque vous reconfigurez une station SIMATIC PC avec STEP 7 à partir de V5.1, Servicepack 2, STEP 7 part du principe que la station PC cible a été configurée au moyen du CD SIMATIC NET à partir du 7/2001), à savoir que le S7RTM (gestionnaire Runtime) est installé. Les données de configuration sont ainsi créées de sorte à pouvoir être interprétées par une "nouvelle" station PC.
- Lorsque vous utilisez une station configurée SIMATIC PC à partir d'une version précédente (par exemple, une station PC qui a été configurée avec STEP 7 V5.1, Servicepack 1), STEP 7 **ne** part **pas** du principe que la station PC cible a été configurée au moyen du CD SIMATIC NET à partir du 7/2001. Les données de configuration sont créées de sorte à pouvoir être interprétées par une "ancienne" station PC.

Si ce comportement prédéfini n'est pas adapté, vous pouvez le modifier, comme décrit ci-après :

### Paramétrage dans le contexte "Configuration du matériel"

1. Ouvrez la configuration matérielle de la station PC.
2. Cliquez avec le bouton droit de la souris dans la fenêtre de la station (dans la surface blanche).
3. Sélectionnez le menu contextuel "Propriétés de la station".
4. Activez ou désactivez la case à cocher sous "Compatibilité".

### Paramétrage dans le contexte "Configuration de réseaux"

1. Ouvrez la configuration du réseau.
2. Sélectionnez la station PC.
3. Sélectionnez la commande **Edition > Propriétés de l'objet**.
4. Sélectionnez l'onglet "Configuration" dans la boîte de dialogue.
5. Activez ou désactivez la case à cocher sous "Compatibilité".

## **Configurations PC à partir des projets STEP 7 V5.0**

Si vous souhaitez continuer à exploiter, avec STEP 7 à partir de V5.0, Servicepack 3, une station SIMATIC PC configurée afin de configurer de nouveaux composants qui sont acceptés uniquement à partir de la version Servicepack 3, vous devez alors convertir la station :

1. Sélectionnez la station SIMATIC PC dans SIMATIC Manager et sélectionnez la commande **Edition > Propriétés de l'objet**.
2. Dans l'onglet "Fonctionnalité" de la boîte de dialogue des propriétés, cliquez sur le bouton "Extension".

La station SIMATIC PC est ensuite convertie et doit être exploitée uniquement avec STEP 7 V5.0, Servicepack 3 ou des versions plus récentes.

## 7.5 Représentation de modules ayant été configurés avec des versions plus récentes de STEP 7 ou avec un progiciel optionnel




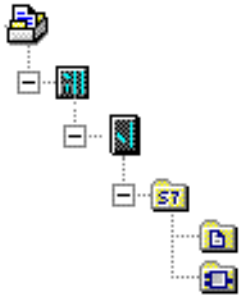

A partir du Servicepack 3 de STEP 7 V5.1, tous les modules sont représentés, même ceux qui ont été configurés dans une version plus récente de STEP 7 et qui, de ce fait, sont inconnus pour une version plus "ancienne" de STEP 7. Il est également possible de représenter les modules qui ont été configurés avec un progiciel optionnel - même lorsque ce dernier n'est pas installé sur la PG sur laquelle vous ouvrez le projet.

Dans les versions précédentes de STEP 7, ces modules et les objets de hiérarchie inférieure qu'ils contiennent n'étaient pas représentés. Dans la version actuelle, ils sont visibles et en partie éditables. A l'aide de cette fonction, vous pouvez p. ex. également modifier des programmes utilisateur lorsque le projet a été créé sur un autre ordinateur avec une version plus récente de STEP 7 et que le module (p. ex. une CPU) ne peut pas être configuré avec la version existante de STEP 7 du fait qu'il possède p. ex. de nouvelles propriétés et de nouveaux paramètres.

Le module "inconnu" pour STEP 7 apparaît sous forme de objet générique avec la représentation suivante :



Lorsque vous ouvrez le projet avec une version appropriée de STEP 7 ou avec le progiciel optionnel adéquat, tous les modules s'affichent dans la représentation habituelle et sont à nouveau utilisables sans restriction.

PG avec version de STEP 7 plus récente / avec progiciel optionnel		PG avec version de STEP 7 plus ancienne / sans progiciel optionnel
		
	>>>---Données projet--->>>	
Représentation avec module plus récent "connu" 		Représentation du module plus récent sous forme de module "inconnu" 



## Utilisation de l'objet générique dans SIMATIC Manager

Lorsque vous ouvrez le projet, un message vous informe qu'un ou plusieurs objets d'un type donné ne peuvent pas être représentés. L'objet générique est cependant visible sous la station. Tous les objets de niveau hiérarchique inférieur, comme p. ex. le programme utilisateur, les données système et la table des liaisons sont visibles et peuvent être chargés depuis SIMATIC Manager.

Vous avez la possibilité d'ouvrir, d'éditer, de compiler et de charger le programme utilisateur (p. ex. des blocs).

Les projets contenant des objets génériques présentent cependant les restrictions suivantes :

- La copie d'une station contenant un objet générique n'est pas possible.
- La commande "Enregistrer le projet sous..." ne peut pas être exécutée avec l'option "avec réorganisation".

## Utilisation de l'objet générique dans la configuration matérielle

L'objet générique est représenté à l'emplacement sur lequel il a été configuré.

Vous avez la possibilité d'ouvrir le module, mais ne pouvez ni le reparamétrer, ni le charger. Les propriétés du module sont limitées à l'onglet "Objet générique". La configuration de la station ne peut pas être modifiée (vous ne pouvez p. ex. pas ajouter de nouveaux modules).

Le diagnostic du matériel (p. ex. l'ouverture de la station en ligne) est également possible (avec des restrictions ; les nouvelles possibilités de diagnostic et les textes de diagnostic ne sont pas pris en compte).

## Utilisation de l'objet générique dans la configuration de réseau

L'objet générique est également représenté dans NetPro. Le nom du module dans la station correspondante est précédé d'un point d'interrogation.

Vous pouvez ouvrir un projet contenant un objet générique seulement protégé contre l'écriture dans NetPro.

Lorsque vous ouvrez le projet avec protection contre l'écriture vous pouvez afficher et imprimer la configuration de réseau. L'état des liaisons est également possible ; du moins, les informations prises en charge dans la version précédentes de STEP 7 sont affichées.

Vous n'avez cependant pas la possibilité d'effectuer de modifications, de les enregistrer, de les compiler ou de les charger.

## Réinstallation de modules

Si le module provient d'une version plus récente de STEP 7 et qu'une mise à jour matérielle est disponible pour ce module, vous pouvez remplacer l'objet générique par le module "réel". Dès l'ouverture de la station, vous obtenez des informations sur les mises à jour matérielles ou logiciels optionnels nécessaires et avez la possibilité d'installer les mises à jour matérielles depuis cette boîte de dialogue. Une alternative consiste à réinstaller les modules en choisissant la commande de menu **Outils > Installer mises à jour matérielles**.



## 8 Définition de mnémoniques

### 8.1 Adressage absolu et adressage symbolique

Dans un programme STEP 7, vous utilisez des opérandes comme des signaux d'E/S, des mémentos, des compteurs, des temporisations, des blocs de données et des blocs fonctionnels. Vous pouvez accéder à ces opérandes par adressage absolu dans votre programme. Toutefois, la lisibilité de vos programmes sera grandement améliorée si vous faites plutôt appel à des mnémoniques (par exemple, Moteur\_A\_Marche ou désignations usuelles dans le système d'identification de votre secteur d'activité). Il est alors possible d'accéder aux opérandes de votre programme utilisateur via ces mnémoniques.

#### Adresse absolue

Une adresse absolue est composée d'un identificateur d'opérande et d'une adresse (par exemple A 4.0, E 1.1, M 2.0, FB21).

#### Adressage symbolique

Vous pouvez structurer votre programme de manière plus lisible et faciliter ainsi la correction d'erreurs en affectant des noms symboliques (mnémoniques) aux adresses absolues.

STEP 7 est en mesure de convertir automatiquement les mnémoniques dans les adresses absolues requises. Si vous préférez adresser des ARRAY, STRUCT, blocs de données, données locales, blocs de code et types de données utilisateur de manière symbolique, vous devez cependant d'abord affecter un mnémonique aux adresses absolues, avant de pouvoir réaliser l'adressage symbolique.

Vous pouvez par exemple affecter le mnémonique Moteur\_Marche à l'opérande A 4.0, puis utiliser Moteur\_Marche comme adresse dans une instruction de programme. L'adressage symbolique vous permet de déterminer plus aisément dans quelle mesure des éléments du programme correspondent aux composants de votre projet de commande du processus.

---

#### Nota

Dans un mnémonique (désignation d'une variable), l'utilisation successive de deux caractères de soulignement n'est pas autorisée (comme par exemple : Moteur\_Marche).

---

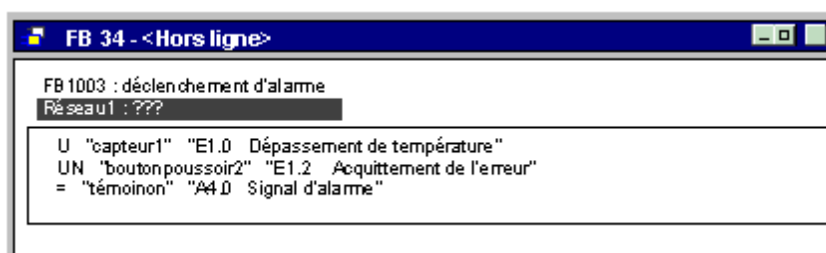
### Assistance lors de la saisie d'un programme

Dans les langages de programmation CONT, LOG et LIST, vous pouvez saisir les adresses, paramètres et noms de blocs de manière absolue ou symbolique.

La commande **Affichage > Représentation symbolique** permet d'aller et venir entre l'affichage de l'adressage absolu et celui de l'adressage symbolique.

Pour faciliter la programmation utilisant l'adressage symbolique, vous pouvez afficher les adresses absolues et commentaires correspondant aux mnémoniques utilisés. Choisissez à cet effet la commande **Affichage > Informations mnémonique**. Après chaque instruction LIST, la ligne de commentaire est remplacée en conséquence. Vous ne pouvez pas éditer l'affichage ; vous devez effectuer les modifications dans la table des mnémoniques ou dans la table de déclaration des variables.

La figure suivante montre une informations sur mnémoniques dans le langage LIST.



A l'impression d'un bloc, la représentation en cours de l'écran est imprimée avec le commentaire d'instruction ou le commentaire de mnémonique.

## 8.2 Mnémoniques globaux et mnémoniques locaux

Un mnémonique (nom symbolique) vous permet d'utiliser des désignations parlantes à la place d'adresses absolues. En combinant l'usage de mnémoniques courts et de commentaires explicites, vous répondez à la fois aux besoins d'une programmation concise et d'une programmation bien documentée.

L'on distingue les mnémoniques locaux des mnémoniques globaux.

	<b>Mnémoniques globaux</b>	<b>Mnémoniques locaux</b>
Domaine de validité	<ul style="list-style-type: none"> <li>ils sont valables dans l'ensemble du programme utilisateur,</li> <li>ils peuvent être utilisés par tous les blocs,</li> <li>leur signification est la même dans tous les blocs,</li> <li>leur nom doit être univoque dans l'ensemble du programme utilisateur.</li> </ul>	<ul style="list-style-type: none"> <li>ils sont connus uniquement dans le bloc dans lequel ils ont été définis,</li> <li>vous pouvez utiliser le même nom dans différents blocs à des fins différentes.</li> </ul>
Caractères autorisés	<ul style="list-style-type: none"> <li>lettres, chiffres, caractères spéciaux,</li> <li>trémas à l'exclusion de 0x00, 0xFF et des guillemets,</li> <li>lorsque vous utilisez des caractères spéciaux dans un mnémonique, ce dernier doit être placé entre guillemets.</li> </ul>	<ul style="list-style-type: none"> <li>lettres,</li> <li>chiffres,</li> <li>caractère de soulignement (<u>  </u>),</li> </ul>
Utilisation	<p>Vous pouvez définir des mnémoniques globaux pour :</p> <ul style="list-style-type: none"> <li>entrées/sorties (E, EB, EW, ED, A, AB, AW, AD)</li> <li>entrées, sorties de périphérie (PE, PA)</li> <li>mémentos (M, MB, MW, MD)</li> <li>temporisations (T)/ compteurs (Z)</li> <li>blocs de code (OB, FB, FC, SFB, SFC)</li> <li>blocs de données (DB)</li> <li>types de données utilisateur</li> <li>table des variables (VAT)</li> </ul>	<p>Vous pouvez définir des mnémoniques locaux pour :</p> <ul style="list-style-type: none"> <li>paramètres de blocs (paramètres d'entrée, de sortie, d'entrée/sortie),</li> <li>données statiques d'un blocs</li> <li>données temporaires d'un bloc</li> </ul>
Endroit de définition	table des mnémoniques	table de déclaration des variables du bloc

### 8.3 Représentation des mnémoniques globaux et des mnémoniques locaux

Dans la section des instructions d'un programme, vous pouvez distinguer les mnémoniques globaux des mnémoniques locaux de la manière suivante :

- Les mnémoniques de la table des mnémoniques (globaux) sont représentés entre guillemets (" ").
- Les mnémoniques de la table de déclaration des variables du bloc (locaux) sont précédés du signe "#".

Vous n'avez pas besoin de saisir vous-même les guillemets ou le signe #. Le mnémonique sera automatiquement complété après vérification de la syntaxe lors de la saisie du programme en CONT, LOG ou LIST.

Toutefois, lorsque la confusion est possible, par exemple parce que des mnémoniques identiques ont été utilisés dans la table des mnémoniques et dans la table de déclaration des variables, vous devez identifier de manière explicite le mnémonique global que vous souhaitez utiliser. En effet, le logiciel interprète les mnémoniques non identifiés comme étant des variables locales.

De plus, l'identification des mnémoniques globaux s'avère nécessaire lorsque ceux-ci contiennent des caractères d'espacement.

Ces règles et l'identification des mnémoniques valent également pour la programmation dans une source LIST. Dans le cas de la saisie orientée source, les identifications ne sont pas complétées automatiquement, cependant elles ne sont requises que s'il y a un risque de confusion.

---

#### Nota

La commande **Affichage > Représentation symbolique** permet d'aller et entre l'affichage des mnémoniques globaux déclarés et celui des adresses absolues correspondantes.

---

## 8.4 Définition de la priorité de l'opérande (symbolique/absolu)

Grâce à la priorité de l'opérande, vous pouvez rétablir le code du programme à souhaits en cas de modifications dans la table des mnémoniques, de modification des noms de paramètres de blocs de données ou de blocs fonctionnels ou de modification de noms de composants d'UDT ou de multiinstances qui y font référence.

Définissez la priorité de l'opérande pour chacune des situations suivantes. Afin que vous puissiez profiter des avantages de la priorité de l'opérande, nous vous recommandons de terminer toute procédure de modification avant d'en démarrer une d'un autre type.

Pour paramétrer la priorité de l'opérande, sélectionnez le dossier Blocs dans SIMATIC Manager, puis choisissez la commande de menu **Editio > Propriétés de l'objet**. Vous pouvez effectuer le paramétrage souhaité dans la page d'onglet "Priorité de l'opérande".

Pour paramétrer la priorité de l'opérande optimale, il convient de différencier les situations suivantes pour les modifications :

- correction de noms individuels
- remplacement de noms ou d'affectations
- mnémoniques, variables, paramètres ou composants nouveaux

### Nota

Sachez que lors d'appels de blocs ("Call FC" ou "Call FB, DB"), seul le numéro de bloc absolu est déterminant pour le bloc de code - même lorsque la priorité de l'opérande symbolique est paramétrée !

### Correction de noms individuels

#### Exemples :

Il s'agit de corriger un nom dans la table des mnémoniques ou dans l'éditeur de programmes/éditeur de blocs en raison d'une erreur de frappe. Ceci s'applique à tous les noms dans la table des mnémoniques, de même qu'à tous les noms de paramètres, variables ou composants qu'il est possible de modifier avec l'éditeur de programmes/éditeur de blocs.

### Définition de la priorité de l'opérande :

	Comportement comme dans STEP7 < V5.2	Recommandé pour programmation symbolique
Valeur absolue prioritaire	<input checked="" type="radio"/> Les mnémoniques de la table des mnémoniques sont utilisées à chaque accès [E,A,M,T,Z et DB] et appliqués dans le DB en cours	<input type="radio"/> Exception : Des accès à des mnémoniques dans des DB sont toujours programmés comme dans un bloc de code
Mnémonique est prioritaire	<input type="radio"/> Exception : Les mnémoniques en cours sont appliqués en cas d'accès à des types de données dont la structure n'a pas été modifiées	<input type="radio"/> Pour tous les accès : [E,A,M,T,Z et DB]

### Réalisation des modifications :

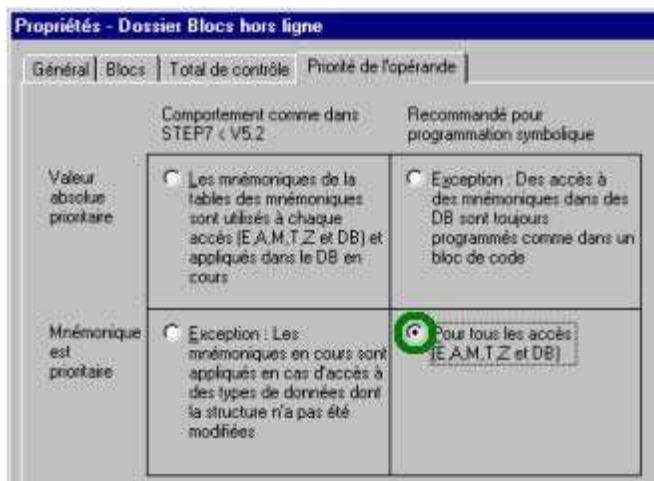
Sélectionnez le dossier Blocs dans SIMATIC Manager, puis choisissez la commande de menu **Edition > Vérifier la cohérence des blocs**. La fonction "Vérifier la cohérence des blocs" réalise les modifications nécessaires dans les blocs individuels.

### Remplacement de noms ou d'affectations

#### Exemples :

- Les noms d'affectations existantes sont remplacés dans la table des mnémoniques.
- De nouvelles adresses sont affectées à des affectations existantes dans la table des mnémoniques.
- Des noms de variables, paramètres ou composants sont remplacés dans l'éditeur de programmes/éditeur de blocs.

### Paramétrage de la priorité de l'opérande :



### Réalisation des modifications:

Sélectionnez le dossier Blocs dans SIMATIC Manager, puis choisissez la commande de menu **Edition > Vérifier la cohérence des blocs**. La fonction "Vérifier la cohérence des blocs" réalise les modifications nécessaires dans les blocs individuels.

### Mnémoniques, variables, paramètres ou composants nouveaux

#### Exemple :

- Vous définissez de nouveaux mnémoniques pour des adresses utilisées dans le programme.
- Vous ajoutez de nouvelles variables ou de nouveaux paramètres aux blocs de données, UDT ou blocs fonctionnels.



### Définition de la priorité de l'opérande :

- Dans le cas de modifications dans la table des mnémoniques.

	Comportement comme dans STEP7 < V5.2	Recommandé pour programmation symbolique
Valeur absolue prioritaire	<input checked="" type="radio"/> Les mnémoniques de la table des mnémoniques sont utilisés à chaque accès (E,A,M,T,Z et DB) et appliqués dans le DB en cours	<input checked="" type="radio"/> Exception : Des accès à des mnémoniques dans des DB sont toujours programmés comme dans un bloc de code
Mnémonique est prioritaire	<input type="radio"/> Exception : Les mnémoniques en cours sont appliqués en cas d'accès à des types de données dont la structure n'a pas été modifiées	<input type="radio"/> Pour tous les accès (E,A,M,T,Z et DB)

- Dans le cas de modifications dans l'éditeur de programmes/éditeur de blocs.

	Comportement comme dans STEP7 < V5.2	Recommandé pour programmation symbolique
Valeur absolue prioritaire	<input checked="" type="radio"/> Les mnémoniques de la table des mnémoniques sont utilisés à chaque accès (E,A,M,T,Z et DB) et appliqués dans le DB en cours	<input checked="" type="radio"/> Exception : Des accès à des mnémoniques dans des DB sont toujours programmés comme dans un bloc de code
Mnémonique est prioritaire	<input type="radio"/> Exception : Les mnémoniques en cours sont appliqués en cas d'accès à des types de données dont la structure n'a pas été modifiées	<input type="radio"/> Pour tous les accès (E,A,M,T,Z et DB)

### Réalisation des modifications:

Sélectionnez le dossier Blocs dans SIMATIC Manager, puis choisissez la commande de menu **Edition > Vérifier la cohérence des blocs**. La fonction "Vérifier la cohérence des blocs" réalise les modifications nécessaires dans les blocs individuels.

## 8.5 Table des mnémoniques pour mnémoniques globaux

Dans la table des mnémoniques, vous définissez les mnémoniques globaux.

Une table des mnémoniques (objet "Mnémoniques") vide est automatiquement générée lorsque vous créez un programme S7 ou M7.

### Domaine de validité

La table des mnémoniques vaut pour le module auquel le programme est associé. Si vous voulez vous servir des mêmes mnémoniques dans différentes CPU, vous devez vous-même faire en sorte que les entrées correspondent dans les différentes tables de mnémoniques (par exemple, par copie).

### 8.5.1 Structure et éléments de la table des mnémoniques

#### Structure de la table des mnémoniques

The screenshot shows a software window titled 'Editeur de mnémoniques - Programme S7(1)(Mnémoniques)--ZFr01\_01...'. The window contains a menu bar (Table, Edition, Insertion, Affichage, Outils, Fenêtre, ?) and a toolbar with various icons. Below the toolbar is a dropdown menu showing 'Tous les mnémoniques'. The main area displays a table for 'Programme S7(1)(Mnémoniques)--ZFr01\_01\_STEP7\_\_LIST\_1-9\SIMATIC 300(1)\'. The table has columns for 'Etat', 'D', 'O', 'S', 'C', 'N', 'Mnémonique', 'Opérande', 'Type de d', and 'Commentaire'. The rows are numbered 1 to 6. Row 1 is 'Automatique Marche' with 'E 0.5' and 'BOOL', and a comment 'Activation de la bascule'. Rows 2-4 are 'Commutateur 1', 'Commutateur 2', and 'Commutateur 3' respectively, all with 'E' and 'BOOL' types. Row 5 is 'Commutateur 4' with 'E 0.4' and 'BOOL'. Row 6 is 'Diesel' with 'DB 2', 'FB 1', and comment 'Données du moteur dies'. A status bar at the bottom says 'Pour obtenir de l'aide, appuyez sur F1.' and has a 'NUM' button.

	Etat	D	O	S	C	N	Mnémonique ▲	Opérande	Type de d	Commentaire
1		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Automatique Marche	E 0.5	BOOL	Activation de la bascule
2		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Commutateur 1	E 0.1	BOOL	Pour la connexion en sé
3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Commutateur 2	E 0.2	BOOL	Pour la connexion en sé
4		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Commutateur 3	E 0.3	BOOL	Pour la connexion en pé
5		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Commutateur 4	E 0.4	BOOL	Pour la connexion en pé
6		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Diesel	DB 2	FB 1	Données du moteur dies

#### Colonne "Ligne"

	Si les colonnes des "Propriétés spécifiques de l'objet" sont masquées (commande de menu <b>Affichage &gt; Colonnes D, O, S, C, N</b> désélectionnée), cette icône s'affiche dans la colonne "Ligne" lorsqu'une "Propriété spécifique de l'objet" au moins a été définie dans la ligne correspondante.
--	---

#### Colonne "Etat"

	Le mnémonique ou l'opérande sont identiques à une autre entrée dans la table des mnémoniques.
	Le mnémonique est encore incomplet (il manque son nom ou son adresse).

## Colonnes "D/O/S/C/N"

Vous pouvez voir dans ces colonnes si des propriétés spécifiques (attributs) ont été attribuées au mnémonique :

- D signifie que des définitions d'erreur servant au diagnostic du processus ont été créées pour le mnémonique avec le logiciel optionnel S7 PDIAG (V5).
- O est la seconde lettre de contrôle-commande et signifie que le mnémonique peut faire l'objet de cette fonction dans WinCC.
- S signifie qu'un message sur mnémonique (SCAN) a été affecté au mnémonique.
- C signifie que le mnémonique a des propriétés servant à la communication.
- N signifie qu'un contrôle-commande rapide et direct du mnémonique est possible depuis l'éditeur de programmes (commande-commande par contact).

Cliquez sur les cases d'option pour activer ou désactiver les "Propriétés spécifiques de l'objet". Vous pouvez également modifier les "Propriétés spécifiques de l'objet" en choisissant la commande de menu **Edition > Propriétés spécifiques de l'objet**.

## Colonne "Mnémonique"

Le nom du mnémonique ne doit pas dépasser 24 caractères.

Vous ne pouvez pas affecter de mnémoniques aux opérandes de blocs de données (DBD, DBW, DBB, DBX) dans la table des mnémoniques. Les noms de ces opérandes sont définis par la déclaration dans les blocs de données.

Il existe, pour les blocs d'organisation (OB) et quelques blocs fonctionnels système (SFB) et fonctions système (SFC), des mnémoniques prédéfinis que vous pouvez importer dans la table des mnémoniques de votre programme S7. Le fichier d'importation se trouve dans le répertoire STEP 7, sous ...\\S7data\\Symbol\\Symbol.sdf.

## Colonne "Opérande"

Il s'agit de l'adresse d'un opérande précis.

Exemple : entrée E 12.1

La syntaxe de l'opérande est vérifiée lors de la saisie.

## Colonne "Type de données"

Vous pouvez choisir parmi les différents types de données que STEP 7 met à votre disposition. Un type de données pris par défaut est inscrit dans ce champ, mais vous pouvez le modifier. Si votre modification n'est pas compatible avec l'opérande ou que la syntaxe est erronée, un message d'erreur s'affiche lorsque vous quittez le champ.

## Colonne "Commentaire"

Vous pouvez affecter des commentaires à tous les mnémoniques. La combinaison de mnémoniques courts et de commentaires détaillés permet d'assurer une bonne documentation du programme ainsi qu'une programmation efficace. Un commentaire ne doit pas dépasser 80 caractères.

## Conversion en variables C

Vous pouvez sélectionner des mnémoniques dans la table des mnémoniques d'un programme M7 et les convertir en variables C en liaison avec le logiciel optionnel ProC/C++.


## 8.5.2 Opérandes et types de données autorisés dans la table des mnémoniques

La notation employée doit être la même pour toute la table des mnémoniques. Pour effectuer un changement de la notation allemande (ancienne SIMATIC) à la notation anglaise (ancienne CEI), ou inversement, il faut avoir recours à la commande **Outils > Paramètres**, onglet "Langue" dans SIMATIC Manager.


Anglais	Allemand	Désignation	Type de données	Plage d'adresses
I	E	Bit d'entrée	BOOL	0.0..65535.7
IB	EB	Octet d'entrée	BYTE, CHAR	0..65535
IW	EW	Mot d'entrée	WORD, INT, S5TIME, DATE	0..65534
ID	ED	Double mot d'entrée	DWORD, DINT, REAL, TOD, TIME	0..65532
Q	A	Bit de sortie	BOOL	0.0..65535.7
QB	AB	Octet de sortie	BYTE, CHAR	0..65535
QW	AW	Mot de sortie	WORD, INT, S5TIME, DATE	0..65534
QD	AD	Double mot de sortie	DWORD, DINT, REAL, TOD, TIME	0..65532
M	M	Bit de memento	BOOL	0.0..65535.7
MB	MB	Octet de memento	BYTE, CHAR	0..65535
MW	MW	Mot de memento	WORD, INT, S5TIME, DATE	0..65534
MD	MD	Double mot de memento	DWORD, DINT, REAL, TOD, TIME	0..65532
PIB	PEB	Octet de périphérie d'entrée	BYTE, CHAR	0..65535
PQB	PAB	Octet de périphérie de sortie	BYTE, CHAR	0..65535
PIW	PEW	Mot de périphérie d'entrée	WORD, INT, S5TIME, DATE	0..65534
PQW	PAW	Mot de périphérie de sortie	WORD, INT, S5TIME, DATE	0..65534
PID	PED	Double mot de périphérie d'entrée	DWORD, DINT, REAL, TOD, TIME	0..65532
PQD	PAD	Double mot de périphérie de sortie	DWORD, DINT, REAL, TOD, TIME	0..65532
T	T	Temporisation	TIMER	0..65535
C	Z	Compteur	COUNTER	0..65535
FB	FB	Bloc fonctionnel	FB	0..65535
OB	OB	Bloc d'organisation	OB	1..65535
DB	DB	Bloc de données	DB, FB, SFB, UDT	1..65535
FC	FC	Fonction	FC	0..65535
SFB	SFB	Bloc fonctionnel système	SFB	0..65535
SFC	SFC	Fonction système	SFC	0..65535
VAT	VAT	Table des variables		0..65535
UDT	UDT	Type de données utilisateur	UDT	0..65535

### 8.5.3 Mnémoniques incomplets ou non univoques dans la table des mnémoniques

#### Mnémoniques incomplets

Vous pouvez également sauvegarder des mnémoniques incomplets ce qui vous permet, par exemple, de ne définir dans un premier temps que leur nom et de compléter l'indication de l'adresse (opérande) plus tard. Vous pouvez, en particulier, interrompre votre travail dans la table des mnémoniques à tout moment et enregistrer l'état intermédiaire de cette dernière. Les mnémoniques incomplets sont caractérisés par l'icône  dans la colonne "Etat". Pour pouvoir utiliser le mnémonique lors de la création du logiciel sans recevoir de message d'erreur, il faut toutefois que le mnémonique, l'opérande et le type de données soient indiqués.

#### Formation de mnémoniques non univoques

Vous pouvez avoir des mnémoniques non univoques lorsque vous ajoutez un mnémonique à la table des mnémoniques et que le nom ou l'adresse spécifiée figure déjà dans la table pour un autre mnémonique. Le nouveau et l'ancien mnémonique ne sont donc plus univoques. Cet état est caractérisé dans la colonne "Etat" par l'icône .

C'est ce qui se produit, par exemple, quand vous copiez et insérez un mnémonique pour modifier ensuite légèrement l'entrée dans la copie.

#### Repérage des mnémoniques non univoques

Les mnémoniques non univoques sont repérés, dans la table, par une mise en valeur graphique (couleur, police) afin d'attirer votre attention sur la nécessité d'une correction. Vous pouvez afficher tous les mnémoniques ou, par un filtre, seulement les mnémoniques univoques ou seulement les mnémoniques non univoques.

#### Correction de la non-univocité

Un mnémonique non univoque le devient lorsque vous modifiez le composant - nom et (ou) opérande - qui a engendré la non-univocité. Le mnémonique ayant auparavant la même adresse reprend automatiquement son unicité.

## 8.6 Possibilités de saisie de mnémoniques globaux

Il existe trois manières de saisir les mnémoniques qui seront réutilisés ultérieurement lors de la programmation :

- Saisie via une boîte de dialogue  
Vous pouvez, dans la fenêtre de saisie du programme, ouvrir une boîte de dialogue et y définir un nouveau mnémonique. Ce procédé convient à la définition de mnémoniques individuels quand vous constatez, par exemple, au cours de la programmation qu'un mnémonique manque ou doit être corrigé. Vous évitez ainsi l'affichage de la table des mnémoniques.
- Saisie directe dans la table des mnémoniques  
Vous pouvez inscrire les mnémoniques et leur adresse associée directement dans une "table des mnémoniques". Ce procédé est recommandé pour la saisie de plusieurs mnémoniques et pour la création initiale de la table des mnémoniques, car les mnémoniques déjà définis sont affichés à l'écran et vous conservez ainsi une meilleure vue d'ensemble.
- Importation de tables des mnémoniques depuis d'autres tableurs  
Vous pouvez créer les données pour la table des mnémoniques à l'aide de votre tableur préféré, par exemple Microsoft Excel, et ensuite importer le fichier créé dans la table des mnémoniques.

### 8.6.1 Remarques générales sur la saisie de mnémoniques

Pour entrer de nouveaux mnémoniques dans la table des mnémoniques, vous vous positionnez dans la première ligne vide de la table et en complétez les champs. Vous pouvez insérer de nouvelles lignes avant la ligne en cours via la commande **Insertion > Mnémonique**. Quand la ligne précédant la position du curseur contient déjà un opérande, l'insertion d'un nouveau mnémonique vous est facilitée par des valeurs par défaut s'inscrivant automatiquement dans les colonnes "Opérande" et "Type de données" : un opérande dérivé de celui de la ligne précédente et le type de données par défaut.

Les commandes du menu "Edition" permettent de copier, puis de modifier des entrées existantes. Ensuite, vous sauvegardez et fermez la table des mnémoniques. Vous pouvez également sauvegarder des mnémoniques qui ne sont pas encore entièrement définis.

En définissant les mnémoniques, vous devez tenir compte des particularités suivantes :

Colonne	Nota
Mnémonique	Ce nom doit être univoque dans l'ensemble de la table des mnémoniques. Quand vous confirmez votre saisie ou quittez ce champ, un repère est placé devant un mnémonique non univoque. Un mnémonique ne doit pas dépasser 24 caractères. Les guillemets ne sont pas autorisés.
Opérande	Quand vous validez ou quittez ce champ, le programme vérifie si l'opérande indiqué est autorisé.
Type de données	Quand vous avez entré un opérande, une valeur par défaut s'inscrit dans ce champ. Si vous la modifiez, le programme vérifie si le nouveau type de données convient à l'opérande.
Commentaire	Ce champ vous permet de saisir des remarques (80 caractères au maximum) décrivant la fonction du mnémonique. La saisie d'un commentaire est facultative.

## 8.6.2 Saisie de mnémoniques globaux individuels dans des boîtes de dialogue

La procédure suivante vous montre comment modifier ou créer, lors de la programmation de blocs, des mnémoniques via des boîtes de dialogue sans devoir afficher la table des mnémoniques.

Cette méthode est utile lorsque vous ne désirez éditer qu'un seul mnémonique. Nous vous conseillons, pour la modification de plusieurs mnémoniques, d'ouvrir la table des mnémoniques et d'y travailler directement.

### Activation de l'affichage des mnémoniques dans le bloc

Lorsqu'un bloc est ouvert, vous pouvez activer l'affichage des mnémoniques dans la fenêtre de bloc avec la commande **Affichage > Représentation symbolique**. Cette commande est cochée lorsque la représentation symbolique est active.

### Définition de mnémoniques lors de la saisie du programme

1. Assurez-vous que la représentation symbolique est activée dans la fenêtre de bloc (commande **Affichage > Représentation symbolique**).
2. Sélectionnez, dans la section des instructions de votre programme, l'adresse absolue à laquelle vous voulez affecter un mnémonique.
3. Choisissez la commande **Edition > Mnémonique**.
4. Complétez la boîte de dialogue affichée en y inscrivant en particulier un mnémonique, puis fermez-la.

Le mnémonique défini s'inscrit dans la table des mnémoniques. Les données qui engendreraient des mnémoniques non univoques sont refusées et un message d'erreur est émis.

### Edition dans la table des mnémoniques

La commande **Outils > Table des mnémoniques** vous permet d'ouvrir la table des mnémoniques en vue de son édition.

### 8.6.3 Saisie de plusieurs mnémoniques globaux dans la table des mnémoniques

#### Ouverture d'une table des mnémoniques

Pour ouvrir la table des mnémoniques, vous pouvez :

- effectuer un double clic sur la table des mnémoniques dans la fenêtre de projet,
- sélectionner la table des mnémoniques dans la fenêtre de projet et choisir la commande **Edition > Ouvrir l'objet**.

La table des mnémoniques pour le programme en cours s'affiche dans sa propre fenêtre. Vous pouvez alors créer ou modifier des mnémoniques. La table est vide lorsque vous l'ouvrez pour la première fois après sa création.

#### Saisie de mnémoniques

Pour entrer de nouveaux mnémoniques dans la table des mnémoniques, vous vous positionnez dans la première ligne vide de la table et en complétez les champs. Vous pouvez insérer de nouvelles lignes vides avant la ligne en cours via la commande **Insertion > Mnémonique**. Les commandes du menu "Edition" permettent de copier, puis de modifier des entrées existantes. Ensuite, vous sauvegardez et fermez la table des mnémoniques. Vous pouvez également sauvegarder des mnémoniques qui ne sont pas encore entièrement définis.

#### Tri des mnémoniques

Il est possible de classer les enregistrements logiques de la table des mnémoniques dans l'ordre alphabétique des mnémoniques, des opérandes, des types de données ou des commentaires.

Vous pouvez par exemple modifier le classement dans la boîte de dialogue que vous appelez en choisissant la commande **Affichage > Tri...**

#### Filtres pour les mnémoniques

Les filtres vous permettent de choisir des sous-ensembles parmi tous les enregistrements de la table.

La commande **Affichage > Filtre** ouvre la boîte de dialogue "Filtre".

Vous pouvez y définir des critères auxquels les enregistrements logiques doivent satisfaire pour être affichés. Vous pouvez sélectionner un filtre pour

- les noms, adresses, types de données, commentaires
- mnémoniques possédant un attribut de contrôle-commande, mnémoniques possédant des propriétés de communication, mnémoniques pour variable binaire dans les messages (memento ou entrée du processus)
- mnémoniques avec l'état "valide", "invalidé (non univoque, incomplet)"

Les différents critères sont combinés par ET. Les enregistrements logiques affichés commencent par les chaînes de caractères indiquées.

Pour en savoir plus sur les possibilités offertes dans la boîte de dialogue "Filtrer", ouvrez l'aide en ligne en appuyant sur la touche de fonction F1.



## 8.6.4 Majuscules/minuscules pour les mnémoniques

### Aucune différenciation n'est faite entre les majuscules et minuscules.

Jusqu'à présent, vous aviez la possibilité de définir des mnémoniques dans STEP 7, qui se distinguaient uniquement par l'emploi de majuscules et de minuscules de certains caractères. Ceci a été modifié à partir de STEP 7 V4.02. La distinction des mnémoniques selon l'emploi de majuscules et de minuscules n'est plus possible à partir de cette version.

Avec cette modification, nous avons répondu aux attentes de nos clients, puisqu'ainsi les sources d'erreurs possibles dans un programme sont considérablement réduites. Cette restriction dans la définition des mnémoniques va également dans le sens des objectifs de PLCopen quant à la définition d'une norme pour les programmes transmissibles.

Une définition distincte de mnémoniques, uniquement due à l'emploi de majuscules ou de minuscules n'est désormais plus possible. Jusqu'à présent, la définition suivante, par exemple, était possible dans la table des mnémoniques :

Moteur1 = E 0.0

moteur1 = E 1.0

Les mnémoniques se distinguaient par la graphie (majuscules/minuscules) du premier caractère. Ce mode de distinction occasionne un important risque de confusion. Avec la définition en vigueur, cette probable source d'erreur est à présent exclue.

#### Effet sur les programmes existant

Si jusqu'à présent vous avez utilisé ce critère de distinction dans la définition de mnémoniques, des conflits sont possibles avec la nouvelle définition lorsque :

- des mnémoniques se distinguent **uniquement** par l'emploi de majuscules/minuscules
- des paramètres se distinguent **uniquement** par l'emploi de majuscules/minuscules
- des mnémoniques se distinguent **uniquement** des paramètres par l'emploi de majuscules/minuscules

Il est cependant possible d'analyser et de corriger ces trois cas comme décrit ci-après.

#### Mnémoniques se distinguant uniquement par l'emploi de majuscules/minuscules

Conflit :

Si la table des mnémoniques n'a pas encore été éditée avec la version logicielle en cours, c'est le premier des mnémoniques non univoques de la table des mnémoniques qui est utilisé lors de la compilation de fichiers source.

Si la table des mnémoniques a déjà été éditée, de tels mnémoniques sont invalides, ce qui signifie qu'à l'ouverture des blocs, aucune symbolique ne s'affiche et qu'une compilation exempte d'erreurs des fichiers source utilisant ces mnémoniques n'est plus possible.

Solution :

Vérifiez les conflits dans la table des mnémoniques en l'ouvrant, puis en l'enregistrant une nouvelle fois. Ceci permet de détecter les mnémoniques non univoques. Vous pouvez à présent afficher ces mnémoniques non univoques au moyen du filtre "Mnémoniques non univoques" et les corriger. Corrigez ensuite les fichiers source présentant des conflits. Aucune autre modification n'est nécessaire pour les blocs, puisqu'à leur ouverture, c'est la table des mnémoniques actuelle (sans conflit) qui est automatiquement utilisée ou affichée.

### **Paramètres se distinguant uniquement par l'emploi de majuscules/minuscules**

Conflit :

Les fichiers source présentant de telles interfaces ne peuvent plus être compilés. Bien que les blocs présentant de telles interface peuvent encore être ouverts, aucun accès au second de ces paramètres n'est possible. Lors de l'enregistrement, l'accès au second de ces paramètres est automatiquement remplacé par l'accès au premier paramètre.

Solution :

Afin de déterminer quels blocs présentent de tels conflits, il est recommandé de générer un fichier source pour tous les blocs d'un programme, à l'aide de la fonction "Générer source". Si des erreurs surviennent lorsque vous tentez de recompiler le fichier source généré, un conflit se présente.

Corrigez vos fichiers source en rendant les paramètres univoques, par exemple à l'aide de la fonction Rechercher/Remplacer, puis recompilez le fichier source.

### **Mnémoniques se distinguant uniquement de paramètres par l'emploi de majuscules/minuscules**

Conflit :

Si les mnémoniques globaux et locaux d'un fichier source ne se distinguent que par l'emploi de majuscules/minuscules et si aucun caractère d'identification des mnémoniques globaux ("Mnémonique") ou locaux (#Mnémonique) n'a été utilisé, c'est le mnémonique local qui est toujours utilisé lors de la compilation. Ceci engendre un code machine modifié.

Solution :

Dans ce cas, il est recommandé de générer une nouvelle source à partir de tous les blocs. Les caractères d'identification correspondants seront ainsi automatiquement attribués aux adresses locales et globales, qui seront traitées correctement lors des compilations ultérieures.

### 8.6.5 Exportation et importation de tables de mnémoniques

Vous pouvez exporter dans un fichier de texte la table des mnémoniques affichée, pour la traiter avec un éditeur de texte de votre choix, par exemple.

Vous pouvez importer, dans votre table des mnémoniques, des tables créées avec une autre application et poursuivre leur traitement dans la table des mnémoniques. Cette fonction vous servira, par exemple, à enregistrer dans la table des mnémoniques et après leur conversion, des listes d'assignation créées sous STEP 5/ST.

Vous disposez des formats de fichier \*.SDF, \*.ASC, \*.DIF et \*.SEQ.

#### Règles pour l'exportation

Vous pouvez exporter la table des mnémoniques entière, un sous-ensemble de cette table défini par filtre ou des lignes sélectionnées dans la représentation de la table.

Les propriétés des mnémoniques que vous pouvez définir à l'aide de la commande **Edition > Propriétés spécifiques de l'objet...** ne sont pas exportées.

#### Règles pour l'importation

- Pour les blocs fonctionnels système (SFB), les fonctions système (SFC) et les blocs d'organisation (OB) les plus fréquemment utilisés, vous trouverez dans le fichier ...\\S7DATA\\SYMBOL\\SYMBOL.SDF des mnémoniques prédéfinis que vous pouvez importer si besoin est.
- Les propriétés des mnémoniques que vous pouvez définir à l'aide de la commande **Edition > Propriétés spécifiques de l'objet...** ne sont pas prises en compte lors de l'exportation et de l'importation.

### 8.6.6 Formats de fichier pour l'importation/exportation d'une table des mnémoniques

Vous pouvez importer les formats de fichier suivants dans la table des mnémoniques ou les en exporter :

- Format de fichier ASCII (ASC)
- Format de fichier DIF (Data Interchange Format)  
Vous pouvez ouvrir, éditer, puis enregistrer les fichiers DIF (Data Interchange Format) avec l'application Microsoft Excel.
- Format de fichier SDF (System Data Format)  
Vous pouvez ouvrir, éditer, puis enregistrer les fichiers SDF (System Data Format) avec l'application Microsoft Access.
  - Utilisez le format SDF pour importer des données dans l'application Microsoft ACCESS ou pour les en exporter.
  - Sélectionnez, dans ACCESS, le format de fichier "Texte (avec séparateurs)".
  - Utilisez le guillemet (") comme séparateur de texte.
  - Utilisez la virgule (,) comme séparateur de champ.
- Liste d'assignation (SEQ)  
**Avertissement** : Lors de l'exportation de la table des mnémoniques dans un fichier de type Typ .SEQ, les commentaires de plus de 40 caractères sont tronqués après le 40ème caractère !

## Format de fichier ASCII (ASC)

Type de fichier	*.ASC				
Structure	Longueur de l'enregistrement, séparateur (virgule), enregistrement				
Exemple	126,phase_verte_piet,	T2	TIMER	Durée de la phase verte pour piétons	
	126,rouge_piet A	0.0	BOOL	Rouge pour piétons	

## Data Interchange Format (DIF)

Type de fichier	*.DIF				
Structure	Un fichier DIF est composé d'un en-tête (header) et de données :				

En-tête	TABLE	Début d'un fichier DIF
	0,1	
	"<Titre>"	Chaîne de caractères du commentaire
	VECTORS	Nombre d'enregistrements dans le fichier
	0,<Nombre d'enregistrements>	
	""	
	TUPLES	Nombre de champs de données dans un enregistrement
	0,<Nombre de colonnes>	
	""	
	DATA	Identification de fin d'en-tête et de début de données
	0,0	
	""	
Données (par enregistrement)	<Type>,<valeur numérique>	Identification du type de données, valeur numérique
	<chaîne de caractères>	Partie alphanumérique, ou
	V	si la partie alphanumérique n'est pas utilisée.

**En-tête :** l'en-tête du fichier doit comporter les types d'enregistrement TABLE, VECTORS, TUPLES et DATA dans l'ordre indiqué ; dans les fichiers DIF, le type d'enregistrement DATA peut être précédé d'autres types d'enregistrements optionnels, mais l'éditeur de mnémoniques n'en tiendra pas compte.

**Données :** dans la section des données, chaque entrée comporte trois parties, à savoir l'identification du type de données, une valeur numérique et une partie alphanumérique.

Vous pouvez ouvrir, éditer et enregistrer les fichiers DIF dans l'application Microsoft Excel. N'utilisez toutefois aucun caractère particulier à la langue, comme par exemple une lettre accentuée.

**System Data Format (SDF)**

Type de fichier	*.SDF
Structure	Chaînes de caractères entre guillemets, sections séparées par des virgules.
Exemple	"phase_verte_piet","T 2","TIMER","Durée de la phase verte pour piétons" "rouge_piet","A 0.0","BOOL","Rouge pour piétons"

Pour ouvrir un fichier SDF dans Microsoft Access, choisissez le format de fichier "texte (avec séparateur)". Indiquez comme séparateur de texte les guillemets (") et comme séparateur de champ la virgule (,).

**Liste d'assignation (SEQ)**

Type de fichier	*.SEQ
Structure	TAB opérande TAB mnémonique TAB commentaire CR
Exemple	T 2 phase_verte_piet Durée de la phase verte pour piétons A 0.0 rouge_piet Rouge pour piétons

TAB représente le caractère de tabulation (09H),  
CR un saut de ligne (retour chariot) avec la touche d'entrée (0DH).

### 8.6.7 Edition de zones dans des tables de mnémoniques

A partir de la version V5.3 de STEP 7, il est possible de sélectionner et d'éditer des zones consécutives d'une table de mnémoniques. Vous pouvez p. ex. copier, couper, coller ou effacer ces zones dans une autre table des mnémoniques.

Vous pouvez ainsi transférer aisément des données d'une table des mnémoniques dans une autre table des mnémoniques, ce qui facilite leur actualisation.

#### Zones pouvant être sélectionnées

- Vous pouvez sélectionner des lignes entières en cliquant dans la première colonne de la ligne. Une sélection de toutes les cellules de la colonne "Etat" à la colonne "Commentaire" correspond également à la sélection de la ligne.
- Vous pouvez sélectionner des cellules individuelles ou plusieurs cellules consécutives comme zone. Pour pouvoir éditer cette zone sélectionnée, toutes les cellules doivent appartenir aux colonnes "Mnémonique", "Adresse", "Type de donnée" et "Commentaire". Si votre sélection est incorrecte, les commandes d'édition ne sont pas actives.
- Les colonnes D, O, S, C, N contiennent les propriétés spécifiques de l'objet des mnémoniques respectifs et sont uniquement copiées si la case à cocher "Copier les propriétés spécifiques de l'objet" est activée dans la boîte de dialogue "Paramètres", que vous appelez en choisissant la commande de menu Outils > Paramètres, et si, comme zone, vous avez sélectionné des lignes entières.
- La copie du contenu des colonnes D, O, S, C, N ne dépend pas du fait que ces colonnes soient affichées ou pas. Vous pouvez commuter l'affichage en choisissant la commande de menu **Affichage > Colonnes D, O, S, C, N**.

#### Procédez de la manière suivante

1. Sélectionnez la zone que vous souhaitez éditer dans la table des mnémoniques.
  - Cliquez à cet effet dans la cellule de début souhaitée avec la **souris**, puis déplacez cette dernière sur la zone à sélectionner, tout en maintenant le bouton gauche enfoncé ou
  - Sélectionnez la zone souhaitée via le **clavier** en appuyant sur les touches Maj + Touche de direction.
2. La zone sélectionnée s'affiche dans la couleur de la sélection. La cellule sélectionnée en premier dans la table des mnémoniques est affichée dans la couleur normale et entourée d'un cadre.
3. Editez ensuite la zone sélectionnée à votre convenance.

## 9 Création de blocs et de bibliothèques

### 9.1 Choix de la méthode de création

Selon le langage de programmation que vous utilisez pour créer votre programme, vous pouvez le saisir de manière incrémentielle et/ou orientée source.

#### Editeurs incrémentiels pour les langages de programmation CONT, LOG, LIST et GRAPH

Avec les éditeurs incrémentiels pour CONT, LOG, LIST et GRAPH, vous créez des blocs qui sont stockés dans le programme utilisateur. Choisissez la saisie incrémentielle lorsque vous souhaitez que vos entrées soient immédiatement vérifiées. Ce mode de saisie convient également aux débutants en programmation. Dans la saisie incrémentielle, une vérification de la syntaxe est immédiatement réalisée pour chaque ligne ou élément. D'éventuelles erreurs sont affichées et doivent être corrigées avant la fin de la saisie. Les entrées correctes du point de vue syntaxique sont automatiquement compilées et rangées dans le programme utilisateur.

Les mnémoniques utilisés doivent avoir été définis avant l'édition de l'instruction. En cas d'absence de certains mnémoniques, la compilation du bloc est incomplète ; vous pouvez cependant enregistrer cet "état provisoire incohérent".

#### Editeurs (de texte) source pour les langages de programmation LIST, SCL ou HiGraph

Dans les éditeurs source, vous créez des **sources** qui seront ensuite compilées en blocs.

Choisissez la saisie orientée source pour entrer ou écrire rapidement un programme.

Dans la saisie orientée source, le programme ou un bloc sont édités dans un fichier de texte qui est ensuite compilé.

Les fichiers de texte (sources) sont stockés dans le dossier Sources de votre programme S7, par exemple comme **source LIST** ou **source SCL**. Un fichier source peut contenir le code pour un ou plusieurs blocs. Les éditeurs de texte pour LIST et SCL vous permettent d'écrire le code pour des **OB, FB, FC, DB et UDT** (types de données utilisateur), c'est-à-dire aussi pour un programme utilisateur complet. L'ensemble du programme d'une CPU (c'est-à-dire tous les blocs) peut être contenu dans un fichier de texte unique.

Les blocs sont générés et stockés dans le programme utilisateur lors de la compilation du fichier source correspondant. Les mnémoniques utilisés doivent avoir été définis avant la compilation. D'éventuelles erreurs ne sont signalées qu'après compilation par le compilateur correspondant.

Il est important, pour la compilation, que vous respectiez la syntaxe du langage de programmation. Cette syntaxe n'est contrôlée que lorsque vous effectuez la vérification de cohérence ou la compilation en blocs.

## 9.2 Choix du langage de programmation

### Choix du langage de programmation et de l'éditeur

Lors de la création d'un bloc ou d'une source, vous déterminez dans les propriétés de l'objet avec quel langage de programmation et quel éditeur vous voulez écrire ce bloc ou cette source. L'éditeur correspondant à ce choix est appelé lorsque vous ouvrez le bloc ou le fichier source.

### Appel de l'éditeur

Vous lancez l'éditeur de langage choisi dans SIMATIC Manager par double clic sur l'objet correspondant (bloc, fichier source, etc.), à l'aide de la commande **Edition > Ouvrir l'objet** ou via le bouton correspondant dans la barre d'outils.

Vous disposez des langages de programmation indiqués dans le tableau pour créer le programme S7. Les langages de programmation CONT, LOG et LIST font partie du logiciel de base de STEP 7. Les autres peuvent être commandés comme logiciels optionnels.

Vous pouvez ainsi faire votre choix parmi différentes philosophies de programmation (schéma à contacts, logigramme, liste d'instructions, langage évolué, commande séquentielle ou graphe d'état) et entre la programmation textuelle ou graphique.

Le choix du langage de programmation détermine également les méthodes de saisie possibles (•).

Langage de programmation	Groupe d'utilisateurs	Application	Saisie incrémentale	Saisie orientée source	Possibilité de redocumenter le bloc de la CPU
Liste d'instructions LIST	Utilisateurs voulant une programmation proche de la machine	Programmes optimisés en temps d'exécution et en espace mémoire	•	•	•
Schéma à contacts CONT	Utilisateurs habitués aux schémas de circuits	Programmation de commandes combinatoires	•	–	•
Logigramme LOG	Utilisateurs habitués aux boîtes logiques de l'algèbre booléenne	Programmation de commandes combinatoires	•	–	•
CONT F, LOG F Progiciel optionnel	Utilisateurs habitués aux langages de programmation CONT et LOG	Programmation de programmes de sécurité pour systèmes F	•	–	•
SCL (Structured Control Language) Progiciel optionnel	Utilisateurs ayant programmé en langages évolués comme Pascal ou C	Programmation de tâches de programmation de données	–	•	–



Langage de programmation	Groupe d'utilisateurs	Application	Saisie incrémentale	Saisie orientée source	Possibilité de redocumenter le bloc de la CPU
GRAPH Progiciel optionnel	Utilisateurs se basant sur la technologie, ayant peu de connaissances approfondies de la programmation ou des automates programmables	Description souple de processus séquentiels	•	–	•
HiGraph Progiciel optionnel	Utilisateurs se basant sur la technologie, ayant peu de connaissances approfondies de la programmation ou des automates programmables	Description souple de processus asynchrones non séquentiels	–	•	–
CFC Progiciel optionnel	Utilisateurs se basant sur la technologie, ayant peu de connaissances approfondies de la programmation ou des automates programmables	Description de processus continus	–	–	–

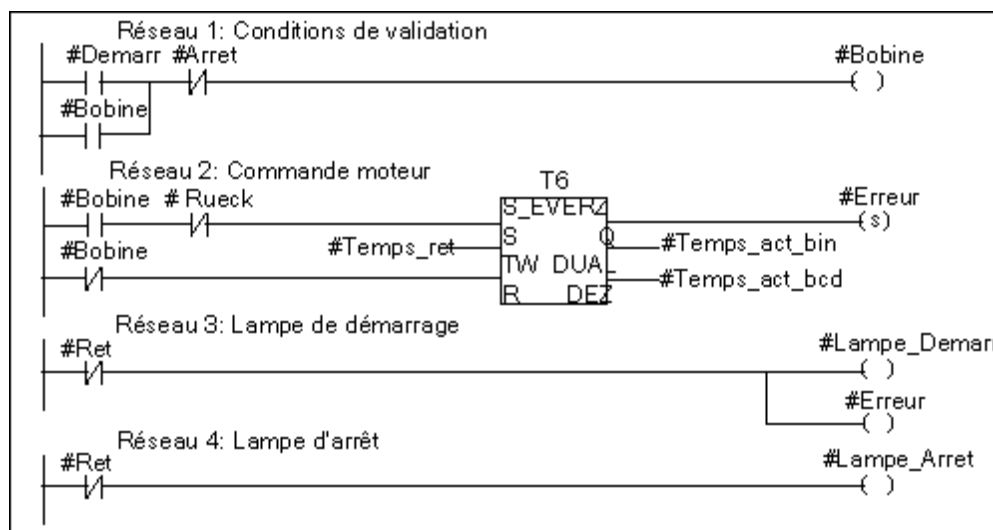
Pour les blocs exempts d'erreur, vous pouvez faire le va-et-vient entre les représentations de bloc dans les langages CONT, LOG et LIST. Les parties de programme ne pouvant pas être représentées dans le langage cible sont représentées en LIST.

Vous pouvez créer des blocs à partir de fichiers source en LIST et à partir de ces blocs, également à nouveau générer des sources.

### 9.2.1 Langage de programmation CONT (schéma à contacts)

La représentation en langage de programmation CONT (schéma à contacts) s'inspire des schémas de circuits. Les éléments d'un schéma de circuit, tels que contacts à fermeture et contacts à ouverture, sont rassemblés dans des réseaux. Un ou plusieurs réseaux forment la section des instructions complète d'un bloc de code.

#### Exemple de réseaux en CONT



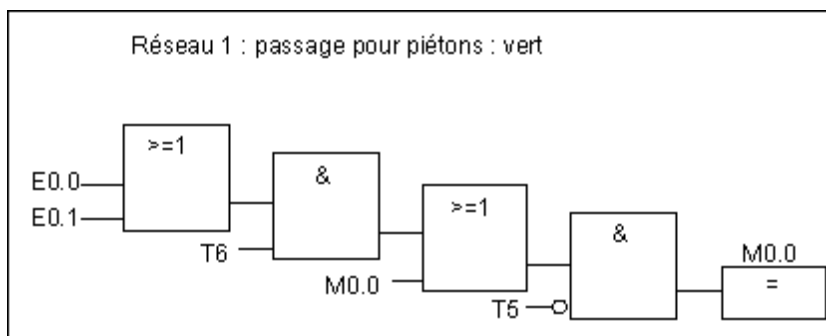
Le langage de programmation CONT fait partie du logiciel de base STEP 7. Dans le langage CONT, vous créez le programme en utilisant un éditeur incrémental.

## 9.2.2 Langage de programmation LOG (logigramme)

Le langage de programmation LOG (logigramme) utilise les boîtes fonctionnelles graphiques de l'algèbre booléenne pour représenter des éléments logiques. Il permet en outre de représenter des fonctions complexes, telles que les fonctions mathématiques en les mettant directement en liaison avec ces boîtes logiques.

Le langage de programmation LOG fait partie du logiciel de base STEP 7.

### Exemple d'un réseau en LOG



Dans le langage LOG, vous créez le programme en utilisant un éditeur incrémental.

### 9.2.3 Langage de programmation LIST (liste d'instructions)

Le langage de programmation LIST (liste d'instructions) est un langage textuel proche du langage machine. Chaque instruction correspond à une étape de l'exécution du programme par la CPU. Vous pouvez regrouper plusieurs instructions en réseaux.

#### Exemple de réseaux en LIST

```

Réseau 1 : Commande soupape de vidange
U(
O
O #Bobine
)
UN #Femer
= #Bobine
Réseau 2 : Indication "Soupape ouverte"
U #Bobine
= #Indic_Ouverte
Réseau 3 : Indication "Soupape fermée"
UN #Bobine
= #Indic_Fermee
```

Le langage de programmation LIST fait partie du logiciel de base STEP 7. Il vous permet d'éditer des blocs S7 avec des éditeurs incrémentaux ou de créer votre programme dans une source LIST avec un éditeur orienté source, puis de le compiler en blocs.

## 9.2.4 Langage de programmation SCL

Le langage de programmation SCL (Structured Control Language) optionnel est un langage évolué textuel, dont la structure du langage correspond pour l'essentiel à la norme CEI 1131-3. Grâce à ses instructions en langage évolué et contrairement au langage LIST, ce langage proche du PASCAL simplifie entre autres la programmation de boucles et de branches conditionnelles. SCL est de ce fait tout particulièrement adapté au calcul de formules, aux algorithmes d'optimisation complexes ou à la gestion de grandes quantités de données.

Dans le langage SCL, vous créez le programme dans une source SCL, en utilisant un éditeur orienté source.

### Exemple :

```
FUNCTION_BLOCK FB 20
VAR_INPUT
VALFINALE          INT;
END_VAR
VAR_IN_OUT
IQ1:               REAL;
END_VAR
VAR
INDEX:            INT;
END_VAR

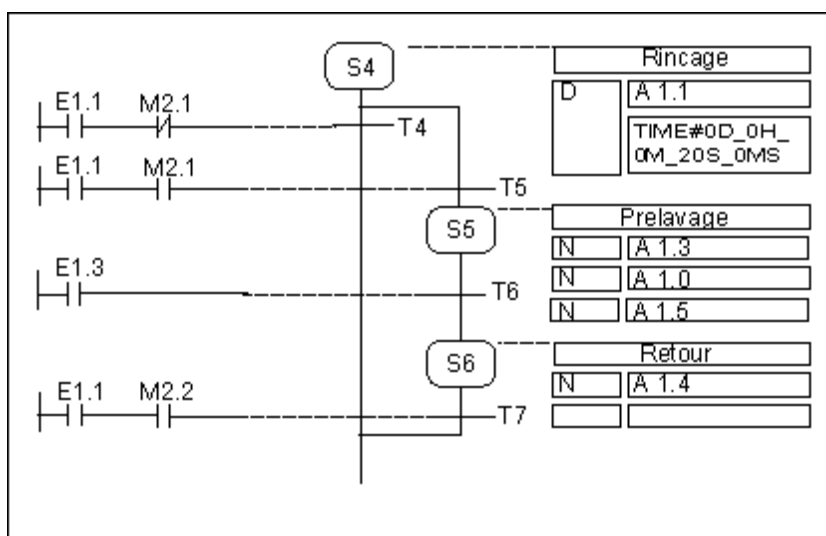
BEGIN
CONTROL          := FALSE;
FOR INDEX := 1 TO VALFINALE DO
    IQ1 := IQ1 * 2;
    IF IQ1 > 10000 THEN
        CONTROL = TRUE
    END_IF;
END_FOR;
END_FUNCTION_BLOCK
```

## 9.2.5 Langage de programmation GRAPH (commande séquentielle)

Le langage de programmation graphique optionnel GRAPH vous permet de programmer des commandes séquentielles. Ceci implique la création d'une succession d'étapes, la définition des actions associées à chaque étape et celle des transitions indiquant les possibilités d'évolution entre deux étapes successives. Pour définir les actions associées aux étapes, vous utilisez un langage de programmation spécial (similaire à LIST), alors que pour déterminer les conditions de réceptivité des transitions, vous utilisez une représentation sous forme de schéma à contacts (langage de programmation CONT restreint).

GRAPH permet la représentation très claire de séquences même complexes, ce qui favorise une programmation et une recherche d'erreurs efficaces.

### Exemple de commande séquentielle en GRAPH



### Blocs créés

Vous programmez le bloc fonctionnel contenant le graphe séquentiel avec l'éditeur GRAPH. Un bloc de données d'instance associé contient les données du graphe séquentiel, par exemple les paramètres du FB et les conditions pour les étapes et transitions. Vous créez ce DB d'instance automatiquement dans l'éditeur GRAPH.

### Fichier source

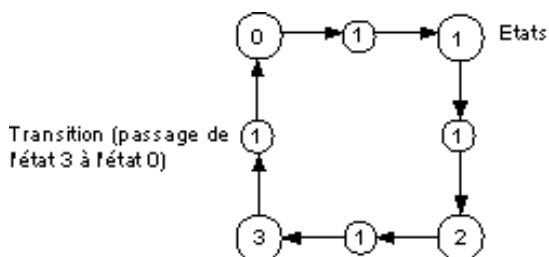
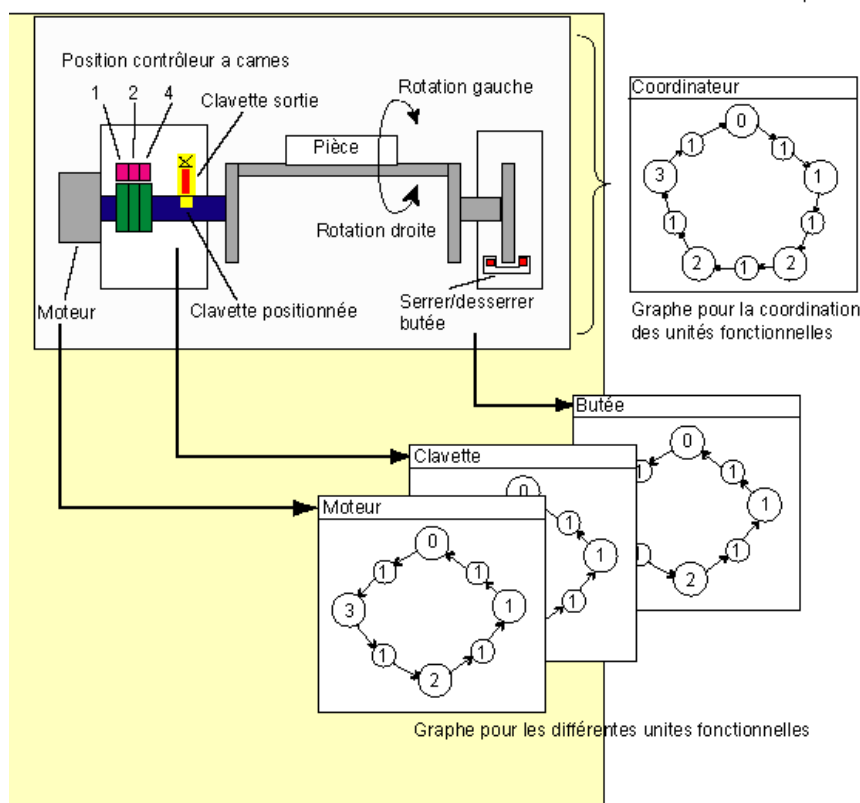
Il est possible de générer, à partir d'un FB créé avec GRAPH, un fichier source textuel (source GRAPH) pouvant être interprété par des pupitres opérateur (OP) ou des afficheurs de texte (TD) pour l'affichage de la commande séquentielle.

### 9.2.6 Langage de programmation HiGraph (graphe d'état)

Le langage de programmation graphique optionnel HiGraph vous permet de programmer certains blocs de votre programme comme graphes d'état. Vous décomposez alors votre installation en unités fonctionnelles indépendantes pouvant adopter différents états. Pour le passage d'un état à un autre, vous définissez des transitions. Vous décrivez les actions associées aux états, de même que les conditions de transition entre les états, dans un macro-langage fondé sur LIST.

Pour chaque unité fonctionnelle, vous créez un graphe qui en décrit le comportement. Tous les graphes d'une installation sont assemblés en groupes de graphes. Des informations de synchronisation d'unités fonctionnelles peuvent être échangées entre les graphes.

La représentation claire des transitions dans une unité fonctionnelle autorise une programmation systématique et facilite la recherche d'erreurs. A un instant donné, il n'y a jamais qu'un seul état actif dans HiGraph, contrairement à ce qui se passe avec les étapes de GRAPH. La figure suivante représente la création de graphes pour des unités fonctionnelles (exemple).



Un groupe de graphes est enregistré dans une source HiGraph dans le dossier Sources, sous le programme S7. Cette source sera ensuite compilée en blocs S7 pour le programme utilisateur.

La vérification de la syntaxe et des paramètres formels est réalisée après la dernière entrée pour un graphe (lorsque la fenêtre de travail se ferme). Les opérandes et mnémoniques ne sont vérifiés que durant la compilation de la source.



### 9.2.7 Langage de programmation CFC

Le logiciel optionnel CFC (*Continuous Function Chart*) est un langage de programmation permettant de regrouper des fonctions complexes en réseaux graphiques.

Ce langage de programmation vous permet de programmer en reliant graphiquement des fonctions données. De nombreuses fonctions standard que vous n'avez pas besoin de programmer vous-même sont accessibles dans des bibliothèques sous forme de blocs standard (par exemple pour des fonctions logiques, arithmétiques, ou encore des fonctions de régulation ou de gestion de données). L'utilisation du langage de programmation CFC ne requiert ni connaissance approfondie de la programmation, ni compétence spécifique dans le domaine des automates programmables. Vous pouvez ainsi porter toute votre attention sur la technologie propre à votre secteur d'activité.

Le programme créé est enregistré sous forme de diagrammes CFC. Ceux-ci sont stockés dans le dossier "Diagrammes" sous le programme S7. Les blocs S7 pour le programme utilisateur seront compilés à partir de ces diagrammes.

Si vous souhaitez créer vous-même les blocs à relier, vous pouvez les programmer dans l'un des langages de programmation S7 pour SIMATIC S7 ou dans les langages C/C++ pour SIMATIC M7.

## 9.3 Ce qu'il faut savoir pour créer des blocs

### 9.3.1 Dossier Blocs

Vous pouvez créer le programme pour une CPU S7 sous forme de :

- Blocs
- Sources

Pour stocker les blocs, vous disposez du dossier "Blocs" sous Programme S7.

Le dossier Blocs contient les blocs que vous allez charger dans la CPU S7 pour réaliser votre tâche d'automatisation. Ces blocs à charger englobent les blocs de code (OB, FB, FC) et les blocs de données (DB). Un bloc de code OB 1 vide est créé automatiquement dans le dossier Blocs, car sa présence dans la CPU S7 est indispensable pour l'exécution de votre programme.

Le dossier Blocs contient en outre les objets suivants :

- Les types de données utilisateur que vous créez. Ils vous facilitent la programmation, ne sont cependant pas chargés dans la CPU.
- Les tables de variables (VAT), que vous pouvez créer pour tester votre programme en visualisant et forçant des variables. Elles ne sont pas chargées dans la CPU.
- L'objet "Données système" (blocs de données système), contenant des informations relatives au système (configuration ou paramètres du système). Ces blocs de données système sont créés et des données y sont inscrites lors de la configuration du matériel.
- Les fonctions système (SFC) et les blocs fonctionnels système (SFB) que vous voulez appeler dans votre programme utilisateur. Les SFC et SFB eux-mêmes ne peuvent pas être édités.

Les blocs du programme utilisateur peuvent être édités dans les éditeurs correspondants, à l'exception des blocs de données système (qui ne sont créés et édités que lors de la configuration du système d'automatisation). Lorsque vous cliquez deux fois sur un bloc, l'éditeur correspondant démarre automatiquement.

---

#### Nota

Les blocs que vous avez programmés sous forme de sources, puis compilés sont également enregistrés dans le dossier Blocs.

---

### 9.3.2 Types de données utilisateur (UDT)

Les types de données utilisateur (user data type, UDT) sont des structures de données particulières, créées par vous. Vous pouvez les utiliser, une fois définis, dans l'ensemble du programme utilisateur S7.

- Vous pouvez utiliser les UDT comme types de données simples ou comme types de données complexes dans la déclaration des variables de blocs de code (FC, FB, OB) ou encore comme types de données pour des variables dans un bloc de données (DB). L'avantage réside dans le fait que vous ne définissez qu'une seule fois une structure de données spéciale, que vous allez utiliser plusieurs fois en l'affectant à un nombre illimité de variables.
- Les UDT peuvent servir de modèle afin de créer des blocs de données de même structure. Cela signifie que vous ne définissez qu'une seule fois la structure et créez ensuite les blocs de données requis par simple affectation du type de données utilisateur (exemple d'une recette : la structure du DB est toujours la même, seules les quantités varient).

Vous créez les types de données utilisateur – de la même manière que les autres blocs, – dans SIMATIC Manager ou dans l'éditeur incrémental.

#### Structure d'un UDT

Après l'ouverture de l'UDT, une nouvelle fenêtre de travail s'ouvre montrant la table - et, plus précisément, la vue des déclarations - pour ce type de données utilisateur.

- Les première et dernière lignes contiennent déjà les déclarations STRUCT et END\_STRUCT pour début et fin de type de données utilisateur ; vous ne pouvez pas modifier ces lignes.
- Pour éditer un type de données utilisateur, vous saisissez vos données dans les colonnes correspondantes à partir de la deuxième ligne de la table de déclaration.
- Vous pouvez structurer les types de données utilisateur à partir de :
  - types de données simples,
  - types de données complexes,
  - types de données utilisateur existant.

Les types de données utilisateur du programme utilisateur S7 ne sont pas chargés dans la CPU S7. Ils sont soit créés et édités directement avec des éditeurs incrémentaux, soit ils résultent de la compilation de sources.

### 9.3.3 Attributs de bloc

Les attributs de bloc vous permettent de mieux identifier les blocs créés (par exemple, grâce au numéro de version) ou de les protéger de modifications non autorisées.

Vous ne devez éditer les attributs d'un bloc que si ce bloc est ouvert. Outre les attributs éditables, la boîte de dialogue correspondante affiche également des données pour votre information : vous ne pouvez pas les éditer.

SIMATIC Manager affiche également les attributs de bloc et les attributs système dans les propriétés d'objet pour un bloc. Vous ne pouvez cependant y éditer que les attributs NOM, FAMILLE, AUTEUR et VERSION.

Vous pouvez éditer les propriétés de l'objet après insertion du bloc avec SIMATIC Manager. Pour un bloc qui n'a pas été créé avec SIMATIC Manager, mais avec l'un des éditeurs disponibles, ces informations (par exemple langage de programmation) figurent automatiquement dans les propriétés de l'objet.

#### Nota

Pour définir les abréviations à utiliser dans la programmation de vos blocs S7, choisissez dans SIMATIC Manager la commande **Outils > Paramètres**, puis l'onglet "Langue".

### Tableau des attributs de bloc

Lorsque vous indiquez des attributs de bloc, vous devez respecter l'ordre donné dans le tableau suivant.

Mot-clé / Attribut	Signification	Exemple
[KNOW_HOW_PROTECT]	Protection du bloc : il est impossible de visualiser la section des instructions d'un bloc compilé avec cette option. Il est uniquement possible de lire l'interface du bloc et non de la modifier.	KNOW_HOW_PROTECT
[AUTHOR:]	Nom de l'auteur, nom de la société, du service ou autres noms (8 caractères au maximum, sans espace)	AUTHOR : Siemens, mais pas de mot-clé
[FAMILY:]	Nom de la famille du bloc : par exemple, Regul (8 caractères au maximum, sans espace)	FAMILY : Regul, mais pas de mot-clé
[NAME:]	Nom du bloc (8 caractères au maximum)	NAME : PID, mais pas de mot-clé
[VERSION: int1 . int2]	Numéro de version du bloc (ces deux nombres entre 0 et 15, soit 0.0 à 15.15)	VERSION : 3.10
[CODE_VERSION1]	Identification indiquant si un FB admet des multi-instances ou non. Si vous voulez déclarer des multi-instances, le FB ne doit pas avoir cet attribut.	CODE_VERSION1
[UNLINKED] seulement pour DB	Les blocs de données possédant l'attribut UNLINKED sont uniquement stockés dans la mémoire de chargement, n'occupent pas d'espace dans la mémoire de travail et ne sont pas reliés au programme. Il n'est pas possible d'y accéder avec des commandes MC7. Les contenus de tels DB peuvent uniquement être transférés dans la mémoire de travail au moyen des SFC 20 BLKMOV ou SFC 83 READ_DBL, selon la CPU respective.	
[Non-Retain]	Les blocs de données possédant cet attribut reprennent leurs valeurs de chargement après chaque mise hors tension ou sous tension et après chaque commutation STOP-RUN de la CPU.	

Mot-clé / Attribut	Signification	Exemple
[READ_ONLY] seulement pour DB	Protection pour blocs de données : il est uniquement possible de lire les données et non de les modifier.	READ_ONLY
Bloc en lecture seule	Copie d'un bloc ayant été sauvegardé en lecture seule à des fins de référence. Cette propriété peut uniquement être attribuée dans l'éditeur de programmes avec la commande de menu <b>Fichier &gt; Sauvegarder en lecture seule.</b>	

Protéger un bloc par KNOW\_HOW\_PROTECT a les conséquences suivantes :

- Lorsque vous afficherez plus tard un bloc compilé dans l'éditeur CONT, LOG ou LIST incrémental, vous n'aurez pas accès à la section des instructions de ce bloc.
- Seules les variables de types de déclaration var\_in, var\_out et var\_in\_out seront visualisées dans la table de déclaration des variables du bloc. Les variables internes déclarées comme var\_stat et var\_temp seront masquées.

### Attributs de blocs possibles

Le tableau suivant présente les attributs que vous pouvez déclarer pour les différents types de blocs.

Attribut	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	•	•	•	•	–
AUTHOR	•	•	•	•	–
FAMILY	•	•	•	•	–
NAME	•	•	•	•	–
VERSION	•	•	•	•	–
UNLINKED	–	–	–	•	–
READ_ONLY	–	–	–	•	–
Non-Retain	–	–	–	•	–
Bloc en lecture seule	•	•	•	•	•

Vous pouvez définir l'attribut KNOW\_HOW\_PROTECT dans une source, lors de la programmation du bloc. Il sera affiché dans la boîte de dialogue des propriétés de bloc, mais ne pourra pas y être modifié.

### 9.3.4 Affichage de la longueur des blocs

La longueur des blocs est affichée en "octets".

#### Affichage dans les propriétés du dossier Blocs

Dans les propriétés du dossier Blocs, les longueurs suivantes sont affichées dans la vue hors ligne :

- longueur (somme de tous les blocs sans données système) dans la mémoire de chargement du système cible
- longueur (somme de tous les blocs sans données système) dans la mémoire de travail du système cible
- Les propriétés du dossier Blocs n'affichent pas les longueurs des blocs dans l'outil de développement (PG/PC).

#### Affichage dans les propriétés du bloc

Dans les propriétés du bloc, sont affichés :

- le nombre de données locales requises : longueur des données locales en octets,
- MC7 : longueur du code MC7 en octets ou longueur des données utiles de DB,
- longueur dans la mémoire de chargement du système cible,
- longueur dans la mémoire de travail du système cible : n'est affichée que lorsque l'affectation matérielle est connue.

Les affichages ne dépendent pas du fait que le bloc se trouve dans la fenêtre d'une vue en ligne ou hors ligne.

#### Affichage dans SIMATIC Manager (vue détaillée)

Lorsque vous ouvrez un dossier Blocs et que vous avez sélectionné la "vue détaillée", la mémoire de travail requise s'affiche dans la fenêtre du projet, que le dossier Blocs se trouve dans la fenêtre d'une vue en ligne ou hors ligne.

Vous pouvez additionner les longueurs de plusieurs blocs, en sélectionnant ces derniers. Leur somme s'affiche alors dans la ligne d'état de SIMATIC Manager.

Aucune longueur ne s'affiche pour les blocs qui ne peuvent pas être chargés dans le système cible (par exemple VAT).

La vue détaillée ne permet pas d'afficher les longueurs de bloc dans l'outil de développement (PG/PC).

### 9.3.5 Comparaison de blocs

#### Introduction

Pour comparer des blocs qui se trouvent à des endroits différents, vous avez les possibilités suivantes :

- Appel de la comparaison de blocs dans SIMATIC Manager avec la commande de menu **Outils > Comparer les blocs**. Lorsque vous cliquez sur le bouton "Aller à" dans la boîte de dialogue "Comparaison de blocs – Résultats" qui s'ouvre, la page d'onglet "Comparaison" est complétée dans l'éditeur de programmes (CONT/LOG/LIST).
- Appel de la comparaison de blocs dans l'éditeur de programmes avec la commande de menu **Outils > Comparer les partenaires en/hors ligne**.

Pour vous expliquer comment fonctionne la comparaison de blocs, nous allons distinguer les blocs de code (OB, FB, FC) et les blocs de données (DB).

#### Fonctionnement de la comparaison de blocs : bloc de code

Dans un premier temps, STEP 7 compare les horodatages des interfaces des blocs de code. Si ces horodatages sont les mêmes, STEP 7 considère que les interfaces sont identiques.

Si les horodatages des interfaces sont différents, STEP 7 compare les types de données des interfaces, section après section. En cas de différences, STEP 7 détermine respectivement la première différence d'une section, c'est-à-dire la **première** différence dans les sections de déclaration respectives. Les multiinstances et les UDT sont également comparés. Si les types de données dans les sections sont identiques, STEP 7 compare les valeurs de début des variables. Toutes les différences sont affichées.

Dans un second temps, STEP 7 vérifie le code, réseau par réseau (si l'option "Effectuer comparaison de code" n'était pas activée, une comparaison de code est malgré tout réalisée lorsque vous cliquez sur le bouton "Aller à" dans l'éditeur de programmes !).

STEP 7 détermine d'abord les réseaux ayant été insérés ou supprimés. Dans le résultat de la comparaison, les réseaux figurant dans un seul bloc sont commentés par "uniquement contenu dans".

Les réseaux restant sont ensuite respectivement comparés jusqu'à la **première** instruction différente. La comparaison est réalisée de la manière suivante :

- dans le cas du paramétrage "Priorité de l'opérande absolu", en fonction des opérandes absolus,
- dans le cas du paramétrage "Priorité du mnémonique", en fonction du mnémonique.

Des instructions sont identiques lorsque l'opérateur et l'opérande sont identiques.

Si les blocs à comparer ont été programmés dans des langages différents, STEP 7 les compare dans le langage de programmation LIST.

Particularité de la comparaison hors ligne :

- A la différence d'une comparaison hors ligne – en ligne, la comparaison hors ligne – hors ligne permet à STEP 7 de déterminer également les noms de variable différents. Ceci n'est pas possible lors de la comparaison hors ligne – en ligne, car seuls des mnémoniques de remplacement sont disponibles en ligne.
- Les commentaires de blocs, de réseaux et de lignes ainsi que d'autres attributs de blocs (p. ex. les informations sur S7-PDIAG et les messages) sont exclus de la comparaison.

## Fonctionnement de la comparaison de blocs : blocs de données

Dans un premier temps, STEP 7 compare les horodatages des interfaces des blocs de données, comme pour les blocs de code. Si ces horodatages sont les mêmes, STEP 7 considère que les structures de données sont identiques.

Si les horodatages des interfaces sont différents, STEP 7 compare les structures jusqu'à la **première** différence dans la structure de données. Si les structures de données dans les sections sont identiques, STEP 7 compare les valeurs de début et valeurs actuelles des variables. Toutes les différences sont affichées.

Particularité de la comparaison hors ligne – hors ligne :

- A la différence d'une comparaison hors ligne – en ligne, la comparaison hors ligne – hors ligne permet à STEP 7 de déterminer également les noms de variable différents. Ceci n'est pas possible lors de la comparaison hors ligne – en ligne, car seuls des mnémoniques de remplacement sont disponibles en ligne.
- Les commentaires ainsi que les structures des UDT utilisés dans le bloc de données sont exclus de la comparaison.

## Fonctionnement de la comparaison de blocs : types de données (UDT)

Dans un premier temps, STEP 7 compare les horodatages des interfaces des types de données, comme pour les blocs de données. Si ces horodatages sont les mêmes, STEP 7 considère que les structures de données sont identiques.

Si les horodatages des interfaces sont différents, STEP 7 compare les structures jusqu'à la **première** différence dans la structure de données. Si les structures de données dans les sections sont identiques, STEP 7 compare les valeurs de début. Toutes les différences sont affichées.

## Procédure : comparaison de blocs dans l'éditeur de programmes

1. Ouvrez le bloc que vous souhaitez comparer avec la version chargée.
2. Choisissez la commande de menu **Outils > Comparer le partenaire en ligne/hors ligne**.

Si le partenaire en ligne est accessible, les résultats de la comparaison s'affichent dans la page d'onglet "7 : comparer", dans la partie inférieure de la fenêtre de l'éditeur de programmes.

**Astuce** : si deux réseaux ont été déterminés comme étant "différents", vous pouvez ouvrir le réseau correspondant par double clic sur cette ligne.



### Procédure : comparaison de blocs dans SIMATIC Manager

1. Dans SIMATIC Manager, sélectionnez le dossier Blocs ou uniquement les blocs que vous souhaitez comparer.
2. Choisissez la commande de menu **Outils > Comparer les blocs**.
3. Dans la boîte de dialogue "Comparer les blocs", sélectionnez le type de comparaison (EN LIGNE / hors ligne ou Chemin1/chemin2).
4. Dans le cas de la comparaison Chemin1/chemin2 : dans SIMATIC Manager, choisissez le dossier Blocs ou les blocs avec lesquels vous souhaitez effectuer la comparaison. Les blocs sont alors saisis automatiquement dans la boîte de dialogue.
5. Cochez la case "Y compris SDB", si les SDB doivent également être comparés.
6. Cochez la case "Réaliser la comparaison du code", si vous souhaitez en plus effectuer une comparaison du code. Lorsque cette case à cocher est activée, vous pouvez ensuite définir si une "Comparaison détaillée" doit être réalisée. Dans le cas de la comparaison détaillée, les modifications du nom de variables locales et du nom de paramètres sont également affichées en plus des parties significatives pour l'exécution des blocs (interfaces et code). La case à cocher "également blocs avec langages différents" permet de déterminer si les blocs créés dans un autre langage doivent également être comparés (p. ex. LIST, LOG,...). Dans ce cas, les blocs sont comparés sur la base de LIST.
7. Cliquez sur le bouton "OK" dans la boîte de dialogue.

Les résultats de la comparaison s'affichent dans une boîte de dialogue distincte "Comparaison de blocs – Résultats".

8. Si, dans cette boîte de dialogue, vous cliquez également sur le bouton "Détails", les propriétés des blocs comparés s'affichent (p. ex. le moment des dernières modifications, les sommes de contrôle, etc.).

Lorsque vous cliquez sur le bouton "Aller à ", l'éditeur de programmes s'ouvre et les résultats de la comparaison sont visibles dans la partie inférieure de la fenêtre.

---

#### Nota

Lors de la comparaison d'un dossier Bloc hors ligne avec un dossier Blocs en ligne, seuls les types de blocs qu'il est possible de charger sont comparés (OB, FB, ...).

Lors de la comparaison Hors ligne/en ligne ou Chemin1/chemin2 d'une sélection multiple, tous les blocs sélectionnés sont comparés, même s'il s'agit de blocs qu'il n'est pas possible de charger (p. ex. tables de variables ou UDT).

---

### 9.3.6 Réassignation

Vous pouvez réassigner les blocs et opérands suivants :

- entrées, sorties,
- mémentos, temporisations, compteurs,
- fonctions, blocs fonctionnels.

Procédez de la manière suivante :

1. Dans SIMATIC Manager, sélectionnez le dossier Blocs contenant les blocs que vous souhaitez réassigner.
2. Choisissez la commande **Outils > Réassignation**.
3. Dans la boîte de dialogue "Réassignation" qui s'affiche, entrez les remplacements souhaités (ancien opérande / nouvel opérande) dans le tableau.
4. Sélectionnez l'option "Pour tous les opérandes de la plage spécifiée" si vous souhaitez réassigner des plages d'opérandes (BYTE, WORD, DWORD).  
Par exemple, si vous avez indiqué EW0 et EW4 pour la réassignation de la plage d'opérandes, les opérandes E0.0 à E1.7 deviendront E4.0 à E5.7. Les opérandes de la plage réassignée (par ex. E0.1) ne pourront plus alors être entrés individuellement dans le tableau.
5. Cliquez sur le bouton "OK".

Vous démarrez ainsi la réassignation. Lorsqu'elle est terminée, vous pouvez décider dans une boîte de dialogue si vous souhaitez voir le fichier d'information relatif à la réassignation. Ce fichier contient la liste des opérandes "Ancien opérande" et "Nouvel opérande". Les blocs y figurent individuellement avec le nombre de réassignations qui y ont été effectuées.

Lors d'une réassignation, il faut faire attention aux points suivants :

- Quand vous réassignez un bloc (c'est-à-dire le renommez), le nouveau bloc ne doit pas déjà exister. S'il existe déjà, l'opération sera annulée.
- Quand vous réassignez un bloc fonctionnel (FB), son DB d'instance est associé automatiquement au FB réassigné, mais il ne change pas (son numéro de DB reste le même).

### 9.3.7 Attributs pour blocs et pour paramètres

La description des attributs figure dans l'aide de référence pour les attributs système.

## 9.4 Utilisation de bibliothèques

Une bibliothèque permet de stocker des composants de programmes réutilisables, destinés aux automates programmables SIMATIC S7/M7. Ces éléments peuvent être copiés dans une bibliothèque à partir de projets existants ou y être créés directement, indépendamment de tout projet.

En enregistrant les blocs que vous souhaitez utiliser fréquemment dans un programme S7 sous une bibliothèque, vous évitez de les programmer à nouveau. Il vous suffira à chaque fois de les copier dans le programme utilisateur correspondant.

Pour créer des programmes S7/M7 dans une bibliothèque, vous disposez des mêmes fonctionnalités que dans un projet, à l'exception des fonctions de test.

### Création de bibliothèques

Vous créez les bibliothèques de la même manière que les projets, en choisissant la commande **Fichier > Nouveau**. La nouvelle bibliothèque va être créée dans le répertoire que vous avez sélectionné pour les bibliothèques, lorsque vous avez choisi la commande **Outils > Paramètres** et l'onglet "Général".

---

#### Nota

SIMATIC Manager accepte des noms d'une longueur excédant 8 caractères. Le nom du répertoire de la bibliothèque est tronqué après 8 caractères. Les 8 premiers caractères doivent donc suffire à identifier la bibliothèque. Aucune distinction n'est faite entre les majuscules et minuscules. Lorsque vous voulez ouvrir une bibliothèque à l'aide de "Parcourir", le nom entier s'affiche, alors que lorsque vous feuilletez, il apparaît sous forme abrégée.

Sachez que vous ne pouvez pas utiliser de bibliothèques d'une version de STEP 7 plus récente dans des projets d'une ancienne version de STEP 7.

---

### Ouverture de bibliothèques

Pour ouvrir une bibliothèque, choisissez d'abord la commande **Fichier > Ouvrir**. Sélectionnez ensuite une bibliothèque dans les boîtes de dialogue suivantes. La fenêtre de bibliothèque s'ouvre alors.

---

#### Nota

Si la bibliothèque voulue n'apparaît pas dans la liste de bibliothèques, cliquez sur le bouton "Parcourir" dans la boîte de dialogue "Ouvrir". La boîte de dialogue standard de Windows affiche alors l'arborescence de répertoires dans laquelle vous pouvez rechercher la bibliothèque.

Le nom du fichier correspond toujours au nom initial de la bibliothèque créée, ce qui veut dire que les changements de nom du fichier ne sont pas visibles dans SIMATIC Manager.

Dès lors que vous sélectionnez la bibliothèque, elle est reprise dans la liste de bibliothèques. Vous pouvez modifier les entrées dans la liste de bibliothèques en choisissant la commande **Fichier > Gérer**.

---

### **Copie de bibliothèques**

Pour copier une bibliothèque, vous l'enregistrez sous un autre nom en choisissant la commande **Fichier > Enregistrer sous** .

Pour copier des composants de bibliothèques comme les programmes, blocs, sources, etc., vous choisissez la commande **Edition > Copier**.

### **Suppression de bibliothèques**

Pour supprimer une bibliothèque, vous choisissez la commande **Fichier > Supprimer**.

Pour effacer des composants de bibliothèques comme les programmes, blocs, sources etc., vous choisissez la commande **Edition > Effacer**.

### 9.4.1 Structure hiérarchique des bibliothèques

De même qu'un projet, une bibliothèque possède une structure hiérarchique :

- Une bibliothèque peut contenir des programmes S7/M7.
- Un programme S7 peut contenir exactement un dossier Blocs (programme utilisateur), un dossier Sources, un dossier Diagrammes ainsi qu'un objet "Mnémoniques" (table des mnémoniques).
- Un programme M7 peut contenir des diagrammes et programmes C pour les modules programmables M7 ainsi que l'objet "Mnémoniques" (table des mnémoniques) et un dossier Blocs pour les DB et tables de variables.
- Le dossier Blocs comprend les blocs que vous pouvez charger dans la CPU S7. Il contient également les tables des variables (VAT) et les types de données utilisateur qui eux, ne peuvent pas être chargés dans la CPU.
- Le dossier Sources contient les sources pour les programmes créés dans les différents langages de programmation.
- Le dossier Diagrammes contient les diagrammes CFC (uniquement avec le logiciel optionnel CFC).

Lorsque vous créez un nouveau programme S7/M7, un dossier Blocs et un dossier Sources (uniquement pour S7) ainsi qu'un objet "Mnémoniques" y sont automatiquement insérés.

### 9.4.2 Présentation des bibliothèques standard

Le logiciel de base STEP 7 contient les bibliothèques standard suivantes :

- **System Function Blocks** : fonctions système (SFC) et blocs fonctionnels système (SFB) ;
- **S5-S7 Converting Blocks** : blocs servant à convertir d'anciens programmes de STEP 5 ;
- **IEC Function Blocks** : blocs servant aux fonctions, par exemple, modification de la date et de l'heure, opérations de comparaison, traitement de chaînes et choix de maxima et minima ;
- **Organization Blocks** : blocs d'organisation (OB) standard ;
- **PID Control Blocks** : blocs fonctionnels (FB) pour la régulation PID
- **Communication Blocks** : fonctions (FC) et blocs fonctionnels (FB) pour les CP SIMATICNET ;
- **TI-S7 Converting Blocks** : fonctions standard à usage général ;
- **Miscellaneous Blocks** : blocs pour l'horodatage et la synchronisation d'horloge ;

D'autres bibliothèques peuvent s'ajouter à celle-ci lors de l'installation de logiciels optionnels.

## Suppression et installation de bibliothèques fournies

Vous pouvez, dans SIMATIC Manager, supprimer puis réinstaller les bibliothèques fournies. Pour l'installation, vous exécutez une nouvelle fois le programme Setup de STEP 7.

---

### Nota

Les bibliothèques fournies sont toujours copiées lorsque vous installez STEP 7. Si vous avez modifié des bibliothèques fournies, elles seront écrasées par l'original lors d'une nouvelle installation de STEP 7.

Il est donc recommandé de copier les bibliothèques fournies avant de les modifier et de ne modifier que la copie.

---

# 10 Création de blocs de code

## 10.1 Principes de la création de blocs de code

### 10.1.1 Structure de la fenêtre de l'éditeur de programmes

La fenêtre de l'éditeur de programme comporte les zones suivantes :

#### **Vues d'ensemble**

L'onglet "Éléments de programme" affiche une vue d'ensemble complète des éléments de programme que vous pouvez insérer dans votre programme CONT, LOG ou LIST. L'onglet "Structure d'appels" affiche la hiérarchie d'appel des blocs dans le programme S7 en cours.

#### **Déclaration de variables**

La déclaration de variables est divisée en vue d'ensemble des variables et en vue de détail des variables.

#### **Instructions**

La section d'instructions affiche le code du bloc devant être traité par l'automate programmable. Elle comporte un ou plusieurs réseaux.

#### **Détails**

Les divers onglets de la fenêtre "Détails" permettent p. ex. l'affichage de messages d'erreur, l'affichage d'informations sur les opérandes, l'édition de mnémoniques, le forçage d'opérandes, la comparaison de blocs et l'édition de définitions d'erreur pour le diagnostic du processus.

The screenshot displays the SIMATIC Manager interface. On the left is a project tree with various function blocks. The main workspace shows a ladder logic network (Réseau 3) for a 'Bascule SR' (SR Flip-flop). The network includes two normally open contacts labeled 'Automatique Marche' and 'Manuel Marche', connected to the 'S' (Set) input of an SR block. The 'Q' (Output) of the SR block is connected to a coil labeled 'Mode automatique'. Below the network, the start of 'Réseau 4' is visible.

At the bottom of the window, a table lists the operands used in the network:

	Opérande	Mnémorique	Format d'affichage	Valeur d'état
1	E 0.5	"Automatique Marche"	BOOLEEN	
2	A 4.2	"Mode automatique"	BOOLEEN	
3	E 0.6	"Manuel Marche"	BOOLEEN	

Navigation buttons for '3: Références croisées' and '4: Informations opérandes' are located below the table. The status bar at the bottom indicates 'Pour obtenir de l'aide, appuyez sur F1.' and shows the system is 'offline'.

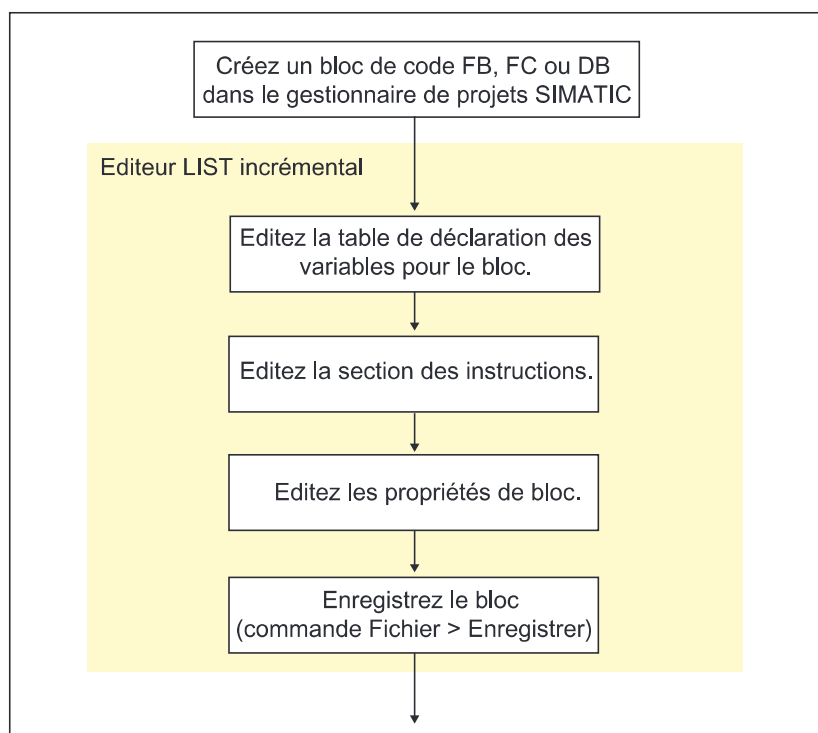


### 10.1.2 Marche à suivre pour la création de blocs de code

Les blocs de code (OB, FB, FC) comportent une section de déclaration de variables et une section d'instructions. De plus, ils possèdent également des propriétés. Lors de la programmation, vous devez par conséquent éditer les trois parties suivantes :

- **déclaration des variables** : vous y définissez les paramètres, attributs système des paramètres ainsi que les variables locales du bloc.
- **section des instructions** : vous y programmez le code du bloc que l'automate programmable doit exécuter. La section des instructions comporte un ou plusieurs réseaux. Pour créer les réseaux, vous disposez par exemple des langages de programmation LIST (liste d'instructions), CONT (schémas à contacts) et LOG (logigramme).
- **propriétés de bloc** : elles contiennent des informations supplémentaires comme l'horodatage ou l'indication du chemin qui sont entrées par le système. Vous pouvez également entrer vous-même des données sur le nom, la famille, la version et l'auteur ou bien affecter des attributs système aux blocs.

En principe, vous pouvez éditer les parties d'un bloc de code dans un ordre quelconque. Vous avez bien entendu aussi la possibilité de les corriger/compléter.



#### Nota

Lorsque vous souhaitez reprendre des mnémoniques de la table des mnémoniques, vous devriez d'abord vérifier qu'ils sont présents et, le cas échéant, les compléter.

### 10.1.3 Présélections pour l'éditeur de programmes CONT/LOG/LIST

Avant de commencer la programmation, vous devriez vous familiariser avec les possibilités de présélection afin de travailler aisément selon vos habitudes.

La commande **Outils > Paramètres** ouvre une boîte de dialogue à onglets. Dans les différents onglets, vous effectuez les présélections pour la programmation de blocs, par exemple dans l'onglet "Général":

- l'écriture (police et taille) dans le texte et les tableaux,
- si vous souhaitez afficher immédiatement les mnémoniques et le commentaire pour un nouveau bloc.

Vous pouvez aussi modifier les présélections pour le langage, le commentaire et les mnémoniques durant l'édition à l'aide des commandes **Affichage > ...**

Vous modifiez la couleur représentant, par exemple, des sélections de réseaux ou de sections des instructions dans la page d'onglet "CONT/LOG".

### 10.1.4 Droits d'accès aux blocs ou aux sources

Une base de données commune est souvent utilisée lors du traitement d'un projet. Il peut donc arriver que plusieurs personnes veuillent accéder au même bloc ou à la même source de données.

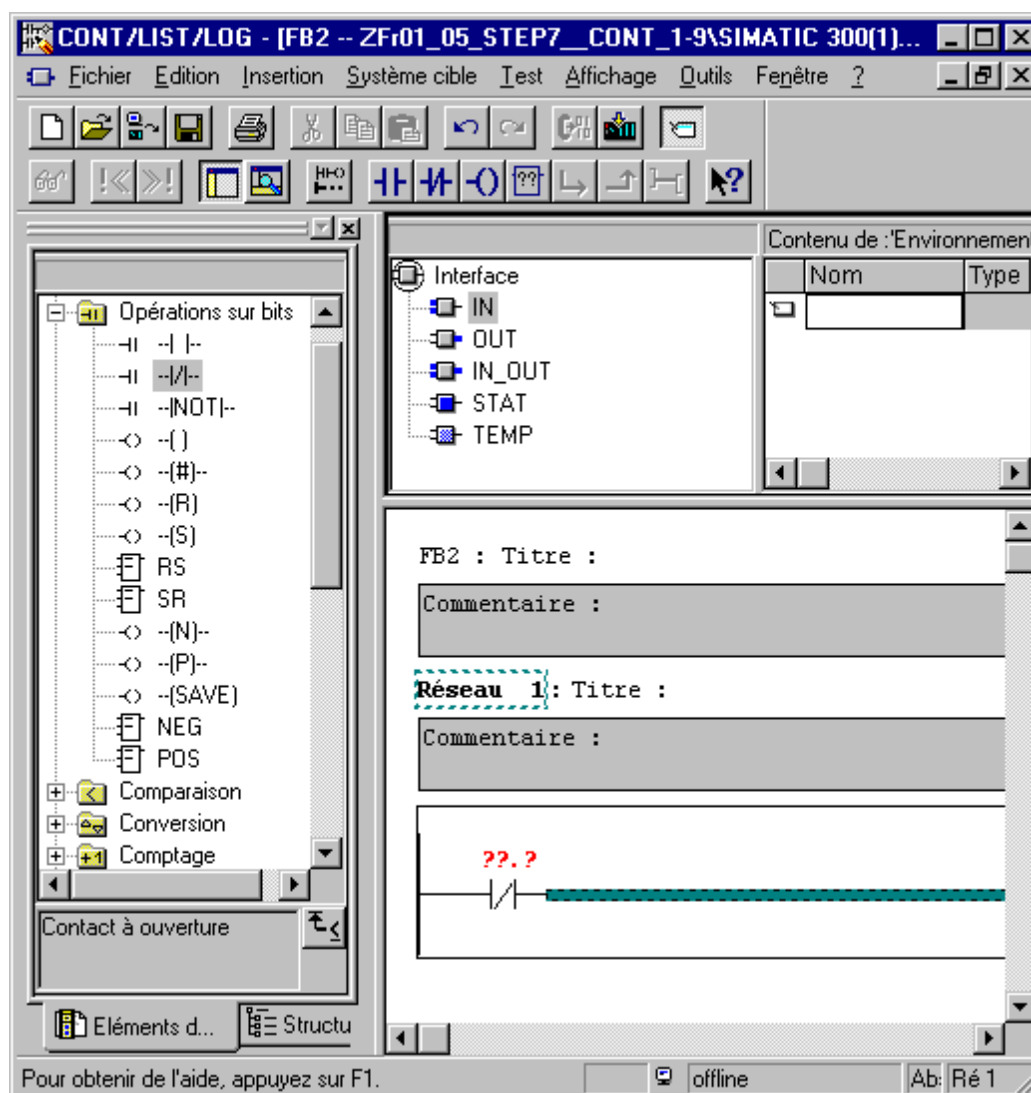
Les droits de lecture et d'écriture sont attribués comme suit :

- **Traitement hors ligne**  
Lors de l'ouverture d'un bloc ou d'une source, le logiciel vérifie si cet objet peut être ouvert avec le droit *Ecriture*. Si la source ou le bloc est déjà ouvert, il n'est possible de travailler que sur une copie. Si vous voulez enregistrer la copie, le logiciel vous demande si vous voulez la substituer à l'original ou si vous désirez la sauvegarder sous un autre nom.
- **Traitement en ligne**  
Si vous ouvrez un bloc en ligne par l'intermédiaire d'une liaison configurée, le bloc hors ligne associé est verrouillé, rendant ainsi sa modification simultanée impossible.

### 10.1.5 Instructions dans la vue d'ensemble des éléments de programme

L'onglet "Éléments de programme" dans la fenêtre de la vue d'ensemble met à votre disposition des éléments de langage CONT et LOG ainsi que des multi-instances déjà déclarées, des blocs finis et des blocs de bibliothèques. Vous appelez cet onglet en choisissant la commande de menu **Affichage > Vues d'ensemble**. Vous pouvez également reprendre des éléments de programme dans la section des instructions en choisissant la commande **Insertion > Éléments de programme**.

#### Exemple de l'onglet "Éléments de programme" dans CONT



## 10.2 Edition de la déclaration des variables

### 10.2.1 Utilisation de la déclaration des variables dans les blocs de code

Lorsque vous ouvrez un bloc de code, la vue d'ensemble et la vue de détail des variables s'affichent dans le volet supérieur d'une fenêtre et la section des instructions, dans laquelle vous éditez le code du bloc proprement dit, dans le volet inférieur.

#### Exemple : vue des variables et section des instructions en LIST

The screenshot shows the SIMATIC Manager interface with the following components:

- Left Panel (Tree View):** Shows a project tree with 'Interface' expanded to 'IN', where 'Switch\_On' is selected.
- Top Right Panel (Table):** Titled 'Contenu de : "Environnement\Interface\IN"', it displays a table of variables.
 

Nom	Type de données	Adresse	Valeur
Switch_On	Bool	0.0	
Switch_Off	Bool	0.1	
Failure	Bool	0.2	
Actual_Speed	Int	2.0	
- Bottom Panel (Code Editor):** Shows the functional block 'FB1 : Bloc fonctionnel "Commande de moteur"' and the instruction list for 'Réseau 1':
 

```

Réseau 1: Mise en marche du moteur, négation des signaux
      U      #Switch_On
      UN    "Mode automatique"
      S      #Engine_On
      O      #Switch_Off
      ON    #Failure
      R      #Engine_On
      
```
- Status Bar:** Displays 'Pour obtenir de l'aide, appuyez sur F1.' and 'offline'.

Dans la vue de détail des variables, vous définissez les variables locales y compris les paramètres formels du bloc et les attributs système pour les paramètres. En voici certains effets :

- La déclaration sert à réserver l'espace mémoire correspondant pour les variables temporaires dans la pile des données locales, dans le cas de blocs fonctionnels, pour les variables statiques dans le DB d'instance ultérieurement affecté.
- En définissant les paramètres d'entrée, de sortie, d'entrée/sortie, vous déterminez également "l'interface" pour l'appel du bloc dans le programme.
- Lorsque vous déclarez les variables dans un bloc fonctionnel, celles-ci (à l'exception des variables temporaires) déterminent également la structure de données pour chaque DB d'instance que vous affecterez au FB.
- En définissant les attributs système, vous affectez aux paramètres des propriétés particulières pour la configuration des messages et des liaisons, pour les fonctions de contrôle-commande et pour la configuration de processus de conduite.

### 10.2.2 Relation entre la vue de détail des variables et la section des instructions

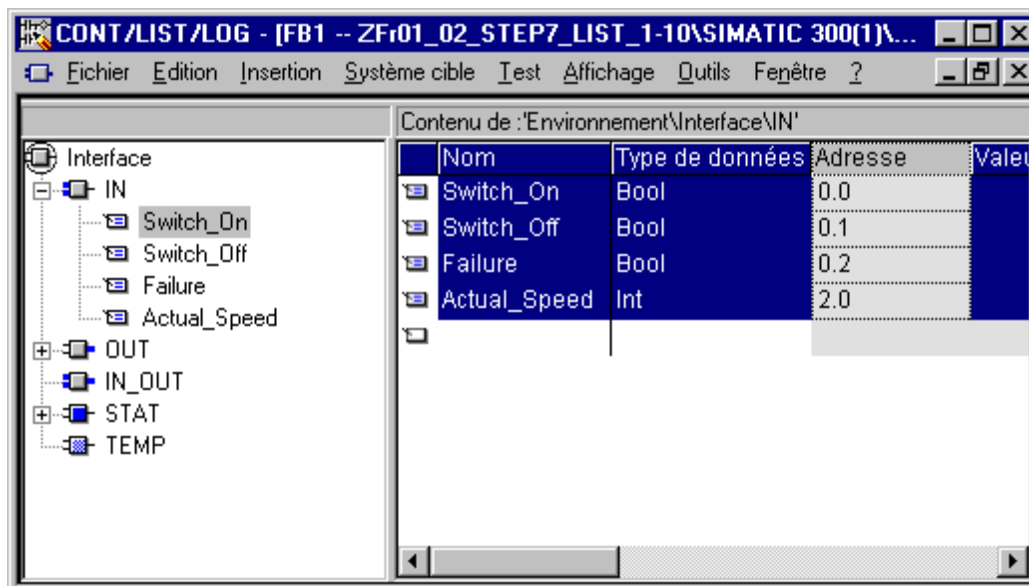
La déclaration des variables et la section des instructions de blocs de code sont fortement liées, puisque lors de la programmation, la section des instructions utilise les noms de la déclaration des variables. Les modifications dans la table de déclaration des variables se répercutent donc dans l'ensemble de la section des instructions.

Action dans la table de déclaration des variables	Réaction dans la section des instructions
Nouvelle entrée correcte	Dans le cas d'un code erroné, la variable précédemment non déclarée devient valide.
Changement de nom sans modification de type correct	Le mnémonique est immédiatement partout représenté avec le nouveau nom.
Nom correct changé en nom invalide	Le code n'est pas modifié.
Nom invalide changé en nom correct	Dans le cas d'un code erroné, celui-ci devient valide.
Modification du type	Dans le cas d'un code erroné, celui-ci devient valide et dans le cas d'un code correct, celui-ci devient invalide dans certaines conditions.
Suppression d'une variable (mnémonique) utilisée dans le code	Le code correct devient invalide.

Les modifications de commentaire, la saisie erronée d'une nouvelle variable, la modification d'une valeur initiale ou la suppression d'une variable non utilisée n'ont aucun effet sur la section des instructions.

### 10.2.3 Structure de la fenêtre de déclaration des variables

La fenêtre de déclaration des variables comporte la vue d'ensemble des variables et la vue de détail des variables.



Aussitôt que vous ouvrez un bloc de code nouvellement créé, la vue d'ensemble des variables par défaut s'affiche. Celle-ci affiche uniquement la liste des types de déclaration autorisés pour le type de bloc sélectionné (in, out, in\_out, stat, temp) et ce, dans l'ordre défini. Lorsque vous créez un nouvel OB, une déclaration de variables standard s'affiche. Vous avez la possibilité d'en modifier les valeurs.

Les types de données autorisés pour les données locales des différents types de blocs sont décrits dans Affectation de types de données aux données locales de blocs de code.

## 10.3 Multi-instances dans la déclaration des variables

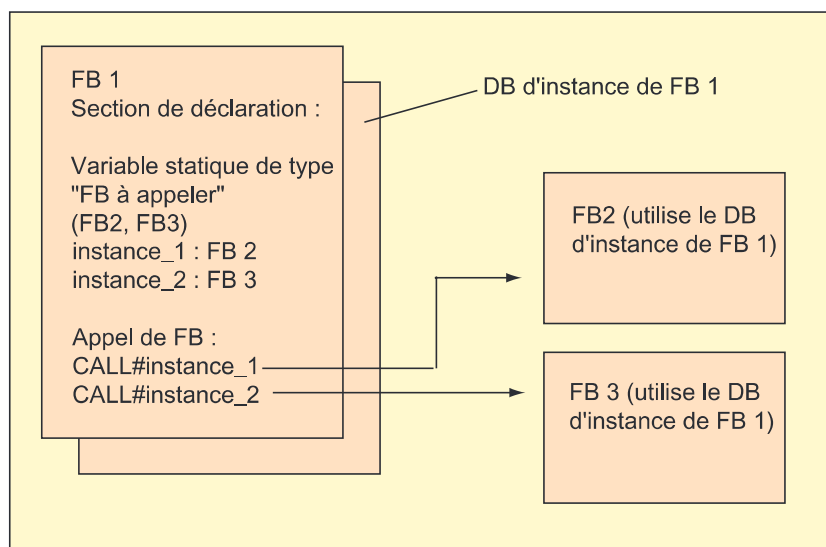
### 10.3.1 Utilisation de multi-instances

Il est probable qu'en raison des caractéristiques (par exemple espace mémoire) des CPU S7 utilisées, vous ne puissiez ou ne souhaitiez affecter qu'un nombre limité de blocs de données aux données d'instance. Lorsque vous appelez des blocs fonctionnels supplémentaires existant dans un FB de votre programme utilisateur (hiérarchie d'appel de FB), vous pouvez appeler ces blocs fonctionnels supplémentaires sans qu'ils ne possèdent leur propres DB d'instance (c'est-à-dire sans DB d'instance supplémentaires).

Vous pouvez appliquer la solution suivante :

- Entrez les FB à appeler comme variables statiques dans la déclaration des variables du FB appelant.
- Dans ce bloc fonctionnel vous pouvez appeler d'autre blocs fonctionnels sans qu'ils ne possèdent leur propres DB d'instance (c'est-à-dire sans DB d'instance supplémentaires).
- Vous pouvez ainsi regrouper les données d'instance dans un seul DB d'instance et, ainsi, mieux utiliser le nombre de DB disponibles.

L'exemple suivant illustre la solution décrite : FB 2 et FB 3 utilisent le DB d'instance du bloc fonctionnel FB 1 qui les appelle.



Condition unique : vous devez "indiquer" au bloc fonctionnel appelant, quelles instances vous appelez et de quel type (de FB) elles sont . Ces indications doivent être réalisées dans la fenêtre de déclaration du FB appelant. Le FB à utiliser doit posséder au minimum une variable ou un paramètre de la zone de données (c'est-à-dire pas VAR\_TEMP).

N'utilisez pas de blocs de données multi-instance tant que des modifications en ligne sont attendues lorsque la CPU est en marche. Un rechargement régulier n'est garanti que si vous utilisez des blocs de données d'instance.

### 10.3.2 Règles pour la formation de multi-instances

Les règles suivantes s'appliquent à la déclaration des multi-instances :

- La déclaration de multi-instances n'est possible que dans les blocs fonctionnels créés dans STEP 7 à partir de la version 2 (cf. attribut de bloc dans les propriétés du FB).
- Pour la déclaration de multi-instances, le bloc fonctionnel doit avoir été créé comme FB admettant les multi-instances (présélection à partir de STEP 7 Version x.x ; peut être désactivée à l'aide de la commande **Outils > Paramètres** dans l'éditeur).
- Vous devez affecter un DB d'instance au bloc fonctionnel dans lequel vous déclarez une multi-instance.
- Une multi-instance ne peut être déclarée que comme variable statique (type de déclaration "stat").

---

#### Nota

- Vous pouvez également créer des multi-instances pour les blocs fonctionnels système.
  - Si le FB n'a pas été conçu pour admettre des multi-instances et que cette qualité s'avère nécessaire par la suite, vous pouvez générer une source à partir du FB dans laquelle vous supprimez la propriété de bloc CODE\_VERSION1 pour la recompiler ensuite en FB.
- 

### 10.3.3 Saisie de multi-instances dans la fenêtre de déclaration des variables

1. Ouvrez le FB à partir duquel les FB imbriqués doivent être appelés.
2. Définissez, dans la déclaration des variables du FB appelant, une variable statique pour chaque appel d'un bloc fonctionnel pour l'instance duquel vous ne voulez pas préciser de DB.
  - Dans la vue d'ensemble des variables, sélectionnez le niveau hiérarchique "STAT".
  - Dans la colonne "Nom" de la vue de détail des variables, saisissez une désignation pour l'appel du FB.
  - Dans la colonne "Type de données", saisissez le FB à appeler, sous forme d'adresse absolue ou sous forme de mnémonique.
  - Vous pouvez saisir d'éventuels explications dans la colonne de commentaire.

