

Post-processeur PL7

*Pour AUTOMGEN V6 et automates SCHNEIDER
TSX 37 (micro) et TSX 57 (premium)*



CHAPITRE N°1.	1
INTRODUCTION	1
CHAPITRE N°2.	2
LA VERSION 6.3X DU POST-PROCESSEUR PL7.	2
CHAPITRE N°3.	3
INSTALLATION	3
PARAMÉTRAGE	3
CONNEXION ENTRE LE PC ET LES AUTOMATES SCHNEIDER	5
INSTALLATION DU DRIVER DE COMMUNICATION UNITELWAY DE SCHNEIDER	6
CHAPITRE N°4.	7
PRISE EN MAIN (GÉNÉRATION DIRECTE DE FICHIERS BINAIRES).	7
CHAPITRE N°5.	9
PRISE EN MAIN (UTILISATION DE LA PROCÉDURE D'IMPORTATION VERS PL7MICRO OU PL7JUNIOR).	9
CHAPITRE N°6.	12
GÉNÉRALITÉS	12
PREMIÈRE PHASE DE TRADUCTION DU GÉNÉRATEUR DE CODE	12
DEUXIÈME PHASE DE TRADUCTION DU GÉNÉRATEUR DE CODE	12
TROISIÈME PHASE DE TRADUCTION DU GÉNÉRATEUR DE CODE	12
MODULE DE DIALOGUE	13
CHAPITRE N°7.	14
LES QUATRE FICHIERS TEXTES ASSOCIÉS À UNE APPLICATION	14
LA DÉCLARATION UNITAIRE	14
LA DÉCLARATION DE TABLE LINÉAIRE	15
LA DÉCLARATION DE TABLE POUR AFFECTATION AUTOMATIQUE	15
LES TYPES DE VARIABLES AUTOMGEN	15
LE FICHIER MODÈLE « DEFPL7.VUS »	16
LES FICHIERS « .SRT » ET « .END »	17
CHAPITRE N°8.	18
LES DIRECTIVES DE GÉNÉRATION DE CODE	18
;£E : GESTION DE L'ÉTAT DES ÉTAPES DANS LE TEMPS À LA CHARGE DU PROGRAMMEUR	18
;£O : OPTIMISATION DU CODE GÉNÉRÉ	18
;£LINEPERPHRASE= : FIXE LE NOMBRE MAXIMUM DE LIGNES PAR PHRASE	18

; <code>PRAGMA_USEIOBIT</code> : FORCE LE POST-PROCESSEUR À UTILISER DES BITS AUTOMATES ASSOCIÉS À CHAQUE BIT D'E/S.	18
<u>CHAPITRE N°9.</u>	<u>19</u>
PARAMÉTRAGE	19
PARAMÉTRAGE DE L' AUTOMATE	19
LOCALISATION DES ERREURS SIGNALÉES PAR PL7 MICRO OU PL7 JUNIOR	19
PLANTAGE DU LOGICIEL PL7 MICRO OU PL7 JUNIOR	19
GÉNÉRATION DE FICHIERS BINAIRES	19
<u>CHAPITRE N°10.</u>	<u>21</u>
LE MODULE DE DIALOGUE	21
PARAMÉTRAGE DU MODULE DE DIALOGUE	21
<u>CHAPITRE N°11.</u>	<u>22</u>
TECHNIQUES AVANCÉES	22
INSERTION DE CODE PL7 DANS LES APPLICATIONS	22
UTILISATION DE LA TÂCHE RAPIDE ET DES TÂCHES ÉVÉNEMENTIELLES	22
<u>CHAPITRE N°12.</u>	<u>23</u>
LIMITE DE COMPATIBILITÉ	23
<u>CHAPITRE N°13.</u>	<u>24</u>
EXEMPLE COMPLET	24

CHAPITRE N°1.

Introduction

Le post-processeur TSX 37 & 57 est un module logiciel composé de plusieurs fichiers exécutables.

Il traduit les applications développées avec l'atelier logiciel AUTOMGEN en fichiers textes compatibles avec le logiciel PL7 MICRO V1.1, V1.7, V1.8 ou V3.1 ou PL7 JUNIOR V1.7 ou V3.1 de SCHNEIDER et les télécharge dans les automates TSX 37 (MICRO) et TSX 57 (PREMIUM).

La version 6.3x du post-processeurs PL7 permet de générer directement les fichiers binaires pour les automates TSX 37-10, TSX 37-21 et TSX 37-22. La configuration de ces automates nécessite un fichier contenant la configuration généré par PL7 MICRO ou PL7 JUNIOR de SCHNEIDER.

Pour les autres modèles d'automates le logiciel PL7 MICRO¹ version 1.1, version 1.7, version 1.8 ou version 3.1 ou le logiciel PL7 JUNIOR version 1.7 ou version 3.1 de SCHNEIDER sont nécessaires pour compiler les applications générées par AUTOMGEN et pour paramétrer les automates TSX 37 ou TSX 57.

La procédure d'import automatique pour PL7 MICRO 3.1 ou PL7 JUNIOR V3.1 ne fonctionne qu'avec la version 32 bits d'AUTOMGEN.

TSX 37, TSX 57, PL7 MICRO et PL7 JUNIOR sont des marques déposées de SCHNEIDER.

Le post-processeur permet d'effectuer la mise au point des applications depuis l'atelier logiciel AUTOMGEN en donnant accès aux fonctions de visualisation dynamique, de changement d'état et de forçage des variables et de modification du mode de marche.

Pour l'ensemble de cette notice, veuillez noter que PL7ICRO, PL7 JUNIOR, TSX 37, TSX 57, TSX MICRO et TSX PREMIUM sont des marques déposées de SCHNEIDER AUTOMATION.

¹les logiciels PL7 MICRO et PL7 JUNIOR de SCHNEIDER sont soumis aux lois sur le Copyright

CHAPITRE N°2.

La version 6.3x du post-processeur PL7.

Les fonctionnalités suivantes ont été ajoutées à partir de la version 3.0 du post-processeur PL7 :

- Génération automatique de fichiers binaires pour les automates TSX 37-10, TSX 37-21, TSX 37-22. Pour ces trois CPUs la génération des fichiers binaires ne nécessite plus l'utilisation des outils constructeurs (PL7 JUNIOR ou PL7 MICRO). La configuration de l'automate nécessite cependant la génération d'un fichier d'application créé par l'un ou l'autre de ces logiciels. Sur simple demande et en nous communiquant la configuration souhaitée nous vous feront parvenir le fichier de configuration approprié. La génération automatique de fichiers binaires est soumise à certaines restrictions concernant les instructions constructeurs supportées. Veuillez vous reporter au chapitre consacrée à la génération directe de fichiers binaires pour plus de détails.
- Une directive permet de régler le nombre de lignes par phrase. Ceci permet de réduire sensiblement la mémoire consommée par le programme dans l'automate.
- Une procédure d'import automatique est disponible pour la version 3.1 de PL7 MICRO.

CHAPITRE N°3.