#### Appels dans la section des instructions

Si vous avez déclaré des multi-instances, vous pouvez utiliser des appels de FB sans indication d'un DB d'instance.

Exemple : la variable statique "Nom : Moteur\_1, Type de données : FB20" étant définie, l'instance peut être appelée de la manière suivante :

```
CALL Moteur_1      // Appel du FB 20 sans DB d'instance
```



## 10.4 Remarques générales sur la saisie d'instructions et de commentaires

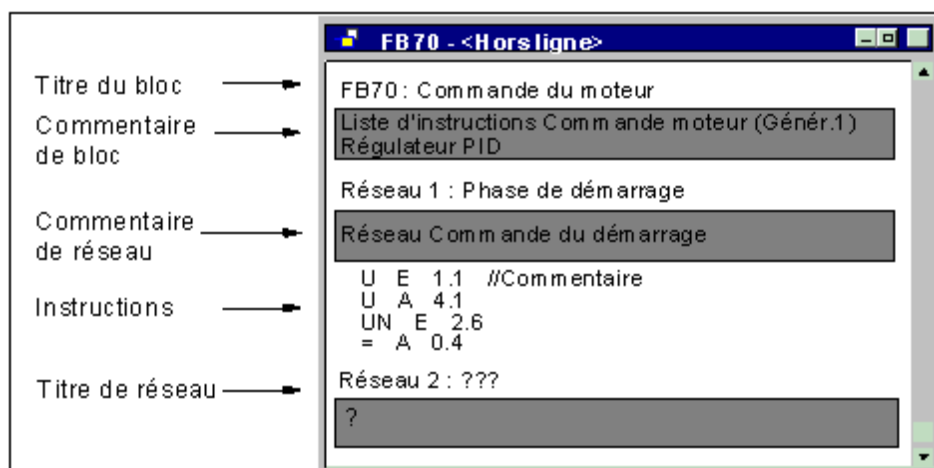
### 10.4.1 Structure de la section des instructions

Vous décrivez l'exécution du programme de votre bloc de code dans la section des instructions. Vous écrivez à cet effet les instructions correspondantes dans des réseaux, en fonction du langage de programmation activé. L'éditeur effectue une vérification de la syntaxe immédiatement après la saisie d'une instruction et représente les éventuelles erreurs en italique et en rouge.

Dans la plupart des cas, la section des instructions d'un bloc de code comporte plusieurs réseaux, eux-mêmes composés d'une liste d'instructions.

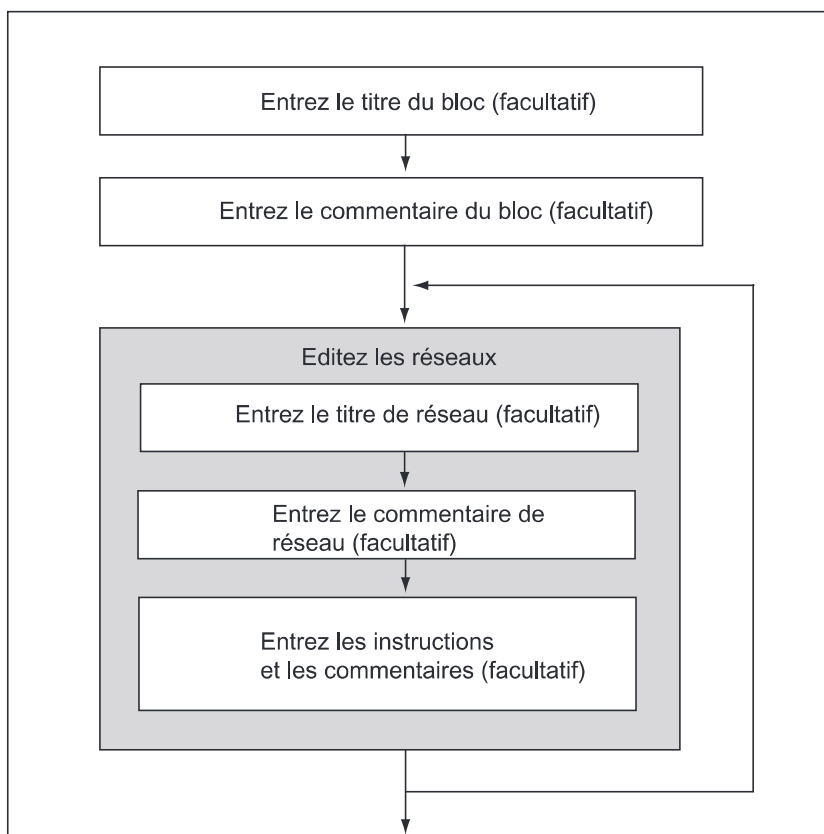
Vous pouvez éditer le titre de bloc, les commentaires de bloc, les titres de réseaux, les commentaires de réseaux ainsi que les lignes d'instructions dans les réseaux d'une section d'instructions.

#### Structure de la section des instructions dans l'exemple du langage de programmation LIST



## 10.4.2 Marche à suivre pour la saisie d'instructions

De manière générale, vous pouvez éditer les composants de la section des instructions dans un ordre quelconque. Lorsque vous programmez un bloc pour la première fois, il est recommandé de procéder dans l'ordre suivant :



Vous pouvez effectuer les modifications dans les modes d'insertion et de substitution. Vous pouvez passer d'un mode à l'autre avec la touche **INSERTION**.

### 10.4.3 Saisie de mnémoniques globaux dans un programme

En choisissant la commande **Insertion > Mnémonique**, vous pouvez insérer des mnémoniques dans la section des instructions de votre programme. Lorsque le curseur est positionné au début, à la fin ou à l'intérieur d'une chaîne de caractères, le mnémonique commençant par cette chaîne de caractères est déjà sélectionné - si un tel mnémonique existe. Si vous modifiez la chaîne de caractères, la sélection est reprise dans la liste.

Les séparateurs de début et de fin d'une chaîne de caractères sont par exemple le caractère d'espacement, le point et le double point. Les séparateurs ne sont pas interprétés dans les mnémoniques globaux.

Pour insérer des mnémoniques, vous pouvez procéder de la manière suivante :

1. Saisissez les caractères de début du mnémonique souhaité dans le programme.
2. Appuyez simultanément sur la touche CTRL et sur la touche J, pour afficher la liste des mnémoniques. Le premier mnémonique commençant par les caractères de début saisis est déjà sélectionné.
3. Validez le mnémonique en appuyant sur la touche ENTREE ou sélectionnez un autre mnémonique.

Le mnémonique entre guillemets remplace alors les caractères de début.

De manière générale : lorsque, lors de l'insertion d'un mnémonique, le curseur se trouve au début, à la fin ou à l'intérieur d'une chaîne de caractères, cette chaîne de caractères est remplacée par le mnémonique entre guillemets.

## 10.4.4 Titres et commentaires de blocs et de réseaux

Les commentaires améliorent la lisibilité de votre programme utilisateur et facilitent ainsi la mise en service et la recherche des erreurs éventuelles. Vous devez absolument y faire appel, car ils constituent une part importante de la documentation du programme.

### Commentaires pour les programmes CONT, LOG et LIST

Vous disposez des commentaires suivants :

- Titre de bloc : titre d'un bloc de 64 caractères au maximum.
- Commentaire de bloc : documentation du bloc de code entier, indique par exemple la fonction du bloc.
- Titre de réseau : titre d'un réseau de 64 caractères au maximum.
- Commentaire de réseau : documentation de la fonction des différents réseaux.
- Colonne de commentaire de la vue de détail des variables : commentaires pour les données locales déclarées.
- Commentaire de mnémonique : commentaires ayant été saisis pour un opérande lors de la définition du nom dans la table des mnémoniques.

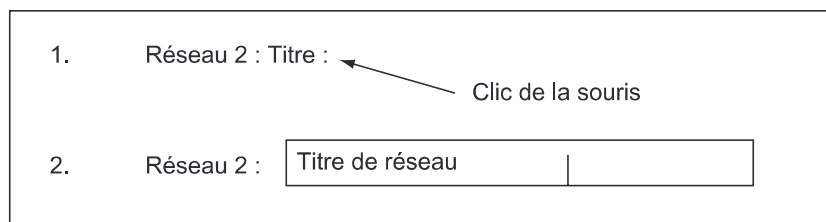
Vous pouvez afficher ces commentaires en choisissant la commande **Affichage > Afficher avec > Informations mnémonique**.

Vous pouvez saisir le titre de bloc, les titres de réseaux ainsi que les commentaires de bloc et les commentaires de réseaux dans la section des instructions d'un bloc de code.

### Titre de bloc ou de réseau

Pour saisir un titre de bloc ou de réseau, positionnez le curseur sur le mot "Titre :" à droite du nom de bloc ou de réseau (par exemple, Réseau 1 : Titre :). Une zone de texte dans laquelle vous pouvez entrer le titre s'ouvre. Ce titre peut comporter 64 caractères au maximum.

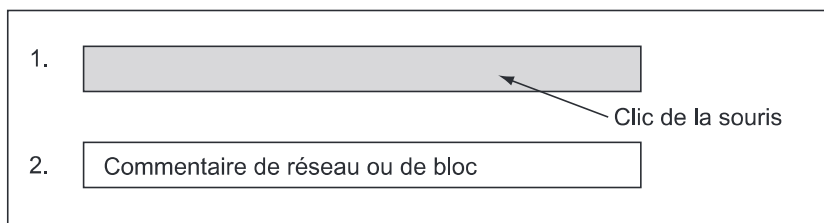
Le commentaire d'un bloc se rapporte au bloc de code entier : vous pouvez en décrire la fonction. Quant aux commentaires de réseaux, ils font référence aux réseaux individuels et décrivent donc les caractéristiques de chacun.



Pour attribuer automatiquement des titres de réseau, choisissez la commande de menu **Outils > Paramètres**, puis dans l'onglet "Général" activez l'option "Attribution automatique de titres de réseau". Le commentaire de mnémonique du premier opérande saisi sera alors repris comme titre de réseau.

## Commentaires de blocs et de réseaux

Vous pouvez activer/désactiver l'affichage des zones de commentaires grises en choisissant la commande **Affichage > Afficher avec > Commentaire**. Un double clic sur une telle zone de commentaires ouvre une zone de texte dans laquelle vous pouvez saisir vos explications. Vous disposez de 64 kilo-octets par bloc pour les commentaires de blocs et de réseaux.



### 10.4.5 Saisie de commentaires de blocs et de réseaux

1. Activez - commande cochée - l'affichage des commentaires via la commande **Affichage > Afficher avec > Commentaire**.
2. Positionnez le curseur par un clic de la souris dans la zone grise sous le nom de bloc ou de réseau. Un cadre délimite maintenant la zone de commentaire qui est passée du gris au blanc.
3. Saisissez le commentaire dans la zone de texte ouverte. Vous disposez de 64 kilo-octets en tout par bloc pour les commentaires de blocs et de réseaux.
4. Quittez la zone de texte par un clic de la souris en dehors de la zone de texte, via la touche TABULATION ou via la combinaison de touches MAJ + TAB.
5. Désactivez éventuellement l'affichage des commentaires via la commande **Affichage > Afficher avec > Commentaire**.

## 10.4.6 Utilisation de modèles de réseau

Si lors de la programmation de blocs, vous souhaitez utiliser certains réseaux plusieurs fois, vous pouvez les enregistrer dans une bibliothèque, sous forme de modèle de réseau contenant éventuellement des emplacements de réserve (p. ex. pour des opérandes). La bibliothèque doit déjà exister avant la création d'un modèle de réseau.

### Création d'un modèle de réseau

Créez, le cas échéant, une nouvelle bibliothèque dans SIMATIC Manager, puis insérez un programme dans cette bibliothèque en choisissant la commande de menu **Insertion > Programme > Programme S7**.

1. Ouvrez le bloc contenant le ou les réseaux qui vont servir de modèle.
2. Dans le bloc ouvert, remplacez éventuellement le titre, le commentaire ou les opérandes par des chaînes de caractères pour réserver de l'emplacement. Vous pouvez utiliser les chaînes de caractères %00 à %99. Celles-ci sont représentées en rouge. Ceci ne pose pas de problème, puisque vous n'enregistrez pas le bloc après création du modèle de réseau.  
Lors de l'insertion d'un modèle de réseau dans un bloc, vous pourrez ultérieurement remplacer les caractères non définis par des opérandes appropriés.
3. Sélectionnez "Réseau <Nr>" pour le ou les réseaux qui doivent figurer dans le modèle.
4. Choisissez la commande **Edition > Créer modèle de réseau**.
5. Dans la boîte de dialogue affichée, saisissez un commentaire évocateur pour les divers caractères non définis utilisés.
6. Fermez la boîte de dialogue en cliquant sur "OK".
7. Dans la boîte de dialogue qui s'ouvre ensuite, sélectionnez le **Dossier Sources** du programme S7 dans votre bibliothèque de modèles de réseau et entrez un nom pour le modèle.
8. Confirmez ensuite votre saisie en cliquant sur le bouton "OK". Le modèle de réseau s'affiche dans la bibliothèque sélectionnée.
9. Fermez le bloc sans l'enregistrer.

### Insertion d'un réseau à partir d'un modèle de réseau

1. Ouvrez le bloc dans lequel vous souhaitez insérer le nouveau réseau.
2. Dans le bloc, cliquez dans le réseau après lequel vous souhaitez insérer le réseau correspondant au modèle de réseau.
3. Ouvrez l'onglet "Eléments de programme" (commande de menu **Insertion > Eléments de programme**).
4. Ouvrez le dossier "Programme S7" de la bibliothèque correspondante.
5. Effectuez un double clic sur le modèle de réseau.
6. Dans la boîte de dialogue affichée, remplacez les caractères non définis dans le modèle de réseau.
7. Cliquez sur le bouton "OK". Le modèle de réseau est inséré à la suite du réseau actuel.

---

#### Nota

Vous pouvez également glisser le modèle de réseau depuis l'onglet dans la fenêtre de l'éditeur par glisser-lâcher.

---

### 10.4.7 Fonction de recherche d'erreurs dans la section des instructions

Puisqu'elles sont représentées en rouge, les erreurs peuvent être localisées aisément dans la section des instructions. Pour passer facilement aux erreurs se trouvant hors de la zone visible, l'éditeur propose les deux fonctions de recherche **Edition > Aller à > Erreur précédente/suivante**.

Cette recherche se fait par-delà les réseaux, c'est-à-dire dans toute la section des instructions et pas uniquement dans le réseau en cours ou dans la zone actuellement visible.

Lorsque vous activez la barre d'état en choisissant la commande **Affichage > Barre d'état**, des informations sur les erreurs s'y affichent.

Vous avez également la possibilité de corriger des erreurs et d'effectuer des modifications en mode de substitution. Vous passez du mode d'insertion au mode de substitution et inversement avec la touche INSERTION.

## 10.5 Edition d'instructions CONT dans la section des instructions

### 10.5.1 Paramètres pour le langage de programmation CONT

#### Paramétrage de la mise en page pour CONT

Vous pouvez définir la mise en page pour la programmation en langage CONT. Le format choisi (format vertical DIN A4, format horizontal DIN A4, taille maximale) a une influence sur le nombre d'éléments CONT représentables dans une branche.

1. Choisissez la commande **Outils > Paramètres**.
2. Choisissez l'onglet "CONT/LOG" dans la boîte de dialogue qui s'affiche alors.
3. Choisissez le format désiré dans la zone de liste "Mise en page". Indiquez la taille de format désirée.

#### Paramètres pour l'impression :

Si vous désirez imprimer la section des instructions CONT, nous vous conseillons de définir le format de page approprié avant même d'écrire la section des instructions.

#### Paramètres dans la page d'onglet "CONT/LOG"

Dans la page d'onglet "CONT/LOG", à laquelle vous accédez par la commande **Outils > Paramètres**, vous pouvez définir les paramètres généraux comme par exemple le format ou la largeur du champ de l'opérande.

### 10.5.2 Règles pour la saisie d'instructions CONT

La description du langage "CONT" figure dans le manuel "CONT pour S7-300/400 - Programmation de blocs" ou dans l'aide en ligne sur CONT.

Un réseau CONT peut être composé de plusieurs éléments dans plusieurs branches. Tous les éléments et branches doivent être reliés entre eux, la barre conductrice gauche n'étant pas considérée comme une liaison (CEI 1131-3).

Vous devez observer quelques règles lors de la programmation en CONT. Des messages vous signaleront d'éventuelles erreurs.

#### Terminaison d'un réseau CONT

Tout réseau CONT doit posséder une terminaison sous forme de bobine ou de boîte. Vous ne pouvez cependant pas utiliser les éléments CONT suivants comme terminaison de réseau :

- boîtes de comparaison
- bobines pour connecteurs `_(#)_/`
- bobines pour le traitement de fronts positifs `_(P)_/` ou négatifs `_(N)_/`



## Placement de boîtes

Le point de départ de la branche pour la connexion d'une boîte doit toujours être la barre conductrice gauche. La branche précédant la boîte peut cependant contenir des fonctions logiques ou d'autres boîtes.

## Placement de bobines

Les bobines sont automatiquement placées à l'extrémité droite du réseau, où elles forment la terminaison d'une branche.

Exceptions : les bobines pour connecteurs  $_{/(\#)_/}$  et les traitements de front positifs  $_{/(P)_/}$  ou négatifs  $_{/(N)_/}$  ne peuvent être placés ni complètement à gauche, ni complètement à droite dans la branche. Ils ne sont pas non plus autorisés dans les branches parallèles.

Il existe des bobines pour lesquelles une fonction logique booléenne est requise et d'autres qui l'interdisent.

- Bobines nécessitant une fonction logique :
  - sortie  $_{/(\ )_/}$ , mise à 1 sortie  $_{/(S)_/}$ , remise à 0 sortie  $_{/(R)_/}$
  - connecteur  $_{/(\#)_/}$ , front positif  $_{/(P)_/}$ , front négatif  $_{/(N)_/}$
  - toutes les bobines pour compteurs et temporisations
  - sauts si 0  $_{/(JMPN)_/}$
  - activation relais de masquage  $_{/(MCR<)_/}$
  - chargement du résultat logique dans le registre RB  $_{/(SAVE)_/}$
  - retour saut  $_{/(RET)_/}$
- Bobines interdisant une fonction logique :
  - début de relais de masquage  $_{/(MCRA)_/}$
  - fin de relais de masquage  $_{/(MCRD)_/}$
  - ouverture bloc de données  $_{/(OPN)_/}$
  - désactivation relais de masquage  $_{/(MCR>)_/}$

Pour toutes les autres bobines, les fonctions logiques sont autorisées sans être requises.

Vous **ne devez pas** utiliser les bobines suivantes comme sortie parallèle :

- sauts si 0  $_{/(JMPN)_/}$
- sauts si 1  $_{/(JMP)_/}$
- appel de bloc  $_{/(CALL)_/}$
- retour  $_{/(RET)_/}$

## Entrée de validation / sortie de validation

La connexion de l'entrée de validation "EN" ou de la sortie de validation "ENO" de boîtes est possible mais pas requise.

## Suppression et modification

Lorsqu'une branche n'est composée que d'un élément, la suppression de cet élément supprime la branche entière.

La suppression d'une boîte entraîne également la suppression de toutes les branches reliées avec les entrées booléennes de cette boîte, à l'exception de la branche principale.

Pour le simple échange d'éléments du même type, vous disposez du mode de substitution.

## Branches parallèles

- Insérez les branches OU de la gauche vers la droite.
- Les branches parallèles s'ouvrent vers le bas et se ferment vers le haut.
- Les branches parallèles s'ouvrent toujours après l'élément CONT sélectionné.
- Les branches parallèles se ferment toujours après l'élément CONT sélectionné.
- Pour effacer une branche parallèle, vous devez effacer tous les éléments CONT qu'elle contient. La suppression du dernier élément CONT de la branche entraîne également l'effacement du reste de celle-ci.

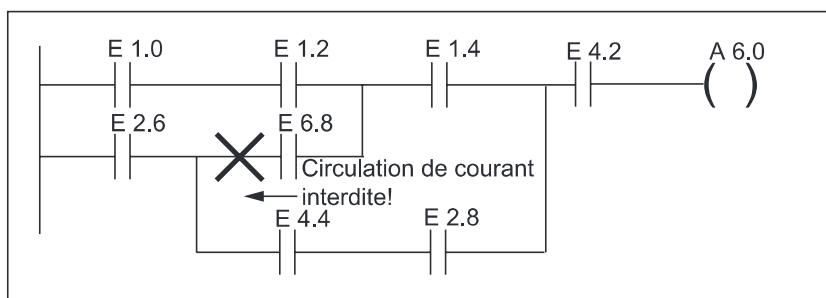
## Constantes

Il n'est pas possible d'affecter des constantes (p. ex. TRUE ou FALSE) à des combinaisons binaires. Utilisez à cet effet des opérandes de type BOOL.

### 10.5.3 Branchements interdits en CONT

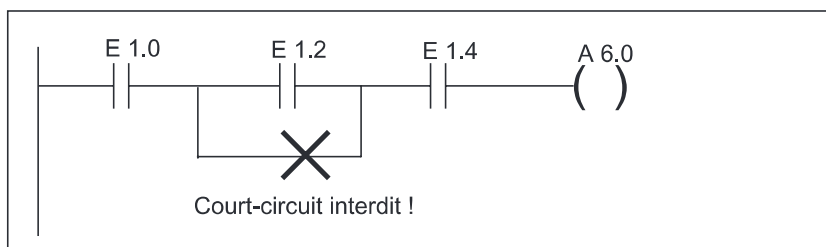
#### Circulation du courant de la droite vers la gauche

Vous ne pouvez pas éditer de branches qui provoqueraient une circulation inverse du courant. La figure suivante en montre un exemple. Si E 1.4 a l'état de signal "0", E 6.8 entraînerait une circulation du courant de la droite vers la gauche. Ceci est interdit.



#### Court-circuit

Vous ne pouvez pas éditer de branches qui entraînent un court-circuit. La figure suivante en montre un exemple :



## 10.6 Edition d'instructions LOG dans la section des instructions

### 10.6.1 Paramètres pour le langage de programmation LOG

#### Définition de la mise en page pour LOG

Vous pouvez définir la mise en page pour la programmation en langage LOG. Le format choisi (format vertical DIN A4, format horizontal DIN A4, taille maximale) a une influence sur le nombre d'éléments LOG représentables dans une branche.

1. Choisissez la commande **Outils > Paramètres**.
2. Choisissez l'onglet "CONT/LOG" dans la boîte de dialogue qui s'affiche alors.
3. Choisissez le format désiré dans la zone "Mise en page". Indiquez la taille de format désirée.

#### Remarques pour l'impression

Si vous désirez imprimer la section des instructions LOG, nous vous conseillons de définir le format de page approprié avant même d'écrire la section des instructions.

#### Onglet CONT/LOG sous Outils > Paramètres

Dans la page d'onglet "CONT/LOG", à laquelle vous parvenez via la commande **Outils > Paramètres**, vous pouvez définir les paramètres généraux comme par exemple le format ou la largeur du champ de l'opérande.

## 10.6.2 Règles pour la saisie d'instructions LOG

La description du langage "LOG" figure dans le manuel "LOG pour S7-300/400 - Programmation de blocs" ou dans l'aide en ligne sur LOG.

Un réseau LOG peut être composé de plusieurs éléments. Tous les éléments doivent être reliés entre eux (CEI 1131-3).

Vous devez observer quelques règles lors de la programmation en LOG. Des messages vous signaleront d'éventuelles erreurs.

### Insertion et édition d'adresses et de paramètres

Lors de l'insertion d'un élément LOG, les chaînes de caractères "???" et "..." sont utilisées pour réserver l'emplacement des adresses ou des paramètres.

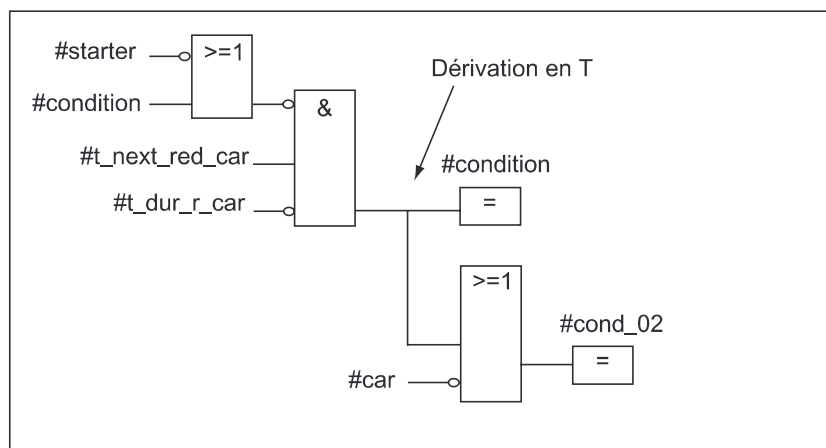
- La chaîne de caractères "???" en rouge signale les adresses et paramètres devant être définis.
- La chaîne de caractères "..." en noir signale les adresses et paramètres pouvant être définis.

Le type de données escompté s'affiche lorsque vous placez le pointeur de la souris sur les adresses ou paramètres non définis.

### Placement de boîtes

Aux boîtes avec des fonctions logiques binaires (&, >=1, XOR), vous pouvez accoler des boîtes standard (bascules, compteurs, temporisations, opérations de calcul, etc.). Les boîtes de comparaison sont exclues de cette règle.

Dans un réseau, vous n'êtes pas autorisé à programmer des fonctions logiques séparées par des sorties distinctes. A l'aide de la branche, vous pouvez cependant attribuer plusieurs affectations à une séquence de boîtes logiques. La figure suivante représente un réseau contenant deux affectations.



Les boîtes suivantes ne doivent pas être placées à l'extrémité droite de la séquence logique, où elles forment la terminaison de la séquence :

- Réinitialiser le compteur
- Comptage, décomptage
- Activer la temporisation "Impulsion" / "Impulsion prolongée"
- Activer la temporisation "Retard à la montée" / "retard à la retombée".

Il existe des boîtes pour lesquelles une fonction logique booléenne est requise et d'autres qui l'interdisent.

### **Boîtes nécessitant une fonction logique :**

- sortie, mise à 1 sortie, remise à 0 sortie `_[R]`
- connecteur `_[#]_`, front positif `_[P]_`, front négatif `_[N]_`
- toutes les boîtes pour compteurs et temporisations
- sauts si 0 `_[JMPN]`
- activation relais de masquage `_[MCR<]`
- chargement du résultat logique dans le registre RB `_[SAVE]`
- retour saut `_[RET]`

### **Boîtes interdisant une fonction logique :**

- début de relais de masquage `[MCRA]`
- fin de relais de masquage `[MCRD]`
- ouverture bloc de données `[OPN]`
- désactivation relais de masquage `[MCR>]`

Pour toutes les autres boîtes, les fonctions logiques sont autorisées sans être requises.

### **Entrée de validation/ sortie de validation**

La connexion de l'entrée de validation "EN" ou de la sortie de validation "ENO" de boîtes est possible mais pas requise.

### **Suppression et modification**

La suppression d'une boîte entraîne également la suppression de toutes les branches reliées avec les entrées booléennes de cette boîte, à l'exception de la branche principale.

Pour le simple échange d'éléments du même type, vous disposez du mode de substitution.

### **Constantes**

Il n'est pas possible d'affecter des constantes (p. ex. TRUE ou FALSE) à des combinaisons binaires. Utilisez à cet effet des opérands de type BOOL.

## 10.7 Edition d'instructions LIST dans la section des instructions

### 10.7.1 Paramètres pour le langage de programmation LIST

#### Sélection des abréviations

Vous pouvez choisir parmi deux types d'abréviations :

- allemandes ou
- anglaises

Avant d'ouvrir un bloc, vous sélectionnez les abréviations dans SIMATIC Manager, en choisissant la commande **Outils > Paramètres**, puis l'onglet "Langage". Vous ne pouvez pas modifier les abréviations durant l'édition d'un bloc.

Vous éditez les **propriétés de bloc** dans une boîte de dialogue distincte.

Vous pouvez ouvrir plusieurs blocs dans l'éditeur et les éditer les uns après les autres.

### 10.7.2 Règles pour la saisie d'instructions LIST

La description du langage "LIST" figure dans le manuel "LIST pour S7-300/400 - Programmation de blocs" ou dans l'aide en ligne sur LIST (Description des langages).

Vous devez respecter les règles suivantes lors de la saisie d'instructions LIST en mode de saisie incrémentale :

- Respectez l'ordre de programmation des blocs : il faut programmer les blocs appelés avant les blocs appelants.
- Une instruction se compose d'un repère de saut (facultatif), d'une opération, d'un opérande et d'un commentaire (facultatif).  
Exemple M001: U E1.0 //commentaire
- Une ligne contient une seule instruction.
- Un bloc peut contenir 999 réseaux au maximum.
- Un réseau peut contenir environ 2000 lignes. Le nombre de lignes possibles varie selon que l'affichage est agrandi ou réduit.
- Vous pouvez saisir les opérations et les adresses absolues indifféremment en majuscules ou en minuscules.

## 10.8 Actualisation d'appels de blocs

En choisissant la commande **Edition > Appel > Actualiser** dans "CONT/LOG/LIST : Programmation de blocs S7", vous pouvez mettre à jour automatiquement des appels de bloc devenus invalides. Après les modifications d'interfaces suivantes, vous devez effectuer une actualisation :

- insertion de nouveaux paramètres formels,
- suppression de paramètres formels,
- changement du nom de paramètres formels,
- modification du type de paramètres formels,
- changement de l'ordre (recopie) de paramètres formels.

L'affectation entre paramètres formels et paramètres effectifs est réalisée d'après les règles suivantes, dans l'ordre spécifié :

1. **Noms de paramètre identiques :**

Les paramètres effectifs sont affectés automatiquement lorsque le nom du paramètre formel est resté identique.

Cas particulier : dans CONT et LOG, la fonction logique précédente de paramètres d'entrée binaires ne peut être affectée automatiquement que si le type de données est identique (BOOL). Si dans un tel cas, le type de données a été modifié, la fonction logique précédente reste conservée sous forme de branche ouverte.

2. **Types de données de paramètre identiques :**

Une fois que les paramètres de nom identique ont été affectés, les paramètres effectifs non encore affectés le seront à des paramètres formels de même type de données que "l'ancien" paramètre formel.

3. **Position de paramètre identique :**

Les paramètres effectifs non encore affectés d'après les règles 1 et 2, seront à présent affectés aux nouveaux paramètres formels, d'après leur position dans "l'ancienne" interface.

4. Si des paramètres effectifs ne peuvent pas être affectés d'après les trois règles précitées, ils seront supprimés ou, dans le cas de fonctions logiques précédentes dans CONT ou LOG, resteront conservés sous forme de branches ouvertes.

Après avoir exécuté cette fonction, vérifiez les modifications réalisées dans la table de déclaration des variables ainsi que dans la section des instructions du programme.



### 10.8.1 Modifications d'interfaces

Pour les blocs hors ligne édités dans la version 5 de STEP 7, vous pouvez également effectuer les modifications d'interface dans l'éditeur incrémental.

1. Assurez-vous que tous les blocs ont été compilés avec la version 5 de STEP 7. Générez à cet effet une source de tous les blocs, puis compilez cette dernière.
2. Modifiez l'interface du bloc concerné.
3. Ouvrez à présent successivement tous les appels - les appels correspondants sont représentés en rouge.
4. Utilisez la commande **Edition > Appel > Actualiser**.
5. Générez une nouvelle fois les blocs de données d'instance concernés.

---

#### Nota

- Une modification d'interface dans un bloc ouvert en ligne peut conduire à l'arrêt de la CPU.
  - Réassignation d'appels de blocs  
Modifiez d'abord le numéro des blocs appelés, puis exécutez la fonction de réassignation pour adapter les appels.
-

## 10.9 Enregistrement de blocs de code

Pour intégrer les blocs nouvellement créés ou bien les modifications apportées à la section des instructions de blocs de code ou aux tables de déclaration au système de gestion de données de la PG, vous devez sauvegarder les blocs correspondants. Ainsi, les données sont écrites sur le disque dur de la console de programmation.

### Enregistrement de blocs sur le disque dur de la PG

1. Activez la fenêtre de travail du bloc à enregistrer.
2. Choisissez :
  - la commande **Fichier > Enregistrer** si vous voulez sauvegarder le bloc sous le même nom ;
  - la commande **Fichier > Enregistrer sous** si vous voulez sauvegarder le bloc sous un autre programme utilisateur S7 ou sous un autre nom. Indiquez, dans la boîte de dialogue qui apparaît alors, le nouveau chemin d'accès ou le nouveau bloc.

Dans les deux cas, le bloc n'est enregistré que si sa syntaxe est correcte. Les erreurs de syntaxe éventuelles sont immédiatement reconnues lors de la création et sont affichées en rouge. Vous devez les corriger avant l'enregistrement du bloc.

---

#### Nota

- Vous pouvez enregistrer des blocs ou des sources sous d'autres projets ou bibliothèques également dans SIMATIC Manager (par exemple, par glisser-lâcher).
  - L'enregistrement de blocs ou de programmes utilisateur complets sur une carte mémoire n'est possible que dans SIMATIC Manager.
  - En cas de problème lors de l'enregistrement ou de la compilation de blocs volumineux, vous devriez réorganiser le projet. Choisissez à cet effet la commande **Fichier > Réorganiser** dans SIMATIC Manager. Faites ensuite une nouvelle tentative d'enregistrement ou de compilation.
-

# 11 Création de blocs de données

## 11.1 Principes de la création des blocs de données

Dans un bloc de données, vous stockez, par exemple, les valeurs auxquelles votre machine ou installation accède. Contrairement au bloc de code que vous avez programmé dans l'un des langages de programmation CONT/LOG ou LIST, le bloc de données ne comporte que la section de déclaration des variables. Il ne possède pas de section d'instructions, et ne nécessite donc pas de programmation de réseaux.

Après avoir ouvert un bloc de données, vous pouvez l'afficher dans la vue des déclarations ou dans la vue des données. Vous passez d'une vue à l'autre en choisissant les commandes **Affichage > Vue des déclarations** et **Affichage > Vue des données**.

### Vue des déclarations

Choisissez la vue des déclarations pour

- lire ou déterminer la structure de données des DB globaux,
- lire la structure de données des DB associés à un type de données utilisateur (UDT),
- lire la structure de données des DB associés à un bloc fonctionnel.

Vous ne pouvez pas modifier la structure des blocs de données associés à un FB ou à un type de données utilisateur. Il vous faudrait d'abord modifier le FB ou l'UDT correspondant, puis créer un nouveau DB.

### Vue des données

Choisissez la vue des données pour modifier des données. Seule la vue des données vous permet d'afficher, de saisir ou de modifier la valeur en cours de chaque élément. Dans la vue des données des blocs de données, les éléments des variables avec types de données complexes sont énumérés chacun avec leur nom complet.

### Différence entre un bloc de données d'instance et un bloc de données global

Le bloc de données global n'est pas affecté à un bloc de code. Il contient des valeurs qui sont requises par les installations ou machines et peut être appelé directement à un endroit quelconque du programme.

Le bloc de données d'instance est quant à lui directement affecté à un bloc de code, par exemple à un bloc fonctionnel. Le bloc de données d'instance contient les données ayant été stockées dans la table de déclaration des variables d'un bloc fonctionnel.

## 11.2 Vue des déclarations de blocs de données

Vous ne pouvez pas modifier la vue des déclarations de blocs de données non globaux.

Colonne	Signification
Adresse	Ici s'affiche l'adresse que STEP 7 affecte automatiquement à la variable lorsque vous achevez la saisie d'une déclaration.
Déclaration	Cette colonne n'existe que pour les blocs de données d'instance. Elle indique comment les variables ont été déclarées dans la déclaration des variables du FB : <ul style="list-style-type: none"><li>• Paramètres d'entrée (IN)</li><li>• Paramètres de sortie (OUT)</li><li>• Paramètres d'entrée/sortie (IN_OUT)</li><li>• Données statiques (STAT)</li></ul>
Nom	Indiquez ici le nom que vous devez affecter à chaque variable.
Type	Indiquez ici le type de données pour la variable (BOOL, INT, WORD, ARRAY, etc.). Les variables peuvent avoir un type de données simple, un type de données complexe ou un type de données utilisateur.
Valeur initiale	Vous pouvez préciser ici une valeur initiale si le logiciel ne doit pas prendre la valeur par défaut pour le type de données indiqué. Toutes les valeurs saisies doivent être compatibles avec les types de données.  Cette valeur initiale est prise comme valeur en cours de la variable lors de la première sauvegarde du bloc de données si vous n'affectez pas explicitement une valeur en cours à cette variable.  Nota : les valeurs initiales ne peuvent pas être chargées dans la CPU !
Commentaire	Vous pouvez saisir dans cette zone un commentaire pour la documentation de la variable. La longueur du commentaire ne doit pas dépasser 79 caractères.

## 11.3 Vue des données de blocs de données

La vue des données affiche les valeurs en cours de toutes les variables du bloc de données. Vous ne pouvez modifier ces valeurs que dans cette vue. La représentation sous forme de tableau est identique pour tous les blocs de données globaux. Elle contient une colonne "Déclaration" supplémentaire pour les blocs de données d'instance.

Dans la vue des données, les éléments des variables avec types de données complexes ou types de données utilisateur sont chacun listés dans leur propre ligne avec leur nom complet. Si ces éléments se situent dans la zone IN\_OUT d'un bloc de données d'instance, le pointeur apparaît sur le type de données complexe ou utilisateur dans la colonne "Valeur en cours".

La vue des données représente les colonnes suivantes :

Colonne	Signification
Adresse	Cette colonne affiche l'adresse que STEP 7 affecte automatiquement à la variable.
Déclaration	Cette colonne n'existe que pour les DB d'instance. Elle indique comment les variables ont été déclarées dans la déclaration des variables du FB : <ul style="list-style-type: none"> <li>• Paramètres d'entrée (IN)</li> <li>• Paramètres de sortie (OUT)</li> <li>• Paramètres d'entrée/sortie (IN_OUT)</li> <li>• Données statiques (STAT)</li> </ul>
Nom	Il s'agit ici du nom affecté à la variable. Vous ne pouvez pas éditer cette zone dans la vue des données.
Type	Il s'agit ici du type de données affecté à la variable. Puisque, dans la vue des données, les éléments sont énumérés individuellement pour les variables avec type de données complexe ou utilisateur, il n'y a plus ici que des types de données simples pour un bloc de données global. Pour un bloc de données d'instance, cette colonne contient également les types de paramètre. Un pointeur désigne le type de données pour les paramètres d'entrée/sortie (IN_OUT) avec type de données complexe ou utilisateur dans la colonne "Valeur en cours".
Valeur initiale	Il s'agit ici de la valeur initiale que vous avez définie pour la variable si le logiciel ne doit pas utiliser la valeur par défaut pour le type de données indiqué. La valeur initiale est prise comme valeur en cours de la variable lors de la première sauvegarde du bloc de données si vous n'affectez pas explicitement une valeur en cours à cette variable. Nota : contrairement aux valeurs en cours, les valeurs initiales ne peuvent pas être chargées dans la CPU !
Valeur en cours	Hors ligne : il s'agit de la valeur de la variable à l'ouverture du bloc de données ou après sa dernière modification enregistrée (même si le DB est ouvert en ligne, la mise à jour de cet affichage n'a pas lieu !). En ligne : il s'agit de la valeur en cours à l'ouverture du bloc de données. Elle n'est pas mise à jour automatiquement ; vous devez actualiser l'affichage avec la touche F5. Vous pouvez éditer cette zone si elle ne correspond pas à un paramètre d'entrée/sortie (IN_OUT) avec type de données complexe ou utilisateur. Toutes les valeurs saisies doivent être compatibles avec les types de données. Nota : seules les valeurs en cours peuvent être chargées dans la CPU !
Commentaire	Il s'agit ici du commentaire défini dans la déclaration des variables pour la documentation de la variable. Vous ne pouvez pas éditer cette zone dans la vue des données.

## 11.4 Saisie et enregistrement des blocs de données

### 11.4.1 Saisie de la structure de données de blocs de données globaux

Si vous avez ouvert un bloc de données qui n'est associé ni à un UDT ni à un FB, vous pouvez définir sa structure dans la vue des déclarations. Vous ne pouvez pas modifier la vue des déclarations de blocs de données non globaux.

1. Ouvrez un bloc de données global, c'est-à-dire un bloc qui n'est associé ni à un UDT, ni à un FB.
2. Affichez la vue des déclarations du bloc de données, si ce n'est déjà le cas.
3. Définissez sa structure en complétant la table affichée avec les données suivantes.

Vous ne pouvez pas modifier la vue des déclarations de blocs de données non globaux.

Colonne	Signification
Adresse	Ici s'affiche l'adresse que STEP 7 affecte automatiquement à la variable lorsque vous achevez la saisie d'une déclaration.
Nom	Indiquez ici le nom que vous devez affecter à chaque variable.
Type	Indiquez ici le type de données pour la variable (BOOL, INT, WORD, ARRAY, etc.). Les variables peuvent avoir un type de données simple, un type de données complexe ou un type de données utilisateur.
Valeur initiale	Vous pouvez préciser ici une valeur initiale si le logiciel ne doit pas prendre la valeur par défaut pour le type de données indiqué. Toutes les valeurs saisies doivent être compatibles avec les types de données.  Cette valeur initiale est prise comme valeur en cours de la variable lors de la première sauvegarde du bloc de données si vous n'affectez pas explicitement une valeur en cours à cette variable.
Commentaire	Vous pouvez saisir dans cette zone un commentaire pour la documentation de la variable. La longueur du commentaire ne doit pas dépasser 80 caractères.

## 11.4.2 Saisie / affichage de la structure de données de blocs de données associés à un FB (DB d'instance)

### Saisie

Lorsque vous associez un bloc de données à un bloc fonctionnel (DB d'instance), la déclaration des variables du FB détermine la structure du bloc de données. Les modifications ne sont possibles que dans le bloc fonctionnel associé.

1. Ouvrez le bloc fonctionnel associé.
2. Editez la déclaration des variables du bloc fonctionnel.
3. Recréez ensuite le bloc de données d'instance.

### Affichage

Vous pouvez afficher dans la vue des déclarations du DB d'instance la manière dont les variables ont été déclarées dans le FB.

1. Ouvrez le bloc de données.
2. Affichez la vue des déclarations du bloc de données, si ce n'est pas déjà le cas.
3. Les explications relatives à la table affichée figurent ci-après.

Vous ne pouvez pas modifier la vue des déclarations de blocs de données non globaux.

Colonne	Signification
Adresse	Cette colonne affiche l'adresse que STEP 7 affecte automatiquement à la variable.
Déclaration	Cette colonne vous indique comment les variables ont été déclarées dans le bloc fonctionnel : <ul style="list-style-type: none"> <li>• Paramètres d'entrée (IN)</li> <li>• Paramètres de sortie (OUT)</li> <li>• Paramètres d'entrée/sortie (IN_OUT)</li> <li>• Données statiques (STAT)</li> </ul> Les données temporaires déclarées du FB ne sont pas dans le bloc de données d'instance.
Nom	Il s'agit ici du nom affecté à la variable dans la déclaration des variables du FB.
Type	Il s'agit ici du type de données affecté à la variable dans la déclaration des variables du FB. Les variables peuvent avoir un type de données simple, un type de données complexe ou un type de données utilisateur. Si d'autres blocs fonctionnels pour l'appel desquels des variables statiques ont été déclarées sont appelés à l'intérieur du FB, il est possible d'indiquer ici un FB ou un bloc fonctionnel système (SFB) comme type de données.
Valeur initiale	Il s'agit ici de la valeur initiale que vous avez définie pour la variable dans la déclaration des variables du FB si le logiciel ne doit pas utiliser la valeur par défaut du type de données. La valeur initiale est prise comme valeur en cours de la variable lors de la première sauvegarde du bloc de données si vous n'affectez pas explicitement une valeur en cours à cette variable.
Commentaire	Il s'agit ici du commentaire défini dans la déclaration des variables du FB pour la documentation de l'élément de données. Vous ne pouvez pas éditer cette zone.

### Nota

Dans les blocs de données associés à un FB, vous ne pouvez éditer que les valeurs en cours des variables. La saisie de ces valeurs en cours se fait dans la vue des données des blocs de données.

### 11.4.3 Saisie de la structure de types de données utilisateur (UDT)

1. Ouvrez le type de données utilisateur (UDT).
2. Affichez la vue des déclarations, si ce n'est déjà le cas.
3. Définissez la structure du type de données utilisateur en déterminant l'ordre des variables, leur type de données et, éventuellement, leur valeur initiale en fonction des données ci-après.
4. Vous achevez la saisie d'une variable en quittant la ligne avec la touche TAB ou ENTREE.

Colonne	Signification
Adresse	Ici s'affiche l'adresse que STEP 7 affecte automatiquement à la variable lorsque vous achevez la saisie d'une déclaration.
Nom	Indiquez ici le nom que vous devez affecter à chaque variable.
Type	Indiquez ici le type de données pour la variable (BOOL, INT, WORD, ARRAY, etc.). Les variables peuvent avoir un type de données simple, un type de données complexe ou un type de données utilisateur.
Valeur initiale	Vous pouvez préciser ici une valeur initiale si le logiciel ne doit pas prendre la valeur par défaut pour le type de données indiqué. Toutes les valeurs saisies doivent être compatibles avec les types de données.  Lorsque vous sauvegardez pour la première fois une instance de l'UDT (variable ou bloc de données), la valeur initiale est prise comme valeur en cours pour la variable si vous n'indiquez pas explicitement une autre valeur en cours.
Commentaire	Vous pouvez saisir dans cette zone un commentaire pour la documentation de la variable. La longueur du commentaire ne doit pas dépasser 80 caractères.



## 11.4.4 Saisie / affichage de la structure de blocs de données associés à un UDT

### Saisie

Lorsque vous associez un bloc de données à un type de données utilisateur (UDT), la structure des données de l'UDT détermine celle du bloc de données. Les modifications ne sont possibles que dans le type de données utilisateur associé.

1. Ouvrez le type de données utilisateur.
2. Editez la structure du type de données utilisateur.
3. Recréez le bloc de données.

### Affichage

Vous pouvez uniquement afficher dans la vue des déclarations du DB la manière dont les variables ont été déclarées dans l'UDT.

1. Ouvrez le bloc de données.
2. Affichez la vue des déclarations du bloc de données, si ce n'est déjà le cas.
3. Les explications relatives à la table affichée figurent ci-après.

Vous ne pouvez pas modifier la vue des déclarations. Les modifications ne sont possibles que dans le type de données utilisateur associé.

Colonne	Signification
Adresse	Cette colonne affiche l'adresse que STEP 7 affecte automatiquement à la variable.
Nom	Il s'agit ici du nom affecté à la variable dans l'UDT.
Type	Il s'agit ici du type de données affecté à la variable dans l'UDT. Les variables peuvent avoir un type de données simple, un type de données complexe ou un type de données utilisateur.
Valeur initiale	Il s'agit ici de la valeur initiale que vous avez définie pour la variable dans l'UDT si le logiciel ne doit pas utiliser la valeur par défaut du type de données. Cette valeur initiale est prise comme valeur en cours de la variable lors de la première sauvegarde du bloc de données si vous n'affectez pas explicitement une valeur en cours à cette variable.
Commentaire	Il s'agit ici du commentaire défini dans l'UDT pour la documentation de l'élément de données.

### Nota

Dans les blocs de données associés à un UDT, vous ne pouvez éditer que les valeurs en cours des variables. La saisie de ces valeurs en cours se fait dans la vue des données des blocs de données.

### 11.4.5 Modification de valeurs dans la vue des données

L'édition de valeurs en cours n'est possible que dans la vue des données de blocs de données.

1. Activez, si nécessaire, la vue des données de la table à l'aide de la commande **Affichage > Vue des données**.
2. Saisissez les valeurs en cours désirées pour les éléments de données dans la colonne "Valeur en cours". Ces valeurs doivent être compatibles avec le type de données des éléments ;

Les saisies erronées - par exemple, une valeur saisie incompatible avec le type de données - sont reconnues immédiatement lors de l'édition et affichées en rouge. Vous devez les corriger avant de sauvegarder le bloc de données.

---

#### Nota

Les modifications des valeurs ne sont sauvegardées que lors de l'enregistrement des blocs de données.

---

### 11.4.6 Réinitialisation de valeurs en leur substituant leur valeur initiale

La réinitialisation est uniquement possible dans la vue des données de blocs de données.

1. Activez, si nécessaire, la vue des données de la table à l'aide de la commande **Affichage > Vue des données**.
2. Choisissez pour ce faire la commande **Edition > Réinitialiser bloc de données**.

Toutes les variables sont réinitialisées, c'est-à-dire aux valeurs en cours de toutes les variables est substituée la valeur initiale correspondante.

---

#### Nota

Les modifications des valeurs ne sont sauvegardées que lors de l'enregistrement des blocs de données.

---

### 11.4.7 Enregistrement de blocs de données

Pour intégrer les blocs de données nouvellement créés ou les modifications apportées aux valeurs dans les blocs de données au système de gestion de données de la PG, vous devez sauvegarder les blocs correspondants. Ainsi, les données sont écrites sur le disque dur de la console de programmation.

#### Enregistrement de blocs sur le disque dur de la PG

1. Activez la fenêtre de travail du bloc à enregistrer.
2. Choisissez :
  - la commande **Fichier > Enregistrer** si vous voulez sauvegarder le bloc sous le même nom ;
  - la commande **Fichier > Enregistrer sous** si vous voulez sauvegarder le bloc sous un autre programme utilisateur S7 ou sous un autre nom. Indiquez, dans la boîte de dialogue qui apparaît alors, le nouveau chemin d'accès ou le nouveau bloc. Il est interdit de spécifier DB0 qui est réservé au système.

Dans les deux cas, le bloc n'est enregistré que si sa syntaxe est correcte. Les erreurs de syntaxe éventuelles sont immédiatement reconnues lors de la création et sont affichées en rouge. Vous devez les corriger avant l'enregistrement du bloc.

---

#### Nota

- Vous pouvez enregistrer des blocs ou des sources sous d'autres projets ou bibliothèques également dans SIMATIC Manager (par exemple, par glisser-lâcher).
  - L'enregistrement de blocs ou de programmes utilisateur complets sur une carte mémoire n'est possible que dans SIMATIC Manager.
  - En cas de problème lors de l'enregistrement ou de la compilation de blocs volumineux, vous devriez réorganiser le projet. Choisissez à cet effet la commande **Fichier > Réorganiser** dans SIMATIC Manager. Faites ensuite une nouvelle tentative d'enregistrement ou de compilation.
-



## 12 Paramétrage de blocs de données

La fonction "Paramétrage de blocs de données" vous permet

- d'éditer les valeurs actuelles de blocs de données d'instance et de les charger dans le système cible sans devoir charger le blocs de données complet,
- de visualiser en ligne les blocs de données d'instance,
- de paramétrer et de visualiser en ligne dans la vue des paramètres, les blocs de données d'instance et les multiinstances possédant l'attribut système S7\_techparam (fonctions technologiques),

en dehors de l'éditeur de programmes CONT/LIST/LOG.

### Procédez de la manière suivante :

1. Ouvrez le bloc de données d'instance par double clic dans SIMATIC Manager.
2. Répondez par "Oui" à la demande d'ouverture de l'application "Paramétrage de blocs de données".  
**Résultat** : le DB d'instance s'ouvre dans l'application "Paramétrage de blocs de données".
3. En choisissant l'une des commandes de menu **Affichage > Vue des données** ou **Affichage > Vue des déclarations**, précisez la vue dans laquelle vous souhaitez afficher le bloc de données.  
La vue de paramétrage technologique s'ouvre automatiquement pour les blocs de données d'instance ou les multiinstances possédant l'attribut système S7\_techparam.
4. Editez ensuite le blocs de données d'instance. D'éventuels informations, avertissements et erreurs s'affichent dans la fenêtre des messages. En effectuant un double clic sur un avertissement ou une erreur, vous parvenez à l'occurrence de l'erreur.
5. Choisissez la commande de menu **Système cible > Charger les données de paramétrage** pour charger les valeurs actuelles modifiées depuis votre PG dans la CPU que vous avez affectée au programme S7 actuel.
6. Choisissez la commande de menu **Test > Visualiser** pour afficher l'état du programme pour le bloc ouvert et visualiser en ligne l'édition des valeurs actuelles chargées.

---

### Nota

Pour savoir si un bloc de données d'instance possède l'attribut système S7\_techparam, sélectionnez le bloc dans SIMATIC Manager, choisissez la commande de menu **Edition > Propriétés de l'objet**, puis ouvrez l'onglet "Attributs".

---

## 12.1 Paramétrage de fonctions technologiques

La fonction "Paramétrage de blocs de données" vous permet de paramétrer et de visualiser en ligne les blocs de régulation de température FB 58 "TCONT\_CP" et FB 59 "TCONT\_S" livrés dans la bibliothèque "Standard Library", grâce à une interface graphique facile à utiliser.

### Procédez de la manière suivante :

1. Ouvrez la bibliothèque standard "Standard Library" de STEP 7 en choisissant la commande **Fichier > Ouvrir > Bibliothèques** dans SIMATIC Manager.
2. Sélectionnez "PID Control Blocks", puis cliquez sur "Blocks". Les blocs fonctionnels suivants avec l'attribut "S7\_techparam" vous sont proposés :
  - **FB 58 "TCONT\_CP"** : régulateur de température pour actionneurs à signal d'entrée en continu ou pulsé
  - **FB 59 "TCONT\_S"** : régulateur de température pour actionneurs à action par intégration
3. Copiez le bloc fonctionnel souhaité (FB 58 ou FB 59) depuis la bibliothèque standard dans votre projet.
4. Créez un DB d'instance pour le FB sélectionné en choisissant la commande **Insertion > Bloc S7 > Bloc de données**.
5. Dans SIMATIC Manager, ouvrez le DB d'instance par double clic afin de démarrer la fonction "Paramétrage de blocs de données".  
**Résultat** : le DB d'instance s'ouvre dans la vue technologique. Vous pouvez à présent le paramétrer et le visualiser facilement en ligne.
6. Entrez les valeurs de régulation souhaitées dans la vue technologique. Les informations, avertissements et erreurs éventuels sont affichés dans la fenêtre de message. En effectuant un double-clic sur un avertissement ou une erreur, vous parvenez à l'endroit correspondant.

---

### Nota

Pour reconnaître un bloc qui possède l'attribut système "S7\_techparam", sélectionnez-le dans SIMATIC Manager, choisissez la commande de menu **Edition > Propriétés de l'objet** et ouvrez la page d'onglet "Attributs".

---

# 13 Création de sources LIST

## 13.1 Principes de la programmation dans des sources LIST

Vous pouvez saisir tout ou partie de votre programme sous forme de source LIST que vous compilez ensuite en blocs. La source peut contenir le code pour plusieurs blocs qui seront ensuite compilés en une seule opération en blocs.

Les avantages qui résident dans la création d'un programme dans une source sont les suivants :

- Vous pouvez créer et traiter votre source avec l'éditeur ASCII de votre choix, puis l'importer et la compiler en blocs individuels avec cette application. La compilation entraîne la génération des différents blocs et leur sauvegarde dans le programme utilisateur S7.
- Vous pouvez programmer plusieurs blocs dans une même source.
- Vous pouvez enregistrer une source malgré la présence éventuelle d'erreurs de syntaxe, ce qui n'est pas possible lors de la création de blocs de code avec vérification de syntaxe incrémentale. Cela signifie toutefois également que les erreurs de syntaxe ne vous seront signalées que lors de la compilation de la source.

Vous créez votre source dans la syntaxe du langage de programmation "liste d'instructions (LIST)". L'organisation de la source en blocs, déclaration de variables ou réseaux se fait à l'aide de mots-clés.

Lors de la création de blocs dans des sources LIST, vous devez considérer :

- Règles pour la programmation de sources LIST
- Syntaxe et formats autorisés dans les sources LIST
- Structure de bloc autorisée des sources LIST

## 13.2 Règles pour la programmation dans une source LIST

### 13.2.1 Règles pour la saisie d'instructions dans une source LIST

Une source LIST contient essentiellement un texte continu. Vous devez respecter des structures et des règles de syntaxe précises pour qu'elle puisse être traduite en blocs.

Les règles générales suivantes sont valables pour la création de programmes utilisateur sous forme de sources LIST :

Thème	Règle
Syntaxe	La syntaxe des instructions LIST est identique à celle dans l'éditeur LIST incrémental. L'opération d'appel CALL constitue une exception.
CALL	<p>Dans une source, vous indiquez les paramètres entre parenthèses. Les paramètres individuels sont séparés par une virgule.</p> <p>Exemple d'appel de FC (une ligne) :</p> <pre>CALL FC 10 (param1 :=E0.0,param2 :=E0.1);</pre> <p>Exemple d'appel de FB (une ligne) :</p> <pre>CALL FB10, DB100 (para1 :=E0.0,para2 :=E0.1);</pre> <p>Exemple d'appel de FB (plusieurs lignes) :</p> <pre>CALL FB10, DB100 (                 para1 :=E0.0,                 para2 :=E0.1);</pre> <p>Nota :</p> <p>Lors d'un appel de bloc dans l'éditeur ASCII, vous devez transmettre les paramètres selon un ordre défini. Sinon, l'affectation des commentaires de ces lignes ne sera pas correct dans l'affichage en LIST ou dans l'affichage de la source, le cas échéant.</p>
Majuscules ou minuscules	L'éditeur de cette application ne tient, en général, pas compte des majuscules et minuscules, si ce n'est pour les attributs système et les repères de saut. Vous devez également respecter les majuscules et minuscules lors de la saisie de chaînes de caractères (type de données STRING). Les mots-clés sont représentés en majuscules. Mais vous pouvez les indiquer en majuscules, en minuscules ou encore en majuscules et minuscules mélangées, car il n'est pas fait de différence entre majuscules et minuscules lors de la compilation.
Point-virgule	Vous devez signaler la fin de chaque instruction LIST et de chaque déclaration de variable par un point-virgule. Vous pouvez écrire plusieurs instructions par ligne.
Deux barres obliques (//)	Introduisez chaque commentaire par deux barres obliques (//) et achevez la saisie des commentaires avec la touche ENTREE.



### 13.2.2 Règles pour la déclaration de variables dans une source LIST

Vous devez déclarer les variables correspondant à chaque bloc de la source.

La déclaration des variables précède la section des instructions du bloc.

Si elles existent, il faut déclarer les variables dans l'ordre indiqué des types de déclarations. Ainsi, toutes les variables d'un même type de déclaration sont regroupées.

Alors qu'en CONT, LOG ou LIST vous remplissez une table de déclaration des variables, vous devez ici utiliser les mots-clés appropriés.

#### Mots-clés pour la déclaration des variables

Type de déclaration	Mots-clés	Possible pour ...
Paramètres d'entrée	"VAR_INPUT" Liste de déclaration "END_VAR"	FB, FC
Paramètres de sortie	"VAR_OUTPUT" Liste de déclaration "END_VAR"	FB, FC
Paramètres d'entrée/sortie	"VAR_IN_OUT" Liste de déclaration "END_VAR"	FB, FC
Variables statiques	"VAR" Liste de déclaration "END_VAR"	FB
Variables temporaires	"VAR_TEMP" Liste de déclaration END_VAR"	OB, FB, FC

Le mot-clé END\_VAR caractérise la fin d'une liste de déclaration.

La liste de déclaration correspond à la liste des variables d'un type de déclaration. Vous pouvez y affecter une valeur initiale aux variables (à l'exception de VAR\_TEMP). La figure suivante illustre la structure d'une entrée dans la liste de déclaration :

<b>Durée_moteur1</b>	<b>S5TIME</b>	<b>:=</b>	<b>S5T#1H_30M</b>	<b>;</b>
<b>Variable</b>	<b>Type de données</b>		<b>Valeur par défaut</b>	

#### Nota

- Le nom de la variable doit commencer par une lettre ou le caractère de soulignement. Vous ne devez pas indiquer de nom de variable correspondant à un mot-clé réservé.
- Si des noms de variables sont identiques dans les déclarations locales et dans la table des mnémoniques, faites précéder les noms des variables locales du signe # et écrivez les variables de la table des mnémoniques entre guillemets. Sinon, le bloc interprète la variable comme variable locale.

### 13.2.3 Règles pour l'ordre des blocs dans une source LIST

Les blocs appelés doivent précéder les blocs appelants, c'est-à-dire :

- L'OB1 utilisé dans la plupart des cas et qui appelle d'autres blocs vient en dernier. De même, les blocs appelés par des blocs eux-mêmes appelés dans l'OB1 doivent précéder ces blocs.
- Les types de données utilisateur (UDT) doivent précéder les blocs où ils sont utilisés.
- Les blocs de données associés à un type de données utilisateur (UDT) doivent se trouver après cet UDT.
- Les blocs de données globaux doivent précéder tous les blocs qui les appellent.
- Un bloc de données d'instance doit se trouver après le bloc fonctionnel auquel il est associé.
- Le DB0 est réservé ; vous ne pouvez pas générer de DB portant ce numéro.

### 13.2.4 Règles pour la définition d'attributs système dans une source LIST

Il est possible d'affecter à des blocs et à des paramètres des attributs système qui régissent la configuration des messages et des liaisons, les fonctions de contrôle-commande et la configuration du système.

Règles à respecter dans la source :

- Les mots-clés des attributs système commencent toujours par S7\_.
- Il faut écrire les attributs système entre accolades.
- Syntaxe : {S7\_identificateur := 'chaîne'}  
Il faut séparer les différents identificateurs par un point-virgule.
- Les attributs système pour blocs précèdent les propriétés de bloc, mais suivent les mots-clés ORGANIZATION\_ ou TITLE.
- Les attributs système pour paramètres figurent dans la déclaration des paramètres, c'est-à-dire avant le deux-points de la déclaration des données.
- La distinction est faite entre les majuscules et les minuscules, ce qui signifie que les majuscules et minuscules sont significatives pour la saisie d'attributs système !

Vous pouvez contrôler et modifier les attributs système pour blocs en saisie incrémentale à l'aide de la commande **Fichier > Propriétés** qui ouvre la page d'onglet "Attributs".

Vous pouvez contrôler et modifier les attributs système pour paramètres en saisie incrémentale à l'aide de la commande **Edition > Propriétés de l'objet**. Le curseur doit se trouver dans le champ de nom de la déclaration de paramètre.

### 13.2.5 Règles pour la définition de propriétés de bloc dans une source LIST

Les propriétés de bloc vous permettent de mieux identifier les blocs créés (par exemple, grâce au numéro de version) ou de les protéger de modifications non autorisées.

Vous pouvez les contrôler et les modifier en saisie incrémentale, à l'aide de la commande **Fichier > Propriétés**, dans les pages d'onglet "Fiche d'identité, partie 1" et "Fiche d'identité, partie 2".

Vous ne pouvez indiquer les autres propriétés de bloc que dans la source.

Règles à respecter dans la source :

- Il faut indiquer les propriétés de bloc avant la section de déclaration des variables.
- Utilisez une ligne par propriété de bloc.
- N'achevez pas la ligne par un point-virgule.
- Faites précéder chaque propriété de bloc de son mot-clé.
- Si vous désirez saisir des propriétés de bloc, respectez l'ordre du tableau des propriétés de bloc.
- Vous trouverez les propriétés possibles pour un type de bloc dans Affectation de propriétés de bloc selon les types de blocs.

---

#### Nota

Les propriétés de bloc sont également affichées dans SIMATIC Manager, dans les propriétés d'objet pour un bloc. Là, il est aussi possible d'éditer les propriétés AUTHOR, FAMILY, NAME et VERSION.

---

## Propriétés de bloc et ordre

Lorsque vous indiquez des propriétés de bloc, vous devez respecter l'ordre donné dans le tableau suivant.

Ordre	Mot-clé / Propriété	Signification	Exemple
1.	[KNOW_HOW_PROTECT]	Protection du bloc : il est impossible de visualiser la section des instructions d'un bloc compilé avec cette option. Il est possible de lire l'interface du bloc et non de la modifier.	KNOW_HOW_PROTECT
2.	[AUTHOR:]	Nom de l'auteur, nom de la société, du service ou autres noms (8 caractères au maximum, sans espace)	AUTHOR : Siemens, mais pas de mot-clé
3.	[FAMILY:]	Nom de la famille du bloc : par exemple, Regul (8 caractères au maximum, sans espace)	FAMILY : Regul, mais pas de mot-clé
4.	[NAME:]	Nom du bloc (8 caractères au maximum)	NAME : PID, mais pas de mot-clé
5.	[VERSION: int1 . int2]	Numéro de version du bloc (ces deux nombres entre 0 et 15, soit 0.0 à 15.15)	VERSION : 3.10
6.	[CODE_VERSION1]	Identification indiquant si un FB admet des multi-instances ou non. Si vous voulez déclarer des multi-instances, le FB ne doit pas avoir cette propriété.	CODE_VERSION1
7.	[UNLINKED] seulement pour DB	Les blocs de données possédant l'attribut UNLINKED sont uniquement stockés dans la mémoire de chargement, n'occupent pas d'espace dans la mémoire de travail et ne sont pas reliés au programme. Il n'est pas possible d'y accéder avec des commandes MC7. Les contenus de tels DB peuvent uniquement être transférés dans la mémoire de travail au moyen des SFC 20 BLKMOV (S7-300. S7-400)	
8.	[READ_ONLY] seulement pour DB	Protection pour blocs de données : il est uniquement possible de lire les données et non de les modifier.	FAMILY= Exemples VERSION= 3.10 READ_ONLY

### 13.2.6 Propriétés de bloc autorisées pour chaque type de bloc

Le tableau suivant présente les propriétés que vous pouvez déclarer pour les différents types de blocs.

Propriété	OB	FB	FC	DB	UDT
KNOW_HOW_PROTECT	•	•	•	•	–
AUTHOR	•	•	•	•	–
FAMILY	•	•	•	•	–
NAME	•	•	•	•	–
VERSION	•	•	•	•	–
Attribut "Unlinked"	–	–	–	•	–
READ_ONLY	–	–	–	•	–

#### Définition d'une protection de bloc avec KNOW\_HOW\_PROTECT

Vous pouvez protéger un bloc en indiquant le mot-clé KNOW\_HOW\_PROTECT dans la source LIST lors de la programmation du bloc.

Protéger un bloc a les conséquences suivantes :

- Lorsque vous afficherez plus tard un bloc compilé dans l'éditeur CONT, LOG ou LIST incrémental, vous n'aurez pas accès à la section des instructions de ce bloc.
- Seules les variables de types de déclarations IN, OUT et IN\_OUT seront visualisées dans la table de déclaration des variables du bloc. Les variables internes déclarées comme STAT et TEMP seront masquées.
- Indiquez le mot-clé KNOW\_HOW\_PROTECT avant toutes les autres propriétés du bloc.

#### Définition d'une protection en écriture pour les blocs de données avec READ\_ONLY

Vous pouvez définir une protection en écriture pour les blocs de données afin que leur contenu ne soit pas remplacé lors de l'exécution du programme. A cet effet, le bloc de données doit exister sous forme de source LIST.

Dans la source, indiquez le mot-clé READ\_ONLY. Il doit se trouver juste avant les déclarations de variables dans sa propre ligne.

## 13.3 Structure des blocs dans une source LIST

La structuration de blocs dans une source LIST s'effectue au moyen de mots-clés. Selon le type de bloc, l'on distingue la structure de :

- blocs de code
- blocs de données
- types de données utilisateur

### 13.3.1 Structure des blocs de code dans une source LIST

Un bloc de code se compose des zones suivantes, introduites par leur mot-clé respectif :

- Début de bloc,
- identifié par un mot-clé et un numéro ou un nom, par exemple :
  - "ORGANIZATION\_BLOCK OB 1" pour un bloc d'organisation
  - "FUNCTION\_BLOCK FB 6" pour un bloc fonctionnel
  - "FUNCTION FC 1 : INT" pour une fonction. Son type est également indiqué. Il peut s'agir d'un type de données simple ou complexe (sauf ARRAY et STRUCT) et c'est lui qui détermine le type de données de la valeur en retour (RET\_VAL). Indiquez le mot-clé VOID si la fonction ne doit pas renvoyer de valeur.
- Titre de bloc facultatif, introduit par le mot-clé TITLE= (longueur maximale de 64 caractères).
- Commentaire supplémentaire, introduit par deux barres obliques // en début de ligne
- Indication des propriétés du bloc (facultative)
- Section de déclaration des variables
- Section des instructions introduite par BEGIN. Cette section contient un ou plusieurs réseaux identifiés par le mot-clé NETWORK. Vous ne pouvez pas indiquer de numéro de réseau.
- Titre de réseau facultatif pour chaque réseau réalisé, introduit par le mot-clé TITLE = (longueur maximale de 64 caractères).
- Commentaire supplémentaire pour chaque réseau, introduit par deux barres obliques // en début de ligne
- Fin de bloc identifiée par END\_ORGANIZATION\_BLOCK, END\_FUNCTION\_BLOCK ou END\_FUNCTION
- Le type de bloc et le numéro de bloc sont séparés par un espace. Vous pouvez écrire le mnémonique du bloc entre guillemets afin de garantir l'univocité entre noms de variables locales et noms dans la table des mnémoniques.

### 13.3.2 Structure des blocs de données dans une source LIST

Un bloc de données se compose des zones suivantes, introduites par leur mot-clé respectif :

- Début de bloc, identifié par le mot-clé et le numéro ou le nom du bloc, par exemple DATA\_BLOCK DB 26
- Indication (facultative) du type de données utilisateur ou du bloc fonctionnel auquel le DB est associé
- Titre de bloc facultatif, introduit par le mot-clé TITLE =. Ce titre est tronqué au-delà de 64 caractères.
- Commentaire de bloc facultatif, introduit par deux barres obliques //
- Indication des propriétés du bloc (facultative)
- Section de déclaration des variables (facultative)
- Section d'affectation avec valeurs initiales, introduite par BEGIN (facultative)
- Fin de bloc identifiée par END\_DATA\_BLOCK

Il existe trois types de blocs de données :

- les blocs de données (définis par l'utilisateur),
- les blocs de données associés à un type de données utilisateur (UDT),
- les blocs de données associés à un bloc fonctionnel (DB d'instance).

### 13.3.3 Structure des types de données utilisateur dans une source LIST

Un type de données utilisateur se compose des zones suivantes, introduites par leur mot-clé respectif :

- Début, identifié par le mot-clé TYPE et un numéro ou un nom, par exemple TYPE UDT 20
- Indication d'un type de données structuré
- Fin, identifiée par le mot-clé END\_TYPE

N'oubliez pas que la définition de types de données utilisateur doit se situer avant les blocs qui utilisent ces types de données.

## 13.4 Syntaxe et formats pour les blocs dans une source LIST

Les tableaux présentent la syntaxe et les formats que vous devez respecter lors de la programmation de <18>sources LIST. La syntaxe est indiquée comme suit :

- Chaque élément est décrit dans la colonne de droite.
- Les éléments obligatoires sont indiqués entre guillemets.
- Les indications entre crochets [...] sont facultatives.
- Les mots-clés sont donnés en majuscules.

### 13.4.1 Tableau du format pour les OB

Le tableau suivant présente, sous forme condensée, le format pour les blocs d'organisation dans les sources LIST :

Organisation	Description
"ORGANIZATION_BLOCK" n°-OB ou nom-OB	n°-OB est le numéro du bloc, par exemple OB 1 nom-OB est le mnémonique du bloc selon la table des mnémoniques
[TITLE= ]	Titre jusqu'au retour chariot ; il est tronqué au-delà de 64 caractères.
[Commentaire de bloc]	Commentaire facultatif précédé de //
[Attributs système pour blocs]	Attributs système pour blocs
[Propriétés de bloc]	Propriétés de bloc
Section de déclaration des variables	Déclaration des variables temporaires
"BEGIN"	Mot-clé séparant la section de déclaration des variables des instructions LIST
NETWORK	Début d'un réseau
[TITLE= ]	Titre de réseau (64 caractères au maximum)
[Commentaire de réseau]	Commentaire facultatif précédé de //
Liste des instructions LIST	Instructions du bloc
"END_ORGANIZATION_BLOCK"	Mot-clé indiquant la fin du bloc d'organisation



### 13.4.2 Tableau du format pour les FB

Le tableau suivant présente, sous forme condensée, le format pour les blocs fonctionnels dans les sources LIST :

Organisation	Description
"FUNCTION_BLOCK" n°-FB ou nom-FB	n°-FB est le numéro du bloc, par exemple FB 6 nom-FB est le mnémonique du bloc selon la table des mnémoniques
[TITLE= ]	Titre jusqu'au retour chariot ; il est tronqué au-delà de 64 caractères.
[Commentaire de bloc]	Commentaire facultatif précédé de //
[Attributs système pour blocs]	Attributs système pour blocs
[Propriétés de bloc]	Propriétés de bloc
Section de déclaration des variables	Déclaration des paramètres d'entrée, de sortie et d'entrée/sortie, ainsi que des variables temporaires ou statiques La déclaration des paramètres peut contenir les déclarations des attributs système pour paramètres.
"BEGIN"	Mot-clé séparant la section de déclaration des variables des instructions LIST
NETWORK	Début d'un réseau
[TITLE= ]	Titre de réseau (64 caractères au maximum)
[Commentaire de réseau]	Commentaire facultatif précédé de //
Liste des instructions LIST	Instructions du bloc
"END_FUNCTION_BLOCK"	Mot-clé indiquant la fin du bloc fonctionnel

### 13.4.3 Tableau du format pour les FC

Le tableau suivant présente, sous forme condensée, le format pour les fonctions dans les sources LIST :

Organisation	Description
"FUNCTION"    n°-FC : type-FC ou nom-FC : type-FC	n°-FC est le numéro de la fonction, par exemple FC 5 nom-FC est le mnémonique de la fonction selon la table des mnémoniques  type-FC indique le type de données de la valeur en retour (RET_VAL) de la fonction. Il peut s'agir d'un type de données simple ou complexe (sauf ARRAY et STRUCT) ou bien de VOID.  Si vous souhaitez utiliser des attributs système pour la valeur en retour (RET_VAL), vous devez inscrire les attributs système pour paramètres avant les deux-points de la déclaration des données.
[TITLE= ]	Titre jusqu'au retour chariot ; il est tronqué au-delà de 64 caractères.
[Commentaire de bloc]	Commentaire facultatif précédé de //
[Attributs système pour blocs]	Attributs système pour blocs
[Propriétés de bloc]	Propriétés de bloc
Section de déclaration des variables	Déclaration des paramètres d'entrée, de sortie et d'entrée/sortie, ainsi que des variables temporaires
"BEGIN"	Mot-clé séparant la section de déclaration des variables des instructions LIST
NETWORK	Début d'un réseau
[TITLE= ]	Titre de réseau (64 caractères au maximum)
[Commentaire de réseau]	Commentaire facultatif précédé de //
Liste des instructions LIST	Instructions du bloc
"END_FUNCTION"	Mot-clé indiquant la fin de la fonction

### 13.4.4 Tableau du format pour les DB

Le tableau suivant présente, sous forme condensée, le format pour les blocs de données dans les sources LIST :

Organisation	Description
"DATA_BLOCK" n°-DB ou nom-DB	n°-DB est le numéro du bloc, par exemple DB 5 nom-DB est le mnémonique du bloc selon la table des mnémoniques
[TITLE= ]	Titre jusqu'au retour chariot ; il est tronqué au-delà de 64 caractères.
[Commentaire de bloc]	Commentaire facultatif précédé de //
[Attributs système pour blocs]	Attributs système pour blocs
[Propriétés de bloc]	Propriétés de bloc
Section de déclaration	Indication de l'UDT ou du FB auquel le DB est associé sous forme de numéro de bloc ou de mnémonique selon la table des mnémoniques ou bien indication du type de données complexe
"BEGIN"	Mot-clé séparant la section de déclaration de la liste des affectations de valeurs
[Affectation de valeurs initiales]	Il est possible d'affecter des valeurs initiales aux variables : des constantes sont affectées à certaines variables ou il est fait référence à d'autres blocs.
"END_DATA_BLOCK"	Mot-clé indiquant la fin du bloc de données

## 13.5 Création d'une source LIST

### 13.5.1 Création d'une source LIST

Vous devez créer la source sous le programme S7, dans un dossier Sources. Vous pouvez créer une source dans SIMATIC Manager ou dans la fenêtre d'édition.

#### Création d'une source dans SIMATIC Manager

1. Ouvrez le dossier Sources correspondant en cliquant deux fois dessus.
2. Pour insérer une source LIST, choisissez la commande **Insertion > Logiciel S7 > Source LIST**.

#### Création d'une source dans la fenêtre d'édition

1. Choisissez la commande **Fichier > Nouveau**.
2. Choisissez, dans la boîte de dialogue, le dossier Sources du programme S7 dans lequel se trouve également le programme utilisateur avec les blocs.
3. Attribuez un nom à la source à créer.
4. Confirmez par "OK".

La source est créée avec le nom que vous avez indiqué et affichée dans une fenêtre de travail pour édition.

### 13.5.2 Edition d'une source S7

Le langage de programmation et l'éditeur avec lesquels vous éditez une source donnée sont définis dans les propriétés de l'objet. Ainsi, l'éditeur correct sera toujours démarré avec le langage de programmation correspondant à la source. Le logiciel de base STEP 7 permet la programmation dans une source LIST.

D'autres langages de programmation sont toutefois disponibles sous forme de logiciels optionnels. Le logiciel optionnel doit être installé sur votre ordinateur, pour que vous puissiez démarrer l'éditeur correspondant en cliquant deux fois sur la source.

Procédez de la manière suivante :

1. Ouvrez le dossier Sources correspondant en cliquant deux fois dessus.
2. Démarrez l'éditeur nécessaire à l'édition de la manière suivante :
  - Cliquez deux fois sur la source correspondante dans la partie droite de la fenêtre.
  - Sélectionnez la source correspondante dans la partie droite de la fenêtre et choisissez la commande **Edition > Ouvrir l'objet**.

### 13.5.3 Définition de la mise en page du texte source

Pour augmenter la lisibilité du texte dans les fichiers source, choisissez la commande de menu **Outils > Paramètres**, puis dans l'onglet "Texte source" définissez la police et la couleur des divers éléments du texte source.

Vous pouvez p. ex. choisir d'afficher les numéros de ligne et de représenter les mots-clés en majuscules.

### 13.5.4 Insertion de modèles de blocs dans une source LIST

Vous disposez, pour la programmation de sources LIST, de modèles de blocs pour OB, FB, FC, DB, DB d'instance, DB associés à des UDT et UDT. Ces modèles de blocs vous facilitent la saisie et le respect de la syntaxe et de l'organisation des différents blocs.

Procédez de la manière suivante :

1. Activez la fenêtre de travail de la source dans laquelle vous voulez insérer un modèle de bloc.
2. Positionnez le curseur à l'emplacement après lequel vous voulez insérer le modèle de bloc.
3. Choisissez la commande correspondante **Insertion > Modèle de bloc \_\_ > OB/FB/FC/DB/IDB/DB** à partir de UDT/UDT.

Le modèle de bloc est alors inséré après la position du curseur.

### 13.5.5 Insertion du contenu d'autres source LIST

Vous pouvez insérer dans votre source LIST le contenu d'autres sources.

Procédez de la manière suivante :

1. Activez la fenêtre de travail de la source dans laquelle vous voulez insérer le contenu d'une autre source.
2. Positionnez le curseur à l'emplacement après lequel vous voulez insérer le contenu de cette autre source.
3. Choisissez la commande **Insertion > Objet > Fichier**.
4. Choisissez, dans la boîte de dialogue qui apparaît alors, la source désirée.

Le contenu de la source sélectionnée est inséré après la position du curseur. Les retours à la ligne y sont conservés.

### 13.5.6 Insertion du code source de blocs existant dans une source LIST

Vous pouvez insérer dans votre source LIST le code source de blocs programmés en CONT, LOG ou LIST. Cela est possible pour les OB, les FB, les FC, les DB et les types de données utilisateur (UDT).

Procédez de la manière suivante :

1. Activez la fenêtre de travail de la source dans laquelle vous voulez insérer un bloc.
2. Positionnez le curseur à l'emplacement après lequel vous voulez insérer le texte du bloc.
3. Choisissez la commande **Insertion > Objet > Bloc**.
4. Choisissez, dans la boîte de dialogue qui apparaît alors, le bloc désiré.

Une source est implicitement générée à partir du bloc désiré. Le contenu de cette source est inséré après la position du curseur.

### 13.5.7 Insertion d'une source externe

Vous pouvez créer et traiter votre source avec l'éditeur ASCII de votre choix, puis l'importer dans un projet et la compiler en blocs individuels. Les sources doivent être importées dans le dossier Sources du programme S7, pour que les blocs résultant de la compilation soient enregistrés dans le programme utilisateur S7 de ce même programme S7.

Procédez de la manière suivante :

1. Sélectionnez le dossier Sources du programme S7 dans lequel vous voulez importer les sources externes.
2. Choisissez la commande **Insertion > Source externe**.
3. Dans la boîte de dialogue suivante, indiquez la source à importer.

L'extension donnée au nom de fichier de la source à importer doit être valide. En effet, c'est l'extension qui permet de déterminer le type de la source dans STEP 7. Ainsi, un fichier d'extension **.awl** sera importé comme source LIST. Les extensions de fichier possibles sont indiquées dans la boîte de dialogue sous "Type de fichier".

---

#### Nota

La commande **Insertion > Source externe** vous permet également d'insérer d'anciennes sources qui ont été créées dans la version 1 de STEP 7.

---

### 13.5.8 Génération d'une source LIST à partir de blocs

Vous pouvez générer, à partir de blocs existants, une source LIST que vous pouvez traiter avec un éditeur de texte de votre choix. La source générée est créée dans le dossier source du programme S7.

Procédez de la manière suivante :

1. Choisissez la commande **Fichier > Générer source** dans l'éditeur de programmes.
2. Sélectionnez, dans la boîte de dialogue, le dossier Sources dans lequel vous désirez ranger la nouvelle source.
3. Attribuez un nom à la source dans la zone correspondante.
4. Sélectionnez, dans la boîte de dialogue "Sélection de blocs STEP 7", le ou les blocs à partir desquels vous voulez générer la source choisie auparavant. Les blocs sélectionnés s'affichent dans la liste à droite.
5. Confirmez par "OK".

Les blocs sélectionnés sont alors compilés en une source LIST continue qui s'affiche pour édition dans une fenêtre de travail.

### 13.5.9 Importation d'une source

Pour importer une source depuis un répertoire quelconque dans un projet, procédez de la manière suivante :

1. Dans SIMATIC Manager, sélectionnez le dossier Sources dans lequel vous souhaitez importer la source.
2. Choisissez la commande **Insertion > Source externe**.
3. Dans la boîte de dialogue affichée, sélectionnez le répertoire et la source à importer.
4. Cliquez sur le bouton "Ouvrir".

### 13.5.10 Exportation d'une source

Pour exporter la source d'un projet dans un répertoire cible quelconque, procédez de la manière suivante :

1. Sélectionnez la source dans le dossier Sources.
2. Choisissez la commande **Edition > Exporter une source**, dans SIMATIC Manager.
3. Indiquez le répertoire cible ainsi que le nom de fichier dans la boîte de dialogue affichée.
4. Cliquez sur le bouton "Enregistrer".

---

#### Nota

Si le nom de l'objet ne possède pas d'extension, il en sera ajoutée une à son nom de fichier à partir de son type. Une source LIST "**prog**", par exemple, sera exportée dans le fichier "**prog.awl**".

Si le nom de l'objet possède déjà une extension valide, celle-ci sera conservée et aucune autre ne lui sera ajoutée. Une source LIST "**prog.awl**", par exemple, sera exportée dans le fichier "**prog.awl**".

Si un nom d'objet possède une extension erronée (c'est-à-dire le nom comporte un point), aucune extension ne sera ajoutée au fichier.

Les extensions de fichier possibles sont indiquées dans la boîte de dialogue "Exporter source" sous "Type de fichier".

---



## 13.6 Enregistrement, compilation et vérification d'une source LIST

### 13.6.1 Enregistrement d'une source LIST

Vous pouvez sauvegarder une source LIST à tout moment dans l'état où elle est. Dans ce cas, le programme n'est pas compilé et sa syntaxe n'est pas vérifiée (les erreurs éventuelles sont donc également enregistrées).

Les erreurs de syntaxe ne seront détectées et signalées que lors de la compilation de la source ou lors d'une vérification de cohérence.

#### Pour enregistrer une source sous le même nom :

1. Activez la fenêtre de travail de la source à enregistrer.
2. Choisissez la commande **Fichier > Enregistrer**.

#### Pour enregistrer une source sous autre nom ou dans un autre projet :

1. Activez la fenêtre de travail de la source à enregistrer.
2. Choisissez la commande **Fichier > Enregistrer sous**.
3. Sélectionnez, dans la boîte de dialogue qui apparaît alors, le dossier Sources auquel la source doit être affectée et indiquez le nouveau nom de la source.

### 13.6.2 Vérification de la cohérence d'une source LIST

En choisissant la commande **Fichier > Vérifier la cohérence**, vous pouvez afficher d'éventuelles erreurs de syntaxe dans une source LIST. Contrairement à la compilation, cette vérification n'entraîne pas la génération des blocs.

Une fois la vérification de cohérence achevée, apparaît une boîte de dialogue qui indique le nombre total d'erreurs trouvées.

S'il existe des erreurs, elles sont toutes énumérées dans la partie inférieure de la fenêtre de travail avec indication de leur ligne. Vous devez les éliminer avant la compilation de la source pour que tous les blocs soient générés.

### 13.6.3 Recherche d'erreurs dans une source LIST

La fenêtre de travail pour les sources comporte deux parties. Dans sa moitié inférieure sont énumérées les erreurs suivantes :

- erreurs détectées après déclenchement d'une compilation via la commande **Fichier > Compiler** ;
- erreurs détectées après déclenchement d'une vérification de cohérence via la commande **Fichier > Vérifier la cohérence**.

Pour trouver l'emplacement d'une erreur dans la source, positionnez le curseur sur le message d'erreur en question dans l'onglet "Erreurs" de la fenêtre des erreurs et avertissements. La ligne de texte correspondante est alors automatiquement sélectionnée dans la section des instructions. Le message d'erreur est, en outre, repris dans la barre d'état.

## 13.6.4 Compilation d'une source LIST

### Condition préalable

Afin que le programme créé dans une source puisse être compilé en blocs, les conditions suivantes doivent être remplies :

- Seules peuvent être compilées les sources qui sont enregistrées dans le dossier Sources sous un programme S7.
- Un dossier Blocs doit se trouver sous le programme S7, au même niveau que le dossier Sources pour que les blocs compilés puissent y être enregistrés. Les blocs programmés dans la source ne sont créés que si aucune erreur n'est décelée durant la compilation de la source. Seuls les blocs d'une source exempts d'erreurs sont compilés. Vous pouvez ensuite ouvrir ces blocs individuellement, les éditer, les charger dans une CPU et les tester.

### Marche à suivre dans l'éditeur

1. Ouvrez la source que vous voulez compiler. Elle doit se trouver dans le dossier Sources du programme S7 dans le programme utilisateur duquel les blocs compilés doivent être rangés.
2. Choisissez la commande **Fichier > Compiler**.
3. S'ouvre alors la boîte de dialogue "Journal de la compilation" qui montre le nombre de lignes compilées et le nombre d'erreurs de syntaxe détectées.

Les blocs indiqués dans la source ne sont générés que si la source a été compilée sans erreur. Seuls les blocs d'une source exempts d'erreurs sont compilés. Les avertissements n'empêchent pas la génération des blocs.

Les erreurs de syntaxe détectées lors de la compilation sont représentées dans la moitié inférieure de la fenêtre de travail et doivent être corrigées pour que les blocs correspondants puissent également être générés.

### Marche à suivre dans SIMATIC Manager

1. Ouvrez le dossier Sources correspondant en cliquant deux fois dessus.
2. Sélectionnez une ou plusieurs sources à compiler. Vous ne pouvez pas effectuer la compilation d'un dossier Sources fermé afin de compiler toutes les sources qu'il contient.
3. Choisissez la commande **Fichier > Compiler** pour démarrer la compilation. Le compilateur correspondant à la source sélectionnée est appelé. Les blocs correctement compilés sont ensuite enregistrés dans le dossier Blocs sous le programme S7. Les erreurs de syntaxe décelées durant la compilation sont signalées dans une boîte de dialogue et doivent être corrigées afin que ces blocs puissent eux aussi être créés.

## 13.7 Exemples de sources LIST

### 13.7.1 Exemples de déclarations de variables dans une source LIST

#### Variables de type de données simple

```

// Les commentaires sont séparés de la section de déclaration par //.
VAR_INPUT // Mot-clé variable d'entrée
    in1 : INT; // Nom de variable et type sont séparés par ":"
    in3 : DWORD; // Un point-virgule met fin à chaque déclaration de variable.
    in2 : INT := 10; // Définition facultative de la valeur initiale dans la déclaration
END_VAR // Fin de la déclaration des variables de même type de déclaration
VAR_OUTPUT // Mot-clé variable de sortie
    out1 : WORD;
END_VAR // Mot-clé variable temporaire
VAR_TEMP
    temp1 : INT;
END_VAR

```

#### Variables de type de données ARRAY

```

VAR_INPUT // Variable d'entrée
    champ1 : ARRAY [1..20] of INT; // champ1 est un champ unidimensionnel
    champ2 : ARRAY [1..20, 1..40] of DWORD; // champ2 est un champ
bidimensionnel
END_VAR

```

#### Variables de type de données STRUCT

```

VAR_OUT // Variable de sortie
SORTIE1: STRUCT // SORTIE1 est de type de données STRUCT.
    var1 : BOOL; // Élément 1 de la structure
    var2 : DWORD; // Élément 2 de la structure
END_STRUCT; // Fin de la structure
END_VAR

```

### 13.7.2 Exemple d'OB dans une source LIST

```

ORGANIZATION_BLOCK OB 1
TITLE = Exemple d'OB 1 avec différents appels de blocs
//Les 3 réseaux représentés montrent des appels de blocs
//avec et sans paramètres.

{S7_pdiag := 'true'} //Attribut système pour blocs
AUTHOR:      Siemens
FAMILY:      Exemple
Nom          OB_test
VERSION:     1.1
VAR_TEMP
VallInterm : INT; // Mémoire intermédiaire
END_VAR

BEGIN

NETWORK
TITLE = Appel d'une fonction avec transmission de paramètres
// Transmission de paramètres en une ligne
CALL FC1 (param1 :=E0.0,param2 :=E0.1);

NETWORK
TITLE = Appel d'un bloc fonctionnel avec
//transmission de paramètres
// Transmission de paramètres en plusieurs lignes
CALL Regul_feux , DB 6 ( // Nom du FB, DB d'instance
dur_v_p      := S5T#10S, // Affectation de valeurs effectives aux paramètres

eff_r_p      := S5T#30S,
demarr       := TRUE,
t_dur_o_voit := T 2,
t_dur_v_piet := T 3,
t_ret_o_voit := T 4,
t_dur_r_voit := T 5,
t_rou_suiv_voit := T 6,
r_voit       := "ro_main", // Les guillemets identifient les noms
o_voit       := "or_main", // de la table des mnémoniques.
v_voit_      := "ve_main",
r_piet       := "ro_int",
v_piet       := "ve_int");

NETWORK
TITLE = Appel d'un bloc fonctionnel avec
//transmission de paramètres
// Transmission de paramètres en une ligne
CALL FB10, DB100 (para1 :=E0.0,para2 :=E0.1);

END_ORGANIZATION_BLOCK

```

### 13.7.3 Exemple de FC dans une source LIST

```
FUNCTION FC 1: VOID
// Seulement pour l'appel
VAR_INPUT
  param1 : bool;
  param2 : bool;
END_VAR
Begin
end_function

FUNCTION FC2 : INT
TITLE = Augmentation de la production
// Tant que la valeur transmise est < 1000, cette fonction
//augmente la valeur transmise. Si le nombre de pièces est
//supérieur à 1000, "-1" est renvoyé via la valeur en retour
//de la fonction (RET_VAL).

AUTHOR:      Siemens
FAMILY:      Product
Nom          PIECES
VERSION:     1.0

VAR_IN_OUT
PRODUCTION : INT;           // Nombre de pièces effectivement produites
END_VAR

BEGIN

NETWORK
TITLE = Augmentation production de 1
// Tant que la production effective est inférieure à 1000,
// elle peut être augmentée d'1.
L PRODUCTION; L 1000;           // Exemple de plusieurs
> I; SPB ERR;                   // instructions dans une ligne
L 0; T RET_VAL;
L PRODUCTION; INC 1; T PRODUCTION; BEA;
ERR: L -1;
T RET_VAL;
END_FUNCTION

FUNCTION FC3 {S7_pdiag := 'true'} : INT
TITLE = Augmentation de la production
// Tant que la valeur transmise est < 1000, cette fonction
//augmente la valeur transmise. Si le nombre de pièces est
//supérieur à 1000, "-1" est renvoyé via la valeur en retour
//de la fonction (RET_VAL).
//
//RET_VAL comporte ici un attribut système pour paramètres.
```

```
AUTHOR:      Siemens
FAMILY:      PRODUCT
Nom          PIECES
VERSION:     1.0

VAR_IN_OUT
PRODUCTION {S7_visible := 'true'}: INT;    // Nb. de pièces effectivement produites
//Attributs système pour paramètres
END_VAR
BEGIN

NETWORK
TITLE = Augmentation production de 1
// Tant que la production effective est inférieure à 1000,
// elle peut être augmentée d'1.
L PRODUCTION; L 1000;                      // Exemple de plusieurs
> I; SPB ERR;                               // instructions dans une ligne
L 0; T RET_VAL;
L PRODUCTION; INC 1; T PRODUCTION; BEA;
ERR: L -1;
T RET_VAL;

END_FUNCTION
```

### 13.7.4 Exemple de FB dans une source LIST

```
FUNCTION_BLOCK FB 6
TITLE = Réglage simple des feux
// Régulation des feux pour un passage piétons
// sur la rue principale

{S7_m_c := 'true'} //Attribut système pour blocs
AUTHOR:      Siemens
FAMILY:      Feux
Nom          Feux
VERSION:     1.3

VAR_INPUT

demarr:      BOOL := FALSE; // Demande de traversée piétons
t_dur_o_voit : TIMER; // Durée vert piétons
t_r_suiv_voit : TIMER; // Durée entre rouge pour voitures
t_dur_r_voit : TIMER;
nombre       {S7_server := 'alarm_archiv'; S7_a_type := 'alarm_8'} :DWORD;
// Nb. Voitures
// nombre comporte des attributs système pour paramètres

END_VAR
VAR_OUTPUT

v_voit      :      BOOL      :=      FALSE; // VERT pour voitures

END_VAR
VAR
condition   :      BOOL := FALSE; // Notification rouge pour voitures
END_VAR

END_FUNCTION_BLOCK
```

```
FUNCTION_BLOCK FB 10
```

```
VAR_INPUT
```

```
  para1 : bool;
```

```
  para2: bool;
```

```
end_var
```

```
begin
```

```
end_function_block
```

```
data_block db 10
```

```
fb10
```

```
begin
```

```
end_data_block
```

```
data_block db 6
```

```
fb6
```

```
begin
```

```
end_data_block
```



## 13.7.5 Exemples de DB dans une source LIST

### Bloc de données

```
DATA_BLOCK DB 10
TITLE = Exemple DB 10
STRUCT
    aa : BOOL;      // Variable aa de type BOOL
    bb : INT; // Variable bb de type INT
    cc : WORD;
END_STRUCT;
BEGIN // Affectation de valeurs initiales
    aa := TRUE;
    bb := 1500;
END_DATA_BLOCK
```

### DB associé à un type de données utilisateur

```
DATA_BLOCK DB 20
TITLE = Exemple DB (UDT)
UDT 20 // Indication de l'UDT associé
BEGIN
    demarr := TRUE; // Affectation de valeurs initiales
    consigne := 10;
END_DATA_BLOCK
```

#### Nota

L'UDT utilisé doit se situer avant le bloc de données dans la source.

### DB associé à un bloc fonctionnel

```
DATA_BLOCK DB 30
TITLE = Exemple DB (FB)
FB 30 // Indication du FB associé
BEGIN
    demarr := TRUE; // Affectation de valeurs initiales
    consigne := 10;
END_DATA_BLOCK
```

#### Nota

Le FB associé doit se situer avant le bloc de données dans la source.

### 13.7.6 Exemple d'UDT dans une source LIST

```
TYPE UDT 20
STRUCT
    demarr : BOOL;           //Variable de type BOOL
    consigne : INT;         //Variable de type INT
    valeur : WORD;          //Variable de type WORD
END_STRUCT;
END_TYPE
```

## 14 Affichage des données de référence

### 14.1 Présentation des données de référence possibles

Afin de faciliter le test et la modification de votre programme utilisateur, vous pouvez créer et exploiter des données de référence. Les données de référence servent par exemple :

- de vue d'ensemble sur votre programme utilisateur complet,
- de base pour les modifications et les tests,
- à compléter la documentation de votre programme.

Le tableau suivant présente les informations que vous retrouvez dans les différentes vues :

Vue	Application
Liste des références croisées	Vue d'ensemble de l'emploi d'opérandes des zones de mémoire E, A, M, P, T, Z et d'appels de DB, FB, FC, SFB et SFC au sein du programme utilisateur. La commande <b>Affichage &gt; Références croisées pour l'opérande</b> vous permet d'afficher toutes les références croisées, y compris les accès multiples à l'opérande sélectionné.
Tableau d'affectation	La vue d'ensemble montrant quels bits des opérandes appartenant aux zones de mémoire E, A et M ou quels temporisations et compteurs sont déjà affectés au sein du programme utilisateur constitue une base importante pour la recherche d'erreurs et les modifications dans le programme utilisateur.
Structure du programme	Hiérarchie d'appel des blocs au sein du programme utilisateur et vue d'ensemble des blocs utilisés et de leurs relations de dépendance.
Opérandes libres	Vue d'ensemble de tous les mnémoniques définis dans la table des mnémoniques, mais qui ne sont pas utilisés dans les parties du programme utilisateur pour lesquelles il y a des données de référence.
Mnémoniques manquants	Vue d'ensemble de toutes les adresses absolues (opérandes et blocs) qui sont utilisées dans les parties du programme utilisateur et pour lesquelles il y a des données de référence, mais pour lesquelles aucun mnémonique n'est défini dans la table des mnémoniques.

Les données de référence du programme utilisateur sélectionné englobent les listes contenues dans le tableau. Vous pouvez créer et afficher plusieurs listes pour un programme utilisateur ou pour divers programmes.

#### Affichage simultané de plusieurs vues

L'affichage de listes dans des fenêtres supplémentaires vous permet par exemple :

- de comparer les mêmes types de listes de programmes utilisateur S7 différents,
- d'afficher côte à côte sur l'écran une même liste (par exemple, une liste des références croisées) optimisée différemment. Dans une liste de références croisées, vous pouvez par exemple afficher uniquement les entrées d'un programme utilisateur S7 et dans la seconde liste des références croisées uniquement les sorties.
- d'ouvrir simultanément plusieurs listes d'un même programme utilisateur S7, par exemple la structure du programme et la liste des références croisées.

### 14.1.1 Liste des références croisées

La liste des références croisées offre un aperçu de l'utilisation des opérandes dans un programme utilisateur S7.

La liste des références croisées indique les opérandes des zones de mémoire entrée(E), sortie (A), memento (M), temporisation (T), compteur (Z), bloc fonctionnel (FB), fonction (FC), bloc fonctionnel système (SFB), fonction système (SFC), périphérie (P) et bloc de données (DB) qui sont utilisés dans le programme utilisateur S7, leur adresse (adresse absolue, nom) ainsi que leur utilisation. Cette liste s'affiche dans une fenêtre de travail. La barre de titre de cette fenêtre donne le nom du programme utilisateur auquel appartient la liste des références croisées.

Chaque ligne de la fenêtre correspond à une entrée de la liste des références croisées. Une fonction de recherche permet de retrouver facilement des opérandes et mnémoniques précis.

La liste des référence croisées est la vue par défaut pour l'affichage des données de référence. Vous pouvez modifier cette valeur par défaut.

#### Structure

Elle comprend les colonnes suivantes :

Colonne	Contenu/signification
Opérande (mnémon.)	Adresse de l'opérande
Bloc (mnémon.)	Bloc dans lequel l'opérande est utilisé.
Accès	Indique si l'accès à l'opérande est un accès en lecture (R) ou en écriture (W).
Langage	Informations relatives au langage de création du bloc
Occurrence	Pour sauter à l'occurrence de l'opérande sélectionné, effectuez un double clic dans le champ d'une occurrence.

Les colonnes Bloc, Type, Langage et Occurrence ne s'affichent que lorsque les propriétés correspondantes ont été sélectionnées pour la liste des références croisées. Ces informations sur le bloc varient en fonction du langage dans lequel le bloc a été créé.

A l'aide de la souris, adaptez la largeur des colonnes à vos besoins dans la liste des références croisées affichée à l'écran.

#### Tri

Par défaut, la liste des références croisées est classée par zones de mémoire. Pour la classer selon les entrées d'une colonne précise, cliquez avec la souris sur le titre de cette colonne.

#### Exemple de structure de la liste des références croisées

Opérande (mnémon.)	Bloc (mnémon.)	Accès	Langage	Occurrence
E 1.0 (Moteur marche)	OB 2	R	LIST	NW 2 Anw 33 /O
M1.2 (Bit de memento)	FC 2	R	CONT	NW 33
Z2 (Compteur2)	FB2		LOG	NW2

## 14.1.2 Structure du programme

La structure du programme décrit la hiérarchie d'appel des blocs à l'intérieur d'un programme utilisateur S7. Vous obtenez en outre un aperçu des blocs utilisés, de leur relations et de leur besoin en données locales.

En choisissant la commande **Affichage > Filtrer** dans la fenêtre "Afficher les données de référence S7", vous ouvrez une boîte de dialogue à onglets. Dans la page d'onglet "Structure du programme", vous pouvez choisir la représentation de la structure du programme :

Vous avez le choix entre

- structure d'appels et
- structure des dépendances.

Vous pouvez demander la représentation de tous les blocs ou l'affichage de la hiérarchie à partir d'un bloc précis.

### Icônes utilisées dans la structure du programme

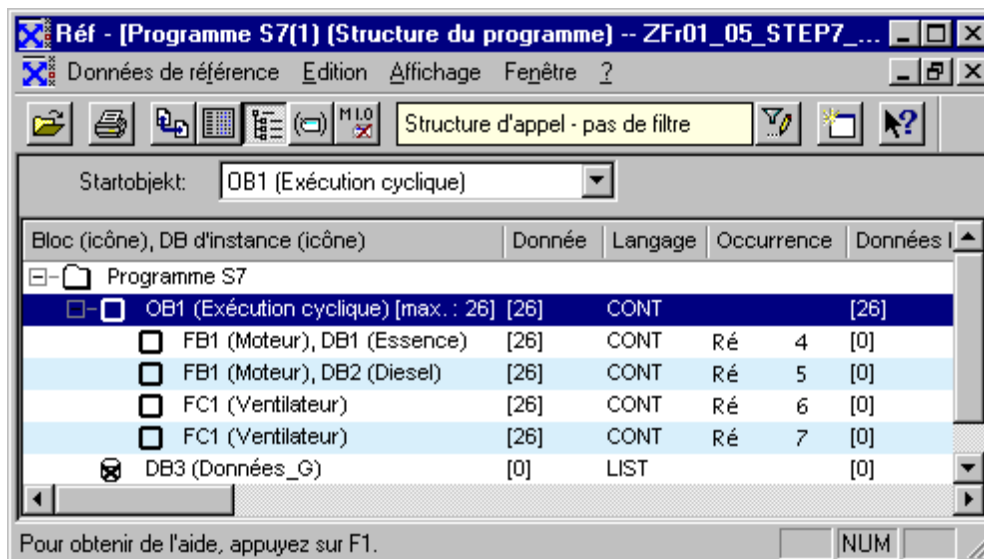
Icône Signification

- |                                     |  |
|-------------------------------------|--|
| <input type="checkbox"/>            | Appel normal d'un bloc (CALL FB10)       |
| <input type="checkbox"/>            | Appel inconditionnel d'un bloc (UC FB10) |
| <input type="checkbox"/>            | Appel conditionnel d'un bloc (CC FB10)   |
| <input type="checkbox"/>            | Bloc de données                          |
| <input type="checkbox"/>            | Récurrence                               |
| <input type="checkbox"/>            | Récurrence et appel conditionnel         |
| <input type="checkbox"/>            | Récurrence et appel inconditionnel       |
| <input checked="" type="checkbox"/> | Bloc non appelé                          |

- Les récurrences d'appel sont détectées et marquées comme telles graphiquement dans la structure d'appels.
- Des récurrences au sein de la hiérarchie d'appel sont représentées par des boutons différents.
- L'appel normal d'un bloc (CALL), l'appel conditionnel d'un bloc (CC) et l'appel inconditionnel d'un bloc (UC) sont caractérisés par des boutons différents.
- Les blocs non appelés sont indiqués à la fin de la structure arborescente et marqués d'une croix noire. Leur structure d'appel ne sera pas détaillée davantage.

## Affichage sous forme de structure d'appels

La totalité de la structure d'appels est représentée.



Si l'arborescence doit être réalisée pour tous les blocs d'organisation (OB) et si l'OB 1 ou le bloc de départ indiqués ne se trouvent pas dans le programme utilisateur S7, le logiciel vous invitera automatiquement à spécifier un autre bloc qui servira de racine à l'arborescence.

L'affichage d'appels multiples de blocs peut être désactivé, aussi bien pour la structure d'appels que pour les "Structure des dépendances", grâce aux options proposées.

## Affichage du besoin maximal en données locales dans la structure d'appels

Pour voir immédiatement le besoin en données locales des OB dans le programme utilisateur affiché, vous pouvez afficher dans la représentation arborescente :

- le besoin maximal en données locales par OB
- le besoin en données locales par chemin.

Vous pouvez activer ou désactiver cet affichage dans l'onglet "Structure du programme".

En cas de présence d'OB d'erreur synchrones (OB 121, OB 122), un signe plus ainsi que la place requise pour les OB d'erreur synchrones sont affichés après la valeur des données locales maximales.

## Structure des dépendances

La structure des dépendances affiche pour chaque bloc du projet, ses dépendances avec les autres blocs. Le bloc en question est représenté entièrement à gauche, les blocs qui l'appellent ou l'utilisent étant représentés en-dessous de manière décalée.

## Affichage de blocs supprimés

Les lignes correspondant à des blocs supprimés sont représentées en rouge.

### 14.1.3 Tableau d'affectation

Le tableau d'affectation vous montre quels opérandes sont déjà occupés dans le programme utilisateur. Cet affichage constitue un élément important pour la recherche d'erreurs ou les modifications dans le programme utilisateur.

L'affichage du tableau d'affectation permet de savoir quel bit est utilisé dans quel octet des zones de mémoire entrée (E), sortie (A), et memento (M) de même que quelles temporisations (T) et quels compteurs (Z) sont utilisés.. Le tableau d'affectation s'affiche dans une fenêtre de travail. La barre de titre de la fenêtre de travail donne le nom du programme utilisateur S7 auquel il appartient.

#### Vue d'ensemble E/A/M

Chaque ligne contient un octet de la zone de mémoire, dont les huit bits sont différenciés selon leur mode d'accès. En outre, il sera également précisé s'ils sont adressés par octet, mot ou double mot.

#### Identificateur dans la vue d'ensemble E/A/M :

arrière-plan blanc	l'opérande n'est pas adressé et n'est donc pas encore occupé
x	l'opérande est utilisé directement
arrière-plan bleu	l'opérande est traité indirectement (accès par octet, mot ou double mot)

#### Colonnes dans la vue d'ensemble E/A/M :

Colonne	Contenu/signification
7	numéro du bit de l'octet correspondant
6	
5	
4	
3	
2	
1	
0	
O	l'octet est adressé par un octet
W	l'octet est adressé par un mot
D	l'octet est adressé par un mot double

### Exemple

L'exemple suivant représente la structure typique d'un tableau d'affectation pour les entrées, sorties et mémentos (E/A/M).

△	7	6	5	4	3	2	1	0	B	W	D
EB0	X	X	X	X	X	X					
EB1		X	X	X		X	X	X			
AB4						X	X	X			
AB5		X	X	X		X	X	X			
MB1										↑	
MB2										↑	
MB3										↑	
MB4										↑	
MB5										↑	

La première ligne montre l'occupation de l'octet d'entrée EB 0. Les entrées de l'opérande EB 0 sont utilisées directement (accès par bit). La lettre "X" qui figure respectivement dans les colonnes "0", "1", "2", "3", "5", et "6" caractérise l'accès par bit.

De plus, les octets de memento 1 et 2, 2 et 3 ou 4 et 5 sont accédés par mot, ce qui est représenté par une "barre" dans la colonne "W". De plus, les cellules sont représentées sur fond bleu clair. La pointe noire de la barre désigne le début de l'accès par mot..

### Vue d'ensemble T/Z

Chaque ligne représente 10 temporisations ou compteurs.

### Exemple

	0	1	2	3	4	5	6	7	8	9
T 00-09	.	T1	.	.	.		T6	.	.	.
T 10-19	.	.	T12	.	.	.	.	T17	.	T19
T 20-29	.	.	.	.	T24	.	.	.	.	.
Z 00-09	.	.	Z2	.	.	.	.	Z7	.	.
Z 10-19	.	.	.	.	.	.	.	.	.	Z19
Z 20-29	.	.	.	.	.	.	.	.	.	.
Z 30-39	.	.	.	.	Z34	.	.	.	.	.

Dans le présent exemple, les temporisations T1, T6, T12, T17, T19, T24 et les compteurs Z2, Z7, Z19, Z34 sont utilisés.

Ces tableaux sont classés par ordre alphabétique. Vous pouvez trier leurs entrées en cliquant sur le titre de la colonne correspondante.



### 14.1.4 Opérandes libres

Vous obtenez une vue d'ensemble de tous les mnémoniques possédant les propriétés suivantes :

- Il s'agit de mnémoniques définis dans la table des mnémoniques.
- Ces mnémoniques ne sont toutefois pas utilisés dans les parties de programme pour lesquelles des données de référence existent.

Cette liste s'affiche dans une fenêtre de travail. La barre de titre de la fenêtre de travail donne le nom du programme utilisateur auquel appartient la liste.

Chaque ligne de la fenêtre correspond à une entrée de la liste. Elle comprend l'opérande, le mnémonique, le type de données et le commentaire.

Colonne	Contenu/signification
Opérande	Adresse absolue de l'opérande
Type de données	Type de données de l'opérande.
Commentaire de l'opérande	Commentaire de l'opérande extrait de la table des mnémoniques

#### Exemple de liste des opérandes libres

Mnémonique	Opérande	Type de données	Commentaire de l'opérande
MS1	E103.6	BOOL	Disjoncteur de protection1
MS2	E120.5	BOOL	Disjoncteur de protection2
MS3	E121.3	BOOL	Disjoncteur de protection3

Vous pouvez trier leurs entrées en cliquant sur le titre de la colonne correspondante.

Vous pouvez également sélectionner des opérandes libres affichés dans la liste et les effacer avec la fonction "Effacer les mnémoniques" de la table des mnémoniques.

### 14.1.5 Mnémoniques manquants

L'affichage des mnémoniques manquants donne la liste des éléments utilisés dans le programme utilisateur S7 qui ne sont pas définis dans la table des mnémoniques. Cette liste s'affiche dans une fenêtre de travail. La barre de titre de la fenêtre de travail donne le nom du programme utilisateur auquel appartient la liste.

Elle comprend l'opérande et ses occurrences. Les entrées sont classées selon les opérandes.

#### Exemple

Opérande	Nombre
A 2.5	4
E 23.6	3
M 34.1	20

Vous pouvez sélectionner les mnémoniques manquants affichés dans la liste et leur affecter des noms de mnémoniques avec la fonction "Editer les mnémoniques".

### 14.1.6 Affichage d'informations sur le bloc pour CONT, LOG, LIST

Les informations sur le bloc pour CONT, LOG ou LIST s'affichent dans la liste des références croisées et dans la structure du programme. Elles comportent le langage du bloc et des détails.

Dans le mode d'affichage "Structure du programme", vous pouvez uniquement afficher les informations sur le bloc si dans dans l'onglet "Structure du programme" des options de filtrage, vous avez sélectionné la représentation "Structure d'appels" avec les options correspondantes.

Dans le mode d'affichage "Références croisées", vous pouvez activer ou désactiver l'affichage des informations sur le bloc en choisissant la commande **Affichage > Filtre**.

- Pour afficher les informations sur le bloc, cochez les cases "Langage du bloc" et "Détails" dans l'onglet "Références croisées" de la boîte de dialogue "Filtre".

Les informations sur le bloc varient selon le langage dans lequel le bloc a été créé et sont représentées par des abréviations.

Langage	Réseau	Instruction	Opération
LIST	Re	Inst	/
CONT	Re		
LOG	Re		

**Re**, **Inst** indiquent dans quel réseau et dans quelle instruction l'opérande est utilisé (références croisées) ou le bloc est appelé (structure du programme).

### Affichage d'informations sur le bloc pour des langages de programmation optionnels

L'aide relative à l'information sur les blocs est disponible lorsque le logiciel optionnel est installé.

## 14.2 Utilisation des données de référence

### 14.2.1 Affichage des données de référence

Pour afficher les données de référence, vous avez les possibilités suivantes :

#### Affichage dans SIMATIC Manager :

1. Dans la vue du projet hors ligne de la fenêtre du projet, sélectionnez le dossier Blocs.
2. Choisissez la commande **Outils > Données de référence > Afficher**.

#### Affichage dans la fenêtre de l'éditeur :

1. Ouvrez un bloc dans le dossier Blocs.
2. Dans la fenêtre de l'éditeur approprié, vous choisissez la commande **Outils > Données de référence**.

La boîte de dialogue "Paramètres" s'ouvre. Vous pouvez y sélectionner la première vue qui doit s'afficher. Par défaut, il s'agit de la vue fermée en dernier dans l'application permettant d'afficher les données de référence. Cette boîte de dialogue peut être ignorée pour les appels futurs.

Si les données de référence sont incomplètes, une boîte de dialogue s'ouvre, vous permettant de déclencher leur mise à jour.

#### Affichage directement depuis le bloc compilé

Vous pouvez afficher directement depuis l'éditeur de langage les données de référence d'un bloc compilé et obtenir ainsi une vue d'ensemble actuelle de votre programme utilisateur.

### 14.2.2 Affichage de listes dans des fenêtres supplémentaires

La commande **Fenêtre > Nouvelle fenêtre** permet d'ouvrir des fenêtres de travail supplémentaires pour y afficher d'autres vues des données de référence déjà affichées (par exemple la liste des opérands libres).

Vous pouvez ouvrir une fenêtre de travail pour des données de référence pas encore affichées en choisissant la commande **Données de référence > Ouvrir**.

Vous pouvez passer à une autre vue des données de référence en choisissant la commande **Affichage** ou en cliquant sur le bouton correspondant dans la barre d'outils :

Vue des données de référence	Commande pour l'affichage de cette vue des données de référence
Mnémoniques manquants	Affichage > Mnémoniques manquants
Opérands libres	Affichage > Opérands libres
Tableau d'affectation	Affichage > Tableau d'affectation
Structure du programme	Affichage > Structure du programme
Références croisées	Affichage > Références croisées

### 14.2.3 Création et affichage de données de référence

#### Création de données de référence

1. Dans SIMATIC Manager, sélectionnez le dossier Blocs pour lequel vous souhaitez générer les données de référence.
2. Dans SIMATIC Manager, choisissez la commande **Outils > Données de référence > Générer**.

Avant de générer des données de référence, le logiciel vérifie si elles sont présentes et actuelles.

- En l'absence de données de référence, celles-ci sont générées.
- Lorsque les données de référence ne sont pas actuelles, vous pouvez choisir, dans une boîte de dialogue, si vous souhaitez les actualiser ou les générer de nouveau.

#### Affichage de données de référence :

La commande **Outils > Données de référence > Afficher** vous permet d'afficher les données de référence.

Avant d'afficher des données de référence, le système vérifie si elles sont présentes et actuelles.

- En l'absence de données de référence, celles-ci sont générées.
- Si les données de référence existantes sont incomplètes, une boîte de dialogue vous informe de leur incohérence. Vous pouvez alors décider quelles données de référence vous souhaitez actualiser. Les choix suivants vous sont proposés :

Choix	Signification
pour les blocs modifiés seulement :	dans ce cas, les données de référence des blocs modifiés et des nouveaux blocs sont actualisées et les informations relatives aux blocs effacés sont supprimées des données de référence;
pour tous les blocs :	les données de référence de tous les blocs sont alors générées en totalité;
ne pas les actualiser :	les données de référence ne sont pas actualisées.

Cette mise à jour des données de référence est obtenue par une nouvelle compilation des blocs, ce pour quoi le compilateur convenant à chaque bloc est appelé. La commande **Affichage > Actualiser** permet de mettre à jour, dans la fenêtre active, des données de référence déjà affichées.

## 14.2.4 Positionnement rapide sur les occurrences dans le programme

Vous pouvez vous servir des données de référence pour vous positionner sur les occurrences d'un opérande lors de la programmation. Les données de référence doivent être actuelles. Il n'est pas nécessaire d'appeler l'application permettant d'afficher les données de référence.

### Marche à suivre

1. Dans SIMATIC Manager, choisissez la commande **Outils > Données de référence > Générer** pour créer les données de référence actuelles. Cette étape ne s'avère nécessaire que si les données de référence n'ont pas été créées ou si elles ne sont pas actuelles.
2. Sélectionnez l'opérande souhaité dans un bloc ouvert.
3. Choisissez la commande **Edition > Aller à > Occurrence**. Une boîte de dialogue s'ouvre avec la liste des occurrences de l'opérande dans le programme.
4. Si la case d'option "Pour tous les opérandes de la plage d'adresses spécifiée" est activée, le tableau affichera les occurrences de tous les opérandes de la plage d'adresses spécifiée.
5. Sélectionnez une occurrence dans la liste et cliquez sur le bouton "Aller à".

Si les données de référence ne sont pas actuelles lorsque vous appelez cette boîte de dialogue, un message vous en informe. Vous pouvez alors actualiser les données de référence.

### Liste des occurrences

La liste des occurrences dans la boîte de dialogue fournit les données suivantes :

- bloc dans lequel l'opérande est utilisé,
- mnémonique du bloc, le cas échéant,
- détails, c'est-à-dire informations fonction du langage de création du bloc/de la source (SCL) sur l'occurrence et le cas échéant sur l'opération,
- informations spécifiques au langage,
- type d'accès à l'opérande : lecture (R), écriture (W), lecture et écriture (RW), indéterminable (?),
- langage du bloc.

Vous avez la possibilité de filtrer l'affichage des occurrences, afin d'afficher uniquement les accès en écriture à un opérande, par exemple. De plus amples informations sur les possibilités de saisie et sur l'affichage sont données dans l'aide en ligne de cette boîte de dialogue.

---

### Nota

Les données de référence existent uniquement hors ligne. Cette fonction utilise donc toujours les références croisées des blocs hors ligne, même si vous appelez cette fonction dans un bloc en ligne.

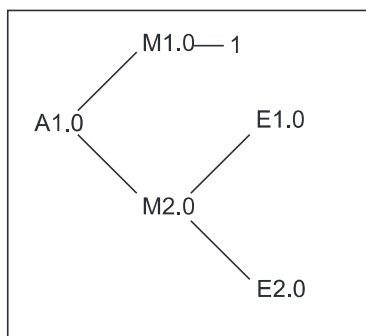
---

## 14.2.5 Exemple de recherche d'occurrences

Vous souhaitez rechercher les occurrences pour lesquelles la sortie A1.0 (directe/indirecte) est mise à 1. Comme exemple, nous allons utiliser le code LIST suivant dans l'OB 1:

```
Réseau 1: .....  
U A 1.0 // dans notre exemple,  
= A 1.1 // insignifiant  
  
Réseau 2:  
U M1.0  
U M2.0  
= A 1.0 // Affectation  
  
Réseau 3:  
//uniquement ligne de commentaire  
SET  
= M1.0 // Affectation  
  
Réseau 4:  
U E 1.0  
U E 2.0  
= M2.0 // Affectation
```

Pour A1.0, on obtient donc le schéma d'affectation suivant :



Vous procédez alors de la manière suivante :

1. Dans l'éditeur CONT/LIST/LOG, positionnez-vous sur A1.0 (Ré1, inst 1) dans l'OB 1.
2. Choisissez la commande **Edition > Aller à > Occurrence** ou cliquez sur le bouton droit de la souris pour appeler la boîte de dialogue "Aller à occurrence".  
Toutes les affectations de A1.0 sont entre autres affichées dans la boîte de dialogue :  
OB1 Cycle Execution Ré 2 Inst 3 /= W LIST  
OB1 Cycle Execution Ré 1 Inst 1 /U R LIST
3. Sélectionnez "Aller à" dans la boîte de dialogue pour sauter à "Ré 2 Inst 3" dans l'éditeur :  
Réseau 2:  
U M1.0  
U M2.0  
= A 1.0
4. Vous devez vérifier aussi bien les affectations de M1.0 que de M2.0. Positionnez-vous d'abord sur M1.0 dans l'éditeur CONT/LIST/LOG.
5. Choisissez la commande **Edition > Aller à > Occurrence** ou cliquez sur le bouton droit de la souris pour appeler la boîte de dialogue "Aller à occurrence". Celle-ci affiche entre autres toutes les affectations de M1.0 :  
OB1 Cycle Execution Ré 3 Inst 2 /= W LIST  
OB1 Cycle Execution Ré 2 Inst 1 /U R LIST
6. Sélectionnez "Aller à" pour sauter à "Ré 3 Inst 2" dans l'éditeur CONT/LIST/LOG.
7. Dans le réseau 3 de l'éditeur CONT/LIST/LOG, nous constatons que l'affectation de M1.0 ne nous intéresse pas (toujours TRUE) et que nous devons donc vérifier celle de M2.0.

**Dans les versions de STEP 7 antérieure à V5, il fallait pour cela repasser par l'ensemble de la chaîne d'affectations. Les boutons ">>" et "<<" facilitent la suite de la procédure :**

8. Amenez au premier plan la boîte de dialogue "Aller à occurrence" encore ouverte ou appelez-la à partir de la position actuelle dans l'éditeur CONT/LIST/LOG.
9. Appuyez une ou deux fois sur le bouton "<<", jusqu'à ce que toutes les occurrences de A 1.0 s'affichent, la dernière, "Ré 2 Inst 3" étant sélectionnée.
10. Sélectionnez "Aller à" (comme à l'étape 3) dans la boîte de dialogue des occurrences pour sauter à "Ré 2 Inst 3" dans l'éditeur :  
Réseau 2:  
U M1.0  
U M2.0  
= A 1.0
11. A l'étape 4 et aux suivantes, nous avons vérifié l'affectation de M1.0. Nous devons à présent vérifier toutes les affectations (directes/indirectes) de M2.0. Positionnez-vous donc sur M2.0 dans l'éditeur et appelez "Aller à occurrence" : toutes les affectations de M2.0, entre autres, s'affichent :  
OB1 Cycle Execution Ré 4 Inst 3 /= W LIST  
OB1 Cycle Execution Ré 2 Inst 2 /U R LIST
12. Sélectionnez "Aller à" pour sauter à "Ré 4 Inst 3" dans l'éditeur CONT/LIST/LOG :  
Réseau 4:  
U E 1.0  
U E 2.0  
= M2.0

13. Vous devez à présent vérifier les affectations de E1.0 et E2.0. Nous n'allons pas le faire dans cet exemple, car la procédure est similaire à celle utilisée jusqu'à présent (étape 4 et suivantes).

En commutant entre l'éditeur CONT/LIST/LOG et la boîte de dialogue des occurrences, vous pouvez ainsi déterminer et vérifier les occurrences dans votre programme.



# 15 Vérifier la cohérence des blocs et horodatage comme propriété de bloc

## 15.1 Vérifier la cohérence des blocs

### Introduction

Si vous êtes obligé d'adapter ou de compléter les interfaces ou le code de certains objets au cours de l'écriture du programme ou après, il peut en résulter des conflits d'horodatage qui à leur tour risquent d'entraîner des incohérences entre l'objet appelant et l'objet appelé ou entre blocs de référence. Ceci peut nécessiter d'importantes corrections.

La fonction "Vérifier la cohérence des blocs" simplifie cette tâche de correction. Elle supprime automatiquement la majeure partie de tous les conflits d'horodatage et des incohérences entre blocs. Dans les objets dans lesquels elle n'est pas capable de supprimer ces erreurs automatiquement, cette fonction ouvre l'éditeur correspondant et positionne le curseur à l'endroit que vous devez modifier. Vous pouvez alors y effectuer les modifications nécessaires. Vous pouvez ainsi supprimer pas à pas toutes les incohérences et compiler les objets.

### Conditions requises

La vérification de cohérence des blocs n'est possible que pour les projets créés avec la version V5.0, Servicepack 3 de STEP 7 ou V5.1. Avec les projets plus anciens, vous devez donc commencer par compiler tout (commande **Programme > Compiler tout**).

Pour les objets créés avec un logiciel optionnel, il faut que le logiciel optionnel pour vérification de cohérence soit installé.

## Démarrage de la vérification de cohérence des blocs

Au démarrage, la fonction commence par vérifier les horodatages des interfaces et met en valeur, dans la vue de l'arborescence (structure d'appel : arborescence références/dépendances), les objets susceptibles de présenter des incohérences.

1. Dans la fenêtre du projet de SIMATIC Manager, sélectionnez le dossier Blocs souhaité et démarrez la vérification de cohérence des blocs en choisissant la commande de menu **Edition > Vérification de la cohérence des blocs**.
2. Choisissez la commande **Programme > Compiler** dans la "Vérification de cohérence des blocs".  
STEP 7 reconnaît le langage de création utilisé pour les objets concernés et il ouvre l'éditeur approprié. Autant que possible, les conflits d'horodatage et incohérences sont supprimés automatiquement et les objets compilés. Ceux dont la suppression automatique n'est pas possible sont signalées par un message dans la fenêtre de résultats (il faut alors continuer à l'étape 3). Cette opération se répète pour tous les objets de la vue de l'arborescence.
3. Si toutes les incohérences entre blocs n'ont pas pu être supprimées durant la compilation, les objets correspondants sont signalés comme messages d'erreur dans la fenêtre de résultats. Positionnez le curseur sur l'entrée erronée et choisissez, avec le bouton droit de la souris, la commande **Afficher les erreurs** dans le menu contextuel. L'éditeur approprié s'ouvre alors et saute à la position à modifier. Éliminez toutes les incohérences, puis fermez l'objet et enregistrez-le. Répétez cette opération pour tous les objets signalés comme erreurs.
4. Répétez les étapes 2 et 3 jusqu'à ce qu'aucune erreur ne soit plus signalée dans la fenêtre de résultats.

## 15.2 Horodatage comme propriété de bloc et conflits d'horodatage

Les blocs contiennent un horodatage du code et un horodatage des interfaces. Ces horodatages s'affichent dans la boîte de dialogue des propriétés de bloc. Ils permettent de vérifier la cohérence des programmes STEP 7.

STEP 7 signale un conflit d'horodatage lorsque l'un des manquements à la règle suivants est détecté lors de la comparaison d'horodatages.

- Un bloc appelé est plus récent que le bloc appelant (CALL).
- Un bloc référencé est plus récent que le bloc qui l'utilise.
- Exemples relatifs au second point :
- Un UDT est plus récent que le bloc qui l'utilise, p. ex. un DB ou un autre UDT, ou un FC, FB, OB qui utilise cet UDT dans la table de déclaration des variables.
- Un FB est plus récent que le DB d'instance correspondant.
- Un FB 2 est défini comme multi-instance dans un FB 1 et FB 2 est plus récent que FB 1.

---

### Nota

Des incohérences sont également susceptibles de se produire, même lorsque la relation entre les horodatages d'interfaces est correcte :

- La définition de l'interface du bloc référencé ne correspond pas à l'interface utilisée à son occurrence.

De telles incohérences sont appelées conflits d'interface. Ils peuvent par exemple résulter de la copie de blocs de programmes différents ou de la compilation d'une source ASCII lors de laquelle seule une partie des blocs d'un programme complet est créée.

---

## 15.3 Horodatage dans les blocs de code

### Horodatage du code

L'instant de création du bloc y est notifié. L'horodatage est actualisé en cas de

- modification de code du programme
- modification de la description d'interfaces
- modification du commentaire
- génération et de compilation d'une source ASCII
- modification des propriétés de bloc (boîte de dialogue : Propriétés)

### Horodatage des interfaces

L'horodatage est actualisé en cas de

- modification de la description d'interfaces (modification de types de données ou de valeurs initiales, nouveaux paramètres)
- génération et compilation d'une source ASCII, si la structure de l'interface change.

L'horodatage n'est pas actualisé en cas de :

- modification de mnémoniques
- modification de commentaires dans la déclaration de variables
- modification dans la zone TEMP.

### Règles pour l'appel de blocs

- L'horodatage des interfaces du bloc appelé doit être antérieur à l'horodatage du code du bloc appelant.
- Ne modifiez l'interface d'un bloc que si aucun bloc qui appelle celui-ci n'est ouvert. En effet, si vous enregistrez les blocs appelant après le bloc modifié, cette incohérence ne pourra pas être détectée par l'horodatage.

### En cas d'erreur d'horodatage

Un conflit d'horodatage est signalé à l'ouverture du bloc appelant. Après modification d'une interface de FC ou de FB, tous les appels de ce bloc seront représentés sous forme étendue dans les blocs appelant.

Si vous modifiez l'interface d'un bloc, vous devez adapter tous les blocs qui appellent ce bloc.

Après modification d'une interface de FB, vous devez actualiser les définitions de multi-instances et les blocs de données existant.

## 15.4 Horodatage dans les blocs de données globaux

### Horodatage du code

L'horodatage est actualisé en cas de

- génération,
- compilation d'une source ASCII,
- modification dans la vue des déclarations ou dans la vue des données du bloc.

### Horodatage des interfaces

L'horodatage est actualisé en cas de

- modification de la description des interfaces dans la vue des déclarations (modification de types de données ou de valeurs initiales, nouveaux paramètres)

## 15.5 Horodatage dans les blocs de données d'instance

Un bloc de données d'instance enregistre les paramètres formels et les données statiques de blocs fonctionnels.

### Horodatage du code

L'instance de création du bloc de données d'instance y est notifié. L'horodatage est actualisé lorsque vous saisissez des valeurs effectives dans la vue des données du bloc de données d'instance. L'utilisateur ne peut pas modifier la structure d'un bloc de données d'instance ; en effet, sa structure est reprise du bloc fonctionnel (FB) ou du bloc fonctionnel système (SFB) correspondant.

### Horodatage des interfaces

Lors de la création d'un bloc de données d'instance, l'horodatage des interfaces du FB ou du SFB correspondant y est notifié.

### Règles pour une ouverture exempte de conflits

L'horodatage des interfaces du FB/SFB et celui du bloc de données d'instance correspondant doivent concorder.

### En cas de conflit d'horodatage

Lorsque vous modifiez l'interface d'un FB, l'horodatage des interfaces de ce FB est actualisé. A l'ouverture d'un bloc de données d'instance correspondant, un conflit d'horodatage est signalé, puisque les horodatages du bloc de données d'instance et du FB ne concordent plus. L'interface est représentée avec les mnémoniques générés par le compilateur (pseudo-mnémoniques) dans la section de déclaration du DB. Le bloc de données d'instance peut uniquement être visualisé.

Afin de résoudre de tels conflits d'horodatage, vous devez créer une nouvelle fois le DB d'instance appartenant à un FB modifié.

## 15.6 Horodatage dans les UDT et DB repris d'UDT

Vous pouvez utiliser des types de données utilisateur (UDT), par exemple pour créer plusieurs blocs de données de structure identique.

### Horodatage du code

L'horodatage du code est actualisé à chaque modification.

### Horodatage des interfaces

L'horodatage des interfaces est actualisé lors de la modification de la description des interfaces (modification de types de données ou de valeurs initiales, nouveaux paramètres).

L'horodatage des interfaces d'un UDT est également actualisé lors de la compilation de la source ASCII.

### Règles pour une ouverture exempte de conflits

- L'horodatage des interfaces du type de données utilisateur doit être antérieur à celui des blocs de code dans lequel ce type de données est utilisé.
- L'horodatage des interfaces du type de données utilisateur doit être identique à l'horodatage d'un DB repris d'un UDT.
- L'horodatage des interfaces du type de données utilisateur doit être postérieur à l'horodatage d'un UDT qui y est contenu.

### En cas de conflit d'horodatage

Lorsque vous modifiez une définition d'UDT utilisée dans un DB, une FC, un FB ou une autre définition d'UDT, STEP 7 signale un conflit d'horodatage à l'ouverture d'un tel bloc.

Le composant UDT est représenté non assemblé, sous forme de structure. Tous les noms de variables sont remplacés par des valeurs par défaut du système.

## 15.7 Correction des interfaces dans une FC, un FB ou un UDT

Si vous avez à corriger l'interface dans un FB, une FC ou un UDT, procédez de la manière suivante afin d'éviter des conflits d'horodatage :

1. Générez une source LIST avec le bloc à modifier ainsi qu'avec tous les blocs qui s'y réfèrent de manière directe ou indirecte.
2. Enregistrez les modifications dans la source générée.
3. Recompilez la source modifiée en blocs.

Vous pouvez à présent enregistrer/charger la modification d'interface.

## 15.8 Comment éviter des erreurs lors de l'appel de blocs

### STEP 7 écrase des données dans le registre DB

STEP 7 modifie les registres des CPU S7-300/S7-400 lors de certaines opérations. Les contenus des registres DB et DI sont, par exemple, permutés lors de l'appel d'un FB : cela permet d'ouvrir le DB d'instance du FB appelé sans perdre l'adresse du DB d'instance précédent.

En adressage absolu, des erreurs peuvent se produire lors de l'accès à des données figurant dans les registres : dans certains cas, les adresses dans le registre d'adresse1 (AR1) et dans le registre de DB sont écrasées. Il se peut donc que vous lisiez des adresses erronées ou que vous écriviez à des adresses erronées.



#### Danger

Il existe un risque de dégâts matériels et de dommages physiques lorsque vous utilisez :

1. CALL FC, CALL FB, CALL multi-instance,
2. des accès à un DB indiqués intégralement (par exemple, DB20.DBW10),
3. des accès à des variables de type de données complexe,

Il est possible que les contenus des registres de bloc de données (DB et DI), des registres d'adresse (AR1, AR2) et des accumulateurs (ACCU1 et ACCU2) soient modifiés.

En outre, il n'est pas possible d'utiliser le résultat logique RLG comme paramètre supplémentaire (implicite) lors de l'appel d'une fonction ou d'un bloc fonctionnel.

Si vous utilisez les méthodes de programmation ci-dessus, vous devez vous-même faire en sorte que ces contenus soient corrects afin d'éviter tout dysfonctionnement.

---

### Sauvegarde de données correctes

Le contenu du registre de DB s'avère tout particulièrement important lorsque vous accédez à des données en format abrégé de l'adresse absolue. Si, par exemple, vous partez du principe que le DB20 est ouvert (son numéro est donc enregistré dans le registre DB), vous pouvez indiquer DBX0.2 pour accéder aux données figurant dans le bit 2 de l'octet 0 du DB dont l'adresse figure dans le registre DB, donc le DB20. Toutefois, si le registre DB contient une autre adresse, vous accédez à des données erronées.

Pour éviter toute erreur lors d'accès aux données du registre DB, nous vous conseillons :

- d'utiliser l'adressage symbolique,
- de donner l'adresse absolue complète (par exemple, *DB20.DBX0.*).

Avec ces deux méthodes d'adressage, STEP 7 ouvre automatiquement le bon DB. Si vous utilisez le registre AR1 pour l'adressage indirect, vous devez toujours charger l'adresse correcte dans AR1.

### Situation dans lesquelles les registres sont modifiés

La manipulation des registres d'adresse pour l'adressage indirect ne concerne que le langage LIST. Les autres langages n'autorisent pas l'accès indirect aux registres d'adresse.

En revanche, il faut tenir compte de la modification du registre du DB par le compilateur dans tous les langages de programmation afin de garantir une transmission correcte des paramètres lors d'appels de blocs.

Le contenu du registre d'adresse AR1 et du registre de DB du bloc appelant est écrasé dans les situations suivantes :

Situation	Signification
Paramètres effectifs provenant d'un DB	<ul style="list-style-type: none"><li>• Une fois que vous avez affecté à un bloc un paramètre effectif qui est sauvegardé dans un bloc de données (par exemple, DB20.DBX0.2), STEP 7 ouvre ce bloc de données (DB20) et modifie le contenu du registre de DB en conséquence. Après l'appel de bloc, le programme utilise alors le DB modifié.</li></ul>
Appel de bloc en relation avec des types de données complexes	<ul style="list-style-type: none"><li>• Le contenu du registre AR1 et du registre de DB du bloc appelant est modifié après un appel de bloc dans une FC qui transmet un composant d'un paramètre formel de type de données complexe (chaîne, tableau, structure ou UDT) au bloc appelé.</li><li>• Il en est de même lors d'un appel dans un FB si le paramètre se situe dans la zone VAR_IN_OUT du bloc appelant.</li></ul>
Accès à des composants de type de données complexe	<ul style="list-style-type: none"><li>• STEP7 utilise le registre d'adresse AR1 et le registre de DB lors de l'accès d'un FB à un composant d'un paramètre formel de type de données complexe dans la zone VAR_IN_OUT (chaîne, tableau, structure ou UDT). Cela entraîne donc la modification du contenu de ces deux registres.</li><li>• STEP7 utilise le registre d'adresse AR1 et le registre de DB lors de l'accès d'une FC à un composant d'un paramètre formel de type de données complexe (chaîne, tableau, structure ou UDT). Cela entraîne donc la modification du contenu de ces deux registres.</li></ul>

---

#### Nota

- Lors de l'appel d'un FB dans un bloc de version 1, le paramètre effectif pour le premier paramètre booléen IN ou IN\_OUT n'est pas transmis correctement si l'opération avant l'appel ne délimite pas les RLG. Dans ce cas, ce paramètre est combiné au RLG existant.
  - Il y a écriture dans le registre d'adresse AR2 lors de l'appel d'un FB (simple ou multi-instance).
  - Le traitement correct d'un FB n'est plus garanti si le registre d'adresse AR2 est modifié à l'intérieur de ce FB, p. ex. par les opérations UC, CC ou CALL (appel de FC/SFC sans paramètres).
  - Si l'adresse absolue du DB n'est pas transmise en entier à un paramètre ANY, le pointeur ANY ne contient pas le numéro du DB ouvert, mais toujours le numéro 0.
-



# 16 Configuration de messages

## 16.1 Concept de signalisation

Les messages vous permettent de détecter rapidement, de localiser avec précision et de corriger les erreurs d'exécution du processus dans les automates programmables. Les temps d'immobilisation de votre installation s'en trouvent considérablement réduits.

Avant que les messages ne puissent s'afficher, ils doivent être configurés.

STEP 7 vous permet de créer, d'éditer, de compiler et d'afficher des messages qui sont fonction d'événements sur des visuels, avec les textes et attributs correspondants.

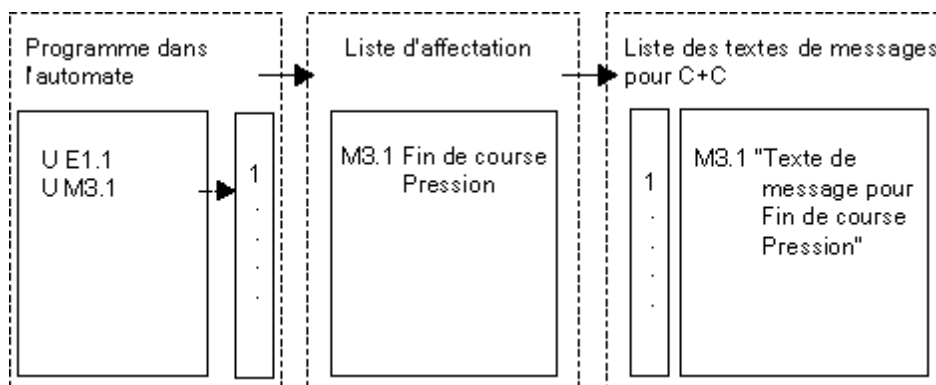
### 16.1.1 Quels procédés de signalisation existe-t-il ?

Il existe différents procédés de création de messages.

#### Procédé de signalisation par bit

Pour le procédé de signalisation par bit, le programmeur doit réaliser 3 étapes :

- Il crée le programme utilisateur sur la PG et met le bit souhaité à 1.
- Il crée une liste d'affectation dans un éditeur de texte quelconque, dans laquelle il affecte un texte au bit de signalisation (par exemple B. M 3.1 = Commutateur de fin de course Pression).
- Dans le système de commande, il crée la liste des textes de message sur la base de la liste d'affectation.

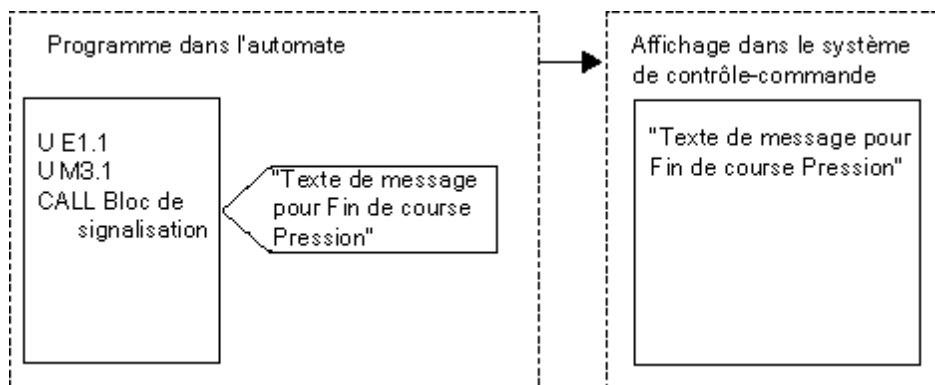


Le système de contrôle-commande interroge cycliquement l'automate programmable pour vérifier si le bit de signalisation a été modifié. Si l'automate programmable signale une modification, le message correspondant s'affiche. Ce message comporte l'horodatage du système de contrôle-commande.

## Procédé de numéro de message

Pour le procédé de numéro de message, le programmeur ne doit réaliser qu'une étape :

- Il crée le programme utilisateur sur la PG, met le bit souhaité à 1 et affecte immédiatement lors de la programmation le texte souhaité au bit.



Il n'y a pas d'interrogation cyclique de l'automate programmable. Aussitôt que ce dernier signale une modification, le numéro de message correspondant est transmis au système de contrôle-commande et le message correspondant s'affiche. Le message comporte l'horodatage de l'automate programmable et peut de ce fait être affecté avec plus de précision que dans le cas du procédé de signalisation par bit.

## 16.1.2 Sélection du procédé de signalisation

### Généralités

Le tableau suivant précise les caractéristiques et conditions additionnelles des différents procédés de signalisation.

Procédé de numéro de message	Procédé de signalisation par bit
<ul style="list-style-type: none"> <li>Les messages sont gérés dans une base de données commune à la PG et au système de commande.</li> <li>La charge pour le bus est faible (l'AP est signalé actif).</li> <li>Les messages reçoivent l'horodatage de l'automate programmable.</li> </ul>	<ul style="list-style-type: none"> <li>Il n'y a pas de base de données commune à la PG et au système de commande.</li> <li>La charge pour le bus est élevée (le système de commande interroge).</li> <li>Les messages reçoivent l'horodatage du système de commande.</li> </ul>

Le procédé de numéro de message englobe les trois types de messages suivants :

Messages sur bloc	Messages sur mnémorique	Messages de diagnostic personnalisés
<ul style="list-style-type: none"> <li>Synchrone avec le programme</li> <li>Affichage via ProTool (uniquement ALARM_S) et WinCC</li> <li>Possible pour S7-300/400</li> <li>Programmation à l'aide de blocs de signalisation : <ul style="list-style-type: none"> <li>- ALARM</li> <li>- ALARM_8</li> <li>- ALARM_8P</li> <li>- NOTIFY</li> <li>- ALARM_S(Q)</li> <li>- AR_SEND</li> <li>- ALARM_D(Q)</li> </ul> </li> <li>Transmission au système de commande <ul style="list-style-type: none"> <li>- pour WinCC, via AS-OS-Engineering</li> <li>- pour ProTool via les fonctions ProTool</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>Asynchrone avec le programme</li> <li>Affichage via WinCC</li> <li>Possible seulement pour S7-400</li> <li>Configuration à l'aide de la table des mnémoriques</li> <li>Transmission à l'AP à l'aide de blocs de données système (SDB)</li> <li>Transmission au système de commande à l'aide de AS-OS-Engineering</li> </ul>	<ul style="list-style-type: none"> <li>Synchrone avec le programme</li> <li>Affichage dans la mémoire tampon de diagnostic sur la PG</li> <li>Possible pour S7-300/400</li> <li>Programmation à l'aide de blocs de signalisation (fonction système) <ul style="list-style-type: none"> <li>- WR_USMSG</li> </ul> </li> <li>Pas de transmission au système de commande</li> </ul>

STEP 7 utilise uniquement le procédé le plus confortable, le procédé de numéro de message que nous allons décrire en détail ci-après. Le procédé de messages binaires est configuré dans les appareils IHM et y est décrit.

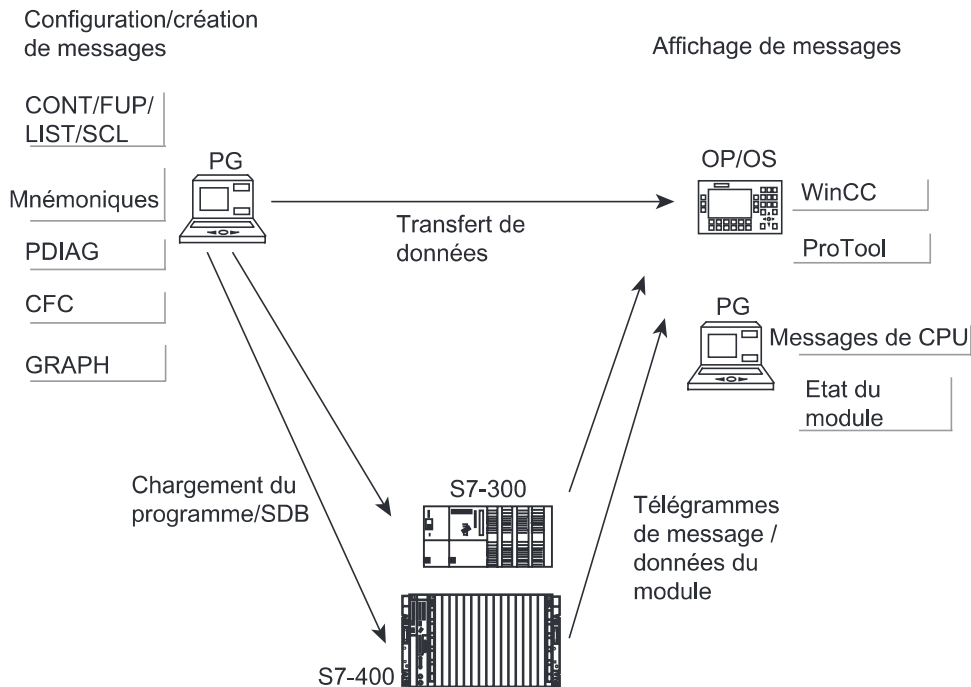
### Exemples du procédé de numéro de message

<b>Procédé de signalisation</b>	<b>Domaine d'application</b>
Messages sur bloc	Pour signaler des événements synchrones avec le programme, il s'agit par exemple d'indiquer qu'un régulateur a atteint une valeur limite
Messages sur mnémonique	Pour signaler des événements indépendants du programme, il s'agit par exemple de surveiller la position d'un commutateur
Messages personnalisés	Pour signaler des événements de diagnostic dans la mémoire tampon de diagnostic, à chaque appel de la SFC

### 16.1.3 Composants SIMATIC

#### Généralités

La figure suivante donne une vue d'ensemble des composants SIMATIC participant à la configuration et à l'affichage de messages.



### 16.1.4 Eléments constitutants d'un message

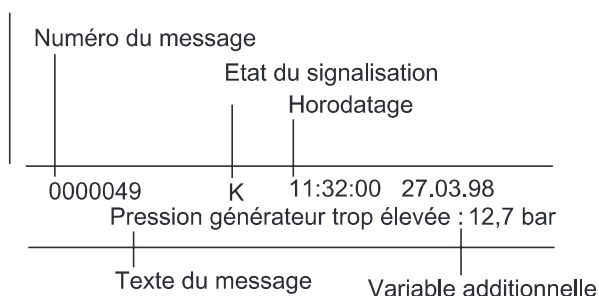
La manière de laquelle un message s'affiche dépend du procédé de signalisation, du bloc de signalisation utilisé et du visuel.

Le tableau suivant contient la liste des éléments constitutants possibles :

Elément constituant	Description
Horodatage	Créé dans l'automate programmable à l'apparition de l'événement de signalisation
Etat de signalisation	Possibilités : arrivant, partant, partant sans acquittement, partant avec acquittement
Variable	Il est possible d'ajouter à certains messages une valeur de processus pouvant être exploitée par le bloc de signalisation utilisé.
Image	En cas de plantage du système, les messages arrivants peuvent être affichés après coup sur l'OS.
Numéro du message	Numéro univoque dans l'ensemble du projet ou de la CPU, attribué par le système et qui identifie un message.
Texte du message	Configurés par l'utilisateur

## Exemple

L'exemple suivant montre un message d'alarme sur un pupitre opérateur (Operator Panel).



### 16.1.5 Quels blocs de signalisation existe-t-il ?

Vous disposez des blocs de signalisation suivants, dans lesquels une fonction de signalisation est déjà programmée :

- SFB 33 : "ALARM"
- SFB 34 : "ALARM\_8"
- SFB 35: "ALARM\_8P"
- SFB 36 : "NOTIFY"
- SFC 18 : "ALARM\_S" et SFC 17 : "ALARM\_SQ"
- SFB 37 : "AR\_SEND" (pour envoyer des archives ; configuration de textes et d'attributs de signalisation impossible)
- SFB 31 : "NOTIFY\_8P"
- SFC 107 : "ALARM\_DQ"
- SFC 108 : "ALARM\_D"

De plus amples informations à ce sujet sont données dans l'aide de référence sur les blocs .

### Quand utiliser quel bloc de signalisation ?

Le tableau ci-après vous aidera à choisir le bloc de signalisation convenant à votre cas. Ce choix est guidé par :

- le nombre de voies disponibles dans le bloc et donc le nombre de signaux surveillés par appel de bloc,
- la possibilité d'acquitter des messages,
- la possibilité d'accompagner ceux-ci de variables,
- les visuels mis en oeuvre,
- la capacité de votre CPU.

Bloc de signalisation	Voies	Acquittement	Variable	Affichage WinCC	Affichage PRO-TOOL	Affichage mess. de CPU/état S7	AP	Particularités
ALARM SFB 33	1	possible	10 au plus	oui	non	non	S7-400	Emet un message à chaque front arrivant ou partant
ALARM_8 SFB 34	8	possible	non	oui	non	non	S7-400	Emet un message à chaque front arrivant ou partant d'un ou de plusieurs signaux
ALARM_8P SFB 35	8	possible	10 au plus	oui	non	non	S7-400	Comme ALARM_8
NOTIFY SFB 36	1	non	10 au plus	oui	non	non	S7-400	Comme ALARM
NOTIFY_8P SFB 31	8	non	10 au plus	oui	non	non	S7-400	Comme NOTIFY
AR_SEND SFB 37	1	-	-	oui	non	non	S7-400	Sert à envoyer des archives ; configuration de textes et d'attributs de signalisation impossible
ALARM_SQ SFC 17	1	possible	1	oui	oui*	oui	S7-300/400	Le message n'est pas engendré par un changement de front, mais à chaque appel de la SFC
ALARM_S SFC 18	1	non	1	oui	oui*	oui	S7-300/400	Comme ALARM_SQ
ALARM_DQ SFC 107	1	possible	1	oui	oui *	oui	S7-300/400	Comme ALARM_SQ
ALARM_D SFC 108	1	non	1	oui	oui *	oui	S7-300/400	Comme ALARM_SQ
* en fonction du type d'OP								

## 16.1.6 Paramètres formels, attributs système et blocs de signalisation

### Paramètre formel comme entrée de numéro de message

Pour chaque message ou groupe de messages, vous avez besoin, dans votre programme, d'un paramètre formel que vous indiquez en tant que paramètre IN dans la table de déclaration des variables de votre programme. Ce paramètre formel est utilisé comme entrée de numéro de message et constitue la base d'un message.

### Valorisation des paramètres formels avec des attributs système

Pour passer dans la configuration des messages, il faut d'abord que vous ayez valorisé les paramètres formels avec des attributs système.

1. Vous avez ajouté les attributs système suivants pour les paramètres : "S7\_server" et "S7\_a\_type".
2. Vous leur avez donné des valeurs convenant aux blocs de signalisation que vous avez appelés dans votre code de programme: pour s7\_server, c'est toujours alarm\_archiv, pour s7\_a\_type, elle dépend du bloc de signalisation appelé.

### Attributs système et blocs de signalisation correspondants

Les objets qui s'affichent dans le serveur de messages ne sont pas les blocs de signalisation à proprement parler, mais les valeurs correspondantes de l'attribut S7\_a\_type. Ces valeurs portent le même nom que les blocs de signalisation existant en tant que SFB ou SFC (exception : alarm\_s)

S7_a_type	Bloc de signalisation	Désignation	Propriétés
alarm_8	ALARM_8	SFB 34	8 voies, acquittement possible, pas de variable additionnelle
alarm_8p	ALARM_8P	SFB 35	8 voies, acquittement possible, jusqu'à 10 variables additionnelles par voie
notify	NOTIFY	SFB 36	1 voie, pas d'acquittement, jusqu'à 10 variables additionnelles
alarm	ALARM	SFB 33	1 voie, acquittement possible, jusqu'à 10 variables additionnelles
alarm_s	ALARM_S	SFC 18	1 voie, pas d'acquittement, jusqu'à 1 variable additionnelle
alarm_s	ALARM_SQ	SFC 17	1 voie, acquittement possible, jusqu'à 1 variable additionnelle
ar_send	AR_SEND	SFB 37	sert à envoyer des données d'archives
notify_8p	NOTIFY_8P	SFB 31	8 voies, pas d'acquittement, jusqu'à 10 variables additionnelles
alarm_s	ALARM_DQ	SFC 107	1 voie, acquittement possible, jusqu'à 1 variable additionnelle
alarm_s	ALARM_D	SFC 108	1 voie, pas d'acquittement, jusqu'à 1 variable additionnelle

De plus amples informations à ce sujet sont données dans l'aide de référence sur les attributs système

Les attributs système sont affectés automatiquement lorsque les blocs de signalisation que vous utilisez dans votre programme sont des SFB ou FB avec des attributs système correspondants et lorsque vous les appelez comme multi-instances.



## 16.1.7 Modèle de message et messages

La configuration des messages vous permet de créer, par des opérations différentes, soit un modèle de message, soit des messages. Ceci dépend du bloc à fonctions de signalisation par lequel vous accédez à la configuration des messages.

### Ce bloc ayant des fonctions de signalisation peut être un FB ou un DB d'instance.

- Si c'est un FB, vous pouvez créer un modèle de message pour les messages. Toutes les entrées que vous effectuez pour le modèle de message seront automatiquement reprises dans les messages. Si vous affectez au FB un DB d'instance, des messages seront générés automatiquement sur ce modèle pour le DB d'instance, et des numéros de message leur seront attribués.
- Si c'est un DB d'instance, vous pouvez modifier, pour chaque instance, les messages générés à partir du modèle de message.

La différence visible, c'est que des numéros sont attribués aux messages mais pas au modèle de message.

### Verrouillage des données dans le modèle de message

La configuration des messages sert à saisir des textes et des attributs pour des messages déclenchés par événement. Ce faisant, vous pouvez, par exemple, définir l'aspect des messages sur certains visuels. Pour faciliter la création des messages, il faut créer d'abord des modèles de message.

- En saisissant les données (attributs et textes) pour le modèle de message, vous pouvez décider de les verrouiller ou pas. Quand les attributs sont verrouillés, un symbole de clé figure à côté de la zone de saisie ou ils sont cochés dans la colonne "Verrouillé". Les textes verrouillés sont cochés dans la colonne "Verrouillé".
- Dans le modèle de message données verrouillées, vous ne pouvez plus modifier les messages propres aux instances. Elles seront seulement affichées.
- Si vous avez pourtant besoin de les modifier, vous devrez revenir au modèle de message pour y annuler le verrouillage et effectuer les modifications. Toutefois, ces modifications ne s'appliquent pas aux instances qui ont été générées avant la modification.

### Modification des données dans le modèle de message

Selon que lors de la création du projet, vous avez attribué les numéros de message à tout le projet ou à la CPU, les modifications des données dans dans le modèle de message s'appliquent aux instances ou pas.

- Attribution des numéros de message pour tout le projet : si vous souhaitez modifier ultérieurement des données dans le modèle de message et que ces modifications doivent aussi s'appliquer aux instances, alors vous devez également effectuer ces modifications dans les instances.
- Attribution de numéros de message pour la CPU : si vous modifiez ultérieurement des données dans le modèle de message, elles s'appliqueront automatiquement aux instances.

**Exceptions :** vous avez précédemment modifié ces données dans les instances ou les avez verrouillées ou déverrouillées ultérieurement dans le modèle de message. Lorsque vous copiez un FB et un DB d'instance depuis un projet avec attribution des numéros de message pour le projet dans un projet avec attribution des numéros de message pour la

CPU, vous devez également réaliser dans l'instance, les modifications des données que vous avez précédemment faites dans le type de message.

---

**Important**

- Si vous copiez des instances dans un autre programme, sans également copier le modèle de message, l'instance risque de ne pas être complète. Dans ce cas, copiez le modèle de message dans le nouveau programme.
  - Si des textes et attributs sont représentés en vert dans un instance, cela signifie que ces textes et attributs sont encore tels qu'ils ont été configurés dans le modèle. Entre-temps, ils n'ont pas été modifiés dans l'instance.
- 

### 16.1.8 Création d'une source LIST à partir de blocs de signalisation

Lorsque vous créez une source LIST à partir de blocs de signalisation, les informations projet sont également intégrées à cette source.

L'information est inscrite dans un pseudo-commentaire commençant par "\$ALARM\_SERVER" et se terminant par "".

---

**Important**

Lorsque vous référencez un bloc de manière symbolique, veillez à ne pas modifier la table des mnémoniques avant de réaliser la compilation de la source.

---

Lorsque la source comporte plusieurs blocs, plusieurs blocs de pseudo-commentaires sont regroupés en un bloc de commentaires. Vous ne pouvez pas supprimer des blocs individuels possédant des attributs de message dans une source LIST.

### 16.1.9 Attribution de numéros de message

Vous pouvez décider d'attribuer les numéros de message pour tout le projet ou pour la CPU. L'avantage de l'attribution des numéros de message pour la CPU est que vous pouvez copier un programme sans que les numéros de messages ne changent et doivent être recompilés. Il n'est possible d'afficher les numéros de messages pour la CPU sur un appareil IHM qu'avec "WinCC V6.0" et/ou "ProTool V6.0". Si vous travaillez avec des versions antérieures à la version V6.0 de ces applications, choisissez les numéros de message pour le projet.

### 16.1.10 Différences entre l'attribution de numéros de message pour tout le projet et celle pour la CPU

Le tableau suivant indique les différences entre l'attribution de numéros de message pour l'ensemble du projet et celle pour la CPU :

Tout le projet	CPU
Certains attributs et textes de message dépendent de l'appareil IHM mis en œuvre et doivent être configurés en fonction du visuel.	Les attributs et textes affectés sont indépendants de l'appareil IHM mis en œuvre, c'est-à-dire que vous ne devez plus insérer de visuel et configurer un message spécifique au visuel pour cet appareil.
Lors de la copie de programmes, une nouvelle compilation doit être réalisée.	Les programmes peuvent être copiés au sein d'un projet et d'un projet vers un autre. Lors de la copie de blocs individuels, une nouvelle compilation est cependant nécessaire.
Lorsque vous modifiez ultérieurement des données (textes et attributs) dans le modèle de message, vous devez reporter ces modifications dans les instances.	Lorsque vous modifiez ultérieurement des données (textes et attributs) dans le modèle de message, ces modifications sont reprises automatiquement dans les instances (exception : vous avez déjà modifié ces données dans les instances).
Vous pouvez uniquement saisir des textes à une ligne.	Vous pouvez également saisir des textes à plusieurs lignes.

### 16.1.11 Possibilités de modification de l'attribution des numéros de message d'un projet

Le mode d'attribution des numéros de message peut être présélectionné pour les futurs nouveaux projets et bibliothèques dans la page d'onglet "Numéros de message" (commande de menu **Outils > Paramètres**) de SIMATIC Manager. Dans cette page d'onglet, vous indiquez si les numéros de message doivent être attribués de manière univoque pour la CPU ou pour le projet. Vous pouvez également sélectionner "Demande préalable" pour réaliser l'attribution ultérieurement.

Si au moment de la création d'un projet ou d'une bibliothèque, la présélection "Numéros de message univoques pour la CPU" ou "Numéros de message univoques pour le projet" est activée, ce mode d'attribution des numéros de message ne peut plus être modifié dans ce projet ou cette bibliothèque.

Si l'attribution des numéros de message "Numéros de message univoques pour le projet" a été paramétrée et que vous souhaitez une attribution des "Numéros de message univoques pour la CPU", procédez de la manière suivante :

1. Dans SIMATIC Manager, sélectionnez le projet ou la bibliothèque souhaités.
2. Choisissez la commande de menu **Fichier > Enregistrer sous**.
3. Dans la boîte de dialogue qui s'ouvre, activez la case à cocher "Avec réorganisation" et entrez un nouveau nom.
4. Démarrez la procédure "Enregistrez sous" en cliquant sur le bouton "OK".
5. Dans l'une des boîtes de dialogue suivantes, vous pouvez définir l'attribution des numéros de message "Numéros de message univoques pour la CPU".

La commande de menu **Fichier > Supprimer** vous permet de supprimer le projet initial ou la bibliothèque initiale.

## 16.2 Configuration de messages pour tout le projet

### 16.2.1 Attribution de numéros de message pour tout le projet

Les messages sont identifiés par un numéro univoque dans l'ensemble du projet. Une plage de numéros appartenant à la plage totale disponible (1 à 2097151) est à cet effet attribuée à chaque programme S7. En cas de conflit - lorsque vous copiez un programme et que des numéros de messages identiques ont déjà été attribués dans la plage cible -, vous devez attribuer une autre plage de numéros au nouveau programme. Dans un tel cas, STEP 7 ouvre automatiquement la boîte de dialogue dans laquelle vous pouvez attribuer la nouvelle plage de numéros.

La commande **Edition > Propriétés spécifiques de l'objet > Numéros de messages** vous permet, lorsqu'aucun message n'a été configuré dans le programme, en outre de définir ou de modifier la plage de numéros pour un programme S7.

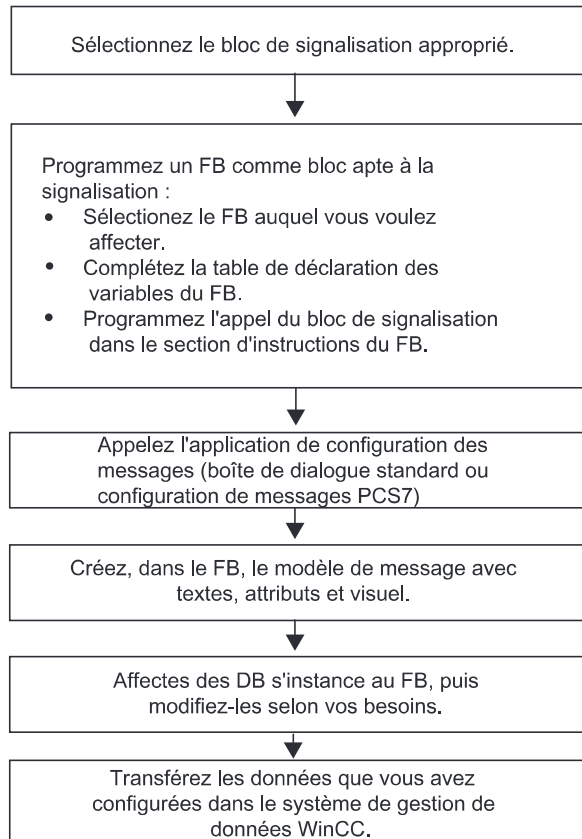
Par défaut, les plages de numéros de message sont attribuées par tranche de 20 000.

### 16.2.2 Affectation et édition de messages sur bloc

Les messages sur bloc sont affectés à un bloc (DB d'instance). Pour créer un message sur bloc, vous pouvez utiliser des blocs fonctionnels système (SFB) et des fonctions système (SFC) comme blocs de signalisation.

### 16.2.2.1 Création de messages sur bloc (pour tout le projet)

#### Marche à suivre



#### Programmation d'un bloc apte à la signalisation (FB)

1. Dans SIMATIC Manager, sélectionnez le bloc fonctionnel (FB) pour lequel vous souhaitez créer un message et ouvrez-le par double clic.  
**Résultat** : le bloc sélectionné s'ouvre et s'affiche dans la fenêtre "CONT/LOG/LIST".
2. Complétez la table de déclaration des variables. Pour chaque bloc de signalisation appelé dans le FB, vous devez déclarer des variables dans le FB appelant.

Pour ce faire, entrez les variables suivantes dans la vue des variables :

- pour le paramètre "IN", un mnémonique pour l'entrée du bloc de signalisation, par exemple "Mess01" (pour l'entrée du message 01) ainsi que le type de données correspondant (il doit s'agir de "DWORD" sans valeur initiale).
  - pour le paramètre "STAT", un mnémonique pour le bloc de signalisation à appeler, par exemple "alarme" ainsi que le type de données correspondant, en l'occurrence "SFB33".
3. Dans la section des instructions du FB, insérez l'appel du bloc de signalisation sélectionné, dans notre exemple "CALL alarme", puis validez votre saisie par la touche ENTREE.  
**Résultat** : les variables d'entrée du bloc de signalisation appelé, dans notre exemple le SFB33, s'affichent dans la section des instructions du FB.

- Affectez à la variable "EV\_ID" le mnémonique que vous aviez affecté à l'étape 2 à l'entrée du bloc de signalisation, dans notre cas "Mess01". A présent les attributs système sont repris pour le modèle de message "alarme" (pour certaines SFC, vous devez affecter vous-même les attributs système pour le paramètre "IN" (commande de menu **Edition > Propriétés de l'objet**, onglet "Attributs"))).

**Résultat** : si la colonne "Nom" n'est pas sélectionnée, un "drapeau" y apparaît pour le paramètre "IN". Le bloc sélectionné acquiert ainsi des fonctions de signalisation. Les attributs système requis (par exemple S7\_server et S7\_a\_type) ainsi que les valeurs correspondantes sont affectés automatiquement.

**Important** : si au lieu d'un SFB, vous appelez un FB avec des multi-instances et dans lequel des messages sont également configurés, vous devez alors configurer les messages du FB avec des multi-instances dans le bloc appelant.

- Renouvelez les étapes 2 à 4 pour tous les appels de blocs de signalisation dans ce FB.
- Enregistrez le bloc en choisissant la commande **Fichier > Enregistrer**.
- Fermez la fenêtre "CONT, LIST, LOG".

### Appel de l'application de configuration des messages

- Sélectionnez le bloc de signalisation souhaité, puis dans SIMATIC Manager, choisissez la commande **Edition > Propriétés spécifiques de l'objet > Signalisation**.

**Résultat** : la boîte de dialogue de la configuration des messages de STEP 7 (boîte de dialogue par défaut) s'ouvre. Pour savoir comment appelez la configuration des messages PCS7, reportez vous à Configuration des messages PCS7.

### Edition d'un modèle de message

- Sélectionnez le bloc de signalisation voulu, ouvrez la configuration des messages, puis saisissez le texte souhaités dans les pages d'onglet "Texte" et "Attributs" ou sélectionnez les attributs de message souhaités.  
Si vous avez sélectionné un bloc de signalisation à plusieurs voies (par exemple "ALARM\_8"), vous pouvez affecter à chaque sous-numéro ses propres textes et en partie des attributs propres.
- Affectez les visuels souhaités au modèle de message en cliquant sur le bouton "Nouveau visuel", puis en sélectionnant des visuels dans la boîte de dialogue "Insérer un visuel" qui apparaît.

Saisissez également, dans les pages d'onglet suivantes, les textes et attributs souhaités pour les visuels. Quittez la boîte de dialogue par "OK".

---

#### Nota

Pour éditer les textes et attributs propres au visuel, veuillez consulter la documentation livrée avec ce visuel.

---

## Création de DB d'instance

1. Après avoir créé un modèle de message, vous pouvez lui affecter des blocs de données d'instance (DB) et éditer le message propre à chaque instance.  
Dans SIMATIC Manager, ouvrez à cet effet le bloc qui doit appeler votre FB préalablement configuré, par exemple l'"OB1", en cliquant deux fois dessus. Dans la section des instructions ouverte de l'OB, entrez l'appel ("CALL") suivi du nom et du numéro du FB à appeler ainsi que du DB que vous voulez affecter au FB comme instance. Confirmez votre saisie par la touche ENTREE.

**Exemple :**entrez "CALL FB1, DB1". Si le DB1 n'existe pas encore, confirmez par "Oui" la demande de génération du DB d'instance.

**Résultat :** le DB d'instance est créé. Les variables d'entrée du FB correspondant, dans notre cas "Mess01" ainsi que le numéro de message attribué par le système, ici "1" s'affichent dans la section des instructions.

2. Enregistrez l'OB avec la commande **Fichier > Enregistrer** et fermez la fenêtre "CONT, LIST, LOG".

## Edition de messages

1. Dans SIMATIC Manager, sélectionnez le DB d'instance créé, par exemple le "DB1" et appelez la configuration des messages en choisissant la commande **Edition > Propriétés spécifiques de l'objet > Signalisation....**

**Résultat :** la boîte de dialogue "Configuration des messages" s'ouvre et le DB d'instance sélectionné s'affiche avec le numéro de message attribué par le système.

2. Effectuez les modifications souhaitées pour le DB d'instance respectif dans les diverses pages d'onglet et si vous le souhaitez, ajoutez d'autres visuels. Quittez la fonction par "OK".

**Résultat :** la configuration des messages est ainsi terminée pour le DB d'instance sélectionné.

## Transfert des données de configuration

- Transférez les données configurées dans la base de données de WinCC (avec la fonction Transfert des données vers l'OS) ou dans celle de ProTool.

### 16.2.2.2 Edition de messages sur bloc (pour tout le projet)

1. Sélectionnez le bloc souhaité dans SIMATIC Manager, puis choisissez la commande de menu **Edition > Propriétés spécifiques de l'objet > Message**.
2. Dans la structure des répertoires, cliquez sur une entrée de bloc de signalisation ou sur l'un de ses sous-numéros (s'il en existe)

**Résultat** : la partie de la page d'onglet correspondant à un message général s'affiche.

3. Saisissez les textes et attributs dans les onglets "Texte" et "Attributs".

**Résultat** : vous avez créé un message standard qui peut s'afficher sur tous les visuels.

4. En cliquant sur le bouton "Nouveau visuel", insérez un nouveau visuel de type "ProTool" (Opx) ou "WinCC". Seuls les visuels sur lesquels les messages configurés peuvent être affichés vous sont proposés.

**Résultat** : le nouveau visuel se trouve inséré et sélectionné, l'onglet correspondant s'affiche.

5. Saisissez des attributs et des textes pour le message propre à un visuel dans les onglets "Textes" et "Attributs" propres à un visuel.

**Résultat** : vous avez créé une variante de message qui sera utilisée seulement pour le message sur le visuel choisi.

Pour éditer d'autres variantes du message pour des visuels déjà existants,

- ouvrez le bloc de signalisation par un double clic dans la vue de détail.

**Résultat** : le premier visuel se trouve automatiquement choisi et vous pouvez éditer la variante de message qui lui est propre.

### 16.2.2.3 Configuration des messages PCS7 (pour tout le projet)

Pour l'édition de modèles de messages et de messages devant s'afficher sur les visuels WinCC, la configuration des messages PCS7 mise à votre disposition par STEP 7 vous offre la possibilité de

- simplifier la configuration des visuels (ils sont créés automatiquement),
- simplifier la saisie d'attributs et de textes pour les messages,
- garantir l'homogénéité des messages.



## Appel de la configuration des messages PCS 7

1. Dans SIMATIC Manager, sélectionnez le bloc (FB ou DB) dont vous souhaitez éditer les textes de message et choisissez la commande **Edition > Propriétés de l'objet** pour ouvrir la boîte de dialogue de saisie des attributs système.
2. Entrez dans la table qui s'affiche l'attribut système "S7\_alarm\_ui" et la valeur : "1" (la valeur "0" désactive la configuration de messages PCS 7). Les paramètres des attributs peuvent être attribués dans CONT/LOG/LIST. Les DB ensuite générés et affectés aux FB correspondants reçoivent ces paramètres et peuvent être commutés indépendamment du type de message (FB) dans leurs propres paramètres d'attributs.

---

### Nota

Une vérification syntaxique accompagne la saisie des attributs système et les entrées erronées sont repérées en rouge.

---

3. Quittez la boîte de dialogue par "OK".
4. Choisissez la commande **Edition > Propriétés spécifiques de l'objet > Signalisation**.

**Résultat** : la boîte de dialogue de la configuration des messages PCS7 s'ouvre.

## Edition de modèles de message

1. Dans SIMATIC Manager, sélectionnez le FB dont vous souhaitez éditer les textes de message et appelez la configuration des messages PCS7.

**Résultat** : Dans la boîte de dialogue, une page d'onglet s'affiche pour chaque bloc de signalisation pour lequel vous avez déclaré une variable dans le FB.

2. Complétez les zones de texte pour les éléments constituant des messages "Origine", "Secteur OS" et "Code Batch".
3. Pour tous les événements des blocs de signalisation utilisés, indiquez la classe du message ainsi que le texte de l'événement et définissez si chaque événement doit être acquitté individuellement .
4. Pour les éléments constituant des messages valables pour toutes les instances et qui ne doivent pas y être modifiés, cochez la case "Verrouillé".

## Edition de messages

1. Dans SIMATIC Manager, sélectionnez le DB d'instance dont vous souhaitez éditer les textes de message et appelez la configuration des messages PCS7.
2. Modifiez les éléments constituant de message qui sont spécifiques aux instances et ne sont pas verrouillés.

## 16.2.3 Affectation et édition de messages sur mnémonique

### 16.2.3.1 Affectation et édition de messages sur mnémonique (pour tout le projet)

Les messages sur mnémonique (SCAN) sont directement affectés à un signal dans la table des mnémoniques. Les signaux autorisés sont tous les opérandes booléens, à savoir les entrées (E), les sorties (A) et les mémentos (M). Dans l'application de configuration des messages vous pouvez leur affecter différents attributs, textes de message et jusqu'à 10 variables additionnelles. La définition de filtres vous facilite le choix des signaux dans la table des mnémoniques.

Un message sur mnémoniques vous permet de scruter un signal selon un intervalle de temps donné afin de détecter un éventuel changement d'état.

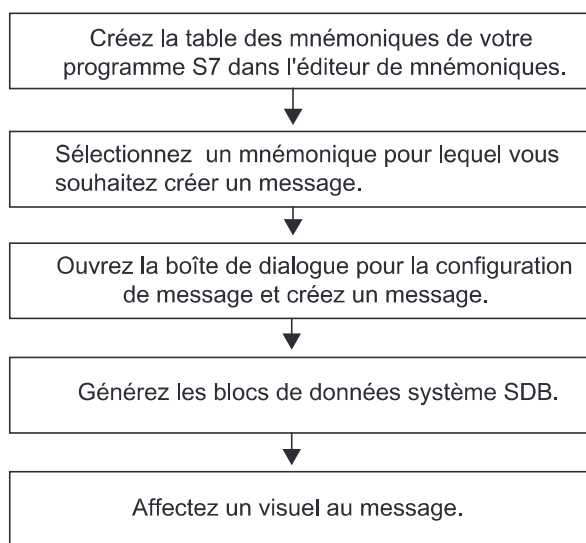
---

#### Nota

L'intervalle de temps dépend de la CPU utilisée.

---

#### Marche à suivre



Les signaux pour lesquels vous avez configuré des messages sont scrutés de manière asynchrone à l'exécution de votre programme. La scrutation a lieu aux intervalles de temps configurés. Les messages s'affichent sur les visuels affectés.

---

#### Important

Si vous souhaitez affecter ou éditer des messages sur mnémonique et avez précédemment copié des mnémoniques entre deux tables de mnémoniques, vous devez d'abord fermer la table de mnémoniques dans laquelle vous ne souhaitez plus travailler. Sinon, vous ne pouvez pas enregistrer la configuration des messages. Le cas échéant, les dernières entrées dans la boîte de dialogue de la configuration des messages sont perdues.

---

## 16.2.4 Création et édition de messages de diagnostic personnalisés

Cette fonction vous permet d'écrire une entrée utilisateur dans la mémoire de diagnostic et d'émettre un message correspondant que vous créez dans l'application de configuration des messages. C'est la fonction SFC52 (WR\_USMSG ; classe d'erreur A ou B) qui, utilisée comme bloc de signalisation, réalise ces messages de diagnostic personnalisés. Vous devez insérer l'appel de la SFC52 dans votre programme utilisateur et lui affecter l'ID d'événement.

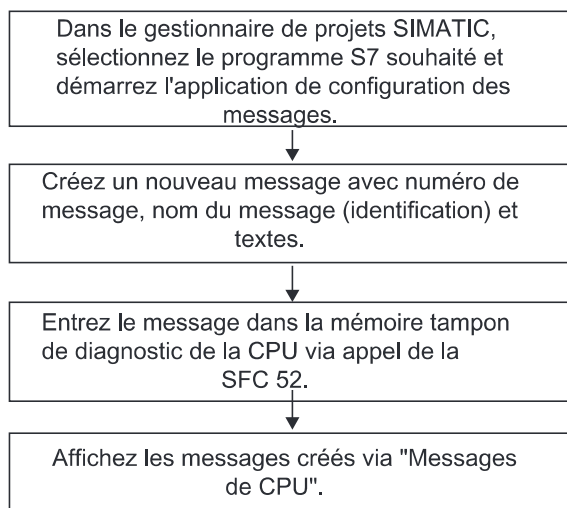
### Conditions préalables

Pour pouvoir créer un message de diagnostic personnalisé, vous devez avoir :

- créé un projet dans SIMATIC Manager et
- dans ce projet, créé un programme S7 auquel vous souhaitez affecter un ou plusieurs messages.

### Marche à suivre

Pour créer et afficher un message de diagnostic personnalisé, procédez de la manière suivante :



## 16.3 Configuration de messages pour la CPU

### 16.3.1 Attribution de numéros de message pour la CPU

Les messages sont identifiés par un numéro univoque pour la CPU. Une plage de numéros est à cet effet attribuée à chaque CPU. Lorsque vous copiez un programme, vous ne devez pas attribuer de nouvelle plage de numéros au nouveau programme - contrairement à ce que vous faites lors de l'attribution de numéros de message pour l'ensemble du projet. Une nouvelle compilation des programmes ne s'avère donc plus nécessaire. La copie de blocs individuels constitue une exception. Dans ce cas, une nouvelle compilation est nécessaire pour intégrer les numéros de message modifiés au programme.

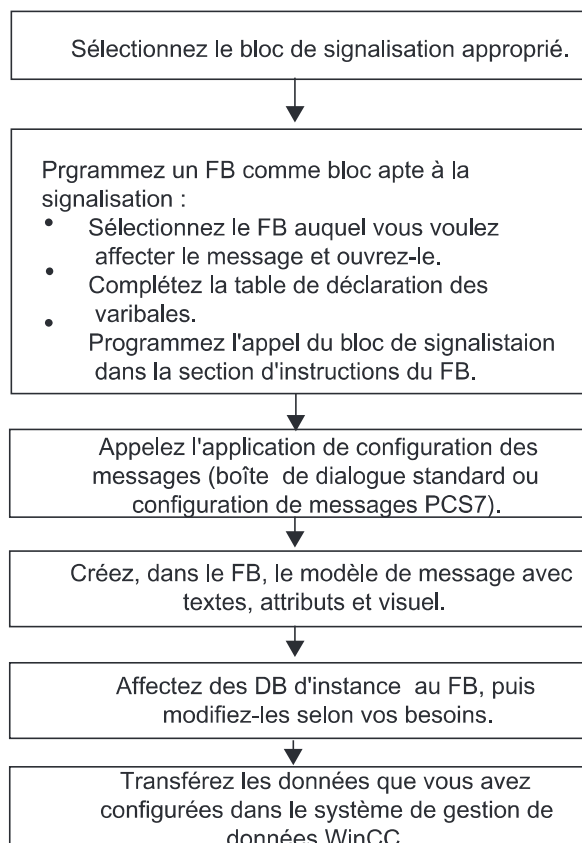
#### Conditions

- WinCC V6.0
- ProTool V6.0

### 16.3.2 Affectation et édition de messages sur bloc

#### 16.3.2.1 Création de messages sur bloc (pour la CPU)

#### Marche à suivre



## Programmation d'un bloc apte à la signalisation (FB)

1. Dans SIMATIC Manager, sélectionnez le bloc fonctionnel (FB) pour lequel vous souhaitez créer un message sur bloc et ouvrez-le par double clic.

**Résultat :** le bloc sélectionné s'ouvre et s'affiche dans la fenêtre "CONT/LOG/LIST".

2. Complétez la table de déclaration des variables. Pour chaque bloc de signalisation appelé dans le FB, vous devez déclarer des variables dans le FB appelant.

Pour ce faire, entrez les variables suivantes dans la vue des variables :

- pour le paramètre "IN", un mnémonique pour l'entrée du bloc de signalisation, par exemple "Mess01" (pour l'entrée du message 01) ainsi que le type de données correspondant (il doit s'agir de "DWORD" sans valeur initiale).
- pour le paramètre "STAT", un mnémonique pour le bloc de signalisation à appeler, par exemple "alarme" ainsi que le type de données correspondant, en l'occurrence "SFB33".

3. Dans la section des instructions du FB, insérez l'appel du bloc de signalisation sélectionné, dans notre exemple "CALL alarme", puis validez votre saisie par la touche ENTREE.

**Résultat :** les variables d'entrée du bloc de signalisation appelé, dans notre exemple le SFB33, s'affichent dans la section des instructions du FB.

4. Affectez à la variable "EV\_ID" le mnémonique que vous aviez affecté à l'étape 2 à l'entrée du bloc de signalisation, dans notre cas "Mess01".

**Résultat :** si la colonne "Nom" n'est pas sélectionnée, un "drapeau" y apparaît pour le paramètre "IN". Le bloc sélectionné acquiert ainsi des fonctions de signalisation. Les attributs système requis (par exemple S7\_server et S7\_a\_type) ainsi que les valeurs correspondantes sont affectés automatiquement (pour certaines SFC, vous devez affecter vous-même les attributs système pour le paramètre "IN" (commande de menu **Edition > Propriétés de l'objet**, onglet "Attributs")).

**Important :** si au lieu d'un SFB, vous appelez un FB avec des multi-instances et dans lequel des messages sont également configurés, vous devez alors configurer les messages du FB avec des multi-instances dans le bloc appelant.

5. Renouvelez les étapes 2 à 4 pour tous les appels de blocs de signalisation dans ce FB.
6. Enregistrez le bloc en choisissant la commande **Fichier > Enregistrer**.
7. Fermez la fenêtre "CONT, LIST, LOG".

## Appel de l'application de configuration des messages

- Sélectionnez le bloc de signalisation souhaité, puis dans SIMATIC Manager, choisissez la commande **Edition > Propriétés spécifiques de l'objet > Signalisation**.

**Résultat :** la boîte de dialogue de la configuration des messages de STEP 7 s'ouvre. Pour savoir comment appeler la configuration des messages PCS 7, reportez vous à Configuration des messages PCS7 (pour la CPU).

### Edition d'un modèle de message

- Sélectionnez le bloc de signalisation voulu
- Saisissez les textes souhaités dans les colonnes correspondantes ou sélectionnez les attributs souhaités.  
Vous pouvez également cliquer sur le bouton "Etendu" dans la boîte de dialogue "Configuration des messages", puis entrer le texte de message et le texte complémentaire souhaités dans l'onglet "Textes par défaut".  
Si vous avez sélectionné un bloc de signalisation à plusieurs voies (par exemple "ALARM\_8"), vous pouvez affecter à chaque sous-numéro ses propres textes et en partie ses propres attributs. Les attributs valent pour tous les sous-numéros.
- Si vous souhaitez que les textes et attributs ne puissent pas être modifiés dans l'instance, verrouillez-les dans le modèle de message.

### Création de DB d'instance

1. Après avoir créé un modèle de message, vous pouvez lui affecter des blocs de données d'instance (DB) et éditer le message propre à chaque instance.  
Dans SIMATIC Manager, ouvrez à cet effet le bloc qui doit appeler votre FB préalablement configuré, par exemple l'"OB1", en cliquant deux fois dessus. Dans la section des instructions ouverte de l'OB, entrez l'appel ("CALL") suivi du nom et du numéro du FB à appeler ainsi que du DB que vous voulez affecter au FB comme instance. Confirmez votre saisie par la touche ENTREE.

**Exemple :** entrez "CALL FB1, DB1". Si le DB1 n'existe pas encore, confirmez par "Oui" la demande de génération du DB d'instance.

**Résultat :** le DB d'instance est créé. Les variables d'entrée du FB correspondant, dans notre cas "Mess01" ainsi que le numéro de message attribué par le système, ici "1" s'affichent dans la section des instructions.

2. Enregistrez l'OB avec la commande **Fichier > Enregistrer** et fermez la fenêtre "CONT LIST LOG".

## Edition de messages

1. Dans SIMATIC Manager, sélectionnez le DB d'instance créé, par exemple le "DB1" et appelez la configuration des messages en choisissant la commande **Edition > Propriétés spécifiques de l'objet > Signalisation...**

**Résultat** : la boîte de dialogue "Configuration des messages" s'ouvre et le DB d'instance sélectionné s'affiche avec le numéro de message attribué par le système.

2. Effectuez les modifications souhaitées pour le DB d'instance respectif dans les diverses pages d'onglet et si vous le souhaitez, ajoutez d'autres visuels. Quittez la fonction par "OK".

**Résultat** : la configuration des messages est ainsi terminée pour le DB d'instance sélectionné.

---

### Nota

Si des textes et attributs sont représentés en vert dans un instance, cela signifie que ces textes et attributs sont encore tels qu'ils ont été configurés dans le modèle. Entre-temps, ils n'ont pas été modifiés dans l'instance.

---

## Transfert des données de configuration

- Transférez les données configurées dans la base de données de WinCC (avec la fonction de Transfert des données vers l'OS) ou dans celle de ProTool.

### 16.3.2.2 Edition de messages sur bloc (pour la CPU)

1. Sélectionnez un bloc de signalisation et appelez la configuration des messages en choisissant la commande de menu **Edition > Propriétés spécifiques de l'objet > Signalisation**.
2. Entrez les textes souhaités dans les colonnes "Textes par défaut" et "Textes complémentaires".  
Vous pouvez également cliquer sur le bouton "Etendu", puis entrer les souhaités avec des retours chariot dans les onglets "Textes par défaut" et "Textes complémentaire".

**Résultat** : vous avez créé un message standard.

---

#### Nota

Si des textes et attributs sont représentés en vert dans un instance, cela signifie que ces textes et attributs sont encore tels qu'ils ont été configurés dans le modèle. Entre-temps, ils n'ont pas été modifiés dans l'instance.

---

### 16.3.2.3 Configuration des messages PCS7 (pour la CPU)

Pour l'édition de modèles de messages et de messages devant s'afficher sur les visuels WinCC (à partir de V6.0), la configuration des messages PCS7 mise à votre disposition par STEP 7 vous offre la possibilité de

- simplifier la configuration des visuels (ils sont créés automatiquement),
- simplifier la saisie d'attributs et de textes pour les messages,
- garantir l'homogénéité des messages.

#### Appel de la configuration des messages PCS7

1. Dans SIMATIC Manager, sélectionnez le bloc (FB ou DB) dont vous souhaitez éditer les textes de message et choisissez la commande **Edition > Propriétés de l'objet** pour ouvrir la boîte de dialogue de saisie des attributs système.
2. Entrez dans la table qui s'affiche l'attribut système "S7\_alarm\_ui" et la valeur : "1" (la valeur "0" désactive la configuration de messages PCS 7). Les paramètres des attributs peuvent être attribués dans CONT/LOG/LIST. Les DB ensuite générés et affectés aux FB correspondants reçoivent ces paramètres et peuvent être commutés indépendamment du type de message (FB) dans leurs propres paramètres d'attributs.

---

#### Nota

Une vérification syntaxique accompagne la saisie des attributs système et les entrées erronées sont repérées en rouge.

---

3. Quittez la boîte de dialogue par "OK".
4. Choisissez la commande **Edition > Propriétés spécifiques de l'objet > Signalisation**.

**Résultat** : la boîte de dialogue de la configuration des messages PCS7 s'ouvre.



### **Edition de modèles de message**

1. Dans SIMATIC Manager, sélectionnez le FB dont vous souhaitez éditer les textes de message et appelez la configuration des messages PCS7.
2. Cliquez sur le bouton "Etendu", puis dans la page d'onglet "Bloc Texte de message" complétez les zones de texte pour les éléments constituant des messages "Origine", "Secteur OS" et "Code Batch".
3. Pour tous les événements des blocs de signalisation utilisés, indiquez la classe du message ainsi que le texte de l'événement et définissez si chaque événement doit être acquitté individuellement .
4. Pour les éléments constituant des messages valables pour toutes les instances et qui ne doivent pas y être modifiés, cochez la case "Verrouillé".

### **Edition de messages**

1. Dans SIMATIC Manager, sélectionnez le DB d'instance dont vous souhaitez éditer les textes de message et appelez la configuration des messages PCS7.
2. Modifiez les éléments constituant de message qui sont spécifiques aux instances et ne sont pas verrouillés.

## 16.3.3 Affectation et édition de messages sur mnémonique

### 16.3.3.1 Affectation et édition de messages sur mnémonique (pour la CPU)

Les messages sur mnémonique (SCAN) sont directement affectés à un signal dans la table des mnémoniques. Les signaux autorisés sont tous les opérandes booléens, à savoir les entrées (E), les sorties (A) et les mémentos (M). Dans l'application de configuration des messages vous pouvez leur affecter différents attributs, textes de message et jusqu'à 10 variables additionnelles. La définition de filtres vous facilite le choix des signaux dans la table des mnémoniques.

Un message sur mnémoniques vous permet de scruter un signal selon un intervalle de temps donné afin de détecter un éventuel changement d'état.

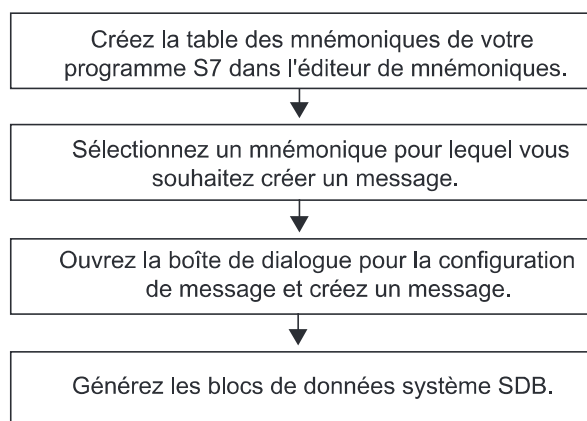
---

#### Nota

L'intervalle de temps dépend de la CPU utilisée.

---

#### Marche à suivre



Les signaux pour lesquels vous avez configuré des messages sont scrutés de manière asynchrone à l'exécution de votre programme. La scrutation a lieu aux intervalles de temps configurés. Les messages s'affichent sur les visuels affectés.

---

#### Important

Si vous souhaitez affecter ou éditer des messages sur mnémonique et avez précédemment copié des mnémoniques entre deux tables de mnémoniques, vous devez d'abord fermer la table de mnémoniques dans laquelle vous ne souhaitez plus travailler. Sinon, vous ne pouvez pas enregistrer la configuration des messages. Le cas échéant, les dernières entrées dans la boîte de dialogue de la configuration des messages sont perdues.

---

### 16.3.4 Création et édition de messages de diagnostic personnalisés

Cette fonction vous permet d'écrire une entrée utilisateur dans la mémoire de diagnostic et d'émettre un message correspondant que vous créez dans l'application de configuration des messages. C'est la fonction SFC52 (WR\_USMSG ; classe d'erreur A ou B) qui, utilisée comme bloc de signalisation, réalise ces messages de diagnostic personnalisés. Vous devez insérer l'appel de la SFC52 dans votre programme utilisateur et lui affecter l'ID d'événement.

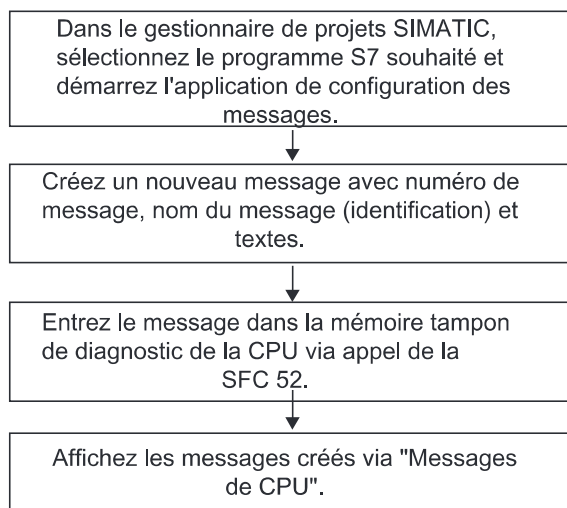
#### Conditions préalables

Pour pouvoir créer un message de diagnostic personnalisé, vous devez avoir :

- créé un projet dans SIMATIC Manager et
- dans ce projet, créé un programme S7 auquel vous souhaitez affecter un ou plusieurs messages.

#### Marche à suivre

Pour créer et afficher un message de diagnostic personnalisé, procédez de la manière suivante :



## 16.4 Conseils pour l'utilisation de messages

### 16.4.1 Insertion d'une variable dans un message

Afin d'ajouter des informations actuelles, par exemple du processus, à des messages sur bloc ou sur mnémonique, vous pouvez insérer des variables additionnelles à un endroit quelconque d'un texte de message.

#### Procédez de la manière suivante :

1. Composez un bloc construit comme suit :  
@<N° de la variable><Type d'élément><Indication de format>@.
2. Insérez ce bloc dans le texte du message aux endroits où vous voulez que la variable additionnelle apparaisse.

#### Type d'élément

Il détermine sans ambiguïté le type de données de la variable additionnelle :

Type d'élément	Type de données
Y	BYTE
W	WORD
X	DWORD
I	entier
D	entier
B	BOOL
C	CHAR
R	REAL

Le type d'élément sert seulement à préciser le type de données transféré dans l'AS. Il n'est pas utilisé comme casting operator.

## Indication de format

Précisez ici le format à utiliser pour représenter la variable additionnelle sur le visuel. L'indication du format est introduite par le caractère %. Pour les textes de message, il y a des indications de format fixes qui sont les suivantes :

Indication du format	Description
%[i]X	Nombre hexadécimal à i positions
%[i]u	Nombre décimal sans signe à i positions
%[i]d	Nombre décimal avec signe à i positions
%[i]b	Nombre binaire à i positions
%[i].y]f	Nombre à virgule fixe Valeur avec signe de la forme [ - ]dddd.dddd dddd : un ou plusieurs chiffres à y positions après la virgule et i positions au total
%[i]s	Chaîne de caractères (ANSI String) à i positions Les caractères sont imprimés jusqu'au premier octet 0 (00hexa).
%t#<nom de la bibliothèque de textes>	Accès à une bibliothèque de textes

Si l'indication du format est trop petite, la valeur est malgré tout affichée dans sa longueur totale.

Si l'indication du format est trop grande, la valeur est affichée précédée d'un nombre correspondant de caractères d'espacement.

### Nota

Sachez que vous pouvez entrer "[i]" de manière optionnelle, sans crochets.

## Exemples de variables additionnelles :

@1I%6d@ : la valeur contenue dans la variable 1 est représentée comme nombre décimal à 6 positions au plus.

@2R%6f@ : la valeur "5.4" contenue dans la variable 2 est représentée comme nombre à virgule fixe "5.4" (trois espacements en tête).

@2R%2f@ : la valeur "5.4" contenue dans la variable 2 est représentée comme nombre à virgule fixe "5.4" (pas de tronquage s'il n'y a pas assez de positions).

@1W%t#BibText1@ : la variable 1 du type de données WORD est l'indicatif caractérisant, dans la bibliothèque BibText1, le texte à incorporer.

### Nota

S7-PDIAG requiert toujours le type d'élément "C" pour CHAR et "R" pour REAL. Les autres types d'éléments valables de S7-PDIAG, à savoir BOOL, BYTE, WORD, INT, DWORD et DINT requièrent toujours "X".

Si vous désirez transmettre plus d'une variable à l'un des blocs ALARM\_S, vous pouvez transmettre un tableau de 12 octets maximum de longueur. Il peut s'agir par exemple de 12 octets ou caractères maximum, 6 mots ou entiers maximum ou 3 doubles mots réels ou entiers (Real ou Dint) maximum.

## 16.4.2 Incorporer dans des messages des textes tirés de bibliothèques

Vous pouvez incorporer dans un message autant de textes que vous le souhaitez provenant de 4 bibliothèques différentes au plus. C'est à vous de choisir la position des textes, ce qui en permet l'emploi dans des messages en langue étrangère.

Procédez de la manière suivante :

1. Sélectionnez la CPU dans SIMATIC Manager, ou un objet subordonné à la CPU, et choisissez la commande **Outils > Bibliothèques de textes > Bibliothèque de textes système ou Outils > Bibliothèques de textes > Bibliothèque de textes utilisateur** pour ouvrir une bibliothèque de textes.

---

### Important

Vous pouvez intégrer des textes de bibliothèques de texte utilisateur dans des messages uniquement si vous avez sélectionné l'attribution des numéros de message pour la CPU.

---

2. Cherchez l'indicatif du texte que vous souhaitez incorporer dans votre message.
3. Tapez un joker de format @[Index]%t#[Textbib]@ à l'endroit du message où vous voulez faire apparaître le texte.

---

### Nota

[Index] = p. ex. 1W, 1W correspondant à la première variable additionnelle du message de type WORD.

---

### Exemple

Texte de message configuré : La pression est @1734W%t#Bibtext1@.

Bibliothèque de textes portant le nom "Bibtext1" :

Indicatif	Allemand	Français
1734	zu hoch	trop haut(e)

La deuxième variable additionnelle transmise a reçu la valeur 1734.

Le texte de message suivant s'affiche : La pression est trop haut(e).

## 16.4.3 Effacer des variables additionnelles

Pour effacer une variable, effacez dans le texte du message la chaîne de caractères représentant la variable.

### Procédez de la manière suivante :

1. Dans le message de texte, cherchez le bloc correspondant à la variable à effacer. Ce bloc commence par le caractère @, il contient le numéro de la variable ainsi qu'une indication de format et se termine par le caractère @.
2. Quand vous l'avez trouvé, effacez le bloc du texte du message.

## 16.5 Traduction et édition de textes destinés à l'utilisateur

Les textes qui s'affichent sur les visuels durant le traitement du processus ont été saisis habituellement dans la langue employée pour programmer la solution d'automatisation.

Il arrive pourtant fréquemment que l'utilisateur devant réagir plus tard à ces messages ne connaisse pas cette langue. Il lui faudrait les textes dans sa langue maternelle. C'est la condition pour garantir un traitement sûr et une réaction rapide aux messages affichés.

STEP 7 vous offre la possibilité de traduire les textes destinés à l'utilisateur dans toutes les langues. Il suffit que vous ayez déjà installé la langue dans votre projet (commande **Outils > Langue de visuel** dans SIMATIC Manager). Le nombre de langues proposées est déterminé lors de l'installation de Windows (propriété du système).

Ceci vous donne l'assurance que tous ceux qui seront confrontés à l'avenir aux messages affichés pourront les obtenir dans la langue qui leur convient. La sûreté du processus s'en trouvera considérablement améliorée.

Parmi les textes destinés à l'utilisateur, on distingue les textes personnalisés et les bibliothèques de textes.

### 16.5.1 Traduction et édition de textes personnalisés

Vous pouvez créer des textes personnalisés pour un projet complet, pour des programmes S7, pour le dossier Blocs ou pour des blocs individuels ainsi que pour la table des mnémoniques, dans la mesure où des messages sont configurés dans ces objets. Vous pouvez établir pour un même projet plusieurs listes de textes personnalisés et les traduire dans les langues dont vous avez besoin.

C'est vous qui choisissez les langues qui seront disponibles dans un projet (commande **Outils > Langue de visuel**). Vous pouvez aussi ajouter ou supprimer des langues ultérieurement.

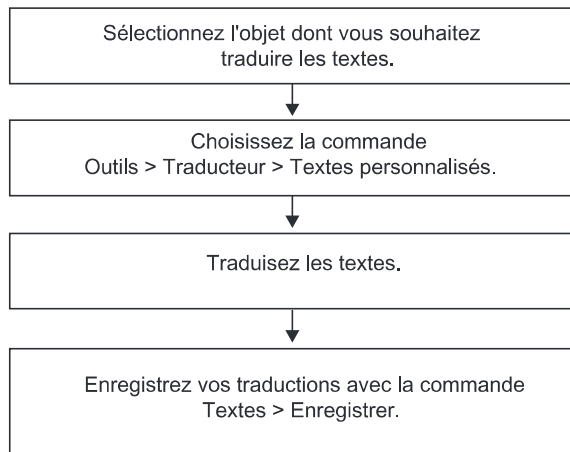
### Exportation et importation de textes personnalisés

Vous pouvez traduire ou éditer en dehors de STEP 7 des textes personnalisés créés dans STEP 7. Pour ce faire, exportez la liste de textes affichée dans des fichiers d'exportation que vous éditez dans un éditeur ASCII ou dans un tableur, comme l'application Microsoft EXCEL par exemple (commande de menu **Outils > Gestion multilingue des textes > Exporter**). Lorsque vous ouvrez ce fichier, l'écran affiche un tableau dont chaque colonne représente une langue. La première colonne affiche toujours la langue paramétrée comme langue standard. Après avoir traduit les textes, réimportez-les dans STEP 7.

Les textes personnalisés ne peuvent être importés que dans la partie de projet d'où ils ont été exportés.

### Marche à suivre

Vérifiez d'abord que vous avez bien choisi, avec la commande **Outils > Langue de visuel** de SIMATIC Manager, les langues dans lesquelles vous souhaitez traduire les textes personnalisés.



---

### Nota

Vous pouvez uniquement imprimer les textes utilisateur dans l'application dans laquelle vous les traduisez.

---



## 16.6 Edition et traduction de bibliothèques de textes

### 16.6.1 Bibliothèques de texte utilisateur

En utilisant une bibliothèque de textes utilisateur, vous pouvez afficher des textes ou parties de texte de manière dynamique en fonction d'une variable additionnelle. Cette dernière fournit à cet effet au texte actuel l'index de la bibliothèque de textes. Un joker est inscrit à l'endroit où le texte dynamique doit être intégré.

Vous avez la possibilité de créer des bibliothèques de textes utilisateur pour un programme, d'y saisir des textes et leur affecter votre propre index. L'application vérifie automatiquement si l'index est univoque dans la bibliothèque de textes utilisateur. Tous les messages disponibles de cette CPU peuvent faire référence à une bibliothèque de textes utilisateur.

Un dossier Bibliothèques de textes peut contenir un nombre illimité de bibliothèques de textes. Ceci permet p. ex. d'utiliser le même programme pour différentes tâches d'automatisation en adaptant simplement les bibliothèques de textes à vos exigences.

---

#### Important

Lorsque vous copiez, dans un autre programme, un bloc de signalisation faisant référence à des bibliothèques de textes, vous devez également copier les bibliothèques de textes correspondantes ou créer une autre bibliothèque de textes avec le même nom et le même contenu ou encore modifier la référence dans le texte de message.

---

Lorsque vous créez une entrée de texte, l'attribution d'un index est indispensable. Lorsque vous créez une nouvelle ligne, l'application propose par défaut le prochain index libre. Des index doubles ne sont pas autorisés dans une bibliothèque de textes et sont rejetés par l'application.

### 16.6.2 Création de bibliothèques de textes utilisateur

Pour créer une bibliothèque de textes utilisateur, procédez de la manière suivante :

1. Dans SIMATIC Manager, sélectionnez le programme ou l'un des objets qu'il contient pour lequel vous souhaitez créer une bibliothèque de textes utilisateur, puis choisissez la commande de menu **Insertion > Bibliothèque de textes > Dossier Bibliothèque de textes**.

**Résultat :** le dossier "Bibliothèque de textes" est créé.

2. Sélectionnez ensuite le dossier "Bibliothèque de textes", choisissez la commande de menu **Insertion > Bibliothèque de textes > Bibliothèque de textes utilisateur** et entrez un nom pour la bibliothèque de textes.
3. Pour ouvrir la bibliothèque de textes, choisissez la commande de menu **Outils > Bibliothèques de textes > Bibliothèque de textes utilisateur**.
4. Vous pouvez à présent entrer les textes souhaités.

---

#### Nota

Lorsque vous créez une entrée de texte, l'attribution d'un index est forcée. Lors de la création d'une nouvelle ligne, l'application propose par défaut le prochain index libre. Les index doubles ne sont pas autorisés dans une bibliothèque de textes et sont rejetés par l'application.

### 16.6.3 Edition de bibliothèques de textes utilisateur

Pour éditer des bibliothèques de textes utilisateur existantes, procédez de la manière suivante :

1. Sélectionnez dans SIMATIC Manager le programme ou l'un des objets qu'il contient et dont vous souhaitez éditer la bibliothèque de textes, puis choisissez la commande de menu **Outils > Bibliothèque de textes > Bibliothèque de textes utilisateur**.
2. Dans la boîte de dialogue "Bibliothèques de textes disponibles", sélectionnez la bibliothèque de textes utilisateur que vous souhaitez ouvrir.
3. Editez les textes affichés. Diverses fonctions d'édition vous sont proposées à cet effet (p. ex. Recherche et remplacement).  
Vous pouvez entrer vos propres textes. L'index des textes utilisateur est créé automatiquement, mais peut être modifié à tout moment. Si vous entrez un index déjà attribué, cette valeur s'affiche en rouge.  
Pour insérer une nouvelle ligne, choisissez la commande de menu **Insertion > Nouvelle ligne** ou cliquez sur l'icône correspondante dans la barre d'outils.
4. Imprimez les textes dont vous souhaitez disposer sur papier.
5. Une fois toutes vos tâches réalisées, enregistrez la bibliothèque de textes utilisateur.
6. Après avoir édité tous les textes souhaités, quittez l'application.

---

#### Important

Lorsque vous copiez, dans un autre programme, un bloc de signalisation faisant référence à des bibliothèques de textes, vous devez également copier les bibliothèques de textes correspondantes ou créer une autre bibliothèque de textes avec le même nom ou encore modifier la référence dans le texte de message.

Lorsque vous modifiez le nom d'une bibliothèque de textes existante, toutes les variables additionnelles qui font référence à cette bibliothèque de textes ne seront plus valides dans les messages déjà configurés !

---

### 16.6.4 Bibliothèques de textes système

Une bibliothèque de textes système est créée automatiquement, p. ex. lors de la génération de blocs dans la "Signalisation d'erreurs système". Vous ne pouvez pas créer vous-même de bibliothèques de textes système, mais pouvez uniquement éditer celles qui existent.

Tous les messages disponibles de cette CPU peuvent faire référence à une bibliothèque de textes.

## 16.6.5 Traduction de bibliothèques de textes

Les bibliothèques de textes système et les bibliothèques de textes utilisateur proposent une liste de textes pouvant être incorporés dans des messages, actualisés dynamiquement durant l'exécution du processus et affichés sur la PG ou sur d'autres visuels.

Les textes contenus dans les bibliothèques de textes système sont fournis par STEP 7 ou par des logiciels optionnels de STEP 7. A une même CPU peuvent être affectées plusieurs bibliothèques que vous traduirez dans les langues dont vous avez besoin.

C'est vous qui choisissez les langues qui seront disponibles dans un projet (commande **Outils > Langue de visuel**). Vous pouvez aussi ajouter ou supprimer des langues ultérieurement.

Lorsque vous souhaitez traduire une bibliothèque de textes (commande de menu **Outils > Gestion multilingue de textes > Exporter**), un fichier d'exportation que vous pouvez éditer p. ex. avec Microsoft EXCEL est créé. Lorsque vous ouvrez ce fichier, l'écran affiche un tableau dont les colonnes représentent respectivement une langue.

---

### Important

Un fichier d'exportation au format CSV ne doit pas être ouvert par double clic. Ouvrez ce fichier en choisissant la commande de menu **Fichier > Ouvrir** de Microsoft EXCEL.

---

### Nota

Vous pouvez uniquement imprimer les textes utilisateur dans l'application dans laquelle vous les traduisez.

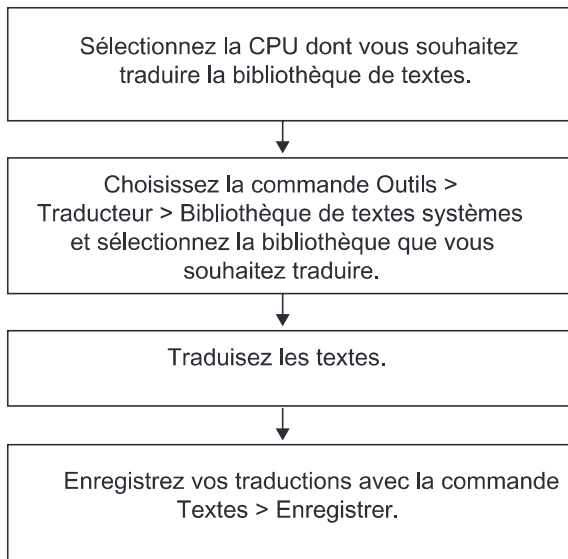
---

### Exemple pour un fichier d'exportation

français	allemand
ne répond pas	ausgefallen
défaillant	gestört
erreur de paramétrage	Parametrierfehler

## Procédé

Vérifiez d'abord que vous avez bien choisi, avec la commande **Outils > Langue de visuel** de SIMATIC Manager, les langues dans lesquelles vous souhaitez traduire la bibliothèque de textes



## 16.7 Transfert des données de configuration dans le système cible

### Introduction

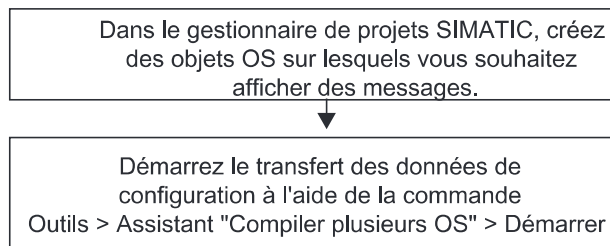
L'application **Transfert des données vers l'OS** vous permet de transférer les données de configuration que vous avez créées pour le contrôle-commande dans le stock de données de WinCC.

### Conditions préalables

Pour pouvoir commencer le transfert, il faut que les conditions suivantes soient remplies :

- vous avez installé le programme AS-OS-Engineering,
- vous avez créé les données de configuration pour la création de messages.

### Marche à suivre



## 16.8 Affichage des messages de CPU et des messages de diagnostic personnalisés

La fonction "Messages de CPU" (commande de menu **Système cible > Messages de CPU**) permet de visualiser des messages asynchrones d'événements de diagnostic, des messages de diagnostic personnalisés aussi bien que des messages de blocs ALARM\_S (SFC 18 et SFC 108 pour la création de messages sur bloc toujours acquittés ainsi que SFC 17 et SFC 107 pour la création de messages sur bloc acquittables).

En outre, vous pouvez démarrer la configuration des messages et créer des messages de diagnostic personnalisés depuis l'application "Messages de CPU", grâce à la commande **Edition > Message > Diagnostic personnalisé**.

### Possibilités d'affichage

La fonction "Messages de CPU" vous permet de décider si et comment des messages seront affichés pour des CPU sélectionnées.

- **"Mise en valeur dans la barre des tâches"** : aussitôt qu'un message est reçu et que la fenêtre ne se trouve pas au premier plan, "Messages de la CPU" est mis en valeur dans la barre des tâches de Windows.
- **"A l'arrière-plan"** : la réception des messages de CPU se déroule à l'arrière-plan. La fenêtre reste à l'arrière-plan à la réception de nouveaux messages, mais peut être amenée au premier plan si besoin est.
- **"Ignorer le message"** : les nouveaux messages de CPU **ne sont pas affichés** et, contrairement aux deux cas précédents, **ne sont pas non plus archivés**.

Dans la fenêtre "Messages de CPU", vous pouvez choisir entre les onglets "Archive" et "Alarme". Dans chacun de ces onglets, vous pouvez choisir si vous souhaitez afficher les messages avec ou sans texte d'information grâce à la commande de menu **Affichage > Afficher le texte d'information**.

### Onglet "Archive"

Les messages arrivants y sont affichés et archivés selon leur ordre d'apparition. Vous pouvez définir la taille de l'archive (entre 40 et 3000 messages de CPU) en choisissant la commande de menu **Outils > Paramètres** dans la boîte de dialogue "Paramètres - Messages de CPU". Lorsque la taille paramétrée pour l'archive est dépassée, c'est le message le plus ancien qui est respectivement effacé.

Les messages acquittables (ALARM\_SQ et ALARM\_DQ) sont représentés en caractères gras et peuvent être acquittés par la commande de menu **Edition > Acquitter le message de la CPU**.

La figure ci-après en présente quelques exemples.

The screenshot shows the 'Messages de la CPU' window with the following data:

Date/Heure	ID	Texte du message	Etat
08.01.94 04:45:24:535	1	<b>Châssis: 0 Emplacement: 8: Module débroché</b> Nom: UR1 Module: CP 443-1 Adresse périph.: E16380	<b>A</b>
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: SFM			
08.01.94 04:45:27:536	1	Châssis: 0 Emplacement: 8: Module débroché Nom: UR1 Module: CP 443-1 Adresse périph.: E16380	D
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: SFM			
08.01.94 04:45:28:993	1.	Message de diagnostic défini par l'utilisateur, classe A, N° 1	A
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Diagnostic utilisateur			
08.01.94 04:45:30:023	1.	Message de diagnostic défini par l'utilisateur, classe A, N° 1	D
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Diagnostic utilisateur			
08.01.94 04:45:31:800	8	<b>Texte libre pour un message Alarm_SQ</b>	<b>A</b>
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Programme			
08.01.94 04:45:31:800	9	Texte libre pour un message Alarm_S	A Q
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Programme			
08.01.94 04:45:32:821	8	Texte libre pour un message Alarm_SQ	D
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Programme			
08.01.94 04:45:32:822	9	Texte libre pour un message Alarm_S	D
Module: Alarmtest1\SIMATIC 417-4\CPU 417-4 Source: Programme			

Archives / Alarme / Prêt Message 1 de 8 sélectionné

## Onglet "Alarme"

Les messages de blocs ALARM\_S encore présents ou non acquittés sont également affichés avec leur état dans l'onglet "Alarme".

En choisissant la commande de menu **Affichage > Messages multilignes**, vous pouvez choisir d'afficher les messages sur une ou plusieurs lignes. De plus, vous pouvez trier les colonnes comme vous le souhaitez.

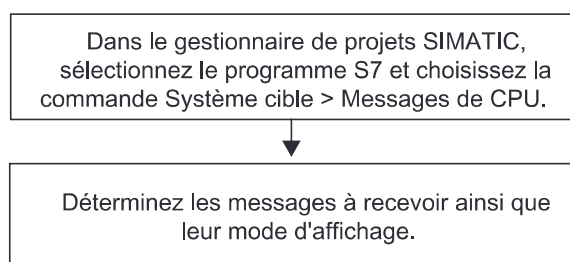
## Actualisation des messages de blocs ALARM\_S

Lors de l'actualisation, tous les messages encore présents ou non acquittés sont une nouvelle fois inscrits dans l'archive. Les messages sont actualisés :

- au redémarrage du module auquel se rapportent les messages (pas au démarrage),
- lorsque vous activez le champ "A" pour les messages de blocs ALARM\_S dans la liste des modules.

## Marche à suivre

Pour configurer des messages de CPU pour des modules sélectionnés :





## 16.8.1 Configuration des messages de CPU

Pour configurer des messages de CPU pour des modules sélectionnés, procédez de la manière suivante :

1. Dans SIMATIC Manager, démarrez l'application "Messages de CPU" à partir d'un projet en ligne. Pour ce faire, sélectionnez un programme S7 en ligne et appelez l'application pour la CPU choisie à l'aide de la commande **Système cible > Messages de CPU**.

**Résultat :** la fenêtre de l'application "Messages de CPU" apparaît avec la liste des CPU déclarées.

2. Au besoin, déclarez d'autres CPU en répétant l'étape 1 pour d'autres programmes ou interfaces.
3. Cochez les cases placées devant les entrées de la liste et déterminez quels messages doivent être reçus pour le module.

A : active des messages de blocs ALARM\_S (SFC 18 et SFC 108 pour la création de messages sur bloc toujours acquittés ainsi que SFC 17 et SFC 107 pour la création de messages sur bloc acquittables), par ex. messages de diagnostic du processus provenant de S7-PDIAG, S7-GRAPH ou "Signalisation d'erreurs système".

W : active les événements de diagnostic.

4. Définissez la taille du fichier d'archives.

**Résultat :** dès que les messages en question se présenteront, ils seront écrits dans les archives de messages sous la forme que vous avez définie, et affichés.

---

### Nota

La liste des modules déclarés dans la fenêtre de l'application "Messages de CPU" énumère les CPU pour lesquelles vous avez appelé la commande **Système cible > Messages de CPU**. Les entrées restent dans la liste jusqu'à ce que vous les effaciez.

---

## 16.8.2 Affichage des messages de CPU enregistrés

Les messages de CPU sont toujours enregistrés dans le fichier d'archives, à moins que vous n'ayez activé la commande **Affichage > Ignorer message**. L'affichage montre toujours tous les messages archivés.

## 16.9 Configuration de la signalisation d'erreurs système

### Introduction

Les composants SIMATIC S7 et les "esclaves DP normés" (esclaves dont les propriétés sont déterminées par leur fichier GSD) peuvent déclencher des appels de blocs d'organisation à l'apparition d'une erreur système.

Exemple : en cas de rupture de fil, un module capable de diagnostic peut déclencher une alarme de diagnostic (OB82).

Les composants SIMATIC S7 fournissent des informations sur l'erreur système survenue. Les informations d'événement déclencheur, c'est-à-dire les données locales de l'OB associé (contenant entre autres l'enregistrement 0) donnent des renseignements d'ordre général sur le lieu de l'erreur (par ex. adresse logique du module) et sur son type (par ex. erreur de voie ou défaillance de sauvegarde).

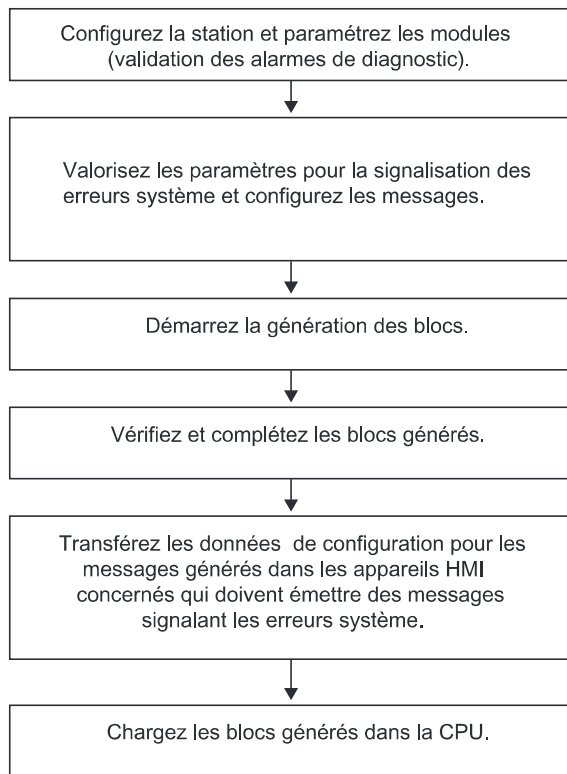
De plus, des informations de diagnostic complémentaires (lecture de l'enregistrement 1 avec SFC51 ou du télégramme de diagnostic d'esclaves DP normés avec SFC 13) donnent plus de détail sur l'erreur : par ex. voie 0 ou 1, rupture de fil ou dépassement de plage de mesure.

La fonction "Signalisation d'erreurs système" de STEP 7 offre un moyen convivial d'afficher sous forme de messages les informations de diagnostic fournies par le composant.

Les blocs et les textes de message nécessaires à cet effet sont générés automatiquement par STEP 7. L'utilisateur n'a plus qu'à charger les blocs générés dans la CPU et à transférer les textes dans les appareils IHM (interface homme-machine) connectés.

Un tableau précis des informations de diagnostic prises en charge selon les différents esclaves se trouve à la rubrique Composants prises en charge et fonctionnalités.

### Marche à suivre de principe



Les messages sont envoyés par le chemin de signalisation standard ALARM\_S/SQ à la fonction "Messages de CPU" sur la PG ou aux appareils IHM connectés.

## 16.9.1 Composants pris en charge et fonctionnalités

Les composants des stations S7-300, des stations S7-400, des esclaves DP et Win AC sont pris en charge par la "Signalisation d'erreurs système" supportant les alarmes de diagnostic, de débrogage/enfichage et le diagnostic spécifique à chaque voie.

Les composants suivants **ne sont pas** pris en charge par la "Signalisation d'erreurs système" :

- les configurations M7, C7 et PROFIBUS DP utilisant coupleurs de maître DP (CP 342-5 DP) dans les stations S7-300,

En cas de redémarrage, il faut savoir que certains messages d'alarme peuvent manquer. En effet, la mémoire d'acquiescement des messages n'est pas effacée dans la CPU au redémarrage, mais la "Signalisation d'erreurs système" remet les données internes à zéro.

Les deux tableaux ci-après indiquent tous les blocs de diagnostic de différents esclaves PROFIBUS pris en charge par la "Signalisation d'erreurs système".

Bloc de diagnostic	Identificateur (emplacement d'enfichage erroné)	Voie (voie erronée) <sup>1)</sup>	Etat du module (erreur de module, module incorrect/ manquant)	Appareil
<b>En-tête</b> <sup>2)</sup>	<b>0x01</b>	<b>0x10</b>	<b>0x00 type 0x82</b>	<b>0x00 + 1 octet info diagnostic</b>
ET 200 S	Message : "Présence d'un diagnostic"	Message en clair	Message en clair	-
ET 200 M	N'est pas évalué	N'est pas évalué	N'est pas évalué	-
ET 200 X	Message : "Présence d'un diagnostic"	-	-	-
ET 200 X DESINA	Message : "Présence d'un diagnostic"	Message en clair	Message en clair	-
ET 200 L	N'est pas évalué	-	-	-
ET 200 B TOR	-	-	-	Message : "Module défectueux"
ET 200 B analogique	-	-	-	-
ET 200 C TOR	-	-	-	-
ET 200 C analogique	Message : "Présence d'un diagnostic"	-	-	Message : "Module défectueux"
ET 200 U	Message : "Présence d'un diagnostic"	-	-	Message : "Module défectueux"
ET 200 iS	Message : "Présence d'un diagnostic"	Message en clair	Message en clair	-
ET 200 eco	-	-	-	Message en texte clair
DP AS-i Link	Message : "Présence d'un diagnostic"	-	Message en clair	-

Bloc de diagnostic	Enregistrement 0/1 <sup>1)</sup>	Autres caractéristiques
<b>En-tête <sup>2)</sup></b>	<b>0x00 type 0x01</b>	<b>0x00 autre type</b>
ET 200 S	-	-
ET 200 M	Message en clair	N'est pas évalué
ET 200 X	-	-
ET 200 X DESINA	Message en clair	-
ET 200 L	Message en clair	-
ET 200 B TOR	-	-
ET 200 B analogique	Message en clair	-
ET 200 C TOR	-	-
ET 200 C analogique	Message en clair	-
ET 200 iS	Message en clair	-
ET 200 eco	-	-
DP AS-i Link	Message : "Erreur de module"	-

Les tableaux ci-après indiquent tous les blocs de diagnostic de différents esclaves PROFINET pris en charge par la "Signalisation d'erreurs système".

Bloc de diagnostic	Identificateur (emplacement d'enfichage erroné)	Voie (voie erronée) <sup>1)</sup>	Etat du module (erreur de module, module incorrect/ manquant)	Appareil
<b>En-tête <sup>2)</sup></b>	<b>0x01</b>	<b>0x10</b>	<b>0x00 type 0x82</b>	<b>0x00 + 1 octet info diagnostic</b>
ET 200 S	Message : "Présence d'un diagnostic"	Message en clair	Message en clair	-
SCALANCE Switches	-	-	Message en clair	-

Bloc de diagnostic	DS0/DS1 <sup>1)</sup>	Autres caractéristiques
<b>En-tête <sup>2)</sup></b>	<b>0x00 type 0x01</b>	<b>0x00 autre type</b>
ET 200 S	-	-
SCALANCE Switches	Message en clair	N'est pas évalué

1) DS0 : Diagnostic standard, p. ex. défaillance du module, tension d'alimentation externe ou connecteur frontal manquants.

Taille 4 octets contenus dans les données locales de l'OB 82.

DS1 : erreur de voie, définie différemment pour chaque type de voie, lisible dans le programme utilisateur via la SFC 51.

Les textes sont issus du diagnostic matériel S7.

2) En-tête : identification de différents éléments de diagnostic dans le télégramme de diagnostic.

Le télégramme de diagnostic (également appelé télégramme esclave normé) comporte les blocs de diagnostic précédents. Vous pouvez le lire dans le programme utilisateur avec la SFC 13.

Dans STEP 7, le télégramme de diagnostic s'affiche sous "Représentation hexadécimale", dans la page d'onglet "Diagnostic de l'esclave DP" de la fenêtre en ligne "HW Config" (diagnostic du matériel) lorsque vous appelez l'état du module.

Répéteur de diagnostic : les messages du répéteur de diagnostic s'affichent en clair. Les textes sont extraits du fichier GSD.

## **PROFINET**

- Avec PROFINET IO, le diagnostic de voie s'affiche en clair.
- ET 200S : les esclaves avec adresses comprimées sont pris en charge.
- Pour les appareils PROFINET IO, le diagnostic spécifique au fabricant est pris en charge.

## 16.9.2 Paramétrage de la signalisation d'erreurs système

Il y a plusieurs façons d'appeler la boîte de dialogue des paramètres :

- Sélectionnez dans HW Config la CPU pour laquelle vous voulez configurer la signalisation d'erreurs système. Choisissez ensuite la commande **Outils > Signalisation d'erreurs système**.
- Si vous avez déjà généré des blocs pour la signalisation d'erreurs système, il vous suffira de cliquer deux fois sur un bloc généré (FB, DB).
- Dans la boîte de dialogue des propriétés de la station, activez l'option qui en détermine l'appel automatique lorsque vous enregistrez et compilez la configuration.

Pour obtenir cette option d'appel automatique, procédez comme suit :

1. Sélectionnez la station concernée dans SIMATIC Manager.
2. Choisissez la commande **Edition > Propriétés de l'objet**.
3. Sélectionnez l'onglet "Paramètres".

---

### Nota

Vous pouvez également ouvrir la page d'onglet "Paramètres" de la boîte de dialogue des propriétés dans HW Config en choisissant la commande **Station > Propriétés**.

---

Dans la boîte de dialogue des paramètres, vous indiquez entre autres :

- quel FB et quel DB d'instance associé il faut générer ;
- s'il faut générer des données de référence ;
- s'il faut toujours afficher les avertissements durant la génération de la signalisation d'erreurs système ;
- si la boîte de dialogue doit s'afficher lorsque la signalisation d'erreurs système est appelée automatiquement après enregistrement et compilation de la configuration (voir ci-dessus) ;
- création d'OB d'erreur : si les OB d'erreur n'existent pas encore dans le programme S7 doivent être créés ou pas et dans quels OB la "Signalisation d'erreurs système" doit être appelée ;
- comportement de la CPU en cas d'erreur : vous pouvez paramétrer quelles classes d'erreur doivent entraîner le passage à l'arrêt de la CPU après leur apparition ;
- l'aspect des messages (composition et ordre des parties de texte possibles).
- si les messages doivent être acquittables ;
- quels paramètres doivent être contenus dans l'interface du bloc utilisateur ;

Vous trouverez des informations plus détaillées dans l'aide de la boîte de dialogue appelée.

### 16.9.3 Génération de blocs pour la signalisation d'erreurs système

Après avoir effectué le paramétrage pour la signalisation d'erreurs système, vous pouvez créer les blocs nécessaires (FB, DB, et en fonction du paramétrage également les OB n'existant pas encore). Cliquez à cet effet sur le bouton "Générer" dans la boîte de dialogue "Signalisation d'erreurs système".

Les blocs suivants sont créés :

- un FB de diagnostic (par défaut FB49),
- un DB d'instance pour le FB de diagnostic (par défaut DB49),
- les OB d'erreur (si vous avez activé cette option dans l'onglet "Configuration des OB"),
- un bloc utilisateur optionnel qui est appelé par le FB de diagnostic.

### 16.9.4 FB, DB générés

Le FB généré évalue les données locales de l'OB d'erreur, lit éventuellement les informations de diagnostic complémentaires du composant SIMATIC S7 qui a provoqué l'erreur, puis génère automatiquement le message approprié.

Il possède les attributs suivants :

- langage de création SFM ("Signalisation d'erreurs système", s'applique également au DB d'instance généré),
- protection Know-How (s'applique également au DB d'instance généré),
- il retarde durant l'exécution les alarmes apparaissantes,
- il appelle sur double clic la boîte de dialogue servant à paramétrer la fonction "Signalisation d'erreurs système" (s'applique également au DB d'instance généré).



## Bloc utilisateur

Le FB de diagnostic ayant la protection Know-how, vous ne pouvez pas l'éditer. Il met cependant à disposition une interface pour le programme utilisateur, de sorte que vous avez accès à l'état d'erreur, par exemple, ou au numéro de message.

Le bloc qui sert à l'évaluation dans le programme utilisateur (paramétrable dans la page d'onglet "Bloc utilisateur" de la boîte de dialogue) est appelé dans le FB généré avec les paramètres sélectionnés. Vous pouvez choisir les paramètres suivants :

Nom	Type de données	Commentaire
EV_C	BOOL	//Message apparaissant (TRUE) ou disparaissant (FALSE)
EV_ID	DWORD	//Numéro de message généré r
IO_Flag	BYTE	//Module d'entrées : B#16#54, Module de sorties : B#16#55
logAdr	WORD	//Adresse logique
TextlistId	WORD	//ID de la bibliothèque de texte (bibliothèque de textes par défaut = 1)
ErrorNo	WORD	//Numéro d'erreur généré
Channel_Error	BOOL	//Erreur de voie (TRUE)
ChannelNo	WORD	//Numéro de voie
ErrClass	WORD	//Classe d'erreur
HErrClass	WORD	//Classe d'erreur des systèmes H

Si le FB utilisateur n'existe pas encore, il est créé par SFM avec les paramètres sélectionnés.