Installation

Pour installer le post-processeur PL7 sur votre disque dur, procédez de la façon suivante :

1. Placez la disquette repérée « Post-processeur PL7 » dans le lecteur de disquette A ou B,
2. Lancez l'exécution de « IRAIINST. EXE »,
3. Suivez ensuite les instructions du programme d'installation.

Paramétrage

Une fois l'installation terminée, lancez l'environnement AUTOMGEN suivez les instructions suivantes :

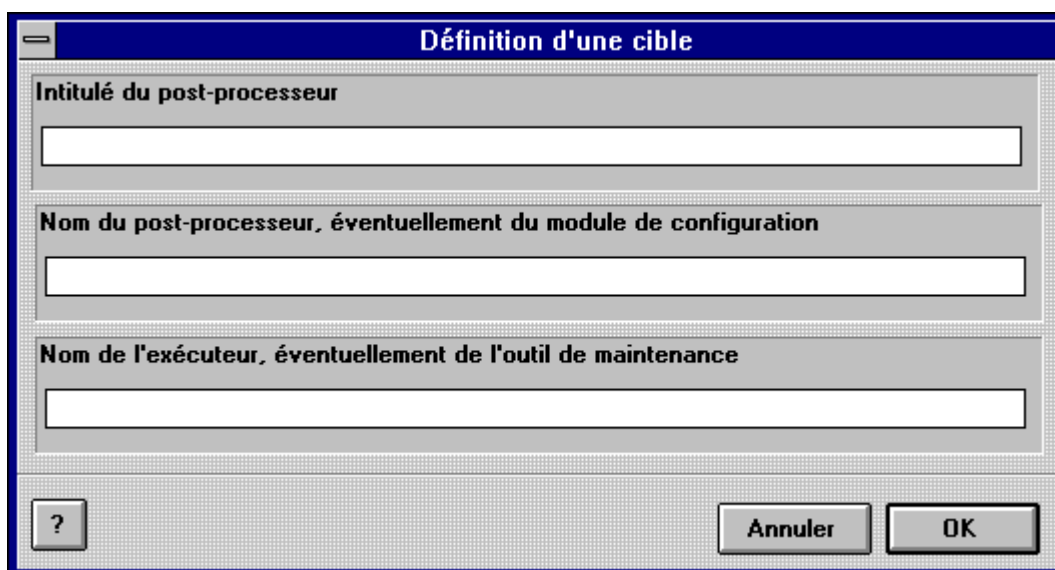


1. Sélectionnez la commande « Cibles ... » du menu « Option »,
2. Choisissez la ligne « PL7 » puis cliquez sur « OK »,
3. Choisissez le menu « Options », « Paramètres par défaut », le bouton poussoir « PL7DRV32.EXE »,
4. La boîte de dialogue de paramétrage du module de dialogue s'ouvre.



1. Sélectionnez la commande « Cibles ... » du menu « Option »,
2. Cliquez sur le bouton « Insérer »,

La fenêtre suivante est alors affichée :



The image shows a Windows-style dialog box titled "Définition d'une cible". It contains three text input fields stacked vertically. The first field is labeled "Intitulé du post-processeur". The second field is labeled "Nom du post-processeur, éventuellement du module de configuration". The third field is labeled "Nom de l'exécuteur, éventuellement de l'outil de maintenance". At the bottom left is a button with a question mark "?". At the bottom right are two buttons labeled "Annuler" and "OK".

3. Entrez sur la première ligne l'intitulé du post-processeur. Cette ligne n'est pas fonctionnelle, elle permettra simplement de sélectionner le post-processeur dans l'environnement. Il est pratique d'entrer un nom générique (« **PL7** » par exemple),
4. Entrez sur la deuxième ligne le nom du générateur de code PL7. Entrez « **PPPL7.EXE** »,
5. Entrez sur la troisième ligne le nom du module de communication.

Si vous souhaitez utiliser le driver de communication UNITELWAY de SCHNEIDER (recommandé):

Entrez « **PL7DRV.EXE** ». Vérifiez que le driver de communication UNITELWAY de SCHNEIDER est correctement installé et configuré.

Si vous ne souhaitez pas utiliser le driver de communication UNITELWAY de SCHNEIDER :

Entrez « **PL7DIAL.EXE** » pour utiliser le port COM1 ou entrez « **PL7DIAL.EXE - C2** » pour utiliser le port COM2. Il faut un et un seul espace entre PL7DIAL.EXE et - C2.

Exemple d'installation du post-processeur PL7 utilisant le driver de communication UNITELWAY de SCHNEIDER:

The screenshot shows a Windows-style dialog box titled "Définition d'une cible". It contains three text input fields. The first field, labeled "Intitulé du post-processeur", contains the text "PL7". The second field, labeled "Nom du post-processeur, éventuellement du module de configuration", contains the text "PPPL7.EXE". The third field, labeled "Nom de l'exécuteur, éventuellement de l'outil de maintenance", contains the text "PL7DRV.EXE". At the bottom left is a button with a question mark "?". At the bottom right are two buttons labeled "Annuler" and "OK".

Exemple d'installation du post-processeur PL7 sans utilisation du driver de communication UNITELWAY de SCHNEIDER. Le driver de communication AUTOMGEN utilise COM2.:

The screenshot shows a similar Windows-style dialog box titled "Définition d'une cible". The first two fields are identical to the previous one: "Intitulé du post-processeur" with "PL7" and "Nom du post-processeur, éventuellement du module de configuration" with "PPPL7.EXE". The third field, labeled "Nom de l'exécuteur, éventuellement de l'outil de maintenance", contains the text "PL7DIAL.EXE -C2". At the bottom left is a button with a question mark "?". At the bottom right are two buttons labeled "Annuler" and "OK".

6. L'installation est terminée, cliquez sur le post-processeur que vous souhaitez utiliser puis sur le bouton « OK ».

Si vous voulez vérifier ou modifier l'installation d'un post-processeur, cliquez sur la ligne correspondante, puis sur le bouton « Modifier ».

Connexion entre le PC et les automates SCHNEIDER

AUTOMGEN utilise le même boîtier de communication TFTXCBPO25 que les logiciels de SCHNEIDER.

Installation du driver de communication UNITELWAY de SCHNEIDER

L'installation du driver UNITELWAY de SCHNEIDER est recommandée pour la version 16 bits d'AUTOMGEN, elle est obligatoire pour la version 32 bits.

Trois versions du driver de communication UNITELWAY SCHNEIDER existe : une pour WINDOWS 3.1, 3.11, une pour WINDOWS 95 et 98 et une pour WINDOWS NT.

Placez, selon la version de WINDOWS que vous utilisez, la disquette repérée « DRIVER UNITELWAY SCHNEIDER WINDOWS 3.1x » ou « DRIVER UNITELWAY SCHNEIDER WINDOWS 9x » ou « DRIVER UNITELWAY SCHNEIDER WINDOWS NT » dans le lecteur de disquette de votre PC et lancez l'exécutable « SETUP.EXE » sur cette disquette. Suivez ensuite les instructions du programme d'installation.