Les textes d'erreur créés pour les erreurs standard correspondent à l'affectation suivante :

Numéro d'erreur (décimal)		OB d'erreur concerné	Code d'erreur dans l'OB	
de	à		de	à
1	86	OB 72	B#16#1	B#16#56
162	163	OB 70	B#16#A2	B#16#A3
193	194	OB 72	B#16#C1	B#16#C2
224		OB 73	B#16#E0	
289	307	OB 81	B#16#21	B#16#33
513	540	OB 82		
865	900	OB 83	B#16#61	B#16#84
1729	1763	OB 86	B#16#C1	B#16#C8

Les numéros d'erreur supérieurs à 12288 correspondent à des erreurs de voie. Lorsque vous affichez les numéros d'erreur en représentation hexadécimale, vous pouvez calculer le type de voie et reconnaître le bit d'erreur. La description précise est donnée dans l'aide respective du module ou dans l'aide de la voie.

Exemple :

12288 = W#16#3000 -> octet de poids fort 0x30 - 0x10 = type de voie 0x20 (interface CP) ;  
octet de poids faible 0x00, c'est-à-dire bit d'erreur 0

32774 = W#16#8006 -> octet de poids fort 0x80 - 0x10 = type de voie 0x70 (entrée TOR) ;  
octet de poids faible 0x06, c'est-à-dire bit d'erreur 6

### 16.9.5 Création de textes de message dans une autre langue dans la 'Signalisation d'erreurs système'

Vous pouvez afficher les messages configurés dans la "Signalisation d'erreurs système" dans les langues dans lesquelles vous avez installé STEP 7.

Procédez de la manière suivante :

1. Dans SIMATIC Manager, choisissez la commande de menu **Outils > Langue de visuel...**, puis dans la boîte de dialogue qui s'ouvre, ajoutez la langue souhaitée à votre projet.
2. Confirmez vos paramètres en cliquant sur OK.
3. Dans HW Config, choisissez la commande de menu **Outils > Signalisation des erreurs système...**, puis dans la boîte de dialogue qui s'ouvre, cliquez sur le bouton "Générer".  
**Résultat** : les textes de messages sont créés dans toutes les langues dans lesquelles vous avez installé STEP 7, mais sont cependant uniquement affichés dans la langue que vous avez définie en cliquant sur le bouton "Par défaut" dans la boîte de dialogue "Ajouter, supprimer une langue, choisir une langue par défaut".

#### Exemple

Vous avez installé STEP 7 en allemand, anglais et français et ces langues sont définies dans votre projet. Générez à présent les textes de message comme décrit ci-dessus. Pour afficher les textes de message dans la langue respective, définissez la langue souhaitée comme langue par défaut dans la boîte de dialogue "Ajouter, supprimer une langue, choisir une langue par défaut".

# 17 Contrôle-commande de variables

## 17.1 Configuration de variables pour le contrôle-commande

### Généralités

Avec WinCC, STEP 7 vous offre un moyen aisé d'effectuer le contrôle-commande des grandeurs variables de votre processus ou automate programmable.

L'avantage par rapport aux anciennes méthodes réside dans le fait que vous n'avez plus à réaliser la configuration des données individuellement pour chaque station de contrôle-commande (OS), mais une seule fois dans STEP 7. Vous pouvez transmettre les données configurées dans STEP 7 à la base de données de WinCC en utilisant l'application "Transfert des données vers l'OS" (qui fait partie du progiciel "Process Control System PCS7"), la cohérence et la compatibilité des données avec le visuel étant vérifiée. WinCC utilise les données sous forme de blocs d'image et d'objets graphiques.

Vous pouvez configurer ou modifier les attributs de contrôle-commande suivants dans STEP 7 :

- paramètres d'entrée, de sortie et d'entrée/sortie de blocs fonctionnels,
- mémentos et signaux d'entrée/sortie,
- paramètres de blocs CFC dans des diagrammes CFC.

### Marche à suivre

La marche à suivre pour configurer des variables à contrôler et à commander dépend du langage de programmation ou de configuration choisis ainsi que du type des variables à contrôler et à commander. Mais les étapes suivantes sont toujours nécessaires :

1. Affectez des attributs système pour le contrôle-commande aux paramètres d'un bloc fonctionnel ou aux mnémoniques dans une table des mnémoniques.

Cette étape s'avère inutile dans CFC, car vous prenez des blocs déjà configurés dans une bibliothèque.

2. Dans une fenêtre d'édition, affectez aux variables à contrôler et à commander les attributs de contrôle-commande requis (S7\_m\_c). La boîte de dialogue "Contrôle-commande" (commande **Edition > Propriétés spécifiques de l'objet > Contrôle-commande**) vous permet de modifier des attributs WinCC tels que valeurs limite, valeurs de remplacement, caractéristiques du journal, etc.
3. Avec "Transfert des données vers l'OS", transférez dans votre visuel (WinCC) les données de configuration créées dans STEP 7.

## Convention pour l'attribution de noms

Afin de pouvoir être sauvegardées et transférées, les données de configuration pour WinCC sont enregistrées sous un nom univoque attribué automatiquement par STEP 7. Ce nom est formé à partir des noms des variables à contrôler et à commander, des noms des diagrammes CFC et de ceux des programmes S7 qui doivent de ce fait adopter des conventions particulières :

- Les noms des programmes S7 doivent être univoques au sein d'un projet S7 (diverses stations ne doivent pas contenir des programmes S7 de même nom).
- Les noms des variables, programmes S7 et diagrammes CFC ne doivent comporter ni caractère de soulignement, ni caractère d'espacement ou l'un des caractères spéciaux suivant : [ ' ] [ . ] [ % ] [ - ] [ / ] [ \* ] [ + ].

## 17.2 Configuration d'attributs de contrôle-commande avec LIST, CONT, LOG

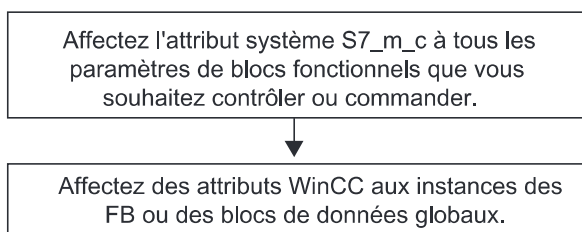
### Introduction

Le procédé que nous allons décrire ci-après va vous permettre de préparer des paramètres de blocs fonctionnels au contrôle-commande et d'affecter les attributs de contrôle-commande requis aux DB d'instance ou blocs de données globaux correspondants dans votre programme utilisateur.

### Condition préalable

Vous avez créé un projet STEP 7, un programme S7 ainsi qu'un bloc fonctionnel (FB).

### Marche à suivre



## 17.3 Configuration des attributs de contrôle-commande au moyen de la table des mnémoniques

### Introduction

Quel que soit le langage de programmation utilisé, le procédé que nous allons décrire ci-après vous permet de configurer les variables suivantes :

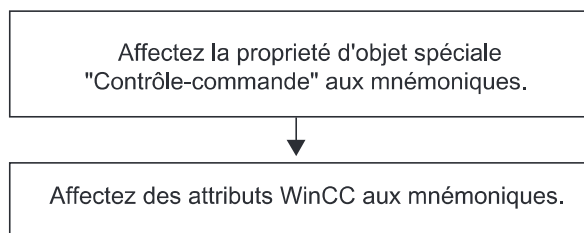
- mémentos
- signaux d'E/S

### Conditions préalables

Avant que vous ne commenciez, les conditions suivantes doivent être satisfaites :

- vous avez créé un projet dans SIMATIC Manager,
- ce projet contient un programme S7 avec une table des mnémoniques,
- la table des mnémoniques est ouverte.

### Marche à suivre



## 17.4 Modification des attributs de contrôle-commande avec CFC

### Introduction

CFC vous permet de créer votre programme utilisateur en sélectionnant des blocs déjà préparés pour le contrôle-commande dans une bibliothèque, puis en les disposant et les reliant dans un diagramme.

### Condition préalable

Vous avez inséré un programme S7 dans un projet STEP 7 et créé un diagramme CFC dans lequel vous avez disposé des blocs.

### Marche à suivre

Editez les propriétés des blocs.

---

### Nota

Si vous utilisez des blocs que vous avez créés vous-même et auxquels vous avez affecté l'attribut système S7\_m\_c dans leur langage de création, vous pouvez les préparer au contrôle-commande en activant la case d'option "Contrôle-commande" dans la boîte de dialogue "Contrôle-commande" (commande **Edition > Propriétés spécifiques de l'objet > Contrôle-commande**).

---

## 17.5 Transfert des données de configuration dans le système cible de contrôle-commande

### Introduction

L'application "Transfert des données vers l'OS" vous permet de transférer les données de configuration créées pour le contrôle-commande dans la base de données de WinCC.

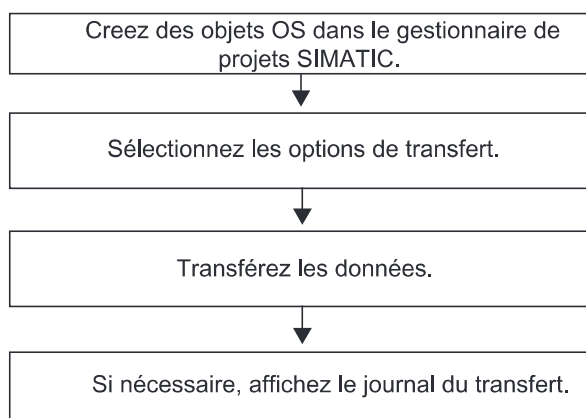
### Condition préalable

Pour pouvoir commencer le transfert, il faut que les conditions suivantes soient satisfaites :

- vous avez installé le programme AS-OS-Engineering,
- vous avez créé les données de configuration pour le contrôle-commande.

### Marche à suivre

Pour transférer les données de configuration pour le contrôle-commande dans la base de données de WinCC, procédez de la manière suivante :





## **18 Etablissement d'une liaison en ligne et choix de la CPU**

### **18.1 Etablissement de liaisons en ligne**

Une liaison en ligne est requise entre l'outil de développement et le système cible pour le chargement de programmes utilisateurs S7 et de blocs, le chargement de blocs depuis le système cible S7 dans l'outil de développement ainsi que pour les tâches suivantes :

- Test de programmes utilisateur
- Affichage et modification de l'état de fonctionnement de la CPU
- Affichage et réglage de l'heure et de la date de la CPU
- Affichage de l'état du module
- Comparaison en ligne/hors ligne de blocs
- Diagnostic du matériel

Pour qu'une liaison en ligne puisse être établie, l'outil de développement et le système cible doivent être reliés entre-eux par l'intermédiaire d'une interface appropriée (par exemple une interface multipoint (MPI)). Vous pouvez ensuite accéder au système cible depuis la fenêtre en ligne du projet ou depuis la fenêtre "Partenaires accessibles".

### 18.1.1 Etablissement d'une liaison en ligne depuis la fenêtre "Partenaires accessibles"

Ce mode d'accès permet une maintenance rapide. Vous pouvez accéder à tous les modules programmables du réseau. Choisissez cette méthode en cas d'absence de données de projet relatives au système cible sur votre PG.

Vous ouvrez la fenêtre "Partenaires accessibles" en choisissant la commande **Système cible > Afficher les partenaires accessibles**. L'objet "Partenaires accessibles" affiche tous les participants accessibles dans le réseau avec leur adresse.

Même les partenaires que vous ne pouvez pas programmer avec STEP 7 sont affichés (par exemple les consoles de programmation ou les pupitres opérateur).

Les informations supplémentaires suivantes peuvent également être affichées entre parenthèses :

- (directe) : ce participant est directement connecté à l'outil de développement (PG ou PC).
- (passif) : la programmation et la visualisation/forçage via PROFIBUS DP ne sont pas possibles pour ce participant.
- (attend) : il n'est pas possible de communiquer avec ce participant, car sa configuration n'est pas compatible avec les autres paramètres du réseau.

#### Détermination du participant directement connecté

L'information supplémentaire "directe" n'est pas disponible pour les participants PROFINET. Afin de pouvoir identifier malgré tout le participant directement connecté, choisissez la commande de menu **Système cible > Diagnostic/Paramètres > Test du clignotement**.

Dans la boîte de dialogue qui s'ouvre, vous pouvez paramétrer la durée de clignotement et lancer le test du clignotement. Le participant directement connecté se reconnaît à sa LED FORCE qui clignote.

Le test du clignotement ne peut pas être réalisé lorsque la fonction de FORCAGE PERMANENT est active.

### 18.1.2 Etablissement d'une liaison en ligne depuis la fenêtre en ligne du projet

Choisissez cette méthode si vous avez configuré le système cible dans un projet sur votre PG/PC. Vous ouvrez la fenêtre en ligne en choisissant la commande **Affichage > En ligne** dans SIMATIC Manager. Elle affiche les données de projet sur le système cible (contrairement à la fenêtre hors ligne qui affiche les données de projet sur votre PG/PC). La fenêtre en ligne affiche les données qui se trouvent sur le système cible, aussi bien pour le programme S7, que pour le programme M7.

Vous utilisez la vue du projet en ligne pour accéder au système cible. Ainsi, vous pouvez appeler certaines commandes du menu "Système cible" de SIMATIC Manager dans la fenêtre en ligne, mais pas dans la fenêtre hors ligne.

On distingue :

- **Accès lorsque le matériel est configuré**  
Dans ce mode d'accès, vous ne pouvez accéder qu'aux modules configurés hors ligne. Les modules en ligne auxquels vous pouvez accéder sont déterminés par l'adresse MPI définie lors de la configuration des modules programmables.
- **Accès lorsque le matériel n'est pas configuré**  
Dans ce mode d'accès, la présence d'un programme S7 ou d'un programme M7 créés indépendamment du matériel (c'est-à-dire se trouvant directement sous le projet) est impérative. Vous déterminez les modules en ligne auxquels vous pouvez accéder en indiquant leur adresse MPI respective dans les propriétés de l'objet du programme S7/M7.

L'accès depuis la fenêtre en ligne combine les données sur le système cible avec les données correspondantes sur l'outil de développement. Si vous ouvrez par exemple un bloc S7 dans un projet en ligne, l'affichage se composera :

- du code du bloc dans la CPU du système cible S7 et
- des commentaires et des mnémoniques qui font partie des données dans l'outil de développement (s'ils existent hors ligne). Si vous ouvrez des blocs directement dans la CPU connectée, hors structure de projet, ils seront affichés tels qu'ils se trouvent dans la CPU, c'est-à-dire sans mnémoniques ni commentaires.

### 18.1.3 Accès en ligne aux systèmes cibles dans le multiprojet

#### Accès à plusieurs projets avec une PG ou un PC affectés

La fonction "Affectation PG/PC" est également disponible dans le multiprojet pour les objets "PG/PC" et "Station SIMATIC PC".

Vous pouvez sélectionner le module cible pour l'accès en ligne dans un projet quelconque du multiprojet. La procédure est identique à celle n'utilisant qu'un seul projet.

#### Conditions

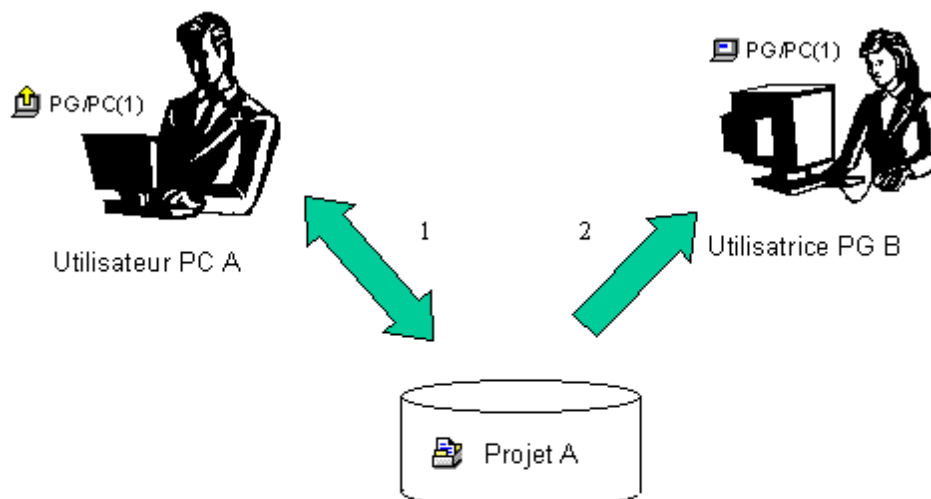
- Pour les PG/PC ou stations PC devant accéder en ligne à des systèmes cibles, l'affectation doit se trouver dans un projet quelconque au sein du multiprojet.  
Nota : la PG, le PC ou la station PC affectés sont mis en valeur en jaune lorsque le projet correspondant est ouvert.  
Une affectation PG/PC est uniquement visible lorsque l'affectation de la PG ouvrant le projet est correcte.
- Les sous-réseaux qui s'étendent sur plusieurs projets sont regroupés.
- Tous les projets du multiprojet sont compilés et l'information de configuration est chargée dans les stations correspondantes, de manière à ce que toutes les informations de routage, p. ex., soient disponibles pour tous les modules participant à l'établissement de la liaison entre votre PG/PC et le module cible.
- Le module cible est accessible via le réseau.

### Eventuels problèmes des projets répartis

Lorsque les attributions de projets changent et qu'un projet est ouvert sur une PG ou un PC sur lequel il n'a pas été créé, l'affectation PG/PC n'est pas visible.

L'objet PG/PC configuré possède bien la propriété "affecté" – mais pas avec la PG ou le PC "corrects".

Dans ce cas, vous devez d'abord annuler l'affectation existante, puis réaliser une nouvelle affectation de l'objet PG/PC. L'accès en ligne aux modules accessibles dans le multiprojet est alors possible sans restrictions.



1. Enregistrer le projet A dans le réseau avec le PG/PC affecté
2. Ouvrir le même projet A avec un autre ordinateur

### Conseil en cas de projets partagés

Lorsque plusieurs collaborateurs accèdent avec des PG en ligne à des systèmes cibles, il est recommandé de configurer, dans le multiprojet, un objet "PG/PC" ou "Station SIMATIC PC" pour chacune de ces PG, puis de créer une affectation pour chaque PG.

En fonction de PG qui ouvre le projet, seul l'objet qui lui est affecté est représenté par une flèche jaune dans SIMATIC Manager.

### 18.1.4 Protection par mot de passe contre l'accès aux systèmes cible

La protection par mot de passe vous permet

- de protéger le programme utilisateur et ses données dans la CPU contre les modifications involontaires (protection en écriture),
- de préserver le savoir faire (know how) contenu dans votre programme utilisateur (protection en lecture),
- d'empêcher les fonctions en ligne qui gêneraient le processus

Vous ne pouvez protéger un module ou le contenu d'une MMC (par exemple pour une CPU 31xC) par un mot de passe que si le module possède cette fonctionnalité.

Si vous souhaitez protéger un module ou le contenu d'une MMC par un mot de passe, vous devez définir le niveau de protection et le mot de passe dans le cadre du paramétrage du module, puis charger le paramétrage modifié dans le module.

Lorsque la saisie d'un mot de passe est requise pour l'exécution d'une fonction en ligne ou l'accès au contenu d'une MMC, la boîte de dialogue "Saisie du mot de passe" s'affiche. La saisie du mot de passe correct vous donne l'autorisation d'accéder à des modules pour lesquels vous avez défini un niveau de protection particulier dans le cadre de leur paramétrage. Vous avez alors la possibilité d'établir des connexions en ligne avec le module protégé et d'exécuter les fonctions en ligne correspondant au niveau de protection.

La commande **Système cible > Droit d'accès > Instaurer** vous permet d'appeler directement la boîte de dialogue de saisie du mot de passe. Vous pouvez ainsi par exemple changer votre mot de passe en début de session et vous n'aurez plus à l'entrer par la suite lors d'accès en ligne. Ce mot de passe reste valide jusqu'à la fin de la session dans le SIMATIC Manager ou son annulation avec la commande de menu **Système cible > Droit d'accès > Annuler**.

Paramètres de la CPU	Observations
Mode de test / mode de processus (pas pour S7-400 ni CPU 318-2)	<p>Paramétrable dans la page d'onglet "Protection".</p> <p>En mode de processus, les fonctions de test comme "Etat du programme" ou "Visualisation/forçage de variables" sont restreintes, afin que l'augmentation du temps de cycle autorisée ne soit pas dépassée. Ainsi, par exemple, pour la fonction "Etat du programme", les conditions d'appel ne sont pas autorisées et la visualisation d'état d'une boucle programmée est interrompue à la position de retour.</p> <p>Le test utilisant des points d'arrêt et l'exécution pas à pas du programme n'est pas possible en mode de processus.</p> <p>En mode de test, toutes les fonctions de test via PG/PC sont possibles sans restrictions, même celles qui sont susceptibles d'entraîner une augmentation importante du temps de cycle.</p>
Niveau de protection	Sélectionnable dans la page d'onglet "Protection". Vous pouvez soumettre les accès en lecture et en écriture à la CPU à la saisie d'un mot de passe, qui doit être défini dans cette page d'onglet.

### **18.1.5 Remarque sur l'actualisation du contenu de la fenêtre**

Sachez que :

- des modifications dans la fenêtre en ligne d'un projet, résultant d'une manipulation de l'utilisateur (par exemple chargement ou effacement de blocs) ne sont pas automatiquement reprises dans une fenêtre "Partenaires accessibles" éventuellement ouverte.
- des modifications dans la fenêtre "Partenaires accessibles" ne sont pas automatiquement reprises dans une fenêtre du projet en ligne éventuellement ouvert.

Pour obtenir un affichage actuel dans une fenêtre ouverte simultanément, vous devez également actualiser cette fenêtre (à l'aide de la commande de menu ou de la touche F5).

## 18.2 Affichage et modification de l'état de fonctionnement

Cette fonction vous permet, par exemple, de remettre la CPU à l'état de marche (RUN) après avoir corrigé une erreur.

### Affichage de l'état de fonctionnement

1. Ouvrez votre projet et sélectionnez un programme S7/M7 ou ouvrez la fenêtre "Partenaires accessibles" en choisissant la commande **Système cible > Afficher partenaires accessibles** et sélectionnez un partenaire (MPI=...).
2. Choisissez la commande **Système cible > Diagnostic/Paramètres > Etat de fonctionnement**.

Cette boîte de dialogue affiche le dernier et le plus actuel état de fonctionnement du module, de même que la position actuelle de son commutateur à clé. Pour les modules sur lesquels la position du commutateur à clé ne peut pas être lue, le texte "indéfini" s'affiche.

### Modification de l'état de fonctionnement

Vous pouvez modifier l'état de fonctionnement de la CPU à l'aide des boutons. Les boutons actifs sont ceux que vous avez la possibilité de sélectionner à l'état de fonctionnement actuel.



## 18.3 Affichage et réglage de l'heure et de la date

### 18.3.1 Horloges CPU avec réglage des zones horaires et heure d'été/heure d'hiver

A partir de la version STEP 7 V5.1, Servicepack 2, vous pouvez également effectuer et/ou exploiter les réglages suivants dans le cas des nouvelles CPU (à partir de la version 3 de firmware), outre l'heure/la date :

- Heure d'été/d'hiver
- Facteur de correction pour la présentation des zones horaires

#### Représentation des zones horaires

Une heure unique et sans interruption, à savoir l'heure du module, est prévue sur toute l'installation.

Une heure locale supplémentaire, différente de l'heure du module est calculée au niveau local dans le système d'automatisation et est utilisée par le programme utilisateur. L'heure locale n'est pas directement saisie, mais se calcule à partir de l'heure du module, en sus et/ou en déduction d'un décalage horaire par rapport à l'heure du module.

#### Heure d'été/d'hiver

Lors du réglage de l'heure et de la date, vous pouvez également régler l'heure d'été ou d'hiver. Lorsque l'on passe de l'heure d'été à l'heure d'hiver, par exemple pour chaque programme utilisateur, cela entre en ligne de compte uniquement pour le décalage horaire par rapport à l'heure du module. Vous pouvez procéder à la commutation au moyen d'un bloc qui est mis à votre disposition sur Internet.

#### Lecture et réglage de l'heure et de l'état de l'heure

L'identification de l'heure d'été/d'hiver ainsi que le décalage horaire par rapport à l'heure du module figurent dans l'état de l'heure.

Vous disposez des possibilités suivantes pour la lecture et/ou le réglage de l'heure et de l'état de l'heure :

Avec STEP 7 (en ligne)

- Par la commande **Système cible > Diagnostic/Paramètres > Régler l'heure** (lecture et réglage)
- Par la boîte de dialogue "Etat du module", onglet "Horodatage" (lecture uniquement)

dans le programme utilisateur

- SFC 100 «SET\_CLKS» (lecture et réglage)
- SFC 51 «RDSYSST» avec SZL 132, index 8 (lecture uniquement)

## **Horodatage dans le tampon de diagnostic, dans les messages et les informations de déclenchement des OB**

Les horodatages sont créés avec l'heure du module.

### **Alarmes horaires**

Si, suite au passage de l'heure d'hiver à l'heure d'été, les alarmes horaires n'ont pas été déclenchées en raison du changement d'heure, l'OB 80 est alors appelé.

Lors du passage de l'heure d'été à l'heure d'hiver, la périodicité est conservée pour les alarmes horaires avec périodicité des minutes et des heures.

### **Synchronisation de l'heure**

Une CPU qui est paramétrée comme maître de synchronisation (par exemple, dans l'onglet CPU "Diagnostic/Heure") synchronise toujours d'autres horloges avec l'heure du module et leur état actuel de l'heure.

## **18.4 Mise à jour du microprogramme**

### **18.4.1 Mise à jour en ligne du microprogramme de modules**

A partir du Servicepack 3 de STEP 7 V5.1, vous pouvez effectuer la mise à jour en ligne des modules d'une station de manière uniforme. La procédure est décrite ci-après.

#### **Concept**

Des fichiers (\*.UPD) sont fournis avec le microprogramme actuel afin de vous permettre d'effectuer la mise à jour du microprogramme d'un module tel que p. ex. une CPU, un CP ou un IM.

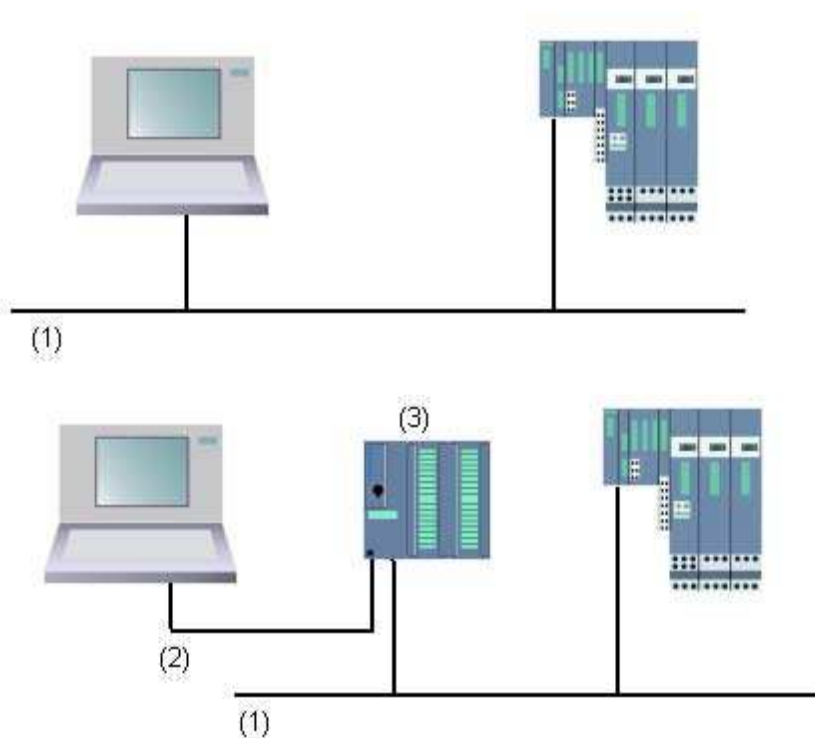
Il vous suffit de sélectionner l'un de ces fichiers et de le charger dans le module (menu Système cible).

#### **Conditions**

Le module dont vous souhaitez mettre à jour le microprogramme doit être accessible en ligne dans la station, p. ex. lorsque la PG est connectée au même réseau MPI, PROFIBUS ou Ethernet que le module dont vous souhaitez mettre à jour le microprogramme. Une mise à jour du microprogramme est également possible lorsque la PG est connectée à l'interface MPI de la CPU maître DP et le module, dont vous souhaitez mettre à jour le microprogramme, au réseau PROFIBUS de l'interface DP ou au réseau Ethernet de l'interface PN. A cet effet, la CPU doit prendre en charge le routage S7 entre l'interface MPI et l'interface DP ou entre l'interface MPI et l'interface PN.

Le module doit prendre en charge la mise à jour du microprogramme.

Les fichiers contenant les versions actuelles du microprogramme doivent être disponibles dans le système de fichiers de votre PG/PC. Chaque dossier doit uniquement contenir les fichiers pour **une** version de firmware.



(1) Sous-réseau PROFIBUS ou Ethernet

(2) Sous-réseau MPI

(3) CPU avec interface MPI et interface DP ou interface PN (avec routage S7)

### Procédure dans HW Config

1. Ouvrez la station dans laquelle se trouve le module à mettre à jour.
2. Sélectionnez ce module.  
Dans le cas de modules de couplage PROFIBUS DP, comme p. ex. un IM 151, sélectionnez l'icône de l'esclave DP, dans le cas présent l'ET 200S correspondant.
3. Choisissez la commande **Système > Actualiser le firmware**.  
Cette commande de menu peut uniquement être activée si le module ou l'esclave DP sélectionnés prennent en charge la fonction "Actualiser le firmware".
4. Dans la boîte de dialogue "Actualiser le firmware" qui s'ouvre, naviguez jusqu'aux fichiers de mise à jour du microprogramme (\*.UPD) en cliquant sur le bouton "Parcourir".
5. Une fois que vous avez sélectionné un fichier, une information s'affiche dans les champs de la partie inférieure de la boîte de dialogue "Actualiser le firmware", spécifiant les modules pour lesquels ce fichier est valable et à partir de quelle version de microprogramme.

6. Cliquez sur le bouton "Exécuter".  
STEP 7 vérifie que le fichier sélectionné peut être interprété par le module et, si cela est le cas, le charge dans le module.  
Si l'état de fonctionnement de la CPU doit être modifié à cet effet, des dialogues vous y sollicitent.  
Le module effectue ensuite automatiquement la mise à jour du microprogramme.  
**Nota** : pour la mise à jour du microprogramme, p. ex. d'une CPU 317-2 PN/DP, une liaison indépendante est en général établie avec la CPU. Dans ce cas, la procédure peut être annulée. Si aucune ressource n'est disponible pour établir une autre liaison, une liaison existante est automatiquement utilisée. Dans ce cas, la procédure ne peut pas être annulée ; le bouton "Annuler" est représenté de manière estompée dans la boîte de transfert et ne peut pas être activé.
7. Vérifiez dans STEP 7 (lecture de la mémoire tampon de la CPU), si le module fonctionne correctement avec son nouveau microprogramme.

### Procédure dans SIMATIC Manager

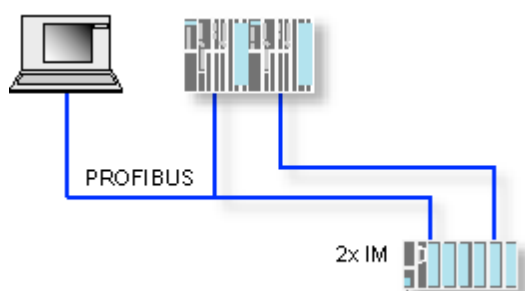
La fonction peut être activée si la fenêtre „Partenaires accessibles" est ouverte. La procédure correspond à celle de HW Konfig et la commande de menu s'appelle également **Système cible > Mise à jour du firmware**. STEP 7 ne vérifie cependant qu'au moment de l'exécution de la fonction si celle-ci prend le module en charge.

### Mise à jour du firmware pour les modules en mode redondant

A partir de STEP 7 V5.4, la mise à jour du firmware est prise en charge pour les modules en mode redondant, p. ex. pour l'IM 153-2BA00 avec bus de fond de panier actif sur une station H. Pour les IM redondants, vous pouvez réaliser une mise à jour du firmware en une seule procédure ; l'IM redondant reçoit automatiquement la nouvelle version du firmware.

Condition : la PG est connectée au même PROFIBUS que l'un des IM et vous réalisez la mise à jour du firmware via "Partenaires accessibles" dans SIMATIC Manager.

### Principe



### **Conséquences lors de l'actualisation du firmware durant le fonctionnement**

Dans la boîte de dialogue de l'actualisation du firmware, une option vous permet de définir que le nouveau firmware doit être activé immédiatement après son actualisation.

Lorsque cette option est activée, la station réalise une remise en route comme après une MISE HORS TENSION/MISE SOUS TENSION. Une CPU peut p. ex. rester à l'état STOP ou l'exécution du programme utilisateur peut en être erronée. En raison de ces restrictions de fonctionnement, vous devez prendre des mesures préventives.

En cas de redémarrage, cela entraîne la défaillance de nombreux modules de la station - y compris de la périphérie de sécurité éventuellement présente.

Lors d'une MISE HORS TENSION, la périphérie de sécurité signale une erreur de communication au coupleur et réalise un arrêt de sécurité - elle est passivée. Cette passivation n'est pas annulée lors du redémarrage du coupleur. Vous devez dépassiver les modules individuellement. Ceci a pour conséquence que les applications de sécurité ne fonctionnent pas.



# 19 Chargement

## 19.1 Chargement dans le système cible depuis la PG

### 19.1.1 Conditions préalables au chargement

#### Conditions préalables au chargement dans le système cible

- Une liaison est établie entre votre PG et la CPU du système cible (par exemple via l'interface MPI).
- L'accès au système cible est possible.
- Lors du chargement de blocs dans le système cible, l'entrée "STEP 7" doit être sélectionnée comme "Utilisation" dans les propriétés de l'objet du projet.
- La compilation du programme à charger s'est faite sans erreur.
- La CPU doit se trouver dans un état de fonctionnement autorisant le chargement (arrêt ou marche (RUN-P)).  
A l'état de fonctionnement de marche (RUN-P), veillez à charger le programme bloc par bloc. En effet, des conflits risqueraient de survenir si vous écrasiez un ancien programme CPU et que, par exemple, des paramètres de bloc seraient modifiés. Durant l'exécution du cycle, la CPU passerait alors à l'état d'arrêt. C'est pourquoi, il est recommandé de mettre la CPU à l'état d'arrêt avant de réaliser le chargement.
- Si vous avez ouvert hors ligne le bloc que vous voulez charger, un programme utilisateur en ligne doit être affecté à la CPU dans SIMATIC Manager.
- Nous vous recommandons d'effectuer un effacement général de la CPU avant de charger votre programme utilisateur, afin d'être sûr qu'il n'y a plus d'anciens blocs dans la CPU.

#### Etat de fonctionnement "Arrêt" (STOP)

Vous devez faire passer la CPU de l'état de fonctionnement "Marche" (RUN) à l'état "Arrêt" (STOP) avant :

- de charger le programme utilisateur complet ou certaines de ses parties dans la CPU,
- d'effectuer un effacement général de la CPU,
- de comprimer la mémoire utilisateur.

## Démarrage à chaud (passage à l'état de fonctionnement "Marche")

Lorsque vous demandez un démarrage à chaud à partir de l'état de fonctionnement "Arrêt" (STOP), le programme recommence au début et le programme de mise en route - dans le bloc OB100 - est d'abord exécuté en mode mise en route. Si la mise en route s'achève sans erreur, la CPU passe à l'état "Marche" (RUN). Un démarrage à chaud est nécessaire après :

- l'effacement général de la CPU,
- le chargement du programme utilisateur à l'état "Arrêt" (STOP).

### 19.1.2 Différence entre l'enregistrement et le chargement de blocs

L'enregistrement et le chargement de blocs présentent une différence fondamentale.

	Enregistrement	Chargement
Commande	Fichier > Enregistrer Fichier > Enregistrer sous	Système cible > Charger
Fonction	L'état en cours du bloc de l'éditeur est sauvegardé sur le disque dur de la PG.	L'état en cours du bloc de l'éditeur est chargé seulement dans la CPU.
Vérification de la syntaxe	La syntaxe est vérifiée. Le cas échéant, les erreurs vous seront signalées dans des boîtes de dialogue, avec indication de la cause et de l'emplacement des erreurs. Vous devez corriger ces erreurs avant d'enregistrer ou de charger le bloc. Si la syntaxe est correcte, le bloc est ensuite compilé en code machine et enregistré ou chargé.	La syntaxe est vérifiée. Le cas échéant, les erreurs vous seront signalées dans des boîtes de dialogue, avec indication de la cause et de l'emplacement des erreurs. Vous devez corriger ces erreurs avant d'enregistrer ou de charger le bloc. Si la syntaxe est correcte, le bloc est ensuite compilé en code machine et enregistré ou chargé.

Ce tableau s'applique aussi bien à des blocs ouverts en ligne qu'hors ligne:

### Remarque sur l'enregistrement de modifications de blocs préalable au chargement

Pour valider les blocs nouvellement créés ou les modifications apportées à la section des instructions de blocs de code, aux tables de déclaration ou aux valeurs dans les blocs de données, vous devez enregistrer les blocs correspondants. Il est indispensable d'enregistrer également sur le disque dur de la PG, avant de quitter l'éditeur, les modifications que vous avez faites et que vous avez transférées directement dans la CPU avec la commande **Système cible > Charger** – par exemple, parce qu'elles étaient minimales et que vous vouliez les tester immédiatement. Sinon en effet, vous aurez des versions différentes de votre programme utilisateur dans la CPU et dans la console de programmation. En général, il est conseillé de toujours enregistrer d'abord les modifications et de ne les charger qu'ensuite.



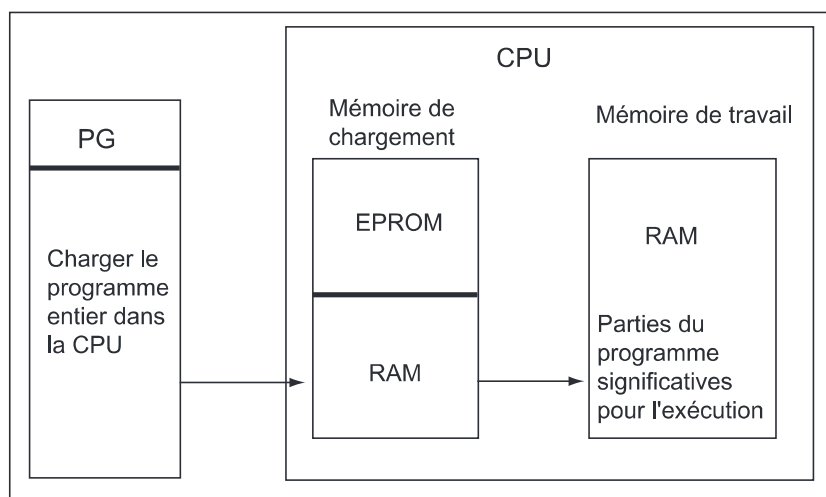
### 19.1.3 Mémoire de chargement et mémoire de travail dans la CPU

Une fois la configuration, le paramétrage et l'écriture du programme achevés et la liaison en ligne établie, vous pouvez transférer des programmes utilisateur entiers ou des blocs individuels dans un système cible. Vous devez, pour tester des blocs individuels, charger au moins un OB ainsi que les FB et les FC qui y sont appelés et les DB utilisés. Pour transférer dans le système cible les données système obtenues après la configuration du matériel et des réseaux ou la création d'une table des liaisons, vous chargez l'objet "Blocs de données système".

Dans le SIMATIC Manager, vous chargez des programmes utilisateur dans un système cible, par exemple dans la phase finale du test du programme ou pour exécuter le programme utilisateur achevé.

#### Relation entre la mémoire de chargement et la mémoire de travail de la CPU

Le programme utilisateur entier est chargé dans la mémoire de chargement, les parties du programme significatives pour l'exécution étant également dans la mémoire de travail.



#### Mémoire de chargement de la CPU

- La mémoire de chargement sert à l'enregistrement du programme utilisateur sans table des mnémoniques ni commentaires (ces derniers restent dans la mémoire de la console de programmation).
- Les blocs identifiés comme non significatifs pour l'exécution sont exclusivement chargés dans la mémoire de chargement.
- Selon le système cible, il peut s'agir pour la mémoire de chargement de mémoire vive (RAM), de mémoire morte (ROM) ou de mémoire EPROM
- Pour le S7-300, la mémoire de chargement peut comporter une partie EEPROM intégrée en plus de la partie RAM (par exemple CPU312 IFM et CPU314 IFM).
- Pour le S7-400, l'utilisation d'une carte mémoire (RAM ou EEPROM) s'avère indispensable pour l'extension de la mémoire de chargement.

## Mémoire de travail de la CPU

La mémoire de travail (mémoire vive intégrée) contient les parties du programme significatives pour l'exécution du programme.

### Procédures de chargement possibles

La fonction de chargement vous permet de charger le programme utilisateur ou des objets chargeables (par exemple des blocs) dans le système cible. Si un bloc se trouve déjà dans la mémoire vive de la CPU, un message vous demande, lors du chargement, si ce bloc doit être écrasé ou non.

- Vous pouvez sélectionner les objets chargeables dans la fenêtre de projet et les charger à partir de SIMATIC Manager (commande **Système cible > Charger**).
- Lors de la programmation de blocs et lors de la configuration du matériel et de réseaux, vous pouvez charger directement l'objet en cours de traitement via le menu dans la fenêtre principale associée à votre activité (commande **Système cible > Charger**).
- Une autre possibilité consiste à ouvrir une fenêtre en ligne avec la vue du système cible (par exemple via la commande **Affichage > En ligne** ou **Système cible > Afficher les partenaires accessibles**) et de copier l'objet à charger dans la fenêtre en ligne.

Inversement, vous pouvez charger le contenu actuel de blocs de la mémoire vive de chargement de la CPU dans votre console de programmation.

#### 19.1.4 Possibilités de chargement selon la mémoire de chargement

La structure de la mémoire de chargement formée d'une partie RAM et d'une partie EEPROM n'est pas sans conséquence sur les possibilités de chargement de votre programme utilisateur ou de blocs individuels. Le chargement des données dans la CPU peut s'effectuer de plusieurs manières :

Mémoire de chargement	Possibilités de chargement	Type de communication entre outil de développement et système cible
Mémoire vive (RAM)	Chargement et effacement de blocs individuels	Liaison en ligne PG - système cible
	Chargement et effacement d'un programme utilisateur entier	Liaison en ligne PG - système cible
	Chargement de blocs individuels	Liaison en ligne PG - système cible
EPROM intégrée (uniquement S7-300) ou enfichable	Chargement de programmes utilisateur entiers	Liaison en ligne PG - système cible
EPROM enfichable	Chargement de programmes utilisateur entiers	Chargement externe de l'EPROM et enfichage de la carte mémoire ou chargement via la liaison en ligne à l'EPROM enfichée dans le système cible

## Chargement dans la RAM via une liaison en ligne

En cas de panne secteur, les données ne sont pas protégées dans le système cible si la mémoire vive ne possède pas de sauvegarde. Dans ce cas, les données sont perdues dans la RAM.

## Enregistrement sur une carte mémoire EPROM

Les blocs ou le programme utilisateur sont enregistrés sur une carte mémoire EPROM que l'on enfiche ensuite dans un emplacement prévu à cet effet sur la CPU.

Les cartes mémoire sont des supports de données portatifs. Vous y inscrivez les données depuis l'outil de développement, puis les enfichez à l'emplacement prévu à cet effet sur la CPU.

Les données qui y sont sauvegardées sont conservées en cas de coupure de courant ou d'effacement général de la CPU. Après effacement général de la CPU, et après retour du courant suite à une panne secteur quand la mémoire vive n'est pas sauvegardée, le contenu de l'EPROM est à nouveau copié dans la zone de mémoire vive de la mémoire de la CPU.

## Enregistrement dans la mémoire intégrée EPROM

Pour la CPU 312, il existe encore la possibilité de charger le contenu de la mémoire vive dans la mémoire intégrée EPROM, où les données sont rémanentes en cas de panne secteur. Après retour du courant suite à une panne secteur, quand la mémoire vive n'est pas sauvegardée, et après effacement général de la CPU, le contenu de l'EPROM est à nouveau copié dans la zone de mémoire vive de la mémoire de la CPU.

## 19.1.5 Chargement du programme dans la CPU S7

### 19.1.5.1 Chargement dans la gestion du projet

1. Dans la fenêtre du projet, sélectionnez le programme utilisateur ou les blocs à charger.
2. Chargez les objets sélectionnés dans le système cible en choisissant la commande **Système cible > Charger**.

### Autre procédure possible (glisser-lâcher)

1. Ouvrez à la fois une fenêtre hors ligne et une fenêtre en ligne de votre projet.
2. Sélectionnez ensuite les objets à charger dans la fenêtre hors ligne, puis glissez-les dans la fenêtre en ligne.

### 19.1.5.2 Chargement hors gestion du projet

1. Choisissez la commande **Système cible > Partenaires accessibles** ou cliquez sur le bouton correspondant dans la barre d'outils pour ouvrir la fenêtre "Partenaires accessibles".
2. Dans la fenêtre "Partenaires accessibles", effectuez un double-clic sur le partenaire correspondant ("MPI=...") pour voir le dossier Blocs.
3. Ouvrez la bibliothèque ou le projet, à partir desquels vous voulez charger le programme utilisateur ou les blocs dans le système cible. Choisissez à cet effet la commande **Fichier > Ouvrir**.
4. Dans la fenêtre du projet ou de la bibliothèque ouverte, sélectionnez les objets que vous voulez charger.
5. Chargez ces objets dans le système cible, en les copiant à l'aide de la souris ou des commandes de menu sur le dossier Blocs dans la fenêtre "Partenaires accessibles".

### 19.1.5.3 Chargement de blocs dans le système cible

Vous pouvez écraser les blocs présents dans la mémoire de chargement (mémoire vive) ou dans la mémoire de travail de la CPU du système cible S7 avec une nouvelle version (chargement dans le système cible). L'ancienne version des blocs est alors écrasée.

La manière de procéder pour charger des blocs S7 est la même que celle pour les charger, à ceci près que le système vous demandera si vous voulez écraser les blocs présents.

Un bloc enregistré dans une EPROM ne peut être écrasé, mais sera déclaré non valable après le chargement dans le système cible. Le nouveau bloc est chargé dans la mémoire vive. Dans la mémoire de chargement ou dans la mémoire de travail, il en résulte des intervalles, susceptibles d'empêcher le chargement de nouveaux blocs. Dans ce cas, il est recommandé de comprimer les mémoires.

---

#### Nota

En cas de retour de courant suite à une panne secteur, quand la mémoire vive n'est pas sauvegardée, ou en cas d'effacement général de la CPU, les "anciens blocs" de l'EPROM seront à nouveau valables et chargés dans le système cible !

---

#### 19.1.5.4 Enregistrement de blocs chargés dans la mémoire intégrée EPROM

Dans les modules CPU disposant d'une EPROM intégrée (par exemple la CPU 312), vous pouvez copier des blocs de la mémoire vive dans l'EPROM intégrée afin de ne pas perdre les données en cas de coupure de courant ou d'effacement général.

1. Affichez une fenêtre avec la vue en ligne d'un projet ouvert en choisissant la commande **Affichage > En ligne** ou alors affichez la fenêtre "Partenaires accessibles" en cliquant sur le bouton "Partenaires accessibles" dans la barre d'outils ou en choisissant la commande **Système cible > Afficher les partenaires accessibles**.
2. Choisissez le programme S7 ou M7 dans la fenêtre en ligne du projet ou le partenaire dans la fenêtre "Partenaires accessibles".
3. Sélectionnez le dossier Blocs de la CPU que vous voulez enregistrer dans :
  - la fenêtre en ligne du projet si vous travaillez dans la gestion du projet.
  - la fenêtre "Partenaires accessibles", si vous travaillez hors gestion du projet.
4. Choisissez la commande **Système cible > Copier de RAM en ROM**.

#### 19.1.5.5 Chargement via des cartes mémoire EPROM

##### Condition préalable

Pour pouvoir accéder depuis l'outil de développement aux cartes mémoire EPROM destinées à un système cible S7, vous devez avoir installé le pilote d'EPROM correspondant. Pour pouvoir accéder aux cartes mémoire EPROM destinées à un système cible M7, vous devez avoir installé le système Flash File (ceci n'est possible que sur les PG720/740/760, Field PG et Power PG). Le pilote d'EPROM et le système Flash File sont proposés en option lors de l'installation du logiciel de base STEP 7. Si vous utilisez un PC, vous devez en plus posséder un programmeur d'EPROM externe pour pouvoir réaliser l'enregistrement sur une carte mémoire EPROM.

Vous pouvez également installer les pilotes ultérieurement. Ouvrez à cet effet la boîte de dialogue correspondante via la barre des tâches (**Démarrer > Simatic > STEP 7 > Paramétrage de cartes mémoire**) ou via le panneau de configuration (double clic sur l'icône "Paramétrage de cartes mémoire").

## Sauvegarde sur carte mémoire

Pour sauvegarder des blocs ou des programmes utilisateur sur une carte mémoire, procédez de la manière suivante :

1. Enfichez la carte mémoire dans l'emplacement prévu à cet effet sur votre outil de développement.
2. Ouvrez la fenêtre "Carte mémoire S7" de la manière suivante :
  - Cliquez sur le bouton de la carte mémoire dans la barre d'outils. Si cette dernière n'est pas affichée, vous pouvez le faire en choisissant la commande **Affichage > Barre d'outils**.
  - Une alternative consisterait à choisir la commande **Fichier > Carte mémoire S7 > Ouvrir**.
3. Ouvrez ou activez la fenêtre dans laquelle vous allez afficher les blocs à enregistrer. Il peut s'agir d'une :
  - fenêtre du projet, vue du projet en ligne
  - fenêtre du projet, vue du projet hors ligne
  - fenêtre de bibliothèque
  - fenêtre "Partenaires accessibles"
4. Sélectionnez le dossier Blocs ou les blocs à enregistrer, puis copiez-les dans la fenêtre "Carte mémoire S7".
5. Si un bloc se trouve déjà sur la carte mémoire, un message d'erreur est émis. Dans ce cas, effacez le contenu de la carte mémoire et répétez les étapes à partir de l'étape 2.

## 19.2 Compilation et chargement de plusieurs objets depuis la PG

### 19.2.1 Conditions et remarques pour le chargement

#### Chargement de dossiers Bloc

Dans les dossiers Bloc, il est uniquement possible de charger les blocs de code. Les autres objets du dossier Bloc, tels que les données système (SDB), etc. ne peuvent pas être chargés à partir de là. Les SDB sont chargés à partir de l'objet "Matériel".

---

#### Nota

Pour les projets PCS 7, le chargement de blocs n'est possible ni depuis SIMATIC Manager, ni depuis la boîte de dialogue "Compiler et charger les objets". En effet, pour les projets PCS 7, le chargement des systèmes cibles ne peut être réalisé que dans CFC afin de garantir un ordre de chargement correct et ainsi éviter le passage en STOP de la CPU.

Pour savoir s'il s'agit d'un projet PCS 7, veuillez consulter les propriétés spécifiques de l'objet du projet.

#### Chargement de parties F de commandes de sécurité

Pour des raisons de sécurité, le chargement de parties F de commandes de sécurité est uniquement possible après saisie d'un mot de passe. Dans "Compilation et chargement d'objets", la procédure de chargement est de ce fait interrompue par un message d'erreur. Le chargement des parties de programme correspondantes dans le système cible doit être réalisé avec le progiciel optionnel.

#### Chargement du matériel

Le chargement du matériel (c'est-à-dire le chargement des SDB hors ligne) au moyen de la fonction "Compilation et chargement d'objets" peut uniquement être exécuté sans interruption pour tous les objets sélectionnés si aucun message d'erreur ou interrogations ne sont émis. Des informations sur la façon d'éviter des messages d'erreur et des interrogations sont données au paragraphe suivant.

#### Conditions requises pour le chargement du matériel

- Les CPU doivent se trouver à l'état de fonctionnement ARRET.
- Les liaisons en ligne avec les CPU doivent être possibles ; les CPU protégées par mot de passe requièrent une liaison autorisée ou la saisie d'un mot de passe. Pour saisir le mot de passe, cliquez sur le bouton "Edition" lorsque la CPU est sélectionnée ou sélectionnez le dossier Bloc avant de démarrer la fonction "Compilation et chargement d'objets".
- Certains paramétrages ne sont pas autorisés pour l'interface du système cible via laquelle le chargement est réalisé :
  - l'adresse de l'interface ne doit pas être modifiée,
  - une modification des paramètres de réseau risque d'empêcher l'accès à certains modules.

- Dans le cas des CPU H, vous pouvez sélectionner la CPU à charger (CPU H 0 ou CPU H 1) avant le démarrage de la fonction "Compilation et chargement d'objets" (sélectionnez l'objet "CPU" et cliquez sur le bouton "Edition").
- Les paramètres de CPU suivants ne doivent pas avoir été modifiés :
  - la taille maximale pour les données locales et les ressources de communication de la CPU (onglet "Mémoire"),
  - la protection par mot de passe de la CPU F (onglet "Protection").
- Pour chaque module configuré, les conditions suivantes doivent être remplies :
  - le numéro de référence du module configuré doit être identique à celui du module effectivement enfiché,
  - la version du firmware du module configuré ne doit pas être supérieure à celle du module effectivement enfiché,
  - le nom de la station, le nom du module et le repère d'installation ne doivent pas avoir été modifiés depuis le dernier chargement. La nouvelle attribution d'un repère d'installation est autorisée.

### Remarques sur le déroulement du chargement

- Tous les SDB hors ligne sont chargés (c'est-à-dire outre la configuration matérielle, également les SDB de liaison et les SDB résultant de la configuration des données globales).
- Le chargement est uniquement effectué si précédemment aucune erreur n'est survenue lors de la compilation.
- Durant le chargement les messages sont bloqués ; en cas d'insuffisance de mémoire de la CPU, une compression est p. ex. effectuée sans demande de confirmation.
- Après le chargement, les modules chargés restent à l'état d'ARRET (à l'exception des modules qui sont automatiquement arrêtés et redémarrés sans demande de confirmation).

### Conseil

Si, après le chargement, un message signale que le chargement de l'objet a été réalisé avec des avertissements, il est indispensable de consulter le journal. Il est probable que l'objet n'ait pas été chargé ou l'ait été partiellement.



## 19.2.2 Compilation et chargement d'objets

Dans la boîte de dialogue "Compilation et chargement d'objets" vous préparez les objets pouvant être sélectionnés dans votre projet ou multiprojet pour leur chargement dans le système cible, puis les chargez dans ce dernier si vous le désirez. Cette boîte de dialogue peut être utilisée pour les objets d'une station, d'un projet ou d'un multiprojet.

Selon l'objet sélectionné, certaines informations ne peuvent pas être affichées ou de nombreuses fonctions décrites ci-après ne sont pas disponibles pour ces objets. Des restrictions peuvent survenir, en particulier pour des objets créés par des progiciels optionnels.

Pour les blocs d'un dossier Blocs, la "compilation" signifie que la cohérence des blocs est vérifiée. Pour plus de facilité, la vérification de cohérence des blocs sera par la suite décrite comme procédure de compilation.

Procédez de la manière suivante :

1. Dans SIMATIC Manager, sélectionnez l'objet que vous souhaitez compiler ou charger et charger. Vous pouvez sélectionner les objets suivants dans SIMATIC Manager :
  - multiprojet,
  - projet,
  - station,
  - programme S7 non affecté à une station.
2. Dans SIMATIC Manager, choisissez la commande de menu **Système cible > Compiler et charger les objets**.
3. Sélectionnez "Compilation uniquement", si vous souhaitez vérifier les blocs sans les charger dans le système cible. Activez cette option lorsque vous ne souhaitez charger aucun des objets dans le système cible.
4. Si vous souhaitez empêcher le chargement incomplet des stations en raison d'erreurs de compilation, activez la case à cocher "Ne pas charger en cas d'erreur de compilation". Dans ce cas, aucun chargement n'est effectué. Lorsque cette case à cocher n'est pas activée, tous les objets ayant été compilés sans erreur sont chargés. Les objets ayant entraîné une erreur lors de la compilation ne sont pas chargés.
5. Si vous souhaitez compiler et charger des liaisons, activez la case à cocher correspondante pour l'objet "Liaisons".
6. Le multiprojet convient tout particulièrement comme objet d'entrée, car à partir de cet objet, tous les partenaires de liaison entre plusieurs projets peuvent être chargés automatiquement.
7. Dans les colonnes "Compiler" et "Charger", sélectionnez les objets que vous souhaitez compiler ou charger. Ils seront cochés. Si à l'étape 3, vous avez activé l'option "Compilation uniquement", la colonne "Charger" est masquée.
8. Démarrez la procédure de compilation en cliquant sur le bouton "Démarrer".
9. Suivez les instructions qui s'affichent à l'écran.

Après la procédure de compilation ou de chargement, le journal de la procédure complète s'affiche automatiquement. Vous pouvez cependant ouvrir à tout moment le journal de la procédure complète ou le journal d'objets individuels :

- Cliquez sur le bouton "Tout" si vous souhaitez visualiser le journal de la procédure complète.
- Cliquez sur le bouton "Objet individuel", si vous souhaitez visualiser un journal de l'objet sélectionné dans la liste.

### Particularités lors de la compilation et du chargement de liaisons

Si, pour un module, vous sélectionnez l'objet "Liaisons" afin de le **compiler**, STEP 7 sélectionne automatiquement les objets "Liaisons" correspondants des partenaires de liaison. Ainsi, STEP 7 crée toujours des données de configuration cohérentes (blocs de données système). Les objets sélectionnés automatiquement ne peuvent pas être désélectionnés directement manuellement. Leur sélection est toutefois supprimée automatiquement lorsque vous désélectionnez l'objet "Liaisons" initialement sélectionné.

Si, pour un module, vous sélectionnez l'objet "Liaisons" afin de le **charger**, STEP 7 active également automatiquement la case à cocher "Compiler". De plus, STEP 7 active les cases à cocher "Compiler" et "Charger" de tous les partenaires de liaison. Si seuls des objets du type "Liaisons" ont été sélectionnés, vous pouvez également charger les liaisons à l'état de fonctionnement RUN-P de la CPU.

Vous pouvez charger des liaisons individuelles en utilisant NetPro.

### Compilation et chargement du matériel : effets sur les liaisons

Lorsque vous sélectionnez l'objet "Matériel" afin de le **compiler** ou de le **charger**, les objets "Liaisons" placés sous le matériel sélectionné sont automatiquement sélectionnés pour être compilés ou chargés. Dans ce cas, les objets de liaison des partenaires de liaison ne sont **pas** sélectionnés automatiquement !

## 19.3 Chargement depuis le système cible dans la PG

Cette fonction vous permet de réaliser les tâches suivantes :

- enregistrement d'informations du système cible (p. ex. à des fins de maintenance)
- configuration et édition rapides d'une station, lorsque les composants matériels sont présentes au début de la tâche de configuration

### Enregistrement d'informations du système cible

Cette mesure peut s'avérer nécessaire lorsque, par exemple, les données du projet hors ligne correspondant à l'état actuel dans la CPU ne sont pas présentes ou uniquement partiellement. Vous pouvez alors au moins charger la partie des données du projet disponible en ligne dans votre PG.

### Configuration rapide

Vous pouvez vous faciliter la saisie de la configuration de la station en chargeant les données de configuration depuis le système cible dans votre PG, après avoir configuré le matériel et réinitialisé la station (démarrage à chaud). Vous obtenez ainsi la configuration de la station avec les indication de type des différents modules. Il vous suffira ensuite de spécifier précisément les différents modules (numéro de référence) et de les paramétrer.

Les informations suivantes sont chargées dans la PG :

- S7-300 : configuration du profilé support de base et profilés support d'extension éventuellement présents.
- S7-400 : configuration du châssis de base avec une CPU et des modules de signaux sans châssis d'extension.
- Les données de configuration de la périphérie décentralisée ne peuvent pas être chargées dans la PG.

Il s'agit des informations qui sont chargées lorsque le système cible ne possède pas encore d'informations de configuration, par exemple en cas d'effacement général des systèmes. Sinon la fonction de "Chargement dans la PG" fournit de bien meilleurs résultats.

Dans le cas de systèmes S7-300 sans périphérie décentralisée, il vous suffit alors de spécifier précisément les modules (numéro de référence) et de les paramétrer.

---

#### Nota

Lors du chargement dans la PG (en l'absence d'une configuration hors ligne) STEP 7 n'est pas en mesure de fournir tous les numéros de référence des composants dans leur intégralité. Vous pouvez compléter les numéros de référence "incomplets" lors de la configuration du matériel en choisissant la commande **Outils > Spécifier le module**. Vous pouvez ainsi paramétrer des modules inconnus pour STEP 7 (c'est-à-dire qui ne figurent pas dans la fenêtre "Catalogue du matériel"), sans que les règles de paramétrage ne soient toutefois prises en compte !

---

## Restrictions lors du chargement depuis le système cible

Les restrictions suivantes s'appliquent aux données chargées depuis le système cible dans la PG :

- Les blocs ne contiennent pas de mnémoniques pour les paramètres, variables et repères.
- Les blocs ne contiennent pas de commentaires.
- L'ensemble du programme est chargé dans la PG avec toutes les données système. Seule la partie des données système relative à la "configuration matérielle" pourra cependant être éditée, comme à l'accoutumée.
- Les données relatives à la "communication par données globales (GD)" et à la "configuration de messages sur mnémoniques" ne pourront pas être éditées.
- Les commandes de forçage permanent ne sont pas chargées dans la PG. Elles doivent être enregistrées séparément sous forme de table de variables via l'affichage de la tâche de forçage permanent.
- Les commentaires dans les boîtes de dialogue des modules ne sont pas chargés.
- Les noms des modules ne s'affichent que si vous avez sélectionné cette option lors de la configuration (HW Config : Outils > Paramètres, Enregistrer les noms d'objet dans le système cible).

### 19.3.1 Chargement d'une station dans la PG

La commande **Système cible > Charger station dans la PG** vous permet de charger dans la PG la configuration actuelle ainsi que tous les blocs de l'automate programmable à sélectionner.

STEP 7 crée à cet effet une nouvelle station dans le projet en cours, sous laquelle la configuration est enregistrée. Vous pouvez renommer le nom présélectionné de la station insérée (par exemple "Station SIMATIC 300 (1)"). La station insérée s'affiche aussi bien dans la vue "en ligne" que dans la vue "hors ligne".

Cette commande peut être sélectionnée lorsqu'un projet est ouvert. La sélection d'un objet dans la fenêtre du projet ou la vue (en ligne ou hors ligne) de sont pas significatives pour cette commande.

Cette fonction vous permet de vous faciliter la configuration.

- Pour les systèmes cible S7-300, la configuration est chargée avec les profilés support d'extension sans périphérie décentralisée (DP).
- Pour les systèmes cible S7-400, la configuration du châssis de base est chargée sans châssis d'extension ni périphérie décentralisée.

Dans le cas de systèmes S7-300 sans périphérie décentralisée, il vous suffit alors de spécifier précisément les modules (numéro de référence) et de les paramétrer.

## Restrictions lors du chargement d'une station dans la PG

Les restrictions suivantes s'appliquent aux données chargées dans la PG :

- Les blocs ne contiennent pas de mnémoniques pour les paramètres formels, les variables temporaires et les repères.
- Les blocs ne contiennent pas de commentaires.
- L'ensemble du programme est chargé dans la PG avec toutes les données système ("AG-Abzug"). Toutes les données ne peuvent cependant pas être éditées.
- Les données relatives à la "communication par données globales (GD)", à la "configuration de messages sur mnémoniques" et à la "configuration de réseaux" ne pourront pas être éditées.
- Les commandes de forçage permanent ne peuvent pas être chargées dans la PG, puis à nouveau dans le système cible.

### 19.3.2 Chargement de blocs depuis la CPU S7

SIMATIC Manager vous permet de charger des blocs S7 de la CPU sur le disque dur de l'outil de développement. Vous utilisez par exemple cette possibilité pour :

- effectuer une sauvegarde du programme utilisateur actuel, chargé dans la CPU. En cas de maintenance consécutive à un éventuel effacement général de la CPU, le personnel compétent serait en mesure de charger cette copie de sauvegarde.
- charger le programme utilisateur de la CPU dans l'outil de développement pour l'y éditer afin par exemple d'y rechercher des erreurs lors de la maintenance. Vous ne disposez alors ni des mnémoniques, ni des commentaires documentant le programme. Cette procédure n'est donc vraiment destinée qu'à la maintenance.

### 19.3.3 Edition de blocs chargés dans votre PG/PC

La possibilité d'éditer des blocs dans la PG vous permet :

- de corriger directement un bloc dans la CPU en phase de test et de documenter le résultat ;
- de charger le contenu actuel de blocs de la mémoire vive de chargement de la CPU dans votre console de programmation.

---

#### **Nota**

##### *Conflits d'horodatage lors de l'édition en ligne et hors ligne*

Les procédures suivantes entraînent des conflits d'horodatage et doivent de ce fait être évitées.

Des conflits d'horodatage se produisent à l'ouverture en ligne d'un bloc lorsque

- des modifications effectuées en ligne n'ont pas été enregistrées dans le programme utilisateur S7 hors ligne
- des modifications effectuées hors ligne n'ont pas été chargées dans la CPU.

Des conflits d'horodatage se produisent à l'ouverture hors ligne d'un bloc lorsque

- un bloc en ligne présentant un conflit d'horodatage a été copié dans le programme utilisateur S7 hors ligne, puis est ouvert hors ligne.
- 

### **Cas possibles**

Il faut distinguer deux cas pour le chargement de blocs de la CPU dans la console de programmation.

1. le programme utilisateur auquel les blocs appartiennent se trouve dans la console de programmation.
2. le programme utilisateur auquel les blocs appartiennent ne se trouve pas dans la console de programmation.

Cela signifie que des parties de programme qui ne peuvent pas être chargées dans la CPU ne sont pas disponibles. Il s'agit :

- de la table des mnémoniques et commentaires pour les opérandes,
- des commentaires de réseaux d'un programme LOG ou CONT,
- des commentaires de lignes d'un programme LIST,
- des types de données utilisateur.

### 19.3.3.1 Edition de blocs chargés lorsque le programme utilisateur se trouve dans votre PG/PC

Pour éditer des blocs de la CPU, procédez de la manière suivante :

1. Dans SIMATIC Manager, ouvrez la fenêtre du projet en ligne.
2. Dans la fenêtre en ligne, sélectionnez un dossier Blocs. La liste des blocs chargés s'affiche.
3. Vous pouvez à présent sélectionner des blocs, puis les ouvrir et les éditer.
4. La commande **Fichier > Enregistrer** vous permet de sauvegarder les modifications hors ligne dans la PG.
5. Choisissez la commande **Système cible > Charger** pour charger les blocs modifiés dans le système cible.

### 19.3.3.2 Edition de blocs chargés lorsque le programme utilisateur ne se trouve pas dans votre PG/PC

Pour éditer des blocs de la CPU, procédez de la manière suivante :

1. Dans SIMATIC Manager, cliquez sur le bouton "Partenaires accessibles" ou choisissez la commande **Système cible > Afficher les partenaires accessibles**.
2. Sélectionnez le partenaire (objet "MPI=...") dans la liste affichée et ouvrez le dossier Blocs pour afficher les blocs.
3. Vous pouvez alors ouvrir des blocs et, si besoin est, les éditer, les surveiller ou les copier.
4. Choisissez la commande **Fichier > Enregistrer sous...** en indiquant, dans la boîte de dialogue correspondante, le chemin d'accès désignant l'emplacement de mémoire désiré sur la PG.
5. Choisissez la commande **Système cible > Charger** pour charger les blocs modifiés dans le système cible.

## 19.4 Effacement sur le système cible

### 19.4.1 Effacement de la mémoire de chargement/travail et effacement général de la CPU

Avant de charger votre programme utilisateur dans le système cible S7, nous vous recommandons d'effectuer un effacement général de la CPU, afin de vous assurer qu'elle ne contient plus "d'anciens" blocs.

#### Condition préalable à l'effacement général

Pour qu'un effacement général soit possible, la CPU doit se trouver à l'état de fonctionnement "arrêt" (le commutateur de mode de fonctionnement doit être positionné sur arrêt (STOP) ou encore sur marche (RUN-P), auquel cas vous devez mettre la CPU à l'arrêt en choisissant la commande **Système cible > Diagnostic/Paramètres > Etat de fonctionnement**).

#### Effacement général de la CPU S7

Voici ce qui se déroule lors de l'effacement général d'une CPU S7 :

- La CPU est remise à 0.
- Toutes les données utilisateur sont effacées (les blocs et les blocs de données système (SDB) à l'exclusion des paramètres MPI).
- La CPU suspend toutes les liaisons en cours.
- S'il existe des données dans une EPROM (carte mémoire ou EPROM intégrée), la CPU copie, après l'effacement général, le contenu de l'EPROM dans la zone RAM de la mémoire.

Le contenu de la mémoire tampon de diagnostic et les paramètres de l'interface MPI sont conservés.

#### Effacement général de CPU/FM M7

Voici ce qui se déroule lors de l'effacement général de CPU/FM M7 :

- L'état initial est restauré.
- Les SDB à l'exclusion des paramètres MPI sont effacés.
- La CPU/le FM suspend toutes les liaisons en cours. Les programmes utilisateur sont conservés et leur exécution reprend aussitôt que vous commutez la CPU de STOP en RUN.

La fonction d'effacement général vous permet de restaurer l'état initial de la CPU ou du FM M7 après une erreur majeure. Vous devez pour cela effacer les blocs de données système (SDB) dans la mémoire de travail et charger ceux qui se trouvent dans la mémoire permanente. Dans certains cas, il faut effectuer en plus un démarrage à chaud du système d'exploitation. Pour cela, vous devez effectuer un effacement général du M7 en actionnant le commutateur de mode de fonctionnement (position MRES). Une remise à zéro via le commutateur de mode de fonctionnement des CPU/FM SIMATIC M7 n'est possible que sous le système d'exploitation RMOS32.



## 19.4.2 Effacement de blocs S7 sur le système cible

Durant la phase de test du programme de la CPU, il peut s'avérer nécessaire d'effacer certains blocs dans la CPU. Les blocs sont sauvegardés dans la mémoire utilisateur de la CPU soit dans l'EPROM, soit dans la RAM (en fonction de la CPU et de la procédure de chargement).

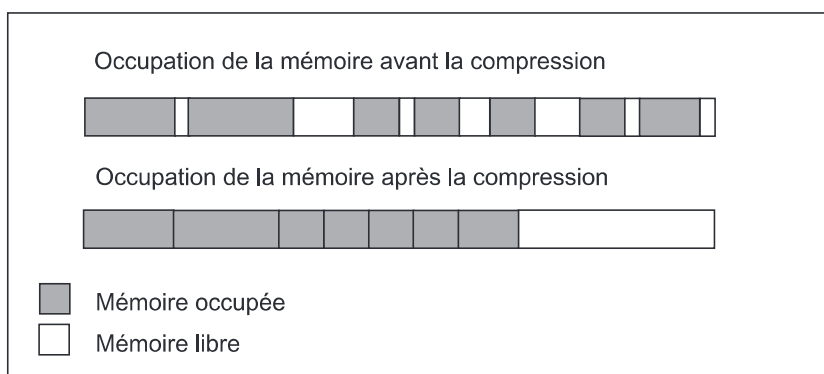
- Vous pouvez effacer directement les blocs chargés dans la mémoire vive. L'espace mémoire qui était occupé dans les mémoires de chargement et de travail est alors libéré.
- Les blocs enregistrés dans l'EPROM intégrée sont toujours copiés dans la zone de mémoire vive après effacement de la CPU. Vous pouvez effacer directement ces copies dans la mémoire vive. Les blocs effacés seront alors déclarés non valables dans l'EPROM jusqu'au prochain effacement général ou jusqu'à la prochaine panne secteur, lorsque la mémoire vive n'est pas sauvegardée. En cas d'effacement général ou de panne secteur lorsque la mémoire vive n'est pas sauvegardée, les blocs "effacés" sont à nouveau copiés de l'EPROM dans la mémoire vive, où ils sont alors à nouveau actifs. Les blocs enregistrés dans l'EPROM intégrée (par exemple de la CPU 312) sont effacés par écrasement par le nouveau contenu de la mémoire vive.
- Les cartes mémoire EPROM doivent être effacées dans l'outil de développement.

## 19.5 Compression de la mémoire utilisateur (RAM)

### 19.5.1 Intervalles dans la mémoire utilisateur (RAM)

Les intervalles résultant des effacements et chargements successifs de blocs réduisent l'espace mémoire utilisable. Il importe de comprimer la mémoire utilisateur en réorganisant les blocs existants afin de créer une zone de mémoire libre d'un seul tenant.

La figure ci-après montre schématiquement comment les blocs de mémoire occupés sont déplacés par la fonction "Compression de la mémoire".



#### **La compression à l'état de fonctionnement "arrêt" est recommandée.**

Seule la compression à l'état de fonctionnement "Arrêt" (STOP) permet d'éliminer tous les intervalles en mémoire. Si vous effectuez la compression à l'état de fonctionnement RUN-P (position du commutateur de mode de fonctionnement), les blocs en cours d'édition ne seront pas déplacés, puisqu'ils sont ouverts. La fonction de compression ne peut pas être exécutée à l'état de fonctionnement RUN (position du commutateur de mode de fonctionnement) (protection en écriture !).

## 19.5.2 Compression du contenu de la mémoire d'une CPU S7

### Possibilités de compression

Vous avez deux possibilités pour comprimer la mémoire utilisateur.

- Si un manque de mémoire apparaît dans le système cible lors du chargement, une boîte de dialogue vous signalant l'incident s'affiche. Vous pouvez comprimer la mémoire en cliquant sur le bouton correspondant dans cette boîte de dialogue.
- En guise de mesure préventive, vous pouvez afficher l'occupation de la mémoire (commande **Système cible > Diagnostic/Paramètres > Etat du module...**/page d'onglet "Mémoire"), et, éventuellement, déclencher la compression.

### Procédure

1. Sélectionnez le programme S7 dans la vue en ligne ou dans la fenêtre "Partenaires accessibles".
2. Choisissez la commande **Système cible > Diagnostic/Paramètres > Etat du module**.
3. Dans la boîte de dialogue suivante, choisissez l'onglet "Mémoire" Vous y trouvez le bouton de compression, si la CPU permet la mise en œuvre de cette fonction.



## 20 Test avec des tables de variables

### 20.1 Introduction au test avec des tables de variables

Les tables de variables permettent d'enregistrer des environnements de test différents et donc de reproduire sans peine les tests et les observations au cours d'une mise en service ou à des fins de maintenance. Il n'y a pas de limite au nombre de tables de variables enregistrées.

Pour effectuer le test avec des tables de variables, vous disposez des fonctions suivantes :

- **Visualisation de variables**  
Cette fonction vous permet d'afficher sur la PG ou le PC les valeurs en cours de certaines variables d'un programme utilisateur ou d'une CPU.
- **Forçage de variables**  
Cette fonction vous permet d'attribuer des valeurs fixes à certaines variables d'un programme utilisateur ou d'une CPU. Le test avec l'état du programme permet également le forçage unique et immédiat.
- **Débloquer sorties périphériques et Activer valeurs de forçage**  
Ces deux fonctions vous permettent d'attribuer des valeurs fixes à certaines sorties de périphérie d'une CPU à l'état d'arrêt.
- **Forçage permanent de variables**  
Cette fonction vous permet d'attribuer à certaines variables d'un programme utilisateur ou d'une CPU, une valeur fixe que le programme utilisateur ne peut pas écraser.

Vous pouvez forcer et visualiser les variables suivantes :

- entrées, sorties, mémentos, temporisations et compteurs
- contenus de blocs de données,
- périphérie.

Vous indiquez les variables à forcer ou à visualiser en établissant une table de variables.

Vous déterminez à quel point et à quelle fréquence visualiser ou forcer les variables en définissant un point de déclenchement et une condition de déclenchement.

## 20.2 Marche à suivre pour la visualisation et le forçage avec des tables de variables

Pour exécuter les fonctions de **visualisation** et de **forçage**, procédez de la manière suivante :

1. Créez une nouvelle table de variables ou ouvrez-en une existante.
2. Editez ou vérifiez la table de variables.
3. Etablissez une liaison en ligne entre la table de variables en cours et la CPU de votre choix à l'aide de la commande **Système cible > Etablir la liaison à....**
4. Choisissez, avec la commande **Variable > Déclenchement**, un point de déclenchement approprié et définissez la condition de déclenchement.
5. Les commandes **Variable > Visualiser** et **Variable > Forcer** activent et désactivent les fonctions correspondantes.
6. Sauvegardez la table de variables achevée en choisissant la commande **Table > Enregistrer** ou **Table > Enregistrer** sous, afin de pouvoir la rappeler à tout moment.

## 20.3 Edition et enregistrement de tables de variables

### 20.3.1 Création et ouverture d'une table de variables

Pour visualiser ou forcer des variables, il faut d'abord créer une table de variables (VAT) et y entrer les variables concernées. Pour créer une table de variables, vous disposez des possibilités suivantes :

#### Dans SIMATIC Manager :

- Sélectionnez le dossier Blocs et choisissez la commande **Insertion > Bloc S7 > Table des variables**. Vous pouvez donner un nom à la table dans la boîte de dialogue qui apparaît alors (champ de saisie "Nom symbolique"). Ce nom s'affichera dans la fenêtre de projet. Pour ouvrir la table de variables, effectuez un double clic sur l'objet.
- Sélectionnez une liaison dans la liste des partenaires accessibles ou un programme S7/M7 dans la vue en ligne. Vous pouvez alors créer une table de variables sans nom à l'aide de la commande **Système cible > Visualiser/forcer des variables**.

#### Dans la fenêtre "Visualisation et forçage de variables" :

- Vous pouvez créer, avec la commande **Table > Nouvelle**, une nouvelle table qui n'est encore affectée à aucun programme S7 ou M7. Vous ouvrez les tables existantes avec la commande **Table > Ouvrir**.
- Vous pouvez vous servir des boutons de la barre d'outils pour créer ou ouvrir une table de variables.

Une fois créée, vous pouvez sauvegarder, imprimer et réutiliser la table des variables pour la visualisation et le forçage.

### 20.3.2 Copie ou déplacement de tables de variables

Vous pouvez copier ou déplacer des tables de variables dans le dossier Blocs d'un programme S7/M7.

Ce faisant, tenez compte des points suivants :

- Les mnémoniques existant déjà dans la table des mnémoniques du programme cible y sont mis à jour.
- Lorsque vous déplacez une table de variables, les mnémoniques correspondants figurant dans la table des mnémoniques du programme source sont eux aussi déplacés et écrits dans la table des mnémoniques du programme cible.
- Lorsque vous effacez des tables de variables du dossier Blocs, les mnémoniques correspondants sont effacés eux aussi de la table des mnémoniques du programme S7/M7.
- Si le programme cible comporte déjà une table des variables portant le même nom, vous aurez la possibilité de la renommer lors de la copie (par défaut, un numéro sera ajouté au nom existant).

### 20.3.3 Enregistrement d'une table de variables

Vous pouvez réutiliser les tables de variables enregistrées pour visualiser ou forcer les variables lors d'un nouveau test de votre programme.

1. Enregistrez la table de variables en choisissant la commande **Table > Enregistrer**.
2. Si vous venez de créer la table des variables, vous devez maintenant lui attribuer un nom, par exemple "Test\_prog1".

Lorsque vous enregistrez une table de variables, tous les paramètres actuels et le format de la table le sont également, c'est-à-dire aussi les paramètres définis avec la commande Déclenchement.

## 20.4 Saisie de variables dans des tables de variables

### 20.4.1 Insertion d'opérandes ou de mnémoniques dans une table de variables

Déterminez les variables dont vous souhaitez visualiser l'état ou que vous désirez forcer, et entrez-les dans la table. Pour ce faire, procédez de "l'extérieur" vers "l'intérieur", c'est-à-dire choisissez d'abord les entrées, puis les variables influencées par les entrées ou influençant les sorties et, pour finir, les sorties.

Par exemple, si vous désirez visualiser l'état du bit d'entrée 1.0, du mot de memento 5 et de l'octet de sortie 0, entrez les valeurs suivantes dans la colonne de l'opérande:

#### Exemple

E 1.0  
MW 5  
AB 0



## Exemple de table de variables complétée

La figure suivante montre une table des variables avec les colonnes suivantes : Opérande, Mnémonique, Format d'affichage, Valeur d'état et Valeur de forçage.

	Opérande	Mnémonique	Format affichage	Valeur état	Val forçage
1	//OB1 Réseau 1				
2	E 0.1	"Bouton-poussoir 1"	BOOL	true	
3	E 0.2	"Bouton-poussoir 2"	BOOL	true	
4	A 4.0	"Feu vert"	BOOL	false	
5	//OB1 Réseau 3				
6	E 0.5	"Automatique activé"	BOOL	true	
7	E 0.6	"Manuel activé"	BOOL	true	
8	A 4.2	"Fonction automatique"	BOOL	true	true
9	//OB1 Appel FB1 pour démarrage moteur essence				
10	E 1.0	"ME_démarrage"	BOOL	false	
11	E 1.1	"ME_arrêt"	BOOL	false	
12	E 1.2	"ME_Défaillance"	BOOL	false	
13	A 5.1	"ME_Consigne_atteinte"	BOOL	false	
14	A 5.0	"ME_marche"	BOOL		true
15	//OB1 Appel FB1 pour démarrage moteur diesel				
16	E 1.4	"MD_démarrage"	BOOL	false	
17	E 1.5	"MD_arrêt"	BOOL		

## Remarques sur l'insertion de mnémoniques

- Vous indiquez la variable à forcer par son opérande (adresse absolue) ou son mnémonique. Vous pouvez saisir des opérandes et des mnémoniques aussi bien dans la colonne "Opérande" que dans la colonne "Mnémonique". L'entrée est automatiquement reportée dans la colonne qui convient.  
Si un tel mnémonique est défini dans la table des mnémoniques, l'entrée correspondante de la colonne de mnémonique ou d'opérande est automatiquement complétée.
- Vous ne pouvez inscrire que des mnémoniques déjà définis dans la table des mnémoniques.
- Vous devez saisir les mnémoniques exactement comme ils ont été définis dans la table des mnémoniques.
- Ecrivez entre guillemets les mnémoniques contenant des caractères spéciaux (par exemple, "Moteur.Arrêt", " Moteur+Arrêt", " Moteur- Arrêt").
- Utilisez la commande **Outils > Table des mnémoniques** pour définir de nouveaux mnémoniques dans la table des mnémoniques. Vous pouvez également copier des mnémoniques de la table des mnémoniques, puis les insérer dans une table de variables.

## Vérification de la syntaxe

Lorsque vous inscrivez des variables dans la table, une vérification de la syntaxe est exécutée avant l'abandon de la ligne. Les entrées erronées sont marquées en rouge.

Lorsque vous placez le curseur sur une ligne signalée en rouge, une info-bulle précise la cause de l'erreur. La touche F1 vous donne alors des indications pour remédier à cette erreur.

---

### Nota

Si vous préférez éditer la table des variables en utilisant le clavier (sans souris), nous vous recommandons d'activer l'option "Infos-bulles pour une utilisation du clavier".

Le cas échéant, modifiez ce paramétrage dans la table des variables en choisissant la commande de menu **Outils > Paramètres**, onglet "Général".

---

## Taille maximale

Une table de variables peut comprendre au maximum 255 caractères par ligne. Il n'est pas possible d'obtenir une seconde ligne par retour chariot. La longueur d'une table de variables est fixée à 1024 lignes. La taille maximale de la table est alors atteinte.

### 20.4.2 Insertion d'une plage d'opérandes continue dans une table de variables

1. Ouvrez une table de variables.
2. Positionnez le curseur dans la ligne à la suite de laquelle vous souhaitez insérer la plage d'opérandes.
3. Choisissez la commande **Insertion > Plage** : la boîte de dialogue "Insérer une plage d'opérandes" s'affiche.
4. Dans le champ "Opérande initial", tapez une adresse d'opérande comme adresse de début de plage.
5. Dans le champ "Nombre", tapez le nombre de lignes à insérer.
6. Choisissez le format d'affichage souhaité dans la liste affichée.
7. Cliquez sur OK.

La plage d'opérandes est alors insérée dans la table des variables.

### 20.4.3 Insertion de valeurs de forçage

#### Mise en commentaire

Si vous souhaitez mettre en commentaire la "valeur de forçage" d'une variable, choisissez la commande **Variable > Mise en commentaire**. L'indicatif de commentaire "//", placé devant la valeur de forçage de la variable indique sa mise en commentaire. Au lieu d'appeler la commande, vous pouvez également saisir l'indicatif de commentaire "//" devant la "valeur de forçage". Pour annuler la mise en commentaire de la "valeur de forçage", choisissez à nouveau la commande **Variable > Mise en commentaire** ou supprimez l'indicatif de commentaire.

## 20.4.4 Limites supérieures pour la saisie de temporisations

Pour la saisie des temporisations, veuillez respecter les limites supérieures suivantes :

W#16#3999 (valeur maximale en format DCB)

### Exemples :

Opérande		Format d'affichage	Frappe	Valeur de forçage affichée	Signification
T	1	DUREE SIMATIC	137	S5TIME#130MS	Conversion en millisecondes
MW	4	DUREE SIMATIC	137	S5TIME#890MS	Représentation en format DCB possible
MW	4	HEXA	137	W#16#0089	Représentation en format DCB possible
MW	6	HEXA	157	W#16#009D	Représentation en format DCB impossible ; aussi ne pouvez-vous pas sélectionner le format d'affichage DUREE SIMATIC.

### Nota

- Vous pouvez saisir les temporisations avec une précision d'une milliseconde, mais la valeur entrée est corrigée en fonction d'une base de temps interne. La grille de temps dépend de la valeur entrée (137 donne 130ms, les 7ms ayant été arrondies).
- Les valeurs de forçage d'opérandes de type de données WORD, par exemple EW1, sont converties en format DCB. Mais chaque profil binaire n'est pas un nombre DCB correct ! Quand, pour un opérande de type WORD, la valeur saisie ne peut être représentée comme DUREE SIMATIC, elle est représentée automatiquement dans le format par défaut, ici : HEXA ; voir Choisir format d'affichage, Format par défaut (menu Affichage) afin d'être affichée.

### Format DCB pour les variables en format DUREE SIMATIC

Les valeurs de variables en format DUREE SIMATIC sont saisies en format DCB.

Les 16 bits ont la signification suivante.

| 0 0 x x | c c c c | d d d d | u u u u |

Bits 15 et 14 sont toujours à 0.

Bits 13 et 12 (indiqués par xx) déterminent le multiplicateur pour les bits 0 à 11 :

00 => multiplicateur 10 millisecondes    01 => multiplicateur 100 millisecondes

10 => multiplicateur 1 seconde

11 => multiplicateur 10 secondes

Bits 11 à 8    centaines (cccc)

Bits 7 à 4    dizaines (dddd)

Bits 3 à 0    unités (uuuu)

## 20.4.5 Limites supérieures pour la saisie de compteurs

Pour la saisie des compteurs, veuillez respecter les limites supérieures suivantes :

- valeur limite pour compteur : C#999
- W#16#0999 (valeur maximale en format DCB)

### Exemples :

Opérande		Format d'affichage	Frappe	Valeur de forçage affichée	Signification
Z	1	COMPTEUR	137	C#137	Conversion
MW	4	COMPTEUR	137	C#89	Représentation en format DCB possible
MW	4	HEXA	137	W#16#0089	Représentation en format DCB possible
MW	6	HEXA	157	W#16#009D	Représentation en format DCB impossible ; aussi ne pouvez-vous pas sélectionner le format d'affichage COMPTEUR.

### Nota

- Si vous entrez un nombre décimal pour un compteur sans caractériser la valeur par C#, elle sera convertie automatiquement en format DCB (137 donne C#137).
- Les valeurs de forçage d'opérandes de type de données WORD, par exemple EW1, sont converties en format DCB. Mais chaque profil binaire n'est pas un nombre DCB correct ! Quand, pour un opérande de type WORD, la valeur saisie ne peut être représentée comme COMPTEUR, elle est représentée automatiquement dans le format par défaut, ici : HEXA ; voir Choisir format d'affichage, Format par défaut (menu Affichage) afin d'être affichée.

## 20.4.6 Insertion de lignes de commentaire

Les lignes de commentaire sont introduites par le signe //.

Si vous souhaitez mettre en commentaire une ou plusieurs lignes de la tables des variables choisissez la commande **Edition > Désactiver ligne** ou cliquez sur le bouton correspondant



dans la barre d'outils.

## 20.4.7 Exemples

### 20.4.7.1 Exemple de saisie d'opérandes dans une table de variables

Opérande admissible	Type de données	Exemple (abréviations allemandes)
Entrée   Sortie   Mémento	BOOL	E 1.0   A 1.7   M 10.1
Entrée   Sortie   Mémento	BYTE	EB 1   AB 10   MB 100
Entrée   Sortie   Mémento	WORD	EW 1   AW 10   MW 100
Entrée   Sortie   Mémento	DWORD	ED 1   AD 10   MD 100
Périphérie (Entrée   Sortie )	BYTE	PEB 0   PAB 1
Périphérie (Entrée   Sortie )	WORD	PEW 0   PAW 1
Périphérie (Entrée   Sortie )	DWORD	PED 0   PAD 1
Temporisations	TIMER	T 1
Compteurs	COUNTER	Z 1
Bloc de données	BOOL	DB1.DBX 1.0
Bloc de données	BYTE	DB1.DBB 1
Bloc de données	WORD	DB1.DBW 1
Bloc de données	DWORD	DB1.DBD 1

#### Nota

La saisie de "DB0. .." n'est pas autorisée en raison de son utilisation interne.

### Dans la fenêtre des valeurs de forçage

Avec les modules S7-300, vous ne pouvez forcer que les entrées, sorties et périphéries (sorties).

Avec les modules S7-400, vous ne pouvez forcer que les entrées, sorties, mémentos et périphéries (entrées/sorties).

### 20.4.7.2 Exemple de saisie d'une plage d'opérandes continue

Ouvrez une table de variables et choisissez la commande **Insertion > Plage** pour afficher la boîte de dialogue "Insérer une plage d'opérandes".

Durant la saisie dans la boîte de dialogue, les lignes suivantes seront ajoutées à la table de variables pour des mementos (M) :

- Opérande initial : M 3.0
- Nombre : 10
- Format d'affichage : BIN

Opérande	Format d'affichage
M 3.0	BIN
M 3.1	BIN
M 3.2	BIN
M 3.3	BIN
M 3.4	BIN
M 3.5	BIN
M 3.6	BIN
M 3.7	BIN
M 4.0	BIN
M 4.1	BIN

Notez comme dans le présent exemple, la désignation change après la huitième entrée dans la colonne "Opérande".

### 20.4.7.3 Exemples de saisie de valeurs de forçage/forçage permanent

#### Opérandes de type bit

Opérandes de type bit possibles	Valeurs de forçage/forçage permanent autorisées
E1.0	true
M1.7	false
A10.7	0
DB1.DBX1.1	1
E1.1	2#0
M1.6	2#1

#### Opérandes de type octet

Opérandes de type octet possibles	Valeurs de forçage/forçage permanent autorisées
EB 1	2#00110011
MB 12	b#16#1F
MB 14	1F
AB 10	'a'
DB1.DBB 1	10
PAB 2	12

#### Opérandes de type mot

Opérandes de type mot possibles	Valeurs de forçage/forçage permanent autorisées
EW 1	2#0011001100110011
MW 12	w#16#ABCD
MW 14	ABCD
AW 10	b#(12,34)
DB1.DBW 1	'ab'
PAW 2	12345
MW 3	12345
MW 5	S5t#12s340ms
MW 7	0.3s ou 0,3s
MW 9	C#123
MW 11	d#1990-12-31

## Opérandes de type double mot

Opérandes de type double mot possibles	Valeurs de forçage/forçage permanent autorisées
ED 1	2#0011001100110011001100110011
MD 0	1.23e4
MD 4	1.2
AD 10	dw#16#abcdef10
AD 12	ABCDEF10
DB1.DBD 1	b#(12,34,56,78)
PAD 2	'abcd'
MD 8	L# -12
MD 12	L#12
MD 16	123456789
MD 20	123456789
MD 24	T#12s345ms
MD 28	Tod#1:2:34.567
MD 32	p#e0.0

## Temporisations

Opérandes de type temporisation possibles	Valeurs de forçage/ forçage permanent autorisées	Signification
T 1	0	Valeur de temps en millisecondes (ms)
T 12	20	Valeur de temps en millisecondes (ms)
T 14	12345	Valeur de temps en millisecondes (ms)
T 16	s5t#12s340ms	Valeur de temps égale à 12s 340ms
T 18	1.3	Valeur de temps égale à 1s 300 ms
T 20	1.3s	Valeur de temps égale à 1s 300 ms

Le forçage d'une temporisation n'influe que sur la valeur, pas sur l'état. Ainsi, il est possible de forcer la temporisation T1 à la valeur 0, mais le résultat logique pour U T1 n'est pas modifié.

Les chaînes de caractères "s5t" et "s5time" peuvent être écrites aussi bien en minuscules qu'en majuscules.

## Compteurs

Opérandes de type compteur possibles	Valeurs de forçage/forçage permanent autorisées
Z 1	0
Z 14	20
Z 16	c#123

Le forçage d'un compteur n'influe que sur la valeur, pas sur l'état. Ainsi, il est possible de forcer le compteur Z1 à la valeur 0, mais le résultat logique pour U Z1 n'est pas modifié.



## 20.5 Etablissement d'une liaison à la CPU

Pour visualiser ou forcer les variables que vous avez définies dans la table de variables (VAT) en vigueur, vous devez établir une liaison à la CPU correspondante. Vous pouvez établir une liaison à une CPU différente pour chaque table de variables.

### Affichage de la liaison en ligne

En présence d'une liaison en ligne, la barre de titre de la fenêtre de la table des variables affiche "En ligne". En fonction de la CPU, la barre d'état affiche les états de fonctionnement "Marche", "Arrêt", "Déconnectée" ou "Connectée".

### Etablissement d'une liaison en ligne à la CPU

S'il n'existe pas de liaison en ligne avec la CPU de votre choix, vous en définissez une à l'aide de la commande **Système cible > Etablir la liaison à > ...** afin de pouvoir visualiser ou forcer les variables.

### Suspension d'une liaison en ligne à la CPU

La commande **Système cible > Suspendre la liaison** permet d'interrompre la liaison entre table de variables et CPU.

---

#### Nota

Si vous avez créé une table de variables en possédant pas de nom à l'aide de la commande **Table > Nouvelle**, vous pouvez établir une liaison à la dernière CPU configurée si elle est définie.

---

## 20.6 Visualisation de variables

### 20.6.1 Introduction à la visualisation de variables

Vous disposez des possibilités suivantes pour visualiser des variables :

- Activez la fonction de visualisation avec la commande **Variable > Visualiser**. Les valeurs des variables sélectionnées sont alors affichées dans la table des variables en fonction du point et de la condition de déclenchement définis. Si vous avez choisi la condition de déclenchement "Cyclique", vous pouvez à nouveau désactiver la fonction de visualisation avec la commande **Variable > Visualiser**.
- Actualisez les valeurs des variables sélectionnées de manière unique et immédiatement avec la commande **Variable > Actualiser les valeurs d'état**. Les valeurs actuelles des variables sélectionnées sont alors affichées dans la table des variables.

#### Interruption de la visualisation par la touche ECHAP

La fonction "Visualisation" étant en cours d'exécution, une pression de la touche ECHAP y met fin sans demande de confirmation.

### 20.6.2 Définition du déclenchement pour la visualisation de variables

Pour la visualisation, vous pouvez afficher à la PG les valeurs en cours de certaines variables d'un programme utilisateur, en un point déterminé du programme, le point de déclenchement.

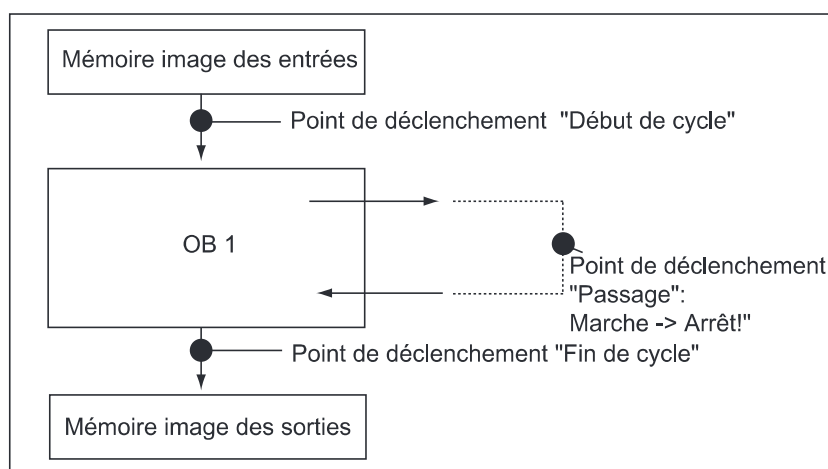
En choisissant un point de déclenchement, vous définissez à quel instant les valeurs d'état des variables vont être affichées.

La commande **Variable > Déclenchement** vous permet de définir un point et une condition de déclenchement.

Déclenchement	Possibilités de paramétrage
Point de déclenchement	Début de cycle Fin de cycle Passage de "Marche" à "Arrêt"
Condition de déclenchement	unique cyclique

## Point de déclenchement

La figure suivante montre les différents points de déclenchement.



Pour afficher la valeur forcée dans la colonne "Valeur d'état", définissez comme point de déclenchement de la visualisation "Début de cycle" et comme point de déclenchement du forçage "Fin de cycle".

## Déclenchement immédiat

Vous pouvez actualiser les valeurs de variables sélectionnées en choisissant la commande **Variable > Actualiser les valeurs d'état**. Cette tâche est exécutée une seule fois et le plus rapidement possible, sans relation avec un endroit précis dans le programme utilisateur. Ces fonctions s'utilisent principalement à l'état d'"Arrêt" (STOP) pour la visualisation et le forçage.

## Condition de déclenchement

Le tableau suivant montre l'effet de la condition de déclenchement sélectionnée sur la visualisation de variables :

	Condition de déclenchement "Unique"	Condition de déclenchement "Cyclique"
Visualiser des variables	Actualisation unique dépend du point de déclenchement	Visualisation avec déclenchement défini Lorsque vous testez un bloc, vous pouvez suivre avec précision la poursuite du traitement.

## 20.7 Forçage de variables

### 20.7.1 Introduction au forçage de variables

Vous disposez des possibilités suivantes pour forcer des variables :

- Activez la fonction de forçage avec la commande **Variable > Forcer**. Le programme utilisateur affecte aux variables sélectionnées les valeurs de forçage figurant dans la table des variables, en fonction du point et de la condition de déclenchement définis. Si vous avez choisi la condition de déclenchement "Cyclique", vous pouvez à nouveau désactiver la fonction de forçage avec la commande **Variable > Forcer**.
- Actualisez les valeurs des variables sélectionnées de manière unique et immédiatement avec la commande **Variable > Actualiser valeurs d'état**.

Des possibilités supplémentaires vous sont offertes avec les fonctions "Forçage permanent" et "Débloquer sorties périphériques".

#### Important lors du forçage :

- Le forçage s'applique seulement aux opérandes qui étaient visibles dans la table des variables au début du forçage.  
Si la zone visible de la table de variables se trouve réduite après le début du forçage, il peut arriver que des opérandes devenus invisibles soient forcés.  
Si la zone visible de la table de variables se trouve agrandie, il peut arriver que des opérandes devenus visibles ne soient pas forcés.
- Il n'est pas possible d'annuler le forçage (par exemple avec la commande **Edition > Annuler**).



#### Danger

Modifier les valeurs des variables alors que l'installation est en marche peut, en cas de défaut de fonctionnement ou d'erreurs dans le programme, entraîner des blessures corporelles graves et des dégâts matériels importants.  
Assurez-vous qu'aucun état dangereux ne peut apparaître avant d'exécuter la fonction "Forçage".

---

#### Interruption du forçage par la touche ECHAP

La fonction "Forçage" étant en cours d'exécution, une pression de la touche ECHAP y met fin sans demande de confirmation.

## 20.7.2 Définition du déclenchement pour le forçage de variables

Vous pouvez affecter de manière unique ou cyclique des valeurs fixes à des variables d'un programme utilisateur en un point défini dans l'exécution du programme (point de déclenchement).

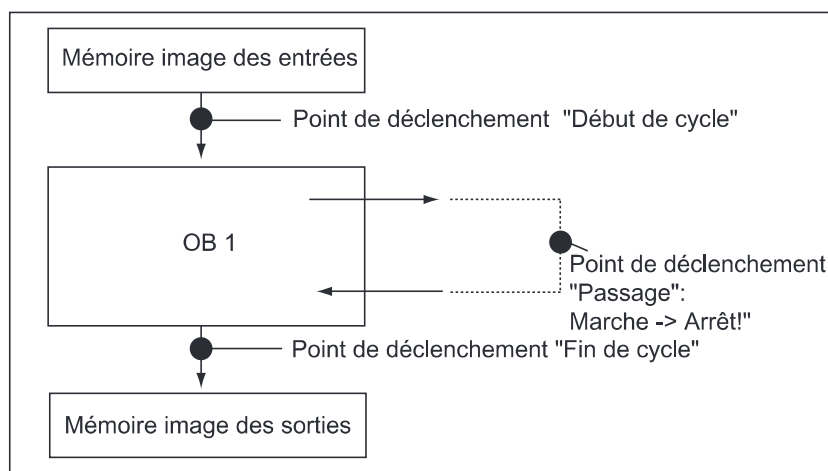
En choisissant un point de déclenchement, vous définissez à quel instant les valeurs de forçage seront affectées aux variables.

La commande **Variable > Déclenchement** vous permet de définir le point et une condition de déclenchement.

Déclenchement	Possibilités de paramétrage
Point de déclenchement	Début de cycle Fin de cycle Passage de "Marché" à "Arrêt"
Condition de déclenchement	unique cyclique

### Point de déclenchement

La figure suivante illustre la position des points de déclenchement.



Le choix du point de déclenchement a les effets suivants :

- Le forçage des entrées n'a de sens que si le point de déclenchement "Début de cycle" a été choisi, les entrées étant sinon écrasées juste après le forçage en raison de l'actualisation de la mémoire image des entrées au début de l'exécution de l'OB cyclique (OB1).
- Les forçage des sorties n'a de sens que si le point de déclenchement "Fin de cycle" a été choisi, la mémoire image des sorties étant sinon écrasée par le programme utilisateur .

Pour afficher la valeur forcée dans la colonne "Valeur d'état", définissez comme point de déclenchement de la visualisation "Début de cycle" et comme point de déclenchement du forçage "Fin de cycle".

Lors du forçage de variables, tenez compte des indications suivantes en ce qui concerne les points de déclenchement :

- Si vous avez choisi la condition de déclenchement "Unique", vous obtenez un message lorsque les variables sélectionnées ne peuvent pas être forcées.
- Si la condition de déclenchement est "Cyclique", vous n'obtenez pas de message .

### Déclenchement immédiat

Vous pouvez forcer les valeurs de variables sélectionnées en choisissant la commande **Variable > Activer valeurs de forçage**. Cette tâche est exécutée une seule fois et le plus rapidement possible, sans relation avec un endroit précis dans le programme utilisateur. Cette fonction s'utilise principalement à l'état d'"Arrêt" (STOP) pour le forçage.

### Condition de déclenchement

Le tableau suivant montre l'effet de la condition de déclenchement sélectionnée sur le forçage de variables :

	Condition de déclenchement "Unique"	Condition de déclenchement "Cyclique"
Forcer des variables	<i>Activation unique (forçage de variables)</i> Vous pouvez affecter des valeurs à des variables une fois en fonction du point de déclenchement.	<i>Forçage avec déclenchement défini</i> Le forçage de variables à des valeurs fixes permet de simuler des situations précises pour votre programme utilisateur et de tester ainsi les fonctions programmées.

## 20.8 Forçage permanent de variables

### 20.8.1 Mesures de sécurité pour le forçage permanent de variables



#### Vous devez éviter des lésions corporelles ou un dommage matériel !

Notez bien qu'une erreur de manipulation de la fonction "Forçage permanent" risque

- de mettre en danger la vie ou la santé des opérateurs,
- d'endommager la machine ou l'ensemble de l'installation.



#### Avertissement

- Avant de lancer la fonction de forçage permanent, assurez-vous que personne d'autre ne l'exécute simultanément sur la même CPU.
- Seule la commande **Variable > Annuler forçage permanent** peut effacer une tâche de forçage permanent ou y mettre fin. Les valeurs de forçage permanent ne sont pas effacées par la fermeture de la fenêtre qui les affiche ou par celle de l'application "Visualisation et forçage de variables".
- La commande **Edition > Annuler** ne permet pas d'annuler le forçage permanent.
- Renseignez-vous sur les différences entre forçage de variables et forçage permanent de variables.
- Aucune des commandes du menu "Variable" concernant le forçage permanent n'est disponible quand une CPU n'accepte pas la fonction de forçage permanent.

Tous les modules de sorties faisant l'objet d'un forçage permanent indiquent leur valeur de forçage permanent si vous annulez le blocage des sorties avec la commande **Variable > Débloquer sorties périphériques**.

---

## 20.8.2 Introduction au forçage permanent de variables

Vous pouvez affecter des valeurs fixes à des variables individuelles d'un programme utilisateur afin qu'elles ne puissent ni être modifiées ni être écrasées, même par le programme utilisateur exécuté dans la CPU. Il faut évidemment que la CPU possède cette fonction (comme, par exemple, la CPU S7-400). Le forçage permanent de variables à des valeurs fixes permet de simuler des situations précises pour votre programme utilisateur et de tester ainsi les fonctions programmées.

### Fenêtre des valeurs de forçage permanent

Les commandes de forçage permanent ne sont disponibles qu'une fois la fenêtre des valeurs de forçage permanent ouverte.

Pour afficher cette fenêtre, choisissez la commande **Variable > Afficher valeurs de forçage permanent**.

Vous n'êtes autorisé à ouvrir qu'une seule fenêtre de valeurs de forçage permanent par CPU. Les variables y sont affichées avec leurs valeurs pour la tâche active de forçage permanent.

### Exemple de fenêtre des valeurs de forçage permanent

	Opérand	Mnémon	Format d'affich	Valeur de for
1	<b>EB 0</b>		HEX	<b>B#16#10</b>
2	<b>A 0.1</b>		BOOL	<b>true</b>
3	<b>A 1.2</b>		BOOL	<b>true</b>
4				

La **barre du titre** mentionne le nom de la liaison en ligne actuelle.

La **barre d'état** indique le moment (date et heure) auquel la tâche de forçage permanent a été lue dans la CPU.

La fenêtre est vide si aucune tâche de forçage permanent n'est active.

Les différents types d'**affichage de variables** dans cette fenêtre ont les significations suivantes :

Affichage	Signification
Affichage gras :	variables ayant déjà reçu une valeur fixe dans la CPU
Affichage normal :	variables en cours d'édition
Affichage estompé :	variables d'un module inexistant ou non enfiché ou variables avec erreur d'adressage, un message d'erreur s'affichera.



## Reprise d'opérandes forçables de la table des variables

Lorsque vous souhaitez reprendre des variables d'une table des variables dans la fenêtre des valeurs de forçage permanent, sélectionnez la table, puis les variables souhaitées. Ouvrez ensuite la fenêtre des valeurs de forçage permanent en choisissant la commande de menu **Variable > Valeurs de forçage permanent**. Les variables pouvant être forcées de manière permanente par le module sont reprises dans la fenêtre des valeurs de forçage permanent.

## Reprise de la tâche de forçage permanent de la CPU ou création d'une nouvelle tâche de forçage permanent

Un autre message s'affiche lorsque la fenêtre des valeurs de forçage permanent est ouverte et active :

- Si vous confirmez, les modifications dans la fenêtre des valeurs de forçage permanent sont remplacées par la tâche de forçage permanent se trouvant dans la CPU. La commande **Edition > Annuler** vous permet de rétablir le contenu précédent de la fenêtre.
- Si vous annulez, la fenêtre des valeurs de forçage permanent conserve son contenu actuel.  
Vous pouvez ensuite enregistrer le contenu de la fenêtre en tant que table de variables avec la commande **Table > Enregistrer sous** ou bien choisir la commande **Variable > Forçage permanent** : ainsi, le contenu en cours de la fenêtre des valeurs de forçage permanent est écrit dans la CPU comme nouvelle tâche de forçage permanent.

La visualisation et le forçage de variables ne sont possibles que dans la tables des variables, mais pas dans la fenêtre "Valeurs de forçage permanent".

## Suppression de valeurs de forçage permanent

La commande de menu **Variable > Afficher valeurs forçage permanent** ouvre la fenêtre des valeurs de forçage permanent. Vous pouvez ensuite choisir la commande de menu **Variable > Annuler forçage permanent** pour supprimer les valeurs de forçage permanent dans la CPU sélectionnée.

## Enregistrement d'une fenêtre de valeurs de forçage permanent

Vous pouvez mémoriser le contenu d'une fenêtre de valeurs de forçage permanent dans une table de variables. La commande **Insertion > Table de variables** permet d'insérer de nouveau le contenu mémorisé dans la fenêtre des valeurs de forçage permanent.

## Remarques sur les mnémoniques dans la fenêtre "Valeurs de forçage"

Les mnémoniques de la dernière fenêtre active sont repris, excepté lorsque vous appelez "Visualisation et forçage de variables" à partir d'une autre application qui ne dispose pas de mnémoniques.

La colonne "Mnémonique" ne figure pas dans la table si vous ne pouvez pas saisir de mnémoniques. Dans ce cas, la commande **Outils > Table des mnémoniques** n'est pas non plus disponible.

### 20.8.3 Différences entre forçage de variables et forçage permanent de variables

Le tableau suivant résume les différences entre forçage et forçage permanent.

Caractéristique / fonction	Forçage permanent avec S7-400 (incl. CPU 318-2DP)	Forçage permanent avec S7-300 (sans CPU 318-2DP)	Forçage
Mémentos (M)	oui	–	oui
Temporisations et compteurs (T, Z)	–	–	oui
Blocs de données (DB)	–	–	oui
Entrées de périphérie (PEB, PEW, PED)	oui	–	–
Sorties de périphérie (PAB, PAW, PAD)	oui	–	oui
Entrées et sorties (E, A)	oui	oui	oui
Le programme utilisateur peut écraser les valeurs de forçage/forçage perm.	–	oui	oui
Le remplacement de la valeur de forçage permanent prend effet sans interruption	oui	oui	–
Les variables conservent leurs valeurs après la fermeture de l'application	oui	oui	–
Les variables conservent leurs valeurs une fois la liaison à la CPU suspendue	oui	oui	–
Erreur d'adressage autorisée : par ex. : EW1 valeur de forçage/forçage permanent : 1 EW1 valeur de forçage/forçage permanent : 0	–	–	La dernière prend effet
Définition du déclenchement	Toujours déclenchement immédiat	Toujours déclenchement immédiat	Unique ou cyclique
La fonction ne s'applique qu'aux variables figurant dans la zone visible de la fenêtre active	S'applique à toutes les valeurs de forçage permanent	S'applique à toutes les valeurs de forçage permanent	oui

#### Nota

- Avec la fonction "Déblocage des sorties de périphérie", les valeurs de forçage permanent pour les sorties de périphérie concernées prennent effet aux modules correspondants, mais pas les valeurs de forçage pour les sorties de périphérie forçées de manière cyclique.
- En cas de forçage permanent, la variable possède toujours la valeur de forçage permanent. Cette valeur est lue dans le programme utilisateur à chaque accès en lecture. Tous les accès en écriture sont inefficaces.
- En cas de forçage cyclique, les accès en écriture du programme sont efficaces et le restent jusqu'au point de déclenchement suivant.

## 21 Test avec la visualisation d'état du programme

Vous pouvez tester votre programme en affichant, pour chaque instruction, l'état du programme (RLG, bit d'état) ou le contenu des registres correspondants. Vous sélectionnez les informations à afficher dans la page d'onglet "LIST" de la boîte de dialogue "Paramètres". Pour ouvrir cette boîte de dialogue, choisissez la commande **Outils > Paramètres** dans la fenêtre "CONT/LOG/LIST" : Programmation de blocs".



### Attention

Si vous effectuez le test d'une installation en marche, d'éventuels défauts de fonctionnement ou erreurs de programmation risquent d'occasionner des dommages matériels et personnels graves !

Avant d'exécuter une fonction, assurez-vous qu'aucune situation dangereuse ne peut se produire !

---

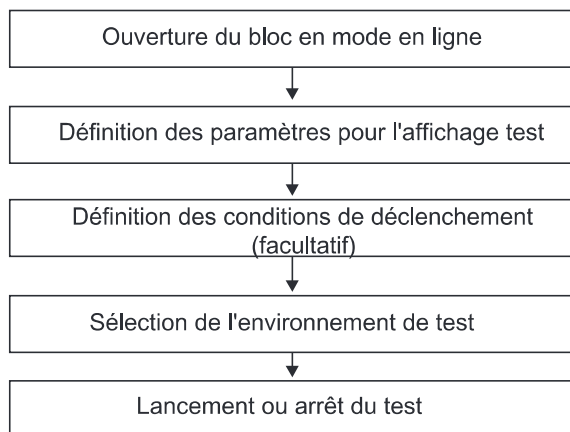
### Conditions préalables

Pour pouvoir afficher l'état du programme, il faut que les conditions suivantes soient remplies :

- Vous avez enregistré le bloc sans erreurs, puis l'avez chargé dans la CPU.
- La CPU est en marche, le programme utilisateur s'exécute.

### Marche à suivre de principe pour la visualisation de l'état du programme :

Il est fortement recommandé de ne pas appeler et tester immédiatement le programme complet, mais d'appeler et tester les blocs les uns après les autres. Ce faisant, il faut commencer par les blocs de niveau inférieur, c'est-à-dire les blocs au dernier niveau d'imbrication de la hiérarchie d'appel. Vous appelez, par exemple, ces blocs dans l'OB1 et créez l'environnement à tester pour ces blocs par visualisation et forçage des variables.



Pour effectuer un test en utilisant la fonction de visualisation d'état du programme, pour définir des points d'arrêt et pour exécuter le programme en mode pas à pas, vous devez sélectionner le mode de fonctionnement test (cf. commande **Test > Mode de fonctionnement**). En mode processus, ces fonctions de test ne sont pas possibles.

## 21.1 Affichage dans la visualisation d'état de programme

L'affichage de la **visualisation d'état de programme** est actualisé cycliquement. Il débute avec le réseau sélectionné.

### Valeurs par défaut

- Etat satisfait : lignes continues en vert
- Etat non satisfait : lignes pointillées en bleu
- Etat inconnu : lignes continues en noir

Vous pouvez modifier ces valeurs pour le type et la couleur des lignes dans la page d'onglet "CONT/LOG" que vous affichez via la commande **Outils > Paramètres**.

### Etat des éléments

- L'état d'un contact :
  - est satisfait lorsque l'opérande a la valeur "1" ;
  - n'est pas satisfait lorsque l'opérande a la valeur "0" ;
  - est inconnu lorsque la valeur de l'opérande est inconnue.
- L'état d'éléments avec sortie de validation (ENO) correspond à l'état d'un contact avec la valeur de la sortie ENO comme opérande.
- L'état d'éléments avec sortie Q correspond à l'état d'un contact avec la valeur de l'opérande.
- L'état pour des opérations CALL est satisfait lorsque le bit de résultat binaire est à 1 après l'appel.
- L'état d'une opération de saut est satisfait lorsque le saut est exécuté, c'est-à-dire lorsque la condition de saut est satisfaite.
- Les éléments avec sortie de validation (ENO) sont représentés en noir lorsque la sortie de validation n'est pas définie.