CHAPITRE N°4.

Prise en main (génération directe de fichiers binaires).

Ce chapitre donne le déroulement précis permettant de compiler le fichier d'exemple et de l'exploiter sur un automate SCHNEIDER TSX 37-10, TSX 37-21 ou TSX 37-22 avec le post-processeur PL7.

Si vous utilisez un autre modèle d'automate ou la version 16 bits du post-processeur PL7, passez au chapitre suivant.

Si vous n'avez pas installé le programme d'exemple, relancez la procédure d'installation pour le faire.

1. Lancez l'environnement AUTOMGEN,
2. Cliquez sur l'option « Ouvrir Folio » du menu « Fichier »,
3. A l'aide du sélecteur de fichiers, ouvrez le folio « PL7.GR7 » qui se trouve dans le sous-répertoire « EXPL7 » du répertoire où est installé AUTOMGEN (probablement « C:\AUTOMV6 »²),
4. Cliquez sur l'option « Cibles ... » du menu « Option », sélectionnez la ligne « PL7 » puis le bouton « OK »,
5. Cliquez sur l'option « Editeur de texte » du menu « Boîte à outils »,
6. Dans l'éditeur de texte cliquez sur l'option « Ouvrir » du menu « Fichier » et sélectionnez le fichier « DEFPL7.VSY » se trouvant dans le répertoire où a été installé AUTOMGEN,
7. Dans ce fichier, modifiez si nécessaire la ligne commençant par « \$CONFAPP= » en fonction de l'automate utilisé :

pour le TSX37-10 la ligne doit être
« \$CONFAPP=C:\AUTOMV6\3710.STX »

pour le TSX37-21 la ligne doit être
« \$CONFAPP=C:\AUTOMV6\3721.STX »

pour le TSX37-22 la ligne doit être
« \$CONFAPP=C:\AUTOMV6\3722.STX »
8. Dans ce même fichier, modifiez si nécessaire la ligne commençant par « \$BUILDBIN= » en « \$BUILDBIN=YES ».
9. Dans ce même fichier, modifiez si nécessaire la ligne commençant par « \$RUNPL7SOFT= » en « \$RUNPL7SOFT=NO ».
10. Dans l'éditeur de texte, cliquez sur l'option « Quitter » du menu « Fichier » et répondez « OUI » à la boîte de dialogue « Voulez-vous sauvegarder le fichier ? »,
11. Cliquez sur l'option « Compiler » du menu « Compiler »,

²si le fichier n'existe pas, vous n'avez pas sélectionné l'option « Installer les exemples » dans la procédure d'installation. Dans ce cas, relancez la procédure d'installation.

La compilation est lancée.

Si le post-processeur n'est pas lancé, c'est probablement que l'installation n'a pas été réalisée correctement, reprenez point par point le chapitre « Installation ».

Si tout s'est bien passé, un fichier « PL7.COD » a été généré dans le même répertoire que le fichier « PL7.GR7 ». C'est ce fichier qui sera transféré dans l'automate.

12. Cliquez sur la commande « Charger » du menu « Exécuter »,
13. Une fenêtre « Module de dialogue » s'ouvre et la connexion est réalisée³,
14. Une boîte de dialogue propose de télécharger le programme, cliquez sur « Oui », le téléchargement est alors effectué,
15. Une boîte de dialogue propose de vérifier le programme, cliquez sur « Oui », la vérification est alors effectuée⁴,
16. Cliquez sur la commande « Exécuter » du menu « Exécuter », l'automate passe en RUN,
17. Cliquez sur la commande « Visualiser » du menu « Debug », la visualisation dynamique est active sur le folio « PL7 ».

Ceci termine le chapitre « Prise en Main ».

³si ce n'est pas le cas, vérifiez que l'installation est en accord avec le port de communication utilisé (voir le chapitre « Installation »).

⁴ la vérification permet de contrôler la cohérence entre le programme présent dans l'automate et le programme présent sur le PC.

CHAPITRE N°5.

Prise en main (utilisation de la procédure d'importation vers PL7MICRO ou PL7JUNIOR).

Ce chapitre donne le déroulement précis permettant de compiler le fichier d'exemple et de l'exploiter sur un automate SCHNEIDER.

Si vous n'avez pas installé le programme d'exemple, relancez la procédure d'installation pour le faire.

12. Lancez le logiciel « PL7 MICRO », ou « PL7 JUNIOR »,
13. Créez une nouvelle application,
14. Configurez l'automate : type de CPU, cartes d'entrée/sortie installées, etc... ,
15. Sauvegardez l'application vide ainsi créée sous le nom « AUTOMGEN.STX » dans :
 - le répertoire « C :\ASAW\STXDIR »
(PL7 MICRO V1.1) ,
 - ou
 - le répertoire « C \ASAWUSER\STXDIR »
(PL7 MICRO ou PL7 JUNIOR V1.7, V1.8),
 - ou
 - le répertoire « C \PL7USER »
(PL7 MICRO V3.1 ou PL7 JUNIOR V3.1),
16. Mettez le logiciel « PL7 MICRO » ou « PL7 JUNIOR » en icône,
17. Lancez l'environnement AUTOMGEN,
18. Cliquez sur l'option « Ouvrir Folio » du menu « Fichier »,
19. A l'aide du sélecteur de fichiers, ouvrez le folio « PL7.GR7 » qui se trouve dans le sous-répertoire « EXPL7 » du répertoire où est installé AUTOMGEN (probablement « C:\AUTOMV6 »⁵),
20. Cliquez sur l'option « Cibles ... » du menu « Option », sélectionnez la ligne « PL7 » puis le bouton « OK »,
21. Cliquez sur l'option « Editeur de texte » du menu « Boîte à outils »,
22. Dans l'éditeur de texte cliquez sur l'option « Ouvrir » du menu « Fichier » et sélectionnez le fichier « DEFPL7.VSY » se trouvant dans le répertoire où a été installé AUTOMGEN,

⁵si le fichier n'existe pas, vous n'avez pas sélectionné l'option « Installer les exemples » dans la procédure d'installation. Dans ce cas, relancez la procédure d'installation.

23. Dans ce fichier, modifiez si nécessaire la ligne commençant par « \$CONFAPP= » en fonction du logiciel constructeur utilisé :

pour PL7 MICRO V1.1 la ligne doit être

« \$CONFAPP=C : \ASAW\STXDIR\AUTOMGEN.STX »

pour PL7 MICRO ou PL7 JUNIOR V1.7 ou V1.8 la ligne doit être

« \$CONFAPP=C : \ASAWUSER\STXDIR\AUTOMGEN.STX »

pour PL7 MICRO V3.1 ou PL7 JUNIOR V3.1 la ligne doit être

« \$CONFAPP=C : \PL7USER\AUTOMGEN.STX »

24. Dans ce même fichier, modifiez si nécessaire la ligne commençant par « \$PL7SOFT= » en fonction du logiciel constructeur utilisé :

pour PL7 MICRO V1.1 la ligne doit être

« \$PL7SOFT=TOPL7MI1.EXE »

pour PL7 MICRO V1.7 ou V1.8 la ligne doit être

« \$PL7SOFT=TOPL7MI2.EXE »

pour PL7 MICRO V3.1 la ligne doit être

« \$PL7SOFT=TOPL7MI3.EXE »

pour PL7 JUNIOR V1.7 ou V1.8 la ligne doit être

« \$PL7SOFT=TOPL7JUN.EXE »

pour PL7 JUNIOR V3.1 la ligne doit être

« \$PL7SOFT=TOPL7JU2.EXE »

25. Dans ce même fichier, modifiez si nécessaire la ligne commençant par « \$BUILDBIN= » en « \$BUILDBIN=NO ».

26. Dans ce même fichier, modifiez si nécessaire la ligne commençant par « \$RUNPL7SOFT= » en « \$RUNPL7SOFT=YES ».

27. Dans l'éditeur de texte, cliquez sur l'option « Quitter » du menu « Fichier » et répondez « OUI » à la boîte de dialogue « Voulez-vous sauvegarder le fichier ? »,

28. Cliquez sur l'option « Compiler » du menu « Compiler »,

La compilation est lancée : elle se termine par l'appel du post-processeur PL7 et le lancement du logiciel PL7 MICRO ou PL7 JUNIOR.

Si PL7 MICRO ou PL7 JUNIOR n'est pas lancé, c'est qu'il manque dans le fichier AUTOEXEC.BAT une directive PATH désignant le répertoire où est installé l'un de ces deux logiciels. Il est également possible de laisser l'un de ses deux logiciel ouvert avant le lancement de la compilation. Le post-processeur active alors automatiquement l'un de ces deux logiciels sans essayer des les lancer.

Si le post-processeur n'est pas lancé, c'est probablement que l'installation n'a pas été réalisée correctement, reprenez point par point le chapitre « Installation ».

Si tout s'est bien passé, un fichier « PL7.COD » a été généré dans le même répertoire que le fichier « PL7.GR7 ». C'est ce fichier qui sera transféré dans l'automate.

13. Cliquez sur la commande « Charger » du menu « Exécuter »,
14. Une fenêtre « Module de dialogue » s'ouvre et la connexion est réalisée⁶,
15. Une boîte de dialogue propose de télécharger le programme, cliquez sur « Oui », le téléchargement est alors effectué,
16. Une boîte de dialogue propose de vérifier le programme, cliquez sur « Oui », la vérification est alors effectuée⁷,
17. Cliquez sur la commande « Exécuter » du menu « Exécuter », l'automate passe en RUN,
18. Cliquez sur la commande « Visualiser » du menu « Debug », la visualisation dynamique est active sur le folio « PL7 ».

Ceci termine le chapitre « Prise en Main ».

⁶si ce n'est pas le cas, vérifiez que l'installation est en accord avec le port de communication utilisé (voir le chapitre « Installation »).

⁷ la vérification permet de contrôler la cohérence entre le programme présent dans l'automate et le programme présent sur le PC.

CHAPITRE N°6.

Généralités

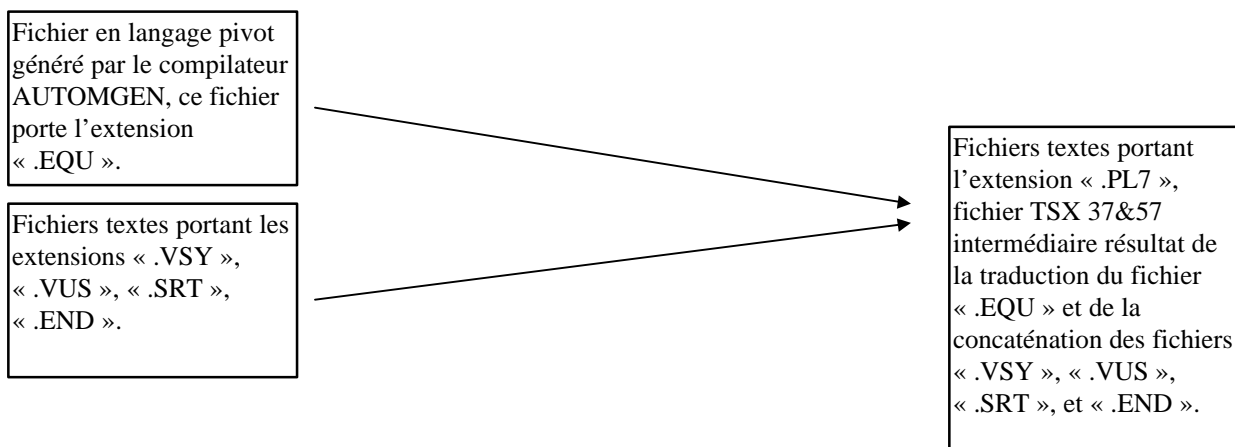
Le post-processeur se compose de différents modules détaillés ci-dessous. Pour rendre les explications plus compréhensibles, seuls les fichiers principaux et accessibles au programmeur seront mentionnés.

Première phase de traduction du générateur de code

« PPPL7.EXE » (version 16 bits) ou « PPPL732.EXE » (version 32 bits) sont les exécutables qui réalisent la première passe de traduction. Il traduisent le langage pivot d'AUTOMGEN représenté par un fichier portant l'extension « .EQU » en fichier pour TSX 37 & 57 intermédiaire qui porte l'extension « .PL7 ». En outre, ces générateurs de code cherchent des fichiers textes qui portent les extensions « .VSY », « .VUS », « .SRT », « .END » et les recopient dans le fichier de sortie.

Ces fichiers sont détaillés dans le chapitre suivant.

Le schéma ci-dessous donne une description de la tâche accomplie par le générateur de code :



Deuxième phase de traduction du générateur de code

« PL7COMP.EXE » (version 16 bits) ou « PL7CO32.EXE » (version 32 bits) sont les exécutables qui réalisent la deuxième phase de traduction. Il traduisent le fichier « .PL7 » en un fichier « .IL » ou « .FEF » compatible avec le logiciel PL7 MICRO ou PL7 JUNIOR. Ces traducteurs effectuent deux tâches : remplacer le nom des variables AUTOMGEN par le nom des variables PL7, et résoudre les sauts dans le programme. Un fichier « .LAP » est également généré, il contient la liste des instructions en langage littéral. A la fin de la traduction, le logiciel PL7 MICRO ou PL7 JUNIOR est lancé pour générer un fichier binaire portant l'extension « .COD ».

Troisième phase de traduction du générateur de code

« BPL716.EXD » (version 16 bits) « BUILDPL7.EXE » (version 32 bits) sont les exécutables qui génèrent les fichiers binaires PL7.

Module de dialogue

« PL7DRV32.EXE » (version 32 bits) ou « WPL7DRV.EXE » (version 16 bits) sont les modules de dialogue utilisant le driver UNITELWAY SCHNEIDER.

« WPL7DIAL.EXE » (version 16 bits uniquement) est le module de dialogue n'utilisant pas le driver UNITELWAY SCHNEIDER.