### Etat des lignes

- Les lignes sont en noir lorsqu'elles n'ont pas été empruntées ou que leur état est inconnu.
- L'état des lignes commençant à la barre d'alimentation est toujours satisfait ("1").
- L'état des lignes au début de branches parallèles est toujours satisfait ("1").
- L'état des lignes après un élément est satisfait lorsque l'état de la ligne avant l'élément et l'état de l'élément sont satisfaits.
- L'état de la ligne après NOT est satisfait lorsque l'état de la ligne avant NOT n'est pas satisfait (et inversement).
- L'état de la ligne **après** la jonction de plusieurs lignes est satisfait :
  - lorsque, d'une part, l'état d'une ligne au moins **avant** la jonction est satisfait
  - et que, d'autre part, l'état de la ligne avant l'ouverture de la branche ou des branches est satisfait.

## Etat des paramètres

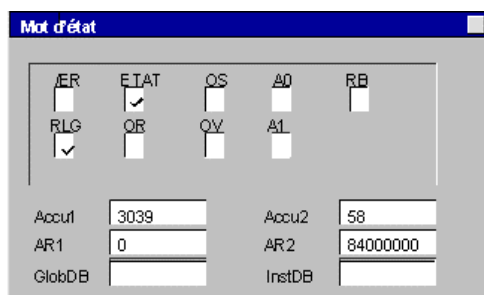
- Les valeurs de paramètres **en gras** sont les valeurs en cours.
- Les valeurs de paramètres en écriture normale proviennent d'un cycle précédent ; il n'y a pas eu de passage par cet endroit du programme pendant le cycle en cours.

## 21.2 Informations sur le test en mode pas à pas et sur les points d'arrêt

Lors du test en mode pas à pas, vous pouvez :

- traiter des programmes instruction par instruction (pas à pas) ;
- définir des points d'arrêt.

La fonction "Test en mode pas à pas" n'est pas réalisée dans tous les automates programmables (voyez la documentation de votre automate).



### Conditions préalables

- Vous devez avoir sélectionné mode test. Le test en mode pas à pas n'est pas possible en mode processus (cf. commande **Test > Mode de fonctionnement**).
- Le test en mode pas à pas n'est possible qu'en LIST. Pour les blocs en CONT ou en LOG, vous devez d'abord changer l'affichage à l'aide de la commande **Affichage > LIST**.
- Le bloc ne doit pas être protégé.
- Le bloc doit être ouvert en ligne.
- Il ne faut pas que le bloc ouvert ait été modifié dans l'éditeur.

### Nombre de points d'arrêt

Le nombre de points d'arrêt varie et dépend :

- du nombre de points d'arrêt déjà définis,
- du nombre d'états de variable en cours,
- du nombre d'états de programme en cours.

Consultez votre documentation pour savoir si votre automate programmable prend en charge le test en mode pas à pas.

Les commandes vous permettant de définir, d'activer ou de supprimer des points d'arrêt font partie du menu "Test". Vous avez en outre la possibilité de choisir ces commandes à l'aide des boutons correspondants dans la barre des points d'arrêt. Pour afficher la barre des points d'arrêt, choisissez la commande **Affichage > Barre de points d'arrêt**.

### **Fonctions de test autorisées**

- Visualisation et forçage de variables
- Etat du module
- Etat de fonctionnement



#### **Danger**

Attention aux états dangereux de l'installation dans l'état de fonctionnement "Attente".

---

## 21.3 Informations sur l'état de fonctionnement "Attente"

Lorsque le programme atteint un point d'arrêt, l'automate programmable passe à l'état de fonctionnement "Attente".

### Signalisation des diodes électroluminescentes (DEL) à l'état "Attente"

- La DEL RUN clignote.
- La DEL STOP est allumée.

### Traitement du programme à l'état de fonctionnement "Attente"

- Le code S7 n'est pas traité à l'état "Attente" : aucun niveau d'exécution n'est plus traité.
- Tous les temps sont suspendus :
  - pas de traitement des cellules de temporisation,
  - arrêt de tous les temps de surveillance,
  - arrêt des impulsions de base des niveaux déclenchés par horloge.
- L'horloge temps réel continue à fonctionner.
- Pour des raisons de sécurité, les sorties sont toujours inhibées à l'état de fonctionnement "Attente" (voir "output disable" des modules de sorties).

### Comportement en cas de coupure secteur à l'état de fonctionnement "Attente"

- Si, lorsqu'un automate programmable avec sauvegarde est à l'état "Attente", il y a une coupure secteur suivie d'un retour de tension, cet automate passe à l'état de fonctionnement "Arrêt" (STOP) et y reste. La CPU n'exécute pas de mise en route automatique. Vous décidez vous-même, à partir de l'état "Arrêt", de la réaction appropriée (par exemple, définir ou effacer des points d'arrêt, exécuter une mise en route manuelle).
- Les automates programmables sans sauvegarde n'ont pas de "mémoire" et exécutent donc un démarrage automatique (démarrage à chaud) lors du retour de la tension, quel qu'ait été l'état de fonctionnement précédent.



## 21.4 Etat du programme de blocs de données

A partir de la version 5 de STEP 7, il est possible de visualiser un bloc de données dans la vue des données en ligne. Cet affichage peut être activé aussi bien depuis un bloc de données en ligne que depuis un bloc de données hors ligne. Dans les deux cas, c'est le contenu du bloc de données en ligne du système cible qui est affiché.

Le bloc de données ne doit pas être modifié avant l'appel de l'état du programme. En cas de différence structurelle (déclaration) entre le bloc de données en ligne et le bloc de données hors ligne, vous pouvez directement charger le bloc de données hors ligne dans le système cible.

Le bloc de données doit se trouver dans la "vue des données", afin que les valeurs en ligne puissent être représentées dans la colonne "Valeur actuelle". Seule la partie du bloc de données visible à l'écran est actualisée. Pendant que l'état est actif, vous ne pouvez pas passer à la vue des déclarations.

Durant l'actualisation, la barre de défilement verte est visible dans la barre d'état du bloc de données et l'état de fonctionnement est affiché.

Les valeurs sont affichées dans le format du type de données respectif. Une modification du format n'est pas possible.

Lorsque vous mettez fin à l'état du programme, le contenu qui était préalablement valable s'affiche à nouveau dans la colonne des valeurs actuelles. Vous ne pouvez pas reprendre les valeurs en ligne actualisées dans le bloc de données hors ligne.

### Actualisation de types de données

Tous les types de données simples sont aussi bien actualisés dans un DB global que dans toutes les déclarations (in/out/inout/stat) d'un bloc de données d'instance.

Certains types de données ne peuvent pas être actualisés. Lorsque l'état du programme est activé, les champs contenant des données non actualisées sont estompés dans la colonne "Valeur actuelle".

- Les types de données complexes DATE\_AND\_TIME et STRING ne sont pas actualisés.
- Dans les types de données complexes ARRAY, STRUCT, UDT, FB, SFB, seuls les éléments qui sont des types de données simples sont actualisés.
- Dans la déclaration INOUT d'un bloc de données d'instance, seul le pointeur sur le type de données complexe est représenté mais ses éléments ne le sont pas. Le pointeur n'est pas actualisé.
- Les types de paramètre ne sont pas actualisés.
- Voir aussi
- Introduction à la visualisation de variables

## 21.5 Définition de l'affichage de l'état du programme

Vous pouvez définir vous-même l'affichage de l'état du programme dans un bloc LIST, LOG ou CONT.

Procédez de la manière suivante :

1. Choisissez la commande **Outils > Paramètres**.
2. Sélectionnez, dans la boîte de dialogue "CONT/LOG/LIST", l'onglet "LIST" ou l'onglet "CONT/LOG".
3. Choisissez les options désirées pour le test. Vous pouvez afficher les champs d'état suivants :

Lorsque vous cochez ...	s'affiche :
Bit d'état	Bit d'état, c.-à-d. bit 2 du mot d'état
Résultat logique	Bit 1 du mot d'état ; indique le résultat d'une fonction logique ou d'une comparaison arithmétique.
Standard	Contenu de l'ACCU 1.
Registre d'adresse 1/2	Contenu du registre d'adresse correspondant dans le cas de l'adressage indirect (intrazone ou interzone) par registre
Accu 2	Contenu de l'ACCU 2
Registre des DB 1/2	Contenu du registre des DB du premier ou du deuxième bloc de données ouvert.
Indirect	Référence mémoire indirecte ; indication du pointeur (adresse), pas du contenu de l'adresse ; possible uniquement pour l'adressage indirect en mémoire, pas pour l'adressage indirect par registre. Contenu d'un mot de temporisation ou d'un mot de compteur lorsque l'instruction comporte les opérations correspondantes.
Mot d'état	Tous les bits d'état du mot d'état

## 21.6 Définition du mode de fonctionnement pour le test

### Marche à suivre

1. Choisissez la commande **Test > Mode de fonctionnement** pour afficher l'environnement de test sélectionné.
2. Sélectionnez le mode de fonctionnement souhaité. Vous disposez du mode test et du mode processus.

Mode de fonctionnement	Signification
Mode test	Toutes les fonctions de test peuvent être utilisées sans restrictions. D'importantes augmentations du temps de cycle de la CPU sont possibles, puisque l'état des instructions dans des boucles programmées est, par exemple, déterminé à chaque passage.
Mode processus	La fonction de test "Etat du programme" est restreinte pour garantir une charge du temps de cycle la plus faible possible. <ul style="list-style-type: none"><li>• Ainsi, par exemple, les conditions d'appel ne sont pas autorisées.</li><li>• La visualisation d'état d'une boucle programmée est interrompue à la position de retour.</li><li>• Les fonctions de test "ATTENTE" et l'exécution du programme pas à pas ne sont pas possibles.</li></ul>

### Nota

Si le mode de test a été défini dans le cadre du paramétrage de la CPU, vous ne pouvez le modifier que par commutation entre les modes de fonctionnement test et processus. Sinon, vous pouvez le modifier dans la boîte de dialogue affichée.



## **22 Test avec le programme de simulation (logiciel optionnel)**

### **22.1 Test avec le programme de simulation S7-PLCSIM (logiciel optionnel)**

Le logiciel optionnel de simulation vous permet d'exécuter et de tester votre programme dans un système d'automatisation que vous simulez dans votre ordinateur ou dans votre console de programmation (par exemple une Power PG). La simulation étant complètement réalisée au sein du logiciel STEP 7, il n'est pas nécessaire que vous soyez connecté à un matériel S7 quelconque (CPU ou modules de signaux). La CPU S7 simulée vous permet de tester les programmes destinés aussi bien aux CPU S7-300 qu'aux CPU S7-400 et de remédier à d'éventuelles erreurs.

Cette application dispose d'une interface simple vous permettant de surveiller et de modifier les différents paramètres utilisés par le programme (comme par exemple d'activer ou de désactiver des entrées). Tout en exécutant votre programme dans la CPU simulée, vous avez en outre la possibilité de mettre en œuvre les différents applications du logiciel STEP 7, comme par exemple la table des variables afin d'y visualiser et d'y forcer des variables.

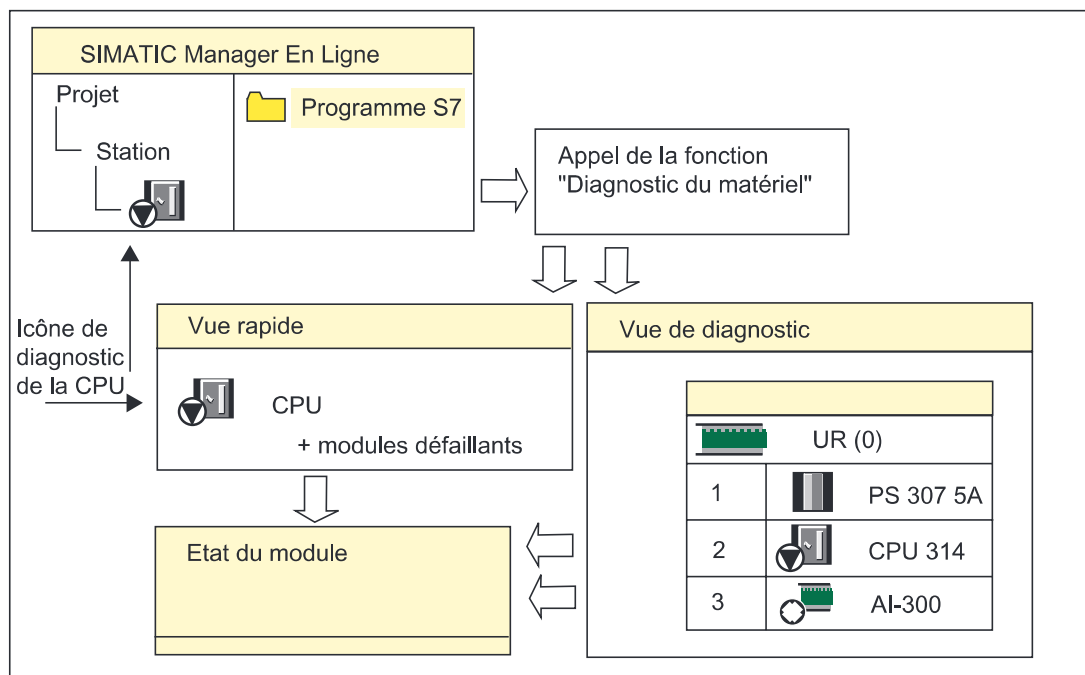


## 23 Diagnostic

### 23.1 Diagnostic du matériel et recherche d'erreurs

Des icônes de diagnostic vous permettent de détecter la présence d'informations de diagnostic pour un module. Elles indiquent l'état du module concerné et, pour les CPU, également leur état de fonctionnement.

Les icônes de diagnostic s'affichent dans la vue en ligne de la fenêtre du projet, dans la vue rapide (présélection) ou encore dans la vue de diagnostic lorsque vous appelez la fonction "Diagnostic du matériel". Des informations de diagnostic détaillées sont données par l'"Etat du module" que vous appellerez par double clic sur une icône de diagnostic dans la vue rapide ou dans la vue de diagnostic.



### Marche à suivre pour localiser les défauts

1. Ouvrez la fenêtre en ligne du projet en choisissant la commande **Affichage > En ligne**.
2. Ouvrez toutes les stations de sorte que les modules programmables qui y sont configurés s'affichent.
3. Vérifiez pour quelle CPU une icône de diagnostic est affichée pour signaler une erreur ou un défaut. En appuyant sur la touche F1, vous obtenez une page d'aide avec les explications relatives aux icônes de diagnostic.
4. Sélectionnez la station que vous souhaitez examiner.
5. Choisissez la commande **Système cible > Diagnostic/Paramètres > Etat du module** pour afficher l'état du module de la CPU appartenant à cette station.
6. Choisissez la commande **Système cible > Diagnostic/Paramètres > Diagnostic du matériel** pour afficher la "vue rapide" avec la CPU et les modules défectueux de cette station. L'affichage de la vue rapide est présélectionné (commande **Outils > Paramètres**, page d'onglet "Affichage").
7. Sélectionnez un module défectueux dans la vue rapide.
8. Cliquez sur le bouton "Etat du module" pour obtenir les informations sur ce module.
9. Dans la vue rapide, cliquez sur le bouton "Station en ligne" pour afficher la vue de diagnostic. La vue de diagnostic affiche tous les modules de la station dans la disposition des emplacements.
10. Effectuez un double clic sur un module dans la vue de diagnostic pour en afficher l'état correspondant. Vous obtenez ainsi également des informations sur les modules non défectueux, qui ne sont donc pas affichés dans la vue rapide.

Il n'est pas impératif de réaliser la totalité de ces étapes et vous pouvez vous arrêter dès que vous avez trouvé l'information de diagnostic recherchée.






## 23.2 Icônes de diagnostic dans la vue en ligne

Les icônes de diagnostic s'affichent aussi bien dans la fenêtre en ligne du projet que dans la vue en ligne des tables de la fenêtre de configuration du matériel.






Les icônes de diagnostic vous facilitent la recherche d'erreur en cas de défaut. Un coup d'œil sur l'icône du module vous indique s'il y a des informations de diagnostic à son sujet. En cas de fonctionnement sans erreur, les icônes des types de module sont représentées sans icône de diagnostic supplémentaire.

Quand il y a des informations de diagnostic au sujet d'un module, une icône de diagnostic s'ajoute à celle du module, ou l'icône du module est représentée estompée.


### Icônes de diagnostic pour modules (exemple FM/CPU)

Icône	Signification
	La configuration sur site diffère de la configuration prévue : le module configuré n'est pas enfiché ou un autre type de module est enfiché.
	Erreur : module défectueux. Causes possibles : détection d'une alarme de diagnostic, d'une erreur d'accès à la périphérie ou d'une DEL d'erreur.
	Le diagnostic n'est pas possible, parce qu'il n'y a pas de liaison en ligne ou que la CPU ne fournit pas d'informations de diagnostic sur le module (par ex. alimentation en courant, cartouches).

### Icônes de diagnostic pour états de fonctionnement (à l'exemple d'une CPU)

Icône	Etat de fonctionnement
	Mise en route
	Arrêt
	Arrêt déclenché par l'état d'arrêt d'une autre CPU en fonctionnement multiprocesseur
	Marche
	Attente

## Icône de diagnostic pour forçage permanent

Icône	Etat de fonctionnement
	<p>Un forçage permanent de variables est effectué sur ce module, ce qui signifie que certaines variables du programme utilisateur ont reçu des valeurs fixes que le programme ne peut pas modifier.</p> <p>La marque de forçage permanent peut être combinée avec d'autres icônes (elle l'est ici avec l'icône représentant l'état Marche).</p>

## Actualisation de l'affichage des icônes de diagnostic

La fenêtre correspondante doit être activée.

- Appuyez sur la touche de fonction F5 ou
- choisissez la commande **Affichage > Actualiser** dans la fenêtre.

## 23.3 Diagnostic du matériel : vue rapide

### 23.3.1 Appel de la vue rapide

La vue rapide vous permet de parvenir rapidement dans le "Diagnostic du matériel" en fournissant des informations réduites par rapport aux informations complètes affichées dans HW Config. La vue rapide s'affiche par défaut à l'appel de la fonction "Diagnostic du matériel".

#### Affichage de la vue rapide

Vous appelez cette fonction dans SIMATIC Manager en choisissant la commande **Système cible > Diagnostic/Paramètres > Diagnostic du matériel**.

Vous pouvez utiliser cette commande de la manière suivante :

- dans la fenêtre en ligne du projet, lorsqu'un module ou un programme S7/M7 sont sélectionnés,
- dans la fenêtre "Partenaires accessibles", lorsqu'un partenaire ("MPI=...") est sélectionné et que cette entrée appartient à une CPU.

Dans les tables de configuration ouvertes, vous pouvez alors sélectionner des modules pour lesquels vous souhaitez afficher l'état.

### 23.3.2 Fonctions d'information de la vue rapide

La vue rapide affiche les informations suivantes :

- données pour la liaison en ligne à la CPU,
- icône de diagnostic de la CPU,
- icônes de diagnostic des modules pour lesquels la CPU a détecté un défaut (par exemple, alarme de diagnostic, erreur d'accès à la périphérie),
- type et adresse du module (profilé support/châssis, emplacement d'enchâssage, réseau maître DP avec numéro de station).

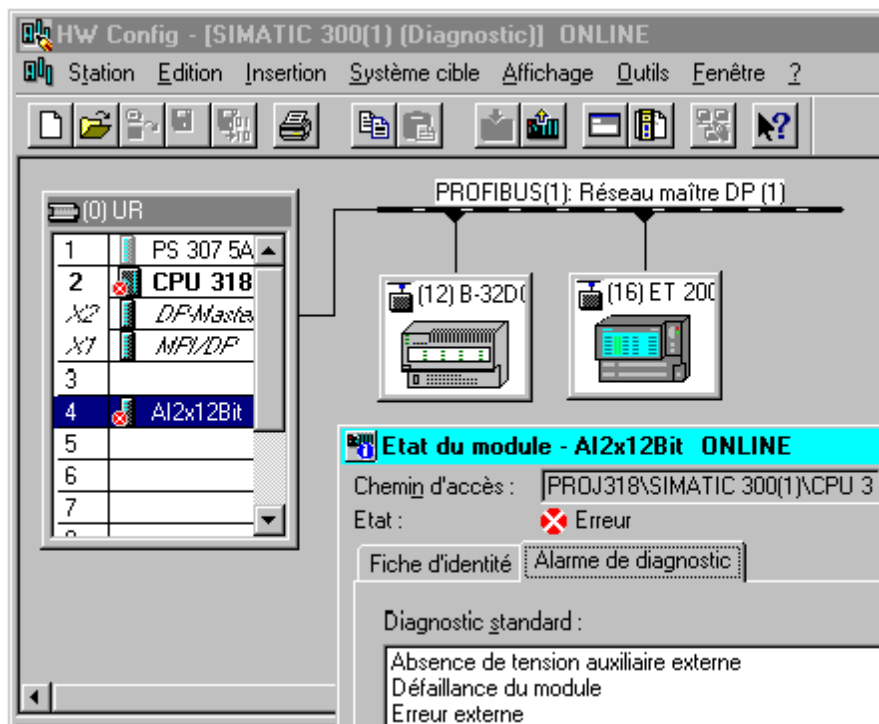
#### Autres possibilités de diagnostic dans la vue rapide

- **Affichage de l'état du module**  
Vous appelez cette boîte de dialogue en cliquant sur le bouton "Etat du module". Selon l'aptitude au diagnostic du module, vous y obtenez des informations détaillées sur le module sélectionné. L'état du module de la CPU vous permet en particulier d'afficher les entrées dans la mémoire tampon de diagnostic.
- **Affichage de la vue de diagnostic**  
En cliquant sur le bouton "Station en ligne" vous appelez cette boîte de dialogue qui, contrairement à la vue rapide, fournit une représentation graphique de l'ensemble de la station ainsi que des informations sur la configuration. Vous êtes positionné sur le module qui est sélectionné dans la liste "CPU / Modules défectueux".

## 23.4 Diagnostic du matériel : vue du diagnostic

### 23.4.1 Appel de la vue de diagnostic de HW Config

Cette méthode vous permet d'afficher la boîte de dialogue à onglets "Etat du module" pour tous les modules du profilé support ou châssis. La vue de diagnostic (table de configuration) montre la composition effective d'une station au niveau des profilés supports ou châssis et des stations DP avec leurs modules.



#### Nota

- Si la table de configuration est déjà ouverte hors ligne, la commande **Station > Ouvrir en ligne** vous donne également la vue en ligne des tables de configuration.
- La boîte de dialogue à onglets "Etat du module" affiche un nombre variable d'onglets selon les fonctions de diagnostic réalisées par le module.
- La fenêtre "Partenaires accessibles" affiche exclusivement les modules possédant leur propre adresse de réseau (adresse Ethernet, MPI ou PROFIBUS).

---

### Appel dans SIMATIC Manager, depuis la vue EN LIGNE d'un projet

1. Dans la vue du projet de SIMATIC Manager, établissez une liaison en ligne avec le système cible en choisissant la commande **Affichage > En ligne**.
2. Sélectionnez une station et ouvrez-la par double-clic.
3. Ouvrez l'objet "Matériel" qu'elle contient ; La vue du diagnostic s'ouvre.

Vous pouvez à présent sélectionner un module et en appeler l'état en choisissant la commande **Système cible > Diagnostic/Paramètres > Etat du module**.

### Appel dans SIMATIC Manager, depuis la vue hors ligne d'un projet

Procédez de la manière suivante :

1. Dans la vue du projet de SIMATIC Manager, sélectionnez une station et ouvrez-la par double clic.
2. Ouvrez l'objet "Matériel" qu'elle contient ; La table de configuration s'ouvre.
3. Choisissez la commande **Station > Ouvrir en ligne**.
4. La vue de diagnostic de HW Config s'ouvre avec la configuration de station telle qu'elle a été fournie par les modules (par exemple CPU). L'état des modules est représenté par des icônes. La signification des icônes est donnée dans l'aide en ligne. Les modules défectueux ou manquants sont énumérés dans une boîte de dialogue à part. Vous pouvez passer directement de cette boîte à l'un des modules mentionnés (bouton "Aller à").
5. Cliquez deux fois sur l'icône du module dont vous souhaitez connaître l'état. Une boîte de dialogue à onglets (qui dépend du type de module) vous permet de réaliser une analyse détaillée de l'état du module.

### Appel dans SIMATIC Manager, depuis la fenêtre "Partenaires accessibles"

Procédez de la manière suivante :

1. Dans SIMATIC Manager, choisissez la commande **Système cible > Partenaires accessibles** pour ouvrir la fenêtre "Partenaires accessibles".
2. Sélectionnez un partenaire dans la fenêtre "Partenaires accessibles".
3. Choisissez la commande **Système cible > Diagnostic/Paramètres > Diagnostic du matériel**.

---

#### Nota

La fenêtre "Partenaires accessibles" affiche exclusivement les modules possédant leur propre adresse de réseau (adresse Ethernet, MPI ou PROFIBUS).

---

### 23.4.2 Fonctions d'information de la vue du diagnostic

Contrairement à la vue rapide, la vue de diagnostic affiche l'ensemble de la configuration de la station accessible en ligne. Celle-ci comprend :

- la configuration des profilés support/châssis,
- les icônes de diagnostic de **tous** les modules configurés  
L'état des modules respectifs est donc affiché et pour les CPU, également l'état de fonctionnement.
- le type de module, le numéro de référence, des informations sur les adresses et des commentaires sur la configuration.

#### Autres possibilités de diagnostic dans la vue de diagnostic

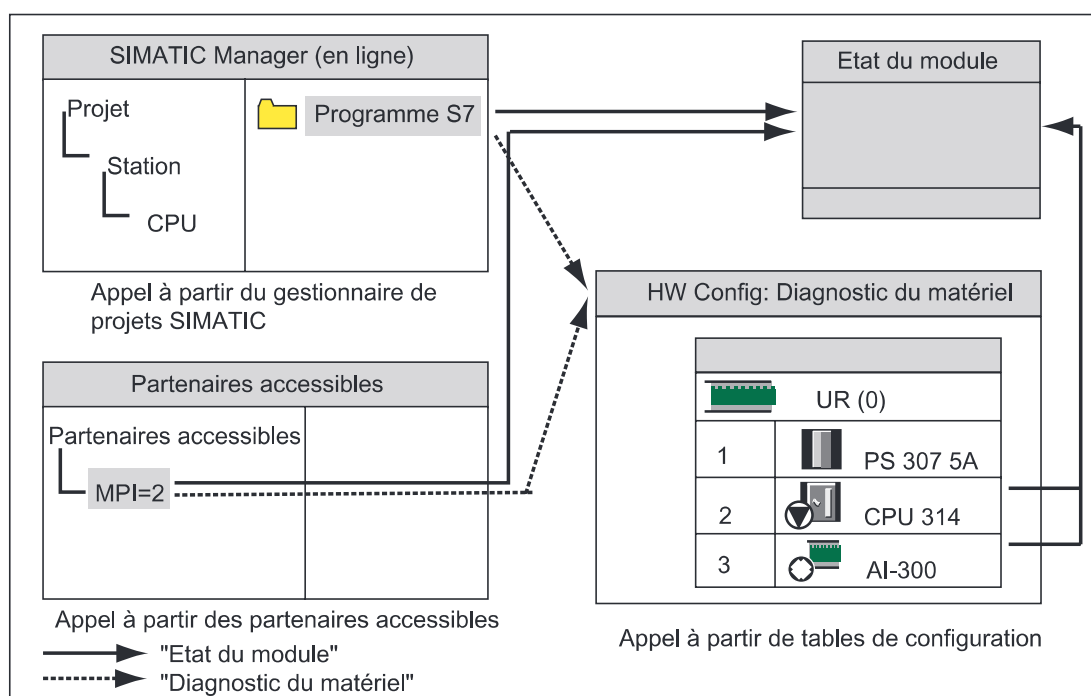
En effectuant un double clic sur un module, vous pouvez en afficher l'état.

## 23.5 Etat du module

### 23.5.1 Possibilités d'appel de l'état du module

Vous pouvez afficher la boîte de dialogue "Etat du module" depuis des points de départ différents. Les procédés ci-dessous sont cités à titre d'exemple, ils sont d'un emploi fréquent.

- Appel dans SIMATIC Manager, depuis une fenêtre avec la vue du projet "En ligne" ou "Hors ligne".
- Appel dans SIMATIC Manager, depuis une fenêtre "Partenaires accessibles".
- Appel depuis la vue de diagnostic de HW Config.



Pour que vous puissiez interroger l'état d'un **module possédant sa propre adresse de réseau**, il faut que vous ayez établi une liaison en ligne avec le système cible. C'est ce que vous faites dans la vue du projet en ligne ou dans la fenêtre "Partenaires accessibles".

## 23.5.2 Fonctions d'information de l'état du module

Les fonctions d'informations sont disponibles dans la page d'onglet de même nom dans la boîte de dialogue "Etat du module". Dans votre exemple d'application concret, seules les pages d'onglet significatives pour le module sélectionné sont affichées.

Fonction d'information	Information	Utilisation
Général	Données d'identification du module sélectionné, par exemple type, numéro de référence, version, état, emplacement dans le châssis/profilé support.	Les informations en ligne du module enfiché peuvent être comparées avec les données de configuration du module.
Mémoire tampon de diagnostic	Vue d'ensemble des événements dans la mémoire tampon de diagnostic ainsi qu'informations détaillées sur l'événement sélectionné.	Pour évaluer la cause du passage à l'état "Arrêt" d'une CPU et pour évaluer les événements précédents sur le module sélectionné.  Grâce à la mémoire tampon de diagnostic, les erreurs dans le système peuvent être évaluées, même bien plus tard, en vue de déterminer l'origine d'un passage à l'état "Arrêt" ou de remonter la trace des événements de diagnostic individuels.
Alarme de diagnostic	Données de diagnostic du module sélectionné.	Pour déterminer la cause d'un défaut de module.
Diagnostic de l'esclave DP	Données de diagnostic de l'esclave DP sélectionné (selon EN 50170)	Pour déterminer la cause d'une erreur d'un esclave DP
Mémoire	Organisation de la mémoire, occupation actuelle de la mémoire de travail, de la mémoire de chargement et des données de la mémoire rémanente de la CPU ou du FM de M7 sélectionné.	Avant de transmettre de nouveau blocs ou des blocs étendus sur une CPU, pour vérifier si la mémoire de chargement est suffisante dans cette CPU/ce FM ainsi que pour comprimer le contenu de la mémoire.
Temps de cycle	Durée du cycle le plus long, du cycle le plus court et du dernier cycle de la CPU ou du FM de M7.	Pour contrôler le temps de cycle minimal paramétré ainsi que les temps de cycle maximal et actuel.
Horodatage	Heure actuelle, compteur d'heures de fonctionnement et informations pour la synchronisation des horloges (intervalles de synchronisation).	Pour afficher l'heure et la date d'un module et contrôler la synchronisation des horloges
Performances	Plages d'opérandes et blocs disponibles pour le module (CPU/FM) sélectionné.	Avant et pendant la création d'un programme utilisateur et pour vérifier si la CPU présente les conditions requises pour l'exécution d'un programme utilisateur, par exemple quant à la taille de la mémoire image.
	Affichage de tous les types de blocs disponibles dans le module sélectionné. Liste des OB, SFB, et SFC pouvant être utilisés dans ce module.	Pour vérifier quels blocs standard votre programme utilisateur peut contenir ou appeler pour pouvoir s'exécuter dans la CPU choisie.



Fonction d'information	Information	Utilisation
Communication	Vitesses de transmission, les liaisons établies, la charge due à la communication ainsi que la taille maximale des télégrammes sur le bus K du module sélectionné	Pour vérifier combien de liaisons et quelles liaisons de la CPU ou du FM de M7 sont possibles ou affectées.
Piles	Onglet <b>Piles</b> : vous ne pouvez ouvrir cet onglet qu'à l'état d'arrêt ou d'attente. La pile des blocs (pile B) du module sélectionné s'affiche. Vous pouvez en outre lire la pile des interruptions (pile I), la pile des données locales (pile L), ainsi que la pile des parenthèses et sauter dans le bloc à l'endroit où l'erreur a causé une interruption.	Pour trouver la cause d'un passage à l'état "Arrêt" et pour corriger un bloc.

### Informations supplémentaires affichées

Les informations suivantes figurent dans chaque page d'onglet :

- chemin d'accès en ligne du module sélectionné,
- état de fonctionnement de la CPU concernée (par exemple "Marche", "Arrêt"),
- état du module sélectionné (par exemple Erreur, OK),
- état de fonctionnement du module sélectionné (par exemple "Marche", "Arrêt") si celui-ci possède son propre état de fonctionnement (par exemple CP342-5).

Il n'est pas possible d'afficher l'état de fonctionnement de la CPU elle-même ni l'état du module sélectionné si l'état du module d'un module autre qu'une CPU a été interrogé depuis la fenêtre "Partenaires accessibles".

### Affichage simultané de plusieurs modules

Vous pouvez interroger et afficher simultanément l'état de plusieurs modules. Pour cela, vous devez retourner au contexte de module qui vous intéresse, sélectionner un autre module et en appeler l'état. Une autre boîte de dialogue à onglets vous est alors proposée. Mais vous ne pouvez ouvrir qu'une boîte de dialogue à onglets par module.

### Actualisation de l'affichage de l'état du module

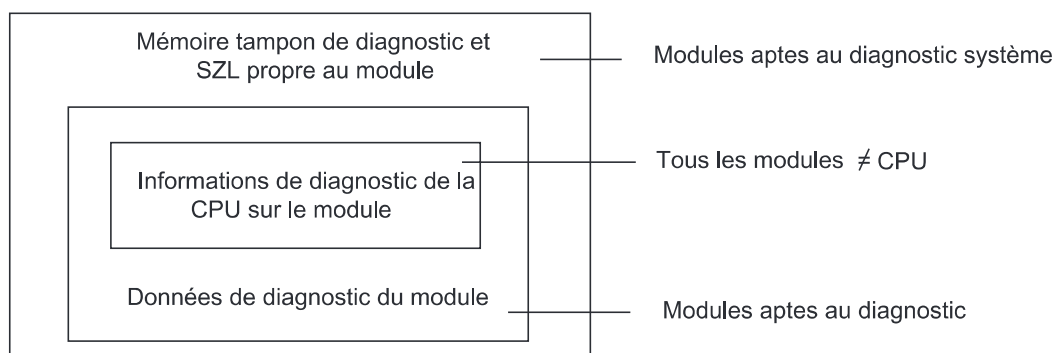
Une nouvelle lecture des données du module a lieu à chaque activation d'une nouvelle page d'onglet de la boîte de dialogue "Etat du module". En revanche, le contenu des pages d'onglet n'est pas mis à jour automatiquement pendant l'affichage d'une page. Pour effectuer une nouvelle lecture des données actuelles du module sans changer de page d'onglet, il vous suffit de cliquer sur le bouton "Actualiser".

### 23.5.3 Volume d'informations selon le type de module dans l'état du module

Le volume des informations susceptibles d'être évaluées et affichées dépend

- du module sélectionné et
- de la vue depuis laquelle vous interrogez l'état du module.  
 Quand vous interrogez l'état du module depuis la vue en ligne des tables de configuration ou depuis la fenêtre du projet, vous obtenez le volume complet des informations.  
 Quand vous interrogez l'état du module depuis la vue de projet "Partenaires accessibles", vous obtenez un volume d'informations restreint.

Selon le volume des informations, on distingue entre modules à diagnostic système, modules à diagnostic ou modules sans diagnostic. C'est ce qu'illustre le schéma ci-dessous.



- Les modules FM 351 et FM 354, par exemple, possèdent des fonctions de diagnostic système.
- La plupart des modules SM analogiques possèdent des fonctions de diagnostic.
- La plupart des modules SM TOR ne possèdent pas de fonctions de diagnostic.

## Pages d'onglet affichées

Le tableau précise quelles pages d'onglets s'affichent dans la boîte de dialogue "Etat du module" pour les différents types de modules.

Onglet	CPU ou FM de M7	Module à diagnostic système	Module à diagnostic	Module sans fonction de diagnostic	Esclave DP normalisé
Général	oui	oui	oui	oui	oui
Mémoire tampon de diagnostic	oui	oui	–	–	–
Alarme de diagnostic	–	oui	oui	–	–
Mémoire	oui	–	–	–	–
Temps de cycle	oui	–	–	–	–
Horodatage	oui	–	–	–	–
Performances	oui	–	–	–	–
Piles	oui	–	–	–	–
Communication	oui	–	–	–	–
Diagnostic de l'esclave DP	–	–	–	–	oui
Etat H <sup>1)</sup>	oui	–	–	–	–

<sup>1)</sup> uniquement pour les CPU dans les systèmes H

Outre les informations contenues dans les pages d'onglet, l'état de fonctionnement est affiché pour les modules qui en possèdent un. Quand vous interrogez l'état du module depuis les tables de configuration en ligne, c'est l'état du point de vue de la CPU qui est indiqué (par exemple ok, erreur, module inexistant).

### 23.5.4 Affichage de l'état du module d'appareils de terrain PA et d'esclaves DP placés derrière un Y-Link

A partir du Servicepack 3 de STEP 7 V5.1, l'état du module peut également être exploité pour des esclaves DP et des appareils de terrain PA placés "derrière" un DP/PA-Link (IM 157).

Ceci concerne les configurations suivantes :

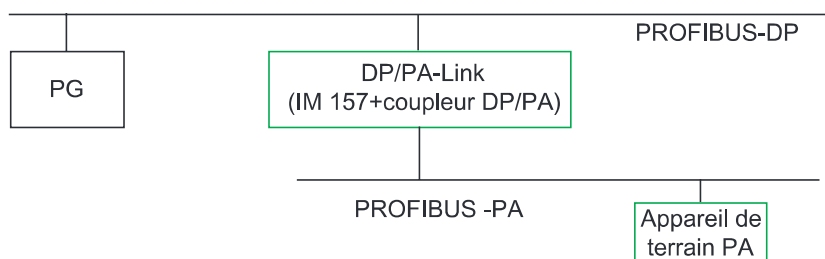
- IM 157 avec coupleurs DP/PA pour la connexion de PROFIBUS-PA.
- IM 157 utilisé comme module de couplage redondant pour la connexion d'un PROFIBUS DP non redondant ("Y-Link").

Dans ces configurations, la PG est reliée au même sous-réseau PROFIBUS que le DP/PA-Link.

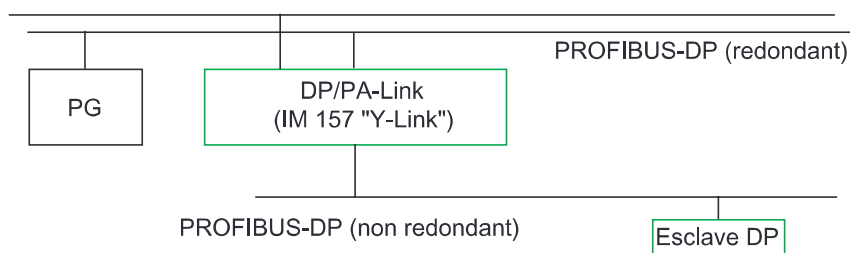
Il existe encore une autre possibilité de configuration, dans laquelle la PG est reliée au réseau Industrial Ethernet et effectue le routage d'une station S7-400 sur le sous-réseau PROFIBUS.

Les conditions requises sont décrites dans les figures suivantes.

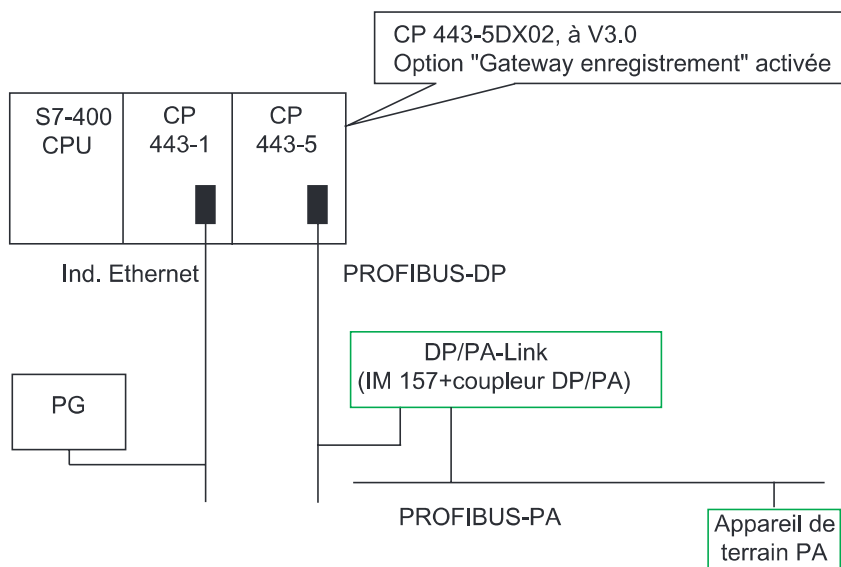
#### IM 157 avec coupleurs DP/PA pour la connexion de PROFIBUS-PA



#### IM 157 en tant que Y-Link



## PG sur Industrial Ethernet



## 23.6 Diagnostic à l'état de fonctionnement STOP

### 23.6.1 Marche à suivre pour déterminer la cause d'un passage à l'état d'arrêt

Pour déterminer la cause d'un passage à l'arrêt de la CPU, procédez de la manière suivante :

1. Sélectionnez la CPU qui est passée à l'état d'arrêt.
2. Choisissez la commande **Système cible > Diagnostic/Paramètres > Etat du module**.
3. Choisissez l'onglet "Tampon de diagnostic".
4. Les dernières entrées vous permettent de déterminer la cause du passage à l'arrêt.

Pour une erreur de programmation :

1. L'entrée "Arrêt car OB d'erreur de programmation non chargé", par exemple, signifie que la CPU a détecté une erreur de programmation, puis a tenté de démarrer l'OB (manquant) de traitement de l'erreur de programmation. L'erreur de programmation est indiquée par l'entrée précédente.
2. Sélectionnez le message d'erreur de programmation.
3. Cliquez sur le bouton "Ouvrir le bloc".
4. Sélectionnez la page d'onglet "Piles".

### 23.6.2 Contenu des piles à l'état d'arrêt

L'exploitation de la mémoire de diagnostic et du contenu des piles vous permet de déterminer la cause d'un défaut dans l'exécution d'un programme utilisateur.

Lorsque la CPU passe à l'état d'arrêt, par exemple suite à une erreur de programmation ou à une instruction d'arrêt, la pile des blocs s'affiche dans la page d'onglet "Piles" de l'état du module. Vous pouvez afficher d'autres contenus de piles grâce aux boutons "Pile des interruptions", "Pile des données locales" et "Pile des parenthèses". Le contenu des piles vous indique quelle instruction dans quel bloc a entraîné le passage à l'état "Arrêt" (STOP) de la CPU.

#### Contenu de la pile des blocs

La pile des blocs donne la liste de tous les blocs appelés avant le passage à l'état de fonctionnement "Arrêt" (STOP) et qui n'ont pas encore été exécutés jusqu'à la fin.

## Contenu de la pile des interruptions

Vous obtenez des informations sur l'emplacement d'interruption lorsque vous cliquez sur le bouton "Pile I". La pile des interruptions contient les données et les états qui étaient valables au moment de l'interruption, par exemple :

- contenu des accumulateurs et des registres,
- DB ouverts et leur taille,
- contenu du mot d'état,
- classe de priorité,
- bloc interrompu,
- bloc dans lequel l'exécution du programme a été poursuivie après l'interruption.

## Contenu de la pile des données locales

Pour chacun des blocs énumérés dans la pile B, vous pouvez afficher les données locales correspondantes en sélectionnant le bloc et en cliquant sur le bouton "Pile L".

La pile des données locales (pile L) contient les valeurs des données locales des blocs que le programme utilisateur a utilisé jusqu'à l'interruption.

L'interprétation et l'exploitation des données locales affichées demandent de très bonnes connaissances du système. La partie avant des données affichées correspond aux variables temporaires du bloc.

## Contenu de la pile des parenthèses

Quand vous cliquez sur le bouton "Pile P", le contenu de la pile des parenthèses est représenté à l'endroit de l'interruption.

La pile des parenthèses est une zone de mémoire utilisée par les opérations combinatoires **U**(, **UN**(, **O**(, **ON**(, **X**( et **XN**(.

Le bouton n'est actif que lorsqu'il y a encore des expressions entre parenthèses ouvertes au moment de l'interruption.

## 23.7 Contrôle des temps de cycle pour éviter les erreurs d'horloge

La page d'onglet "Temps de cycle" de l'état du module vous donne des renseignements sur les temps de cycle du programme utilisateur.

Lorsque la durée du cycle le plus long est proche du temps de surveillance, il peut arriver que des fluctuations dans le temps de cycle entraînent une erreur de temps. Vous pouvez éviter cela en augmentant le temps de cycle maximal du programme utilisateur.

Si la durée de cycle est inférieure au temps de cycle minimal paramétré, la CPU ou le FM l'allongent automatiquement au temps de cycle minimal paramétré. Dans le cas d'une CPU, l'OB d'arrière-plan (OB90) est exécuté durant cette phase, s'il est chargé.

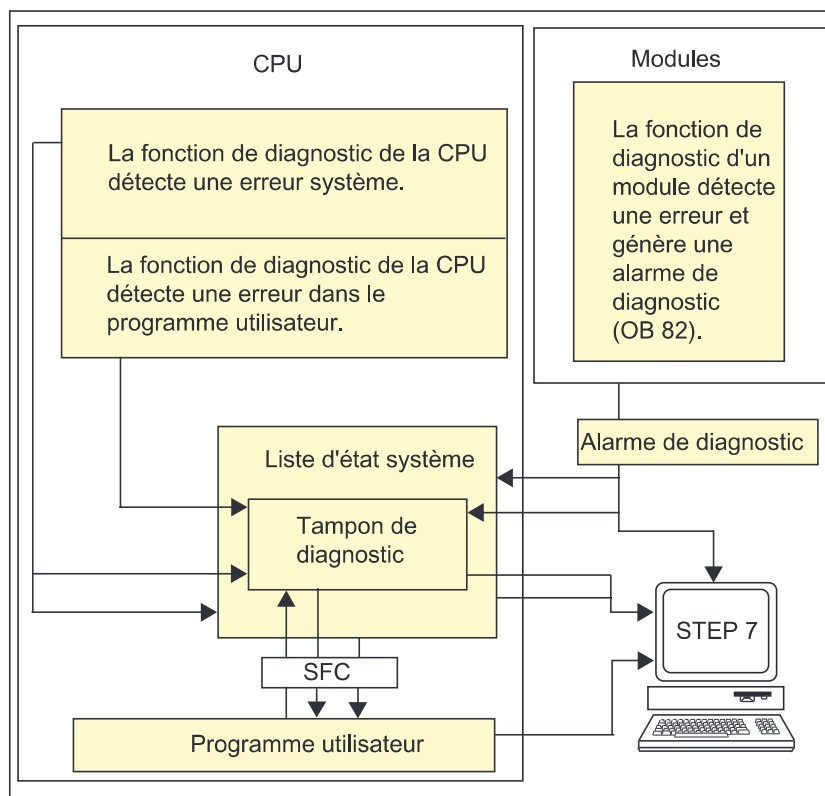
### Définition du temps de cycle

Vous pouvez définir les temps de cycle minimal et maximal lors de la configuration du matériel. Sélectionnez pour ce faire dans la vue hors ligne de la table de configuration la CPU ou le FM et choisissez dans le menu contextuel la commande **Propriétés de l'objet**, afin de définir ses propriétés. Vous pouvez entrer vos valeurs dans la page d'onglet "Cycle/Mémento de cadence".



## 23.8 Transmission d'informations de diagnostic

La figure suivante montre comment les informations de diagnostic sont transmises dans SIMATIC S7.



### Lecture des informations de diagnostic

Vous pouvez lire les entrées de diagnostic dans le programme utilisateur à l'aide de la SFC51 RDSYSST ou afficher les messages de diagnostic en clair avec STEP 7.

Ces informations précisent :

- où et quand l'erreur est apparue,
- à quel type d'événements de diagnostic appartient l'entrée (événement de diagnostic personnalisé, erreur synchrone ou asynchrone, changement d'état de fonctionnement).

### Création de messages groupés système

La CPU inscrit dans la mémoire tampon de diagnostic les événements du diagnostic standard et du diagnostic étendu. Elle génère en outre un message groupé système pour les événements de diagnostic standard si les conditions suivantes sont satisfaites :

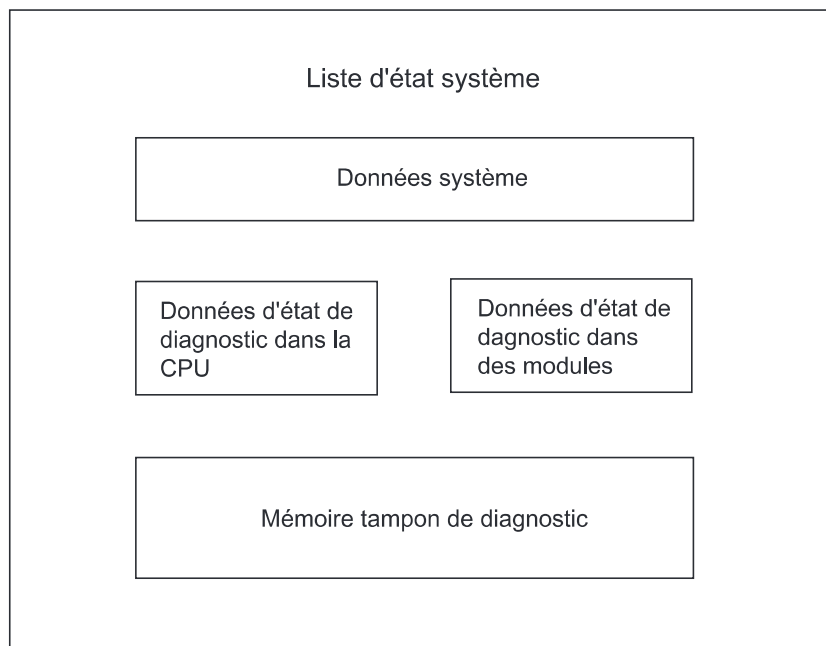
- Vous avez indiqué via STEP 7 que des messages groupés système doivent être générés.
- Un appareil de visualisation au moins s'est déclaré auprès de la CPU pour messages groupés système.
- Un message groupé système n'est créé que lorsqu'il n'existe pas encore de tel message de la classe correspondante (il y a sept classes).
- Il est possible de générer un message groupé système par classe.

### 23.8.1 Liste d'état système (SZL)

La liste d'état système (SZL) décrit l'état en cours de l'automate programmable : elle donne une vue d'ensemble de la configuration, du paramétrage en vigueur, des états et exécutions en cours dans la CPU et les modules associés.

Vous pouvez seulement lire les données de la liste d'état système, et non les modifier. Il s'agit en fait d'une liste virtuelle, générée uniquement sur demande.

On peut subdiviser les informations contenues dans la liste d'état système en quatre domaines.



#### Lecture de la SZL

Il existe deux méthodes pour lire les informations de la liste d'état système :

- implicitement à partir de la console de programmation via des commandes de STEP 7 (par exemple, étendue de la mémoire, données de CPU statiques, mémoire tampon de diagnostic, indications d'état),
- explicitement à partir du programme utilisateur via la fonction système SFC 51 RDSYSST par indication du numéro de liste partielle souhaitée (voir aide sur les blocs).

## Données système de la liste SZL

Les données système sont des caractéristiques fixes ou paramétrées d'une CPU. Le tableau suivant montre pour quels thèmes il est possible d'obtenir des informations (listes partielles de la SZL).

Domaine	Informations
Identificateur de module	Numéro de référence, identification de type et version du module
Caractéristiques de la CPU	Système d'horodatage, comportement du système (par exemple, fonctionnement multiprocesseur) et description de langage de la CPU
Zones de mémoire	Etendue de mémoire du module (taille de la mémoire de travail)
Zones système	Mémoire système du module (par exemple, nombre de mementos, temporisations et compteurs, type de mémoire)
Types de blocs	Types de blocs (OB, DB, SDB, FC, FB) disponibles dans le module, nombre maximal des blocs d'un type et taille maximale d'un type de bloc
Affectation alarmes/erreurs	Affectation d'alarmes/erreurs aux OB
Etat d'alarme	Traitement et génération d'alarmes en cours
Etat des classes de priorité	OB en cours de traitement, classe de priorité verrouillée par paramétrage
Etat de fonctionnement et changement d'état de fonctionnement	Etats de fonctionnement possibles, dernier changement d'état de fonctionnement, état de fonctionnement en vigueur

## Données d'état de diagnostic dans la CPU

Les données d'état de diagnostic décrivent l'état en vigueur des composants surveillés par le diagnostic système. Le tableau suivant montre pour quels thèmes il est possible d'obtenir des informations (listes partielles de la SZL).

Domaine	Informations
Données d'état de la communication	Fonctions de communication actuellement activées dans le système
Partenaires de diagnostic	Modules aptes au diagnostic déclarés à la CPU
Liste d'informations de déclenchement de l'OB	Informations de déclenchement pour les OB de la CPU
Liste d'événements de déclenchement	Evénements de déclenchement et classes de priorité des OB
Informations d'état des modules	Informations d'état de tous les modules affectés, générant des alarmes de processus, défectueux et enfichés

## Données de diagnostic des modules

Il existe, outre la CPU, d'autres modules aptes au diagnostic (SM, CP, FM) dont les données de diagnostic sont inscrites dans la liste d'état système. Le tableau suivant montre pour quels thèmes il est possible d'obtenir des informations (listes partielles de la SZL).

Domaine	Informations
Informations de diagnostic de module	Adresse de début de module, erreurs internes/externes, erreurs de voie, erreurs de paramètres (4 octets)
Données de diagnostic de module	Toutes les données de diagnostic d'un module précis

## 23.8.2 Envoi de vos propres messages de diagnostic

Vous pouvez, en outre, étendre le diagnostic système standard de SIMATIC S7 à l'aide de la fonction système SFC52 WR\_USMSG :

- en inscrivant vos propres informations de diagnostic (par exemple, informations sur l'exécution du programme utilisateur) dans la mémoire tampon de diagnostic,
- en envoyant des messages de diagnostic que vous avez définis à des correspondants déclarés (appareils de contrôle comme PG, OP, TD).

### Événements de diagnostic personnalisé

Les événements de diagnostic sont répartis en classes d'événement 1 à F. Ceux que vous définissez vous-même appartiennent aux classes d'événement 8 à B. On peut les subdiviser en deux groupes :

- Les classes d'événement 8 et 9 comprennent les événements avec un numéro défini et un texte préparé que vous pouvez appeler via le numéro.
- Les classes d'événement A et B regroupent les événements avec numéro (A000 à A0FF, B000 à B0FF) et texte libres.

### Envoi de messages de diagnostic à des correspondants

En plus d'inscrire un événement de diagnostic personnalisé dans la mémoire tampon de diagnostic, vous pouvez, à l'aide de la SFC52 WR\_USMSG, envoyer le message correspondant à des appareils de visualisation déclarés. A l'appel de la SFC52 avec SEND = 1, le message de diagnostic est écrit dans la mémoire tampon d'émission et automatiquement envoyé aux correspondants déclarés à la CPU.

Si l'envoi du message de diagnostic s'avère impossible - par exemple parce qu'un correspondant n'a pas été déclaré ou que la mémoire tampon d'émission est pleine -, l'événement de diagnostic personnalisé est quand même inscrit dans la mémoire tampon de diagnostic.

### Création de messages avec indication d'acquittement

Procédez comme suit si vous voulez acquitter un événement de diagnostic personnalisé et enregistrer cet acquittement par programme :

- Écrivez un 1 dans une variable de type BOOL pour un événement entrant ; écrivez 0 pour un événement sortant.
- Surveillez cette variable à l'aide du bloc SFB33 ALARM.

### 23.8.3 Fonctions de diagnostic

Le diagnostic système détecte, évalue et signale les erreurs survenant au sein d'un automate programmable. Chaque CPU et chaque module possédant la fonction de diagnostic système (par exemple FM354) disposent à cet effet d'une mémoire tampon de diagnostic dans laquelle sont inscrites des informations plus précises sur tous les événements de diagnostic dans l'ordre de leur apparition.

#### Evénements de diagnostic

Les événements suivants provoquent des entrées dans la mémoire tampon de diagnostic, par exemple :

- les erreurs internes et externes sur un module,
- les erreurs système dans la CPU,
- les changements d'état de fonctionnement (par exemple de "Marche" à "Arrêt")
- les erreurs dans le programme utilisateur,
- le débrochage/enfichage de modules.
- les messages personnalisés saisis via la fonction système SFC52

Le contenu de la mémoire tampon de diagnostic est conservé lors de l'effacement général des CPU. Grâce à la mémoire tampon de diagnostic, les erreurs dans le système peuvent être évaluées, même bien plus tard, en vue de déterminer l'origine d'un passage à l'"Arrêt" ou de remonter la trace des événements de diagnostic individuels.

#### Enregistrement des données de diagnostic

Il est inutile de programmer l'enregistrement de données de diagnostic par le diagnostic système, car elle se fait automatiquement. SIMATIC S7 propose différentes fonctions de diagnostic. Certaines d'entre elles sont intégrées dans la CPU et d'autres sont mises à votre disposition par les modules (SM, CP et FM).

#### Affichage d'erreurs

Les erreurs internes et externes aux modules sont signalées par des diodes électroluminescentes en face avant du module concerné. Les signalisations par DEL et leur évaluation sont décrites dans les manuels sur le matériel S7. Dans S7-300, les erreurs internes et externes forment des erreurs groupées.

La CPU détecte les erreurs système ainsi que les erreurs dans le programme utilisateur et inscrit les messages de diagnostic dans la liste d'état système et dans la mémoire tampon de diagnostic. Il est possible de lire ces messages à la console de programmation.

Les modules de fonction et de signaux aptes au diagnostic détectent des erreurs de module internes et externes et génèrent une alarme de diagnostic à laquelle vous pouvez réagir à l'aide d'un OB d'alarme.

## 23.9 Mesures à prendre dans le programme pour traiter les erreurs

Lorsque la CPU détecte des erreurs dans l'exécution du programme (erreurs synchrones) ou des erreurs dans l'automate programmable (erreurs asynchrones), elle appelle l'OB d'erreur correspondant à l'erreur respective :

Erreur survenue	OB d'erreur
Erreur de redondance de périphérie	OB 70
Erreur de redondance de CPU	OB 72
Erreur de temps	OB 80
Erreur d'alimentation	OB 81
Alarme de diagnostic	OB 82
Alarme de débrogage/enfichage	OB 83
Erreur matérielle CPU	OB 84
Erreur d'exécution du programme	OB 85
Défaillance d'un châssis ou d'une station en périphérie décentralisée	OB 86
Erreur de communication	OB 87
Erreur de programmation	OB 121
Erreur d'accès à la périphérie	OB 122

En absence de l'OB correspondant, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) (exceptions : OB 70, OB 72, OB 81, OB 87). Sinon, vous avez la possibilité de saisir des instructions dans l'OB, sur la manière de réagir à cette situation d'erreur. Il est ainsi possible de minimiser ou supprimer les éventuelles conséquences de l'erreur.

### Procédez de la manière suivante

#### Création et ouverture de l'OB

1. Appelez l'état de fonctionnement de votre CPU.
2. Sélectionnez l'onglet "Performances".
3. Vérifiez dans la liste affichée, si l'OB à programmer est autorisé pour cette CPU.
4. Insérez l'OB dans le dossier "Blocs" de votre programme et ouvrez-le.
5. Saisissez le programme de traitement de l'erreur.
6. Chargez l'OB dans le système cible.

#### Programmation des mesures de traitement d'erreur

1. Exploitation des données locales de l'OB pour une détermination plus précise de la cause d'erreur.  
Les variables OB8x\_FLT\_ID ou OB12x\_SW\_FLT des données locales contiennent le code d'erreur. Leur signification est décrite dans le manuel de référence des fonctions système et des fonctions standard.
2. Aller dans la section de programme qui réagit à cette erreur.

Un exemple de traitement d'alarmes de diagnostic est fourni dans l'aide de référence des fonctions système et fonctions standard sous le titre "Exemple de diagnostic de module avec le bloc SFC51 (RDSYSST)".

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

### 23.9.1 Exploitation du paramètre de sortie RET\_VAL

Une fonction système signale par le paramètre de sortie RET\_VAL (valeur en retour) si la CPU a pu l'exécuter avec ou sans erreur.

#### Informations d'erreur dans la valeur en retour

La valeur en retour est de type de données INT (nombre entier), le signe précisant s'il s'agit d'un entier positif ou négatif. La relation de la valeur en retour à la valeur 0 indique si une erreur s'est produite pendant le traitement de la fonction (voir aussi tableau 11-5).

- Si une erreur apparaît pendant le traitement de la fonction, la valeur en retour est inférieure à 0. Le bit de signe du nombre entier est à 1.
- Si la fonction est traitée sans erreur, la valeur en retour est supérieure ou égale à 0. Le bit de signe du nombre entier est à 0.

Traitement de la SFC par la CPU	Valeur en retour	Signe du nombre entier
Avec erreur	Inférieure à 0	Négatif (bit de signe à 1)
Sans erreur	Supérieure ou égale à 0	Positif (bit de signe à 0)

#### Réaction aux informations d'erreur

Si une erreur apparaît pendant le traitement d'une fonction système, la SFC renvoie un code d'erreur par l'intermédiaire de la valeur en retour RET\_VAL.

Ce faisant, on distingue :

- un code d'erreur général que toutes les SFC peuvent émettre et
- un code d'erreur spécifique qu'une SFC peut émettre selon ses fonctions spécifiques.

#### Transmission de la valeur de la fonction

Certaines fonctions système utilisent également le paramètre de sortie RET\_VAL pour renvoyer leur résultat. La SFC64 TIME\_TCK, par exemple, renvoie l'heure système lue via RET\_VAL.

Vous trouverez des informations détaillées sur le paramètre RET\_VAL dans l'aide sur les SFB/SFC.

## 23.9.2 OB d'erreur en réaction à la détection d'une erreur

### Erreurs détectables

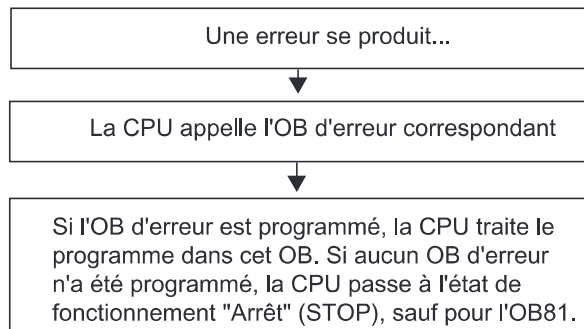
Le programme système peut détecter les erreurs suivantes :

- fonctionnement erroné de la CPU,
- erreurs dans le traitement du programme système,
- erreurs dans le programme utilisateur,
- erreurs dans la périphérie.

Selon le type d'erreur, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) ou un OB d'erreur est appelé.

### Programmation de réactions

Vous pouvez concevoir des programmes pour réagir aux différents types d'erreur et déterminer le comportement de la CPU. Vous pouvez ensuite sauvegarder le programme pour une erreur donnée dans un OB d'erreur. Ce programme sera donc traité à l'appel de cet OB d'erreur.





## OB d'erreur

On distingue entre erreurs synchrones et asynchrones.

- Les erreurs synchrones peuvent être associées à une commande MC7 (par exemple, commande de chargement pour un module de signaux retiré).
- Les erreurs asynchrones peuvent être attribuées à une classe de priorité ou à l'automate programmable entier (par exemple, dépassement du temps de cycle).

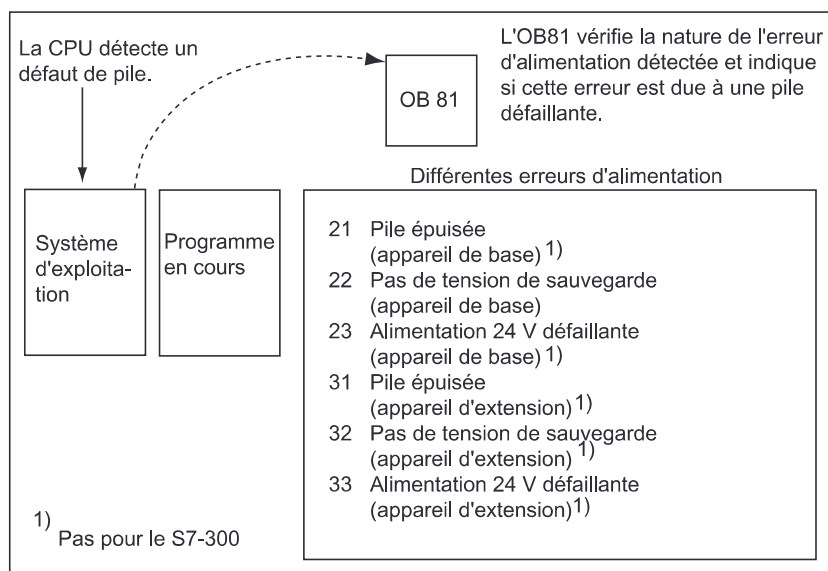
Le tableau ci-après présente les types d'erreur pouvant en principe apparaître. Les OB disponibles pour les différentes CPU sont indiqués dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" ou dans le manuel de référence "Systèmes d'automatisation S7-400/M7-400, Installation et configuration - Caractéristiques des modules".

Catégorie d'erreur	Type d'erreur	OB	Priorité
Redondance	Erreur de redondance de périphérie (uniquement dans les CPU H)	OB 70	25
	Erreur de redondance de CPU (uniquement dans les CPU H)	OB 72	28
Asynchrone	Erreur de temps	OB 80	26
	Erreur d'alimentation	OB 81	(ou 28 lorsque l'OB d'erreur est appelé dans le programme de mise en route)
	Alarme de diagnostic	OB 82	
	Alarme de débrogage/enfichage	OB 83	
	Erreur matérielle CPU	OB 84	
	Erreur d'exécution du programme	OB 85	
	Défaillance d'unité	OB 86	
Erreur de communication	OB 87		
Synchrone	Erreur de programmation	OB 121	Priorité de l'OB à l'origine de l'erreur
	Erreur d'accès	OB 122	

### Exemple d'utilisation de l'OB d'erreur 81

Les données locales (informations de déclenchement) de l'OB d'erreur vous permettent d'évaluer la nature de l'erreur apparue.

Si la CPU détecte un défaut de pile, par exemple, le système d'exploitation appelle l'OB81 (voir figure).



Vous pouvez écrire un programme qui évalue l'ID de l'événement ayant déclenché l'appel de l'OB81. Vous pouvez également écrire un programme de réaction comme, par exemple, l'activation d'une sortie reliée à une lampe du poste d'opération.

## Données locales de l'OB d'erreur 81

Le tableau suivant décrit les variables temporaires (TEMP) inscrites dans la table de déclaration des variables de l'OB81.

Il faut également que la table des mnémoniques identifie le mnémonique *Defaut\_pile* (BOOL) comme étant une sortie (par exemple A 4.0) afin que d'autres parties du programme puissent accéder à ces données.

Décl.	Nom	Type	Description
TEMP	OB81_EV_CLASS	BYTE	Classe et code d'événement 39xx
TEMP	OB81_FLT_ID	BYTE	Code d'erreur : B#16#21 = Au moins une des piles de sauvegarde de l'appareil de base est épuisée. <sup>1</sup> B#16#22 = La tension de sauvegarde manque dans l'appareil de base. B#16#23 = L'alimentation 24 V est défailante dans l'appareil de base. <sup>1</sup> B#16#31 = Au moins une des piles de sauvegarde d'un appareil d'extension est épuisée. <sup>1</sup> B#16#32 = La tension de sauvegarde manque dans un appareil d'extension. <sup>1</sup> B#16#33 = L'alimentation 24 V est défailante dans un appareil d'extension. <sup>1</sup>
TEMP	OB81_PRIORITY	BYTE	Classe de priorité = 26/28
TEMP	OB81_OB_NUMBR	BYTE	81 = OB81
TEMP	OB81_RESERVED_1	BYTE	Réservé
TEMP	OB81_RESERVED_2	BYTE	Réservé
TEMP	OB81_MDL_ADDR	INT	Réservé
TEMP	OB81_RESERVED_3	BYTE	Significatif uniquement pour les codes d'erreur B#16#31, B#16#32 et B#16#33
TEMP	OB81_RESERVED_4	BYTE	
TEMP	OB81_RESERVED_5	BYTE	
TEMP	OB81_RESERVED_6	BYTE	
TEMP	OB81_DATE_TIME	DATE_AND_TIME	Date et heure de déclenchement de l'OB
<sup>1</sup> Pas pour le S7-300			

## Exemple de programme pour l'OB d'erreur 81

L'exemple de programme LIST ci-dessous montre comment lire le code d'erreur dans l'OB81.

Ce programme est organisé comme suit.

- Le code d'erreur figurant dans l'OB81 (OB81\_FLT\_ID) est lu et comparé à l'ID de l'événement "Pile épuisée" (B#16#3921).
- Si ce code d'erreur correspond à l'ID pour "Pile épuisée", le programme saute au repère "DP" et active la sortie *Defaut\_pile*.
- Si ce code d'erreur est différent de l'ID pour "Pile épuisée", le programme le compare à l'ID pour "Pas de tension de sauvegarde".
- Si le code d'erreur correspond à l'ID pour "Pas de tension de sauvegarde", le programme saute au repère "DP" et active la sortie *Defaut\_pile*. Sinon, le bloc s'achève.

LIST		Description
L	B#16#21	// Comparer l'ID de l'événement // "Pile épuisée" (B#16#21)
L	#OB81_FLT_ID	// au code d'erreur de l'OB81.
==I		// Si identiques (pile vide), // sauter à DP.
SPB DP		
L	B#16#22	// Comparer l'ID de l'événement // "Pas de tension de sauvegarde" (B#16#22)
==I		// au code d'erreur de l'OB81.
SPB DP		// Si identiques, sauter à DP.
BEA		// Pas de message sur défaut de pile
DP: L	B#16#39	// Comparer le code pour événement apparaissant
L	#OB81_EV_CLASS	// au code d'erreur de l'OB81.
==I		// Si défaillance de tension de sauvegarde // ou de pile détectée,
S	Defaut_pile	// mettre à 1 Defaut_pile // (variable de la table des mnémoniques).
L	B#16#38	// Comparer le code pour événement // disparaissant
==I		// au code d'erreur de l'OB81.
R	Defaut_pile	// Remettre à 0 Defaut_pile quand éliminée.

Vous trouverez des informations détaillées sur les OB, SFB et SFC ainsi que l'explication des ID d'événement dans les aides sur les blocs correspondantes.

### 23.9.3 Insertion de valeurs de remplacement en cas d'erreur détectée

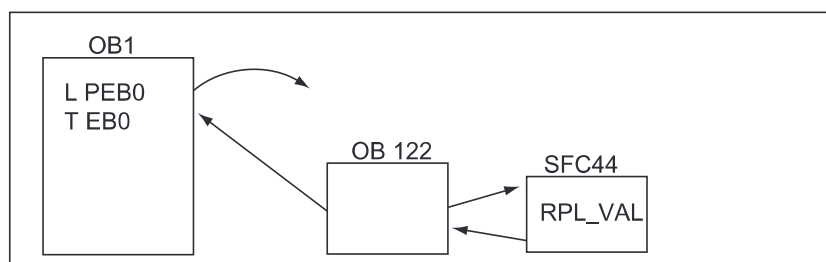
Pour certains types d'erreur (par exemple, rupture de fil pour signal d'entrée), vous pouvez indiquer des valeurs de remplacement pour celles qui s'avèrent indisponibles en raison de l'erreur. Il y a deux façons d'indiquer des valeurs de remplacement.

- Vous pouvez paramétrer avec STEP 7 des valeurs de remplacement pour les modules de sorties paramétrables. Les modules de sorties non paramétrables ont la valeur de remplacement prédéfinie 0.
- Vous pouvez programmer des valeurs de remplacement dans les OB d'erreur à l'aide de la SFC44 RPL\_VAL (uniquement pour les modules d'entrées).

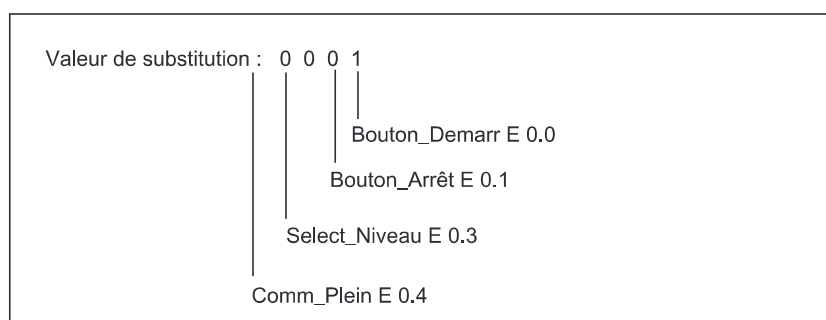
Vous pouvez indiquer, pour toutes les commandes de chargement entraînant des erreurs synchrones, une valeur de remplacement pour le contenu de l'accumulateur dans l'OB d'erreur.

#### Exemple de programme de remplacement de valeur

Dans l'exemple de programme suivant, une valeur de remplacement est mise à disposition dans la SFC44 RPL\_VAL. La figure ci-après montre comment appeler l'OB122 si la CPU s'aperçoit qu'un module d'entrées ne réagit pas.



Dans cet exemple, la valeur de remplacement de la figure ci-après est introduite dans le programme afin que l'exécution de ce dernier se poursuive avec des valeurs cohérentes.



En cas de défaillance d'un module d'entrées, le traitement de l'instruction L PEB0 génère une erreur synchrone et déclenche l'OB122. L'instruction de chargement lit par défaut la valeur 0. Vous pouvez toutefois, à l'aide de la SFC44, définir des valeurs de remplacement adaptées au processus. La SFC remplace le contenu de l'accumulateur par la valeur de remplacement précisée.

L'OB122 pourrait contenir l'exemple de programme suivant. Le tableau suivant présente les variables temporaires supplémentaires à inscrire dans la table de déclaration des variables de l'OB122.

Décl.	Nom	Type	Description
TEMP	OB122_EV_CLASS	BYTE	Classe et code d'événement 29xx
TEMP	OB122_SW_FLT	BYTE	Code d'erreur : 16#42, 16#43, 16#44 <sup>1</sup> , 16#45 <sup>1</sup>
TEMP	OB122_PRIORITY	BYTE	Classe de priorité = priorité de l'OB où s'est produite l'erreur
TEMP	OB122_OB_NUMBR	BYTE	122 = OB122
TEMP	OB122_BLK_TYPE	BYTE	Type du bloc où s'est produite l'erreur
TEMP	OB122_MEM_AREA	BYTE	Zone de mémoire et type d'accès
TEMP	OB122_MEM_ADDR	WORD	Adresse de mémoire où s'est produite l'erreur
TEMP	OB122_BLK_NUM	WORD	Numéro du bloc où s'est produite l'erreur
TEMP	OB122_PRG_ADDR	WORD	Adresse relative de la commande à l'origine de l'erreur
TEMP	OB122_DATE_TIME	DATE_AND_TIME	Date et heure de déclenchement de l'OB
TEMP	Erreur	INT	Contient le code d'erreur provenant de la SFC44.

<sup>1</sup> Pas pour le S7-300

LIST	Description
L B#16#2942	Comparer le code d'événement de l'OB122 au code d'événement (B#16#2942) pour acquittement d'une erreur de temps lors de la lecture de la périphérie. Si identiques, sauter à "ErrA".
L #OB122_SW_FLT	
==I	
SPB QFeh	
L B#16#2943	Comparer le code d'événement de l'OB122 au code d'événement (B#16#2943) pour erreur d'adressage (écriture d'un module inexistant). Si différents, sauter à "Stop".
<> I	
SPB Stop	
ErrA: CALL "REPL_VAL"	Repère "ErrA" : Transmettre DW#16#2912 (10010 binaire) à la SFC44 (REPL_VAL). La SFC44 charge cette valeur dans l'ACCU1, remplaçant ainsi la valeur qui a causé l'appel de l'OB122. Sauvegarder le code d'erreur SFC dans #Erreur.
VAL := DW#16#2912	
RET_VAL := #Erreur	
L #Erreur	
L 0	Comparer #Erreur à 0 (si égal, pas d'erreur lors du traitement de l'OB122). Fin du bloc si pas d'erreur.
==I	
BEB	Repère "Stop" : Appeler la SFC46 STP pour mettre la CPU à l'état de fonctionnement "Arrêt".
Stop: CALL "STP"	

## 23.9.4 Erreur de redondance de périphérie (OB70)

### Description

Le système d'exploitation de la CPU H appelle l'OB70 lorsqu'une perte de redondance se produit sur le réseau PROFIBUS DP (par exemple en cas de défaillance du bus sur le maître DP actif ou en cas d'erreur de couplage de l'esclave DP) ou en cas de changement du maître DP actif pour des esclaves DP avec périphérie couplée

### Programmation de l'OB70

Vous devez créer l'OB70 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB70 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

L'OB70 peut, par exemple, vous servir à :

- exploiter ses informations de déclenchement pour constater quel événement a déclenché la perte de redondance de la périphérie.
- déterminer l'état de votre système à l'aide de la SFC51 RDSYSST (SZL\_ID=B#16#71).

La CPU ne passe pas à l'état d'arrêt lorsqu'une erreur de redondance de périphérie survient et que l'OB70 n'est pas programmé.

Si l'OB70 est chargé et si le système H se trouve en fonctionnement redondant, l'OB70 est traité dans les deux CPU. Le système H reste en fonctionnement redondant.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 23.9.5 Erreur de redondance de CPU (OB72)

### Description

Le système d'exploitation de la CPU H appelle l'OB72 à l'apparition d'un des événements suivants :

- perte de redondance des CPU,
- erreur de comparaison (p. ex. RAM, MIS),
- bascule réserve-maître,
- erreur de synchronisation,
- erreur dans un module SYNC,
- interruption de la procédure d'horodatage.
- L'OB72 est exécuté par toutes les CPU qui se trouvent à l'état de marche ou de mise en route après un événement de déclenchement correspondant.

### Programmation de l'OB72

Vous devez créer l'OB72 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB72 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

L'OB72 peut, par exemple, vous servir :

- à exploiter ses informations de déclenchement pour déterminer l'événement ayant déclenché la perte de redondance de la CPU,
- à déterminer l'état de votre système à l'aide de la SFC51 RDSYSST (SZL\_ID=B#16#71),
- à réagir à la perte de redondance de la CPU en fonction de votre installation.

La CPU ne passe pas à l'état d'arrêt lorsqu'une erreur de redondance de CPU survient et que l'OB72 n'est pas programmé.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.



## 23.9.6 Erreur de temps (OB80)

### Description

Le système d'exploitation de la CPU appelle l'OB80 lorsqu'apparaît une erreur de temps. Les erreurs de temps sont par exemple :

- le dépassement du temps de cycle maximal,
- le saut d'alarmes horaires parce que l'heure a été avancée,
- un retard excessif pour le traitement d'une classe de priorité.

### Programmation de l'OB80

Vous devez créer l'OB80 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB80 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

L'OB80 peut, par exemple, vous servir :

- à exploiter ses informations de déclenchement pour constater quelles alarmes horaires ont été sautées ;
- à désactiver l'alarme horaire sautée à l'aide de la SFC29 CAN\_TINT afin qu'elle ne soit pas exécutée et obtenir une situation nette pour le traitement des alarmes horaires avec la nouvelle heure réglée.

Si vous ne désactivez pas les alarmes horaires sautées dans l'OB80, la première alarme sautée est traitée et il n'est pas tenu compte de toutes les autres.

Si l'OB80 n'est pas programmé, la CPU se met à l'état de fonctionnement "Arrêt" (STOP) à la détection d'une erreur de temps.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 23.9.7 Erreur d'alimentation (OB81)

### Description

Le système d'exploitation de la CPU appelle l'OB81 lors d'une défaillance dans l'appareil de base ou dans un appareil d'extension :

- de la tension d'alimentation 24 V,
- d'une pile,
- du système de sauvegarde entier,

ou bien lorsqu'il a été remédié à cette défaillance (appel pour événement entrant et sortant).

### Programmation de l'OB81

Vous devez créer l'OB81 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB81 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

L'OB81 peut, par exemple, vous servir :

- après exploitation de ses informations de déclenchement, à constater quelle erreur d'alimentation est apparue ;
- à connaître le numéro du profilé support ou du châssis à l'alimentation défaillante ;
- à commander une lampe sur un poste d'opération afin de signaler au personnel de maintenance qu'une pile doit être remplacée.

La CPU ne se met pas à l'état "Arrêt" (STOP) si l'OB81 n'est pas programmé et qu'une erreur d'alimentation est détectée. L'erreur est cependant inscrite dans la mémoire tampon de diagnostic et la DEL correspondante en face avant signale cette erreur.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 23.9.8 Alarme de diagnostic (OB82)

### Description

Le système d'exploitation de la CPU appelle l'OB82 lorsqu'un module avec fonction de diagnostic pour lequel vous avez validé l'alarme de diagnostic détecte une erreur et lorsqu'il a été remédié à cette erreur (appel pour événement entrant et sortant).

### Programmation de l'OB82

Vous devez créer l'OB82 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB82 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB82 pour :

- exploiter ses informations de déclenchement,
- effectuer un diagnostic précis de l'erreur apparue.

Lorsqu'une alarme de diagnostic est déclenchée, le module défectueux inscrit automatiquement 4 octets de données de diagnostic et son adresse de début dans les informations de déclenchement de l'OB d'alarme de diagnostic et dans la mémoire tampon de diagnostic. Vous apprenez ainsi sur quel module et à quel moment s'est produite l'erreur.

Vous pouvez exploiter d'autres données de diagnostic du module défectueux (voie où s'est produite l'erreur, erreur dont il s'agit) à l'aide d'un programme correspondant dans l'OB82. La SFC51 RDSYSST permet de lire les données de diagnostic du module et la SFC52WR\_USRMSG d'inscrire ces informations dans la mémoire tampon de diagnostic. Vous pouvez, en outre, envoyer à un appareil de contrôle déclaré un message de diagnostic que vous définissez vous-même.

Si vous n'avez pas programmé l'OB82, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) au déclenchement d'une alarme de diagnostic.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 23.9.9 Alarme de débrogage/enfichage (OB83)

### Description

Les CPU S7-400 contrôlent cycliquement (environ toutes les secondes) l'enfichage et le débrogage de modules dans l'appareil de base ou les appareils d'extension.

Après la mise sous tension secteur, la CPU vérifie si tous les modules énumérés dans la table de configuration créée avec STEP 7 sont effectivement en place. Si c'est le cas, cette configuration réelle est sauvegardée et sert de référence pour le contrôle cyclique des modules. A chaque cycle de contrôle, la nouvelle configuration réelle constatée est comparée à la configuration réelle valable jusqu'alors. En cas de différences, une alarme de débrogage/enfichage est signalée avec entrée dans la mémoire tampon de diagnostic et la liste d'état système. L'OBde débrogage/enfichage est déclenché à l'état de fonctionnement "Marche" (RUN).

---

### Nota

Il est interdit de retirer modules d'alimentation, CPU et IM à l'état de fonctionnement "Marche" (RUN).

Deux secondes au moins doivent s'écouler entre le retrait et l'enfichage d'un module pour que ce retrait ou cet enfichage soit correctement détecté par la CPU.

---

### Paramétrage d'un module nouvellement enfiché

Si un module est enfiché à l'état "Marche" (RUN), la CPU vérifie si le type du nouveau module enfiché correspond à celui du module précédent. Le paramétrage se fait si les types de modules correspondent : les paramètres par défaut ou ceux que vous avez attribués via STEP 7 sont transmis au module.

### Programmation de l'OB83

Vous devez créer l'OB83 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB83 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB83 pour :

- exploiter ses informations de déclenchement,
- reparamétrer le nouveau module enfiché à l'aide des fonctions système SFC55 à SFC59.

Si l'OB83 n'est pas programmé, la CPU passe de l'état de fonctionnement "Marche" (RUN) à l'état "Arrêt" (STOP) à l'apparition d'une alarme de débrogage/enfichage.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 23.9.10 Erreur matérielle CPU (OB84)

### Description

Le système d'exploitation de la CPU appelle l'OB84 lorsqu'une erreur est détectée pour l'interface au réseau MPI, au bus de communication ou au coupleur pour la périphérie décentralisée : p. ex. : lorsqu'un niveau de signal erroné est détecté sur la voie ou lorsqu'il a été remédié à cette erreur (appel pour événement entrant et sortant).

### Programmation de l'OB84

Vous devez créer l'OB84 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB84 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB84 pour :

- exploiter ses informations de déclenchement,
- envoyer un message à la mémoire tampon de diagnostic à l'aide de la fonction système SFC52 WR\_USMSG.

Si l'OB84 n'est pas programmé, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) lors de la détection d'une erreur matérielle CPU.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

### 23.9.11 Erreur d'exécution du programme (OB85)

#### Description

Le système d'exploitation de la CPU appelle l'OB85

- lorsqu'il existe un événement de déclenchement pour un OB d'alarme, mais que l'OB ne peut pas être exécuté car il n'a pas été chargé dans la CPU ;
- lorsqu'une erreur est apparue lors de l'accès au bloc de données d'instance d'un bloc fonctionnel système ;
- lorsqu'une erreur est apparue lors de la mise à jour de la mémoire image du processus (module absent ou défaillant).

#### Programmation de l'OB85

Vous devez créer l'OB85 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB85 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB85 pour :

- exploiter ses informations de déclenchement et découvrir quel module est défaillant ou manque (indication de l'adresse de début du module),
- calculer l'emplacement du module concerné à l'aide de la fonction système SFC49 LGC\_GADR.

Si l'OB85 n'est pas programmé, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) à la détection d'une erreur de classe de priorité.

## 23.9.12 Défaillance d'unité (OB86)

### Description

Le système d'exploitation de la CPU appelle l'OB86 lorsqu'il a détecté l'un des événements suivants :

- défaillance d'un appareil d'extension centralisé (pas pour S7-300), p. ex. câble de liaison interrompu, coupure de tension décentralisée d'un profilé support ou châssis.
- défaillance d'un réseau maître ou d'un esclave (PROFIBUS DP) ou défaillance d'un réseau IO ou d'un IO Device (PROFINET IO).

L'OB86 est également appelé lorsqu'il a été remédié à cette erreur (appel pour événement entrant et sortant).

### Programmation de l'OB86

Vous devez créer l'OB86 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB86 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB86 pour :

- exploiter ses informations de déclenchement et découvrir quelle unité est défectueuse ou manque,
- inscrire un message dans la mémoire tampon de diagnostic et envoyer ce message à un appareil de contrôle à l'aide de la fonction système SFC52 WR\_USMSG.

Si l'OB86 n'est pas programmé, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) à la détection d'une défaillance d'unité.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

### 23.9.13 Erreur de communication (OB87)

#### Description

Le système d'exploitation de la CPU appelle l'OB87 lorsqu'une erreur de communication apparaît lors de l'échange de données via des blocs fonctionnels de communication ou la communication par données globales, par exemple :

- un identificateur de télégramme erroné a été détecté lors de la réception de données globales,
- le bloc de données pour les informations d'état des données globales manque ou est trop court.

#### Programmation de l'OB87

Vous devez créer l'OB87 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB87 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB87 pour :

- exploiter ses informations de déclenchement,
- créer un bloc de données si le bloc de données pour les informations d'état de la communication par données globales manque.

La CPU ne passe pas à l'état de fonctionnement "Arrêt" (STOP) lorsqu'une erreur de communication est détectée et l'OB87 n'est pas programmé.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.



## 23.9.14 Erreur de programmation (OB121)

### Description

Le système d'exploitation de la CPU appelle l'OB121 lorsqu'une erreur de programmation apparaît, par exemple :

- des temporisations adressées manquent,
- un bloc appelé n'est pas chargé.

### Programmation de l'OB121

Vous devez créer l'OB121 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB121 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB121 pour :

- exploiter ses informations de déclenchement,
- inscrire la cause de l'erreur dans un bloc de données de signalisation.

Si l'OB121 n'est pas programmé, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) à la détection d'une erreur de programmation.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

### 23.9.15 Erreur d'accès à la périphérie (OB122)

#### Description

Le système d'exploitation de la CPU appelle l'OB122 lorsqu'une opération STEP 7 accède à une entrée ou une sortie d'un module de signaux à laquelle aucun module n'était associé lors du dernier démarrage, par exemple :

- erreur en cas d'accès direct à la périphérie (module défaillant ou manquant),
- accès à une adresse de périphérie inconnue de la CPU.

#### Programmation de l'OB122

Vous devez créer l'OB122 avec STEP 7 comme objet dans votre programme S7. Ecrivez le programme devant être traité dans l'OB122 dans le bloc créé et chargez-le dans la CPU en tant que partie de votre programme utilisateur.

Vous pouvez, par exemple, vous servir de l'OB122 pour :

- exploiter ses informations de déclenchement,
- appeler la fonction système SFC44 et indiquer une valeur de remplacement pour un module d'entrées afin que le programme puisse se poursuivre avec une valeur dépendante du processus cohérente.

Si l'OB122 n'est pas programmé, la CPU passe à l'état de fonctionnement "Arrêt" (STOP) à la détection d'une erreur d'accès à la périphérie.

Des informations détaillées sur les OB, SFB et SFC sont données dans les aides sur les blocs correspondantes.

## 24 Impression et archivage

### 24.1 Impression de la documentation du projet

Une fois créé le programme pour votre solution d'automatisation, vous pouvez imprimer toutes les données importantes de votre projet afin de le documenter, en utilisant les fonctions d'impression intégrées à STEP 7.

#### Éléments constituant du projet imprimables

Vous pouvez imprimer un objet soit directement depuis SIMATIC Manager, soit en ouvrant l'objet voulu puis en démarrant l'impression.

Depuis SIMATIC Manager, vous pouvez directement imprimer les éléments constituant suivants d'un projet :

- arborescence des objets (structure du projet/de la bibliothèque)
- liste d'objets (contenu d'un dossier d'objets)
- contenu d'un objet
- messages

En ouvrant l'objet correspondant, vous pouvez imprimer par exemple les éléments constituant suivants du projet :

- blocs, dans les modes de représentation CONT, LOG, LIST ou dans un autre langage (logiciel optionnel),
- table des mnémoniques (avec les mnémoniques d'adresses absolues),
- table de configuration avec la disposition des modules dans l'automate et les paramètres des modules,
- contenu de la mémoire tampon de diagnostic,
- table des variables avec formats de valeur d'état et valeurs d'état et de forçage,
- données de référence, c'est-à-dire listes des références croisées, tableaux d'affectation, structures de programme, opérandes libres, mnémoniques manquants,
- table des données globales,
- caractéristiques du module avec état du module,
- textes destinés à l'utilisateur (textes utilisateur et bibliothèques de textes),
- documents de logiciels optionnels, par exemple de langages de programmation.

#### Logiciel optionnel DOCPRO

Pour créer, éditer et imprimer des dossiers normalisés de vos schémas d'installation, vous pouvez utiliser le logiciel optionnel DOCPRO. Vous obtenez ainsi une documentation de votre installation répondant aux normes DIN et ANSI.

### 24.1.1 Marche à suivre pour l'impression

Pour l'impression, procédez de la manière suivante :

1. Ouvrez l'objet approprié pour afficher à l'écran les informations à imprimer.
2. Appelez la boîte de dialogue d'impression à l'aide de la commande **Fichier > Imprimer...** dans la fenêtre en question. Selon la fenêtre, le premier menu dans la barre des menus peut différer de "Fichier". Il peut par exemple s'agir du menu "Table".
3. Modifiez, si nécessaire, les paramètres d'impression dans la boîte de dialogue (par exemple, l'imprimante, l'étendue, le nombre d'exemplaires), puis fermez celle-ci.

Certaines boîtes de dialogue, comme par exemple celle de l'état du module contiennent le bouton "Imprimer". Cliquez sur ce bouton pour imprimer le contenu de la boîte de dialogue.

Il n'est pas nécessaire d'ouvrir les blocs. Vous pouvez les imprimer directement dans SIMATIC Manager via la commande **Fichier > Imprimer....**

### 24.1.2 Fonctions d'impression

Pour imprimer les objets, vous disposez des fonctions complémentaires suivantes :

Objet à imprimer	Commande	Fonction	Fonction	Fonction
		Aperçu avant impression	Mise en page, onglet "Format papier"	Mise en page, onglet "En-tête et bas de page"
Blocs, sources LIST	Fichier > *	•	•	•
Etat du module		–	•	•
Table de données globales	Table GD > *	•	•	•
Table de configuration	Station > *	•	•	•
Objet, dossier Objets	Fichier > *	–	•	•
Données de référence	Données de référence > *	•	•	•
Table des mnémoniques	Table > *	•	•	•
Table des variables	Table > *	–	•	•
Table des liaisons	Réseau > *	•	•	•
Textes destinés à l'utilisateur (textes personnalisés, bibliothèques de textes)	Textes > *	•	•	•
* : Le caractère * désigne la fonction correspondante dans la commande (p. ex. Aperçu avant impression ou Mise en page)				

La marche à suivre pour imprimer les objets individuels est donnée dans :  
Impression.

## Aperçu avant impression

Grâce à la fonction "Aperçu avant impression", vous pouvez examiner le document avant de l'imprimer.

---

### Nota

L'aperçu avant impression n'affiche pas la présentation de l'impression optimisée.

---

## Définition de la mise en page ainsi que des en-têtes et bas de page

En choisissant la commande de menu **Fichier > Mise en page**, puis l'onglet "Format du papier", vous pouvez définir la mise en page (p. ex. A4, A5, Lettre) et l'orientation (portrait ou paysage) pour le document à imprimer. Vous pouvez également préciser si ces paramètres s'appliquent à l'ensemble du projet ou uniquement à la session actuelle.

Adaptez le formulaire utilisé pour l'impression au format de papier désiré. Si le formulaire est trop large, la partie droite sera imprimée sur une page suivante.

Si vous choisissez une taille de papier avec marge (par exemple : "A4 marge"), le document imprimé aura une marge gauche qui peut servir à le perforer.

Dans l'onglet "En-têtes et bas de page", vous pouvez définir les en-têtes et bas de page des documents à imprimer, pour l'ensemble du projet ou pour la session actuelle.

### 24.1.3 Particularités pour l'impression de l'arborescence des objets

En activant la case d'option "Arborescence" dans la boîte de dialogue "Imprimer la liste d'objets", vous pouvez non seulement imprimer la liste des objets, mais également l'arborescence des objets.

Si vous activez la case d'option "Tous" sous "Etendue", l'arborescence complète s'imprimera. En activant le bouton "Sélection", l'arborescence s'imprimera vers le bas, à partir de l'objet sélectionné.

---

### Nota

Les paramétrages effectués dans la boîte de dialogue ne valent que pour l'impression de la liste ou de l'arborescence, mais pas pour celle des contenus. En effet, pour l'impression des contenus, ce sont les paramétrages des applications concernées qui sont utilisés.

---

## 24.2 Archivage de projets et de bibliothèques

Vous pouvez mémoriser un projet ou une bibliothèque particuliers sous forme comprimée dans un fichier d'archivage. Celui-ci pourra ensuite être copié sur le disque dur ou sur un support d'enregistrement amovible (une disquette, par exemple).

### Programmes d'archivage

Dans STEP 7, vous pouvez appeler votre programme d'archivage préféré pour archiver des projets. Les programmes d'archivage ARJ et PKZIP 4.0 sont livrés avec STEP 7. Vous trouverez les programmes et les descriptions correspondantes dans le répertoire d'installation...\Step7\S7bin\.

Vous pouvez utiliser les programmes d'archivage suivants à condition d'en posséder la bonne version :

- PKZip Commandline V4.0 (fourni)
- WinZip à partir de la version 6.0
- JAR à partir de la version 1.02
- ARJ V2.4.1a (uniquement pour le désarchivage, est fourni)
- ARJ32 V3.x (uniquement pour le désarchivage)
- LHArc à partir de la version 2.13 (uniquement pour le désarchivage)

### Particularités

A partir de STEP 7 V5.2, l'archivage peut uniquement être réalisé avec les programmes d'archivage PKZip 4.0, JAR et WinZip, le désarchivage pouvant également être réalisé avec les autres programmes précités.

Si sous des versions précédentes de STEP 7, l'archivage a été réalisé avec le programme ARJ32 V3.x, le désarchivage de ces archives peut uniquement être réalisé avec ARJ32 V3.x.

L'archivage avec PKZIP V4.0 est beaucoup plus long sur des lecteurs réseaux que sur des lecteurs locaux.

## 24.2.1 Possibilités d'enregistrement / archivage

### Enregistrer sous

Cette fonction permet de créer une **copie** du projet sous un autre nom.

Vous pouvez l'utiliser pour

- créer des copies de sauvegarde
- copier un projet afin de le modifier pour répondre à d'autres besoins.

Si vous voulez que la copie soit rapide, choisissez l'option de sauvegarde sans réorganisation dans la boîte de dialogue. L'arborescence entière du projet sera alors copiée et sauvegardée sous un autre nom sans vérification.

Vérifiez que la mémoire disponible sur le support de données est suffisante pour la copie. N'essayez pas de sauvegarder des projets sur disquettes, car la place mémoire en général ne suffit pas. Pour le transport de vos données de projet sur disquettes, choisissez la fonction "Archiver".

La sauvegarde d'un projet avec réorganisation dure plus longtemps. Un message est toutefois affiché quand un objet ne peut être copié et sauvegardé. La raison peut en être un logiciel optionnel manquant ou des données défectueuses.

### Archiver

Vous pouvez mémoriser un projet ou une bibliothèque particuliers sous forme comprimée dans un fichier d'archivage. Celui-ci pourra ensuite être copié sur le disque dur ou sur un support d'enregistrement amovible (une disquette, par exemple).

Ne transportez vos projets sur disquettes qu'archivés dans des fichiers d'archives. Si le projet est trop grand, sélectionnez un programme d'archivage autorisant l'archivage sur plusieurs disquettes.

Les projets et bibliothèques qui ont été comprimés en fichiers d'archivage ne peuvent pas être utilisés tels quels. Si vous souhaitez les réutiliser, vous devez préalablement décompresser les données, c'est-à-dire désarchiver le projet ou la bibliothèque.

## 24.2.2 Conditions requises pour l'archivage

Pour archiver un projet/une bibliothèque, les conditions suivantes doivent être remplies :

- Le programme d'archivage doit être installé dans votre système. L'intégration à STEP 7 est expliquée dans l'aide en ligne, à la rubrique "Marche à suivre pour l'archivage/le désarchivage."
- Toutes les données du projet sans exception doivent se trouver dans ou sous le répertoire de projet. Il est certes possible, dans l'environnement de développement C, de stocker des données à d'autres endroits, mais ces données ne seront alors pas enregistrées dans le fichier d'archives.
- A partir de STEP 7 V5.2, l'archivage peut uniquement encore être réalisé avec les programmes d'archivage PKZip 4.0, JAR, WinZip, le désarchivage pouvant encore être réalisé avec les programmes ARJ et LHArc.

## 24.2.3 Marche à suivre pour l'archivage/le désarchivage

Vous archivez/désarchivez votre projet/bibliothèque en choisissant l'une des commandes **Fichier > Archiver** ou **Fichier > Désarchiver**.

---

### Nota

Les projets et bibliothèques qui ont été comprimés en fichiers d'archivage ne peuvent pas être utilisés tels quels. Si vous souhaitez les réutiliser, vous devez préalablement décompresser les données, c'est-à-dire désarchiver le projet ou la bibliothèque.

---

Lors du désarchivage, les projets/bibliothèques sont intégrés automatiquement à la liste des projets/bibliothèques.

## Définition du répertoire cible

Pour définir le répertoire cible, vous choisissez la commande **Outils > Paramètres** dans SIMATIC Manager pour ouvrir la boîte de dialogue "Paramètres".

L'onglet "Archiver" de cette boîte de dialogue propose l'option "Répertoire cible désarchivage".

Lorsque cette option est désactivée, le répertoire cible utilisé sera celui défini dans la page d'onglet "Général" sous "Lieu d'archivage des projets" ou "Lieu d'archivage des bibliothèques" de la même boîte de dialogue.

## Copie d'un fichier d'archivage sur disquette

Vous pouvez archiver un projet ou une bibliothèque puis copier le fichier d'archives sur une disquette. Il est également possible de sélectionner directement un lecteur de disquettes dans la boîte de dialogue d'archivage.



## 25 Utilisation des systèmes d'automatisation M7

### 25.1 Marche à suivre pour les systèmes M7

Grâce à son architecture PC standardisée, le système d'automatisation M7-300/400 constitue une extension librement programmable de la plateforme d'automatisation SIMATIC. Vous pouvez programmer les programmes utilisateur pour SIMATIC M7 dans un langage évolué comme C ou dans un langage graphique comme CFC (Continuous Function Chart).

Pour créer ces programmes, vous avez besoin - en plus de STEP 7 - du logiciel système M7-SYS RT pour M7-300/400 ainsi que d'un environnement de développement pour programmes M7 (ProC/C++ ou CFC).

#### Marche à suivre

La réalisation d'une solution d'automatisation avec SIMATIC M7 appelle les tâches fondamentales suivantes. Le tableau suivant indique les tâches à réaliser dans la plupart des projets et indique la marche à suivre sous forme de guide. Il fait référence aux chapitres correspondants du présent manuel ou d'autres manuels.

Marche à suivre	Description
Conception d'une solution d'automatisation	Spécifique à M7 ; voir PHB M7-SYS RT
Démarrage de STEP 7	Comme pour STEP 7
Création de la structure du projet Création d'une station Configuration du matériel	Comme pour STEP 7
Configuration de liaisons de communication	Comme pour STEP 7
Définition de la table des mnémoniques	Comme pour STEP 7
Création d'un programme utilisateur C ou CFC	Spécifique à M7 ; voir ProC/C++
Configuration du système d'exploitation Installation du système d'exploitation sur le M7-300/400 Chargement de la configuration matérielle et du programme utilisateur dans M7	Spécifique à M7 ; voir BHB M7-SYS Rt
Test et débogage du programme utilisateur	ProC/C++
Surveillance du fonctionnement et diagnostic de M7	Comme pour STEP 7, cependant sans diagnostic personnalisé
Impression et Archivage	Comme pour STEP 7

### Quelles sont les différences ?

Pour M7-300/400, STEP 7 ne propose pas les fonctions suivantes :

- Mode multiprocesseur - fonctionnement synchrone de plusieurs CPU
- Forçage permanent de variables
- Communication par données globales
- Diagnostic personnalisé

### Gestion des systèmes M7

STEP 7 vous assiste particulièrement pour la résolution des tâches suivantes avec les systèmes d'automatisation M7 :

- Installation du système d'exploitation sur le M7-300/400
- Configuration du système d'exploitation par édition de fichiers système
- Transfert de programmes utilisateur dans le M7-300/400
- Actualisation du micro-programme

Pour parvenir dans la gestion du système cible M7, vous appelez la commande suivante dans le contexte d'un projet contenant des stations avec des CPU ou des FM S7, le dossier des programmes étant sélectionné :

#### **Système cible > Gérer le système cible M7**

Des instructions détaillées sont données dans l'aide en ligne et dans le manuel utilisateur M7-SYS RT.

## 25.2 Logiciel optionnel pour la programmation M7

### Logiciel optionnel M7

STEP 7 vous offre les fonctions de base nécessaires pour :

- créer et gérer des projets,
- configurer et paramétrer le matériel,
- configurer des réseaux et des liaisons,
- gérer les mnémoniques (données symboliques).

Ces fonctions sont indépendantes du fait que votre système d'automatisation soit un SIMATIC S7 ou un SIMATIC M7.

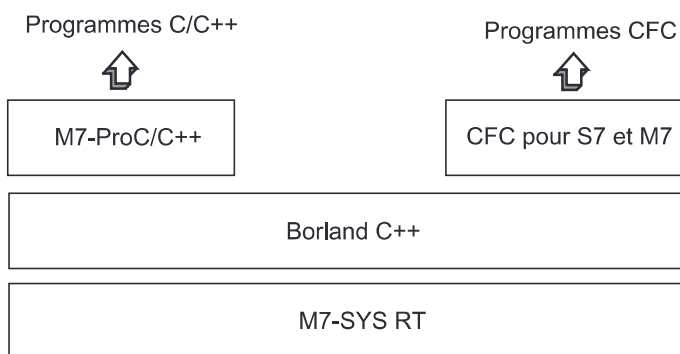
Pour créer des programmes utilisateur M7, vous avez besoin du logiciel optionnel M7 en plus de STEP 7.

Logiciel	Contenu
M7-SYS RT	<ul style="list-style-type: none"> <li>• Système d'exploitation M7 RMOS32</li> <li>• Bibliothèque système M7-API</li> <li>• Prise en charge MPI</li> </ul>
CFC pour S7 et M7	Logiciel de programmation pour applications CFC (Continuous Function Chart)
M7-ProC/C++	<ul style="list-style-type: none"> <li>• Intégration de l'environnement de développement Borland dans STEP 7</li> <li>• Editeur et générateur d'importation de mnémoniques</li> <li>• Débogueur de langages évolués Organon xdb386</li> </ul>
Borland C++	Environnement de développement Borland C++

Avec le logiciel optionnel M7, STEP 7 vous assiste en outre dans les activités suivantes :

- transfert de données dans le M7 via MPI,
- recherche d'informations sur le système d'automatisation M7
- exécution d'opérations précises dans le système d'automatisation M7 et effacement général du M7.

La figure suivante montre les interdépendances au sein du logiciel optionnel M7 pour la programmation M7.



### Récapitulatif

Pour créer des ...	vous avez besoin du logiciel optionnel M7 ...
programmes C/C++	<ul style="list-style-type: none"> <li>• M7-SYS RT</li> <li>• M7-ProC/C++</li> <li>• Borland C++</li> </ul>
programmes CFC	<ul style="list-style-type: none"> <li>• M7-SYS RT</li> <li>• CFC pour S7 et M7</li> <li>• Borland C++</li> </ul>

## Assistance

Les outils spécifiques pour la création d'applications M7 sont intégrés pour partie dans STEP 7 et pour partie dans le logiciel optionnel M7.

Le tableau suivant indique dans quels domaines les différents logiciels vous assistent.

Le logiciel ...	vous aide...
STEP 7	<ul style="list-style-type: none"> <li>• pour l'installation du système d'exploitation M7,</li> <li>• pour la gestion du système d'automatisation M7,</li> <li>• pour le transfert, le lancement et la suppression des programmes utilisateur M7,</li> <li>• pour l'appel de données d'état et de diagnostic,</li> <li>• pour l'effacement général de la CPU.</li> </ul>
M7-SYS RT	via les utilitaires du système d'exploitation M7 et du logiciel système M7 pour : <ul style="list-style-type: none"> <li>• la commande de l'exécution des programmes,</li> <li>• la gestion de la mémoire et des ressources,</li> <li>• l'accès au matériel SIMATIC et ordinateur,</li> <li>• la gestion des alarmes,</li> <li>• le diagnostic,</li> <li>• la surveillance de l'état</li> <li>• et la communication.</li> </ul>
M7-ProC/C++	<ul style="list-style-type: none"> <li>• grâce à la génération de code intégrée (intégration de l'environnement de développement Borland dans STEP7),</li> <li>• grâce à l'intégration des mnémoniques de projet dans le code source,</li> <li>• grâce à la fonction de débogage intégrée.</li> </ul>
Borland C++	<ul style="list-style-type: none"> <li>• pour la création de programmes C et C++.</li> </ul>
CFC pour S7 et M7	<ul style="list-style-type: none"> <li>• pour la création, le test et le débogage de programmes CFC</li> <li>• et pour le lancement et l'exécution de programmes CFC.</li> </ul>

## 25.3 Systèmes d'exploitation pour M7-300/400

Les utilitaires du système d'exploitation sont de première importance pour les applications créées dans les langages évolués C et C++. Le système d'exploitation assure les tâches suivantes pour l'application :

- accès au matériel,
- gestion des ressources,
- intégration dans le système,
- communication avec d'autres composants du système.

Pour la résolution de tâches d'automatisation, nous mettons en oeuvre le système d'exploitation temps réel M7 RMOS32 (RMOS = **R**ealtime-**M**ultitasking-**O**perating-**S**ystem) sur le système d'automatisation SIMATIC M7. Pour permettre son intégration dans le système SIMATIC, M7 RMOS a été complété par une interface d'appel : M7-API (**A**pplication **P**rogramming **I**nterface).

Le système d'exploitation en temps réel RMOS32 de M7 sert à la résolution de problèmes temps réel et multitâches critiques en temps pour des applications de 32 bits. Il est disponible dans les configurations suivantes pour les modules M7 :

- M7 RMOS32
- M7 RMOS32 avec MS-DOS

La configuration du système d'exploitation que vous choisissez pour votre système d'automatisation M7 dépend des modules M7 que vous mettez en oeuvre :

Configurations de système d'exploitation	Module / mémoire centrale	PROFIBUS DP et TCP/IP oui/non	Installation sur mémoire de masse
M7 RMOS32	FM 356-4 / 4Mo	non	carte mémoire ≥4 Mo ou disque dur
	FM 356-4 / 8Mo	oui	
	CPU 388-4 / 8Mo	oui	
	FM 456-4 / 16Mo	oui	
	CPU 488-3 / 16Mo	oui	
	CPU 486-3 / 16Mo	oui	
M7 RMOS32 avec MS-DOS	FM 356-4 / 8Mo	non	carte mémoire ≥4 Mo ou disque dur
	CPU 388-4 / 8Mo	non	
	FM 456-4 / 16Mo	oui	
	CPU 488-3 / 16Mo	oui	
	CPU 486-3 / 16Mo	oui	

## 26 Astuces et conseils

### 26.1 Remplacement de modules dans la table de configuration

Lorsque vous retouchez la configuration d'une station dans HW Config et que vous souhaitez remplacer un module, par exemple par un autre qui a un autre numéro de référence, procédez de la manière suivante :

1. Amenez le module par glisser-lâcher de la fenêtre "Catalogue du matériel" sur le module ("ancien") déjà placé.
2. Lâchez le nouveau module ; dans la mesure du possible, il adopte alors les paramètres du module déjà enfiché.

Cette façon de faire est plus rapide que celle qui consiste à effacer d'abord l'ancien module avant d'insérer le nouveau et de le paramétrer.

Vous pouvez activer ou désactiver cette fonction expressément dans HW Config au moyen de la commande **Outils > Paramètres** ("Remplacement de modules possible").

### 26.2 Projets comportant un grand nombre de stations en réseau

Lorsque vous configurez toutes les stations l'une après l'autre et n'appellez NetPro par **Outils > Configurer le réseau** que dans un deuxième temps pour configurer les liaisons, les stations sont placées automatiquement dans la vue de réseau. L'inconvénient, c'est que vous devez alors classer les stations et les sous-réseaux après coup selon des critères topologiques.

Si votre projet comporte un grand nombre de stations en réseau que vous souhaitez relier par des liaisons, il vaut mieux configurer la structure de l'installation dans la vue de réseau dès le départ, pour conserver une vue d'ensemble.

1. Créez le nouveau projet dans SIMATIC Manager (commande **Fichier > Nouveau**).
2. Démarrez NetPro (commande **Outils > Configurer le réseau**).
3. Générez les stations l'une après l'autre dans NetPro :
  - Prélevez la station dans la fenêtre "Catalogue" et placez-la par glisser-lâcher.
  - Cliquez deux fois sur la station pour démarrer HW Config.
  - Dans HW Config, placez par glisser-lâcher les modules de communication (CPU, CP, FM, cartouches d'interface).
  - Si vous voulez connecter ces modules à un réseau, cliquez deux fois sur les lignes concernées de la table de configuration pour créer des sous-réseaux et connecter les interfaces.
  - Enregistrez la configuration et passez dans NetPro.

- Positionnez stations et sous-réseaux dans NetPro (faites glisser l'objet avec le pointeur de la souris jusqu'à la position qui vous convient).
4. Configurez les liaisons dans NetPro et rectifiez au besoin la mise en réseau.

## 26.3 Réorganisation

Lorsque des problèmes inexplicables surviennent lors de l'utilisation de STEP 7, la solution consiste souvent à réorganiser la base de données du projet ou de la bibliothèque.

Choisissez à cet effet la commande **Fichier > Réorganiser**. La réorganisation permet de supprimer les intervalles consécutifs à l'effacement, c'est-à-dire de réduire l'espace mémoire requis par des données du projet/de la bibliothèque.

Cette fonction optimise la base de données pour le projet ou la bibliothèque tout comme un programme, par exemple, optimise la base de données sur le disque dur pour sa défragmentation.

La durée de la réorganisation dépend des déplacements de données requis et peut occuper un temps relativement long. C'est la raison pour laquelle cette fonction n'est pas exécutée automatiquement (par exemple à la fermeture d'un projet), mais doit être démarrée par l'utilisateur lorsqu'il souhaite réorganiser le projet ou la bibliothèque.

### Condition préalable

Un projet ou une bibliothèque ne peuvent être réorganisés que si aucun des objets qu'ils contiennent n'est utilisé par une autre application est donc inaccessible.

## 26.4 Edition de mnémoniques dans plusieurs réseaux

Dans l'éditeur de programmes CONT/LIST/LOG, vous pouvez afficher et éditer les mnémoniques de plusieurs réseaux.

1. Sélectionnez un réseau en cliquant sur sa désignation (p. ex. "Réseau 1").
2. Maintenez la touche CTRL enfoncée et sélectionnez d'autres réseaux.
3. Avec le bouton droit de la souris, choisissez la commande de menu contextuelle **Editer les mnémoniques**.

Pour sélectionner tous les réseaux d'un bloc, appuyez sur les touches CTRL+A, puis sélectionnez une désignation de réseau.



## 26.5 Test à l'aide de la table des variables

Voici une série de conseils facilitant la visualisation et le forçage de variables dans la table des variables.

- Vous pouvez taper les mnémoniques et les opérands dans la colonne "Mnémonique" comme dans la colonne "Opérande". L'entrée est automatiquement reportée dans la colonne appropriée.
- Pour obtenir l'affichage de la valeur forcée, choisissez comme point de déclenchement "Début de cycle" pour la visualisation et "Fin de cycle" pour le forçage.
- Si vous positionnez le curseur dans une ligne repérée en rouge, une info-bulle vous indique la cause d'erreur. Appuyez alors sur la touche F1 pour afficher la solution.
- Vous ne pouvez entrer que des mnémoniques déjà définis dans la table des mnémoniques.  
Il faut entrer un mnémonique exactement comme il est défini dans la table.  
Les mnémoniques comportant des caractères spéciaux s'écrivent entre guillemets (ex. : "Moteur.stop", "Moteur+stop", "Moteur-stop").
- Suppression des messages d'avertissement (option de l'onglet "Online" de la boîte de dialogue "Paramètres").
- Changement possible de liaison sans avoir coupé la liaison existante préalable.
- Possibilité de définir le déclenchement durant la visualisation des variables.
- Possibilité de forcer des variables par sélection des lignes voulues dans la table et de la fonction "Forcer". Seules les variables visibles dans la table sont alors forcées.
- Quitter sans demande de confirmation

La visualisation, le forçage ou le déblocage des sorties de périphérie étant en cours d'exécution, une pression de la touche ECHAP mettra fin à ces fonctions sans demande de confirmation.

- Saisir une plage d'opérands consécutifs  
Utilisez la commande **Insertion > Plage**.
- Afficher et masquer des colonnes

Utilisez les commandes suivantes pour afficher ou masquer les colonnes de votre choix :

Mnémonique : **Affichage > Mnémonique**

Commentaire de mnémonique : **Affichage > Commentaire de mnémonique**

Format de représentation de la valeur d'état : **Affichage > Format d'affichage**

Valeur d'état de la variable : **Affichage > Valeur d'état**

Valeur de forçage de la variable : **Affichage > Valeur de forçage**

- Modifier le format d'affichage dans plusieurs lignes à la fois
  - Pour sélectionner la partie de la table dans laquelle vous voulez changer de format d'affichage, faites glisser le pointeur dessus en maintenant la touche gauche de la souris enfoncée.
  - Choisissez la nouvelle représentation avec la commande **Affichage > Choisir format d'affichage**. Le format changera seulement pour celles des lignes sélectionnées qui autorisent ce changement.
- Exemples de saisie par la touche F1

- Positionnez le curseur dans la colonne des opérandes et appuyez sur F1 pour obtenir des exemples montrant la saisie d'opérandes.
- Positionnez le curseur dans la colonne des valeurs de forçage et appuyez sur F1 pour obtenir des exemples montrant la saisie de valeurs de forçage ou de forçage permanent.

## 26.6 Forçage de variables depuis l'éditeur de programmes

Dans l'éditeur de programmes, vous pouvez attribuer un bouton aux entrées et aux mémentos binaires. En cliquant sur ce bouton, vous pouvez forcer ces opérandes de manière simple et rapide.

### Conditions

- Dans la table des mnémoniques, vous avez attribué cette propriété à l'opérande que vous souhaitez forcer avec la commande de menu **Propriétés spécifiques de l'objet > Commande par contact**.
- Dans l'onglet "Général" de l'éditeur CONT/LIST/LOG (commande de menu **Outils > Paramètres**), vous avez activé l'option "Commande par contact".
- Vous avez choisi la commande de menu **Test > Visualiser**.

La condition de déclenchement est "permanent/au début du cycle".

Les entrées effectivement présentes dans votre installation sont forcées tant que le bouton reste enfoncé. La sélection multiple (touche CTRL) vous permet de forcer plusieurs opérandes simultanément.

Dans le cas de mémentos ou d'entrées manquantes, l'état est mis à 1 lorsque vous appuyez sur le bouton. Il n'est remis à 0 que si vous le réalisez explicitement dans le menu contextuel ou dans la table des variables ou lorsque l'opérande est remis à 0 par le programme S7.

Le bouton étant enfoncé, le forçage s'effectue avec la valeur de forçage "1" lorsque l'entrée ou le memento ne sont pas inversés et avec la valeur de forçage "0" lorsque l'entrée ou le memento sont inversés.

### Remarque pour WinCC

Si l'éditeur de programmes a été démarré depuis WinCC par le contrôle-commande d'une variable, seul le forçage depuis WinCC est possible, à moins que l'utilisateur dispose des "droits de maintenance" de WinCC, auquel cas les deux possibilités de forçage sont possibles.

## 26.7 Mémoire virtuelle

Une raison de la défaillance de STEP 7 peut être une mémoire virtuelle trop petite.

Pour utiliser STEP 7, il est recommandé d'adapter le paramétrage de la mémoire virtuelle. Procédez de la manière suivante :

1. Ouvrez le panneau de configuration, par exemple depuis le menu de démarrage en choisissant la commande **Démarrer > Paramètres > Panneau de configuration** et effectuez un double clic sur l'icône "Système".  
Sous XP uniquement : choisissez la commande **Démarrer > Poste de travail > Paramètres > Etendu > Performances système > Paramètres**.
2. Sous Windows 2000, sélectionnez l'onglet "Extension" et actionnez le bouton "Options de performance système".  
Sous Windows XP/serveur 2003, sélectionnez l'onglet "Etendu" dans la boîte de dialogue "Paramètres système".
3. Cliquez sur le bouton "Modifier".
4. Entrez comme "Minimum" ou "Taille initiale" (Mo) au moins 40 méga-octets et comme maximum ou "Taille maximale" (Mo) au moins 150 méga-octets.

---

### Nota

Puisque la mémoire virtuelle se trouve sur le disque dur (par défaut C: et de manière dynamique), vous devez vous assurer que l'espace mémoire disponible pour les répertoires TMP ou TEMP est suffisant (environ 20 à 30 Mo) :

- Si le projet S7 devait se trouver sur la même partition que la mémoire virtuelle, il faudrait que le double environ de l'espace mémoire occupé par le projet S7 soit encore disponible.
  - Toutefois, si le projet est géré sur une autre partition, cette condition s'avère inutile.
-



# A Annexe

## A.1 Etats de fonctionnement

### A.1.1 Etats de fonctionnement et changement d'état de fonctionnement

#### Etats de fonctionnement

Les états de fonctionnement décrivent le comportement de la CPU à un instant quelconque. La connaissance de ces états est utile pour la programmation de la mise en route, le test de l'automate, ainsi que pour le diagnostic des erreurs.

Les CPU S7-300 et S7-400 possèdent les états de fonctionnement suivants :

- Arrêt
- Mise en route
- Marche
- Attente

A l'état de fonctionnement "Arrêt" (STOP), la CPU vérifie si tous les modules configurés ou utilisant l'adressage par défaut sont présents et place la périphérie dans un état fondamental prédéfini. Le programme utilisateur n'est pas traité dans cet état.

On distingue en mode de mise en route, entre les modes "Démarrage à chaud", "Démarrage à froid" et "Redémarrage" :

- En cas de démarrage à chaud, le traitement du programme recommence au début avec une "définition de base" des données système et des zones d'opérandes utilisateur (les temporisations, compteurs et mémentos non rémanents sont remis à zéro).
- En cas de démarrage à froid, la mémoire image est lue et le programme utilisateur STEP 7 est exécuté en commençant par la première instruction dans l'OB1 (ceci est également le cas pour le démarrage à chaud).
  - Les blocs de données créés par SFC dans la mémoire de travail sont effacés, les autres blocs de données prennent la valeur par défaut de la mémoire de chargement.
  - La mémoire image ainsi que tous les compteurs, temporisations et mémentos sont remis à zéro, qu'ils aient été paramétrés comme étant rémanents ou pas.
- En cas de redémarrage, le traitement du programme se poursuit au point d'interruption (les temporisations, compteurs et mémentos ne sont pas remis à zéro). Seules les CPU S7-400 peuvent exécuter un redémarrage.

A l'état de fonctionnement "Marche" (RUN), la CPU traite le programme utilisateur, met à jour les entrées et les sorties, traite les alarmes et messages d'erreur.

Le traitement du programme utilisateur est suspendu à l'état de fonctionnement "Attente" et vous pouvez tester ce programme pas à pas. Cet état n'est accessible que lors du test avec la console de programmation.

Dans tous ces états de fonctionnement, la CPU peut communiquer via l'interface MPI.

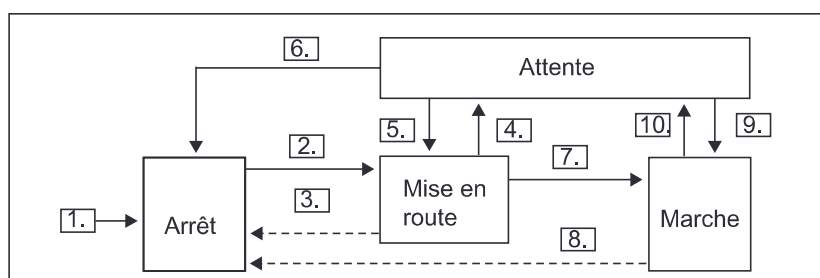
## Autres états de fonctionnement

Lorsque la CPU n'est pas prête à fonctionner, elle se trouve dans l'un des états suivants :

- Sans tension : il n'y a pas de tension secteur.
- Défaillante : une erreur irrémédiable s'est produite.  
Vérifiez si la CPU présente vraiment une défaillance : mettez-la sur STOP et éteignez puis allumez le commutateur secteur. Si la CPU se met en route, lisez la mémoire tampon de diagnostic afin d'analyser l'erreur. Si la CPU ne démarre pas, il faut la remplacer.

## Changements d'état de fonctionnement

La figure suivante représente les états de fonctionnement et les changements d'état de fonctionnement des CPU S7-300 et S7-400 :



Le tableau suivant donne les conditions de changement d'état de fonctionnement.

Changement	Description
1.	La CPU est à l'état "Arrêt" (STOP) lors de la mise sous tension.
2.	La CPU passe à l'état "Mise en route" : <ul style="list-style-type: none"> <li>• lorsque la position RUN ou RUN-P est sélectionnée via le commutateur à clé ou la console de programmation ou</li> <li>• après déclenchement automatique d'un mode de mise en route par mise sous tension.</li> <li>• lorsque la fonction de communication "RESUME" ou "START" est exécutée.</li> </ul> Le commutateur de mode de fonctionnement doit se trouver dans ces deux cas sur RUN ou RUN-P.
3.	La CPU passe de nouveau à l'état "Arrêt" (STOP) lorsque : <ul style="list-style-type: none"> <li>• une erreur est détectée pendant la mise en route ;</li> <li>• la CPU est mise sur STOP via le commutateur de mode de fonctionnement ou depuis la PG ;</li> <li>• une commande d'arrêt est traitée dans l'OB de mise en route ;</li> <li>• la fonction de communication "STOP" est exécutée.</li> </ul>
4.	La CPU passe à l'état de fonctionnement "Attente" lorsqu'un point d'arrêt est atteint dans le programme de mise en route.
5.	La CPU passe en mode de mise en route lorsque le point d'arrêt était défini dans un programme de mise en route et que la commande QUITTER ATTENTE est exécutée (fonction de test).
6.	La CPU passe de nouveau à l'état "Arrêt" (STOP) lorsque : <ul style="list-style-type: none"> <li>• la CPU est mise sur STOP via le commutateur de mode de fonctionnement ou depuis la PG ;</li> <li>• la fonction de communication "STOP" est exécutée.</li> </ul>
7.	La CPU passe à l'état "Marche" (RUN) si la mise en route s'achève sans erreur.
8.	La CPU passe de nouveau à l'état "Arrêt" (STOP) lorsque : <ul style="list-style-type: none"> <li>• une erreur est détectée à l'état "Marche" et que l'OB correspondant n'est pas chargé ;</li> <li>• la CPU est mise sur STOP via le commutateur de mode de fonctionnement ou depuis la PG ;</li> <li>• une commande d'arrêt est traitée dans le programme utilisateur ;</li> <li>• la fonction de communication "STOP" est exécutée.</li> </ul>
9.	La CPU passe à l'état de fonctionnement "Attente" lorsqu'un point d'arrêt est atteint dans le programme utilisateur.
10.	La CPU passe à l'état de fonctionnement "Marche" lorsqu'un point d'arrêt était défini et que la commande QUITTER ATTENTE est exécutée.

## Priorité des états de fonctionnement

Lorsque plusieurs états de fonctionnement sont demandés en même temps, la CPU passe à l'état de fonctionnement ayant la priorité la plus élevée. Si le commutateur de mode se trouve, par exemple, sur RUN et que vous tentez de commuter la CPU à l'état "Arrêt" (STOP) depuis la PG, la CPU passe bien à l'état "Arrêt", car cet état a la priorité la plus élevée.

Priorité	Etat de fonctionnement
La plus élevée	Arrêt
	Attente
	Mise en route
La plus faible	Marche

### A.1.2 Etat de fonctionnement "Arrêt" (STOP)

Le programme utilisateur n'est pas traité dans cet état. Toutes les sorties sont mises à des valeurs de substitution afin d'amener le processus commandé à un état sûr. La CPU vérifie

- s'il y a des problèmes de matériel (par exemple, modules non disponibles) ;
- si la CPU doit prendre les valeurs par défaut ou s'il existe des jeux de paramètres ;
- si les conditions annexes pour le comportement de mise en route programmé sont correctes ;
- s'il y a des problèmes de logiciel système.

Il est possible, à l'état "Arrêt", de recevoir des données globales et d'exécuter une communication à sens unique passive via des SFB de communication pour liaisons configurées et via des SFC de communication pour liaisons non configurées.

### Effacement général

Vous pouvez effectuer un effacement général de la CPU à l'état "Arrêt", soit manuellement en positionnant le commutateur à clé sur MRES, soit à partir de la console de programmation (par exemple, avant le chargement d'un programme utilisateur).

L'effacement général remet la CPU dans son "état fondamental", ce qui signifie que :\_

- Le programme utilisateur complet dans la mémoire de travail et dans la mémoire de chargement RAM ainsi que toutes les zones d'opérandes sont effacés.
- Les paramètres système ainsi que les paramètres des modules et de la CPU reprennent leur valeur par défaut. Seuls les paramètres MPI définis avant l'effacement général sont conservés.
- Lorsqu'une carte mémoire est enfichée (EPROM flash), la CPU copie le programme utilisateur de la carte mémoire dans la mémoire de travail (y compris les paramètres de CPU et de modules si les données de configuration correspondantes se trouvent également sur la carte mémoire).

La mémoire tampon de diagnostic, les paramètres MPI, l'heure et le compteur d'heures de fonctionnement ne sont pas remis à zéro.



### A.1.3 Mode de mise en route

Un programme de mise en route est exécuté avant que la CPU ne commence à traiter le programme utilisateur après la mise sous tension. Dans ce programme de mise en route, vous pouvez définir des présélections précises pour votre programme cyclique en programmant en conséquence les OB de mise en route.

Il existe trois modes de mise en route : démarrage à chaud, démarrage à froid et redémarrage. Le redémarrage n'est possible que pour les CPU S7-400. Il doit avoir été défini avec STEP 7 dans le jeu de paramètres de la CPU.

En mode de mise en route :

- le programme contenu dans l'OB de mise en route (OB100 pour démarrage à chaud, OB101 pour redémarrage et OB102 pour démarrage à froid) est exécuté ;
- aucun traitement de programme déclenché par horloge et par alarme n'est possible ;
- les temporisations sont mises à jour ;
- le compteur d'heures de fonctionnement s'exécute ;
- les sorties TOR des modules de signaux sont verrouillées, mais peuvent être mises à 1 par accès direct.

#### Démarrage à chaud

Un démarrage à chaud est toujours autorisé à moins qu'un effacement général n'ait été demandé par le système. Seul le démarrage à chaud est possible après :

- effacement général,
- chargement du programme utilisateur à l'état de fonctionnement "Arrêt" de la CPU,
- débordement de la pile des interruptions ou de la pile des blocs,
- interruption d'un démarrage à chaud (par mise hors tension ou via le commutateur de mode),
- dépassement de la limite de temps d'interruption paramétrée pour le redémarrage.

#### Démarrage à chaud manuel

Un démarrage à chaud manuel peut être déclenché :

- via le commutateur de mode de fonctionnement,  
(le commutateur CRST/WRST - s'il existe - doit être sur WRST).
- par une commande de menu provenant de la PG ou par des fonctions de communication  
(lorsque le commutateur de mode est en position RUN ou RUN-P).

## Démarrage à chaud automatique

Un démarrage à chaud automatique peut être déclenché à la mise sous tension lorsque :

- la CPU n'était pas à l'arrêt lors de la mise hors tension ;
- le commutateur de mode est en position RUN ou RUN-P ;
- aucun redémarrage automatique après mise sous tension n'est paramétré ;
- la CPU a été interrompue par coupure secteur au démarrage à chaud (indépendamment du paramétrage du mode de mise en route).

La position du commutateur CRST/WRST reste sans effet en cas de démarrage à chaud automatique.

## Démarrage à chaud automatique sans sauvegarde

Si votre CPU fonctionne sans pile de sauvegarde (fonctionnement sans maintenance requis), un effacement général est automatiquement effectué à la mise sous tension ou au retour de la tension après mise hors tension, puis un démarrage à chaud est exécuté. Le programme utilisateur doit être disponible sur carte mémoire (EPROM flash).

## Redémarrage

Après une coupure secteur à l'état de fonctionnement "Marche" (RUN) puis retour de la tension, les CPU S7-400 exécutent un sous-programme d'initialisation puis, automatiquement, un redémarrage. Lors d'un redémarrage, le programme utilisateur se poursuit au point où son traitement a été interrompu. On appelle cycle restant la partie du programme utilisateur qui n'a pas été traitée avant la coupure secteur. Le cycle restant peut contenir des parties de programme déclenchées par horloge ou par alarme.

Un redémarrage n'est par principe possible que si vous n'avez pas modifié le programme utilisateur à l'état "Arrêt" (par exemple, en rechargeant un bloc modifié) ou si un démarrage à chaud n'est pas requis pour d'autres raisons. On distingue entre redémarrage manuel et redémarrage automatique.

## Redémarrage manuel

Un redémarrage manuel n'est possible qu'en cas de paramétrage correspondant dans le jeu de paramètres de la CPU et après un passage à l'arrêt ayant les causes suivantes :

- commutation du commutateur de mode de RUN sur STOP,
- "Arrêt" programmé par l'utilisateur, "Arrêt" après appel d'OB non chargés
- état d'"arrêt" provoqué par la PG ou une fonction de communication.

Un redémarrage manuel peut être déclenché :

- via le commutateur de mode,

Le commutateur CRST/WRST doit être sur WRST.

- par une commande de menu provenant de la PG ou par des fonctions de communication (lorsque le commutateur de mode est en position RUN ou RUN-P).
- lorsque le redémarrage manuel a été paramétré dans le jeu de paramètres de la CPU.

## Redémarrage automatique

Un redémarrage automatique peut être déclenché à la mise sous tension lorsque :

- la CPU n'était pas à l'état d'arrêt ou d'attente lors de la mise hors tension ;
- le commutateur de mode est en position RUN ou RUN-P ;
- le redémarrage automatique après mise sous tension a été paramétré dans le jeu de paramètres de la CPU.

La position du commutateur CRST/WRST reste sans effet en cas de redémarrage automatique.

## Zones de données rémanentes après coupure secteur

Les CPU S7-300 et S7-400 réagissent différemment lors du retour de la tension après coupure secteur.

Les CPU S7-300 (à l'exception de la CPU 318) connaissent uniquement le mode de mise en route "Démarrage". Vous pouvez toutefois, avec STEP 7, définir comme rémanents des mémentos, temporisations, compteurs et zones dans des blocs de données afin d'éviter la perte de données en cas de coupure de courant. Un "démarrage automatique avec rémanence" est alors exécuté au retour de la tension.

Les CPU S7-400 réagissent à un retour de tension selon leur paramétrage par un "Démarrage à chaud" (après une mise sous tension avec ou sans sauvegarde) ou un "Redémarrage" (possible uniquement après une mise sous tension avec sauvegarde).

Le tableau suivant montre le comportement de rémanence des CPU S7-300 et S7-400 en cas de démarrage à chaud, démarrage à froid ou redémarrage.

X	signifie	les données sont conservées
VC	signifie	le bloc de code dans l'EPROM est conservé, le bloc de code éventuellement chargé est perdu
VX	signifie	le bloc de données n'est conservé que s'il est présent dans l'EPROM, les données rémanentes sont reprises de NV-RAM (les blocs de données chargés ou créés dans la RAM sont perdus)
0	signifie	les données sont remises à zéro ou effacées (contenu des DB)
V	signifie	les données prennent la valeur par défaut de l'EPROM
---	signifie	impossible, puisque NV-RAM absente

### Comportement de rémanence dans la mémoire de travail (pour la mémoire de chargement EPROM et RAM)

		EPROM	(carte	mémoire	ou	intégrée)			
		CPU avec	sauveg.			CPU	sans	sauveg.	
Données	blocs ds mém. charg.	DB ds mém. travail	mém. tempos compt.	mém. tempos compt.	blocs ds mém. charg.	DB ds mém. travail	DB ds mém. travail	mém. tempos compt.	mém. tempos compt.
			(param. rémanents)	(param. non rémanents)		(param. rémanents)	(param. non rémanents)	(param. rémanents)	(param. non rémanents)
Démarrag. à chaud pour S7-300	X	X	X	0	VC	VX	V	X	0
Démarrag. à chaud pour S7-400	X	X	X	0	VC	---	V	0	0
Démarr. à froid pour S7-300	X	0	0	0	VC	V	V	0	0
Démarr. à froid pour S7-400	X	0	0	0	VC	---	V	0	0
Redémarrage pour S7-400	X	X	X	X	Seul le démarrage à chaud est autorisé				

## Activités à la mise en route

Le tableau ci-après montre les activités exécutées par la CPU lors de la mise en route.

Activités dans l'ordre de leur traitement	Démarrage à chaud	Démarrage à froid	Redémarrage
Effacer la pile des interruptions et la pile des blocs	X	X	0
Effacer les mémentos, temporisations et compteurs non rémanents	X	0	0
Effacer tous les mémentos, temporisations et compteurs	0	X	0
Effacer la mémoire image des sorties	X	X	paramétrable
Effacer les sorties des modules de signaux	X	X	paramétrable
Rejeter les alarmes de processus	X	X	0
Rejeter les alarmes temporisées	X	X	0
Rejeter les alarmes de diagnostic	X	X	X
Actualiser la liste d'état système (SZL)	X	X	X
Exploiter les paramètres de modules et les transmettre aux modules ou bien leur transmettre les valeurs par défaut	X	X	X
Traiter l'OB de mise en route concerné	X	X	X
Traiter le cycle restant (partie du programme utilisateur n'ayant pu être exécutée en raison d'une mise hors tension)	0	0	X
Actualiser la mémoire image des entrées	X	X	X
Valider les sorties TOR (débloquent les sorties TOR) après passage à l'état de fonctionnement "Marche"	X	X	X
X signifie est exécuté			
0 signifie n'est pas exécuté			

## Interruption de la mise en route

Si des erreurs apparaissent au cours de la mise en route, cette dernière est interrompue et la CPU passe ou reste à l'état "Arrêt".

Un démarrage à chaud interrompu doit être recommencé. Après l'interruption d'un redémarrage, démarrage à chaud ou redémarrage sont tous deux possibles.

Une mise en route (démarrage à chaud ou redémarrage) n'est pas exécutée ou est interrompue :

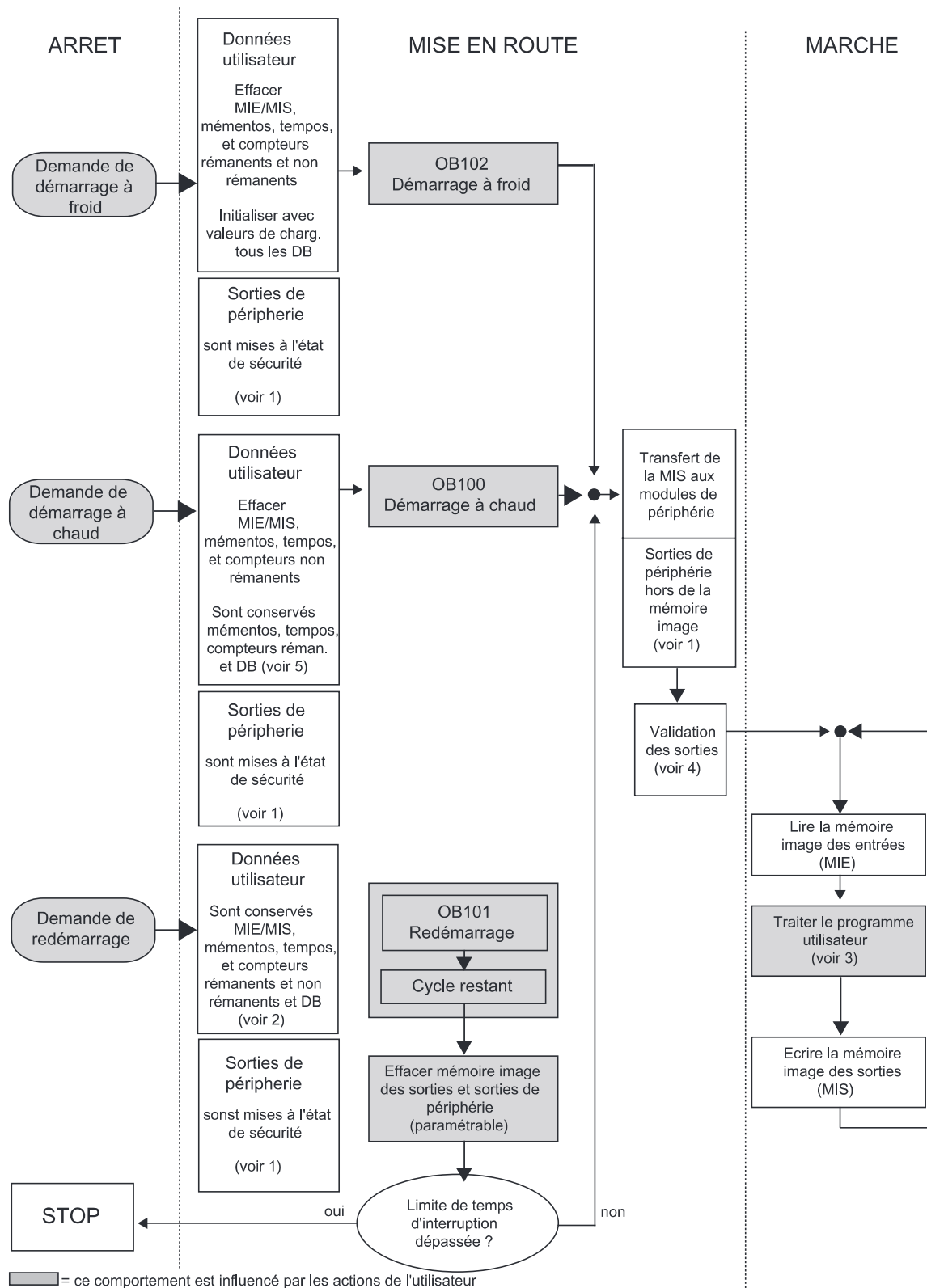
- si le commutateur de mode de fonctionnement de la CPU est en position STOP ;
- si un effacement général est demandé ;
- si une carte mémoire dont l'identificateur d'application n'est pas autorisé pour STEP 7 (par exemple, STEP 5) est enfichée ;
- si plus d'une CPU est enfichée en mode monoprocesseur ;
- si le programme utilisateur contient un OB que la CPU ne connaît pas ou qui a été verrouillé ;
- si la CPU constate, après la mise sous tension, que tous les modules figurant dans la table de configuration créée avec STEP7 ne sont pas enfichés ( entre paramétrage nominal et effectif non autorisée) ;
- si des erreurs apparaissent lors de l'exploitation des paramètres des modules.

Un redémarrage n'est pas non plus exécuté ou est interrompu :

- si la CPU a auparavant subi un effacement général (seul un démarrage à chaud est possible après un effacement général) ;
- si la limite de temps d'interruption a été dépassée (il s'agit du temps qui s'écoule après l'abandon de l'état "Marche" jusqu'au traitement de l'OB de mise en route, cycle restant inclus) ;
- si la configuration des modules a été modifiée (remplacement de modules, par exemple) ;
- si le paramétrage autorise uniquement un démarrage à chaud ;
- si des blocs ont été chargés, effacés ou modifiés à l'état "Arrêt".

## Déroulement

La figure ci-après montre les activités de la CPU dans les états de fonctionnement "Mise en route" et "Marche" (RUN).



Légende de la figure "Activités de la CPU dans les états de fonctionnement Mise en route et Marche"

1. Toutes les sorties de périphérie sont mises à l'état de sécurité (valeur par défaut =0) par le matériel des modules de périphérie, qu'elles soient utilisées dans le programme utilisateur au sein de la zone de la mémoire image du processus ou en dehors.
2. Si vous employez des modules de signaux pouvant traiter une valeur de remplacement, il est possible de paramétrer le comportement des sorties, par exemple Conserver dernière valeur.
3. Est nécessaire à l'exécution du cycle restant.
4. Les OB d'alarme disposent également d'une mémoire image des entrées actuelle lors de leur premier appel.
5. Vous pouvez recourir aux mesures suivantes pour déterminer l'état des sorties de périphérie centralisée et décentralisée dans le premier cycle du programme utilisateur :
  - utiliser des modules de sorties paramétrables pour pouvoir écrire des valeurs de remplacement ou conserver la dernière valeur ;
  - au redémarrage : activer l'option "Remise à 0 des sorties au redémarrage" pour la mise en route de la CPU, afin d'écrire un "0" (qui est la valeur par défaut) ;
  - donner des valeurs par défaut aux sorties dans l'OB de mise en route (OB100, OB101, OB102).
6. Dans les systèmes S7-300 sans sauvegarde, seules les zones DB configurées comme rémanentes sont conservées.



### A.1.4 Etat de fonctionnement "Marche" (RUN)

A l'état "Marche" s'effectue le traitement de programme cyclique et commandé par horloge et par alarme :

- La mémoire image des entrées est lue.
- Le programme utilisateur est traité.
- La mémoire image des sorties est émise.

L'échange actif de données entre les CPU par communication par données globales (table des données globales), par SFB de communication pour les liaisons configurées et par SFC pour les liaisons non configurées n'est possible qu'à l'état de "Marche".

Le tableau ci-après illustre quand l'échange de données est possible dans les différents états de fonctionnement :

Type de communication	Etat de fonctionnement de la CPU 1	Sens de l'échange de données	Etat de fonctionnement de la CPU 2
Communication par données globales	Marche	↔	Marche
	Marche	→	Arrêt/Attente
	Arrêt	←	Marche
	Arrêt	X	Arrêt
	Attente	X	Arrêt/Attente
Communication à sens unique	Marche	→	Marche
Par SFB de communication	Marche	→	Arrêt/Attente
Communication à double sens par SFB de communication	Marche	↔	Marche
Communication à sens unique	Marche	→	Marche
Par SFC de communication	Marche	→	Arrêt/Attente
Communication à double sens par SFC de communication	Marche	↔	Marche
↔	signifie échange de données possible dans les deux sens		
→	signifie échange de données possible dans un sens seulement		
X	signifie échange de données impossible		

### A.1.5 Etat de fonctionnement "Attente"

L'état de fonctionnement "Attente" tient une place à part. Il ne sert qu'à des fins de test à la mise en route ou en marche. Dans l'état de fonctionnement "Attente" :

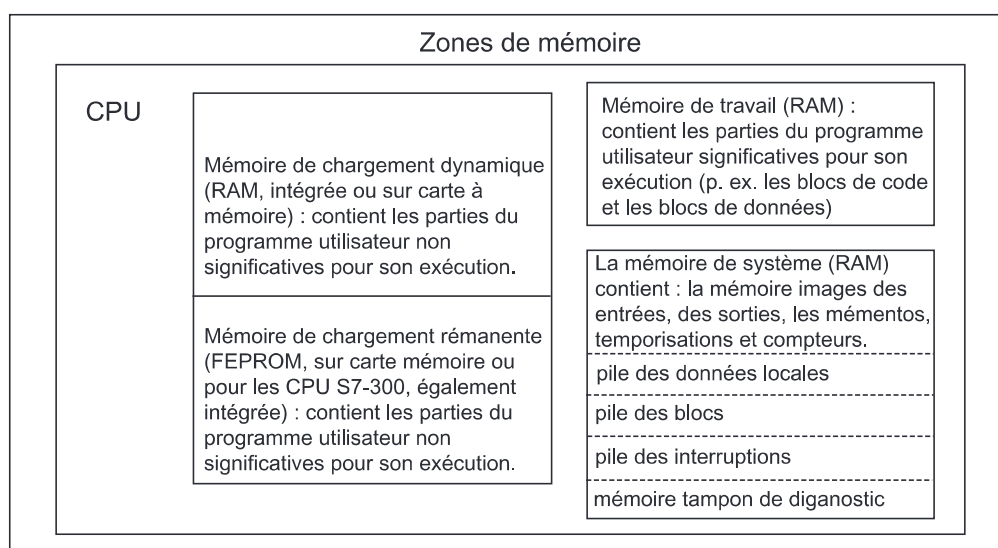
- Tous les temps sont suspendus : les temporisations et les compteurs d'heures de fonctionnement ne sont pas traités. Les temps de surveillance ainsi que les périodes de base des niveaux commandés par horloge sont interrompus.
- L'horloge temps réel fonctionne.
- Les sorties ne sont pas libérées, mais peuvent être validées à des fins de test.
- Il est possible de commander les entrées et les sorties.
- En cas de coupure secteur et de retour de la tension, les CPU avec sauvegarde en "Attente" passent à l'état "Arrêt" et n'exécutent pas de démarrage ni de redémarrage automatique (démarrage à chaud). Au retour de la tension, les CPU sans sauvegarde effectuent un démarrage automatique (démarrage à chaud) sans sauvegarde.
- Il est possible de recevoir des données globales et d'exécuter une communication à sens unique passive par SFB de communication pour les liaisons configurées et par SFC de communication pour les liaisons non configurées (voir aussi tableau à l'état de fonctionnement "Marche").

## A.2 Zones de mémoire des CPU S7

### A.2.1 Organisation des zones de mémoire

La mémoire des CPU S7 comporte trois zones (voir la figure ci-après) :

- La mémoire de chargement sert à l'enregistrement du programme utilisateur sans affectation de mnémoniques ni de commentaires (ces derniers restent dans la mémoire de la console de programmation). La mémoire de chargement peut être soit la mémoire vive (RAM), soit la mémoire EPROM.
- Les blocs identifiés comme non significatifs pour l'exécution sont exclusivement chargés dans la mémoire de chargement.
- La mémoire de travail (mémoire vive intégrée) contient les parties du programme S7 significatives pour l'exécution du programme. Le traitement du programme a lieu exclusivement dans la mémoire de travail et dans la mémoire système.
- La mémoire système (mémoire vive) contient les éléments de mémoire que chaque CPU met à la disposition du programme utilisateur comme, par exemple, mémoire image des entrées, mémoire image des sorties, mémentos, temporisations et compteurs. La mémoire système contient, en outre, la pile des blocs et la pile des interruptions.
- C'est également la mémoire système de la CPU qui fournit la mémoire temporaire (pile des données locales) allouée au programme lors de l'appel d'un bloc pour les données temporaires. Ces données sont valables tant que le bloc est actif.



## A.2.2 Mémoire de chargement et mémoire de travail

Lorsque vous chargez le programme utilisateur de la console de programmation dans la CPU, seuls les blocs de code et les blocs de données sont chargés dans la mémoire de chargement et dans la mémoire de travail de la CPU.

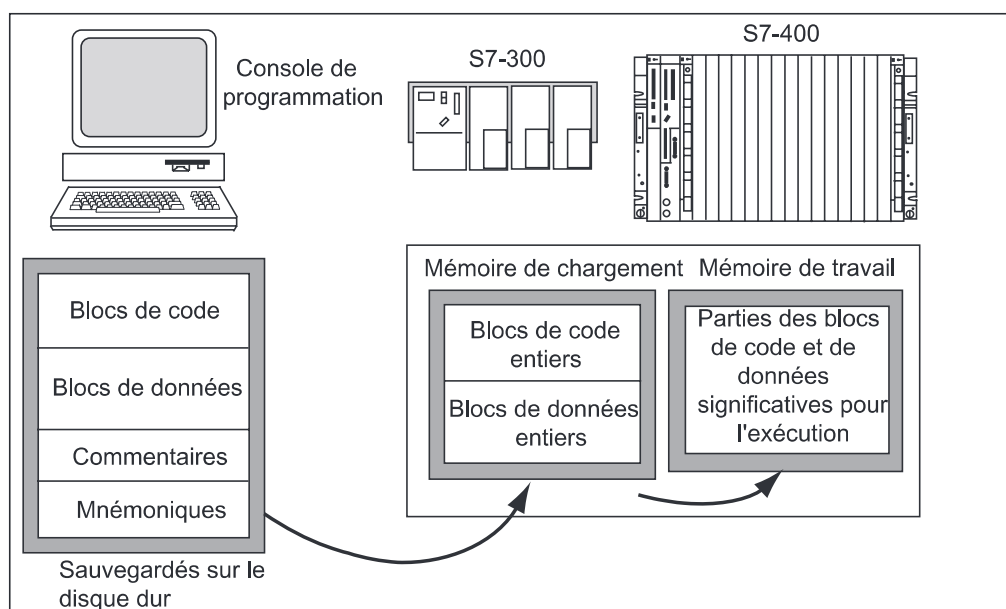
La table des mnémoniques et les commentaires de blocs restent dans la zone de mémoire de la PG.

### Répartition du programme utilisateur

Afin de garantir un traitement rapide du programme utilisateur et de ne pas surcharger inutilement la mémoire de travail non extensible, seules les parties des blocs significatives pour le traitement du programme sont chargées dans la mémoire de travail.

Les parties de blocs non requises pour l'exécution du programme (par exemple, les en-têtes de blocs) restent dans la mémoire de chargement.

La figure suivante représente le chargement du programme dans la mémoire de la CPU.



### Nota

Les blocs de données créés dans le programme utilisateur à l'aide de fonctions système (par exemple SFC 22 CREAT\_DB) sont enregistré complètement dans la mémoire de travail par la CPU.

Certaines CPU disposent de zones gérées séparément pour le code et les données dans la mémoire de travail. Pour ces CPU, la taille et l'occupation de ces zones sont affichées sur la page d'onglet "Mémoire" de l'état du module.

### **Qualification de blocs de données comme "non significatifs pour l'exécution"**

On peut qualifier de "non significatifs pour l'exécution" - mot-clé UNLINKED - les blocs de données programmés comme partie d'un programme LIST dans un fichier source. Lors du chargement dans la CPU, ces DB ne sont donc rangés que dans la mémoire de chargement. Si besoin est, il est possible de copier leur contenu dans la mémoire de travail à l'aide de la SFC20 BLKMOV.

Cela permet donc de gagner de la place dans la mémoire de travail, la mémoire de chargement extensible servant de mémoire intermédiaire (par exemple pour les formules : seule la prochaine formule à traiter est chargée dans la mémoire de travail).

### **Structure de la mémoire de chargement**

Il est possible d'étendre la mémoire de chargement à l'aide de cartes à mémoire. La taille maximale de la mémoire de chargement est donnée dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques de la CPU" et le manuel de référence "Systèmes d'automatisation S7-400/M7-400, Caractéristiques des modules".

Pour les CPU S7-300, la mémoire de chargement peut comporter une partie EPROM intégrée en plus de la partie RAM. Dans les blocs de données, certaines zones peuvent être déclarées comme rémanentes par paramétrage dans STEP 7 (voir Zones de mémoire rémanentes des CPU S7-300).

Pour les CPU S7-400, l'utilisation d'une carte mémoire (RAM ou EPROM) s'avère indispensable pour l'extension de la mémoire de chargement. En effet, la mémoire de chargement intégrée est une mémoire vive qui sert essentiellement au rechargement et à la correction des blocs. Pour les nouvelles CPU S7-400, une mémoire de travail supplémentaire peut également être enfichée.

## Comportement de la mémoire de chargement pour les zones RAM et EPROM

Selon que vous choisissez une carte mémoire RAM ou EPROM pour étendre la mémoire de chargement, il peut s'ensuivre un comportement différent de cette mémoire lors du chargement, du rechargement et de l'effacement général.

Le tableau suivant représente les possibilités de chargement :

Type de mémoire	Possibilités de chargement	Type de chargement
Mémoire vive (RAM)	Chargement et effacement de blocs individuels	Liaison PG-CPU
	Chargement et effacement d'un programme S7 entier	Liaison PG-CPU
	Rechargement de blocs individuels	Liaison PG-CPU
EPROM intégrée (uniquement S7-300) ou enfichable	Chargement de programmes S7 entiers	Liaison PG-CPU
EPROM enfichable	Chargement de programmes S7 entiers	Chargement de l'EPROM sur la PG et enfichage de la carte mémoire dans la CPU. Chargement de l'EPROM sur la CPU.

Les programmes sauvegardés en mémoire vive sont perdus lorsque vous exécutez un effacement général de la CPU (MRES) ou lorsque vous retirez la CPU ou la carte mémoire RAM.

Les programmes enregistrés sur cartes à mémoire EPROM ne sont pas perdus en cas d'effacement général et restent conservés même sans sauvegarde par pile (transport, copies de sûreté).

## A.2.3 Mémoire système

### A.2.3.1 Utilisation des zones de mémoire système

La mémoire système des CPU S7 est subdivisée en zones d'opérandes (voir le tableau ci-après). En utilisant les opérations correspondantes, vous accédez dans votre programme aux données directement dans la plage d'opérandes en question.

Plage d'opérandes	Accès par des unités de taille suivante	Notation S7	Description
Mémoire image des entrées	Entrée (bit)	E	Au début de chaque cycle, la CPU lit les entrées provenant des modules d'entrées et enregistre ces valeurs dans la mémoire image des entrées.
	Octet d'entrée	EB	
	Mot d'entrée	EW	
	Double mot d'entrée	ED	
Mémoire image des sorties	Sortie (bit)	A	Pendant le cycle, le programme calcule les valeurs pour les sorties et les dépose dans la mémoire image des sorties. A la fin du cycle, la CPU écrit les valeurs de sortie calculées dans les modules de sorties.
	Octet de sortie	AB	
	Mot de sortie	AW	
	Double mot de sortie	AD	
Mémentos	Mémento (bit)	M	Cette zone met à disposition de l'espace mémoire pour les résultats intermédiaires calculés dans le programme.
	Octet de memento	MB	
	Mot de memento	MW	
	Double mot de memento	MD	
Temporisations	Temporisation (T)	T	Cette zone sert d'espace mémoire pour les temporisations.
Compteur	Compteur (Z)	Z	Cette zone sert d'espace mémoire pour les compteurs.
Bloc de données	Bloc de données ouvert avec AUF DB :	DB	Les blocs de données contiennent des informations pour le programme. Ils peuvent soit servir à tous les blocs de code (DB globaux), soit être associés à un FB ou à un SFB spécifique (DB d'instance).
	Bit de données	DBX	
	Octet de données	DBB	
	Mot de données	DBW	
	Double mot de données	DBD	
	Bloc de données ouvert avec AUF DI :	DI	
	Bit de données	DIX	
	Octet de données	DIB	
	Mot de données	DIW	
Double mot de données	DID		

Plage d'opérandes	Accès par des unités de taille suivante	Notation S7	Description
Données locales	Bit de données locales	L	Cette zone fournit de l'espace mémoire aux données temporaires d'un bloc pour la durée du traitement de ce bloc. La pile L sert également à la transmission de paramètres de blocs et à la sauvegarde de résultats intermédiaires pour les réseaux CONT.
	Octet de données locales	LB	
	Mot de données locales	LW	
	Double mot de données locales	LD	
Zone de périphérie : entrées	Octet d'entrée de périphérie	PEB	Les zones de périphérie des entrées et des sorties permettent l'accès direct à des modules d'entrées et de sorties centralisés et décentralisés.
	Mot d'entrée de périphérie	PEW	
	Double mot d'entrée de périphérie	PED	
Zone de périphérie : sorties	Octet de sortie de périphérie	PAB	
	Mot de sortie de périphérie	PAW	
	Double mot de sortie de périphérie	PAD	

Vous trouverez les zones d'adresses autorisées pour votre CPU dans les descriptions de CPU ainsi que dans les listes d'opérations suivantes :

- Manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU"
- Liste d'opérations "Automate programmable S7-300".
- Liste d'opérations "Automate programmable S7-400".



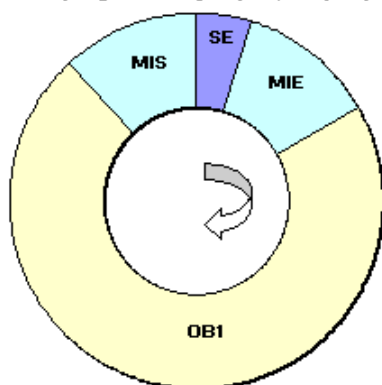
### A.2.3.2 Mémoire image des entrées/sorties

Lorsque le programme utilisateur accède aux zones d'opérandes Entrées (E) et Sorties (A), il n'interroge pas les états de signaux sur les modules de signaux TOR, mais accède à une zone de mémoire dans la mémoire système de la CPU et de la périphérie décentralisée. On appelle cette zone de mémoire "mémoire image du processus".

#### Mise à jour de la mémoire image du processus

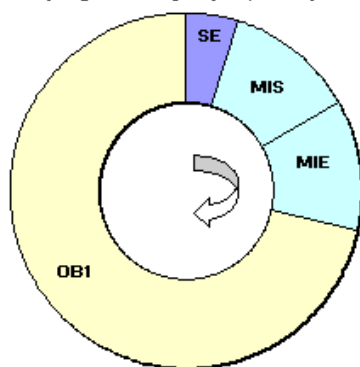
La figure suivante représente les étapes de traitement durant un cycle.

Traitement de programme cyclique (CPU jusqu'à 10/98)



Une fois que les tâches internes du système d'exploitation (SE) ont été réalisées, l'état des entrées est lu dans la mémoire image des entrées (MIE). Puis a lieu le traitement du programme utilisateur avec tous les blocs qui y sont appelés. Le cycle se termine par l'écriture de la mémoire image des sorties (MIS) sur les sorties des modules. La lecture de la mémoire image des entrées et l'écriture de la mémoire image des sorties sur les sorties des modules sont réalisées automatiquement par le système d'exploitation.

Traitement de programme cyclique (CPU à partir de 10/98)



Une fois que les tâches internes du système d'exploitation (SE) ont été réalisées, la mémoire image des sorties (MIS) est inscrite dans les sorties des modules et l'état des entrées est lu dans la mémoire image des entrées (MIE). Puis a lieu le traitement du programme utilisateur avec tous les blocs qui y sont appelés. L'écriture de la mémoire image des sorties (MIS) sur les sorties des modules et la lecture de la mémoire image des entrées sont réalisées automatiquement par le système d'exploitation.

## Avantages de la mémoire image du processus

L'accès à la mémoire image du processus offre, par rapport à l'accès direct aux modules d'entrées et de sorties, l'avantage que la CPU dispose d'une mémoire image des signaux du processus cohérente pendant la durée du traitement de programme cyclique. Si un état de signal change sur un module d'entrées pendant le traitement du programme, l'état de signal dans la mémoire image est conservé jusqu'à la mise à jour de la mémoire image du processus dans le cycle suivant. L'interrogation répétée d'un signal d'entrée dans un programme utilisateur permet de garantir la cohérence de l'information d'entrée.

En outre, l'accès à la mémoire image prend bien moins de temps que l'accès direct aux modules de signaux, car la mémoire image du processus se trouve dans la mémoire interne de la CPU.

## Mémoires image partielles

A côté de la mémoire image du processus (MIE et MIS) automatiquement actualisée par le système d'exploitation, vous pouvez paramétrer pour SIMATIC S7-400 jusqu'à 15 mémoires images partielles du processus par CPU (selon la CPU, n° 1 à n° 15 au plus ; voir manuel de référence "Systèmes d'automatisation S7-400/M7-400, Caractéristiques des modules"). Vous pouvez ainsi, en cas de besoin, mettre à jour des parties de la mémoire image du processus, et ce indépendamment de la mise à jour cyclique.

Chaque adresse d'entrée/sortie que vous avez affectée avec STEP 7 à une mémoire image partielle ne fait plus partie de la mémoire image des entrées-sorties gérée par l'OB1 ! Les adresses d'entrée/sortie ne peuvent être affectées qu'une seule fois via la mémoire image de l'OB1 et via l'ensemble des mémoires images partielles.

Vous définissez une mémoire image partielle avec STEP 7 lors de l'affectation d'adresses (quelles adresses d'entrée/sortie des modules sont mentionnées dans quelle mémoire image partielle). La mise à jour de la mémoire image partielle est effectuée soit par l'utilisateur à l'aide de SFC, soit automatiquement par le système au moyen d'un couplage à un OB.

Exception : Les mémoires image partielles des OB d'alarme de synchronisme d'horloge ne sont pas actualisées par le système quoiqu'elles soient couplées à un OB (OB 61 à OB 64).

---

### Nota

Dans les CPU S7-300, il est possible d'utiliser les entrées et sorties non occupées de la mémoire image comme zones de mémentos supplémentaires. Les programmes utilisant cette méthode ne peuvent s'exécuter sur les CPU S7-400 anciennes (c'est-à-dire avant 4/99) qu'à la condition suivante :

- il faut que les mémoires images utilisées comme mémentos se trouvent en dehors de la "Taille de la mémoire image" paramétrée, ou bien
  - il faut qu'elles se trouvent dans une mémoire image partielle qui n'est mise à jour ni par le système ni par SFC26/SFC27 !
-

### Mise à jour des mémoires images partielles par SFC

Des SFC vous permettent de mettre à jour l'ensemble de la mémoire image ou seulement des mémoires images partielles à partir du programme utilisateur.

Condition : la mémoire image partielle en question n'est pas mise à jour par le système !

- La SFC26 UPDAT\_PI met à jour la mémoire image des entrées.
- La SFC27 UPDAT\_PO met à jour la mémoire image des sorties.

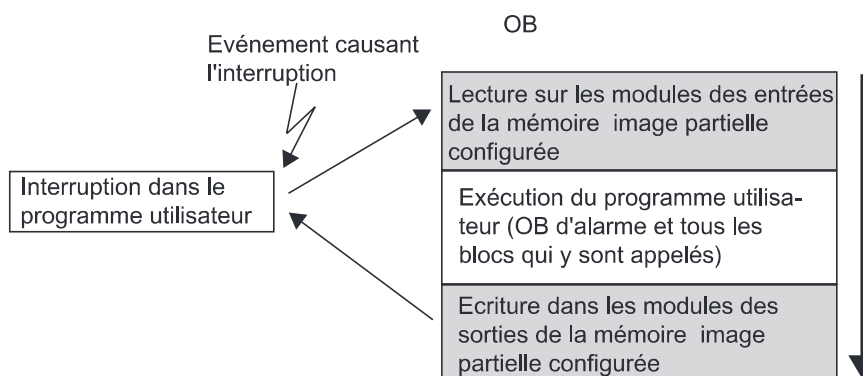
### Mise à jour des mémoires images partielles par le système

Vous pouvez aussi demander la mise à jour automatique par le système des mémoires images partielles à l'appel d'un OB - comme pour la mémoire image générale qui est mise à jour cycliquement avant ou après l'exécution de l'OB1. Cette fonction est paramétrable pour certaines CPU seulement.

En cours de fonctionnement, la mémoire image partielle affectée est mise à jour automatiquement :

- la mémoire image partielle des entrées avant l'exécution de l'OB,
- celle des sorties après l'exécution de l'OB

Vous paramétrez quelle mémoire image partielle est affectée à quel OB en même temps que la priorité des OB.



## Erreur d'accès à la périphérie lors de la mise à jour de mémoire image

Selon la famille de CPU (S7-300 et S7-400), la réaction pré réglée à une erreur durant la mise à jour de mémoire image n'est pas la même.

- S7-300 : pas d'inscription dans le tampon de diagnostic, pas d'appel d'OB, les octets d'entrée concernés sont mis à "0" et restent à "0" jusqu'à ce que l'erreur disparaisse.
- S7-400 : inscription dans le tampon de diagnostic et démarrage de l'OB85 à chaque accès à la périphérie à chaque mise à jour de la mémoire image concernée. Les octets d'entrée erronés sont mis à "0" à chaque accès à la mémoire image.

Avec les nouvelles CPU (à partir de 4/99), vous pouvez modifier par paramétrage la réaction aux erreurs d'accès à la périphérie, afin que la CPU

- ne génère une entrée dans le tampon de diagnostic et ne démarre l'OB85 que pour une erreur d'accès à la périphérie apparaissant ou disparaissant (avant l'appel de l'OB 85, les octets d'entrée erronés sont mis à "0" et ne sont plus écrasés par le système d'exploitation jusqu'à ce que l'erreur d'accès à la périphérie disparaisse) ou
- présente le comportement pré réglé des S7-300 (pas d'appel d'OB85, les octets d'entrée erronés sont mis à "0" et ne sont plus écrasés par le système d'exploitation jusqu'à ce que l'erreur soit corrigée) ou
- présente le comportement pré réglé des S7-400 (appel de l'OB85 à chaque accès à la périphérie les octets d'entrée erronés sont mis à "0" à chaque accès à la mémoire image).

## Nombre de démarrages de l'OB85

En plus de la réaction paramétrée aux erreurs d'accès à la périphérie (apparaissant/disparaissant ou à chaque accès à la périphérie), la plage d'adresses d'un module a aussi une influence sur le nombre de démarrages de l'OB85.

Pour un module dont la plage d'adresses va jusqu'au double-mot, l'OB85 démarre une fois : par exemple pour un module TOR possédant jusqu'à 32 entrées ou sorties ou pour un module analogique à 2 voies.

Pour les modules dont la plage d'adresses est plus grande, l'OB85 démarre autant de fois qu'il est nécessaire d'accéder à la plage avec des instructions sur double-mot : par exemple deux fois pour un module analogique à 4 voies.

### A.2.3.3 Pile des données locales

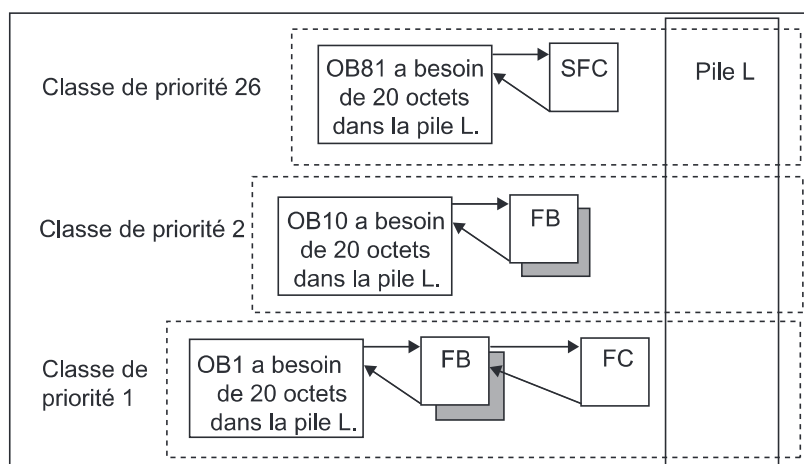
La pile L enregistre :

- les variables temporaires des données locales de blocs,
- les informations de déclenchement des blocs d'organisation,
- des informations pour la transmission de paramètres,
- des résultats intermédiaires dans les programmes CONT.

Vous pouvez, lors de la création de blocs d'organisation, déclarer des variables temporaires (TEMP) disponibles uniquement pendant le traitement du bloc et qui sont ensuite écrasées. Les données locales doivent être initialisées avant le premier accès. Chaque bloc d'organisation nécessite, en outre, 20 octets de données locales pour ses informations de déclenchement.

La CPU possède une mémoire limitée pour les variables temporaires (données locales) des blocs en cours de traitement. La taille de cette zone de mémoire dépend de la CPU. Par défaut, elle est subdivisée par parts égales entre les différentes classes de priorité. Ainsi, chaque classe de priorité dispose d'une zone de données locales en propre. Cela garantit que même les classes de priorité les plus élevées avec leurs OB associés ont suffisamment de place pour leurs données locales.

La figure suivante illustre l'affectation de données locales aux classes de priorité dans un exemple où dans la pile L, l'OB1 est interrompu par l'OB10, puis à nouveau par l'OB81.



#### Avertissement

Toutes les variables temporaires (TEMP) d'un OB et des blocs qui y sont appelés sont sauvegardées dans la pile L. Cette dernière peut déborder lorsque vous imbriquez trop de niveaux dans votre traitement des blocs.

Les CPU S7 passent à l'état "Arrêt" (STOP) lorsque vous dépassez la taille de pile L autorisée pour un programme.

Nous vous conseillons donc de tester la pile L (les variables temporaires) dans votre programme.

Tenez compte de l'espace mémoire requis pour les données locales d'OB d'erreur synchrones.

### **Affectation de données locales aux classes de priorité**

Les classes de priorité n'ont pas toutes besoin du même espace dans la pile des données locales. STEP 7 vous permet de paramétrer différemment la taille de la zone de données locales pour les différentes classes de priorité dans les CPU S7-400 et la CPU 318. Vous pouvez également désactiver les classes de priorité dont vous n'avez pas besoin, ce qui permet d'augmenter la zone de mémoire pour les autres classes de priorité dans les CPU S7-400 et la CPU 318. Les OB inactivés ne sont pas pris en compte lors du traitement du programme ; vous gagnez ainsi du temps de calcul.

En revanche, un volume fixe de données locales (256 octets) est affecté à chaque classe de priorité pour les autres CPU S7-300. Vous ne pouvez pas le modifier.

#### **A.2.3.4 Pile des interruptions**

Lorsque le traitement du programme est interrompu par un OB de priorité plus élevée, le système d'exploitation sauvegarde, dans la pile des interruptions (pile I), le contenu des accumulateurs et des registres d'adresse ainsi que le numéro et la taille des blocs de données ouverts.

A l'achèvement du traitement du nouvel OB, le système d'exploitation charge les informations contenues dans la pile I et reprend le traitement du bloc interrompu au point où s'était produite l'interruption.

A l'état de fonctionnement "Arrêt" (STOP), vous pouvez lire la pile des interruptions à la PG à l'aide de STEP 7. Vous trouverez ainsi plus facilement la cause du passage de la CPU à l'état "Arrêt".

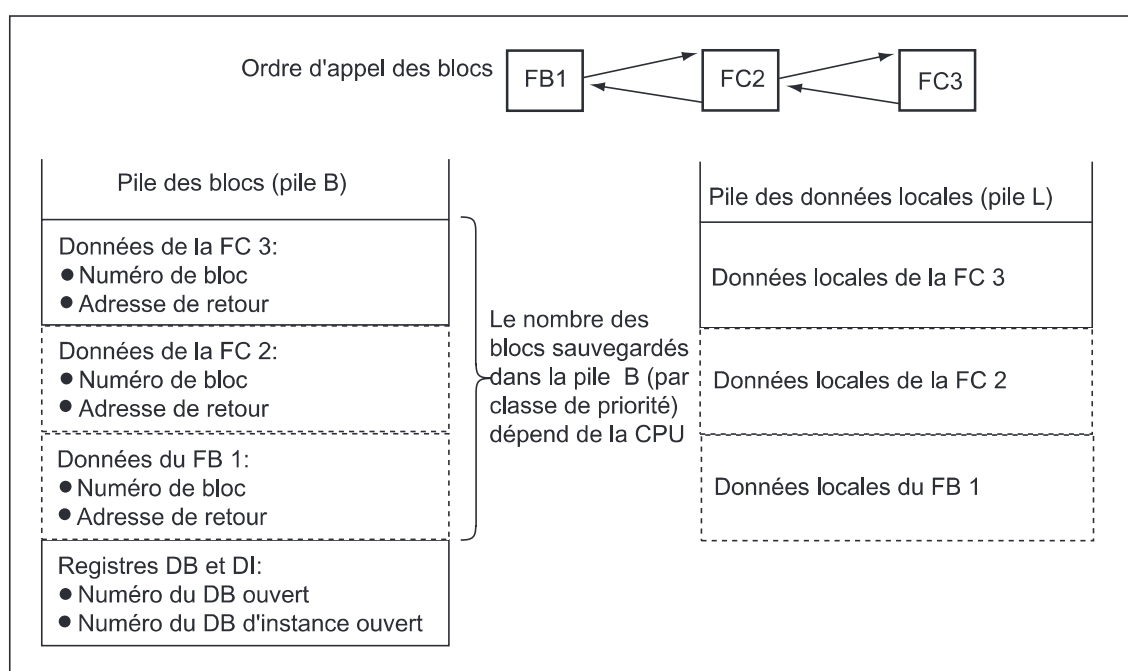
### A.2.3.5 Pile des blocs

Lorsque le traitement d'un bloc est interrompu par l'appel d'un autre bloc ou par une classe de priorité plus élevée (alarme/traitement d'erreur), la pile B enregistre les données suivantes :

- numéro, type (OB, FB, FC, SFB, SFC) et adresse de retour du bloc interrompu,
- numéro des blocs de données (des registres DB et DI) ouverts au moment de l'interruption.

Ces informations permettent de poursuivre l'exécution du programme utilisateur après l'interruption.

Si la CPU est à l'état de fonctionnement "Arrêt" (STOP), vous pouvez afficher la pile des blocs sur la console de programmation à l'aide de STEP 7. Cette pile énumère tous les blocs dont le traitement n'était pas terminé au moment où la CPU est passée à l'état "Arrêt". Ces blocs sont listés dans l'ordre dans lequel leur traitement avait commencé (voir la figure ci-après).



### Registres de bloc de données

Il existe deux registres de bloc de données. Ils contiennent les numéros des blocs de données ouverts.

- Le registre DB contient le numéro du bloc de données global ouvert.
- Le registre DI contient le numéro du bloc de données d'instance ouvert.

### A.2.3.6 Mémoire tampon de diagnostic

Dans la mémoire tampon de diagnostic, les messages de diagnostic sont affichés dans l'ordre de leur apparition. La première entrée contient l'événement le plus récent. Le nombre des entrées dans la mémoire tampon de diagnostic dépend du module et de son état de fonctionnement en cours.

Parmi les événements de diagnostic, on trouve :

- erreur dans un module,
- erreur d'assignation du processus
- erreur système dans la CPU,
- changements d'état de fonctionnement de la CPU
- erreurs dans le programme utilisateur,
- événements de diagnostic personnalisés (via la fonction système SFC 52).

### A.2.3.7 Exploitation de la mémoire tampon de diagnostic

Une partie de la liste d'état système est constituée de la mémoire tampon de diagnostic dans laquelle sont inscrites des informations plus précises sur les événements de diagnostic système et personnalisé dans l'ordre de leur apparition. Les informations relatives à un événement de diagnostic système sont les mêmes que les informations de déclenchement transmises au bloc d'organisation correspondant.

Il n'est pas possible d'effacer les entrées dans la mémoire tampon de diagnostic ; son contenu est conservé même après un effacement général.

La mémoire tampon de diagnostic permet :

- en cas d'arrêt de l'installation, d'évaluer les derniers événements avant le passage à l'état de fonctionnement "Arrêt" (STOP) et la cause de l'arrêt,
- de reconnaître plus rapidement l'origine des erreurs et d'améliorer ainsi la disponibilité de l'installation,
- d'évaluer et d'optimiser le comportement dynamique de l'installation.

### Organisation de la mémoire tampon de diagnostic

La mémoire tampon de diagnostic est organisée comme mémoire circulante pour un nombre maximal d'entrées dépendant du module. L'entrée la plus ancienne est écrasée lorsque le nombre maximal d'entrées est atteint et toutes les entrées sont déplacées en conséquence. Ainsi, l'entrée la plus récente est-elle toujours à la première place. La mémoire tampon de diagnostic de la CPU 314 S7-300 compte, par exemple, 100 entrées :

Le nombre des entrées affichées dans la mémoire tampon de diagnostic dépend du module et de son état de fonctionnement en cours. Pour certaines CPU, la longueur de la mémoire tampon de diagnostic est paramétrable.



## Contenu de la mémoire tampon de diagnostic

La zone **supérieure** contient la liste de tous les événements de diagnostic qui se sont produits, avec les informations suivantes :

- numéro d'ordre de l'entrée (l'événement le plus récent a le numéro 1),
- heure et date de l'événement de diagnostic : l'heure et la date du module sont indiqués, si le module possède une horloge. Il est donc important, pour pouvoir utiliser correctement ces indications horaires, que vous régliez la date et à l'heure du module et que vous vérifiiez ces dernières de temps à autre.
- texte abrégé de l'événement.

La zone **inférieure** affiche des informations supplémentaires sur l'événement sélectionné dans la zone supérieure. Ce sont, par exemple :

- numéro de l'événement,
- désignation de l'événement,
- changement d'état de fonctionnement occasionné par l'événement de diagnostic,
- renvoi à l'endroit de l'erreur dans un bloc (type et numéro de bloc et adresse relative) ayant entraîné l'inscription de l'événement,
- événement arrivant ou partant,
- informations complémentaires spécifiques de l'événement.

En cliquant sur le bouton "A propos de l'événement", vous pouvez afficher des informations complémentaires sur l'événement sélectionné dans la liste.

Vous trouverez des explications sur les ID d'événement dans l'aide sur les fonctions système et les blocs fonctionnels système (Sauts dans les descriptions de langage, aides sur les blocs, attributs système).

## Enregistrement dans un fichier de texte

Le bouton "Enregistrer sous" dans la page d'onglet "Mémoire tampon de diagnostic" de la boîte de dialogue "Etat du module" permet d'enregistrer le contenu du tampon de diagnostic sous forme de texte ASCII.

## Lecture de la mémoire tampon de diagnostic

Vous pouvez afficher le contenu de la mémoire tampon de diagnostic sur la PG/le PC via la page d'onglet "Mémoire tampon de diagnostic" de la boîte de dialogue "Etat du module", ou en effectuer la lecture dans un programme via la SFC51 RDSYSST.

## Dernière entrée avant arrêt

Vous pouvez demander que la dernière entrée de la mémoire tampon de diagnostic avant le passage de l'état "Marche" (RUN) à l'état "Arrêt" (STOP) soit envoyée à un appareil de contrôle déclaré (PG, OP, TD, par exemple). Cela permet de localiser et de corriger plus rapidement la cause du passage à l'état "Arrêt".

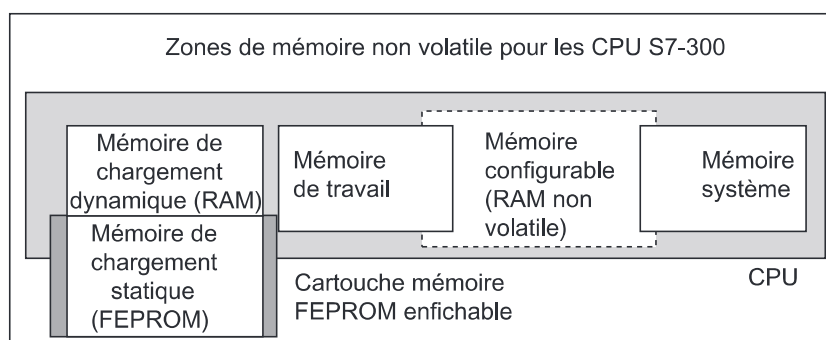
### A.2.3.8 Zones de mémoire rémanentes des CPU S7-300

En cas de coupure de courant ou d'effacement général (MRES), la mémoire de la CPU S7-300 - mémoire de chargement dynamique (RAM), mémoire de travail et mémoire système - est remise à zéro et toutes les données figurant dans ces zones sont perdues. Les CPU S7-300 fournissent toutefois des moyens pour conserver programme et données.

- Vous pouvez sauvegarder, à l'aide d'une pile, toutes les données se trouvant en mémoire de chargement, en mémoire de travail et dans certaines parties de la mémoire système.
- Vous pouvez sauvegarder votre programme dans l'EPROM (soit sous forme de carte mémoire, soit intégrée dans la CPU ; voir le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU")
- Vous pouvez sauvegarder un volume de données dépendant de la CPU dans une zone de mémoire vive non volatile (NVRAM).

#### Mémoire vive non volatile

Votre CPU S7-300 comporte une zone de mémoire vive non volatile (NVRAM ; voir la figure ci-après). Si vous avez rangé votre programme dans l'EPROM de la mémoire de chargement, vous pouvez également sauvegarder certaines données - en cas de coupure de courant ou de passage de l'état de fonctionnement "Arrêt" (STOP) à l'état "Marche" (RUN) - en exécutant la configuration correspondante.



A cet effet, vous réglez votre CPU de manière à sauvegarder les données suivantes en mémoire vive non volatile :

- informations rangées dans un DB (utile uniquement si vous avez aussi sauvegardé votre programme dans une EPROM de la mémoire de chargement),
- valeurs de temporisations et de compteurs,
- informations figurant dans des mémentos.

Chaque CPU permet de sauvegarder un nombre précis de temporisations, de compteurs et de mémentos. Chaque CPU fournit également un nombre donné d'octets où les données figurant dans des DB peuvent être conservées.

L'adresse d'interface multipoint (MPI) de votre CPU est sauvegardée en mémoire vive non volatile afin que votre CPU puisse toujours communiquer, même après une coupure de courant ou un effacement général.

## Utilisation d'une pile de sauvegarde

Avec une pile de sauvegarde, la mémoire de chargement et la mémoire de travail deviennent rémanentes en cas de coupure de courant. Les temporisations, compteurs et mementos que vous avez configurés pour sauvegarde en mémoire vive non volatile sont également conservés, indépendamment de la sauvegarde par pile.

## Configuration des données de la mémoire vive non volatile

Vous déterminez les zones de mémoire rémanentes lors de la configuration de la CPU avec STEP 7.

La taille de mémoire pouvant être configurée en mémoire vive non volatile dépend de la CPU ; vous ne pouvez pas sauvegarder plus de données que le volume précisé pour votre CPU.

### A.2.3.9 Zones de mémoire rémanentes des CPU S7-400

#### Fonctionnement sans sauvegarde

En cas de coupure de courant ou d'effacement général (MRES), la mémoire de la CPU S7-400 - mémoire de chargement dynamique (RAM), mémoire de travail et mémoire système - est remise à zéro et toutes les données figurant dans ces zones sont perdues.

En fonctionnement sans sauvegarde, seul un démarrage à chaud est possible et il n'existe pas de zones de mémoire rémanentes. Seuls sont conservés les paramètres MPI (par exemple, l'adresse MPI de la CPU) après une coupure de courant. Ainsi la CPU est-elle encore capable de communiquer après une coupure de courant ou un effacement général.

#### Fonctionnement avec sauvegarde

En fonctionnement avec sauvegarde :

- le contenu de toutes les zones RAM est intégralement conservé en cas de redémarrage après coupure de courant ;
- les zones d'opérandes mementos, temporisations et compteurs sont effacées. Les contenus des blocs de données conservés lors d'un démarrage à chaud ;
- le contenu de la mémoire de travail RAM est conservé à l'exception des mementos, temporisations et compteurs paramétrés comme non rémanents.

#### Configuration de zones de données rémanentes

Vous pouvez définir comme rémanents un nombre de mementos, temporisations et compteurs dépendant de la CPU. Ces données sont alors conservées en cas de démarrage à chaud en fonctionnement avec sauvegarde.

Dans STEP 7, vous pouvez paramétrer quels mementos, temporisations et compteurs doivent être rémanents au démarrage à chaud. Il n'est pas possible de sauvegarder plus de données que le volume autorisé pour votre CPU.

De plus amples informations sur le paramétrage de zones de mémoire rémanentes sont fournies dans le manuel de référence "Systèmes d'automatisation S7-400/M7-400 - Caractéristiques des CPU".

### **A.2.3.10 Objets mémoire configurables dans la mémoire de travail**

Pour certaines CPU, la taille des objets tels que les données locales ou la mémoire tampon de diagnostic peut être paramétrée dans HW Config. Si, par exemple, vous diminuez les valeurs par défaut, une plus grande partie de la mémoire de travail sera disponibles pour d'autres tâches. Le paramétrage de ces CPU peut être lu dans la page d'onglet "Mémoire" de l'état du module (bouton Détails).

Après modification de la configuration de la mémoire et chargement dans le système cible, un démarrage à froid du système cible s'avère nécessaire pour activer les modifications.

## A.3 Types de données et de paramètre

### A.3.1 Introduction aux types de données et de paramètre

Il faut indiquer le type de données pour toutes les données utilisées dans le programme utilisateur. On distingue entre :

- les types de données simples que STEP 7 met à votre disposition,
- les types de données complexes que vous pouvez créer en combinant des types de données simples et
- les types de paramètre avec lesquels vous définissez des paramètres à transmettre à des FB ou à des FC.

#### Informations générales

Les opérations LIST, LOG ou CONT utilisent des objets de données de taille définie. Les opérations combinatoires sur bit, par exemple, utilisent des bits. Les opérations de chargement et de transfert (LIST) ainsi que les opérations de transfert (LOG et CONT) utilisent des octets, mots et double mots.

Un bit est un chiffre binaire "0" ou "1". Un octet contient 8 bits, un mot 16 bits et un double mot 32 bits.

Les opérations arithmétiques utilisent également des octets, mots ou double mots. Dans ces opérandes de type octet, mot ou double mot vous pouvez coder des nombres de formats différents, comme par exemple les nombres entiers et les nombres à virgule flottante.

Si vous utilisez l'adressage symbolique, vous définissez des mnémoniques et leur affectez un type de données (voir le tableau suivant). Les différents types de données possèdent différentes options pour le format et diverses représentations de nombre.

Le présent chapitre ne décrit que certaines des notations possibles pour les nombres et les constantes. Le tableau suivant liste les formats de nombres et de constantes qui ne seront pas abordés en détail.

Format	Taille en bits	Représentation des nombres
hexadécimal	8, 16 et 32	B#16#, W#16# et DW#16#
binaire,	8, 16 et 32	2#
date CEI	16	D#
durée CEI	32	T#
heure	32	TOD#
CARACTERE	8	'A'

### A.3.2 Types de données simples

Chaque type de données simple a une longueur définie. Le tableau ci-après présente les types de données simples.

Type et description	Taille en bits	Options pour le format :	Plage et représentation des nombres (valeur minimale à valeur maximale)	Exemple
BOOL (bit)	1	Texte booléen	TRUE/FALSE	TRUE
BYTE (octet)	8	Nombre hexadécimal	B#16#0 à B#16#FF	L B#16#10 L byte#16#10
WORD (mot)	16	Nombre en binaire pur  Nombre hexadécimal  BCD Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111 W#16#0 à W#16#FFFF  C#0 à C#999 B#(0,0) à B#(255,255)	L 2#0001_0000_0000_0000  L W#16#1000 L word#16#1000 L C#998 L B#(10,20) L byte#(10,20)
DWORD (double mot)	32	Nombre en binaire pur  Nombre hexadécimal  Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111_1111_1111_1111_1111 DW#16#0000_0000 à DW#16#FFFF_FFFF B#(0,0,0,0) à B#(255,255,255,255)	2#1000_0001_0001_1000_1011_1011_0111_1111  L DW#16#00A2_1234 L dword#16#00A2_1234 L B#(1, 14, 100, 120) L byte#(1,14,100,120)
INT (entier)	16	Nombre décimal signé	-32768 à 32767	L 1
DINT (nombre entier de 32 bits)	32	Nombre décimal signé	L#-2147483648 à L#2147483647	L L#1
REAL (nombre à virgule flottante)	32	IEEE nombre à virgule flottante	Limite supérieure : ±3.402823e+38 Limite inférieure : ±1.175 495e-38	L 1.234567e+13
S5TIME (durée SIMATIC)	16	Durée S7 en pas de 10 ms (valeur par défaut)	S5T#0H_0M_0S_10MS à S5T#2H_46M_30S_0MS et S5T#0H_0M_0S_0MS	L S5T#0H_1M_0S_0MS L S5TIME#0H_1H_1M_0S_0MS
TIME (durée CEI)	32	Durée CEI en incréments de 1 ms, entier signé	-T#24D_20H_31M_23S_648MS à T#24D_20H_31M_23S_647MS	L T#0D_1H_1M_0S_0MS L TIME#0D_1H_1M_0S_0MS
DATE (date CEI)	16	Date CEI en incréments de 1 jour	D#1990-1-1 à D#2168-12-31	L D#1994-3-15 L DATE#1994-3-15

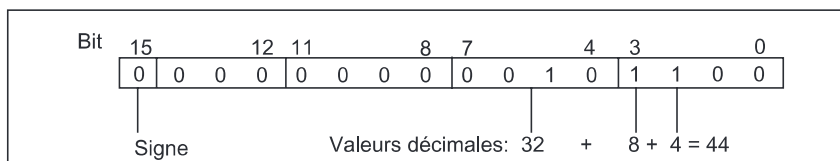
Type et description	Taille en bits	Options pour le format :	Plage et représentation des nombres (valeur minimale à valeur maximale)	Exemple
TIME_OF_DAY (heure)	32	Heure en pas de 1 ms	TOD#0:0:0.0 à TOD#23:59:59.999	L TOD#1:10:3.3 L TIME_OF_DAY#1:10:3.3
CHAR (caractère)	8	Caractères ASCII	'A','B' etc.	L 'E'

### A.3.2.1 Format du type de données INT (entiers de 16 bits)

Un nombre entier comporte un signe précisant s'il s'agit d'un entier positif ou négatif. L'espace occupé par un nombre entier (16 bits) dans la mémoire est d'un mot. Le tableau suivant représente la plage d'un nombre entier (16 bits).

Format	Plage
Entier (16 bits) :	-32 768 à +32 767

La figure suivante représente le nombre entier +44 sous forme de nombre en binaire pur.

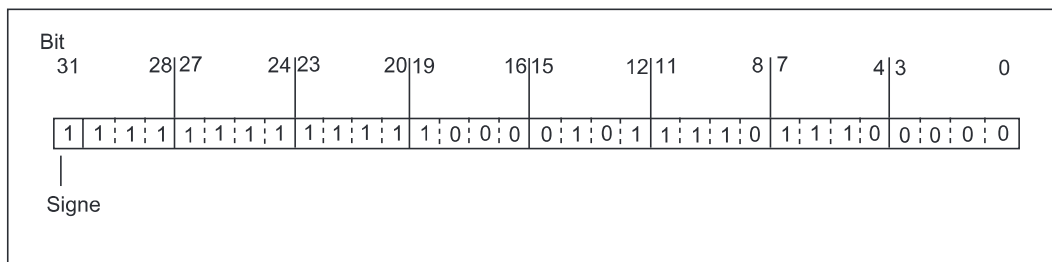


### A.3.2.2 Format du type de données DINT (nombres entiers de 32 bits)

Un nombre entier comporte un signe précisant s'il s'agit d'un entier positif ou négatif. L'espace occupé par un nombre entier (32 bits) dans la mémoire est de deux mots. Le tableau suivant représente la plage d'un nombre entier (32 bits).

Format	Plage
Nombre entier (32 bits) :	-2 147 483 648 à +2 147 483 647

La figure suivante représente le nombre entier -500 000 comme nombre en binaire pur. Dans le système binaire, la forme négative d'un nombre entier est représentée comme complément à deux du nombre entier positif. Vous obtenez le complément à deux d'un nombre entier en inversant les états de signaux de tous les bits, puis en additionnant +1 au résultat.





### A.3.2.3 Format du type de données REAL (nombres à virgule flottante)

La représentation générale d'un nombre à virgule flottante est "nombre =  $m * b^{\text{exposant } E}$ ". La base "b" et l'exposant "E" sont des nombres entiers, la mantisse "m" un nombre rationnel.

Ce type de représentation de nombres offre l'avantage de permettre de représenter de très grandes et de très petites valeurs dans un espace limité. Le nombre limité de bits pour la mantisse et l'exposant permet de représenter une vaste plage de nombres.

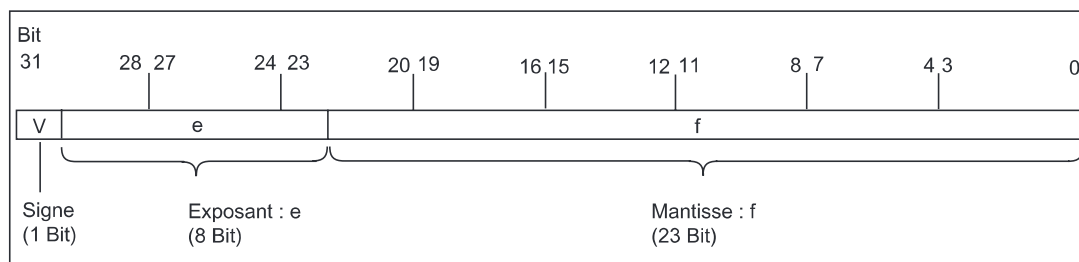
Le désavantage réside dans la limitation de la précision de calcul. Pour le calcul de la somme de deux nombres, par exemple, les exposants doivent être adaptés par décalage de la mantisse (virgule flottante) (addition des mantisses de deux nombres possédant le même exposant).

#### Format virgule flottante dans STEP 7

Dans STEP 7, les nombres à virgule flottante correspondent au format de base de simple largeur, comme décrit dans la norme ANSI/IEEE Std 754-1985, *IEEE Standard for Binary Floating-Point Arithmetic*. Ils sont formés des composants suivants :

- le signe s
- l'exposant  $e = E + \text{Bias}$  augmenté d'une constante (Bias = +127)
- la partie fractionnaire de la mantisse m.  
La partie entière de la mantisse n'est pas indiquée, car elle est toujours égale à 1 dans la plage de nombres valide.

Ces trois composants occupent au total un double mot (32 bits) :



Le tableau suivant illustre la valeur de chaque bit dans le format virgule flottante.

Composant du nombre à virgule flottante	Numéro du bit	Valeur
Signe s	31	
Exposant e	30	$2^{\text{exposant } 7}$
...	...	...
Exposant e	24	$2^{\text{exposant } 1}$
Exposant e	23	$2^{\text{exposant } 0}$
Mantisse m	22	$2^{\text{exposant } -1}$
...	...	...
Mantisse m	1	$2^{\text{exposant } -22}$
Mantisse m	0	$2^{\text{exposant } -23}$

Les trois composants **s**, **e** et **m** définissent la valeur d'un nombre représenté dans ce format par la formule :

$$\text{Nombre} = 1, m * 2^{\text{exposant} (e - \text{Bias})}$$

Où :

- $e : 1 \leq e \leq 254$
- Bias : Bias = 127. Ceci permet d'éviter un signe supplémentaire pour l'exposant.
- $s$  : pour un nombre positif,  $s = 0$  et pour un nombre négatif,  $s = 1$ .

### Plage de valeurs des nombres à virgule flottante

Avec le format virgule flottante représenté ci-avant :

- le nombre à virgule flottante le plus petit =  $1,0 * 2^{\text{exposant} (1-127)} = 1,0 * 2^{\text{exposant} (-126)}$   
= 1,175 495E-38 et
- le nombre à virgule flottante le plus grand =  $2-2^{\text{exposant} (-23)} * 2^{\text{exposant} (254-127)} = 2-2^{\text{exposant} (-23)} * 2^{\text{exposant} (+127)}$   
= 3,402 823E+38

Le nombre zéro est représenté par  $e = m = 0$  ;  $e = 255$  et  $m = 0$  signifie "infini".

Format	Plage <sup>1)</sup>
Nombres à virgule flottante selon la norme ANSI/IEEE	-3,402 823E+38 à -1,175 495E-38 et 0 et +1,175 495E-38 à +3,402 823E+38

Le tableau suivant représente l'état de signal des bits du mot d'état pour le résultat d'opérations sur des nombres à virgule flottante se trouvant hors de la plage admise.

Résultat dans la plage invalide	A1	A0	OV	OS
-1,175494E-38 < résultat < -1,401298E-45 (nombre négatif) Dépassement bas	0	0	1	1
+1,401298E-45 < résultat < +1,175494E-38 (nombre positif) Dépassement bas	0	0	1	1
Résultat < -3,402823E+38 (nombre négatif) Dépassement haut	0	1	1	1
Résultat > 3,402823E+38 (nombre positif) Dépassement haut	1	0	1	1
Pas de nombre à virgule flottante valide ou opération invalide (valeur d'entrée hors de la plage de valeurs admise)	1	1	1	1

#### Attention pour les opérations mathématiques :

L'on obtient par exemple le résultat "Pas de nombre à virgule flottante valide" lorsque l'on tente d'extraire la racine carrée de -2. Dans le cas d'opérations mathématiques, vous devez donc toujours d'abord évaluer les bits d'état avant de poursuivre le calcul avec le résultat.

#### Attention pour le "Forçage de variables" :

Lorsque l'on inscrit les valeurs pour les opérations sur nombres à virgule flottante dans des double mots de mémoire, par exemple, il est possible de modifier ces valeurs avec des modèles binaires quelconques. Chaque modèle binaire ne représente cependant pas un nombre valide !

## Précision dans le calcul sur nombres à virgule flottante



### Avertissement

Des imprécisions peuvent survenir dans des résultats de calculs importants sur des nombres présentant des ordres de grandeur très différents (plusieurs  $10^{\text{aines}}$  de puissances).

Dans STEP 7, la précision des nombres à virgule flottante est de 6 décimales. Lorsque vous saisissez des constantes à virgule flottante, êtes donc limité à 6 décimales au maximum.

### Nota

La précision de calcul de 6 décimales signifie par exemple que l'addition du nombre1 + nombre2 = nombre1, lorsque nombre1 est supérieur à nombre2 \* 10 exposant y, et y > 6 :

100 000 000 + 1 = 100 000 000.

## Exemples de nombres représentés dans le format virgule flottante

La figure suivante illustre le format de nombres à virgule flottante pour les valeurs décimales suivantes :

- 10,0
- Pi (3,141593)
- racine carrée de 2 (1,414214)

Le nombre **10,0** dans le premier exemple résulte de la manière suivante de son format virgule flottante (représentation en HEX : 4120 0000) :

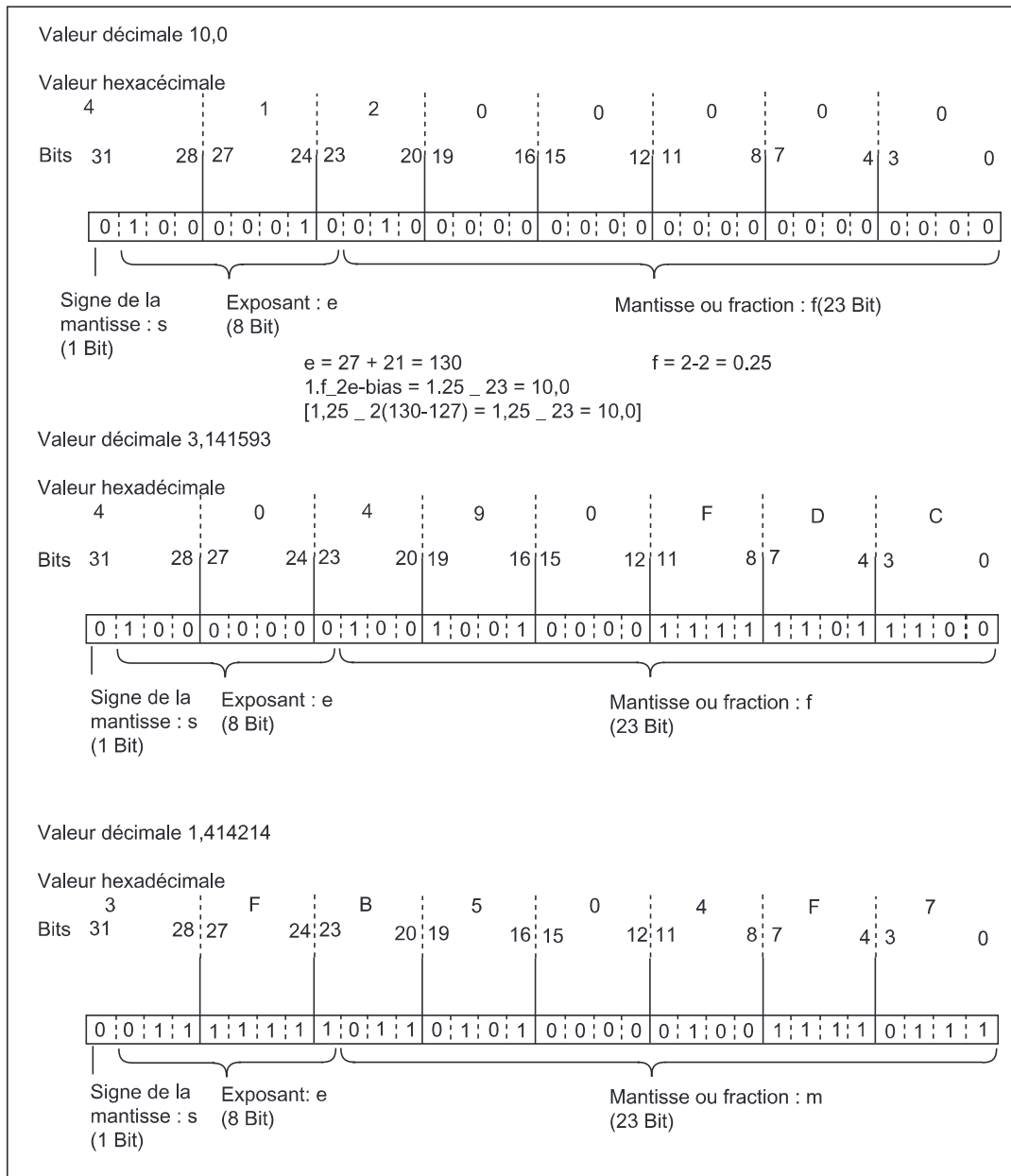
**e** = 2 exposant 7 + 2 exposant 1 = 130

**m** = 2 exposant (-2) = 0,25

Il en résulte :

$(1 + m) * 2^{\text{exposant} (e - \text{Bias})} = 1,25 * 2^{\text{exposant} 3} = 10.0.$

$[1,25 * 2^{\text{exposant} (130-127)} = 1,25 * 2^{\text{exposant} 3} = 10,0]$



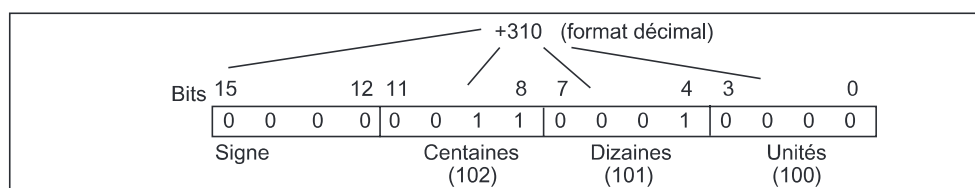
### A.3.2.4 Format des types de données WORD et DWORD pour les nombres décimaux codés binaire

Dans la représentation décimale codée binaire (DCB), un nombre décimal est représenté par des groupes de chiffres binaires (bits). Un groupe de 4 bits représente un chiffre ou le signe d'un nombre décimal. Les groupes de 4 bits forment un mot (16 bits) ou un double mot (32 bits). Les quatre bits de poids le plus fort indiquent le signe du nombre ("1111" signifie moins et "0000" plus). Les instructions comportant des opérandes décimaux codés binaires n'exploitent que le bit de poids le plus fort (15 en format mot, 31 en format double mot). Le tableau suivant indique le format et la plage des deux types de nombre DCB.

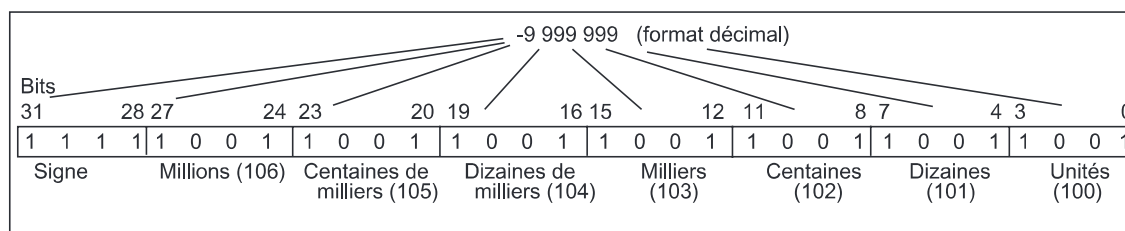
Format	Plage
Mot (16 bits, nombre DCB à trois positions signé)	-999 à +999
Double mot (32 bits, nombre DCB à 7 positions signé)	-9 999 999 à +9 999 999

Les figures suivantes donnent des exemples d'un nombre décimal codé binaire dans les formats suivants :

- Format mot

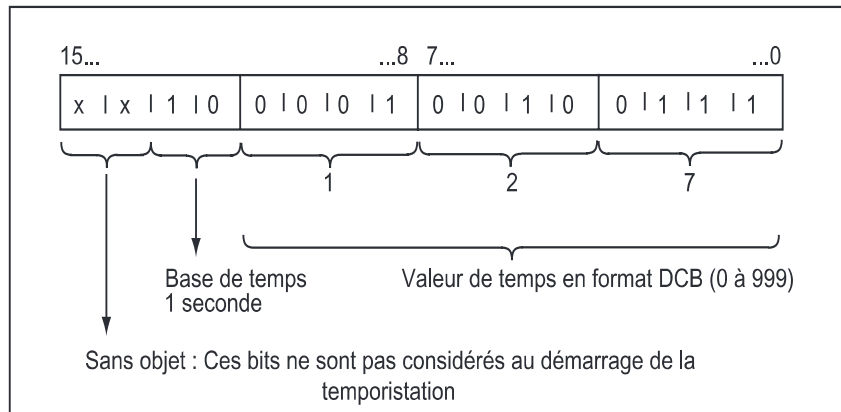


- Format double mot



### A.3.2.5 Format du type de données S5TIME (durée SIMATIC)

Lorsque vous saisissez la durée avec le type de données S5TIME, vos entrées sont enregistrées en format DCB. La figure suivante indique le contenu de l'opérande de temporisation pour une valeur de temps égale à 12 et une base de temps de 1 s.



Lorsque vous utilisez le type de données S5TIME, vous indiquez une valeur de temps comprise dans la plage 0 à 999 et spécifiez une base de temps (cf. tableau suivant). La base de temps correspond à l'intervalle dans lequel une durée diminue la valeur de temps d'une unité, jusqu'à atteindre "0".

Base de temps pour S5TIME

Base de temps	Code binaire pour la base de temps
10 ms	00
100 ms	01
1 s	10
10 s	11

Vous pouvez charger une valeur de temps prédéfinie en utilisant la syntaxe suivante :

- L<sup>1</sup> W#16#wxyz
  - où : w = base de temps (c'est-à-dire intervalle de temps ou résolution)
  - xyz = valeur de temps en format DCB
- L<sup>1</sup> S5T#aH\_bbM\_ccS\_dddMS
  - où : a = heures, bb = minutes, cc = secondes et ddd = millisecondes.
  - La sélection de la base de temps est automatique et la valeur est arrondie au nombre inférieur le plus proche avec cette base de temps.

Vous pouvez entrer une valeur de temps de 9 990 secondes ou 2H\_46M\_30S au maximum.

<sup>1</sup> = L doit uniquement être entré en programmation LIST

### A.3.3 Types de données complexes

Les types de données complexes définissent des groupes de données comportant plus de 32 bits ou des groupes de données composés à partir d'autres types de données. STEP 7 autorise les types de données complexes suivants :

- DATE\_AND\_TIME
- STRING
- ARRAY (tableau)
- STRUCT (structure)
- UDT (types de données utilisateur),
- FB et SFB

Le tableau ci-après décrit les types de données complexes. Vous définissez les structures et les tableaux soit dans la déclaration des variables du bloc de code, soit dans un bloc de données.

Type de données	Description
DATE_AND_TIME DT	Définit une zone de 64 bits (8 octets). Ce type de données sauvegarde en format décimal codé binaire.
STRING	Définit un ensemble de 254 caractères au maximum (type de données CHAR). La zone réservée à une chaîne de caractères est par défaut de 256 octets : c'est la mémoire nécessaire à la sauvegarde de 254 octets et d'un en-tête de deux octets. Vous pouvez indiquer le nombre de caractères dans la chaîne et réduire ainsi l'espace utilisé en mémoire (par exemple, STRING[9] 'Siemens').
ARRAY	Définit un agrégat multidimensionnel d'un même type de données (soit simple, soit complexe). Par exemple, "ARRAY[1..2,1..3] OF INT" correspond à un tableau de nombres entiers de format 2 x 3. Vous accédez aux données sauvegardées dans un tableau via l'indice (ex. : [2,2]). Un tableau peut comporter 6 dimensions au maximum ; l'indice peut être un nombre entier quelconque (de -32768 à 32767).
STRUCT	Définit un agrégat de types de données quelconques combinés. Vous pouvez, par exemple, définir un tableau de structures ou une structure contenant structures et tableaux.
UDT	Un UDT vous permet d'organiser des volumes de données importants et de simplifier la saisie des types de données lorsque vous voulez créer des blocs de données ou déclarer des variables dans la table de déclaration des variables. Dans STEP 7, vous pouvez combiner des types de données simples et complexes et, ainsi, créer votre propre type de données. Les UDT ont un nom propre et peuvent donc être utilisés plusieurs fois.
FB, SFB	Déterminent la structure du bloc de données d'instance associé et permettent la transmission de données d'instance pour plusieurs appels de FB dans un DB d'instance.

Les types de données structurés sont rangés par alignement sur les limites de mots (WORD aligned).

### A.3.3.1 Format du type de données DATE\_AND\_TIME (date et heure)

Lorsque vous saisissez la date et l'heure avec le type de données DATE\_AND\_TIME (DT), vos entrées sont enregistrées dans 8 octets en format DCB. Le type de données DATE\_AND\_TIME est formé de la plage suivante :

DT#1990-1-1-0:0:0.0 à DT#2089-12-31-23:59:59.999

Les exemples suivants indiquent les syntaxes possibles pour la saisie de la date et de l'heure du jeudi, le 25 décembre 1993, 8:12 et 34,567 secondes. Les deux formats suivants sont possibles :

- DATE\_AND\_TIME#1993-12-25-8:12:34.567
- DT#1993-12-25-8:12:34.567

Vous disposez des fonctions standard CEI (International Electrotechnical Commission) suivantes pour traiter le type de données DATE\_AND\_TIME :

- Conversion de la date et de l'heure au format DATE\_AND\_TIME  
FC3 : D\_TOD\_DT
- Détermination de la date à partir du format DATE\_AND\_TIME  
FC6 : DT\_DATE
- Détermination du jour de la semaine à partir du format DATE\_AND\_TIME  
FC7 : DT\_DAY
- Détermination de l'heure à partir du format DATE\_AND\_TIME  
FC8 : DT\_TOD

Le tableau suivant présente le contenu des octets qui contiennent l'information sur la date et l'heure. L'exemple montre la date et l'heure pour jeudi le 25 décembre 1993, 8:12 et 34,567 secondes.

Octet	Contenu	Exemple
0	année	B#16#93
1	mois	B#16#12
2	jour	B#16#25
3	heures	B#16#08
4	minutes	B#16#12
5	secondes	B#16#34
6	les deux chiffres de poids le plus fort de MSEC	B#16#56
7 (4MSB)	le chiffre de poids le plus faible de MSEC	B#16#7_
7 (4LSB)	jour de la semaine 1 = dimanche 2 = lundi ... 7 = samedi	B#16#_5



La plage autorisée pour le type de données "DATE\_AND\_TIME" est :

- min. : DT#1990-1-1-0:0:0.0
- max. : DT#2089-12-31-23:59:59.999

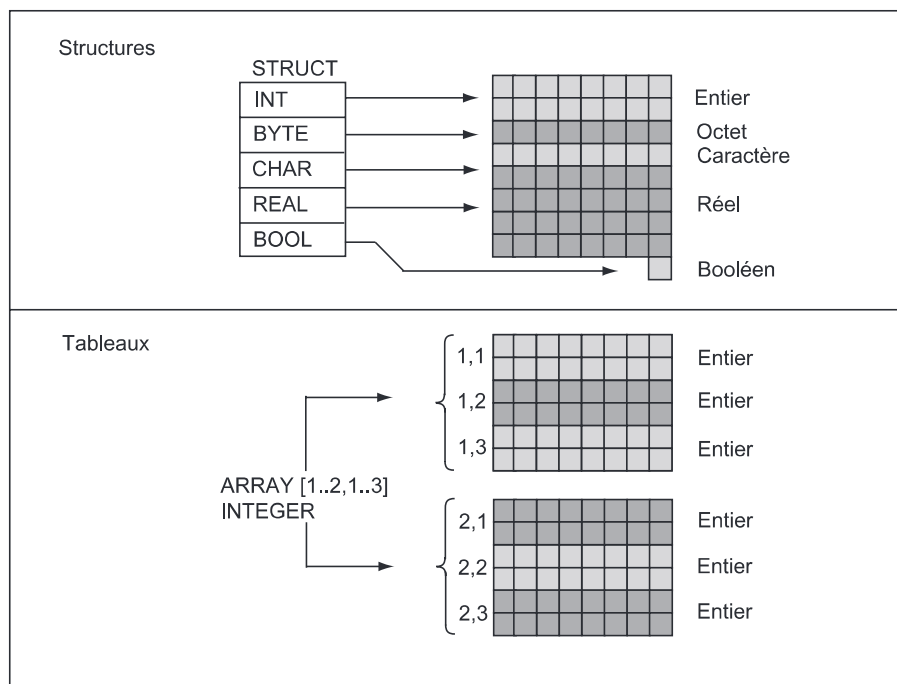
	<b>Plage de valeurs possible</b>	<b>Code DCB</b>
année	1990 - 1999	90 - 99
	2000 - 2089	00 - 89
mois	1 - 12	01 - 12
jour	1 - 31	01 - 31
heures	00 - 23	00 - 23
minutes	00 - 59	00 - 59
secondes	00 - 59	00 - 59
millisecondes	0 - 999	000 - 999
jour de la semaine	dimanche - samedi	1 - 7

### A.3.3.2 Utilisation de types de données complexes

Vous pouvez créer de nouveaux types de données en combinant des types de données simples et complexes pour obtenir les types de données complexes suivants :

- tableau (ARRAY) : agrégat de données de même type,
- structure (STRUCT) : agrégat de données de types différents,
- chaîne (STRING) : tableau à une dimension de 254 caractères (type de données CHAR) au maximum. Une chaîne ne peut être transmise que comme entité complète et la longueur de la chaîne doit être identique pour les paramètres formel et effectif du bloc.
- date et heure (DATE\_AND\_TIME) : année, mois, jour, heures, minutes, secondes, millisecondes et jour de la semaine.

La figure ci-après montre comment les tableaux et structures organisent des types de données en une zone de stockage d'informations. Vous pouvez définir un tableau ou une structure soit dans un DB, soit dans la table de déclaration des variables d'un FB, d'une FC ou d'un OB.



### A.3.3.3 Utilisation de tableaux pour l'accès aux données

#### Tableaux

Un tableau correspond à un agrégat de données de même type (simple ou complexe). Il n'est pas possible de définir un tableau de tableaux. Lorsque vous définissez un tableau :

- vous précisez son nom ;
- vous déclarez son type à l'aide du mot-clé ARRAY ;
- vous indiquez sa taille à l'aide d'indices. Vous entrez le premier et le dernier nombre pour chaque dimension (jusqu'à 6) dans le tableau. Vous indiquez les indices entre crochets, chaque dimension étant séparée par une virgule et les premier et dernier nombres pour chaque dimension par deux points. Voici, par exemple, comment définir un tableau tridimensionnel :

```
[1..5,-2..3,30..32]
```

- vous identifiez le type des données à sauvegarder dans le tableau.

#### Exemple 1

La figure ci-après montre un tableau de trois nombres entiers. Vous accédez aux données rangées dans le tableau à l'aide de l'indice, c'est-à-dire du nombre entre crochets. L'indice pour le deuxième nombre entier est, par exemple, Temp\_fonct[2].

Un indice peut être une valeur entière quelconque, même négative (-32768 à 32767). Il aurait également été possible de définir le tableau de la figure ci-après comme ARRAY [-1..1]. L'indice pour le premier entier serait alors Temp\_fonct[-1], celui pour le deuxième entier Temp\_fonct[0] et celui pour le troisième Temp\_fonct[1].

Adresse	Nom	Type	Val. init.	Commentaire
0.0		STRUCT		
+0.0	Temp_oper	ARRAY [1..3]		
*2.0		INT		
=3.0		END_STRUCT		

Temp\_fonct =  
 ARRAY [1..3]

INTEGER

}

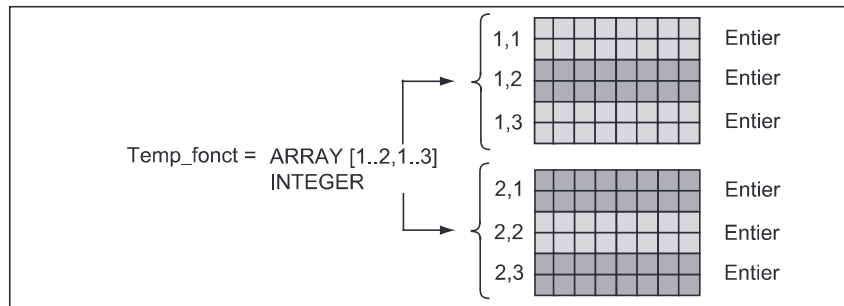
1Temp\_fonct [1]

2Temp\_fonct [2]

3Temp\_fonct [3]

## Exemple 2

Un tableau peut également décrire un agrégat multidimensionnel de types de données. La figure ci-après montre un tableau bidimensionnel de nombres entiers.



Vous accédez aux données de ce tableau à l'aide des indices. Pour l'exemple, le premier nombre entier est `Temp_fonct[1,1]`, le troisième `Temp_fonct[1,3]`, le quatrième `Temp_fonct[2,1]` et le sixième `Temp_fonct[2,3]`.

Un tableau peut avoir jusqu'à six dimensions (six jeux d'indices). Vous définissez la variable `Temp_fonct` comme étant un tableau à six dimensions de la manière suivante par exemple :

```
ARRAY [1..3,1..2,1..3,1..4,1..3,1..4]
```

Le premier élément de ce tableau sera donc `Temp_fonct[1,1,1,1,1,1]` et le dernier `Temp_fonct[3,2,3,4,3,4]`.

## Création d'un tableau

La définition d'un tableau se fait lors de la déclaration de données dans un DB ou dans la table de déclaration des variables. Pour déclarer un tableau, vous entrez le mot-clé `ARRAY` suivi de sa taille entre crochets :

```
[limite inférieure..limite supérieure]
```

Pour un tableau multidimensionnel, vous précisez les limites inférieure et supérieure pour chaque dimension, en les séparant par une virgule. Dans la figure ci-après, on déclare un tableau 2 x 3.

Adresse	Nom	Type	Val. init.	Commentaire
0.0		STRUCT		
+0.0	Chaleur_2X3	ARRAY[1..2,1..3]		
*2.0		INT		
=6.0		END_STRUCT		

## Saisie de valeurs initiales dans un tableau

Vous pouvez affecter une valeur initiale à chaque élément des tableaux que vous créez. Il existe deux méthodes pour saisir les valeurs initiales :

- Saisie de valeurs individuelles : Vous indiquez pour chaque élément du tableau une valeur autorisée (pour le type de données du tableau). Indiquez les valeurs dans l'ordre des éléments, par exemple [1,1]. Les différents éléments sont séparés par une virgule.
- Indication d'un facteur de répétition : Pour des éléments qui se suivent et doivent prendre la même valeur initiale, vous pouvez préciser le nombre d'éléments (facteur de répétition  $x$ ) et leur valeur initiale. Le format de saisie d'un facteur de répétition est  $x(y)$ ,  $x$  étant le facteur de répétition et  $y$  la valeur à répéter.

Pour l'exemple de la figure ci-avant, vous pouvez définir la valeur initiale des six éléments en entrant : 17, 23, -45, 556, 3342, 0. Mais pour leur donner à tous la valeur initiale 10, il vous suffirait d'indiquer : 6(10). Vous pourriez également donner une valeur individuelle aux deux premiers éléments et la valeur nulle aux quatre autres en précisant : 17,23,4(0).

## Accès aux données d'un tableau

Vous accédez aux données d'un tableau par l'indice de l'élément concerné dans le tableau. L'indice est combiné au mnémonique du tableau.

Exemple : si le tableau déclaré dans la figure ci-avant commence au premier octet du DB20 (Moteur), vous accédez au deuxième élément du tableau à l'aide de l'adresse suivante :

```
Moteur.Chaleur_2x3[1,2]
```

## Utilisation de tableaux comme paramètres

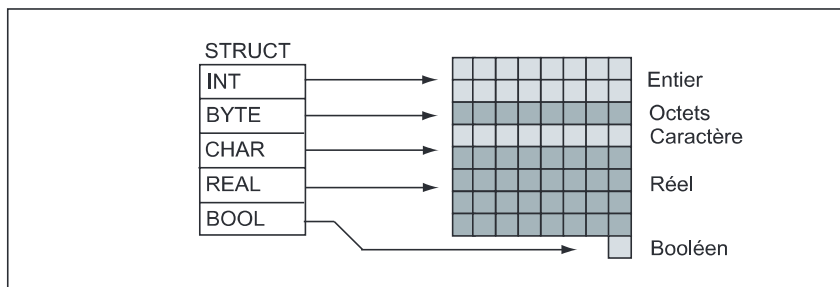
Vous pouvez transmettre des tableaux comme paramètres. Lorsque vous déclarez un paramètre ARRAY dans la déclaration des variables, vous devez transmettre le tableau complet, et non pas des éléments individuels. Il est toutefois possible d'affecter un élément de tableau à un paramètre lorsque vous appelez un bloc si cet élément correspond au type de données du paramètre.

Les tableaux que vous transmettez comme paramètres ne doivent pas nécessairement avoir le même nom (ou même avoir un nom), mais il faut qu'ils soient tous deux - paramètre effectif et paramètre formel - organisés de manière identique. Ainsi, un tableau 2x3 de nombres entiers ne peut-il être transmis comme paramètre que si le paramètre formel du bloc définit un tableau 2x3 de nombres entiers et si le paramètre effectif fourni dans l'opération d'appel est également un tableau 2x3 de nombres entiers.

### A.3.3.4 Utilisation de structures pour l'accès aux données

#### Structures

Une structure correspond à un agrégat de données de types différents (toute combinaison de types de données simples ou complexes, y compris tableaux et structures). Cela permet de regrouper des données selon la logique de votre processus. Cela permet également de transmettre des paramètres comme une entité de données, plutôt que sous la forme d'éléments distincts. La figure ci-après montre une structure constituée d'un nombre entier, d'un octet, d'un caractère, d'un nombre à virgule flottante et d'une valeur booléenne.



Une structure peut être imbriquée jusqu'à huit niveaux (par exemple, une structure de structures contenant des tableaux).

#### Création d'une structure

La définition d'une structure se fait lors de la déclaration de données dans un DB ou dans la déclaration des variables d'un bloc de code.

Dans la figure ci-après, on déclare une structure *lot\_1* constituée des éléments suivants : un nombre entier (pour la quantité), un octet (pour les données brutes), un caractère (pour le code de commande), un nombre à virgule flottante (pour la température) et un memento booléen (pour signaler l'achèvement).

Adresse	Nom	Type	Val. init.	Commentaire
0.0	lot_1	STRUCT		
+0.0	quantite	INT	100	
+2.0	donnees_brutes	BYTE		
+4.0	code_commande	CHAR		
+6.0	temperature	REAL	120	
+8.0	fin	BOOL	FALSE	
=10.0		END_STRUCT		

### Affectation de valeurs initiales à une structure

Pour affecter une valeur initiale à chaque élément d'une structure, vous indiquez une valeur autorisée pour le type de données et le nom de chaque élément. Vous pourriez affecter les valeurs initiales suivantes à l'exemple de la figure ci-avant :

Quantité	=	100
Données brutes	=	B#(0)
Code de commande	=	'Z'
Température	=	120
Achèvement	=	False

### Sauvegarde des données et accès aux données dans une structure

Vous accédez aux éléments individuels d'une structure. Vous pouvez utiliser l'adresse symbolique - *lot\_1.temperature*, par exemple - ou l'adresse absolue sous laquelle est rangé l'élément. Si, par exemple, *lot\_1* est sauvegardé dans le DB20 à partir de l'octet 0, l'adresse absolue de *quantite* est *DB20.DBW0* et celle de *temperature* est *DB20.DBD6*.

### Utilisation de structures comme paramètres

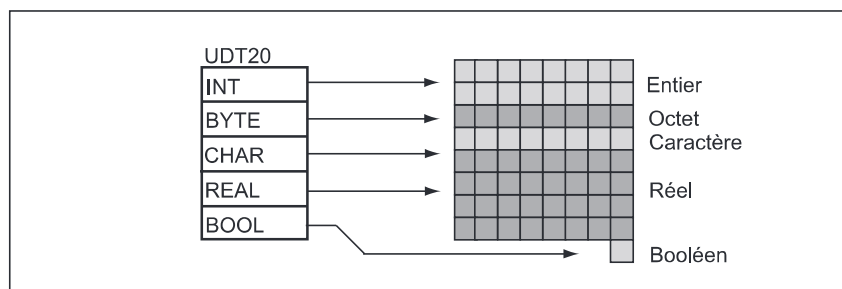
Vous pouvez transmettre des structures comme paramètres. Lorsque vous déclarez un paramètre STRUCT dans la déclaration des variables, vous devez transmettre une structure d'organisation identique. Il est possible d'affecter un élément de structure à un paramètre lorsque vous appelez un bloc si cet élément correspond au type de données du paramètre.

Lorsque vous transmettez des structures comme paramètres, elles doivent être toutes deux - paramètre effectif et paramètre formel - organisées de manière identique : elles doivent avoir les mêmes types de données dans le même ordre.

### A.3.3.5 Utilisation de types de données utilisateur pour l'accès aux données

#### Types de données utilisateur

Les types de données utilisateur (user data type, UDT) peuvent combiner des types de données simples et complexes. Vous pouvez attribuer un nom aux UDT et les utiliser plusieurs fois. La figure ci-après montre la structure d'un type de données utilisateur constitué d'un nombre entier, d'un octet, d'un caractère, d'un nombre à virgule flottante et d'une valeur booléenne.



Il vous suffit alors, au lieu d'entrer tous les types de données individuellement ou sous forme de structure, d'indiquer "UDT20" comme type de données et STEP 7 allouera automatiquement l'espace nécessaire en mémoire.

#### Création d'un type de données utilisateur

Vous définissez les UDT dans STEP 7. La figure suivante montre un UDT composé des éléments suivants : un nombre entier (pour la quantité), un octet (pour les données brutes), un caractère (pour le code de commande), un nombre à virgule flottante (pour la température) et un memento booléen (pour signaler l'achèvement). Vous pouvez affecter un mnémonique à cet UDT dans la table des mnémoniques (donnees\_process, par exemple).

Adresse	Nom	Type	Val. init.	Commentaire
0.0	lot_1	STRUCT		
+0.0	quantite	INT	100	
+2.0	donnees_brutes	BYTE		
+4.0	code_commande	CHAR		
+6.0	temperature	REAL	120	
+8.0	fin	BOOL	FALSE	
=10.0		END_STRUCT		

Après avoir créé un UDT, vous pouvez l'utiliser comme un type de données, par exemple comme si pour une variable vous déclariez le type de données *UDT200* dans un DB (ou dans la table de déclaration des variables d'un FB).

La figure suivante montre un DB avec la variable *donnees\_processus\_1* de type de données UDT200. Vous indiquez uniquement *UDT200* et *donnees\_processus\_1*. Les autres champs sont créés à la compilation du DB.

Adresse	Nom	Type	Val. init.	Commentaire
0.0		STRUCT		
+6.0	donnees_proc_1	UDT200		
=6.0		END_STRUCT		



## Affectation de valeurs initiales à un type de données utilisateur

Pour affecter une valeur initiale à chaque élément d'un UDT, vous indiquez une valeur autorisée pour le type de données et le nom de chaque élément. Vous pourriez affecter les valeurs initiales suivantes à l'exemple de la figure ci-avant :

Quantité	=	100
Données brutes	=	B#16#0
Code de commande	=	'Z'
Température	=	1.200000e+002
Achèvement	=	False

Lorsque vous déclarez une variable comme étant d'un type de données utilisateur, les valeurs initiales pour cette variable seront les valeurs entrées à la création de l'UDT.

## Sauvegarde des données et accès aux données dans un type de données utilisateur

Vous accédez aux éléments individuels d'un UDT. Vous pouvez utiliser l'adresse symbolique - *lot\_1.temperature*, par exemple - ou l'adresse absolue sous laquelle est rangé l'élément. Si, par exemple, *lot\_1* est sauvegardé dans le DB20 à partir de l'octet 0, l'adresse absolue de *quantité* est *DB20.DBW0* et celle de *temperature* est *DB20.DBD6*.

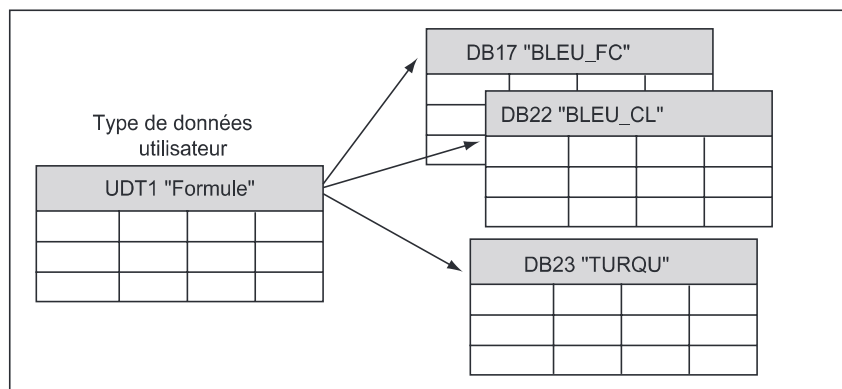
## Utilisation de types de données utilisateur comme paramètres

Vous pouvez transmettre des variables de type de données UDT comme paramètres. Lorsque vous déclarez un paramètre comme UDT dans la déclaration des variables, vous devez transmettre un UDT dont les éléments de données ont une organisation identique. Il est également possible d'affecter un élément d'UDT à un paramètre lorsque vous appelez un bloc si cet élément correspond au type de données du paramètre.