Ces modules sont utilisés pour les fonctions de téléchargement et de comparaison de programme, de visualisation dynamique, de changement d'état des variables et de forçage des modes de marche de l'automate.

CHAPITRE N°7.

Les quatre fichiers textes associés à une application

Comme il a été dit au chapitre précédent, le générateur de code récupère quatre fichiers textes lors de la première phase de traduction.

Si vous êtes débutant dans l'utilisation d'AUTOMGEN ou du post-processeur TSX 37 & 57, nous vous conseillons d'utiliser ces fichiers sans les modifier. Dans un deuxième temps, et si vous désirez réaliser des applications particulières (optimisation de la taille du programme, ajout de morceaux de programmes en langage TSX 37 & 57), vous pourrez étudier ce chapitre ainsi que le chapitre « Techniques avancées ».

Le générateur de code cherche les fichiers portant l'extension « .VSY », « .VUS », « .SRT » et « .END ».

Pour chacun de ces quatre fichiers, le générateur de code recherche :

- d'abord un fichier portant le même nom que le premier folio de l'application,
- si ce fichier n'existe pas, c'est le fichier portant le nom « DEFPL7 » et qui se trouve dans le répertoire où a été installé AUTOMGEN qui est sélectionné,
- si ni l'un ni l'autre de ces fichiers existent, alors le générateur de code considère cette partie comme vide et ne génère aucun message d'erreur.

Ces fichiers peuvent recevoir des commentaires placés à droite d'un caractère « ; » (point virgule).

Les fichiers « .VSY » et « .VUS » sont des fichiers de déclaration de variables. Ils donnent la correspondance entre les variables AUTOMGEN et les variables PL7.

Le fichier « .VSY » donne la correspondance pour les variables dites « Système ». Ce fichier contient également des directives de compilation (voir chapitre « Paramétrage » et « Techniques avancées »). Le fichier « DEFPL7.VSY » est fourni en modèle dans le répertoire où est installé AUTOMGEN.

Le fichier « .VUS » donne la correspondance pour les variables utilisées dans l'application. Le fichier « DEFPL7.VUS » est fourni en modèle dans le répertoire où est installé AUTOMGEN.

Il existe plusieurs syntaxes permettant de déclarer ces affectations.

La déclaration unitaire

Elle donne la correspondance entre une variable AUTOMGEN et une variable automate.

La syntaxe est :

variable AUTOMGEN = variable automate

Exemple :

#m200=%MW1 ; le mot AUTOMGEN 200 est attribué au mot PL7 %MW1

#i10=%I1.2 ; l'entrée 10 d'AUTOMGEN est attribuée à l'entrée PL7 %I1.2

La déclaration de table linéaire

Elle donne la correspondance entre une série de variables AUTOMGEN et une série de variables AUTOMATE.

La syntaxe est :

longueur de la table : première variable AUTOMGEN = première variable automate

Exemple :

##100:m100=%MW0 ; les mots AUTOMGEN 100 à 199 sont attribués aux mots PL7
; 0 à 99

La déclaration de table pour affectation automatique

Elle laisse le soin au générateur de code d'affecter un ou plusieurs types de variables dans une table de variables automate. Le générateur de code affecte automatiquement les variables suivant ses besoins et si la table n'est pas saturée.

La syntaxe est :

type(s) de variable AUTOMGEN = première variable PL7,numéro de la dernière variable PL7

Un type est représenté par un nom de variable sans numéro. Si plusieurs types sont à préciser, alors ils doivent être séparés par le caractère « & ».

Exemples :

###x&bx=%M0,127 ; affecte automatiquement les bits d'étapes d'AUTOMGEN dans les
; bits PL7 0 à 127

###m&c=%MW1,255 ; affecte les mots et les compteurs d'AUTOMGEN dans les mots
; PL7 0 à 255

Ce type de déclaration est moins prioritaire que les types « # » et « ## ».

Plusieurs déclarations de ce type peuvent être utilisées pour un même type de variable AUTOMGEN. Dans ce cas, le générateur de code remplira dans l'ordre de déclaration et suivant ses besoins, une ou plusieurs tables.

Les types de variables AUTOMGEN

Les types de variables AUTOMGEN sont un sur-ensemble de ceux décrits dans le manuel d'utilisation du module principal dans la partie « Compileur ».

En voici la liste exhaustive :

x	étape, état actuel
bx	étape, état futur
b	bit, état actuel
bb	bit, état futur
u	bit utilisateur, état actuel
bu	bit utilisateur, état futur

i	entrée, état actuel
bi	entrée, état futur
o	sortie, état actuel
bo	sortie, état futur
m	mot
c	compteur
l	long
f	flottant
t	temporisation, état de fin
bt	temporisation, état de lancement
tempo	numéro de temporisation
tconsi	consigne de temporisation
tcompt	compteur de temporisation
ac	accumulateur 16 bits d'AUTOMGEN, variable Système
al	accumulateur 32 bits d'AUTOMGEN, variable Système
af	accumulateur flottant d'AUTOMGEN, variable Système
cf	drapeau de retenue, variable Système
of	drapeau de débordement, variable Système

Pour les variables booléennes d'AUTOMGEN, deux variables booléennes PL7 sont utilisées afin de gérer les états dans le temps.

Le fichier modèle « DEFPL7.VUS »

Détail du fichier « DEFPL7.VUS » présent dans le répertoire où est installé AUTOMGEN.

```
; Fichier .VUS pour TSX-37&57 - AUTOMGEN V6

; E/S : modifier suivant la configuration de l'automate
##32:bi0=%I1.0
##32:o0=%Q2.0

; bits : %M0 à %M255
###i&bo&x&bx&bb&bu&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=%M0,255
; bits : %MW6:X0 à %MW255 :X15
###i&bo&x&bx&bb&bu&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=%MW6:X0,255:X15
; 64 temporisations
##64:tempo=0

; mots, compteurs, compteurs et consignes de temporisations
; %MW256 ... %MW398
; étendre suivant la configuration de l'automate
##m&c&tcompt&tconsi=%MW256,398

; longs : %MD400 à %MD498
###l=%MD400,498

; flottants : %MF500 à %MF510
###f=%MF500,510
```

Définition des E/S

Définition des autres variables booléennes

Définition des temporisations

Définition des mots, compteurs, compteurs et consignes de temporisations

Définition des longs

Définition des flottants

Les fichiers « .SRT » et « .END »

Ces fichiers permettent d'insérer du code PL7 IL au sein d'une application AUTOMGEN.

Le code contenu dans le fichier « .SRT » est exécuté à chaque cycle, avant le code généré par la compilation de l'application issue d'AUTOMGEN.

Le code contenu dans le fichier « .END » est exécuté à chaque cycle, après le code généré par la compilation de l'application issue d'AUTOMGEN.

Le fichier « DEFPL7.SRT » est fourni en modèle dans le répertoire où est installé AUTOMGEN. L'état du bit 0 d'AUTOMGEN est géré dans ce fichier, il doit obligatoirement être à l'état 1 au premier cycle de scrutation de l'application.