## Avantages des DB associés à un UDT

Vous pouvez, à l'aide des UDT que vous avez créés, générer de nombreux blocs de données ayant la même organisation de données. Vous pouvez adapter ces blocs de données à chaque tâche en saisissant des valeurs effectives différentes.

Si, par exemple, vous organisez un UDT pour une formule (par exemple, le mélange de couleurs), vous pouvez associer à cet UDT plusieurs DB contenant à chaque fois d'autres indications de quantités.



L'organisation de l'UDT conditionne celle du bloc de données associé.

### A.3.4 Types de paramètre

En plus des types de données simples et complexes, vous pouvez définir des types de paramètre pour des paramètres formels devant être transmis entre blocs. STEP 7 dispose des types de paramètre ci-après.

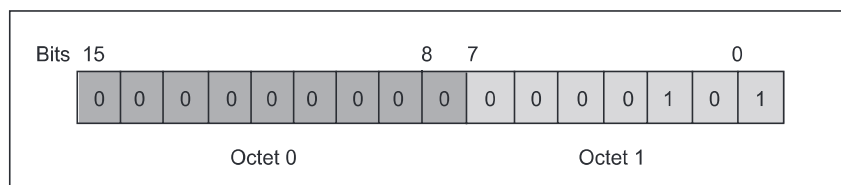
- **TIMER** ou **COUNTER** : identifient une temporisation ou un compteur précis devant être utilisé lors du traitement. Le paramètre effectif que vous fournissez à un paramètre formel de type **TIMER** ou **COUNTER** doit être une temporisation ou un compteur : vous indiquez un **T** ou un **Z** suivi d'un nombre entier positif.
- **BLOCK** : identifie un bloc précis devant être utilisé comme entrée ou comme sortie. La déclaration du paramètre détermine le type de bloc (**FB**, **FC**, **DB**, etc.) à utiliser. Si vous indiquez un paramètre effectif pour un paramètre formel de type **BLOCK**, ce doit être une adresse de bloc. Exemple : "FC101" en adressage absolu ou "Soupape" en adressage symbolique.
- **POINTER** : référence l'adresse d'une variable. Un pointeur contient une adresse au lieu d'une valeur. Lorsque vous indiquez un paramètre effectif pour un paramètre formel de type **POINTER**, ce doit être l'adresse. Dans STEP 7, vous pouvez préciser un pointeur en format de pointeur ou simplement comme adresse (par exemple, M50.0). Exemple de format de pointeur pour l'adressage de données commençant à M 50.0 : P#M50.0
- **ANY** : s'utilise lorsque le type de données du paramètre effectif est inconnu ou lorsqu'on peut faire appel à un type de données quelconque. Vous trouverez de plus amples informations sur le paramètre **ANY** dans les paragraphes Format du type de données Paramètre **ANY** ou Utilisation du type de données Paramètre **ANY**.

Un type de données paramètre peut également être un type de données utilisateur (UDT). Vous trouverez de plus amples informations sur les UDT dans le paragraphe "Utilisation de types de données utilisateur pour l'accès aux données".

Paramètre	Taille	Description
TIMER	2 octets	Identifie une temporisation précise que le programme dans le bloc de code appelé doit utiliser. Format : T1
COUNTER	2 octets	Identifie un compteur précis que le programme dans le bloc de code appelé doit utiliser. Format : Z10
BLOCK_FB BLOCK_FC BLOCK_DB BLOCK_SDB	2 octets	Identifie un bloc précis que le programme dans le bloc de code appelé doit utiliser. Format : FC101 DB42
POINTER	6 octets	Identifie l'adresse. Format : P#M50.0
ANY	10 octets	Utilisé lorsque le type de données du paramètre effectif est inconnu. Format: P#M50.0 BYTE 10 format ANY pour types de données P#M100.0 WORD 5 L#1COUNTER 10 format ANY pour types de paramètre

### A.3.4.1 Format des types de paramètre BLOCK, COUNTER et TIMER

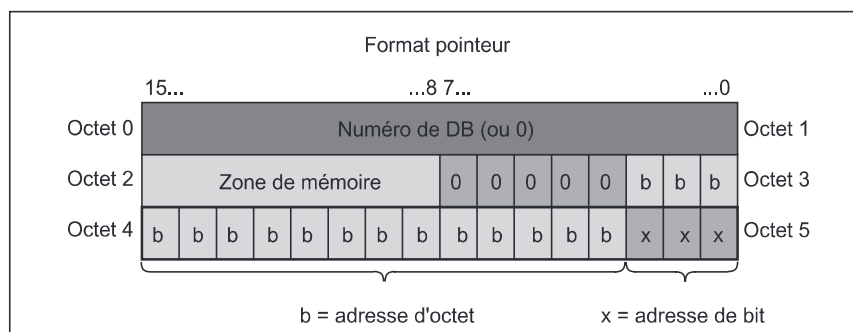
STEP 7 enregistre les types de paramètre BLOCK, COUNTER et TIMER sous forme de nombres binaires dans un mot (32 bits). La figure suivante montre le format de ces types de paramètre.



Le nombre autorisé de blocs, temporisations et compteurs dépend de la version de votre CPU S7. Vous trouverez de plus amples informations sur le nombre autorisé de temporisations et de compteurs ainsi que sur le nombre maximal de blocs disponibles dans les fiches techniques relatives à votre CPU dans le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" ou dans le manuel d'installation "Systèmes d'automatisation S7-400/M7-400, Installation et configuration".

### A.3.4.2 Format du type de données Paramètre POINTER

STEP 7 enregistre le type de données Paramètre POINTER dans 6 octets (48 bits). La figure suivante montre la nature des données enregistrées dans chaque octet.



Le type de données Paramètre POINTER enregistre les informations suivantes :

- Numéro de DB (ou 0, lorsque les données ne sont pas enregistrées dans un DB)
- Zone de mémoire dans la CPU (le tableau suivant indique les codes hexadécimaux des zones de mémoire pour le type de données Paramètre POINTER)

Code hexadécimal	Zone de mémoire	Description
b#16#81	E	Zone de mémoire des entrées
b#16#82	A	Zone de mémoire des sorties
b#16#83	M	Zone de mémoire des mémentos
b#16#84	DB	Bloc de données
b#16#85	DI	Bloc de données d'instance
b#16#86	L	Données locales (pile L)
b#16#87	V	Données locales précédentes

- Adresse des données (en format octet.bit)  
STEP 7 propose le format pointeur : p#zone de mémoire octet.bit\_adresse. (Si le paramètre formel a été déclaré comme type de données Paramètre POINTER, il vous suffit d'indiquer la zone de mémoire et l'adresse. STEP 7 convertit automatiquement votre entrée en format pointeur.) Les exemples suivants montrent comment vous saisissez le type de données Paramètre POINTER pour les données commençant à M50.0 :
- P#M50.0
- M50.0 (si le paramètre formel a été déclaré comme POINTER)

### A.3.4.3 Utilisation du type de données Paramètre POINTER

Un pointeur est utilisé pour adresser un opérande. L'avantage de ce type d'adressage est que vous pouvez modifier de manière dynamique l'opérande de l'instruction durant l'exécution du programme.

#### Pointeur pour l'adressage indirect en mémoire

Les instructions de programme utilisant l'adressage indirect en mémoire, sont composées d'une opération, d'un identificateur d'opérande et d'un décalage (qui doit être indiqué entre crochets).

Exemple de pointeur en format double mot :

L	P#8.7	charger la valeur du pointeur dans l'ACCU 1
T	MD2	transférer le pointeur dans MD2.
U	E [MD2]	interroger l'état de signal à l'entrée E 8.7
=	A [MD2]	et affecter l'état de signal à la sortie A 8.7

#### Pointeur pour l'adressage intrazone et interzone

Les instructions de programme utilisant ce type d'adressage sont composées d'une opération et des éléments suivants : identificateur d'opérande, identificateur de registre d'adresse, décalage.

Le registre d'adresse (AR1/2) et le décalage doivent être indiqués ensemble entre crochets.

#### Exemple d'adressage intrazone

Le pointeur ne contient aucune indication de zone de mémoire :

L	P#8.7	charger la valeur du pointeur dans l'ACCU 1
LAR1		charger le pointeur de l'ACCU 1 dans AR1
U	E [AR1, P#0.0]	interroger l'état de signal de l'entrée E 8.7 et
=	A [AR1, P#1.1]	affecter l'état de signal à la sortie A 10.0

Le décalage 0.0 n'a pas d'effet. La sortie 10.0 se calcule à partir de 8.7 (AR1) plus le décalage 1.1. Le résultat est 10.0 et non pas 9.8, voir le format du pointeur.

### Exemple d'adressage interzone

Dans l'adressage interzone, la zone de mémoire est précisée dans le pointeur (dans l'exemple, E ou A).

L	P# E8.7	charger la valeur du pointeur et l'identification de zone dans l'ACCU 1
LAR1		charger la zone de mémoire E et l'adresse 8.7 dans AR1
L	P# A8.7	charger la valeur du pointeur et l'identification de zone dans l'ACCU 1
LAR2		charger la zone de mémoire A et l'adresse 8.7 dans AR2
U	[AR1, P#0.0]	interroger l'état de signal de l'entrée E 8.7 et
=	[AR2, P#1.1]	affecter l'état de signal à la sortie A 10.0.

Le décalage 0.0 n'a pas d'effet. La sortie 10.0 se calcule à partir de 8.7 (AR2) plus 1.1 (décalage). Le résultat est 10.0 et non pas 9.8, voir format du pointeur.

#### A.3.4.4 Bloc pour modifier le pointeur

L'exemple de bloc FC 3 "Décalage de pointeurs" permet de modifier l'adresse de bit ou l'adresse d'octet d'un pointeur. A l'appel de la FC, le pointeur à modifier est transmis à la variable "Pointeur" (vous pouvez utiliser des pointeurs interzone et intrazone en format double mot).

Le paramètre "bit-octet" vous permet de modifier l'adresse de bit ou l'adresse d'octet du pointeur (0 : adresse de bit, 1 : adresse d'octet). La variable "valeur\_inc" (en format entier) indique la valeur qui doit être additionnée ou soustraite au contenu de l'adresse. Vous pouvez également indiquer des nombres négatifs pour décrémenter l'adresse.

Pour la modification de l'adresse de bit, un transfert dans l'adresse d'octet est effectué (également pour la décrémentation) ; par exemple :

- P#M 5.3, bit\_octet = 0, valeur\_inc = 6 => P#M 6.1 ou
- P#M 5.3, bit\_octet = 0, valeur\_inc = -6 => P#M 4.5.

La fonction n'a pas d'effet sur l'information de zone du pointeur.

Le débordement haut/bas du pointeur corrige la FC. Dans ce cas, le pointeur n'est pas modifié et la variable côté sortie "RET\_VAL" (traitement d'erreur possible) est mise à "1" (jusqu'au prochain traitement correct de la FC 3). Ceci est le cas lorsque :

- L'adresse de bit est sélectionnée et valeur\_inc >7 ou <-7.
- L'adresse de bit ou l'adresse d'octet sont sélectionnées et la modification aurait pour conséquence une adresse d'octet "négative".
- L'adresse de bit ou l'adresse d'octet sont sélectionnées et la modification aurait pour conséquence une adresse d'octet de taille non autorisée.

**Exemple de bloc pour modifier le pointeur dans LIST :**

```

FUNCTION FC 3: BOOL
TITLE =Rangement de pointeurs
//La FC 3 peut être utilisée pour modifier des pointeurs.
AUTHOR : AUT1CS1
FAMILY : INDADR
NAME : ADRPOINT
VERSION : 0.0

VAR_INPUT
    Bit_octet : BOOL ; //0 : adresse de bit, 1 : adresse d'octet
    Valeur_inc : INT ; //Incrément (si valeur négative => décrémentation/si valeur positive
                        => incrémentation)
END_VAR

VAR_IN_OUT
    Pointeur : DWORD ; //Pointeur à utiliser
END_VAR

VAR_TEMP
    Valeur_inc1 : INT ; //Incrément de valeur intermédiaire
    Pointeur1 : DWORD ; //Pointeur de valeur intermédiaire
    Val_int : DWORD ; //Variable auxiliaire
END_VAR

BEGIN
NETWORK
TITLE =
//Le bloc corrige automatiquement les modifications qui modifient les informations de zone
//du pointeur, ou qui conduisent à des pointeurs "négatifs" !
    SET    ; //Mettre le RLG à 1 et
    R      #RET_VAL; //remettre le débordement à zéro
    L      #Pointeur; //Affecter le pointeur temporaire
    T      #Pointeur1; //de valeur intermédiaire
    L      #Valeur_inc; //Affecter l'incrément temporaire
    T      #Valeur_inc1; //de valeur intermédiaire
    U      #Bit_octet; //lorsque =1, alors opération sur l'adresse d'octet
    SPB   Octet; //Saut au calcul de l'adresse d'octet
    L      7; //Si valeur incrément > 7,
    L      #Valeur_inc1;
    <I    ;
    S      #RET_VAL; //alors mettre RET_VAL à 1 et
    SPB   Fin; //sauter à la fin
    L      -7; //Si valeur incrément < -7,
    <I    ;
    S      #RET_VAL; //alors mettre RET_VAL à 1 et
    SPB   Fin; //sauter à la fin

```

```
U      L      1.3; //si bit 4 de la valeur = 1 (valeur_inc négative)
SPB   neg; //alors sauter à la soustraction des adresses de bit
L     #Pointeur1; //Charger l'information d'adresse du pointeur
L     #Valeur_inc1; //et additionner l'incrément
+D    ;
SPA   test; //Sauter au test de résultat négatif
neg:  L     #Pointeur1; //Charger l'information d'adresse du pointeur
      L     #Valeur_inc1; //Charger l'incrément
      NEGI  ; //Effectuer la négation de la valeur négative,
      -D    ; //soustraire la valeur
      SPA   test; //et sauter au test
Octet: L     0; //Début de la modification de l'adresse d'octet
      L     #Valeur_inc1; //Si incrément >=0, alors
      <I    ;
      SPB   pos; //sauter à l'addition, sinon
      L     #Pointeur1; //charger l'information d'adresse du pointeur,
      L     #Valeur_inc1; //charger l'incrément,
      NEGI  ; //effectuer la négation de la valeur négative,
      SLD   3; //décaler l'incrément de 3 positions vers la gauche,
      -D    ; //soustraire la valeur
      SPA   test; //et sauter au test
pos:  SLD   3; //décaler l'incrément de 3 positions vers la gauche
      L     #Pointeur1; //charger l'information d'adresse du pointeur
      +D    ; //additionner l'incrément
test: T     #Valeur_int; //Transférer les calculs de résultat dans Valeur_int,
      U     L     7.3; //Si adresse d'octet invalide (trop grande ou
      S     #RET_VAL; //négative), alors mettre RET_VAL à 1
      SPB   Fin; //et sauter à la fin,
      L     #Valeur_int; //sinon transférer le résultat
      T     #Pointeur; //dans le pointeur
Fin:  NOP   0;
END_FUNCTION
```

### A.3.4.5 Format du type de données Paramètre ANY

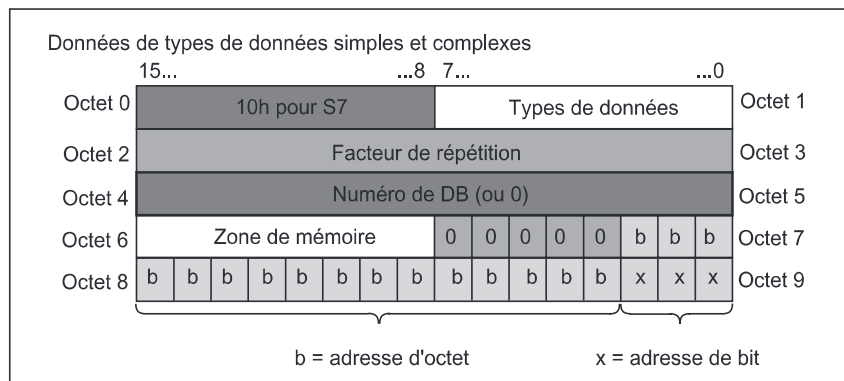
STEP 7 enregistre les données du type de données Paramètre ANY dans 10 octets. Lors de la définition d'un paramètre de type ANY, vous devez veiller à ce que les 10 octets soient tous occupés, car le bloc appelé exploite le contenu entier du paramètre. Si, par exemple, vous spécifiez un numéro de DB dans l'octet 4, vous devez également indiquer de manière explicite la zone de mémoire dans l'octet 6.

STEP 7 gère les types de données simples et complexes différemment que les types de paramètre.

#### Format ANY pour les types de données

Pour les types de données simples et complexes, STEP 7 enregistre les données suivantes :

- types de données,
- facteur de répétition,
- numéro de DB,
- zone de mémoire dans laquelle les informations sont enregistrées,
- adresse de début des données.



Le facteur de répétition désigne une quantité du type de données identifié qui est à transmettre par le type de données Paramètre ANY. Vous pouvez ainsi indiquer une zone de données et également utiliser des tableaux et structures en liaison avec le type de données Paramètre ANY. STEP 7 caractérise les tableaux et structures comme nombre de types de données (à l'aide du facteur de répétition). Pour transmettre 10 mots, par exemple, vous devez entrer la valeur 10 pour le facteur de répétition et la valeur 04 pour le type de données.

L'adresse est enregistrée dans le format octet.bit, l'adresse d'octet étant enregistrée dans les bits 0 à 2 de l'octet 7, dans les bits 0 à 7 de l'octet 8 et dans les bits 3 à 7 de l'octet 9. L'adresse de bit est enregistrée dans les bits 0 à 2 de l'octet 9.

Dans le cas du pointeur zéro de type de données NIL, tous les octets ont la valeur 0 à partir de l'octet 1.



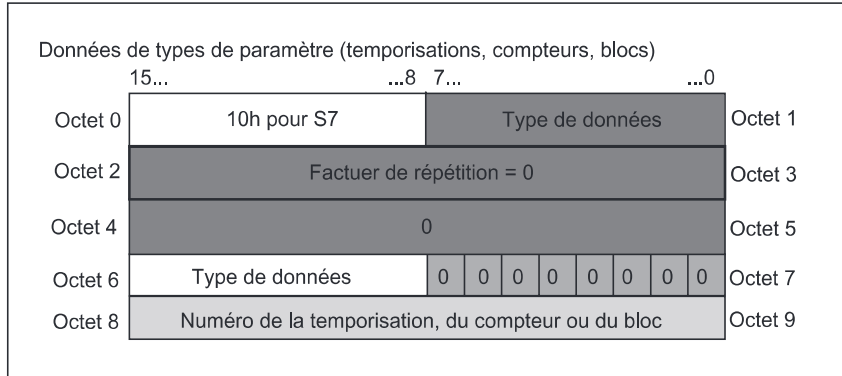
Les tableaux suivants indiquent le codage des types de données ou des zones de mémoire pour le type de données Paramètre ANY.

	<b>Codage des types de données</b>	
<b>Code hexadécimal</b>	<b>Type de données</b>	<b>Description</b>
b#16#00	NIL	Pointeur zéro
b#16#01	BOOL	Bits
b#16#02	BYTE	Octets (8 bits)
b#16#03	CHAR	Caractères (8 bits)
b#16#04	WORD	Mots (16 bits)
b#16#05	INT	Entiers (16 bits)
b#16#06	DWORD	Mots (32 bits)
b#16#07	DINT	Entiers (32 bits)
b#16#08	REAL	Nombres à virgule flottante (32 bits)
b#16#09	DATE	Date
b#16#0A	TIME_OF_DAY (TOD)	Heure
b#16#0B	TIME	Temporisation
b#16#0C	S5TIME	Type de données S5TIME
b#16#0E	DATE_AND_TIME (DT)	Date et heure (64 bits)
b#16#13	STRING	Chaîne de caractères

	<b>Codage des zones de mémoire</b>	
<b>Code hexadécimal</b>	<b>Zone</b>	<b>Description</b>
b#16#81	E	Zone de mémoire des entrées
b#16#82	A	Zone de mémoire des sorties
b#16#83	M	Zone de mémoire des mementos
b#16#84	DB	Bloc de données
b#16#85	DI	Bloc de données d'instance
b#16#86	L	Données locales (pile L)
b#16#87	V	Données locales précédentes

## Format ANY pour les types de paramètre

Pour les types de paramètre, STEP 7 enregistre le type de données et l'adresse des paramètres. Le facteur de répétition est toujours égal à 1. Les octets 4, 5 et 7 sont toujours à 0. Les octets 8 et 9 indiquent le numéro de la temporisation, du compteur ou du bloc.



Le tableau suivant indique le codage des types de données pour le type de données Paramètre ANY avec les types de paramètre.

Code hexadécimal	Type de données	Description
b#16#17	BLOCK_FB	Numéro du FB
b#16#18	BLOCK_FC	Numéro de la FC
b#16#19	BLOCK_DB	Numéro du DB
b#16#1A	BLOCK_SDB	Numéro du SDB
b#16#1C	COUNTER	Numéro du compteur
b#16#1D	TIMER	Numéro de la temporisation

### A.3.4.6 Utilisation du type de données Paramètre ANY

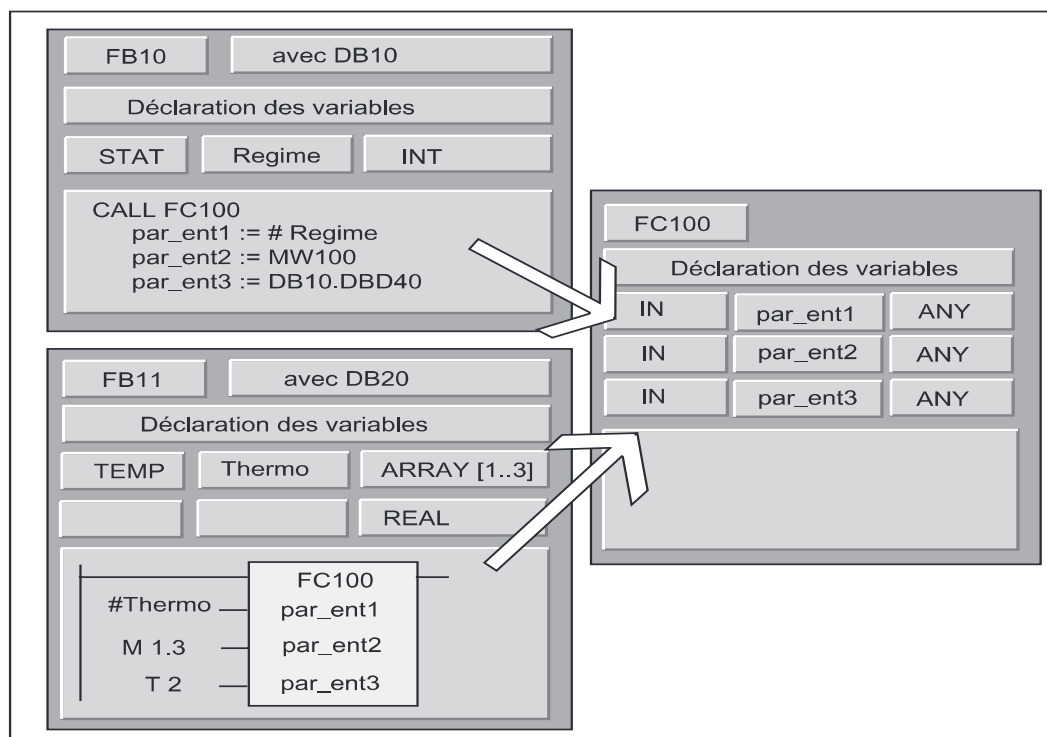
Vous pouvez définir, pour un bloc, des paramètres formels acceptant des paramètres effectifs de n'importe quel type de données. Cela s'avère surtout utile lorsque le type de données du paramètre effectif fourni lors de l'appel du bloc est inconnu ou peut varier (et lorsque tout type de données est acceptable). Dans la déclaration des variables du bloc, vous déclarez le paramètre comme type de données ANY. Vous pourrez ainsi lui affecter un paramètre effectif d'un type de données quelconque dans STEP 7.

STEP 7 réserve 80 bits de mémoire à une variable de type ANY. Lorsque vous affectez un paramètre effectif à un tel paramètre formel, STEP 7 code l'adresse de départ, le type de données et la longueur du paramètre effectif dans ces 80 bits. Le bloc appelé analysera ces 80 bits de données sauvegardées pour le paramètre ANY afin d'obtenir les renseignements nécessaires pour le traitement supplémentaire.

#### Transmission d'un paramètre effectif à un paramètre ANY

En déclarant un paramètre formel de type de données ANY, vous pouvez lui affecter un paramètre effectif de n'importe quel type de données. Vous pouvez indiquer des paramètres effectifs de types de données suivants dans STEP 7 :

- Types de données simples : vous indiquez l'adresse absolue ou le mnémonique du paramètre effectif.
- Types de données complexes : vous entrez le mnémonique correspondant (par exemple, tableaux ou structures).
- Temporisations, compteurs et blocs : vous précisez leur numéro (par exemple, T1, Z20 ou FB6).
- La figure ci-après montre comment transmettre des données à une fonction avec des paramètres de type ANY.



Dans cet exemple de FC100, il s'agit des trois paramètres: *par\_ent1*, *par\_ent2* et *par\_ent3*.

- Lorsque le bloc fonctionnel FB10 appelle la fonction FC100, il transmet un nombre entier (variable statique "Regime"), un mot (MW100) et un double mot du DB10 (DB10.DBD40).
- Lorsque le bloc fonctionnel FB11 appelle la fonction FC100, il transmet un tableau de nombres réels (variable temporaire "Thermo"), une valeur booléenne (M 1.3) et une temporisation (T2).

### Indication d'une zone de données pour un paramètre ANY

Vous pouvez non seulement affecter des opérands individuels à un paramètre ANY (par exemple MW100), mais également indiquer une zone de données. Vous devez utiliser, à cet effet, la notation de constante ci-après pour identifier le volume de données à transmettre :

*p# code-zone octet.bit type-données facteur-répétition*

Vous pouvez indiquer, en notation de constante pour l'élément *type-données*, tous les types de données simples ainsi que le type de données DATE\_AND\_TIME. Hormis pour le type de données BOOL, il faut préciser l'adresse de bit 0 (x.0). Le tableau ci-après présente des exemples de notation constante pour indiquer les zones de mémoire à transmettre à un paramètre ANY.

Paramètres effectifs	Description
p# M 50.0 BYTE 10	Correspond à 10 octets dans la zone de mémoire "Mémentos": de MB50 à MB59
p# DB10.DBX5.0 S5TIME 3	Correspond à 3 unités de données de type S5TIME contenues dans le DB10 : de DB octet 5 à DB octet 10
p# A 10.0 BOOL 4	Correspond à 4 bits dans la zone de mémoire "Sorties" : de A 10.0 à A 10.3

## Exemple d'utilisation du type de données Paramètre ANY

L'exemple suivant montre comment vous pouvez copier une zone de mémoire de 10 octets en utilisant le type de données Paramètre ANY et la fonction système SFC 20 BLKMOV.

LIST	Signification
FUNCTION FC 10: VOID	
VAR_TEMP	
Source : ANY;	
Destination :	
ANY;	
END_VAR	
BEGIN	Charger l'adresse de début du pointeur ANY dans AR1.
LAR1 P#Source;	
L B#16#10;	Charger l'ID de syntaxe et
T LB[AR1,P#0.0];	la transférer dans le pointeur ANY.
L B#16#02;	Charger le type de données octet et
T LB[AR1,P#1.0];	Transférer dans le pointeur ANY.
L 10;	Charger 10 octets et
T LW[AR1,P#2.0];	les transférer dans le pointeur ANY.
L 22;	La source correspond au DB22, DBB11
T LW[AR1,P#4.0];	
L P#DBX11.0;	
T LD[AR1,P#6.0];	
LAR1 P#Destination;	Charger l'adresse de début du pointeur ANY dans AR1.
L B#16#10;	Charger l'ID de syntaxe et
T LB[AR1,P#0.0];	la transférer dans le pointeur ANY.
L B#16#02;	Charger le type de données octet et
T LB[AR1,P#1.0];	Transférer dans le pointeur ANY.
L 10;	Charger 10 octets et
T LW[AR1,P#2.0];	les transférer dans le pointeur ANY.
L 33;	La destination correspond au DB33, DBB202
T LW[AR1,P#4.0];	
L P#DBX202.0;	
T LD[AR1,P#6.0];	
CALL SFC 20 (	Appel de la fonction système Blockmove
SRCBLK := Source,	
RET_VAL := MW 12,	Exploitation du bit RB et du MW 12
DSTBLK :=	
Destination	
);	
END_FUNCTION	




### A.3.4.7 Affectation de types de données aux données locales de blocs de code

STEP 7 limite les types de données - simples, complexes et paramètres - pouvant être affectés aux données locales d'un bloc dans la déclaration des variables.

#### Types autorisés pour les données locales d'un OB

Le tableau ci-près présente les restrictions valables pour les données locales des blocs d'organisation OB. Un OB ne pouvant être appelé, il ne peut pas avoir de paramètres (entrée, sortie ou entrée/sortie). Un OB n'ayant pas de DB d'instance, vous ne pouvez pas déclarer de variables statiques. Les variables temporaires d'un OB peuvent être de type de données simple, complexe ou ANY.

Les affectations valides sont indiquées par le symbole .



















Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée	—	—	—	—	—	—	—
Sortie	—	—	—	—	—	—	—
Entrée/sortie	—	—	—	—	—	—	—
Statique	—	—	—	—	—	—	—
Temporaire	 (1)	 (1)	—	—	—	—	 (1)

(1) Méorisé dans la pile L de l'OB

#### Types autorisés pour les données locales d'un FB

Le tableau ci-près présente les restrictions valables pour les données locales des blocs fonctionnels FB. Les FB disposant d'un DB d'instance, ces restrictions sont moindres. S'il n'y en a pas pour la déclaration de paramètres d'entrée, vous ne pouvez toutefois déclarer aucun type de données Paramètre pour les paramètres de sortie et seuls les types de paramètre POINTER et ANY sont autorisés pour les paramètres d'entrée/sortie. Vous pouvez déclarer des variables temporaires de type ANY, tous les autres types de paramètre étant interdits.


Les affectations valides sont indiquées par le symbole .



















Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée							
Sortie			—	—	—	—	—
Entrée/sortie		 (1)(3)	—	—	—		
Statique			—	—	—	—	—
Temporaire	 (2)	 (2)	—	—	—	—	 (2)

(1) Méorisé dans le DB d'instance comme renvoi (pointeur 48 bits)  
(2) Méorisé dans la pile L du FB  
(3) Les types STRING peuvent être définis seulement dans la longueur standard.

## Types autorisés pour les données locales d'une FC

Le tableau ci-près présente les restrictions valables pour les données locales des fonctions FC. Une FC n'ayant pas de DB d'instance, vous ne pouvez pas déclarer de variables statiques. Les types de paramètre POINTER et ANY sont valables pour les paramètres d'entrée, de sortie et d'entrée/sortie. Vous pouvez également déclarer des variables temporaires de type de données Paramètre ANY.

Les affectations valides sont indiquées par le symbole .

Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée		 (2)					
Sortie		 (2)	—	—	—		
Entrée/sortie		 (2)	—	—	—		
Temporaire	 (1)	 (1)	—	—	—	—	 (1)
(1) Méorisé dans la pile L de la FC							
(2) Les types STRING peuvent être définis seulement dans la longueur standard.							











### A.3.4.8 Types de données autorisés pour la transmission de paramètres







#### Règles pour la transmission de paramètres entre blocs

Lorsque vous affectez des paramètres effectifs à des paramètres formels, vous pouvez indiquer soit une adresse absolue, soit un mnémonique, soit une constante. Ces différents types d'affectation ne sont pas autorisés pour tous les paramètres dans STEP 7. Il est, par exemple, interdit d'affecter des valeurs constantes à des paramètres de sortie ou d'entrée/sortie puisque de tels paramètres doivent, par définition, changer de valeur. Ces restrictions concernent surtout les paramètres de type de données complexe auxquels on ne peut affecter ni adresse absolue, ni constante.

Les tableaux ci-après récapitulent les restrictions relatives aux types de données de paramètres effectifs affectés à des paramètres formels.

Les affectations valides sont indiquées par le symbole .

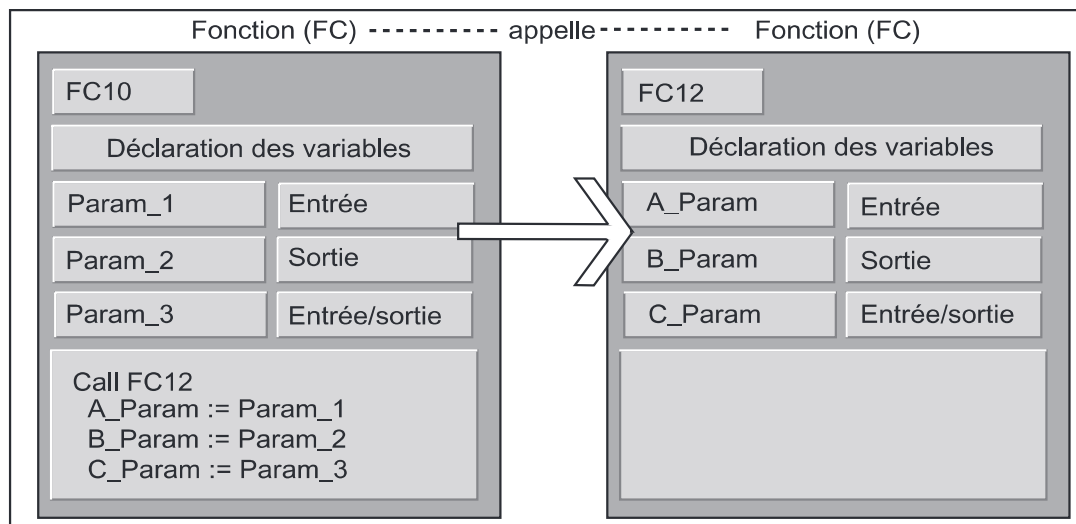
Types de données simples				
Type de déclaration	Adresse absolue	Mnémonique (dans table mnémo.)	Mnémonique bloc loc.	Constante
Entrée				
Sortie				—
Entrée/sortie				—

Types de données complexes				
Type de déclaration	Adresse absolue	Mnémonique de l'élément du DB (dans table mnémo.)	Mnémonique bloc loc.	Constante
Entrée	—			—
Sortie	—			—
Entrée/sortie	—			—



## Types de données autorisés pour l'appel d'une FC par une autre FC

Vous pouvez affecter les paramètres formels d'une FC appelante aux paramètres formels de la FC appelée. La figure ci-après montre les paramètres formels de la FC10 qui sont affectés en tant que paramètres effectifs aux paramètres formels de la FC12.



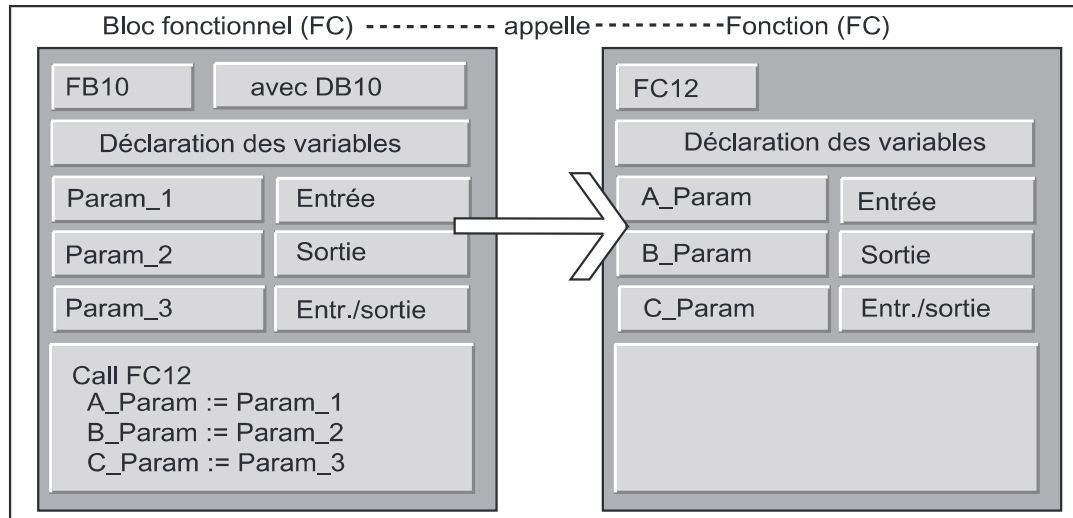
STEP 7 impose toutefois des restrictions dans ce domaine. Ainsi, vous ne pouvez pas affecter en tant que paramètres effectifs des paramètres de type de données complexe ou de type de données Paramètre.

Le tableau ci-après montre les types de données autorisés (☐) lorsqu'une fonction appelle une autre fonction.

Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée → Entrée	☐	—	—	—	—	—	—
Entrée → Sortie	—	—	—	—	—	—	—
Entrée → Entrée/sortie	—	—	—	—	—	—	—
Sortie → Entrée	—	—	—	—	—	—	—
Sortie → Sortie	☐	—	—	—	—	—	—
Sortie → Entrée/sortie	—	—	—	—	—	—	—
Entrée/sortie → Entrée	☐	—	—	—	—	—	—
Entrée/sortie → Sortie	☐	—	—	—	—	—	—
Entrée/sortie → Entrée/sortie	☐	—	—	—	—	—	—

## Types de données autorisés pour l'appel d'une FC par un FB

Vous pouvez affecter les paramètres formels d'un FB appelant aux paramètres formels de la FC appelée. La figure ci-après montre les paramètres formels du FB10 qui sont affectés en tant que paramètres effectifs aux paramètres formels de la FC12.

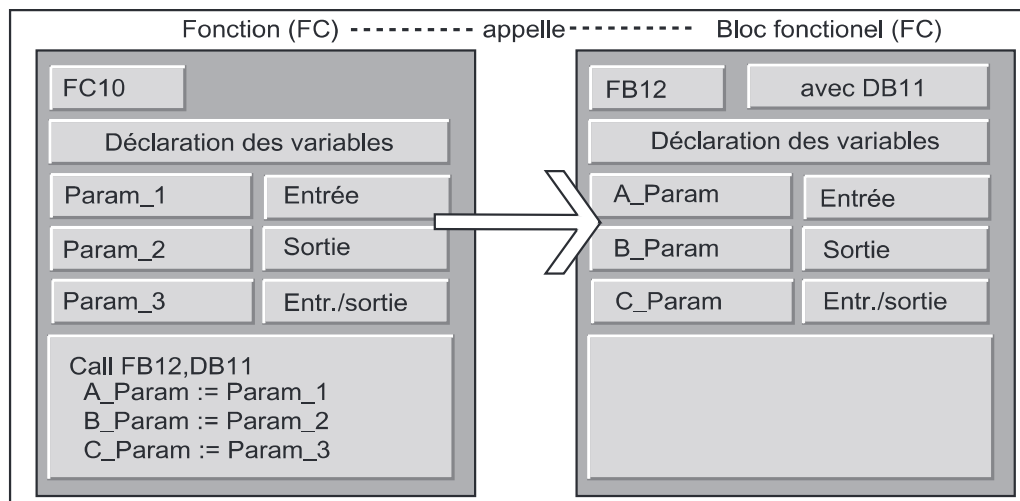


STEP 7 impose toutefois des restrictions dans ce domaine. Ainsi, vous ne pouvez pas affecter en tant que paramètres effectifs des paramètres de type de données paramètre. Le tableau ci-après montre les types de données autorisés (E) lorsque qu'un bloc fonctionnel appelle une fonction.

Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			<b>TIMER</b>	<b>COUNTER</b>	<b>BLOCK</b>	<b>POINTER</b>	<b>ANY</b>
Entrée → Entrée	(E)	(E)	—	—	—	—	—
Entrée → Sortie	—	—	—	—	—	—	—
Entrée → Entrée/sortie	—	—	—	—	—	—	—
Sortie → Entrée	—	—	—	—	—	—	—
Sortie → Sortie	(E)	(E)	—	—	—	—	—
Sortie → Entrée/sortie	—	—	—	—	—	—	—
Entrée/sortie → Entrée	(E)	—	—	—	—	—	—
Entrée/sortie → Sortie	(E)	—	—	—	—	—	—
Entrée/sortie → Entrée/sortie	(E)	—	—	—	—	—	—

## Types de données autorisés pour l'appel d'un FB par une FC

Vous pouvez affecter les paramètres formels d'une FC appelante aux paramètres formels du FB appelé. La figure ci-après montre les paramètres formels de la FC10 qui sont affectés en tant que paramètres effectifs aux paramètres formels du FB12.



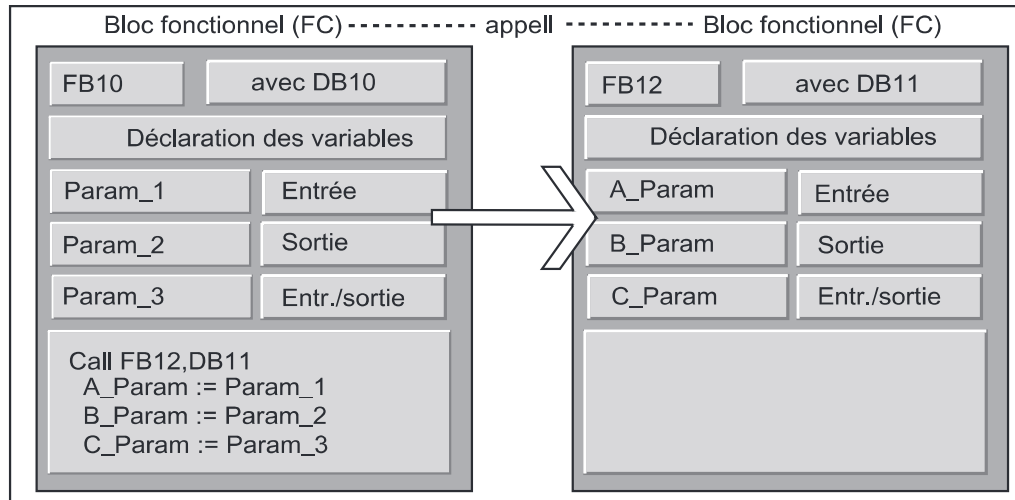
STEP 7 impose toutefois des restrictions dans ce domaine. Ainsi, vous ne pouvez pas affecter en tant que paramètres effectifs des paramètres de type de données complexe, Mais vous pouvez affecter des paramètres d'entrée de type de données Paramètre TIMER, COUNTER et BLOCK aux paramètres d'entrée du FB appelé.

Le tableau ci-après montre les types de données autorisés (EP) lorsqu'une fonction appelle un bloc fonctionnel.


Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée → Entrée	EP	—	EP	EP	EP	—	—
Entrée → Sortie	—	—	—	—	—	—	—
Entrée → Entrée/sortie	—	—	—	—	—	—	—
Sortie → Entrée	—	—	—	—	—	—	—
Sortie → Sortie	EP	—	—	—	—	—	—
Sortie → Entrée/sortie	—	—	—	—	—	—	—
Entrée/sortie → Entrée	EP	—	—	—	—	—	—
Entrée/sortie → Sortie	EP	—	—	—	—	—	—
Entrée/sortie → Entrée/sortie	EP	—	—	—	—	—	—











## Types de données autorisés pour l'appel d'un FB par un autre FB

Vous pouvez affecter les paramètres formels d'un FB appelant aux paramètres formels du FB appelé. La figure ci-après montre les paramètres formels du FB10 qui sont affectés en tant que paramètres effectifs aux paramètres formels du FB12.



STEP 7 impose toutefois des restrictions dans ce domaine. Ainsi, vous ne pouvez pas affecter, en tant que paramètres effectifs, des paramètres d'entrée et de sortie de type de données complexe aux paramètres d'entrée et de sortie du FB appelé. Mais vous pouvez affecter des paramètres d'entrée de type de données Paramètre TIMER, COUNTER et BLOCK aux paramètres d'entrée du FB appelé.

Le tableau ci-après montre les types de données autorisés (  ) lorsqu'un bloc fonctionnel appelle un autre bloc fonctionnel.

Type de déclaration	Types de données simples	Types de données complexes	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre	Type de données Paramètre
			TIMER	COUNTER	BLOCK	POINTER	ANY
Entrée → Entrée						—	—
Entrée → Sortie	—	—	—	—	—	—	—
Entrée → Entrée/sortie	—	—	—	—	—	—	—
Sortie → Entrée	—	—	—	—	—	—	—
Sortie → Sortie			—	—	—	—	—
Sortie → Entrée/sortie	—	—	—	—	—	—	—
Entrée/sortie → Entrée		—	—	—	—	—	—
Entrée/sortie → Sortie		—	—	—	—	—	—
Entrée/sortie → Entrée/sortie		—	—	—	—	—	—

#### **A.3.4.9 Transmission au paramètre IN\_OUT d'un FB**

Pour la transmission de types de données complexes au paramètre IN\_OUT d'un bloc fonctionnel (FB), c'est l'adresse d'opérande de la variable qui est transmise (call by reference).

Pour la transmission de types de données simples au paramètre IN\_OUT d'un FB, les valeurs sont copiées dans le bloc de données d'instance avant l'exécution du FB et extraites du bloc de données d'instance lorsque l'exécution du FB est terminée.

Ainsi, les variables IN\_OUT de type de données simple peuvent être initialisées avec une valeur.

Dans un appel, il n'est toutefois pas possible d'indiquer une constante comme paramètre effectif à la place d'une variable IN\_OUT, car une constante ne peut être écrasée.

Les variables de type de données STRUCT ou ARRAY ne peuvent pas être initialisées, car dans ce cas le bloc de données d'instance ne contient qu'une seule adresse.

## A.4 Utilisation d'anciens projets

### A.4.1 Conversion d'un ancien projet de version 1

Vous avez la possibilité de réutiliser des projets que vous avez créés avec la version 1 de STEP 7. Vous convertissez à cet effet des projets de version 1 en projets de version 2.

Les composants suivants d'un ancien projet de version 1 sont conservées :

- la structure du projet avec les programmes,
- les blocs,
- les fichiers source LIST,
- la table des mnémoniques.

La configuration matérielle n'est pas convertie. Vous pouvez copier les éléments de programmes dans d'autres projets. Vous pouvez également compléter le nouveau projet avec une station, que vous devez configurer et paramétrer en conséquence.

Après avoir converti un projet dans la version 2, vous pouvez indiquer dans une boîte de dialogue si vous souhaitez ensuite le convertir dans la version actuelle de STEP 7.

---

#### Nota

Les propriétés d'un bloc demeurent celles d'un bloc de version 1. Le code généré dans la version 1 n'est pas modifié et les blocs ne peuvent donc pas être utilisés avec des multi-instances.

Si vous souhaitez déclarer des multi-instances dans des blocs convertis, vous devez d'abord générer des sources LIST à partir de ces blocs, en utilisant l'application "CONT/LIST : programmation de blocs". Vous pourrez ensuite à nouveau compiler ces sources en blocs.

La programmation de multi-instances est une nouveauté pour créer des blocs fonctionnels (FB) dans la version 2 de STEP 7. Si vous souhaitez que les blocs fonctionnels créés dans la version 1 conservent la même fonction dans la version 2 du projet, il n'est pas nécessaire de les convertir.

---

### Marche à suivre

Pour convertir des projets de la version 1, procédez de la manière suivante :

1. Choisissez la commande **Fichier > Ouvrir un ancien projet de version 1**.
2. Dans la boîte de dialogue qui s'ouvre, sélectionnez le projet de version 1 que vous souhaitez réutiliser. Un projet de version 1 est caractérisé par son extension de fichier \*.s7a.
3. Tapez le nom du nouveau projet dans lequel vous souhaitez convertir l'ancien projet de version 1.

## A.4.2 Conversion d'un ancien projet de version 2

La commande **Fichier > Ouvrir** de STEP 7 vous permet également d'ouvrir des projets de version 2.

Vous pouvez convertir des projets/bibliothèques de version 2 dans la version actuelle de STEP 7 (migration) en choisissant la commande **Fichier > Enregistrer sous** et en sélectionnant le paramètre "avec réorganisation". Le projet est alors enregistré comme projet de version actuelle de STEP 7.

Pour éditer et enregistrer des projets et bibliothèques d'anciennes versions de STEP 7 tout en conservant leur format, sélectionnez le type de fichier de l'ancienne version de STEP 7 dans la boîte de dialogue "Enregistrer le projet sous". Pour éditer les objets avec la version 2.1 de STEP 7, sélectionnez ici le projet 2.x ou la bibliothèque 2.x (à partir de la version 5.1, il n'est plus possible d'enregistrer sous forme de version 2, voir aussi Edition de projet et de bibliothèques de version 2).

### Désignations du type de fichier

	STEP 7 V3	à partir de STEP 7 V4
Type de fichier de la version actuelle	Projet3.x Bibliothèque3.x	Projet Bibliothèque
Type de fichier de la version précédente	Projet2.x Bibliothèque2.x	Projet2.x Bibliothèque2.x

Dans ce cas, vous ne disposez cependant que des fonctions de l'ancienne version. Les projets et bibliothèques pourront toujours être édités avec cette ancienne version de STEP 7.

#### Nota

Dans la conversion de la version 3 à la version 4, seule la désignation a été modifiée, le format étant resté identique. C'est la raison pour laquelle il n'existe pas de type de fichier Projet3.x dans STEP 7 V4.

### Marche à suivre

Pour convertir des projets de version 2 dans le format de la version actuelle de STEP 7, procédez de la manière suivante :

1. Exécutez la fonction "Enregistrer sous (menu Fichier)" avec réorganisation.
2. Sélectionnez le type de fichier "Projet" dans la boîte de dialogue "Enregistrer le projet sous" et cliquez sur le bouton "Enregistrer".

Pour convertir des projets de version 2, tout en conservant leur format, dans la version actuelle de STEP 7, procédez de la manière suivante :

1. Le cas échéant, procédez à l'étape 1, comme décrit ci-avant.
2. Dans la boîte de dialogue "Enregistrer le projet sous", sélectionnez le type de fichier correspondant à l'ancienne version de STEP 7 et cliquez sur le bouton "Enregistrer".

### A.4.3 Remarque sur les projets STEP 7 de version V2.1 avec communication par données globales

- Lorsque vous souhaitez convertir un projet avec communication par données globales de STEP 7 V2.1 dans STEP 7 V5, vous devez préalablement ouvrir la table des données globales depuis STEP 7 V5.0 dans le projet STEP 7 V2.1. Les données de communication déjà configurées seront ainsi automatiquement converties dans la nouvelle structure par la communication par données globales.
- Lors de l'archivage de projets STEP 7 V2.1, vous pouvez obtenir un message d'erreur émis par un ancien programme de compression (ARJ, PKZIP...), si le projet contient des fichiers dont le nom comporte plus de huit caractères. Ce message s'affiche également lorsque le réseau MPI a été édité avec une désignation supérieure à 8 caractères dans le projet STEP 7 V2.1. Avant de débiter pour la première fois la configuration de la communication par données globales, éditez un nom de 8 caractères au maximum pour le réseau MPI dans les projets STEP 7 V2.1 avec données globales.
- Si vous souhaitez renommer un projet STEP 7 V2.1, vous devez réaffecter les titres des colonnes (CPU) dans la table des données globales en sélectionnant une nouvelle fois la CPU correspondante. Si vous restaurez l'ancien nom de projet, vous obtiendrez les affectations correspondantes.

### A.4.4 Esclaves DP avec fichiers GSD manquants ou erronés

Lorsque vous utilisez d'anciennes configurations de stations dans la version 5.1 de STEP 7, il peut arriver, dans de rares cas, que le fichier GSD d'un esclave DP manque ou ne peut pas être compilé (p. ex. en raison d'erreurs de syntaxe dans le fichier GSD).

Dans ce cas, STEP 7 crée un esclave "Dummy" qui représente l'esclave configuré, par exemple après un chargement de station dans la PG ou après ouverture et édition d'un ancien projet. Vous ne pouvez éditer cet esclave "Dummy" qu'avec de fortes restrictions ; vous ne pouvez modifier ni sa configuration (identification DP), ni ses paramètres. Un nouveau chargement dans la station est cependant possible, la configuration initiale de l'esclave restant conservée. Vous ne pouvez pas non plus effacer l'esclave DP complet.

### Reconfiguration et reparamétrage de l'esclave DP

Si vous souhaitez reconfigurer ou reparamétrer l'esclave DP, vous devez demander un fichier GSD actuel auprès du fabricant de cet esclave DP, puis l'installer via la commande **Outils > Installer les fichiers GSD**.

Lorsque le fichier GSD correct est installé, il permet de représenter l'esclave DP. Celui-ci conserve ses données et peut à nouveau être utilisé.



## A.5 Exemples de programmes

### A.5.1 Exemples de projets et de programmes

Le CD d'installation de STEP 7 contient plusieurs exemples de projets utiles, dont la liste figure dans le tableau ci-après. Une fois installés, les exemples de projets figurent dans la boîte de dialogue "Ouvrir" de SIMATIC Manager (onglet "Exemples de projets"). D'autres exemples de projets peuvent venir les compléter lorsque vous installez des logiciels optionnels. Le contenu de ces exemples de projets est précisé dans la documentation du logiciel optionnel.

Exemples et exemples de projets	Contenus dans le CD	Décrits dans le présent chapitre	Description all/engl. dans l'OB1
Projets "ZFr01_01_STEP7_*" à "ZFr01_06_STEP7_*(Projet exemples Getting Started)	•	Manuel distinct	•
Projet "ZFr01_11_STEP7_P_dec" (exemple d'une configuration PROFIBUS DP)	•	-	-
Projet "ZFr01_08_STEP7_Mixeur" (processus de mélange industriel)	•	•	
Projet "ZFr01_09_STEP7_Feux" (commande de feux à un passage pour piétons)	•		•
Projet "ZFr01_10_STEP7_COM_SFB" (échange de données entre deux CPU S7-400)	•		•
Projet "ZXX01_14_HSystem_S7400H" (projet d'entrée pour systèmes à haute disponibilité)	•	Manuel distinct	•
Projet ZXX01_15_HSystem_RED_IO (projet d'entrée pour systèmes à haute disponibilité avec périphérie redondante)	•	Manuel distinct	•
Projets "ZFr01_11_STEP7_COM_SFC1" et "ZFr01_12_STEP7_COM_SFC2" (échange de données par SFC de communication pour des liaisons non configurées)	•		•
Projet "ZFr01_13_STEP7_PID-Temp" (exemple de régulation de température au moyen des FB 58 et FB 59)	•		•
Exemple d'utilisation d'alarmes horaires		•	
Exemple d'utilisation d'alarmes temporisées		•	
Exemple de masquage et de démasquage d'événements d'erreurs synchrones		•	
Exemple d'inhibition et de validation d'événements d'alarme et d'événements asynchrones		•	
Exemple de traitement différé d'événements d'alarme et d'événements asynchrones		•	

Dans les exemples, il ne s'agit pas tant de montrer un style de programmation ou une compétence technique dans la commande d'un processus particulier, mais bien plutôt de réaliser quelles étapes doivent être exécutées lors de la conception du programme.

## **Suppression, copie et installation d'exemples de projets**

Dans SIMATIC Manager, vous pouvez supprimer, puis à nouveau réinstaller les exemples de projets fournis. Pour l'installation, vous démarrez le programme Setup de STEP 7. Vous pouvez sélectionner les exemples de projets à réinstaller. Les copies des exemples de projets fournis créés par "Enregistrer sous" et les exemples de projets que vous créez vous-même peuvent exclusivement être enregistrés sous forme de projet utilisateur.

---

### **Nota**

Lors d'une installation de STEP 7, les exemples de projets fournis sont copiés, à moins qu'ils soient désélectionnés. Si vous avez modifié des exemples de projets fournis, ils seront remplacés par les originaux lors d'une nouvelle installation de STEP 7.

C'est la raison pour laquelle, il est recommandé de copier les exemples de projets fournis avant de les modifier et d'éditer uniquement la copie.

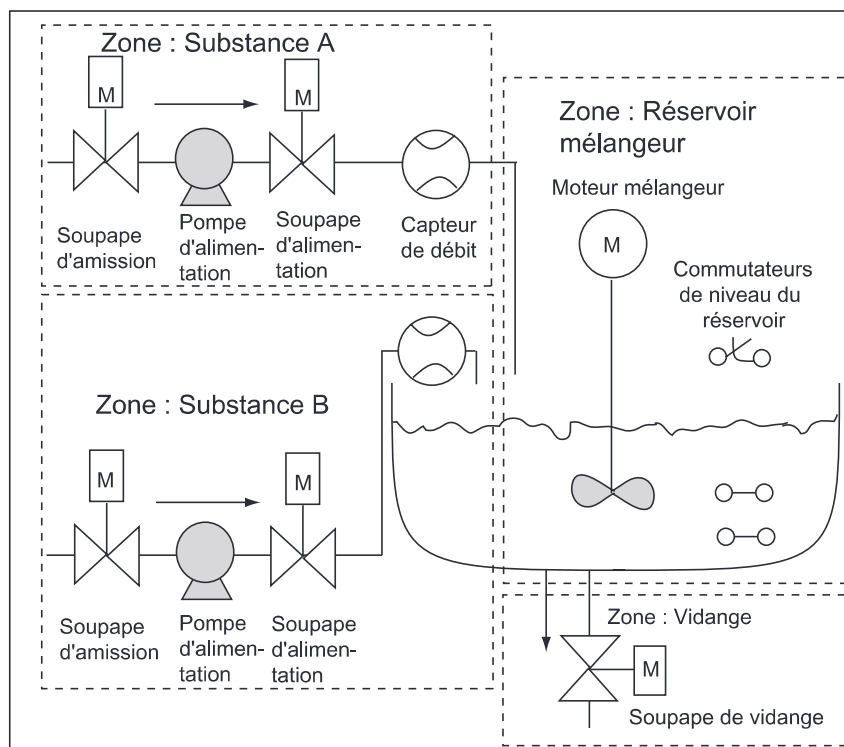
---

## A.5.2 Exemple de programme pour un processus de mélange industriel

Notre exemple de programme se base sur les informations de la première partie du manuel relatives à la commande d'un processus de mélange industriel.

### Problème posé

Deux substances (A et B) doivent être mélangées par un moteur mélangeur dans un réservoir. Cette masse doit ensuite s'écouler du réservoir par une soupape de vidange. La figure ci-après montre un diagramme de notre exemple de processus.



## Description des processus partiels

Nous vous avons expliqué dans la première partie du manuel, comment subdiviser l'exemple de processus en zones fonctionnelles et en différentes tâches. Voici la description des différentes zones.

### Zones pour substances A et B

- Les conduites d'amenée des substances doivent comporter une soupape d'admission, une soupape d'alimentation ainsi qu'une pompe d'alimentation.
- Dans ces conduites se trouvent des capteurs de débit.
- La mise en marche des pompes d'alimentation doit être inhibée lorsque le capteur de niveau indique "Réservoir plein".
- La mise en marche des pompes d'alimentation doit être inhibée lorsque la soupape de vidange est ouverte.
- Les soupapes d'admission et d'alimentation doivent être ouvertes au plus tôt 1 seconde après le déclenchement de la pompe d'alimentation.
- Les soupapes doivent être fermées immédiatement après l'arrêt des pompes d'alimentation (signal du capteur de débit) afin d'éviter un écoulement de la substance en provenance de la pompe.
- Le déclenchement des pompes est surveillé par une temporisation : le capteur de débit doit signaler un débit 7 secondes au maximum après ce déclenchement.
- Les pompes d'alimentation doivent être arrêtées le plus rapidement possible lorsque les capteurs de débit ne signalent plus de débit pendant le fonctionnement des pompes.
- Le nombre de démarrages des pompes d'alimentation doit être comptabilisé (période de maintenance).

### Zone Réservoir de mélange

- Le déclenchement du moteur mélangeur doit être verrouillé lorsque le capteur de niveau indique "Réservoir en dessous du minimum" ou lorsque la soupape de vidange est ouverte.
- Le moteur mélangeur émet un signal en retour une fois le régime nominal atteint. S'il n'émet pas ce signal 10 secondes au maximum après l'activation du moteur, il faut l'arrêter.
- Le nombre de démarrages du moteur mélangeur doit être comptabilisé (période de maintenance).
- Le réservoir de mélange doit comporter trois capteurs :
  - Réservoir plein : contact à ouverture. Lorsque le niveau maximal est atteint, le contact est ouvert.
  - Niveau dans le réservoir en dessous du minimum : contact à fermeture. Lorsque le niveau minimal est atteint, le contact est fermé.
  - Réservoir pas vide : contact à fermeture. Le contact est fermé si le réservoir n'est pas vide.

### **Zone Vidange**

- La vidange doit être commandée par soupape magnétique.
- La soupape magnétique est commandée par l'opérateur, mais doit être refermée au plus tard lors du signal "Réservoir vide".
- L'ouverture de la soupape de vidange est verrouillée
  - lorsque le moteur mélangeur fonctionne ;
  - lorsque le réservoir est vide.

### **Poste d'opération**

Il faut également installer un poste d'opération pour que l'opérateur puisse démarrer et arrêter ainsi que surveiller le processus. Ce poste d'opération comporte :

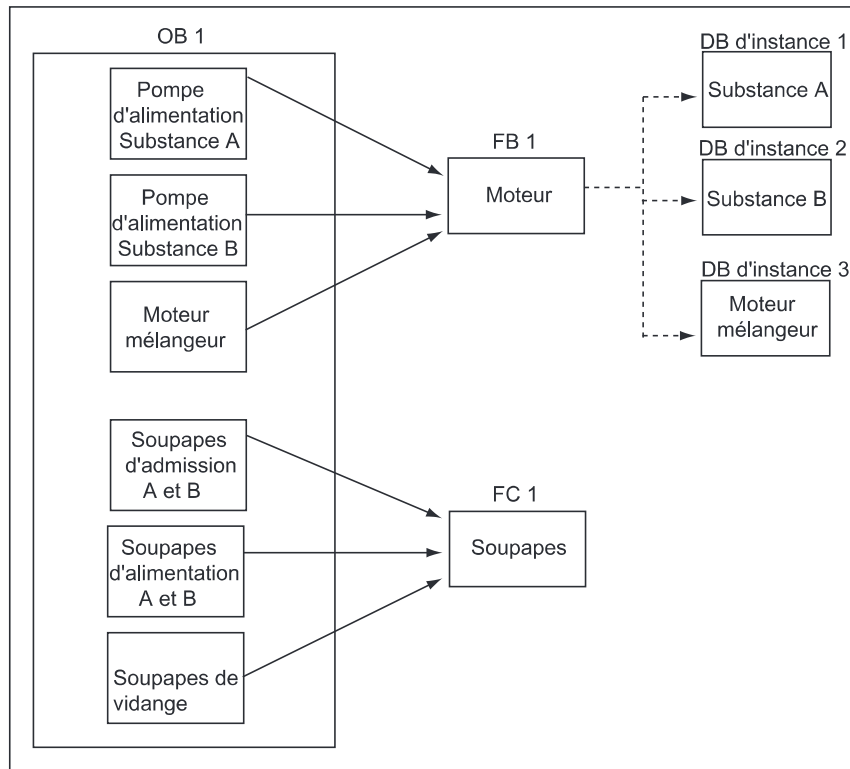
- des commutateurs pour commander les événements les plus importants Le bouton "Mettre à 0 indicateur de maintenance" permet d'éteindre les lampes de signalisation de maintenance pour les moteurs ayant besoin d'une maintenance et de mettre à zéro les valeurs correspondantes des compteurs pour l'intervalle entre les maintenances ;
- des lampes de signalisation indiquant l'état de fonctionnement,
- le commutateur d'arrêt d'urgence.

### A.5.2.1 Définition de blocs de code

Vous définissez la structure de votre programme utilisateur en le répartissant dans différents blocs et en fixant la hiérarchie d'appel de ces blocs.

#### Hiérarchie d'appel des blocs

La figure ci-après présente la hiérarchie des blocs devant être appelés dans le programme structuré.



- OB1 : il s'agit de l'interface avec le système d'exploitation de la CPU ; il contient le programme principal. Le bloc fonctionnel FB1 et la fonction FC1 sont appelés et les paramètres spécifiques nécessaires pour la commande du processus sont transmis dans l'OB1.
- FB1 : la pompe d'alimentation pour la substance A, la pompe d'alimentation pour la substance B et le moteur mélangeur peuvent être commandés par un même bloc fonctionnel, puisque les tâches sont identiques (activation, désactivation, comptage des interventions, etc.).
- DB d'instance 1-3 : les paramètres effectifs et les données statiques pour la commande des pompes d'alimentation pour les substances A et B ainsi que pour celle du moteur mélangeur sont différents et sont donc inscrits dans trois DB d'instance affectés au FB1.
- FC1 : les soupapes d'admission et d'alimentation pour les substances A et B ainsi que la soupape de vidange utilisent également un bloc de code commun. Puisqu'il s'agit uniquement de programmer la fonction d'ouverture et de fermeture, une seule fonction suffit.

## A.5.2.2 Affectation de mnémoniques

### Définition de mnémoniques

Notre exemple de programme utilise des mnémoniques (ou noms symboliques) définis dans la table des mnémoniques avec STEP 7. Les tableaux ci-après présentent les mnémoniques et les adresses absolues correspondantes pour les éléments du programme utilisés.

<b>Mnémoniques pour les pompes d'alimentation et le moteur mélangeur</b>			
<b>Mnémonique</b>	<b>Opérande</b>	<b>Type de données</b>	<b>Description</b>
Feed_pump_A_start	E 0.0	BOOL	Commutateur bouton-poussoir de démarrage de la pompe d'alimentation pour substance A
Feed_pump_A_stop	E 0.1	BOOL	Commutateur bouton-poussoir d'arrêt de la pompe d'alimentation pour substance A
Flow_A	E 0.2	BOOL	La substance A coule.
Inlet_valve_A	A 4.0	BOOL	Commande de la soupape d'admission pour substance A
Feed_valve_A	A 4.1	BOOL	Commande de la soupape d'alimentation pour substance A
Feed_pump_A_on	A 4.2	BOOL	Lampe de signalisation "Pompe d'alimentation pour substance A en marche"
Feed_pump_A_off	A 4.3	BOOL	Lampe de signalisation "Pompe d'alimentation pour substance A arrêtée"
Feed_pump_A	A 4.4	BOOL	Commande de la pompe d'alimentation pour substance A
Feed_pump_A_fault	A 4.5	BOOL	Lampe de signalisation "Erreur de la pompe d'alimentation A"
Feed_pump_A_maint	A 4.6	BOOL	Lampe de signalisation "Maintenance de la pompe d'alimentation A"
Feed_pump_B_start	E 0.3	BOOL	Commutateur bouton-poussoir de démarrage de la pompe d'alimentation pour substance B
Feed_pump_B_stop	E 0.4	BOOL	Commutateur bouton-poussoir d'arrêt de la pompe d'alimentation pour substance B
Flow_B	E 0.5	BOOL	La substance B coule.
Inlet_valve_B	A 5.0	BOOL	Commande de la soupape d'admission pour substance B
Feed_valve_B	A 5.1	BOOL	Commande de la soupape d'alimentation pour substance B
Feed_pump_B_on	A 5.2	BOOL	Lampe de signalisation "Pompe d'alimentation pour substance B en marche"
Feed_pump_B_off	A 5.3	BOOL	Lampe de signalisation "Pompe d'alimentation pour substance B arrêtée"
Feed_pump_B	A 5.4	BOOL	Commande de la pompe d'alimentation pour substance B
Feed_pump_B_fault	A 5.5	BOOL	Lampe de signalisation "Erreur de la pompe d'alimentation B"
Feed_pump_B_maint	A 5.6	BOOL	Lampe de signalisation "Maintenance de la pompe d'alimentation B"
Agitator_running	E 1.0	BOOL	Signal en retour du moteur mélangeur

<b>Mnémoniques pour les pompes d'alimentation et le moteur mélangeur</b>			
<b>Mnémonique</b>	<b>Opérande</b>	<b>Type de données</b>	<b>Description</b>
Agitator_start	E 1.1	BOOL	Commutateur bouton-poussoir de démarrage du moteur mélangeur
Agitator_stop	E 1.2	BOOL	Commutateur bouton-poussoir d'arrêt du moteur mélangeur
Agitator	A 8.0	BOOL	Commande du moteur mélangeur
Agitator_on	A 8.1	BOOL	Lampe de signalisation "Moteur mélangeur en marche"
Agitator_off	A 8.2	BOOL	Lampe de signalisation "Moteur mélangeur arrêté"
Agitator_fault	A 8.3	BOOL	Lampe de signalisation "Erreur du moteur mélangeur"
Agitator_maint	A 8.4	BOOL	Lampe de signalisation "Maintenance du moteur mélangeur"

<b>Mnémoniques pour les capteurs et les indicateurs de niveau du réservoir</b>			
<b>Mnémonique</b>	<b>Opérande</b>	<b>Type de données</b>	<b>Description</b>
Tank_below_max	E 1.3	BOOL	Capteur "Réservoir de mélange pas plein"
Tank_above_min	E 1.4	BOOL	Capteur "Réservoir de mélange au-dessus du minimum"
Tank_not_empty	E 1.5	BOOL	Capteur "Réservoir de mélange pas vide"
Tank_max_disp	A 9.0	BOOL	Lampe de signalisation "Réservoir de mélange plein"
Tank_min_disp	A 9.1	BOOL	Lampe de signalisation "Réservoir en dessous du minimum"
Tank_empty_disp	A 9.2	BOOL	Lampe de signalisation "Réservoir de mélange vide"

<b>Mnémoniques pour la soupape de vidange</b>			
<b>Mnémonique</b>	<b>Opérande</b>	<b>Type de données</b>	<b>Description</b>
Drain_open	E 0.6	BOOL	Commutateur bouton-poussoir d'ouverture de la soupape de vidange
Drain_closed	E 0.7	BOOL	Commutateur bouton-poussoir de fermeture de la soupape de vidange
Drain	A 9.5	BOOL	Commande de la soupape de vidange
Drain_open_disp	A 9.6	BOOL	Lampe de signalisation "Soupape de vidange ouverte"
Drain_closed_disp	A 9.7	BOOL	Lampe de signalisation "Soupape de vidange fermée"



<b>Mnémoniques pour les autres éléments du programme</b>			
<b>Mnémonique</b>	<b>Opérande</b>	<b>Type de données</b>	<b>Description</b>
EMER_STOP_off	E 1.6	BOOL	Commutateur d'arrêt d'urgence
Reset_maint	E 1.7	BOOL	Bouton-poussoir de remise à zéro pour les lampes de signalisation de maintenance de tous les moteurs
Motor_block	FB1	FB1	FB pour commander pompes et moteur
Valve_block	FC1	FC1	FC pour commander les soupapes
DB_feed_pump_A	DB1	FB1	DB d'instance pour la commande de la pompe d'alimentation A
DB_feed_pump_B	DB2	FB1	DB d'instance pour la commande de la pompe d'alimentation B
DB_agitator	DB3	FB1	DB d'instance pour la commande du moteur mélangeur

### A.5.2.3 Création du bloc fonctionnel pour le moteur

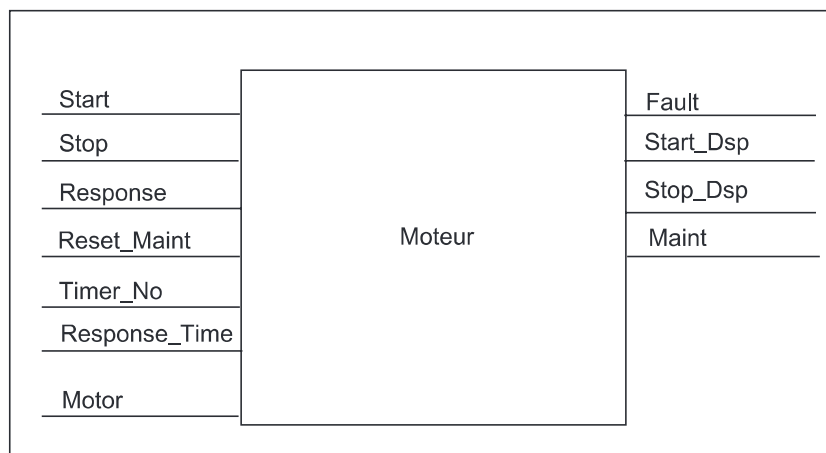
#### Tâches pour le FB

Le FB pour le moteur contient les fonctions logiques ci-après :

- Il existe une entrée de démarrage et une entrée d'arrêt.
- Une série de verrouillages permet le fonctionnement de l'équipement (pompes et moteur mélangeur). L'état des verrouillages est sauvegardé dans les données locales temporaires (pile L) de l'OB1 ("Enable\_Motor") et est combiné aux entrées de démarrage et d'arrêt lors de l'exécution du FB pour le moteur.
- Un signal en retour de l'équipement doit apparaître avant l'expiration d'un temps donné. Sinon, le programme considère qu'une erreur s'est produite et le moteur sera arrêté.
- Il faut définir la temporisation et la valeur de temps pour le cycle signal en retour/erreur.
- Si le bouton de démarrage est actionné et que la validation soit donnée, l'appareil démarre et fonctionne jusqu'à ce que le bouton d'arrêt soit actionné.
- Une temporisation est déclenchée à la mise en marche de l'appareil. L'appareil s'arrête s'il n'émet pas de signal en retour avant que cette temporisation n'expire.

#### Identification des entrées et sorties

La figure ci-après montre les entrées et les sorties du FB générique pour le moteur.



## Définition des paramètres pour le FB

Vous devez définir des noms de paramètres génériques pour les entrées et les sorties afin de créer un FB "Moteur" réutilisable, permettant de commander les deux pompes et le moteur mélangeur.

Dans l'exemple du processus, le FB pour le moteur doit remplir les conditions ci-après :

- Des signaux provenant du poste d'opération sont nécessaires pour le démarrage ou l'arrêt du moteur ou des pompes.
- Un signal en retour provenant des pompes ou du moteur doit indiquer que le moteur est en marche.
- Il faut calculer le temps entre l'émission du signal de mise en marche du moteur et la réception du signal en retour. En l'absence de signal en retour à l'expiration de ce temps, le moteur doit être arrêté.
- Les lampes respectives sur le poste d'opération doivent s'allumer et s'éteindre.
- Le FB fournit un signal pour la commande du moteur.

Ces conditions peuvent être définies comme entrées et sorties du bloc fonctionnel. Le tableau ci-après présente les paramètres du FB pour le moteur.

Nom du paramètre	Entrée	Sortie	Entrée/sortie
START	n		
Stop	n		
Response	n		
Reset_Maint	n		
Timer_No	n		
Response_Time	n		
Fault		n	
Start_Dsp		n	
Stop_Dsp		n	
Maint		n	
Motor			n

## Déclaration des variables du FB pour le moteur

Vous devez déclarer les paramètres d'entrée, de sortie et d'entrée/sortie du FB pour le moteur.

Opérande	Déclaration	Nom	Type	Valeur initiale
0.0	IN	Demarrage	BOOL	FALSE
0.1	IN	Stop	BOOL	FALSE
0.2	IN	Response	BOOL	FALSE
0.3	IN	Reset_Maint	BOOL	FALSE
2.0	IN	Time_No	TIMER	
4.0	IN	Response_Time	S5TIME	S5T#0MS
6.0	OUT	Fault	BOOL	FALSE
6.1	OUT	Start_Dsp	BOOL	FALSE
6.2	OUT	Stop_Dsp	BOOL	FALSE
6.3	OUT	Maint	BOOL	FALSE
8.0	IN_OUT	Motor	BOOL	FALSE
10.0	STAT	Time_bin	WORD	W#16#0
12.0	STAT	Time_BCD	WORD	W#16#0
14.0	STAT	Starts	INT	0
16.0	STAT	Start_Edge	BOOL	FALSE

Pour les FB, les variables d'entrée, de sortie, d'entrée/sortie et statiques sont contenues dans le DB d'instance indiqué dans l'opération d'appel. Quant aux variables temporaires, elles se trouvent dans la pile L.

## Programmation du FB pour le moteur

Dans STEP 7, il faut créer les blocs appelés par d'autres blocs avant les blocs contenant l'appel. Vous devez donc, dans notre exemple de programme, écrire le FB pour le moteur avant l'OB1.

La section des instructions du FB1 se présente comme suit en langage de programmation LIST.

### Réseau 1 Démarrage/arrêt et maintien

```
U(
O #Start
O #Motor
)
UN      #Stop
= #Motor
```

### Réseau 2 Surveillance de la mise en route

```
U #Motor
L #Response_Time
SE      #Timer_No
UN      #Motor
R #Timer_No
L #Timer_No
T #Timer_bin
LC      #Timer_No
T #Timer_BCD
U #Timer_No
UN      #Response
S #Fault
R #Motor
```

### Réseau 3 Lampe de démarrage et remise à zéro erreurs

```
U #Response
= #Start_Dsp
R #Fault
```

### Réseau 4 Lampe d'arrêt

```
UN      #Response
= #Stop_Dsp
```

### Réseau 5 Comptage des démarrages

```
U #Motor
FP      #Start_Edge
SPBN    lab1
L #Starts
+ 1
T #Starts
lab1: NOP0
```

**Réseau 6    Lampe de signalisation de maintenance**

```
L #Starts
L 50
>=|
= #Maint
```

**Réseau 7    Remise à zéro du compteur du nombre de démarrages**

```
U #Reset_Maint
U #Maint
SPBN    END
L 0
T #Starts
END: NOP        0
```

**Création des blocs de données d'instance**






Créez trois blocs de données et ouvrez-les les uns après les autres. Dans la boîte de dialogue "Nouveau bloc de données", activez la case d'option "Bloc de données associé à un bloc fonctionnel". Sélectionnez "FB1" dans la zone de liste "Affectation". Vous venez ainsi de définir les blocs de données comme blocs de données d'instance avec affectation fixe au FB1.

**A.5.2.4    Création de la fonction pour les soupapes****Tâches de la FC**

La fonction pour les soupapes d'admission et d'alimentation ainsi que pour la soupape de vidange contient les fonctions logiques ci-après.

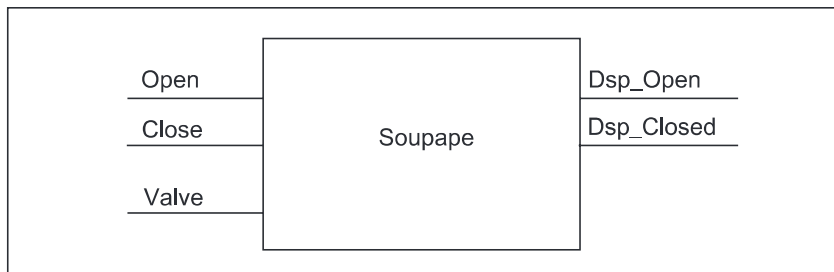
- Il existe une entrée d'ouverture et une entrée de fermeture des soupapes.
- Une série de verrouillages permet l'ouverture des soupapes. L'état des verrouillages est sauvegardé dans les données locales temporaires (pile L) de l'OB1 ("Enable\_Valve") et est combiné aux entrées d'ouverture et de fermeture lors de l'exécution de la FC pour les soupapes.

Le tableau ci-après présente les paramètres à transmettre à la fonction.

Paramètres pour les soupapes	Entrée	Sortie	Entrée/sortie
Open			
Close			
Dsp_Open			
Dsp_Closed			
Valve			

## Identification des entrées et sorties

La figure ci-après montre les entrées et les sorties de la FC générique pour les soupapes. Les appareils appelant le FB pour le moteur transmettent les paramètres d'entrée et la FC pour les soupapes renvoie les paramètres de sortie.



## Déclaration des variables de la FC pour les soupapes

Vous devez déclarer les paramètres d'entrée, de sortie et d'entrée/sortie pour la fonction commandant les soupapes comme vous l'avez fait pour le FB "Moteur" (voir la table de déclaration des variables ci-après).

Opérande	Déclaration	Nom	Type	Valeur initiale
0.0	IN	Open	BOOL	FALSE
0.1	IN	Close	BOOL	FALSE
2.0	OUT	Dsp_Open	BOOL	FALSE
2.1	OUT	Dsp_Closed	BOOL	FALSE
4.0	IN_OUT	Valve	BOOL	FALSE

Pour les FC, les variables temporaires sont sauvegardées dans la pile L. Les variables d'entrée, de sortie et d'entrée/sortie prennent la forme de pointeurs désignant le bloc de code ayant appelé la FC. Un espace mémoire supplémentaire est utilisé pour ces variables dans la pile L (après les variables temporaires).

### Programmation de la FC pour les soupapes

Vous devez également écrire la fonction FC1 pour les soupapes avant l'OB1, car il faut créer le bloc appelé avant le bloc appelant.

La section des instructions de la FC1 se présente comme suit en langage de programmation LIST.

#### Réseau 1 Ouverture/fermeture et maintien

```
U(  
O #Open  
O #Valve  
)  
UN      #Close  
= #Valve
```

#### Réseau 2 , si soupape ouverte

```
U #Valve  
= #Dsp_Open
```

#### Réseau 3 Signalisation, si soupape fermée

```
UN      #Valve  
= #Dsp_Closed
```

### A.5.2.5 Création de l'OB1

L'OB1 détermine la structure de l'exemple de programme. Il contient, en outre, les paramètres transmis aux différents blocs fonctionnels et fonctions. Ainsi :

- Les réseaux LIST pour les pompes d'alimentation et le moteur mélangeur fournissent au FB pour le moteur les paramètres d'entrée pour le démarrage ("Start"), l'arrêt ("Stop"), pour le signal en retour ("Response") et pour la remise à zéro de l'indicateur de maintenance ("Reset\_Maint"). Le FB pour le moteur s'exécute à chaque cycle de l'automate.
- Lorsque le FB pour le moteur s'exécute, les entrées "Timer\_No" et "Response\_Time" déterminent la temporisation à utiliser et la durée pendant laquelle un signal en retour doit être émis.
- La fonction pour les soupapes et le FB "Moteur" s'exécutent à chaque cycle de programme de l'automate, car ils sont appelés dans l'OB1.

Le programme utilise le FB "Moteur" avec différents DB d'instance afin d'accomplir les tâches requises pour la commande des pompes d'alimentation et du moteur mélangeur.



## Déclaration de variables pour l'OB1

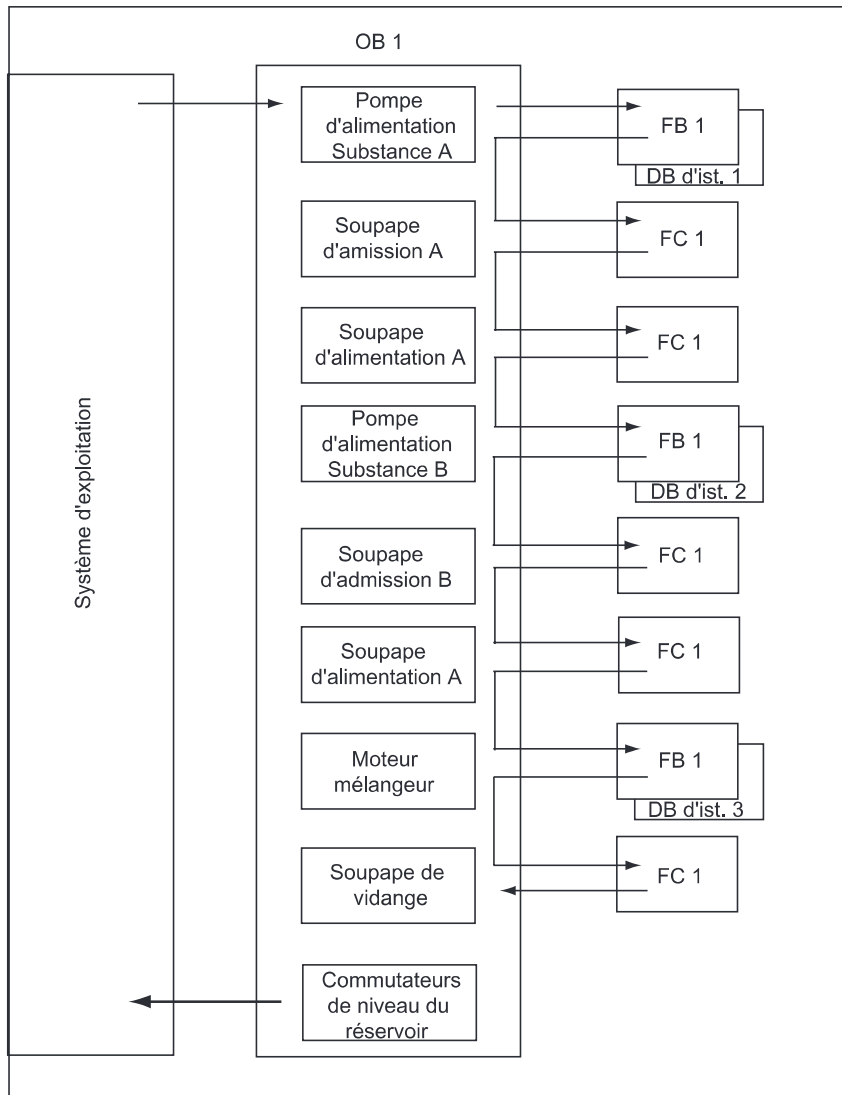
La table de déclaration des variables pour l'OB1 est représentée ci-après. Il ne faut pas modifier les vingt premiers octets qui contiennent les informations de déclenchement de l'OB1.

Opérande	Déclaration	Nom	Type
0.0	TEMP	OB1_EV_CLASS	BYTE
1.0	TEMP	OB1_SCAN1	BYTE
2.0	TEMP	OB1_PRIORITY	BYTE
3.0	TEMP	OB1_OB_NUMBR	BYTE
4.0	TEMP	OB1_RESERVED_1	BYTE
5.0	TEMP	OB1_RESERVED_2	BYTE
6.0	TEMP	OB1_PREV_CYCLE	INT
8.0	TEMP	OB1_MIN_CYCLE	INT
10.0	TEMP	OB1_MAX_CYCLE	INT
12.0	TEMP	OB1_DATE_TIME	DATE_AND_TIME
20.0	TEMP	Enable_Motor	BOOL
20.1	TEMP	Enable_Valve	BOOL
20.2	TEMP	Start_Fulfilled	BOOL
20.3	TEMP	Stop_Fulfilled	BOOL
20.4	TEMP	Inlet_Valve_A_Open	BOOL
20.5	TEMP	Inlet_Valve_A_Closed	BOOL
20.6	TEMP	Feed_Valve_A_Open	BOOL
20.7	TEMP	Feed_Valve_A_Closed	BOOL
21.0	TEMP	Inlet_Valve_B_Open	BOOL
21.1	TEMP	Inlet_Valve_B_Closed	BOOL
21.2	TEMP	Feed_Valve_B_Open	BOOL
21.3	TEMP	Feed_Valve_B_Closed	BOOL
21.4	TEMP	Open_Drain	BOOL
21.5	TEMP	Close_Drain	BOOL
21.6	TEMP	Close_Valve_Fulfilled	BOOL

## Création du programme pour l'OB1

Dans STEP 7, il faut créer les blocs appelés par d'autres blocs avant les blocs contenant l'appel. Vous devez donc, dans notre exemple de programme, écrire le FB pour le moteur et la FC pour les soupapes avant le programme de l'OB1.

Les blocs FB1 et FC1 sont appelés à plusieurs reprises dans l'OB1, le FB1 avec différents DB d'instance :



La section des instructions de l'OB1 se présente comme suit en langage de programmation LIST.

#### Réseau 1 Verrouillages pour la pompe d'alimentation A

```
U "EMER_STOP_off"
U "Tank_below_max"
UN "Drain"
= #Enable_Motor
```

#### Réseau 2 Appel du FB Moteur pour substance A

```
U "Feed_pump_A_start"
U #Enable_Motor
= #Start_Fulfilled
U(
O "Feed_pump_A_stop"
ON #Enable_Motor
)
= #Stop_Fulfilled
CALL "Motor_block", "DB_feed_pump_A"
Start :=#Start_Fulfilled
Stop :=#Stop_Fulfilled
Response :="Flow_A"
Reset_Maint :="Reset_maint"
Timer_No :=T12
Reponse_Time:=S5T#7S
Fault :="Feed_pump_A_fault"
Start_Dsp :="Feed_pump_A_on"
Stop_Dsp :="Feed_pump_A_off"
Maint :="Feed_pump_A_maint"
Motor :="Feed_pump_A"
```

#### Réseau 3 Ajournement de la validation de soupape pour substance A

```
U "Feed_pump_A"
L S5T#1S
SE T 13
UN "Feed_pump_A"
R T 13
U T 13
= #Enable_Valve
```

#### Réseau 4 Commande de la soupape d'admission pour substance A

```
UN "Flow_A"
UN "Feed_pump_A"
= #Close_Valve_Fulfilled
CALL "Valve_block"
Open :=#Enable_Valve
Close :=#Close_Valve_Fulfilled
Dsp_Open :=#Inlet_Valve_A_Open
Dsp_Closed:=#Inlet_Valve_A_Closed
Valve :="Inlet_Valve_A"
```

**Réseau 5 Commande de la soupape d'alimentation pour substance A**

```

UN      "Flow_A"
UN      "Feed_pump_A"
=       #Close_Valve_Fulfilled
CALL    "Valve_block"
  Open  :=#Enable_Valve
  Close :=#Close_Valve_Fulfilled
  Dsp_Open :=#Feed_Valve_A_Open
  Dsp_Closed:=#Feed_Valve_A_Closed
  Valve :="Feed_Valve_A"

```

**Réseau 6 Verrouillages pour la pompe d'alimentation B**

```

U "EMER_STOP_off"
U "Tank_below_max"
UN "Drain"
= "Enable_Motor"

```

**Réseau 7 Appel du FB Moteur pour substance B**

```

U      "Feed_pump_B_start"
U      #Enable_Motor
=      #Start_Fulfilled
U(
O      "Feed_pump_B_stop"
ON     #Enable_Motor
)
=      #Stop_Fulfilled
CALL   "Motor_block", "DB_feed_pump_B"
  Start :=#Start_Fulfilled
  Stop  :=#Stop_Fulfilled
  Response :="Flow_B"
  Reset_Maint :="Reset_maint"
  Timer_No :=T14
  Reponse_Time:=S5T#7S
  Fault :="Feed_pump_B_fault"
  Start_Dsp :="Feed_pump_B_on"
  Stop_Dsp :="Feed_pump_B_off"
  Maint :="Feed_pump_B_maint"
  Motor :="Feed_pump_B"

```

**Réseau 8 Ajournement de la validation de soupape pour substance B**

```

U "Feed_pump_B"
L S5T#1S
SE T 15
UN "Feed_pump_B"
R T 15
U T 15
= #Enable_Valve

```

**Réseau 9 Commande de la soupape d'admission pour substance B**

```

UN      "Flow_B"
UN      "Feed_pump_B"
=       #Close_Valve_Fulfilled
CALL    "Valve_block"
  Open  :=#Enable_Valve
  Close :=#Close_Valve_Fulfilled
  Dsp_Open :=#Inlet_Valve_B_Open
  Dsp_Closed:=#Inlet_Valve_B_Closed
  Valve :="Inlet_Valve_B"

```

**Réseau 10 Commande de la soupape d'alimentation pour substance B**

```

UN      "Flow_B"
UN      "Feed_pump_B"
=       #Close_Valve_Fulfilled
CALL    "Valve_block"
  Open  :=#Enable_Valve
  Close :=#Close_Valve_Fulfilled
  Dsp_Open :=#Feed_Valve_B_Open
  Dsp_Closed:=#Feed_Valve_B_Closed
  Valve :="Feed_Valve_B"

```

**Réseau 11 Verrouillages pour le moteur mélangeur**

```

U "EMER_STOP_off"
U "Tank_above_min"
UN "Drain"
= #Enable_Motor

```

**Réseau 12 Appel du FB Moteur pour moteur mélangeur**

```

U      "Agitator_start"
U      #Enable_Motor
=      #Start_Fulfilled
U(
O      "Agitator_stop"
ON     #Enable_Motor
)
=      #Stop_Fulfilled
CALL   "Motor_block", "DB_Agitator"
  Start :=#Start_Fulfilled
  Stop  :=#Stop_Fulfilled
  Response :="Agitator_running"
  Reset_Maint :="Reset_maint"
  Timer_No :=T16
  Reponse_Time:=S5T#10S
  Fault :="Agitator_fault"
  Start_Dsp :="Agitator_on"
  Stop_Dsp :="Agitator_off"
  Maint :="Agitator_maint"
  Motor :="Agitator"

```

**Réseau 13 Verrouillages pour la soupape de vidange**

```

U "EMER_STOP_off"
U "Tank_not_empty"
UN "Agitator"
= "Enable_Valve"

```

**Réseau 14    Commande de la soupape de vidange**

```
U      "Drain_open"  
U      #Enable_Valve  
=      #Open_Drain  
U(  
O      "Drain_closed"  
ON     #Enable_Valve  
)  
=      #Close_Drain  
CALL  "Valve_block"  
  Open :=#Open_Drain  
  Close :=#Close_Drain  
  Dsp_Open :="Drain_open_disp"  
  Dsp_Closed :="Drain_closed_disp"  
  Valve :="Drain"
```

**Réseau 15    Indication du niveau du réservoir**

```
UN     "Tank_below_max"  
=      "Tank_max_disp"  
UN     "Tank_above_min"  
=      "Tank_min_disp"  
UN     "Tank_not_empty"  
=      "Tank_empty_disp"
```

### A.5.3 Exemple d'utilisation d'alarmes horaires

Structure du programme utilisateur "Alarme horaire"

FC12

OB10

OB1 et OB80

#### A.5.3.1 Structure de l'alarme horaire du programme utilisateur

##### Problème posé

La sortie A 4.0 doit être mise à 1 du lundi, 5.00 heures au vendredi, 20.00 heures. Du vendredi, 20.00 heures au lundi, 5.00 heures la sortie A 4.0 doit être remise à 0.

##### Transcription dans le programme utilisateur

Le tableau suivant montre les tâches partielles des blocs utilisés.

Bloc	Tâche partielle
OB1	Appel de la fonction FC12
FC12	Selon l'état de la sortie A 4.0, de l'état de l'alarme horaire et des entrées E 0.0 et E 0.1 <ul style="list-style-type: none"> <li>• Prédéfinir instant de déclenchement</li> <li>• Mettre alarme horaire à 1</li> <li>• Activer alarme horaire</li> <li>• CAN_TINT</li> </ul>
OB10	Selon le jour de la semaine en cours <ul style="list-style-type: none"> <li>• Prédéfinir instant de déclenchement</li> <li>• Mettre à 1 ou remettre à 0 la sortie A 4.0</li> <li>• Mettre à 1 l'alarme horaire suivante</li> <li>• Activer l'alarme horaire suivante</li> </ul>
OB80	Mises à 1 de la sortie A 4.1 Enregistrer l'information de l'événement de déclenchement de l'OB80 dans la zone des mémentos

## Opérandes utilisés

Le tableau suivant montre les opérandes globaux utilisés. Les variables temporaires des blocs sont déclarées dans la section de déclaration du bloc respectif.

Opérande	Signification
E 0.0	Entrée de validation de "Mettre alarme horaire à 1" et "Activer alarme horaire"
E 0.1	Entrée d'annulation d'une alarme horaire
A 4.0	Sortie mise à 1/remise à 0 par l'OB d'alarme horaire (OB10)
A 4.1	Sortie mise à 1 en cas d'erreur d'horloge (OB80)
MW 16	ETAT de l'alarme horaire (SFC31 "QRY_TINT")
MB 100 à MB 107	Mémoire pour l'information de l'événement de déclenchement de l'OB10 (uniquement horodatage)
MB 110 à MB 129	Mémoire pour l'information de l'événement de déclenchement de l'OB80 (erreur d'horloge)
MW 200	RET_VAL de la SFC28 "SET_TINT"
MB 202	Mémoire intermédiaire des résultats binaires (bit d'état RB) pour les SFC
MW 204	RET_VAL de la SFC30 "ACT_TINT"
MW 208	RET_VAL de la SFC31 "QRY_TINT"

## SFC et FC utilisés

Les fonctions système suivantes sont utilisées dans l'exemple de programme :

- SFC28 "SET\_TINT" : Réglage de l'alarme horaire
- SFC29 "CAN\_TINT" : Annulation de l'alarme horaire
- SFC30 "ACT\_TINT" : Activation de l'alarme horaire
- SFC31 "QRY\_TINT" : Interrogation de l'alarme horaire
- FC3 "D\_TOD\_DT" : Regroupement de DATE et TIME\_OF\_DAY en DT



### A.5.3.2 FC12

#### Section de déclaration

Les variables temporaires de blocs suivantes sont déclarées dans la section de déclaration de la FC12 :

Nom de la variable	Type de données	Déclaration	Commentaire
IN_HEURE	TIME_OF_DAY	TEMP	Prédéfinition de l'heure de déclenchement
IN_DATE	DATE	TEMP	Prédéfinition de la date de déclenchement
OUT_HEURE_DATE	DATE_AND_TIME	TEMP	Date/heure de déclenchement converties
MEMENTO_OK	BOOL	TEMP	Validation pour le réglage de l'alarme horaire

#### Section des instructions en LIST

Dans la section des instructions de la FC12, vous entrez le programme utilisateur LIST suivant :

LIST (FC 12)	Signification
Réseau 1 :	
CALL SFC 31	SFC QRY_TINT
NR_OB := 10	Interrogation de l'ETAT de l'alarme horaire.
VAL_RET:= MW 208	
ETAT := MW 16	
Réseau 2 :	
UN A 4.0	Prédéfinir instant de déclenchement en fonction de A 4.0 (dans la variable
SPB mont	#IN_DATE et #IN_HEURE)
L D#1995-1-27	La date de déclenchement est un vendredi.
T #IN_DATE	
L TOD#20:0:0.0	
T #IN_HEURE	
SPA conv	
mont: L D#1995-1-23	La date de déclenchement est un lundi.
T #IN_DATE	
L TOD#5:0:0.0	
T #IN_HEURE	
conv: NOP 0	

LIST (FC 12)	Signification
Réseau 3 :	
CALL FC 3	Convertir format de DATE et TIME_OF_DAY en DATE_AND_TIME (pour régler l'alarme horaire)
IN1 := #IN_DATE	
IN2 := #IN_HEURE	
VAL_RET := #OUT_HEURE_DATE	
Réseau 4 :	Toutes les conditions pour régler l'alarme horaire remplies ? (entrée de validation mise à 1, alarme horaire non active et OB d'alarme horaire chargé)
U E 0.0	
UN M 17.2	
U M 17.4	Si oui, alors régler l'alarme horaire...
= #MEMENTO_OK	
Réseau 5 :	
U #MEMENTO_OK	
SPBNB m001	
CALL SFC 28	
NR_OB := 10	
SDT := #OUT_HEURE_DATE	
PERIODE := W#16#1201	
VAL_RET := MW 200	...et activer l'alarme horaire.
m001 U RB	
= M 202.3	
Réseau 6 :	
U #MEMENTO_OK	
SPBNB m002	
CALL SFC 30	
NR_OB := 10	
VAL_RET := MW 204	Si l'entrée pour annuler l'alarme horaire est mise à 1, alors annuler l'alarme horaire.
m002 U RB	
= M 202.4	
Réseau 7 :	
U E 0.1	
SPBNB m003	
CALL SFC 29	
NR_OB := 10	
RET_VAL := MW 210	
m003 U RB	
= M 202.5	

### A.5.3.3 OB10

#### Section de déclaration

Selon la section de déclaration prédéfinie pour l'OB10, les variables temporaires de bloc suivantes sont déclarées :

- Structure pour l'ensemble de l'information de l'événement de déclenchement (STARTINFO)
- Dans la structure STARTINFO, une structure pour l'heure (T\_STMP)
- Autres variables temporaire de blocs JOURSEM, IN\_DATE, IN\_HEURE et OUT\_HEURE\_DATE

Nom de la variable	Type de données	Déclaration	Commentaire
STARTINFO	STRUCT	TEMP	Ensemble de l'information de l'événement de déclenchement de l'OB10 déclaré comme structure
ID_E	WORD	TEMP	ID d'événement
CLASSE_PR	BYTE	TEMP	Classe de priorité
NR_OB	BYTE	TEMP	Numéro de l'OB
RESERVED_1	BYTE	TEMP	Réservé
RESERVED_2	BYTE	TEMP	Réservé
PERIODE	WORD	TEMP	Périodicité de l'alarme horaire
RESERVED_3	DWORD	TEMP	Réservé
T_STMP	STRUCT	TEMP	Structure pour les indications d'horodatage
ANNEE	BYTE	TEMP	
MOIS	BYTE	TEMP	
JOUR	BYTE	TEMP	
HEURES	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDES	BYTE	TEMP	
MSEC_JOURSEM	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	
JOURSEM	INT	TEMP	Jour de la semaine
IN_DATE	DATE	TEMP	Variable d'entrée pour FC3 (conversion du format horaire)
IN_HEURE	TIME_OF_DAY	TEMP	Variable d'entrée pour FC3 (conversion du format horaire)
OUT_HEURE_DATE	DATE_AND_TIME	TEMP	Variable de sortie pour FC3 et variable d'entrée pour SFC28

## Section des instructions en LIST

Dans la section des instructions de l'OB10, vous entrez le programme utilisateur LIST suivant :

LIST (OB 10)	Signification
Réseau 1 :	
L     #STARTINFO.T_STMP.MSEC_JOURSEM	Sélectionner jour de la semaine
L     W#16#F	
UW	
T     #JOURSEM	et mémoriser.
Réseau 2 :	
L     #JOURSEM	Si le jour de la semaine n'est pas un lundi,
L     2	alors prédéfinir le lundi, 5.00 heures comme
<>I	prochain instant de déclenchement et remettre
SPB   mont	la sortie A 4.0 à zéro.
Réseau 3 :	
L     D#1995-1-27	
T     #IN_DATE	Sinon, c'est-à-diresi le jour de la semaine
L     TOD#20:0:0.0	= lundi, alors prédéfinir vendredi, 20.00
T     #IN_HEURE	heures comme prochain instant de
SET	déclenchement et mettre la sortie A 4.0 à 1.
=     A 4.0	
SPA   conv	
mont:  L     D#1995-1-23	
T     #IN_DATE	
L     TOD#5:0:0.0	
T     #IN_HEURE	
CLR	
=     A 4.0	
conv:  NOP    0	Prédéfinition de l'instant de déclenchement
	terminée.
Réseau 4 :	
CALL  FC 3	Convertir l'instant de déclenchement
IN1   := #IN_DATE	prédéfini dans le format DATE_AND_TIME (pour
IN2   := #IN_HEURE	SFC28).
VAL_RET := #OUT_HEURE_DATE	
Réseau 5 :	
CALL  SFC 28	Régler l'alarme horaire.
NR_OB  := 10	
SDT    := #OUT_HEURE_DATE	
PERIODE := W#16#1201	
VAL_RET := MW 200	
U     RB	
=     M 202.1	

LIST (OB 10)	Signification
Réseau 6 :	
CALL SFC 30	Activer alarme horaire
NR_OB := 10	
VAL_RET := MW 204	
U RB	
= M 202.2	
Réseau 7 :	
CALL SFC 20	Transfert de bloc : enregistrer indication
SRCBLK := #STARTINFO.T_STMP	horaire de l'information de l'événement de
VAL_RET := MW 206	déclenchement de l'OB10 dans la zone de
DSTBLK := P#M 100.0 OCTET 8	mémento MB 100 à MB 107.

### A.5.3.4 OB1 et OB80

Puisque l'information de l'événement de déclenchement de l'OB1 (OB pour le programme cyclique) n'est pas exploitée dans cet exemple, seule l'information de l'événement de déclenchement de l'OB80 est représentée.

#### Section des instructions de l'OB1

Dans la section des instructions de l'OB1, vous entrez le programme utilisateur LIST suivant :

LIST (OB 1)	Signification
CALL FC 12	Appel de la fonction FC12

#### Section de déclaration de l'OB80

Selon la section de déclaration prédéfinie de l'OB80, les variables temporaires de bloc suivantes sont déclarées :

- Structure pour l'ensemble de l'information de l'événement de déclenchement (STARTINFO)
- Dans la structure STARTINFO, une autre structure pour l'heure (T\_STMP)

Nom de la variable	Type de données	Déclaration	Commentaire
STARTINFO	STRUCT	TEMP	Ensemble de l'information de l'événement de déclenchement de l'OB80 déclaré comme structure
ID_E	WORD	TEMP	ID d'événement
CLASSE_PR	BYTE	TEMP	Classe de priorité
NR_OB	BYTE	TEMP	Numéro de l'OB
RESERVED_1	BYTE	TEMP	Réservé
RESERVED_2	BYTE	TEMP	Réservé
INFO_S1	WORD	TEMP	Information supplémentaire sur l'événement ayant occasionné l'erreur.
INFO_S2	DWORD	TEMP	Information supplémentaire sur l'ID d'événement, la classe de priorité et le numéro d'OB de l'événement d'erreur
T_STMP	STRUCT	TEMP	Structure pour les indications d'horodatage
ANNEE	BYTE	TEMP	
MOIS	BYTE	TEMP	
JOUR	BYTE	TEMP	
HEURES	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDES	BYTE	TEMP	
MSEC_JOURSEM	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

## Section des instructions de l'OB80

Dans la section des instructions de l'OB80, appelé par le système d'exploitation en cas d'erreur d'horloge, vous entrez le programme utilisateur LIST suivant :

LIST (OB 80)	Signification
Réseau 1 :	
UN A 4.1	Mettre la sortie A 4.1 à 1 lorsque l'erreur d'horloge est survenue.
S A 4.1	
CALL SFC 20	Transfert de bloc : enregistrer l'ensemble de l'événement de déclenchement dans la zone de memento MB 110 à MB 129.
SRCBLK := #STARTINFO	
VAL_RET := MW 210	
DSTBLK := P#M 110.0 octet 20	

## A.5.4 Exemple d'utilisation d'alarmes temporisées

Structure du programme utilisateur "Alarme temporisée"

OB20

OB1

### A.5.4.1 Structure de l'alarme temporisée du programme utilisateur

#### Problème posé

Lorsque l'entrée E 0.0 est mise à 1, la sortie A 4.0 doit être mise à 1, et ceci 10 secondes plus tard. Chaque mise à 1 de l'entrée E 0.0 doit déclencher une nouvelle fois le temps de retard.

Comme identificateur spécifique à l'utilisateur, l'instant (secondes et millisecondes) de déclenchement de l'alarme temporisée doit apparaître dans l'information de l'événement de déclenchement de l'OB d'alarme temporisée (OB20).

Si E 0.1 est mise à 1 durant ces 10 secondes, le bloc d'organisation OB20 ne doit pas être appelé, c'est-à-dire la sortie A 4.0 ne doit pas être mise à 1.

Lorsque l'entrée E 0.2 est mise à 1, la sortie A 4.0 doit être remise à 0.

#### Transcription dans le programme utilisateur

Le tableau suivant montre les tâches partielles des blocs utilisés.

Bloc	Tâche partielle
OB1	Lecture et préparation de l'heure actuelle pour le déclenchement de l'alarme temporisée Déclenchement de l'alarme temporisée en fonction du changement de front à l'entrée E 0.0 Annulation de l'alarme temporisée en fonction de l'état de l'alarme temporisée et du changement de front à l'entrée E 0.1 Remise à 0 de la sortie A 4.0 en fonction de l'état de l'entrée E 0.2
OB20	Mise à 1 de la sortie A 4.0 Lecture et préparation de l'heure actuelle Enregistrement de l'information de l'événement de déclenchement dans la zone des mémentos



## Opérandes utilisés

Le tableau ci-après montre les tables de données globales utilisées. Les variables temporaires des blocs sont déclarées dans la section de déclaration du bloc respectif.

Opérande	Signification
E 0.0	Entrée de validation de "Déclencher l'alarme de temporisation"
E 0.1	Entrée d'annulation d'une alarme temporisée
E 0.2	Entrée de remise à 0 de la sortie A 4.0
A 4.0	Sortie mise à 1 par l'OB d'alarme temporisée (OB20)
MB 1	Utilisé pour les mementos de front et la mémoire intermédiaire des résultats binaires (bit d'état RB) pour les SFC
MW 4	ETAT de l'alarme temporisée (SFC34 "QRY_TINT")
MD 10	Secondes et millisecondes en format DCB reprises dans l'information de l'événement de déclenchement de l'OB1
MW 100	VAL_RET de la SFC32 "SRT_DINT"
MW 102	VAL_RET de la SFC34 "QRY_DINT"
MW 104	VAL_RET de la SFC33 "CAN_DINT"
MW 106	VAL_RET de la SFC20 "BLKMOV"
MB 120 à MB 139	Mémoire pour l'information de l'événement de déclenchement de l'OB20
MD 140	Secondes et millisecondes en format DCB reprises dans l'information de l'événement de déclenchement de l'OB20
MW 144	Secondes et millisecondes en format DCB reprises dans l'information de l'événement de déclenchement de l'OB1 ; repris dans l'information de l'événement de déclenchement de l'OB20 (identificateur spécifique à l'utilisateur SIGN)

## SFC utilisées

Les fonctions système suivantes sont utilisées dans le programme utilisateur "Alarmes temporisées" :

- SFC32 "SRT\_DINT" : Déclenchement de l'alarme temporisée
- SFC33 "CAN\_DINT" : Annulation de l'alarme temporisée
- SFC34 "QRY\_DINT" : Interrogation de l'état d'une alarme temporisée

## A.5.4.2 OB20

### Section de déclaration

En fonction de la section de déclaration prédéfinie de l'OB20, les variables temporaires de bloc suivantes sont déclarées :

- Structure pour l'ensemble de l'information de l'événement de déclenchement (STARTINFO)
- Dans la structure STARTINFO, une structure pour l'horodatage (T\_STMP)

Nom de la variable	Type de données	Déclaration	Commentaire
STARTINFO	STRUCT	TEMP	Informations de déclenchement pour l'OB20
ID_E	WORD	TEMP	ID d'événement
NR_NIVEX	BYTE	TEMP	Niveau d'exécution
NR_OB	BYTE	TEMP	Numéro d'OB
IDD1	BYTE	TEMP	Identification de données 1
IDD2	BYTE	TEMP	Identification de données 2
SIGN	WORD	TEMP	Identification spécifique à l'utilisateur
DTIME	TIME	TEMP	Heure de déclenchement de l'alarme temporisée
T_STMP	STRUCT	TEMP	Structure pour les indications d'horodatage (horodateur)
ANNEE	BYTE	TEMP	
MOIS	BYTE	TEMP	
JOUR	BYTE	TEMP	
HEURES	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDES	BYTE	TEMP	
MSEC_JOURSEM	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

## Section des instructions

Dans la section des instructions de l'OB20, vous entrez le programme utilisateur LIST suivant :

LIST (OB 20)	Signification
Réseau 1 :	
SET	Mise à 1 impérative de la sortie A 4.0
=     A 4.0	
Réseau 2 :	
L     AW 4	Actualisation immédiate du mot de sortie
T     PAW 4	
Réseau 3 :	
L     #STARTINFO.T_STMP.SECONDES	Lecture des secondes dans les informations de l'événement de déclenchement
T     MW 140	
L     #STARTINFO.T_STMP.MSEC_JOURSEM	Lecture des millisecondes et du jour de la semaine dans les informations de l'événement de déclenchement
T     MW 142	
L     MD 140	
SRD    4	Elimination du jour de la semaine et
T     MD 140	réinscription des millisecondes (sont à présent en format DCB dans le MW 142).
Réseau 4 :	Lecture de l'instant de déclenchement de l'alarme temporisée (=appel de la SFC32) dans l'information de l'événement de déclenchement
L     #STARTINFO.SIGN	
T     MW 144	
Réseau 5 :	Copie de l'information de l'événement de déclenchement dans la zone de mémoire (MB 120 à MB 139)
CALL SFC 20	
SRCBLK := STARTINFO	
VAL_RET := MW 106	
DSTBLK := P#M 120.0 OCTET 20	

### A.5.4.3 OB1

#### Section de déclaration

En fonction de la section de déclaration prédéfinie de l'OB1, les variables temporaires de bloc suivantes sont déclarées :

- Structure pour l'ensemble de l'information de l'événement de déclenchement (STARTINFO)
- Dans la structure STARTINFO, une structure pour l'horodatage (T\_STMP)

Nom de la variable	Type de données	Déclaration	Commentaire
STARTINFO	STRUCT	TEMP	Informations de déclenchement pour l'OB1
ID_E	WORD	TEMP	ID d'événement
NR_NIVEX	BYTE	TEMP	Niveau d'exécution
NR_OB	BYTE	TEMP	Numéro d'OB
IDD 1	BYTE	TEMP	Identification de données 1
IDD 2	BYTE	TEMP	Identification de données 2
CYC_ACT	INT	TEMP	Temps de cycle en cours
CYC_MIN	INT	TEMP	Temps de cycle minimum
CYC_MAX	INT	TEMP	Temps de cycle maximal
T_STMP	STRUCT	TEMP	Structure pour les indications d'horodatage (horodatage)
ANNEE	BYTE	TEMP	
MOIS	BYTE	TEMP	
JOUR	BYTE	TEMP	
HEURE	BYTE	TEMP	
MINUTES	BYTE	TEMP	
SECONDES	BYTE	TEMP	
MSEC_JOURSEM	WORD	TEMP	
	END_STRUCT	TEMP	
	END_STRUCT	TEMP	

## Section des instructions

Dans la section des instructions de l'OB1, vous entrez le programme utilisateur LIST suivant :

LIST (OB 1)	Signification
Réseau 1 :	
L     #STARTINFO.T_STMP.SECONDES	Lecture des secondes dans l'information de l'événement de déclenchement
T     MW 10	
L     #STARTINFO.T_STMP.MSEC_JOURSEM	Lecture des millisecondes et du jour de la semaine dans les informations de l'événement de déclenchement
T     MW 12	
L     MD 10	Elimination du jour de la semaine et
SRD   4	réinscription des millisecondes (sont à présent en format DCB dans le MW 12)
T     MD 10	Front positif à l'entrée E 0.0 ?
Réseau 2 :	
U     E 0.0	
FP    M 1.0	
=     M 1.1	
Réseau 3 :	
U     M 1.1	Si oui, déclenchement de l'alarme temporisée (instant de déclenchement de l'alarme temporisée affecté au paramètre SIGN)
SPBNB m001	
CALL SFC 32	
NR_OB := 20	
DTME  := T#10S	
SIGN  := MW 12	
VAL_RET:= MW 100	
m001: NOP    0	Interrogation de l'état de l'alarme temporisée (SFC QRY_DINT)
Réseau 4 :	
CALL SFC 34	
NR_OB := 20	
VAL_RET:= MW 102	Front positif à l'entrée E 0.1 ?
ETAT := MW 4	
Réseau 5 :	
U     E 0.1	
FP    M 1.3	
=     M 1.4	... et alarme temporisée activée ? (bit 2 de l'ETAT de l'alarme temporisée)

LIST (OB 1)	Signification
Réseau 6 :	Alors annulation de l'alarme temporisée
U M 1.4	
U M 5.2	
SPBNB m002	
CALL SFC 33	Remise à 0 de la sortie A 4.0 avec l'entrée
NR_OB := 20	E 0.2
VAL_RET:= MW 104	
m002: NOP 0	
U E 0.2	
R A 4.0	

#### A.5.4.4 Exemple de masquage et de démasquage d'événements d'erreurs synchrones

Dans l'exemple suivant d'un programme utilisateur, nous allons vous montrer le masquage et le démasquage d'événements d'erreurs synchrones. La SFC36 "MSK\_FLT" masque les erreurs suivantes dans le masque d'erreurs de programmation :

- Erreur de longueur de zone lors de la lecture
- Erreur de longueur de zone lors de l'écriture

Un second appel de la SFC36 "MSK\_FLT" masque en plus une erreur d'accès :

- Erreur d'accès à la périphérie lors de l'écriture

La SFC38 "READ\_ERR" interroge les événements d'erreurs synchrones masqués. L'Erreur d'accès à la périphérie lors de l'écriture est à nouveau démasquée par la SFC37 "DMSK\_FLT".