La syntaxe utilisée dans ces deux fichiers est celle des fichiers d'import / export du logiciel PL7 MICRO ou PL7 JUNIOR (langage IL). Les compléments de syntaxe suivants sont à prendre en compte :

- il est possible de faire référence aux variables AUTOMGEN en utilisant la syntaxe suivante : « `_variable AUTOMGEN_` ». Par exemple : « `ld _i0_` »,
- un caractère « ' » (apostrophe) placé en début de ligne demande au compilateur de placer la ligne sans modification (commentaires conservés) dans le fichier de sortie,
- trois caractères « ' » (apostrophe) placés en début de ligne demande au compilateur de placer la ligne sans modification (commentaires non conservés) dans le fichier de sortie,
- pour les sauts, il faut utiliser la syntaxe suivante : « `@nom de label` » marque la destination,
« `_nom de label_` » marque le saut, Exemple :

jump %L_fin_

...

@fin

L'examen du fichier portant l'extension « .PL7 » permet de comprendre l'utilisation de ces différentes syntaxes.

CHAPITRE N°8.

Les directives de génération de code

Le générateur de code peut recevoir des directives permettant d'influer sur la forme du code généré. Ces directives sont à écrire dans le fichier « .VSY » de l'application.

Une seule directive peut être écrite par ligne, la forme est la suivante : « ;£x » où x est la directive de compilation à écrire.

;£E : gestion de l'état des étapes dans le temps à la charge du programmeur

Cette directive demande au générateur de code de ne pas générer le code d'évolution des étapes dans le temps. Ce code consiste à recopier les bits bx dans les bits x. Si cette directive est présente, alors il faut écrire le code d'évolution des étapes dans le fichier « .END ».

Cette directive peut être utile pour optimiser le programme généré en réécrivant un code de recopie des bits bx vers x plus efficace (copie de table de bits).

;£O : optimisation du code généré

Cette directive demande au compilateur de produire un code optimisé.

;£LINEPERPHRASE= : fixe le nombre maximum de lignes par phrase

Cette directive permet de placer plus de lignes par phrase. Ceci réduit sensiblement le volume de mémoire consommé par le programme dans l'automate.

;£PRAGMA_USEIOBIT : force le post-processeur à utiliser des bits automates associés à chaque bit d'E/S.

Cette directive peut être utile pour pouvoir accéder de façon standard aux E/S d'un automate à travers un réseau UNITELWAY par exemple.

CHAPITRE N°9.

Paramétrage

Paramétrage de l'automate

Pour paramétrer l'automate, il faut utiliser le logiciel PL7 MICRO ou PL7 JUNIOR de SCHNEIDER.

Par défaut c'est le fichier « C : \ASAW\STXDIR\AUTOMGEN.STX » ou « C : \ASAWUSER\STXDIR\AUTOMGEN.STX » ou « C : \PL7USER\AUTOMGEN.STX » qui sert de modèle aux applications générées. Pour paramétrer l'automate, il faut créer ce fichier avec le logiciel PL7 MICRO ou PL7 JUNIOR, modifier la configuration et sauvegarder le fichier.

Le fichier contenant la configuration ne doit contenir aucun programme.

Un autre fichier peut être utilisé pour stocker la configuration. La directive « \$CONFAPP= » placée dans le fichier « .VSY » désigne le fichier contenant la configuration. Un nom d'accès complet vers le fichier de configuration doit être écrit à la suite de cette directive.

Localisation des erreurs signalées par PL7 MICRO ou PL7 JUNIOR

Lorsqu'une erreur se produit pendant l'importation d'un fichier, le logiciel PL7 MICRO ou PL7 JUNIOR signale la ligne et le type de l'erreur. De telles erreurs peuvent se produire si des variables non déclarées sont utilisées ou si des instructions erronées sont écrites dans des sections de langage constructeur. L'importation manuelle des application permet de localiser plus facilement d'éventuelles erreurs dans le fichier importé.

Plantage du logiciel PL7 MICRO ou PL7 JUNIOR

Il peut arriver que le logiciel PL7 MICRO ou PL7 JUNIOR provoque une erreur fatale à la fin de l'importation automatique. Pour résoudre ce problème il faut lancer le logiciel PL7 MICRO ou PL7 JUNIOR avant de lancer la compilation. Le module d'importation automatique détecte la présence en mémoire de ces logiciels et n'essaie plus ni de les lancer ni de les fermer.

Génération de fichiers binaires

La génération directe de fichier binaire nécessite un fichier généré par PL7 MICRO ou PL7JUNIOR. Ce fichier d'application ne doit contenir aucun programme (pas de code et pas de déclaration de tâche). Les version de CPU suivantes doivent être choisis à la création de ces fichiers : TSX 37-10 V1.0, TSX 37-21 V1.0, TSX 37-22 V1.0.

Les éléments suivants sont récupérés dans le fichier de configuration :

- nombre de variables,
- configuration des cartes d'E/S.

Si vous ne disposez ni du logiciel PL7MICRO ni du logiciel PL7JUNIOR, nous pouvons vous envoyer un fichier de configuration adapté à votre automate. Pour cela précisez nous la configuration souhaitée : type de CPU, cartes d'E/S, etc ...

Les restrictions suivantes existent :

- pas de support des sous-programmes,
- pas de gestion de certaines instructions spécifiques.

Les éléments suivants sont supportés :

- tâche FAST,
- tâches EVT,
- entrées / sorties analogiques.

CHAPITRE N°10.

Le module de dialogue

« PL7DRV32.EXE » (version 32 bits) ou « WPL7DRV.EXE » (version 16 bits) sont les modules de dialogues utilisant le driver de communication UNITELWAY de SCHNEIDER. L'utilisation de ces drivers est recommandée, il faut cependant que le driver de communication SCHNEIDER soit correctement installé et paramétré (voir le chapitre installation).

« WPL7DIAL.EXE » (version 16 bits uniquement) est le module de dialogue n'utilisant le driver de communication UNITELWAY.

Ces modules sont appelés par l'environnement pour le téléchargement et la comparaison des applications, la visualisation dynamique, le changement d'état des variables et le changement d'état des modes de marche, le forçage des variables booléennes.

Le mode pas à pas n'est pas supporté.

Les types de variables PL7 suivants sont visualisables et modifiables : entrées, sorties, bits, bits Système, bits X (visualisables seulement), mots, mots Système, consignes et compteurs de temporisation, longs, flottants.

Le module de dialogue utilise un fichier généré par le générateur de code et qui porte l'extension « .DBG ». Ce fichier donne la correspondance entre les variables AUTOMGEN de l'application et les variables automate.

Paramétrage du module de dialogue



A partir de l'environnement, sélectionnez le menu « Option / Paramètres par défaut » ou « « Option / Paramètres du fichier en cours », le bouton poussoir « PL7DRV32.EXE » puis utilisez la boîte de dialogue de paramétrage.



Les options sont écrites à la suite du nom du module de communication et d'un espace dans la boîte de dialogue de définition d'une cible.