#### Instruction

La suite représente l'OB1, dans lequel l'exemple pour le programme utilisateur a été programmé en LIST.

LIST (réseau 1)	Signification
UN M 255.0	mémento non rémanent M 255.0 (uniquement lors du premier cycle=0)
SPBNB m001	
CALL SFC 36	SFC36 MSK_FLT (Masquage d'événements d'erreurs synchrones)
	Bit2=Bit3=1 (BLFL et BLFS sont masqués)
PRGFLT_SET_MASK :=DW#16#C	tous les bits=0 (aucune erreur d'accès n'est masquée)
ACCFLT_SET_MASK :=DW#16#0	Valeur en retour
VAL_RET :=MW 100	Affichage du masque d'erreurs de programmation actuel dans MD 10
PRGFLT_MASKED :=MD 10	Affichage du masque d'erreurs d'accès actuel dans MD 14
ACCFLT_MASKED :=MD 14	Mise à 1 de M255.0, si masquage réussi
m001: U RB	
S M 255.0	

LIST (réseau 2)	Signification
CALL SFC 36	SFC36 MSK_FLT (masquage d'événements d'erreurs synchrones)
PRGFLT_SET_MASK :=DW#16#0	tous les bits=0 (aucune autre erreur de programmation n'est masquée)
ACCFLT_SET_MASK :=DW#16#8	Bit3=1 (les erreurs d'accès en écriture sont masquées)
VAL_RET :=MW 102	Valeur en retour
PRGFLT_MASKED :=MD 20	Affichage du masque d'erreurs de programmation actuel dans MD 20
ACCFLT_MASKED :=MD 24	Affichage du masque d'erreurs d'accès actuel dans MD 24

LIST (réseau 3)	Signification
UN M 27.3	Fin du bloc, si erreur d'accès en écriture (bit3 dans ACCFLT_MASKED) non masquée
BEB	

LIST (réseau 4)	Signification
L B#16#0	
T PAB 16	Accès en écriture (avec valeur 0) sur PAB 16

LIST (réseau 5)	Signification
CALL SFC 38	SFC38 READ_ERR (interrogation d'événements d'erreurs synchrones)
PRGFLT_QUERY :=DW#16#0	tous les bits=0 (aucune interrogation d'erreur de programmation)
ACCFLT_QUERY :=DW#16#8	Bit3=1 (interrogation d'erreur d'accès en écriture)
VAL_RET :=MW 104	Valeur en retour
PRGFLT_CLR :=MD 30	Affichage du masque d'erreurs de programmation actuel dans MD 30
ACCFLT_CLR :=MD 34	Affichage du masque d'erreurs d'accès actuel dans MD 34
U RB	aucune erreur survenue ni erreur d'accès en écriture détectée
U M 37.3	Inversion du RLG
NOT	M 0.0=1, si PAB 16 existant
= M 0.0	

LIST (réseau 6)	Signification
L B#16#0	
T PAB 17	Erreur d'accès en écriture (avec valeur 0) sur PAB 17



LIST (réseau 7)	Signification
CALL SFC 38	SFC38 READ_ERR (interrogation d'événements d'erreurs synchrones)
PRGFLT_QUERY :=DW#16#0	tous les bits=0 (aucune interrogation d'erreurs de programmation)
ACCFLT_QUERY :=DW#16#8	Bit3=1 (interrogation d'erreur d'accès en écriture)
VAL_RET :=MW 104	Valeur en retour
PRGFLT_CLR :=MD 30	Affichage du masque d'erreurs de programmation actuel dans MD 30
ACCFLT_CLR :=MD 34	Affichage du masque d'erreurs d'accès actuel dans MD 34
U RB	aucune erreur survenue ni erreur d'accès en écriture détectée
U M 37.3	Inversion du RLG
NOT	M 0.1=1, si PAB 17 existant
= M 0.1	

LIST (réseau 8)	Signification
L B#16#0	
T PAB 18	Accès en écriture (avec valeur 0) sur PAB 18

LIST (réseau 9)	Signification
CALL SFC 38	SFC38 READ_ERR (interrogation d'événements d'erreurs synchrones)
PRGFLT_QUERY :=DW#16#0	tous les bits=0 (aucune interrogation d'erreurs de programmation)
ACCFLT_QUERY :=DW#16#8	Bit3=1 (interrogation d'erreur d'accès en écriture)
VAL_RET :=MW 104	Valeur en retour
PRGFLT_CLR :=MD 30	Affichage du masque d'erreurs de programmation actuel dans MD 30
ACCFLT_CLR :=MD 34	Affichage du masque d'erreurs d'accès actuel dans MD 34
U RB	aucune erreur survenue ni erreur d'accès en écriture détectée
U M 37.3	Inversion du RLG
NOT	M 0.2=1, si PAB 18 existant
= M 0.2	

LIST (réseau 10)	Signification
L      B#16#0	
T      PAB 19	Accès en écriture (avec valeur 0) sur PAB 19
LIST (réseau 11)	Signification
CALL    SFC 38	SFC38 READ_ERR (interrogation d'événements d'erreurs synchrones)
PRGFLT_QUERY :=DW#16#0	tous les bits=0 (aucune interrogation d'erreur de programmation)
ACCFLT_QUERY :=DW#16#8	Bit3=1 (interrogation d'erreur d'accès en écriture)
	Valeur en retour
VAL_RET                :=MW 104	Affichage du masque d'erreurs de programmation actuel dans MD 30
PRGFLT_CLR            :=MD 30	Affichage du masque d'erreurs d'accès actuel dans MD 34
ACCFLT_CLR            :=MD 34	aucune erreur survenue ni erreur d'accès en écriture détectée
U      RB	Inversion du RLG
U      M 37.3	M 0.3=1, si PAB 19 existant
NOT	
=      M 0.3	
LIST (réseau 12)	Signification
CALL    SFC 37	SFC37 DMSK_FLT (démasquage d'événements d'erreurs synchrones)
PRGFLT_RESET_MASK    :=DW#16#0	tous les bits=0 (aucun démasquage d'erreur de programmation)
ACCFLT_RESET_MASK    :=DW#16#8	Bit3=1 (démasquage d'erreur d'accès en écriture)
	Valeur en retour
VAL_RET                :=MW 102	Affichage du masque d'erreurs de programmation actuel dans MD 20
PRGFLT_MASKED :=MD 20	Affichage du masque d'erreurs d'accès actuel dans MD 24
ACCFLT_MASKED :=MD 24	
LIST (réseau 13)	Signification
U      M 27.3	Fin de bloc, si erreur d'accès en écriture (bit3 dans ACCFLT_MASKED) non démasquée
BEB	

LIST (réseau 14)	Signification
U M 0.0	
SPBNB m002	
L EB 0	transférer EB 0 dans PAB 16, si existant
T PAB 16	
m002: NOP 0	

LIST (réseau 15)	Signification
U M 0.1	
SPBNB m003	
L EB 1	transférer EB 1 dans PAB 17, si existant
T PAB 17	
m003: NOP 0	

LIST (réseau 16)	Signification
U M 0.2	
SPBNB m004	
L EB 2	transférer EB 2 dans PAB 18, si existant
T PAB 18	
m004: NOP 0	

LIST (réseau 17)	Signification
U M 0.3	
SPBNB m005	
L EB 3	transférer EB 3 dans PAB 19, si existant
T PAB 19	
m005: NOP 0	

### A.5.4.5 Exemple d'inhibition et de validation d'événements d'alarme et d'événements asynchrones (SFC 39 et 40)

Dans cet exemple de programme utilisateur, on considère qu'une partie du programme ne doit pas être interrompue par des alarmes. Pour cette partie du programme, la SFC39 "DIS\_IRT" inhibe les appels de l'OB35 (alarme horaire) et la SFC40 "EN\_IRT" valide à nouveau les appels de l'OB35.

Les SFC39 et SFC40 sont appelées dans l'OB1 :

LIST (OB 1)	Signification
U M 0.0	Partie du programme pouvant être interrompue :
S M 90.1	
U M 0.1	Partie du programme ne devant pas être interrompue par des alarmes :
S M 90.0	
:	Inhiber et rejeter les alarmes
:	
CALL SFC 39	Mode 2 : inhibition d'OB d'alarme individuels inhibition de l'OB35
MODE :=B#16#2	
NR_OB :=35	
VAL_RET :=MW 100	
:	Validation des alarmes
:	
L PEW 100	Mode 2 : validation d'OB d'alarme individuels Validation de l'OB35
T MW 200	
L MW 90	
T MW 92	
:	Partie du programme pouvant être interrompue :
:	
CALL SFC 40	
MODE :=B#16#2	
NR_OB :=35	Partie du programme pouvant être interrompue :
VAL_RET :=MW 102	
U M 10.0	Partie du programme pouvant être interrompue :
S M 190.1	
U M 10.1	Partie du programme pouvant être interrompue :
S M 190.0	
:	Partie du programme pouvant être interrompue :
:	

### A.5.4.6 Exemple de traitement différé d'événements d'alarme et d'événements asynchrones (SFC 41 et 42)

Dans cet exemple de programme utilisateur, on considère qu'une partie du programme ne doit pas être interrompue par des alarmes. Dans cette partie du programme, la SFC41 "DIS\_AIRT" retarde les alarmes qui sont ultérieurement validées par la SFC42 "EN\_AIRT".

La SFC41 et la SFC42 sont appelées dans l'OB1 :

LIST (OB 1)	Signification
U M 0.0	Partie du programme pouvant être interrompue :
S M 90.1	
U M 0.1	Partie du programme ne devant pas être interrompue par des alarmes :
S M 90.0	
:	
:	
CALL SFC 41	Inhibition et retardement des alarmes
VAL_RET :=MW 100	
L PEW 100	
T MW 200	
L MW 90	
T MW 92	
:	
:	
:	
CALL SFC 42	Validation de l'alarme
VAL_RET :=MW 102	
L MW 100	La valeur en retour contient le nombre d'inhibitions d'alarmes mises en oeuvre
DEC 1	
L MW 102	La valeur en retour contient le nombre d'inhibitions d'alarmes mises en oeuvre après validation des alarmes, ce nombre doit être identique
<>I	comme avant l'inhibition des alarmes (dans ce cas "0")
SPB err	
U M 10.0	Partie du programme pouvant être interrompue :
S M 190.1	
U M 10.1	Partie du programme ne devant pas être interrompue par des alarmes :
S M 190.0	
:	
:	
BEA	
erre: L MW 102	Le nombre d'inhibitions d'alarmes mises en oeuvre est affiché
T AW 12	

## A.6 Accès aux zones de données du processus et de la périphérie

### A.6.1 Accès à la zone de données du processus

La CPU peut accéder aux entrées et sorties des modules d'entrées/sorties TOR centralisés ou décentralisés soit indirectement via la mémoire image du processus, soit directement via le bus interne/de fond de panier ou P.

Quant aux entrées et sorties des modules d'entrées/sorties analogiques centralisés et décentralisés, la CPU y accède directement via le bus interne/de fond de panier ou P. Vous avez cependant également la possibilité d'entrer les adresses des modules analogiques dans la zone de mémoire image.

### Adressage des modules

L'association entre les adresses utilisées dans le programme utilisateur et les modules se fait par configuration des modules avec STEP 7 :

- pour la périphérie centralisée : disposition du profilé support ou châssis et affectation des modules aux emplacements dans la table de configuration ;
- pour la périphérie décentralisée (PROFIBUS DP ou PROFINET IO) : disposition des esclaves de périphérie décentralisée (DP) ou IO Devices avec indication de l'adresse PROFIBUS ou du nom d'appareil et affectation des modules aux emplacements.

La configuration des modules remplace le réglage d'adresses des différents modules par commutateurs. La CPU reçoit de la PG des données comme résultat de la configuration, données grâce auxquelles elle reconnaît les modules affectés.

### Adressage de la périphérie

Il existe une zone d'adresses propre pour les entrées et pour les sorties. Aussi, l'adresse d'une zone de périphérie doit-elle contenir l'identification E - pour les entrées - et A - pour les sorties - en plus de l'indication d'octet ou de mot.

Le tableau suivant présente les zones d'adresses de périphérie disponibles.

Plage d'opérandes	Accès par des unités de taille suivante	Notation S7
Zone de périphérie des entrées	Octet de périphérie d'entrée	PEB
	Mot de périphérie d'entrée	PEW
	Double mot de périphérie d'entrée	PED
Zone de périphérie des sorties	Octet de périphérie de sortie	PAB
	Mot de périphérie de sortie	PAW
	Double mot de périphérie de sortie	PAD

Reportez-vous aux manuels suivants pour savoir quelles zones d'adresses sont possibles pour les différents modules.

- CPU 31xC et CPU 31x, Caractéristiques techniques
- Systèmes d'automatisation S7-400, Caractéristiques des CPU

### **Adresse de début de module**

L'adresse de début de module est l'adresse d'octet la plus basse d'un module. Elle représente l'adresse de début de la zone des données utiles du module et est souvent utilisée pour désigner le module entier.

Elle est, par exemple, inscrite dans les informations de déclenchement des blocs d'organisation associés à des alarmes de processus, de diagnostic, de débrogage/enfichage et à des erreurs d'alimentation, et identifie ainsi le module à l'origine de l'alarme.

## A.6.2 Accès à la zone de données de périphérie

La zone de données de périphérie se décompose :

- en données utiles
- et en données de diagnostic et de paramètres.

Ces deux parties comportent une zone d'entrée (accès en lecture uniquement) et une zone de sortie (accès en écriture uniquement).

### Données utiles

On accède aux données utiles via l'adresse d'octet - pour les modules de signaux TOR - et via l'adresse de mot - pour les modules de signaux analogiques - de la zone d'entrée ou de sortie. Vous pouvez accéder à ces données utiles à l'aide de commandes de chargement et de transfert, de fonctions de communication (accès de contrôle-commande) ou par l'intermédiaire du transfert de mémoire image. Parmi les données utiles, on compte :

- les signaux d'entrée et de sortie analogiques et TOR de modules de signaux,
- les informations d'état et de forçage de modules de fonction et
- les informations pour couplages point à point et par bus de modules de communication (uniquement S7-300).

Lors de la transmission de données utiles, il est possible d'atteindre une cohérence des données de quatre octets au maximum (excepté pour les esclaves normés DP ; voir Définition du comportement en fonctionnement). Si vous utilisez l'instruction "Transférer double mot", 4 octets sont transmis en un bloc et sans modification. En revanche, si vous vous servez de quatre instructions "Transférer octet d'entrée", il se pourrait que soit déclenché à une limite d'instruction un OB d'alarme de processus qui transmette des données à la même adresse et modifie ainsi le contenu des quatre octets d'origine.

### Données de diagnostic et de paramètres

Il n'est pas possible d'accéder individuellement aux données de diagnostic et de paramètres d'un module, mais uniquement sous la forme d'enregistrements entiers. En principe, les données de diagnostic et de paramètres sont transmises de manière cohérente.

On accède aux données de diagnostic et de paramètres via l'adresse de début du module concerné et le numéro d'enregistrement. Les enregistrements sont subdivisés en enregistrements d'entrée et de sortie, les enregistrements d'entrée pouvant uniquement être lus et les enregistrements de sortie uniquement être écrits. Vous pouvez accéder aux enregistrements à l'aide de fonctions système ou de fonctions de communication (contrôle-commande). Le tableau suivant montre l'affectation des enregistrements aux données de diagnostic et de paramètres.

Données	Description
Données de diagnostic	Pour les modules capables de diagnostic, vous recevez lors de la lecture des enregistrements 0 et 1 les données de diagnostic de ce module.
Données de paramètres	Pour les modules paramétrables, vous transférez lors de l'écriture des enregistrements 0 et 1 les paramètres de ce module.



## Accès aux enregistrements

Vous pouvez utiliser les informations contenues dans les enregistrements d'un module pour modifier le paramétrage de modules paramétrables et pour lire les informations de diagnostic des modules aptes au diagnostic.

Le tableau suivant présente les fonctions système permettant d'accéder aux enregistrements.

SFC	Application
Paramétrage de modules	
SFC55 WR_PARM	Transfert des paramètres modifiables (enregistrement 1) au module de signaux adressé
SFC56 WR_DPARM	Transfert des paramètres des SDB 100 à 129 au module de signaux adressé
SFC57 PARM_MOD	Transfert de tous les paramètres des SDB 100 à 129 au module de signaux adressé
SFC58 WR_REC	Transfert d'un enregistrement quelconque au module de signaux adressé
Lecture d'informations de diagnostic	
SFC59 RD_REC	Lecture des données de diagnostic

### Nota

Lorsqu'un esclave DPV1 est configuré via un fichier GSD (GSD à partir de la révision 3) et que l'interface DP du maître DP est paramétrée sur "**Compatible S7**", aucun enregistrement ne peut être lu ou écrit dans les modules E/S dans le programme utilisateur avec les SFC 58/59 ou SFB 53/52. Dans ce cas, le maître DP adresse un emplacement erroné (emplacement configuré +3).

Solution : commutez l'interface du maître DP sur "DPV1".

## Adressage de modules S5

Vous avez la possibilité de :

- coupler à un automate S7-400 des châssis d'extension SIMATIC S5 avec la carte de couplage IM 463 et
- enficher certaines cartes S5 en boîtiers d'adaptation dans les châssis centralisés de l'automate S7-400.

Consultez le manuel "Systèmes d'automatisation S7-400, M7-400 - Installation et configuration" ou la description livrée avec le boîtier d'adaptation pour savoir comment adresser les cartes S5 avec SIMATIC S7.

## A.7 Définition du comportement en fonctionnement

Ce chapitre explique comment vous pouvez influencer, à l'aide des paramètres système ou de fonctions système, sur les propriétés des automates programmables S7-300 et S7-400 qui ne sont pas définitivement fixées.

Vous trouverez des informations détaillées sur les paramètres des modules dans l'aide en ligne de STEP 7 ainsi que dans les manuels suivants :

- Manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU"
- Manuel de référence "Systèmes d'automatisation S7-300, M7-300 - Caractéristiques des modules"
- Manuel de référence "Systèmes d'automatisation S7-400, M7-400 -Caractéristiques des modules"

Vous trouverez toutes les informations sur les fonctions système dans le manuel de référence "Logiciel système pour SIMATIC S7-300/400 - Fonctions standard et fonctions système".

### Adressage des esclaves normés DP

Si des esclaves normés DP doivent émettre ou recevoir des données de plus de 4 octets, vous devez faire appel à des fonctions système spécifiques.

Dans le cas de CPU qui prennent en charge l'échange de données cohérentes (> 4 octets) via la zone de périphérie, il n'est pas nécessaire d'utiliser les SFC 14/15 (voir lecture et écriture décentralisée de données cohérentes).

SFC	Application
Paramétrage de modules	
SFC15 DPWR_DAT	Transfert de données quelconques au module de signaux adressé
Lecture d'informations de diagnostic	
SFC13 DPNRM_DG	Lecture des données de diagnostic (lecture asynchrone)
SFC14 DPRD_DAT	Lecture de données cohérentes (longueur 3 ou supérieure à 4 octets)

Une alarme de diagnostic avec 4 octets de données de diagnostic est signalée à la CPU lors de l'arrivée d'un télégramme de diagnostic DP. Il est possible de lire ces quatre octets avec la fonction système SFC13 DPNRM\_DG.

## A.7.1 Modification du comportement et des propriétés des modules

### Paramètres par défaut

- A la livraison, tous les modules paramétrables du système d'automates programmables S7 sont réglés à des valeurs par défaut qui conviennent à des applications standard. Ces valeurs par défaut vous permettent d'utiliser directement les modules sans devoir effectuer d'autres réglages. Elles sont présentées dans les descriptions de modules des manuels suivants :
- Manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU"
- Manuel de référence "Systèmes d'automatisation S7-300, M7-300 - Caractéristiques des modules"
- Manuel de référence "Systèmes d'automatisation S7-400, M7-400 - Caractéristiques des modules"

### Modules paramétrables

Vous pouvez toutefois bien sûr paramétrer le comportement et les propriétés des modules et, ainsi, les adapter aux exigences et aux caractéristiques de votre installation. Les CPU, FM, CP ainsi que certains modules d'entrées et de sorties analogiques et modules d'entrées TOR sont des modules paramétrables.

Il existe des modules paramétrables avec ou sans sauvegarde.

Après chaque coupure de courant, vous devez transmettre à nouveau les données de paramétrage aux modules sans sauvegarde. Les paramètres de ces modules sont sauvegardés dans la zone de mémoire rémanente de la CPU (paramétrage indirect par la CPU).

### Définition et chargement des paramètres

Vous définissez les paramètres des modules à l'aide de STEP 7. Lors de l'enregistrement des paramètres, STEP7 crée l'objet "Blocs de données système" qui est chargé dans la CPU avec le programme utilisateur et qui, de là, est transféré dans les modules concernés lors de la mise en route.

## Paramétrages possibles

Les paramètres des modules sont répartis en blocs de paramètres. Le manuel "Automate programmable S7-300, Installation et configuration - Caractéristiques des CPU" et le manuel de référence "Automate programmable S7-400, M7-400 -Caractéristiques des modules" précisent quels blocs de paramètres sont disponibles sur quelles CPU.

Exemples de blocs de paramètres :

- comportement à la mise en route,
- cycle,
- MPI
- diagnostic,
- rémanence,
- mémentos de cadence,
- traitement d'alarmes,
- périphérie interne (uniquement pour S7-300),
- niveau de protection,
- données locales,
- horloge temps réel,
- erreurs asynchrones.

## Paramétrage à l'aide de SFC

Outre par paramétrage avec STEP 7, il est également possible de modifier les paramètres des modules à l'aide de fonctions système à partir du programme S7. Le tableau suivant indique quelles SFC transfèrent quels paramètres de modules.

SFC	Application
SFC55 WR_PARM	Transfert des paramètres modifiables (enregistrement 1) au module de signaux adressé
SFC56 WR_DPARM	Transfert des paramètres des SDB associés au module de signaux adressé
SFC57 PARM_MOD	Transfert de tous les paramètres des SDB associés au module de signaux adressé
SFC58 WR_REC	Transfert d'un enregistrement quelconque au module de signaux adressé

Vous trouverez des informations détaillées sur les fonctions système dans le manuel de référence "Logiciel système pour SIMATIC S7-300/400 - Fonctions standard et fonctions système".

Les manuels suivants indiquent quels paramètres de module vous pouvez modifier dynamiquement.

- Manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU"
- Manuel de référence "Systèmes d'automatisation S7-300, M7-300 - Caractéristiques des modules"
- Manuel de référence "Systèmes d'automatisation S7-400, M7-400 -Caractéristiques des modules"

## A.7.2 Mise à jour hors ligne du microprogramme (du système d'exploitation) de modules

La manière de transférer une nouvelle version du microprogramme (= nouvelle version du système d'exploitation) sur un module tel qu'une CPU via une carte mémoire est décrite ci-après.

Deux étapes sont nécessaires à la mise à jour :

1. Création d'une "Update-Memory Card" (transfert des fichiers de mise à jour sur une carte mémoire) avec la PG ou le PC et un programmeur externe).
2. Réalisation d'une mise à jour du système d'exploitation avec la "Update-Memory Card" sur la CPU.

### Conditions

- Carte mémoire avec une capacité de mémoire suffisante. Vous obtiendrez des informations sur les pages de téléchargement du service Customer-Supports. Vous y trouverez également les fichiers de mise à jour.
- PG ou PC équipé d'un dispositif de programmation de cartes mémoire.

### Transfert des fichiers de mise à jour sur une carte mémoire

1. Utilisez l'Explorateur de Windows pour créer un nouveau répertoire.
2. Chargez le fichier de mise à jour souhaité dans ce répertoire et décompimez-le. Ce répertoire contient ensuite le fichier de mise à jour.
3. Enfichez la carte mémoire S7 dans la PG ou dans le programmeur.
4. Effacez la carte mémoire (commande de menu **Fichier > Carte mémoire S7 > Effacer** dans SIMATIC Manager).
5. Choisissez la commande **Système cible > Actualiser le système d'exploitation de l'AP** dans SIMATIC Manager.
6. Dans la boîte de dialogue qui s'ouvre, sélectionnez le répertoire contenant le fichier de mise à jour.
7. Effectuez un double clic sur le fichier de mise à jour.  
Vous démarrez ainsi la procédure de programmation. Lorsqu'elle est terminée, le message suivant s'affiche : "La mise à jour du microprogramme pour le module portant le n° de référence ... a été transférée correctement sur la carte mémoire S7.

## Réalisation d'une mise à jour du système d'exploitation

1. Enfichez la carte mémoire contenant la mise à jour dans le système cible (c'est-à-dire dans la CPU).
2. Coupez l'alimentation (PS) de la CPU.
3. Enfichez dans la CPU la carte mémoire préparée contenant la mise à jour.
4. Rétablissez l'alimentation de la CPU.  
Le système d'exploitation est transféré de la carte mémoire S7 dans l'EPROM flash interne.  
Durant cette opération, toutes les DEL de la CPU s'allument.
5. La mise à jour est terminée au bout de 2 min. environ, ce qui est signalé par un clignotement lent de la DEL STOP de la CPU (demande d'effacement général par le système).
6. Coupez l'alimentation et enfichez éventuellement la carte mémoire S7 prévue pour l'exploitation.
7. Rétablissez l'alimentation. La CPU exécute alors un effacement général automatique, après quoi elle est prête à fonctionner.

### A.7.3 Avantage des fonctions d'horodatage

Toutes les CPU S7-300/S7-400 possèdent une horloge (horloge temps réel ou horloge logicielle). Dans l'automate programmable, l'horloge peut aussi bien fonctionner comme horloge maître que comme esclave avec synchronisation externe. Elle permet l'utilisation d'alarmes horaires et de compteurs d'heures de fonctionnement.

#### Format horaire

L'horloge affiche toujours l'heure (résolution minimale 1 s) et la date avec le jour de la semaine. Certaines CPU permettent aussi l'affichage de millisecondes (voir le manuel "Système d'automatisation S7-300, Installation et configuration - Caractéristiques des CPU" et le manuel de référence "Systèmes d'automatisation S7-400, M7-400, Caractéristiques des modules").

#### Réglage et lecture de l'heure

Pour régler l'heure et la date de l'horloge de la CPU, vous appelez la SFC0 SET\_CLK dans le programme utilisateur ou choisissez la commande depuis la PG pour démarrer l'horloge. La SFC1 READ\_CLK ou la commande de menu sur la PG vous permettent de lire la date et l'heure actuelles de la CPU.

---

#### Nota

Pour éviter des indications différentes dans les systèmes IHM, il est conseillé de régler la CPU sur **l'heure d'hiver** !

---

## Paramétrage de l'horloge

Lorsqu'un réseau comporte plus d'un module avec horloge, vous devez paramétrer dans STEP 7, quelle CPU doit fonctionner comme maître et quelle CPU doit fonctionner comme esclave pour la synchronisation de l'heure. Le paramétrage vous permet également de définir si la synchronisation doit être réalisée via le bus de communication ou via l'interface MPI et ce à quels intervalles.

## Synchronisation de l'heure

Afin de garantir que tous les modules du réseau sont réglés à la même heure, les horloges esclave du programme système sont synchronisées à des intervalles réguliers (paramétrables) par le programme système. La fonction système SFC48 SNC\_RTCB vous permet de transmettre la date et l'heure de l'horloge maître aux horloges esclave.

## Mise en œuvre d'un compteur d'heures de fonctionnement

Un compteur d'heures de fonctionnement permet de compter la durée d'activation d'un élément du système connecté ou la durée de fonctionnement de la CPU sous forme de somme du nombre d'heures de fonctionnement.

Le compteur d'heures de fonctionnement est stoppé lorsque la CPU est à l'état d'arrêt. Sa valeur est conservée après un effacement général. Lors d'un démarrage à chaud, le compteur d'heures de fonctionnement doit être démarré par le programme utilisateur, lors d'un redémarrage, il continue à fonctionner s'il était activé.

Vous pouvez affecter une valeur initiale au compteur d'heures de fonctionnement à l'aide de la SFC2 SET\_RTM. La SFC3 CTRL\_RTM vous permet de le démarrer ou de l'arrêter. Avec la SFC4 READ\_RTM, vous pouvez lire le nombre actuel d'heures de fonctionnement ainsi que l'état du compteur ("arrêté" ou "compte").

Une CPU peut posséder jusqu'à 8 compteurs d'heures de fonctionnement. La numérotation débute à 0.

## A.7.4 Utilisation de mémentos de cadence et de temporisations

### Mémentos de cadence

Un memento de cadence est un memento dont l'état binaire change périodiquement dans un rapport impulsion-pause de 1:1. Vous déterminez, lors du paramétrage du memento de cadence avec STEP 7, l'octet de memento de la CPU qui servira de memento de cadence.

### Utilité

Vous pouvez vous servir de mémentos de cadence dans votre programme utilisateur pour, par exemple, commander des avertisseurs lumineux avec lampe clignotante ou pour déclencher des événements périodiques (comme l'enregistrement d'une valeur de mesure).

### Fréquences possibles

A chaque bit de l'octet de memento de cadence est affectée une fréquence. Le tableau suivant présente cette affectation.

Bits de l'octet du memento de cadence	7	6	5	4	3	2	1	0
Période (s)	2,0	1,6	1,0	0,8	0,5	0,4	0,2	0,1
Fréquence (Hz)	0,5	0,625	1	1,25	2	2,5	5	10

### Nota

Les mémentos de cadence s'exécutent de manière asynchrone par rapport au cycle de CPU. Ainsi, dans les cycles longs, l'état du memento de cadence peut changer plusieurs fois.

### Temporisations

Les temporisations sont une zone de la mémoire système. La fonction d'une temporisation est définie par le programme utilisateur (par exemple, retard à la montée). Le nombre de temporisations disponibles dépend de la CPU.

### Nota

- Si vous faites appel, dans votre programme utilisateur, à plus de temporisations que n'en autorise la CPU, une erreur synchrone est signalée et l'OB121 est déclenché.
- Dans les S7-300 (à l'exception de la CPU 318), les temporisations ne peuvent être simultanément démarrées et actualisées que dans l'OB1 et dans l'OB100. Dans tous les autres OB, elles peuvent uniquement être démarrées.



# Index

## A

### Abréviations

sélection 10-25

Accès à la zone de données de périphérie A-124

Accès à la zone de données du processus A-122

Accès en ligne aux systèmes cibles dans le multiprojet 18-4

ACT\_TINT 4-29

### Activation

affichage des mnémoniques dans le bloc 8-13

### Actualisation

appels de blocs 10-26

### Adressage

indirect en mémoire A-56

interzone A-56, A-57

intrazone A-56

### Adressage

absolu 8-1, 8-2

symbolique 8-1, 8-2

vérification 2-10

### Adressage

de modules S5 A-124

Adressage absolu et adressage symbolique 8-1

Adressage des esclaves DP normés A-126

Adressage des modules A-122

Adressage symbolique 8-4

Adressage symbolique

dans l'exemple de programme A-83

Adresse de début A-123

### Affectation

mnémoniques A-83, A-84, A-85

Affectation de types de données aux données

locales de blocs de code A-66

Affectation des interruptions

vérification 2-10

Affectation et édition de messages sur mnémonique

(pour la CPU) 16-26

Affectation et édition de messages sur mnémonique

(pour tout le projet) 16-18

### Affichage

informations sur le bloc pour CONT

LOG

LIST 14-8

langues 16-31

### Affichage

activation des mnémoniques dans le bloc 8-13

besoin maximal en données locales dans la structure arborescente 14-3

bloc supprimé 14-3

longueurs des blocs 9-16

sous forme de paires d'appelants-appelés 14-3

sous forme de structure arborescente 14-3

structure de blocs de données associés à un UDT 11-7

structure de données de blocs de données associés à un FB (DB d'instance) 11-5

### Affichage

données de référence 14-9

### Affichage

données de référence 14-9

### Affichage

données de référence 14-9

### Affichage

données de référence 14-9

### Affichage

données de référence 14-9

### Affichage

listes dans des fenêtres supplémentaires 14-9

### Affichage

mnémoniques manquants 14-9

### Affichage

mnémoniques manquants 14-9

### Affichage

opérandes libres 14-9

### Affichage

opérandes libres 14-9

### Affichage

structure du programme 14-9

### Affichage

structure du programme 14-9

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

### Affichage

données de référence 14-10

- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - données de référence 14-10
- Affichage
  - état de fonctionnement 18-8
- Affichage
  - dans la visualisation d'état de programme 21-3
- Affichage
  - définition pour l'état du programme 21-8
- Affichage
  - état du module 23-1
- Affichage de l'état du module
  - desclaves DP derrière un Y-Link 23-14
- Affichage de l'état du module
  - d'appareils de terrain PA 23-14
- Affichage des messages de CPU et des messages de diagnostic personnalisés 16-38
- Affichage des partenaires accessibles 18-2
- Affichage du journal des modifications 6-4
- Afficher
  - longueur des blocs 9-16
- Afficher les messages enregistrés de la CPU 16-41
- Aide contextuelle 5-5
- Aide en ligne
  - appel 5-5
  - modification de la police 5-5
  - rubriques 5-5
- Alarme cyclique 4-32
  - démarrage 4-32
  - réglage 4-32
- Alarme de débrogage/enfichage (OB83) 23-38
- Alarme de diagnostic (OB82) 23-37
- Alarme de processus 4-34
- Alarme de processus
  - déclenchement 4-34
- Alarme de processus
  - réglage 4-34
- Alarme de processus
  - priorité 4-34
- Alarme de processus
  - priorité 4-34
- Alarme horaire 4-29, 18-9
  - changement de l'heure 4-30
  - démarrage 4-29
  - désactivation 4-30
  - interrogation 4-29
  - priorité 4-30
- Alarme horaire
  - utilisation A-99
- Alarme horaire
  - structure A-99
- Alarme temporisée
  - déclenchement 4-31
  - priorité 4-31
  - réglage 4-31
- Alarme temporisée
  - utilisation A-108
- Alarme temporisée
  - structure A-108
- Allocation de mémoire A-25
  - dans la pile L A-25
- Ancien projet
  - utilisation A-74, A-75
- ANY A-54, A-60, A-62
  - paramètre
    - description et utilisation A-63
- Appareils de terrain PA 23-14
- Appareils PROFIBUS PA 23-14
- Appel
  - état du module depuis la vue du projet (en ligne) 23-6
- Appel de la vue rapide 23-5
- Appel des fonctions d'aide 5-5
- Appels de bloc 4-10
- Appels de blocs
  - actualisation 10-26
- Appels imbriqués de blocs de code
  - effets sur la pile B et la pile L A-27
- Applications techniques 1-12
- Architecture du système
  - cycle 4-12
- Architecture système
  - états de fonctionnement de la CPU A-1
- Archivage
  - conditions requises 24-6
  - marche à suivre 24-6
  - possibilités 24-5
  - projets et bibliothèques 24-4
- Archiver
  - projets STEP 7 de version V2.1 avec communication par données globales A-76
- Archives
  - messages de CPU 16-38, 16-40
- ARRAY A-43
- Arrêt
  - détermination de la cause 23-16
  - état de fonctionnement de la CPU A-1
- Arrêt (STOP) A-4
- Assistant de création d'un projet 6-9
- Attente A-14
- Attente
  - état de fonctionnement de la CPU A-1
- Attribution des numéros de message 16-11
- Attributs de bloc 9-14
- Attributs de contrôle-commande 17-1
- Attributs de contrôle-commande
  - modification avec CFC 17-5
- Attributs de contrôle-commande
  - configuration au moyen de la table des mnémoniques 17-4
  - configuration avec LIST

- CONT
- LOG 17-3
- Attributs pour blocs et pour paramètres 9-20
- Attributs système
  - dans la table des mnémoniques 8-8, 8-9
  - pour la configuration des messages 16-8
  - pour la configuration des messages PCS 7 (pour tout le projet) 16-16
  - pour la configuration des messages PCS7 (pour la CPU) 16-24
  - pour les paramètres 10-6
- Automate logiciel (Software PLC) 6-24
- Automation License Manager 2-1
- Avantages des fonctions d'horodatage A-130
- Avertissement A-25
  - Débordement de la pile L A-25
- B**
- Barre de titre 5-21
- Barre des menus 5-21
- Barre d'état
  - exemple 5-21
- Barre d'outils
  - boutons 5-21
- Bascule entre les différents types de fenêtres 5-37
- Base de temps pour S5 TIME A-42
- Besoin en données locales 14-3, 14-4
- Bibliothèque 5-8
- Bibliothèque de textes
  - textes à incorporer dans des messages 16-30
- Bibliothèque standard 6-12
- Bibliothèques 6-13
- Bibliothèques
  - utilisation 9-21
- Bibliothèques
  - structure hiérarchique 9-23
- Bibliothèques
  - réorganisation 26-2
- Bibliothèques
  - réorganisation 26-2
- Bibliothèques de la version 2
  - édition 7-1
- Bibliothèques de texte utilisateur 16-33
- Bibliothèques de textes 16-33
- Bibliothèques de textes système 16-34
- Bibliothèques de textes utilisateur 16-34
- Bibliothèques de textes utilisateur
  - création 16-33
  - édition 16-34
- Bibliothèques standard
  - présentation 9-23
- BLKMOV A-16
- Bloc
  - pour modifier le pointeur A-58
- Bloc de données (DB) 4-2
- Bloc de données (DB)
  - blocs de données d'instance 4-22
- Bloc de données (DB)
  - structure 4-25
- Bloc de données (DB)
  - global 4-25
- Bloc de données (DB)
  - global 4-25
- Bloc de données (DB)
  - global 4-25
- Bloc de données (DB)
  - rémanent A-30
- Bloc de données d'instance 4-22, 4-24
  - création de plusieurs instances pour un FB 4-19
- Bloc de données d'instance
  - rémanent A-30
- Bloc de signalisation 16-21, 16-22
- Bloc d'organisation (OB)
  - OB d'arrière-plan (OB90) 4-3, 4-37
- Bloc d'organisation pour exécution cyclique du programme (OB1) 4-12
- Bloc d'organisation pour l'exécution du programme en arrière-plan (OB90) 4-37
- BLOCK
  - type de paramètre A-54
- BLOCK\_DB A-54
- BLOCK\_FB A-54
- BLOCK\_FC A-54
- BLOCK\_SDB A-54
- Blocs 4-2, 15-1, 15-2
  - comparer 9-17, 9-18, 9-19
- Blocs
  - dans le programme utilisateur 4-2
- Blocs
  - création avec GRAPH 9-8
- Blocs
  - réassignation 9-20
- Blocs
  - attributs 9-20
- Blocs
  - droits d'accès 10-4
- Blocs
  - saisie en LIST 10-12
- Blocs
  - titres 10-14
- Blocs
  - commentaires 10-14
- Blocs
  - chargement dans le système cible 19-6
- Blocs
  - chargement depuis la CPU S7 19-15
- Blocs
  - effacement sur le système cible 19-19
- Blocs chargés
  - édition dans votre PG/PC 19-16
  - enregistrement dans la mémoire intégrée EPROM 19-7
- Blocs de code
  - dans l'éditeur incrémental 10-3
  - définition
    - exemple A-82
    - enregistrement 10-28
    - horodatage 15-4
    - structure 10-3
- Blocs de données 12-1
- Blocs de données



- BOOL
  - comme type de données A-33
  - plage A-34
- Boutons
  - barre d'outils 5-21
- Branche T 10-23
- Branchements interdits en CONT 10-21
  
- C**
- CAN\_TINT 4-29
- Caractère (CHAR)
  - plage A-34
- Caractère de commentaire 20-4
- Carte à mémoire A-16
- Carte mémoire
  - paramétrage 2-8, 2-9
- Carte MPI dans la PG ou le PC 2-10
- Carte MPI-ISA (Auto) 2-10
- Cartes mémoire micro (MMC) 6-23
- Ce qu'il faut savoir sur les cartes mémoire micro (MMC) 6-23
- Certificate of License 2-1
- CFC 9-3, 9-11
- Changement
  - heure pour l'alarme horaire 4-29
- Changement d'état de fonctionnement A-3
- Charge du cycle due à la communication 4-12
- Charge due à la communication 4-12
- Chargement 19-9, 19-10
- Chargement
  - conditions préalables 19-1
  - dans la gestion du projet 19-5
  - des cartes mémoire EPROM 19-7
  - hors gestion du projet 19-6
  - programmes utilisateur dans le système cible 19-3
- Chargement
  - depuis le système cible dans la PG 19-14
- Chargement
  - configuration actuelle et tous les blocs dans la PG 19-14
- Chargement
  - blocs depuis la CPU S7 19-15
- Chargement
  - programme utilisateur A-16
- Chargement
  - programme utilisateur A-16
- Chargement
  - programme utilisateur A-16
- Chargement
  - programme utilisateur A-16
- Chargement de blocs dans le système cible 19-6
- Chargement de plusieurs objets 19-9
- Choix
  - langage de programmation 9-2
  - méthode de création de création du programme 9-1
- Clé de licence 2-5
- Clignotement de la LED FORCE 18-2
- Code source de blocs existant
  - insertion dans une source LIST 13-16
- Cohérence d'une source LIST
  - vérification 13-19
- Combinaisons de touches
  - accès à l'aide en ligne 5-36
  - bascule entre les différents types de fenêtres 5-37
  - commandes de menu 5-32
  - déplacement du curseur 5-34, 5-35
  - sélection de texte 5-36
- Commande par contact 26-4
- Commande séquentielle 9-8
- Comment éviter des erreurs lors de l'appel de blocs 15-7
- Commentaires
  - de blocs 10-15
  - de réseaux 10-14, 10-15
- Commentaires de blocs
  - saisie 10-15
- Commentaires de réseaux
  - saisie 10-15
- Communication 4-15, 4-16, 4-17
- Communication par données globales A-76
- Comparaison de blocs 9-17, 9-18, 9-19
- compatibilité A-76
- Compatibilité 7-1
- Compatibilité (esclaves DP) A-76
- Compatibilité (projets et bibliothèques de la version 2) 7-1
- Compatibilité aval 7-3
- Compilation
  - source LIST 13-20
- Compilation et chargement d'objets 19-9, 19-11
- Complexes
  - types de données A-43
- Composants pris en charge et fonctionnalités 16-44
- Composants SIMATIC pour la configuration des messages 16-5
- Compression 19-20
- Compression
  - contenu de la mémoire d'une CPU S7 19-21
- Compteur
  - zone de mémoire rémanente A-30
- Compteur d'heures de fonctionnement A-130
- Compteurs 14-5, 14-6
- Compteurs
  - tableau d'affectation 14-5
- Compteurs
  - tableau d'affectation 14-5
- Compteurs
  - tableau d'affectation 14-6
- Compteurs
  - limites supérieures pour la saisie 20-8
- Concept d'utilisation 5-20
- Conception d'une solution d'automatisation 3-1
- Conception d'une solution d'automatisation
  - subdivision du processus en tâches et zones 3-2

- Conception d'une solution d'automatisation
  - description des différentes zones fonctionnelles 3-4
- Conception d'une solution d'automatisation zones fonctionnelles 3-4
- Conception d'une solution d'automatisation
  - liste des entrées
  - sorties et entrées/sorties 3-6
- Conception d'une solution d'automatisation
  - entrées
  - sorties
  - entrées/sorties
  - listes 3-6
- Conception d'une solution d'automatisation
  - création du diagramme d'entrées/sorties pour les moteurs 3-7
- Conception d'une solution d'automatisation
  - création du diagramme d'entrées/sorties pour les soupapes 3-8
- Conception d'une solution d'automatisation
  - définition des exigences en matière de sécurité 3-9
- Conception d'une solution d'automatisation
  - description des éléments de signalisation et de commande 3-10
- Conception d'une solution d'automatisation
  - création du schéma de configuration 3-11
- Concet de signalisation
  - principes 16-1
- Condition de déclenchement 20-14
- Conditions d'installation 2-6
- Conditions et remarques pour le chargement 19-9
- Conditions préalables au chargement 19-1
- Conditions requises
  - pour l'archivage 24-6
- Configuration d'attributs de contrôle-commande avec
  - LIST
  - CONT
  - LOG 17-3
- Configuration de messages
  - transfert vers WinCC 16-37
- Configuration de messages pour les erreurs système 16-42
- Configuration de variables pour le contrôle-commande 17-1
- Configuration des attributs de contrôle-commande au moyen de la table des mnémoniques 17-4
- Configuration des messages
  - composants SIMATIC 16-5
- Configuration des messages de CPU 16-41
- Configuration des messages PCS7 (pour la CPU) 16-24
- Configuration des messages PCS7 (pour tout le projet) 16-16
- configuration prévue-configuration réelle 4-35
- Configurer 26-1
- Conflits d'horodatage 15-3
- Conseils et astuces 26-1, 26-2, 26-3, 26-5
- CONT 9-2, 9-3, 9-4
  - affichage d'informations sur le bloc 14-8
- CONT
  - branchements interdits 10-21
- Contenu des piles à l'état d'arrêt 23-16
- Contrôle des modules
  - configuration prévue-configuration réelle
  - OB de mise en route 4-35
- Contrôle des temps de cycle pour éviter les erreurs d'horloge 23-18
- Contrôle-commande de variables 17-1
- Conventions pour l'attribution de noms
  - pour les données de configuration 17-1
- Conversion A-76
- Conversion
  - projet avec communication par données globales A-76
- Conversion d'un ancien projet de version 1 A-74
- Conversion d'un ancine projet de version 2 A-75
- Convertir A-76
- Copie ou déplacement de tables de variables 20-3
- Correction
  - des interfaces dans une FC
  - un FB ou un UDT 15-6
- Correction d'erreurs
  - exemples de programmation 23-26
- COUNTER
  - type de paramètre A-54
- Coupure secteur A-6, A-7
- Court-circuit 10-21
- CPU
  - effacement général 19-18
  - simulation 22-1
- CPU (Central Processing Unit)
  - états de fonctionnement A-1, A-2, A-4
- CPU 31xC 6-23, 6-24, 6-25
- CREATE\_DB A-16
- Création
  - objets 5-23
- Création
  - diagramme d'entrées/sorties pour les moteurs 3-7
  - diagramme d'entrées/sorties pour les soupapes 3-8
  - schéma de configuration 3-11
- Création de programmes utilisateur 10-3
- Création source LIST 13-14
- Création données de référence 14-10
- Création table de variables 20-3
- Création tableaux A-47
- Création tableaux A-47
- Création tableaux A-49
- Création tableaux A-49
- Création tableaux A-49
- Création tableaux A-49
- Création tableaux A-49

Création  
  structure A-50

Création  
  structure A-50

Création  
  structure A-50

Création  
  structure A-50

Création  
  structure A-50

Création  
  structure A-51

Création  
  structure A-51

Création  
  structure A-51

Création  
  structure A-51

Création  
  type de données utilisateur A-52

Création  
  type de données utilisateur A-52

Création  
  type de données utilisateur A-53

Création  
  type de données utilisateur A-53

Création  
  type de données utilisateur A-53

Création  
  FB pour le moteur A-86

Création  
  FB pour le moteur A-86

Création  
  FB pour le moteur A-87

Création  
  FB pour le moteur A-87

Création  
  FB pour le moteur A-88

Création  
  FB pour le moteur A-88

Création  
  FB pour le moteur A-89

Création  
  FB pour le moteur A-89

Création  
  FC pour les soupapes A-90

Création  
  FC pour les soupapes A-91

Création  
  FC pour les soupapes A-91

Création  
  FC pour les soupapes A-92

Création  
  OB1 pour l'exemple d'un processus de mélange  
  industriel A-92

Création de bibliothèques de texte utilisateur 16-33

Création de bibliothèques de textes utilisateur 16-33

Création de programmes  
  marche à suivre 1-1, 1-3, 1-5

Création de textes de message dans une autre  
  langue dans la 'Signalisation d'erreurs système'  
  16-50

Création du schéma de configuration dans l'exemple  
  d'un processus de mélange industriel 3-11

Création d'un bloc de données dans la mémoire de  
  chargement 6-23

Création d'un projet 6-9

Création et manipulation d'objets 5-23

CRST/WRST A-5, A-6, A-7

CTRL\_RTM A-131

Cycle 4-3, 4-4, 4-5, 4-12, 4-13, 4-14, 4-15, 4-16

Cyclique  
  traitement du programme 4-8

## D

DATE AND TIME (date et heure)  
  format A-44  
  page A-44, A-45

DATE\_AND\_TIME A-43

DB 4-25

DB  
  tableau du format 13-13

DB dans une source LIST  
  exemples 13-27

DCB A-41

de la mémoire utilisateur 19-20

Débordement de la pile L A-25

Décalage de phase 4-32, 4-33

Déclaration de données locales A-66

Déclaration de paramètres  
  FC pour l'exemple d'un processus de mélange  
  industriel A-90

Déclaration de variables locales  
  FB pour l'exemple d'un processus de mélange  
  industriel A-86  
  OB pour l'exemple d'un processus de mélange  
  industriel A-92

Déclarations de variables dans une source LIST  
  exemples 13-21

Déclenchement  
  alarme temporisée 4-31

Déclenchement  
  alarme de processus 4-34

Déclenchement  
  définition pour la visualisation de variables 20-14

Déclenchement  
  définition pour le forçage de variables 20-17

Déclenchement du traitement du programme par  
  alarme 4-3

Défaillance d'unité (OB86) 23-41

Défaillante  
  état de fonctionnement de la CPU A-1

Défauts  
  localisation 23-1

- Définition
  - affichage de l'état du programme 21-8
  - blocs de code A-82
  - comportement en fonctionnement A-126
  - de mnémoniques lors de la saisie du programme 8-13
  - déclenchement pour la visualisation de variables 20-14
  - déclenchement pour le forçage de variables 20-17
  - exigences en matière de sécurité 3-9
  - mode de fonctionnement pour le test 21-9
- Définition de la mise en page du texte source 13-15
- Définition de la priorité de l'opérande (symbolique/absolu) 8-5
- Démarrage A-1
  - alarme cyclique 4-32, 4-33
  - alarme horaire 4-29
  - STEP 7 5-1
- Démarrage
  - installation de STEP 7 2-8
- Démarrage
  - STEP 7 avec des paramètres initiaux prédéfinis 5-3
- Démarrage
  - manuel A-5
- Démarrage
  - automatique A-5
- Démarrage
  - interruption A-5
- Démarrage
  - automatique sans sauvegarde A-6
- Démarrage
  - interruption A-10
- Démarrage à chaud A-5, A-6, A-7, A-8, A-9, A-10
- Démarrage à froid A-1, A-5, A-7, A-9
- Démasquage
  - événements de déclenchement 4-39
- Démasquage d'événements d'erreurs synchrones
  - exemple A-115
- Déplacer
  - un objet 5-23
- Désactivation
  - alarme horaire 4-29
- Désarchivage
  - marche à suivre 24-6
- Description
  - des différentes zones fonctionnelles 3-4
  - des éléments de signalisation et de commande requis 3-10
  - exigences en matière de sécurité pour l'exemple d'un processus de mélange industriel 3-9
- Description des différentes tâches et zones
  - pour l'exemple de mélangeur industriel 3-4
- Description du poste d'opération pour l'exemple d'un processus de mélange industriel 3-10
- Désinstallation
  - de la licence d'utilisation 2-5
  - STEP 7 2-11
- Détection d'erreur
  - exemples de programmes
    - valeurs de remplacement 23-31
  - utilisation d'OB d'erreur en réaction aux erreurs 4-39
- Détection d'erreurs
  - types d'OB
    - OB81 23-26
- Détermination du participant au sous-réseau 18-2
- Diagnostic du matériel 23-2
- Diagnostic système
  - extension 23-22
- Diagramme de sorties pour les moteurs
  - création 3-7
- Diagramme de sorties pour les soupapes
  - création 3-8
- Diagramme d'entrées pour les moteurs
  - création 3-7
- Diagramme d'entrées pour les soupapes
  - création 3-8
- Différence entre l'enregistrement et le chargement de blocs 19-2
- Différence entre paramétrage nominal et effectif A-5
- Différences entre forçage de variables et forçage permanent de variables 20-22
- Différences entre l'attribution de numéros de message pour tout le projet et pour la CPU 16-11
- DINT
  - type de données A-36
- DIS\_AIRT 4-40
- DIS\_IRT 4-40
- Disposition
  - boîtes 10-23, 10-24
- Disposition des fenêtres
  - enregistrement 5-31
  - modification 5-30
  - restauration 5-31
- DMSK\_FLT 4-40
- DOCPRO 24-1
- Documentation 1-1, 1-4
- Documentation des éléments constitutants du projet
  - impression 24-1
- Documentation du projet
  - impression 24-1
- Documentation d'un projet entier
  - impression 24-1
- Données de configuration 17-1, 17-2
- Données de configuration
  - conditions requises pour le transfert 16-37
  - transfert 16-37
- Données de configuration
  - conditions préalables au transfert 17-6
- Données de configuration
  - transfert 17-6
- Données de diagnostic sur les modules 23-20
- Données de périphérie A-124
- Données de référence 14-1
- Données de référence
  - application 14-1



- Données de référence
    - affichage 14-9
  - Données de référence
    - affichage 14-9
  - Données de référence
    - affichage 14-9
  - Données de référence
    - génération 14-10
  - Données de référence
    - création 14-10
  - Données de référence
    - affichage 14-10
  - Données de référence
    - affichage 14-10
  - Données d'état du diagnostic 23-20
  - Données globales
    - communication A-76
  - Données système 23-21
  - Données utiles A-124
  - Dossier
    - Blocs 9-12
  - Dossier Blocs 5-15, 9-12
  - Dossier Sources 5-18
  - Dossiers des schémas de l'installation
    - impression 24-1
  - Double mot (DWORD)
    - plage A-34
    - type de données A-33
  - DP/PA-Link (IM 157) 23-14
  - DPNRM\_DG A-126
  - DPRD\_DAT A-126
  - DPWR\_DAT A-126
  - Droit d'accès 18-6
  - Droits d'accès aux blocs ou aux sources 10-4
  - Durée d'interruption A-5
  - DWORD
    - type de données A-41
- E**
- Echange de données
    - à différent états de fonctionnement A-13
  - Echange de données direct 7-3
  - Ecriture dans un bloc de données dans la mémoire
    - de chargement 6-23
  - Editer des configurations actuelles avec des versions
    - antérieures de STEP 7 7-3
  - Editeur
    - présélections pour LIST 10-4
  - Editeur de langage
    - démarrage 9-2
  - Editeur de programmes 9-17, 10-1
  - Edition
    - dans la table des mnémoniques 8-13
    - de blocs chargés dans votre PG/PC 19-16
    - de blocs chargés lorsque le programme utilisateur ne se trouve pas dans votre PG/PC 19-17
    - de blocs chargés lorsque le programme utilisateur se trouve dans votre PG/PC 19-17
    - source S7 13-14
  - Edition de bibliothèques de textes utilisateur 16-34
  - Edition de mnémoniques dans plusieurs réseaux 26-2
  - Edition de projets et bibliothèques de la version 2 7-1
  - Edition de tables de mnémoniques 8-20
  - Edition de zones dans des tables de mnémoniques 8-20
  - Effacement
    - blocs S7 sur le système cible 19-19
    - mémoire de chargement/travail 19-18
  - Effacement général A-4
  - Effacement général
    - CPU 19-18
  - Effacer
    - des objets STEP 7 5-23
  - Effacer des variables additionnelles 16-30
  - Éléments constitutifs du projet
    - impression 24-1
  - Éléments constitutifs d'un message 16-5
  - Éléments CONT
    - représentation 10-18
  - Éléments dans les boîtes de dialogue 5-22
  - Éléments de commande
    - description dans l'exemple d'un processus de mélange industriel 3-10
  - Éléments de programme
    - insertion 10-5
  - Éléments de signalisation
    - description 3-10
  - Éléments LOG
    - représentation 10-22
  - EN / ENO
    - connexion 10-24
  - En ligne (aide)
    - appel 5-5
    - rubriques 5-5
  - EN\_AIRT 4-40
  - EN\_IRT 4-40
  - Enregistrement
    - accès A-125
    - blocs de code 10-28
    - blocs de données 11-9
    - de blocs chargés dans la mémoire intégrée EPROM 19-7
    - disposition des fenêtres 5-31
    - d'une source LIST 13-19
    - écriture A-124
    - lecture A-124, A-125
    - possibilités 24-5
    - table de variables 20-4
  - Enregistrement des données du projet sur une micro-carte mémoire (MMC) 6-25
  - Enregistrer sous 6-25
  - Entrées
    - mémoire image A-21
  - Entrées
    - listes 3-6
    - tableau d'affectation 14-5

- Entrées/sorties
  - listes 3-6
- Envoi
  - de vos propres messages de diagnostic 23-22
- Envoi de vos propres messages de diagnostic 23-22
- EPROM A-30
- Erreur
  - durant l'installation 2-8
- Erreur asynchrone
  - OB81 23-29
- Erreur d'accès à la périphérie (OB122) 23-44
- Erreur d'accès à la périphérie lors de la mise à jour de la mémoire image A-21
- Erreur d'alimentation (OB81) 23-36
- Erreur de communication (OB87) 23-42
- Erreur de programmation (OB121) 23-43
- Erreur de redondance de CPU (OB72) 23-34
- Erreur de redondance de périphérie (OB70) 23-33
- Erreur de temps (OB80) 23-35
- Erreur détectable 23-26
- Erreur d'exécution du programme (OB85) 23-40
- Erreur matérielle CPU (OB84) 23-39
- Erreur système 23-23
- Erreurs asynchrones
  - utilisation d'OB en réaction aux erreurs 4-39
- Erreurs lors de l'appel de blocs
  - éviter 15-7
- Erreurs synchrones
  - utilisation d'OB en réaction aux erreurs 4-39
- Erreurs système
  - signalisation 16-43
- Esclaves DP 7-1, 7-2
- Esclaves DP avec fichiers GSD manquants ou erronés A-76
- Esclaves Dummy A-76
- Esclaves normés DP A-126
- Etablissement
  - d'une liaison en ligne depuis la fenêtre en ligne du projet 18-3
- Etablissement
  - liaison en ligne depuis la fenêtre "Partenaires accessibles" 18-2
  - liaisons en ligne 18-1
- Etablissement d'une liaison à la CPU 20-13
- Etablissement d'une liaison en ligne depuis la fenêtre en ligne du projet 18-3
- Etat d'arrêt
  - contenu des piles 23-16
- Etat de fonctionnement 21-6
- Etat de fonctionnement
  - affichage et modification 18-8
- Etat de fonctionnement
  - "Arrêt" A-1
- Etat de fonctionnement
  - "Marche" A-1
- Etat de fonctionnement
  - "Attente" A-1
- Etat de fonctionnement
  - "Arrêt" A-3
- Etat de fonctionnement
  - "Mise en route" A-3
- Etat de fonctionnement
  - "Arrêt" A-3
- Etat de fonctionnement
  - "Attente" A-3
- Etat de fonctionnement
  - "Arrêt" A-3
- Etat de fonctionnement
  - "Marche" A-3
- Etat de fonctionnement
  - "Arrêt" A-3
- Etat de fonctionnement
  - "Marche" A-3
- Etat de fonctionnement
  - "Attente" A-3
- Etat de fonctionnement
  - "Marche" A-3
- Etat de fonctionnement
  - "Arrêt" A-4
- Etat de fonctionnement
  - "Arrêt" A-4
- Etat de fonctionnement
  - "Arrêt" (STOP) A-4
- Etat de fonctionnement
  - "Mise en route" A-11
- Etat de fonctionnement "Attente" A-14
- Etat de fonctionnement "Marche" (RUN) A-13
- Etat de l'heure 18-9
- Etat du module 23-1, 23-6, 23-12, 23-14
- Etat du module
  - affichage 23-2
- Etat du module
  - appel depuis la vue du projet (en ligne) 23-6
- Etat du module
  - possibilités d'appel 23-9
- Etat du module
  - fonctions d'information 23-10
- Etat du programme
  - définition de l'affichage 21-8
- Etat du programme de blocs de données 21-7
- Etats de fonctionnement
  - priorité A-4
- Etats de fonctionnement de la CPU A-1
- Etats de fonctionnement et changement d'état de fonctionnement A-1
- Événement de diagnostic 23-23, A-28
- Événements
  - asynchrones 4-16
- Événements asynchrones
  - inhibition et validation A-120
  - traitement différé A-121
- Événements d'alarme
  - inhibition et validation A-120
  - traitement différé A-121
- Événements de déclenchement
  - masquage 4-40
  - OB de mise en route 4-35
  - retardement 4-39
- Événements d'erreurs synchrones
  - masquage et démasquage A-115

- Exemple
    - FB dans une source LIST 13-25
    - FC dans une source LIST 13-23
    - format de nombres à virgule flottante A-39
    - inhibition et validation d'événements d'alarme et d'événements asynchrones (SFC 39 et 40) A-120
    - masquage et démasquage d'événements d'erreurs synchrones A-115
    - OB dans une source LIST 13-22
    - saisie d'opérandes dans une table de variables 20-9
    - saisie d'une plage d'opérandes continue 20-10
    - traitement différé d'événements d'alarme et d'événements asynchrones (SFC 41 et 42) A-121
    - UDT dans une source LIST 13-28
    - utilisation d'alarmes horaires A-99
    - utilisation d'alarmes temporisées A-108
  - Exemple de programme
    - processus de mélange industriel A-79
  - Exemple de recherche d'occurrences 14-12
  - Exemples
    - DB dans une source LIST 13-27
    - déclarations de variables dans une source LIST 13-21
    - saisie de valeurs de forçage/forçage permanent 20-11
  - Exemples de programmation
    - réaction à une défaillance de pile 23-26
  - Exemples de programme
    - exemple de mélangeur industriel
      - description
      - des différentes zones et tâches 3-4
    - exemple d'un processus de mélange industriel
      - description du poste d'opération 3-10
  - Exemples de programmes
    - exemple d'un processus de mélange industriel
      - création du schéma de configuration 3-11
      - définition des exigences en matière de sécurité 3-9
      - description des différentes tâches et zones
        - création d'un diagramme d'entrées/sorties 3-6
        - subdivision d'un processus en tâches 3-2
    - FB pour l'exemple d'un processus de mélange industriel A-86
    - FC pour l'exemple d'un processus de mélange industriel A-90
    - insertion de valeurs de remplacement 23-31
    - OB pour l'exemple d'un processus de mélange industriel A-92
    - valeurs de remplacement 23-31
  - Exemples de projets et de programmes A-77
  - Exigences en matière de sécurité
    - définition pour l'exemple d'un processus de mélange industriel 3-9
  - Exploitation
    - paramètre de sortie RET\_VAL 23-25
  - Exportation
    - source 13-18
  - tables de mnémoniques 8-17
  - Extension d'esclaves DP créés avec des versions antérieures de STEP 7 7-1
- ## F
- Facteur de correction 18-9
  - FB 4-19, 4-20, 4-21, A-43
    - DB générés 16-48
  - FB
    - tableau du format 13-11
  - FB
    - correction de l'interface 15-6
  - FB dans une source LIST
    - exemple 13-25
  - FC 4-17, 4-18
  - FC
    - tableau du format 13-12
  - FC
    - correction de l'interface 15-6
  - FC dans une source LIST
    - exemple 13-23, 13-24
  - FC12 A-101
  - Fenêtre "Partenaires accessibles" 18-2
  - Fenêtre de déclaration des variables
    - saisie de multi-instances 10-10
  - Fenêtre de projet 6-2
  - Fenêtres
    - bascule entre les différents types 5-37
  - FEPROM A-30
  - Fichier \*.awl 6-25
  - Fichier \*.k7e 6-25
  - Fichier \*.k7p 6-25
  - Fichier \*.sdf 6-25
  - Fichier carte mémoire 6-24
  - Fichier de type 7-1
  - Fichier d'exportation S7 6-25
  - Fichier GSD 7-1, 7-2, A-76
  - Fichiers source dans GRAPH 9-8
  - Filtres
    - pour les mnémoniques 8-14
  - Fonction (FC) 4-2, 4-17
  - Fonction (FC)
    - domaine d'application 4-17
  - Fonction (FC)
    - création
      - exemple de FC pour l'exemple d'un processus de mélange industriel A-90
  - Fonction de recherche d'erreurs dans la section des instructions 10-17
  - Fonctionnalités de "Signalisation d'erreurs système" 16-44
  - Fonctionnement avec sauvegarde
    - zones de mémoire rémanentes A-31
  - Fonctionnement sans sauvegarde
    - zones de mémoire rémanentes A-31
  - Fonctions 24-2
  - Fonctions (FC) 4-18
  - Fonctions de diagnostic 23-23
  - Fonctions d'horodatage A-130
  - Fonctions d'information 23-12

Fonctions d'information de la vue du diagnostic 23-8  
 Fonctions d'information de la vue rapide 23-5  
 Fonctions d'information de l'état du module 23-10  
 Fonctions système 4-2, 4-26  
 Fonctions système  
   types 4-26  
 Forçage  
   marche à suivre 20-2  
 Forçage de variables  
   Introduction 20-16  
 Forçage de variables depuis l'éditeur de programmes 26-4  
 Forçage permanent de variables 20-20  
   Introduction 20-20  
 Forçage permanent de variables  
   mesures de sécurité 20-19  
 Format  
   BLOCK A-55  
   COUNTER A-55  
   TIMER A-55  
   type de données DATE\_AND\_TIME (date et heure) A-44  
   type de données DINT (nombres entiers de 32 bits) A-36  
   type de données INT (nombres entiers de 16 bits) A-35  
   type de données Paramètre ANY A-60, A-61, A-62  
   type de données REAL (nombres à virgule flottante) A-37  
   type de données S5TIME (durée) A-42  
   type de données Paramètre POINTER A-55  
   types de données WORD et DWORD pour les nombres décimaux codés binaire A-41  
   types de paramètre BLOCK COUNTER et TIMER A-55  
 Format DCB A-42  
 Format de page  
   définition 24-3  
 Format horaire A-130  
 Format pointeur A-54, A-56  
 Formats de fichier pour l'importation/exportation d'une table des mnémoniques 8-17  
 Formats pour les blocs dans une source LIST 13-10

## G

Génération  
   données de référence 14-10  
   d'une source LIST à partir de blocs 13-17  
 Génération de blocs pour la signalisation d'erreurs système 16-48  
 Gestion de textes utilisateur dont la police de langue n'est pas installée 6-19  
 Gestion multilingue des textes 6-15  
 GRAPH 9-3, 9-8  
 Graphe d'état 9-9  
 Guide de STEP 7 1-1

## H

HALT 21-6  
 Heure  
   modification 4-30  
 Heure  
   réglage A-130  
 Heure  
   lecture A-130  
 Heure (TIME OF DAY)  
   page A-34  
 Heure d'hiver 18-9  
 Heure du module 18-9  
 Heure locale 18-9  
 Hiérarchie d'appel dans le programme utilisateur 4-10  
 Hiérarchie des objets  
   constitution 5-24  
 Hiérarchie d'objets 5-6  
 HiGraph 9-3, 9-9, 9-10  
 Historique des sessions 5-30  
 Horloge  
   paramétrage A-131  
   synchronisation A-130, A-131  
 Horloges CPU avec réglage des zones horaires 18-9  
 Horodatage 18-10  
 Horodatage  
   dans les blocs de code 15-4  
   dans les blocs de données d'instance 15-5  
   dans les blocs de données globaux 15-5  
   dans les UDT et DB repris d'UDT 15-6  
 Horodatage comme propriété de bloc 15-3

## I

Icône de modules inconnus 7-6  
 Icônes de diagnostic  
   dans la vue en ligne 23-3  
 Icônes des objets dans SIMATIC Manager 5-6  
 Identification  
   de mnémoniques 8-4  
 Identification du participant directement connecté à la PG 18-2  
 IM 157 (DP/PA-Link) 23-14  
 Importation  
   source 13-17  
   source externe 6-12  
   table des mnémoniques 8-17  
 Impression  
   blocs 24-1  
   contenu de la mémoire tampon de diagnostic 24-1  
   documentation du projet 24-1  
   données de référence 24-1  
   éléments constitutifs du projet 24-1  
   table de configuration 24-1  
   table des données globales 24-1  
   table des mnémoniques 24-1  
   table des variables 24-1

- Imprimante
    - configuration 24-2
  - IN (déclaration de variables) A-66
  - IN\_OUT (déclaration de variables) A-66
  - Incompatibilité A-76
  - Indicatif
    - dans une bibliothèque de textes 16-30
  - Indirect
    - paramétrage A-127, A-128
  - Informations mnémorique 8-1
  - Informations relatives à la sécurité A-25
    - débordement de la pile L A-25
  - Informations sur la protection d'accès 6-3
  - Informations sur le fichier-journal 6-20
  - Informations sur le test en mode pas à pas et sur les points d'arrêt 21-4
  - Informations sur l'état de fonctionnement "Attente" 21-6
  - Inhibition d'événements d'alarme et d'événements asynchrones
    - exemple A-120
  - Insertion
    - code source de blocs existant dans une source LIST 13-16
    - contenu d'autres source LIST 13-15
    - d'une plage d'opérandes continue dans une table de variables 20-6
    - d'une source externe 13-16
    - modèles de blocs dans une source LIST 13-15
    - opérandes ou mnémoriques dans une table de variables 20-4
    - programme S7/M7 6-12
    - valeurs de forçage 20-6
    - valeurs de remplacement en cas d'erreur détectée 23-31
  - Insertion de lignes de commentaire 20-8
  - Insertion d'une source externe 13-16
  - Insertion d'une station 6-11
  - Insertion d'une variable dans un message 16-28
  - Installation
    - de STEP 7 2-7
  - Installation de Automation License Manager 2-4
  - Installation de STEP 7 2-6
  - Instance 4-22
  - Instruction LIST
    - règles pour la saisie 13-2
  - Instructions
    - saisie
      - marche à suivre 10-12
  - Instructions CONT
    - règles pour la saisie 10-18
  - Instructions dans la vue d'ensemble des éléments de programme 10-5
  - Instructions LIST
    - règles pour la saisie 10-25
  - Instructions LOG
    - règles pour la saisie 10-23
  - INT
    - type de données A-35
  - Interface homme/machine 1-17
  - Interface MPI 2-6
  - Interface PG/PC 2-10
  - Interface PG/PC
    - paramétrage 2-10
  - Interface PG/PC
    - paramétrage 2-10
  - Interface PG/PC
    - paramétrage 2-10
  - Interface PG/PC
    - paramétrage 2-10
  - Interface utilisateur 5-21
  - Interrogation de l'alarme horaire 4-29
  - Intervalle dans la mémoire utilisateur (RAM) 19-20
  - Introduction au forçage permanent de variables 20-20
  - Introduction au test avec des tables de variables 20-1
  - Introduction aux types de données et de paramètre A-33
- J**
- Jeu de paramètres
    - accéder à A-127
  - Journal des modifications 6-3, 6-4
- K**
- k7e 6-25
  - k7p 6-25
- L**
- Langage de programmation
    - CFC 9-11
    - choix 9-2
    - CONT (schéma à contacts) 9-4
    - définition 9-2
    - GRAPH (commande séquentielle) 9-8
    - HiGraph (graphe d'état) 9-9
    - LOG (logigramme) 9-5
    - SCL 9-7
  - Langage de programmation LIST
    - (liste d'instructions) 9-6
  - Langages de programmation 1-6
  - Langues d'affichage 16-31
  - Langues de visuel 16-31
  - Largeur de zone d'opérande 10-18, 10-22
  - Lecture d'un bloc de données dans la mémoire de chargement 6-23
  - Lecture et réglage de l'heure et de l'état de l'heure 18-9
  - Liaison
    - à la CPU 20-13
  - Liaison en ligne 18-3
    - établissement d'une liaison en ligne depuis la fenêtre en ligne du projet 18-3
  - Liaison en ligne
    - établissement depuis la fenêtre "Partenaires accessibles" 18-2

- Liaison en ligne via interface DP 7-3
  - Liaisons en ligne
    - établissement 18-1
  - Licence d'utilisation avec Automation License Manager 2-1
  - License 2-1, 2-2, 2-3
  - License Key 2-1
  - License Manager 2-1, 2-2
  - Ligne de commentaire 20-4
  - Lignes de commentaire
    - insertion 20-8
  - Lignes d'en-tête et de bas de page 24-2
  - Limites supérieures pour la saisie de compteurs 20-8
  - Limites supérieures pour la saisie de temporisations 20-7
  - LIST 9-2, 9-3, 9-6
    - affichage d'informations sur le bloc 14-8
  - LIST
    - saisie de blocs 10-12
  - LIST
    - paramètres 10-25
  - Liste des entrées
    - sorties et entrées/sorties 3-6
  - Liste des références croisée 14-2
  - Liste d'état système
    - contenu 23-20
    - lecture 23-20
  - Liste d'état système (SZL) 23-20
  - Liste d'instructions 9-6
  - Listes de textes destinés à l'utilisateur 16-31
  - Localisation des défauts 23-1
  - LOG 9-5
    - affichage d'informations sur le bloc 14-8
  - Logiciel de base STEP 7 1-6
  - Logiciel optionnel 22-1, 25-3, 25-4, 25-5
  - Logiciel optionnel pour la programmation M7 25-3
  - Logiciels 6-15
  - Logiciels exécutables 1-15
  - Logigramme 9-5
  - Longueur des blocs
    - affichage 9-16
- M**
- M7-300/400
    - systèmes d'exploitation 25-1
  - Majuscules/minuscules pour les mnémoniques 8-15
  - Make (voir Vérifier la cohérence des blocs) 15-1
  - Manipulation
    - objets 5-23, 5-24, 5-25, 5-26, 5-27, 5-28
  - Manuel 1-1, 1-4
  - Marche
    - activités de la CPU A-11
    - état de fonctionnement de la CPU A-1
  - Marche (RUN) A-13
  - Marche à suivre
    - pour l'impression 24-2
  - Marche à suivre
    - affichage et modification de l'état de fonctionnement 18-8
    - pour déterminer la cause d'un passage à l'état d'arrêt 23-16
    - pour la création de blocs de code 10-3
    - pour la saisie d'instructions 10-12
    - pour la visualisation et le forçage 20-2
  - Marche à suivre
    - pour l'archivage/le désarchivage 24-6
  - Marche à suivre
    - pour les systèmes M7 25-1
  - Marche à suivre pour la programmation S7 1-1, 1-3, 1-4
  - Marche à suivre pour les systèmes M7 25-1
  - Marche à suivre pour l'installation de STEP 7 2-7
  - Masquage
    - événements de déclenchement 4-39
  - Masquage d'événements d'erreurs synchrones
    - exemple A-115
  - Masque (voir compilation et chargement d'objets) 19-9
  - Matériel
    - diagnostic 23-1, 23-2
  - Mémento
    - zone de mémoire rémanente A-30
  - Mémentos
    - tableau d'affectation 14-5
  - Mémentos de cadence A-132
  - Mémoire A-32
  - Mémoire
    - configurable A-32
  - Mémoire circulante (mémoire tampon de diagnostic) A-28
  - Mémoire de chargement 19-3, A-15, A-16
  - Mémoire de chargement
    - enregistrement de DB non significatifs pour l'exécution A-16
  - Mémoire de chargement et mémoire de travail A-16
  - Mémoire de chargement et mémoire de travail dans la CPU 19-3
  - Mémoire de chargement/travail
    - effacement 19-18
  - Mémoire de travail 19-3, 19-4, A-15, A-16, A-17
  - Mémoire image
    - entrées/sorties A-21
    - mise à jour 4-13, 4-15
  - Mémoire image
    - effacer 4-35
  - Mémoire image du processus 4-12, A-21, A-22
  - Mémoire image partielle
    - mise à jour par le système A-23
    - mise à jour par SFC A-21
  - Mémoire rémanente
    - des CPU S7-300 A-30
    - des CPU S7-400 A-31
  - Mémoire système A-15
  - Mémoire tampon de diagnostic A-28, A-29
    - contenu A-28, A-29
    - définition A-28
    - exploitation A-28
    - lecture A-29

- Mémoire tampon de diagnostic
  - contenu 23-23
  - lecture 23-19
- Mémoire utilisateur
  - compression 19-20
- Mémoire virtuelle
  - paramétrage 26-5
- Mémoire vive non volatile A-30, A-31
- Message
  - éléments constitutants 16-5
  - exemple 16-6
  - sur bloc 16-12
- Message (pour la CPU)
  - sur mnémotique 16-26
- Message (pour tout le projet)
  - sur mnémotique 16-18
- Message de diagnostic
  - écriture de vos propres messages 23-22
  - envoi aux correspondants 23-22
- Message SCAN
  - voir Message sur mnémotique (pour tout le projet) 16-18
- Message SCAN (pour la CPU)
  - voir Message sur mnémotique 16-26
- Message sur bloc 16-13
- Messages
  - textes tirés de bibliothèques 16-30
- Messages de CPU
  - affichage 16-40
  - configuration 16-41
  - Taille de l'archive 16-38
- Messages de diagnostic personnalisés
  - création et édition 16-19
- Messages de diagnostic utilisateur
  - affichage 16-38
- Messages sur bloc
  - affectation et édition 16-12
- Messages sur bloc (pour la CPU)
  - création 16-23
  - édition 16-24
- Messages sur bloc (pour tout le projet)
  - création 16-15
  - édition 16-16
- Messages sur mnémotique (pour la CPU)
  - affectation à la table des mnémotiques 16-26
  - signaux autorisés 16-26
- Messages sur mnémotique (pour tout le projet)
  - affectation à la table des mnémotiques 16-18
  - signaux autorisés 16-18
- Mesures à prendre dans le programme pour traiter les erreurs 23-24
- Mesures de sécurité pour le forçage permanent de variables 20-19
- Méthodes de conception
  - conception d'un programme structuré A-82
- Micro-carte mémoire (MMC) 6-24, 6-25
- Microprogramme
  - mise à jour 18-10
- Mise à jour 18-10, 18-11, 18-12, A-129, A-130
  - mémoire image A-21
  - mémoire image du processus 4-15
- Mise à jour
  - microprogramme (système d'exploitation) de modules - en ligne 18-10
- Mise à jour
  - microprogramme (système d'exploitation) de modules - hors ligne A-129
- Mise à jour du microprogramme 18-10, 18-11, 18-12
- Mise à jour du système d'exploitation (voir mise à jour en ligne du microprogramme de modules) 18-10
- Mise à jour en ligne du microprogramme de modules 18-10
- Mise à jour hors ligne du microprogramme de modules A-129
- Mise à jour hors ligne du système d'exploitation de modules A-129
- Mise en page CONT 10-18
- Mise en page LOG 10-22
- Mise en place
  - protection d'accès 6-3
- Mise en place d'une protection d'accès 6-3
- Mise en route A-5, A-7, A-9, A-10, A-12
- Mise en route
  - activités de la CPU A-5
  - état de fonctionnement de la CPU A-1
  - interruption A-5
- MMC 6-23, 6-24, 6-25
- Mnémotique
  - affecter A-83
- Mnémotiques 8-15, 8-16
- Mnémotiques
  - définition lors de la saisie du programme 8-13
  - filtres 8-14
  - globaux 8-3
  - locaux 8-3
  - saisie 8-14
  - tri 8-14
- Mnémotiques
  - majuscules/minuscules 8-15
- Mnémotiques
  - majuscules/minuscules 8-15
- Mnémotiques
  - majuscules/minuscules 8-16
- Mnémotiques
  - dans la structure du programme 14-3
- Mnémotiques
  - insertion dans une table de variables 20-4
- Mnémotiques globaux
  - saisie dans la table des mnémotiques 8-14
  - saisie dans un programme 10-13
  - saisie individuelle dans les boîtes de dialogue 8-13
- Mnémotiques globaux et mnémotiques locaux 8-3
- Mnémotiques incomplets ou non univoques dans la table des mnémotiques 8-11
- Mnémotiques manquants 14-8
- Mnémotiques manquants
  - affichage 14-9
- Mode de fonctionnement
  - définition pour le test 21-9

Mode de substitution 10-17  
Mode redondant 18-12  
Modèle de message 16-9, 16-10  
Modèle de message et messages 16-9  
Modèles de blocs  
  insertion dans une source LIST 13-15  
Modification  
  état de fonctionnement 18-8  
  valeurs dans la vue des données de blocs de données 11-8  
Modification de la disposition des fenêtres 5-30  
Modification de l'attribution des numéros de message d'un projet 16-11  
Modification des attributs de contrôle-commande avec CFC 17-5  
Modification d'interfaces 10-27  
Modification du comportement et des propriétés des modules A-127  
Module  
  adresse de début A-122  
  simulation 22-1  
Module de signaux  
  simulation 22-1  
Modules  
  remplacement dans la table de configuration 26-1  
Modules  
  paramétrage A-127  
Modules de signaux aptes aux alarmes de processus  
  paramétrage 4-34  
Modules programmables A-127  
Mot (WORD)  
  page A-34  
  type de données A-33  
Mot de passe 18-6  
Moteurs  
  création du diagramme d'entrées/sorties 3-7  
MPI 18-10, 18-11  
MSK\_FLT 4-40  
Multi-instance 4-19, 4-22  
Multi-instances  
  règles 10-10  
  saisie dans la fenêtre de déclaration des variables 10-10  
  utilisation 10-9  
Multiprojets avec protection d'accès 6-3

## N

Navigateur 5-29  
Nombre à virgule flottante  
  éléments de base A-37  
  exemple A-37, A-38, A-39  
  paramètres A-37  
  zones de composants A-37  
Nombre entier (16 bits)  
  format A-35  
Nombre entier (16 bits) (INT)  
  page A-34

Nombre entier (32 bits)  
  format A-36  
Nombre entier (32 bits) (DINT)  
  page A-34  
Nombre réel  
  page A-34  
  type de données A-34  
Nombres à virgule flottante  
  format A-37, A-38, A-39  
Nombres décimaux codés binaire (DCB) A-41  
Non-Retain 9-14  
Nouveautés dans la version 5.4 de STEP 7 1-9  
Numéros de messages 16-11  
NVRAM A-30

## O

OB 4-3, 4-4, 4-6, 4-7, 4-8  
OB  
  tableau du format 13-10  
OB d'alarme 4-28  
  paramétrage 4-29, 4-30  
  Utilisation 4-28  
OB d'alarme  
  désactivation 4-7  
  paramétrage 4-6  
OB d'alarme de débrogage/enfichage 23-38  
OB d'alarme de diagnostic 23-37, 23-39  
OB dans une source LIST  
  exemple 13-22  
OB d'arrière-plan  
  priorité 4-37  
  programmation 4-38  
OB de défaillance d'unité 23-41  
OB de mise en route 4-35, A-5  
OB de mise en route  
  événements de déclenchement 4-35  
OB de mise en route  
  contrôle des modules 4-36  
OB de mise en route  
  contrôle des modules 4-36  
OB d'erreur 23-26, 23-27  
OB d'erreur  
  types d'OB  
    OB121 et OB122 4-39  
    OB70 et OB72 4-39  
    OB80 à OB87 4-39  
  utilisation d'OB d'erreur en réaction aux événements 4-39  
OB d'erreur d'accès à la périphérie 23-44  
OB d'erreur d'alimentation 23-36  
OB d'erreur de communication 23-42  
OB d'erreur de programmation 23-43  
OB d'erreur de redondance de CPU 23-34  
OB d'erreur de redondance de périphérie 23-33  
OB d'erreur de temps 23-35  
OB d'erreur d'exécution du programme 23-40  
OB d'erreur en réaction à la détection d'une erreur 23-26  
OB d'erreur matérielle CPU 23-39  
OB1 A-112



- OB1 et OB80 A-106
  - OB10 A-103
  - OB100 4-35, A-5
  - OB101 4-35, A-5, A-12
  - OB102 4-35, A-5
  - OB121 23-43
  - OB121 et OB122 4-39
  - OB122 23-44
  - OB20 A-110
  - OB20 à OB23 4-31
  - OB40 à OB47 4-34
  - OB70 23-33
  - OB70 à OB87 4-39
  - OB72 23-34
  - OB80 23-35
  - OB81 23-36
  - OB82 23-37
  - OB83 23-38
  - OB84 23-39
  - OB85 23-40, A-21
  - OB86 23-41
  - OB87 23-42
  - OB90 4-37
  - Objet
    - couper - copier - coller 5-23
    - créer 5-23, 5-24
    - déplacer 5-27
    - effacer 5-23
    - hiérarchie 5-24, 5-25
    - manipuler 5-23
    - ouvrir 5-24, 5-25
    - propriétés 5-24, 5-25, 5-26
    - renommer 5-23
  - Objet
    - sélection 5-29
  - Objet
    - sélection 5-29
  - Objet Bibliothèque 5-8
  - Objet Dossier Blocs 5-15
  - Objet Dossier Sources 5-18
  - Objet générique 7-6, 7-7
  - Objet Module programmable 5-11
  - Objet Programme S7/M7 5-13
  - Objet Projet 5-7
  - Objet Station 5-9
  - Objets 5-6
    - comme supports de propriétés 5-6
  - Objets
    - comme dossiers 5-6
  - Objets
    - comme supports de fonctions 5-6
  - Objets et hiérarchie d'objets 5-6
  - Objets mémoire configurables dans la mémoire de travail A-32
  - Octet
    - comme type de données A-33
    - plage A-34
  - Opérandes
    - insertion dans une table de variables 20-4
    - réassignation 9-20
  - Opérandes et types de données autorisés dans la table des mnémoniques 8-10
  - Opérandes libres 14-7
  - Opérandes libres
    - affichage 14-9
  - Optimisation de la traduction 6-22
  - Optimisation du modèle à traduire 6-21
  - Organisation des zones de mémoire A-15
  - OUT (déclaration de variables) A-66
  - Ouverture
    - table de variables 20-3
    - table des mnémoniques 8-14
- ## P
- Paramétrage
    - avec SFC A-127
    - avec STEP 7 A-128
    - heure A-130, A-131
    - interface PG/PC 2-10
    - mémoire virtuelle 26-5
    - modules de signaux aptes aux alarmes de processus 4-34
  - Paramétrage de blocs de données 12-1
  - Paramétrage de fonctions technologiques 12-2
  - Paramétrage de la langue de Windows 6-8
  - Paramétrage de la signalisation d'erreurs système 16-47
  - Paramétrage de l'interface PG/PC 2-10
  - Paramétrage indirect A-127
  - Paramètre de CPU "Charge du cycle due à la communication" 4-15
  - Paramètre de sortie RET\_VAL
    - exploitation 23-25
  - Paramètre IN\_OUT d'un bloc fonctionnel A-73
  - Paramètres
    - attributs 9-20
      - pour le langage de programmation LIST 10-25
      - pour le langage de programmation LOG 10-22
  - Paramètres de langue de Windows 6-5, 6-6, 6-7
  - Paramètres de modules A-127
  - Paramètres de modules
    - transfert avec SFC A-127
    - transfert avec STEP 7 A-127
  - Paramètres de sortie A-66
  - Paramètres d'entrée/sortie A-66
  - Paramètres d'entrées A-66
  - Paramètres effectifs 4-17
  - Paramètres formels
    - attributs système et blocs de signalisation 16-8
  - Paramètres pour le langage de programmation
    - CONT 10-18
  - Paramètres système A-126
  - PARAM\_MOD A-124, A-127
  - Participant PROFInet 18-2
  - Particularités pour l'impression de l'arborescence des objets 24-3
  - Passage d'un état de fonctionnement à un autre A-1
  - Périphérie
    - zones de données A-122
  - Périphérie décentralisée 7-1, 7-3

- Pile B
  - appels imbriqués A-27
  - données enregistrées dans la pile B A-27
- Pile des blocs A-15, A-27
- Pile des données locales A-15, A-26
- Pile des interruptions A-26
- Pile I
  - description A-26
  - utilisation par la mémoire système A-26
- Pile L A-25
  - allocation de mémoire aux variables locales A-25
  - écraser A-25
  - enregistrement de variables temporaires 4-19
- Pile L
  - traitement de données dans un appel imbriqué A-27
- Piles des interruptions A-15
- Placement
  - boîtes 10-23
- Pointer A-56
- POINTER A-54
- POINTER
  - type de paramètre A-54
- Pointeur A-57, A-58, A-59
- Pointeur zéro A-60, A-61
- Points d'arrêt 21-4
- Positionnement rapide sur les occurrences dans le programme 14-11
- Possibilités d'affichage
  - pour les messages de CPU et les messages de diagnostic utilisateur 16-38
- Possibilités d'appel de l'état du module 23-9
- Possibilités de chargement A-16
- Possibilités de chargement selon la mémoire de chargement 19-4
- Possibilités de saisie de mnémoniques globaux 8-12
- Possibilités d'enregistrement / archivage 24-5
- Possibilités d'extension du logiciel de base STEP 7 1-11
- Poste d'opération
  - description 3-10
  - pour la CPU 16-11
  - pour le projet 16-11
  - Pour le projet 16-11
- Présélections
  - éditeur LIST 10-4
- Présélections pour l'éditeur de programmes CONT/LOG/LIST 10-4
- Présentation
  - bibliothèques standard 9-23
- Présentation des données de référence possibles 14-1
- Prévention de blessures du personnel 20-20
- Prévention de dommages matériels 20-20
- Principes
  - blocs de données 11-1
- Principes de la programmation dans des sources LIST 13-1
- Principes du concept de signalisation 16-1
- Priorité
  - alarme horaire 4-29
  - alarme temporisée 4-31
- Priorité
  - modification 4-6
- Priorité
  - alarme de processus 4-34
- Priorité
  - alarme de processus 4-34
- Priorité
  - OB d'arrière-plan 4-37
- Priorité de l'opérande (symbolique/absolu) 8-5
- Procédé de numéro de message 16-2
- Procédé de signalisation
  - sélection 16-3
- Procédé de signalisation par bit 16-1, 16-2
- Procédure
  - compression du contenu de la mémoire d'une CPU S7 19-21
- Processus
  - subdivision A-79
- Processus
  - subdivision 3-2
  - subdivision en tâches et zones 3-2
  - subdivision en tâches pour l'exemple d'un processus de mélange industriel 3-2
- PROFIBUS 18-10, 18-11, 18-12
- PROFIBUS DP 7-1, 7-3
- Profondeur d'imbrication 4-10
- Programmation
  - transmission de paramètres 4-19
  - utilisation de blocs de données 4-19
- Programmation
  - OB d'arrière-plan 4-37
- Programmation
  - conception d'un programme structuré A-82
- Programmation
  - FB
    - exemple A-89
- Programmation
  - d'une FC
    - exemple A-90
- Programmation
  - d'un OB1
    - exemple A-94
- Programmation linéaire 4-9
- Programmation structurée 4-3
- Programme
  - choix de la méthode de création 9-1
- Programme CFC 25-1
- Programme de mise en route 4-35
- Programme de simulation 22-1
- Programme M7
  - insertion 6-12, 6-13
- Programme S7
  - insertion 6-12
- Programme S7/M7 sans station ni CPU 5-19
- Programme structuré
  - avantages 4-2
  - conception A-82

Programme utilisateur  
 chargement A-16, A-17, A-18  
 dans la mémoire de la CPU A-16  
 éléments 4-2  
 tâches 4-1

Programmes dans une CPU 4-1

Programmes utilisateur  
 chargement dans le système cible 19-3

Projet 5-7

Projet  
 création à l'aide de l'assistant 6-9

Projet  
 création manuelle 6-9

Projet  
 création manuelle 6-9

Projet  
 ouverture 6-14

Projet  
 copie 6-14

Projet  
 copie 6-14

Projet  
 copie 6-14

Projet  
 suppression 6-14

Projet  
 suppression 6-14

Projet  
 suppression 6-14

Projet avec communication par données globales  
 archiver  
 convertir  
 renommer A-76

Projet de version 1  
 conversion A-74

Projet de version 2  
 conversion A-75

Projets  
 renommer 5-26

Projets  
 ordre de traitement 6-9

Projets  
 archivage 24-4

Projets  
 réorganisation 26-2

Projets comportant un grand nombre de stations en  
 réseau 26-1

Projets de la version 2  
 édition 7-1

Projets possédant une protection d'accès 6-3

Projets volumineux 26-1

Propriété de bloc  
 horodatage 15-3

Propriétés de bloc 10-3

Propriétés de bloc  
 affichage de la longueur des blocs 9-16

Propriétés de bloc autorisées pour chaque type de  
 bloc 13-7

Propriétés du dossier Blocs  
 affichage de la longueurs de blocs 9-16

Protection d'accès 6-3

Protection d'accès  
 suppression 6-3

Protection par mot de passe contre l'accès aux  
 systèmes cibles 18-6

Protection par mot de passe contre les accès aux  
 systèmes cible 18-6

## Q

QRY\_TINT 4-29

Quels blocs de signalisation existe-t-il ? 16-6

Quels procédés de signalisation existe-t-il ? 16-1

## R

RAM A-15, A-30

Rapport entre les états de fonctionnement  
 de la CPU A-1

RDSYSST 23-19, 23-20, A-29

READ\_CLK A-130

READ\_RTM A-131

Real  
 type de données A-37

Réassignation  
 blocs 9-20  
 opérandes 9-20

Recherche d'erreurs 23-1

Recherche d'erreurs  
 dans les blocs 10-17

Recherche d'erreurs dans une source LIST 13-19

Recherche des logiciels requis pour un projet 6-15

Redémarrage A-1, A-5, A-6, A-7, A-8, A-9, A-10,  
 A-12

Redémarrage  
 interruption A-5

Redémarrage  
 manuel A-5

Redémarrage  
 manuel A-5

Redémarrage  
 automatique A-6

Redémarrage  
 automatique A-6

Redémarrage  
 automatique A-6

Redémarrage  
 automatique A-6

Redémarrage  
 automatique A-6

Redémarrage  
 automatique A-6

Redémarrage  
 manuel A-6

Redémarrage  
 manuel A-6

Redémarrage  
 manuel A-6

Redémarrage  
 manuel A-6



RPL\_VAL 23-31  
 RUN ("Marche") A-13

## S

### S5 TIME

base de temps A-42  
 format A-42  
 plage A-34

### S5TIME

type de données A-42

### S7-Routing 18-10

### Saisie

commentaires de blocs et de réseaux 10-15  
 de mnémoniques 8-14  
 de mnémoniques globaux individuels dans les  
 boîtes de dialogue 8-13  
 mnémoniques globaux dans un programme 10-13  
 multi-instances dans la table de déclaration des  
 variables 10-10  
 structure de blocs de données associés à un UDT  
 11-7  
 structure de données de blocs de données  
 associés à un FB (DB d'instance) 11-5  
 structure de données de blocs de données  
 globaux 11-4  
 structure de types de données utilisateur (UDT)  
 11-6

Saisie dans les boîtes de dialogue 5-22

Saisie de plusieurs mnémoniques globaux dans la  
 table des mnémoniques 8-14

Saisie du numéro d'identification 2-7

### Sans tension

état de fonctionnement de la CPU A-1

Schéma à contacts 9-4

Schéma de configuration

création 3-11

SCL 9-2, 9-7

sdf 6-25

Section des instructions 10-3, 10-7

Section des instructions

en CONT 10-6

Section des instructions

édition 10-11

Section des instructions

structure 10-11

Section des instructions

fonction de recherche d'erreurs 10-17

Sélection d'objets dans les boîtes de dialogue 5-29

Sélection du procédé de signalisation 16-3

SET\_CLK 4-29, A-130

SET\_CLKS 18-9

SET\_RTM A-131

SET\_TINT 4-29, 4-30

### Setup

paramétrage de la carte mémoire 2-7

saisie du numéro d'identification 2-7

système de fichiers flash 2-9

SFB 4-26, 4-27, A-43

SFB20 STOP 4-12

SFB33 16-6

SFB34 16-6

SFB35 16-6

SFB36 16-6

SFB37 16-6

SFC 4-26, 4-27

utilisation A-21

SFC 55 WR\_PARM A-127

SFC 56 WR\_DPARM A-127

SFC 57 PARM\_MOD A-127

SFC0 SET\_CLK 4-30, A-130

SFC1 READ\_CLK A-130

SFC100 'SET\_CLKS' 18-9

SFC13 DPNRM\_DG A-126

SFC14 DPRD\_DAT A-126

SFC15 DPWR\_DAT A-126

SFC17/18 16-6

SFC2 SET\_RTM A-130

SFC20 BLKMOV A-17

SFC22 CREAT\_DB A-16

SFC26 UPDAT\_PI 4-12, A-21

SFC27 UPDAT\_PO 4-15, A-21

SFC28 SET\_TINT 4-29

SFC28 SET\_TINT

exemple dans LIST A-99

SFC29 CAN\_TINT 4-30

SFC29 CAN\_TINT

exemple dans LIST A-99

SFC3 CTRL\_RTM A-130

SFC30 ACT\_TINT 4-29, 4-30

SFC30 ACT\_TINT

exemple dans LIST A-99

SFC31 QRY\_TINT 4-29

SFC31 QRY\_TINT

exemple dans LIST A-99

SFC32 SRT\_DINT 4-31

SFC32 SRT\_DINT

exemple dans LIST A-108

SFC33 CAN\_DINT

exemple dans LIST A-108

SFC34 QRY\_DINT

exemple dans LIST A-108

SFC36 MSK\_FLT 4-39

SFC36 MSK\_FLT

exemple dans LIST A-115

SFC36 MSK\_FLT

exemple dans CONT A-115

SFC37 DMSK\_FLT 4-39

SFC37 DMSK\_FLT

exemple dans LIST A-115

SFC37 DMSK\_FLT

exemple dans CONT A-115

SFC38 READ\_ERR

exemple dans CONT A-115

exemple dans LIST A-115

SFC39 DIS\_IRT 4-39

SFC39 DIS\_IRT

exemple dans LIST A-120

SFC4 READ\_RTM A-130

SFC40 EN\_IRT 4-39

SFC40 EN\_IRT

exemple dans LIST A-120

- SFC41 DIS\_AIRT 4-39
- SFC41 DIS\_AIRT
  - exemple dans LIST A-121
- SFC42 EN\_AIRT 4-39
- SFC42 EN\_AIRT
  - exemple dans LIST A-121
- SFC44 RPL\_VAL 23-31
- SFC46 STP 4-13
- SFC48 SNC\_RTCB A-130
- SFC51 RDSYSST 23-19, 23-20, A-28
- SFC52 WR\_USMSG 23-22
- SFC55 WR\_PARM A-124
- SFC56 WR\_DPARM A-124
- SFC57 PARM\_MOD A-124
- SFC82 6-23
- SFC83 6-23
- SFC84 6-23
- Signalisation d'erreurs système 16-42, 16-48
  - composants pris en charge 16-44
- SIMATIC Manager 5-1, 9-17, 9-19
- SIMATIC Manager
  - affichage de la longueur des blocs 9-16
- Simple
  - types de données A-34
- Simulation
  - CPU ou module de signaux 22-1
- SlotPLC 6-24
- SNC\_RTCB A-131
- Solution d'automatisation
  - conception 3-1
  - conception
    - création du diagramme d'entrées/sorties pour les moteurs 3-7
  - conception
    - création du diagramme d'entrées/sorties pour les soupapes 3-8
  - conception
    - description des éléments de signalisation et de commande requis 3-10
  - conception
    - création du schéma de configuration 3-11
  - définition des exigences en matière de sécurité 3-9
- Sorties
  - mémoire image A-21, A-22, A-23, A-24
- Sorties
  - listes 3-6
  - tableau d'affectation 14-5
- Soupapes
  - création du diagramme d'entrées/sorties 3-8
- Source
  - exportation 13-18
  - importation 13-17
- Source LIST
  - compilation 13-20
  - création 13-14
  - exemple de FB 13-25
  - exemple de FC 13-23
  - exemple d'OB 13-22
  - exemple d'UDT 13-28
  - exemples de DB 13-27
  - exemples de déclarations de variables 13-21
  - formats pour les blocs 13-10
  - insertion de modèles de blocs 13-15
  - insertion du code source de blocs existant 13-16
  - insertion du contenu d'autres source LIST 13-15
  - recherche d'erreurs 13-19
  - structure des blocs 13-8
  - structure des blocs de code 13-8
  - structure des blocs de données 13-9
  - structure des types de données utilisateur 13-9
  - syntaxe pour les blocs 13-10
  - vérification de la cohérence 13-19
- Source S7
  - édition 13-14
- Sources
  - droits d'accès 10-4
  - enregistrement d'une source LIST 13-19
  - externes 6-13
  - génération d'une source LIST à partir de blocs 13-17
  - insertion d'une source externe 13-16
  - Règles pour la déclaration de variables dans une source LIST 13-3
  - règles pour la définition d'attributs système dans une source LIST 13-4
  - règles pour la définition de propriétés de bloc dans une source LIST 13-5
  - Règles pour la saisie d'instructions dans une source LIST 13-2
  - Règles pour l'ordre des blocs dans une source LIST 13-4
- Sources LIST
  - enregistrement 13-19
  - génération à partir de blocs 13-17
  - insertion d'une source externe 13-16
  - principes de la programmation 13-1
  - règles pour la déclaration de variables 13-3
  - règles pour la définition d'attributs système 13-4
  - règles pour la définition de propriétés de bloc 13-5
  - règles pour l'ordre des blocs 13-4
- SRT\_DINT 4-31
- STAT (déclaration de variables) A-66
- Station 5-9, 5-10
- Station
  - insertion 6-11
- Station
  - insertion 6-11
- Station
  - insertion 6-11
- Station
  - chargement dans la PG 19-14
- Station PC 7-4
- Station SIMATIC PC 7-4, 7-5
- STEP 7
  - lancement du logiciel 5-1
  - langages de programmation 1-6, 1-8, 1-9
  - logiciel de base 1-6, 1-7, 1-8
- STEP 7
  - installation 2-6
- STEP 7
  - erreur durant l'installation 2-7

- STEP 7
    - désinstallation 2-11
  - STEP 7
    - OB d'erreur
      - réaction aux erreurs 4-39
  - STEP 7
    - interface utilisateur 5-21
  - STOP ("Arrêt") A-4
  - STRING A-43
  - STRUCT A-43
  - Structure
    - "alarme horaire" du programme utilisateur A-99
    - création A-50
    - des blocs dans une source LIST 13-8
    - des blocs de code dans une source LIST 13-8
    - des blocs de données dans une source LIST 13-9
    - des types de données utilisateur dans une source LIST 13-9
    - fenêtre 5-21
    - fenêtre de déclaration des variables 10-8
    - liste des références croisées 14-2
    - mémoire de chargement A-16, A-17, A-18
    - section des instructions 10-11
    - UDT 9-13
  - Structure arborescente 14-3
  - Structure de la fenêtre de l'éditeur de programmes 10-1
  - Structure de l'alarme temporisée du programme utilisateur A-108
  - Structure du fichier d'exportation 6-18
  - Structure du programme 14-3
  - Structure du programme
    - affichage 14-9
  - Structure du programme
    - affichage 14-9
  - Structure du programme
    - affichage 14-9
  - Structure du programme
    - affichage 14-9
  - Structure du programme
    - affichage 14-9
  - Structure du projet 6-2
  - Structure et éléments de la table des mnémoniques 8-8
  - Structure hiérarchique des bibliothèques 9-23
  - Subdivision du processus dans l'exemple un processus de mélange industriel 3-2
  - Subdivision du processus en tâches et zones 3-2
  - Subdivision d'un processus en tâches pour l'exemple d'un processus de mélange industriel 3-2
  - Support de données 6-24
  - Suppression de la protection d'accès 6-3
  - Surveillance du processus 20-2
  - Symbolique 8-4
  - Synchronisation
    - horloge A-130, A-131
  - Synchronisation d'horloge 18-9
  - Syntaxe pour les blocs dans une source LIST 13-10
  - Système cible
    - chargement de blocs 19-6
  - Système de fichiers flash 2-7
  - Système d'exploitation
    - tâches 4-1
  - Système d'exploitation de la CPU 4-16
  - Systèmes d'exploitation pour M7-300/400 25-6
  - SZL
    - liste d'état système 23-20, 23-21
- ## T
- Table de déclaration des variables 10-3, 10-6
  - Table de déclaration des variables
    - tâche 10-6
  - Table de déclaration des variables
    - attributs système pour les paramètres 10-7
  - Table de déclaration des variables
    - pour l'OB81 23-26
  - Table de déclaration des variables
    - FB pour l'exemple d'un processus de mélange industriel A-86
  - Table de déclaration des variables
    - FC pour l'exemple d'un processus de mélange industriel A-90
  - Table de déclaration des variables
    - OB pour l'exemple d'un processus de mélange industriel A-92
  - Table de variables
    - copie ou déplacement 20-3
  - Table de variables
    - création et ouverture 20-3
    - enregistrer 20-1
    - utilisation 20-1
  - Table de variables
    - enregistrement 20-4
  - Table de variables
    - édition 20-4
  - Table de variables
    - insertion d'opérandes ou de mnémoniques 20-4
  - Table de variables
    - exemple 20-4
  - Table de variables
    - exemple 20-4
  - Table de variables
    - exemple 20-5
  - Table de variables
    - exemple 20-5
  - Table de variables
    - exemple 20-5
  - Table de variables
    - vérification de la syntaxe 20-6
  - Table de variables
    - vérification de la syntaxe 20-6
  - Table de variables
    - taille maximale 20-6
  - Table de variables
    - taille maximale 20-6
  - Table de variables
    - exemple de saisie d'opérandes 20-9
  - Table des mnémoniques 8-4
    - Structure et éléments 8-8
  - Table des mnémoniques
    - pour mnémoniques globaux 8-8
  - Table des mnémoniques
    - opérandes autorisés 8-10

- Table des mnémoniques
    - types de données autorisés 8-10
  - Table des mnémoniques
    - ouverture 8-14
  - Table des mnémoniques
    - importation/exportation 8-17
  - Table des mnémoniques
    - formats de fichier pour l'importation/exportation 8-17
  - Table des mnémoniques
    - configuration des attributs de contrôle-commande 17-4
  - Table des variables
    - Insertion d'une plage d'opérandes continue 20-6
  - Tableau (type de données ARRAY)
    - description A-47
    - nombre de niveaux imbriqués A-46
  - Tableau des blocs de signalisation 16-6
  - Tableau du format pour les DB 13-13
  - Tableau du format pour les FB 13-11
  - Tableau du format pour les FC 13-12
  - Tableau du format pour les OB 13-10
- Tableaux
- création A-48
  - utilisation pour l'accès aux données A-47
- TEMP (déclaration de variables) A-66
- Temporisation (T)
  - zone de mémoire rémanente A-30
- Temporisations 14-5, 14-6
- Temporisations
  - tableau d'affectation 14-5
- Temporisations
  - limites supérieures pour la saisie 20-7
- Temporisations (T) A-132
- Temps de cycle
  - contrôle pour éviter les erreurs d'horloge 23-18
- Temps de cycle de l'OB1 4-16
- Temps de cycle maximal 4-12
- Temps de cycle minimal 4-14, 4-16
- Temps de référence (voir heure du module) 18-9
- Temps de surveillance 4-36
- Temps de surveillance du cycle 4-12
- Test
  - avec des tables de variables 20-1
  - avec le programme de simulation (logiciel optionnel) 22-1
  - définition du mode de fonctionnement 21-9
- Test à l'aide de la table des variables 26-3
- Test avec la visualisation d'état du programme 21-1
- Test du clignotement 18-2
- Test en mode pas à pas 21-4
- Tester 21-1, 21-2
  - avec la visualisation d'état du programme 21-1
- Tester avec 21-1
- Texte source 13-15
- Texte source
  - définition de la mise en page 13-15
- Textes destinés à l'utilisateur
  - conditions 16-31
  - exportation/importation 16-31
- Textes personnalisés
  - traduction et édition 16-31
- TIMER A-54
- TIMER
  - type de paramètre A-54
- Titres de blocs 10-14
- Titres de réseaux 10-14
- Traduction et édition
  - de textes destinés à l'utilisateur 16-31
- Traitement
  - projet 6-14
- Traitement de programme 4-28
  - déclenché par alarme 4-28
- Traitement d'erreurs 23-24
- Traitement différé d'événements d'alarme et d'événements asynchrones
  - exemple A-121
- Traitement du programme
  - cyclique 4-3, 4-7, 4-8
  - déclenché par alarme 4-3
- Trajet du courant 10-21
- Transfert des données de configuration dans le système cible 16-37
- Transfert des données de configuration dans le système cible de contrôle-commande 17-6
- Transmission au paramètre IN\_OUT d'un FB A-73
- Transmission de paramètres
  - enregistrement des valeurs transmises 4-19
- Transmission de paramètres
  - types de paramètre A-54
- Transmission de paramètres
  - types de paramètre A-54
- Transmission de paramètres
  - exemple de FB pour l'exemple d'un processus de mélange industriel A-86
- Transmission de paramètres
  - conception de paramètres pour un programme structuré A-86
- Transmission d'informations de diagnostic 23-19
- Tri
  - dans la liste des références croisées 14-2
  - mnémoniques 8-14
- Type de déclaration
  - modifier 10-8
- type de données
  - DATE AND TIME\  
Date et heure A-44
- Type de données
  - DWORD A-41
  - S5 TIME A-42
  - UDT 9-13
  - utilisateur 9-13
  - WORD A-41
- Type de données structure (STRUCT)
  - description A-50
  - nombre de niveaux imbriqués A-46
- Type de données utilisateur
  - création A-52, A-53
- Type de paramètre
  - ANY A-54
  - BLOCK\_DB A-54



- BLOCK\_FB A-54
- BLOCK\_FC A-54
- BLOCK\_SDB A-54
- COUNTER A-54
- POINTER A-54
- TIMER A-54
- Types d'alarme 4-3
- Types de données
  - FB
    - SFB 4-19
- Types de données
  - introduction A-33
- Types de données
  - octet A-33
- Types de données
  - BOOL A-33
- Types de données
  - mot A-33
- Types de données
  - double mot A-33
- Types de données
  - octet A-34
- Types de données
  - mot (WORD) A-34
- Types de données
  - double mot (DWORD) A-34
- Types de données
  - nombre entier (16 bits) (INT) A-34
- Types de données
  - nombre entier (32 bits) (DINT) A-34
- Types de données
  - nombre réel (REAL) A-34
- Types de données
  - S5 TIME A-34
- Types de données
  - durée (TIME) A-34
- Types de données
  - heure (TIME OF DAY) A-34
- Types de données
  - caractère (CHAR) A-34
- Types de données
  - description A-34
- Types de données
  - BOOL A-34
- Types de données
  - BOOL A-34
- Types de données
  - date A-34
- Types de données
  - date A-34
- Types de données
  - INT\
    - nombre entier (16 bits) A-35
- Types de données
  - INT\
    - nombre entier (16 bits) A-35
- Types de données
  - DINT\
    - nombre entier (32 bits) A-36
- Types de données
  - DINT\
    - nombre entier (32 bits) A-36
- Types de données
  - DINT\
    - nombre entier (32 bits) A-36
- Types de données
  - REAL
    - nombre à virgule flottante A-37
- Types de données
  - REAL
    - nombre à virgule flottante A-38
- Types de données
  - REAL
    - nombre à virgule flottante A-38
- Types de données
  - REAL
    - nombre à virgule flottante A-38
- Types de données
  - REAL
    - nombre à virgule flottante A-38
- Types de données
  - UDT A-43
- Types de données
  - FB
    - SFB A-43
- Types de données
  - ARRAY A-43
- Types de données
  - DATE\_AND\_TIME A-43
- Types de données
  - définis par l'utilisateur A-43
- Types de données
  - complexes A-43
- Types de données
  - complexes A-43
- Types de données
  - STRING A-43
- Types de données
  - STRUCT A-43
- Types de données
  - complexes A-43
- Types de données
  - complexes A-43
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63
- Types de données
  - type de données Paramètre
    - ANY
      - paramètre A-63



**U**

UDT 9-13, A-43  
 UDT  
   saisie de la structure 11-6  
 UDT  
   correction de l'interface 15-6  
 UDT dans une source LIST  
   exemple 13-28  
 UPDAT\_PI 4-15, A-23  
 UPDAT\_PO 4-12, A-23  
 Utilisation  
   SFC A-22, A-23  
   Type de paramètre POINTER A-56  
 Utilisation  
   bibliothèques 9-21, 9-22  
   zones de mémoire système A-19  
 Utilisation  
   types de données complexes A-46  
 Utilisation  
   de tableaux pour l'accès aux données A-47  
 Utilisation  
   de structures pour l'accès aux données A-50  
 Utilisation  
   de types de données utilisateur pour l'accès aux données A-52  
 Utilisation  
   type de paramètre ANY A-63  
 Utilisation  
   ancien projet A-74  
 Utilisation  
   ancien projet A-74  
 Utilisation  
   ancien projet A-75  
 Utilisation  
   mémentos de cadence et temporisations A-132  
 Utilisation de jeux de caractères de langues étrangères 6-5  
 Utilisation de la déclaration des variables dans les blocs de code 10-6  
 Utilisation de modèles de réseau 10-16  
 Utilisation de multi-instances 10-9  
 Utilisation du clavier 5-32  
 Utilisation du type de données Paramètre POINTER A-56  
 Utilisation d'une micro-carte mémoire (MMC) comme support de données 6-24  
 Utiliser les configurations PC SIMATIC à partir des versions précédentes 7-4

**V**

Valeur de forçage  
   exemples de saisie 20-11  
 Valeur de remplacement  
   utilisation de la SFC44 (RPL\_VAL) 23-31  
 Valeurs  
   modification dans la vue des données de blocs de données 11-8  
   réinitialisation en leur substituant leur valeur initiale 11-8

Valeurs de forçage  
   insertion 20-6  
 Valeurs de forçage permanent  
   exemples de saisie 20-11  
 Validation d'événements d'alarme et d'événements asynchrones  
   exemple A-120  
 Variables 20-16  
   forçage 20-16  
 Variables  
   contrôle-commande 17-1  
   visualisation 20-14  
 Variables temporaires A-66, A-67  
 Vérification  
   cohérence d'une source LIST 13-19  
   données de référence 14-10  
 Vérifier la cohérence des blocs 15-1  
 Visualisation  
   marche à suivre 20-2  
 Visualisation de variables  
   introduction 20-14  
 Visualisation d'état de programme  
   affichage 21-3  
 Visualisation d'état du programme 21-2  
 Visuel  
   langues 16-31  
 Volume d'informations selon le type de module dans l'état du module 23-12  
 Vue de détail des variables 10-7  
 Vue de détail des variables  
   structure 10-8  
 Vue des déclarations de blocs de données 11-2  
 Vue des données de blocs de données 11-3  
 Vue du projet 6-2  
 Vue en ligne  
   icônes de diagnostic 23-3, 23-4

**W**

WinAC 6-24  
 WinLC 6-24  
 WORD  
   type de données A-41  
 WR\_DPARM A-124, A-127  
 WR\_PARM A-124, A-127  
 WR\_USMSG 23-22

**Y**

Y-Link 23-14

**Z**

Zone de combinaison  
   définition 5-22  
 Zone de liste 5-22  
 Zone de mémoire vive A-31  
 Zone EPROM A-16  
 Zone RAM A-16  
 Zones de données rémanentes des CPU S7-300 A-30

- Zones de mémoire A-15
- Zones de mémoire
  - mémoire de chargement A-15
- Zones de mémoire
  - mémoire de travail A-15
- Zones de mémoire
  - mémoire système A-15
- Zones de mémoire
  - mémoire système A-15
- Zones de mémoire
  - mémoire système A-15
- Zones de mémoire
  - mémoire système A-15
- Zones de mémoire
  - particularités pour S7-300 A-16
- Zones de mémoire
  - particularités pour S7-400 A-16
- Zones de mémoire
  - zones d'adresses A-20
- Zones de mémoire
  - mémoire rémanente A-30
- Zones de mémoire rémanentes des CPU S7-400
  - A-31
- Zones d'opérandes
  - description A-19
- Zones horaires 18-9