Port de communication (ne concerne que WPL7DIAL.EXE)

-C1 (utilise COM1)

-C2 (utilise COM2)

Paramètres du driver SCHNEIDER

Le paramétrage du driver de communication SCHNEIDER est réalisé en lançant l'exécution de « CNFUTW.EXE » ou « CNFUTW2.EXE » dans le répertoire « ASAWSYS\DRIVERS ».

CHAPITRE N°11.

Techniques avancées

Ce chapitre décrit des possibilités du post-processeur qui permettent de créer des applications plus performantes.

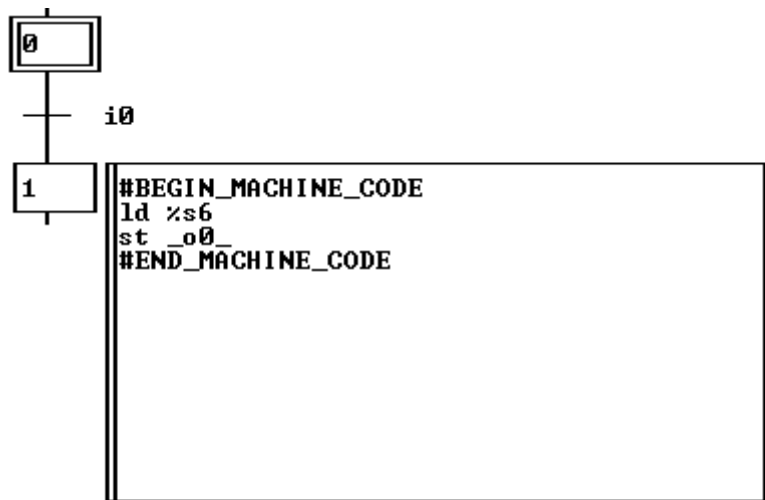
Insertion de code PL7 dans les applications

Comme nous l'avons vu au chapitre 5, les fichiers « .SRT » et « .END » associés à une application peuvent recevoir des lignes de code en langage PL7.

Une autre technique peut être utilisée pour insérer du code PL7 dans une boîte de code AUTOMGEN.

Les directives « #BEGIN_MACHINE_CODE » et « #END_MACHINE_CODE » permettent de définir une section de code constructeur.

Exemple :



Cette technique peut être utilisée pour accéder aux éléments spécifiques à l'automates (les fonctions de communications par exemple).

Utilisation de la tâche rapide et des tâches événementielles

La directive « #In » placée en commentaire sur un folio permet d'associer le code littéral ou le code constructeur écrit sur le folio (dans une boîte de code sous la forme d'un rectangle d'organigramme) à une tâche particulière de l'automate. Le folio ne peut contenir que du code littéral bas niveau ou du code constructeur écrit dans un rectangle d'organigramme. La valeur de n détermine la tâche : si n est égal à 0 c'est la tâche rapide, autrement c'est la tâche événementielle EVT n pour le TSX 37 ou EVT n-1 pour le TSX 57.

CHAPITRE N°12.

Limite de compatibilité

Ce chapitre détaille les limites d'utilisation du post-processeur TSX 37 & 57 par rapport aux possibilités d'AUTOMGEN et des automates TSX 37 & 57.

- les sous-programmes ne sont pas utilisables,
- certaines instructions du langage littéral bas niveau ne sont pas utilisables : JSR, RET.

Si le générateur de code TSX 37 & 57 rencontre une forme de programme intraduisible, alors un message d'erreur de type « combinaison d'instruction et/ou de type d'adressage non supporté à l'adresse XXXX » est généré. XXXX représente une adresse dans le fichier « .EQU » Si vous souhaitez connaître avec précision l'instruction qui a généré cette erreur, utilisez l'utilitaire « CODELIST.EXE⁸ » pour obtenir un listing du fichier « .EQU ».

⁸le chapitre « Techniques avancées » du manuel de l'utilisateur associé au module principal d'AUTOMGEN décrit l'utilisation de l'utilitaire « CODELIST.EXE »

CHAPITRE N°13.

Exemple complet

Ce chapitre contient le dossier de documentation de l'exemple « PL7 ».

Cet exemple est présent dans le sous-répertoire « EXPL7 » du répertoire où est installé AUTOMGEN.

PL7

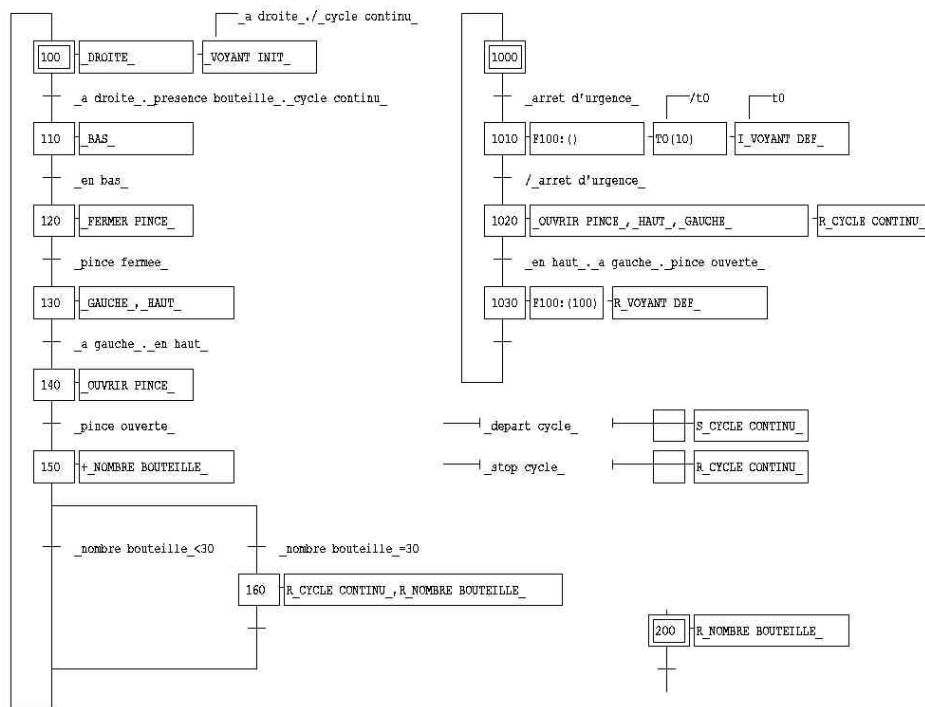


IRAI
BP 14 - 30110 LA GRAND COMBE

PL7

Cible : TSX 37 ou TSX 57
Date de création : 05/09/1998 16:14 02

/* Programme d'exemple */



AUTOMGEN	API N°1	API N°2	SYMBOLS	COMMENTAIRE OU NOM DES FOLIOS OU LA VARIABLE EST UTILISEE
I0000		%I1.0	a droite	capteur verin 1 sorti
I0001		%I1.1	a gauche	capteur verin 1 rentre
I0002		%I1.2	en bas	capteur verin 2 sorti
I0003		%I1.3	en haut	capteur verin 2 rentre
I0004		%I1.4	pince fermee	capteur pince fermee
I0005		%I1.5	pince ouverte	capteur pince ouverte
I0006		%I1.6	presence bouteille	capteur bouteille en attente
I0007		%I1.7	depart cycle	bp depart de cycle
I0008		%I1.8	stop cycle	bp stop
I0009		%I1.9	arret d'urgence	bp arret d'urgence
M0200	%M2		nombre bouteille	nombre de bouteilles passees
O0000	%Q2.0		DROITE	sortir le verin 1
O0001	%Q2.1		GAUCHE	rentre le verin 1
O0002	%Q2.2		BAS	sortir le verin 2
O0003	%Q2.3		HAUT	rentre le verin2
O0004	%Q2.4		FERMER PINCE	fermer la pince
O0005	%Q2.5		OUVRIR PINCE	ouvrir la pince
O0006	%Q2.6		VOYANT INIT	voyant vert
O0007	%Q2.7		VOYANT DEF	voyant rouge
T0000	%M31	%M36		
U0101	%M24	%M25		
U1000	%M24	%M25	cycle continu	etat cycle continu
X0100	%M0	%M1		
X0110	%M2	%M3		
X0120	%M4	%M5		
X0130	%M6	%M7		
X0140	%M8	%M9		
X0150	%M10	%M11		
X0160	%M12	%M13		
X0200	%M14	%M15		
X1000	%M16	%M17		
X1010	%M18	%M19		
X1020	%M20	%M21		
X1030	%M22	%M23		
	%M22	%M23		
PL7 - Page 1 - La liste des références croisées				

Fichier « PL7.PL7 »

```

; ***** Variables systèmes ***** VSY
; pas de fichier pl7.VSY pour l'application, utilisation du fichier
C:\AUTOMV6\defpl7.VSY
; Fichier .VSY pour TSX-37&57 - AUTOMGEN V6

;;f0 ; Active l'optimisation du code généré
;PRAGMA_USEIOBIT ; Utilise des bits images pour les E/S

#cf=%s17
#of=%s18
#ac=%mw0
#al=%md256
#af=%mf512
#b0=%s13

$RUNPL7SOFT=YES ; YES pour lancer automatiquement
PL7MICRO, NO autrement
$CONFAPP=C:\ASAWUSER\STXDIR\AUTOMGEN.STX ; Nom de l'application vide
contenant la configuration de l'api
$PL7SOFT=TOPL7JUN.EXE ; Nom du logiciel d'importation

; Début du fichier d'import
'[HEADER]
'DATE = date #1996-8-3
'STANDARD = 'PLCOpen v0.1 1993'
'SENDER = 'IRAI'
'[APPLICATION]
'NAME = 'IRAI'
'DATE = date #1996-8-2
'VERSION = ''
'[SOURCE_UNIT]
'SU_TYPE = PROG
'NAME = 'MAST_MAIN'
'LANGUAGE = IL
'BODY =
'ADDRESS = MAST MAIN
'VAR_GLOBAL
; ***** Variables utilisateur ***** VUS
; pas de fichier pl7.VUS pour l'application, utilisation du fichier
C:\AUTOMV6\defpl7.VUS
; Fichier .VUS pour TSX-37&57 - AUTOMGEN V6

; E/S : modifier suivant la configuration de l'automate
##16:bi0=%I1.0
##16:o0=%Q2.0
##16:bi16=%I3.0
##16:o16=%Q4.0

; bits : %M0 à %M255 (extension possible vers des bits de mots)
##i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=%M0,25
5
; 64 temporisations
##64:tempo=0

; mots, compteurs, compteurs et consignes de temporisations
; %MW1 à %MW255 (%MW0 est réservé pour _ac_ dans le fichier .VSY)
; étendre suivant la configuration de l'automate
##m&c&t&compt&t&consi=%MW1,255

; longs : %MD258 à %MD511 (%MD256 est réservé pour _al_ dans le fichier
.VSY)
##l=%MD258,511
; flottants : %MF514 à %MF767 (%MF512 est réservé pour _af_ dans le
fichier .VSY)
##f=%MF514,767

```

```

; mots systèmes
#m62=%SW30          ; temps de cycle en ms

; bits systèmes
#b8=%S11            ; débordement chien de garde

; ***** Correspondance bits/étapes
#b100=_x100_
#bb100=_bx100_
#b103=_x110_
#bb103=_bx110_
#b105=_x120_
#bb105=_bx120_
#b107=_x130_
#bb107=_bx130_
#b109=_x140_
#bb109=_bx140_
#b112=_x150_
#bb112=_bx150_
#b115=_x160_
#bb115=_bx160_
#b116=_x200_
#bb116=_bx200_
#b102=_x1000_
#bb102=_bx1000_
#b104=_x1010_
#bb104=_bx1010_
#b106=_x1020_
#bb106=_bx1020_
#b108=_x1030_
#bb108=_bx1030_
; ***** Correspondance bits/bits utilisateur
#b101=_u1000_
#bb101=_bu1000_
''TEMPO_tempo0_ AT %TM_tempo0_:=(TON,100 MS,10,YES);
; ***** Programme de démarrage          ***** SRT
; pas de fichier pl7.SRT pour l'application, utilisation du fichier
C:\AUTOMV6\defpl7.SRT
; Fichier .SRT pour TSX-37&57 - AUTOMGEN V6

'END_VAR
'PROGRAM
; RAZ des bits au premier cycle
'(*PHRASE*)
LD _b0_
[%M0:32:=0]
[%M32:32:=0]
[%M64:32:=0]
[%M96:32:=0]
[%M128:32:=0]
[%M160:32:=0]
[%M192:32:=0]
[%M224:32:=0]
; ***** Prédispositions
'(*END_PHRASE*)
'(*PHRASE*)
LD _b0_
[_tconsi0_:=10]
; ***** Code généré par le post-processeur *****

; ***** Module : PL7.GR7
'(*END_PHRASE*)
'(*PHRASE*)
LD _bi0_

```

```

AND _bi6_
AND _u1000_
R _bx100_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b0_
S _bx100_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x160_
ST _bu99_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x150_
AND [_m200_<30]
ST _bu98_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bu99_
OR _bu98_
S _bx100_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi9_
R _bx1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b0_
S _bx1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x1030_
S _bx1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi2_
R _bx110_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi0_
AND _bi6_
AND _u1000_
ST _bu99_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x100_
AND _bu99_
S _bx110_
' (*END_PHRASE*)
' (*PHRASE*)
LDN _bi9_
R _bx1010_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x1000_
AND _bi9_
S _bx1010_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi4_
R _bx120_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x110_
AND _bi2_
S _bx120_
' (*END_PHRASE*)

```

```

' (*PHRASE*)
LD _bi3_
AND _bi1_
AND _bi5_
R _bx1020_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x1010_
ANDN _bi9_
S _bx1020_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi1_
AND _bi3_
R _bx130_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x120_
AND _bi4_
S _bx130_
' (*END_PHRASE*)
' (*PHRASE*)
LD 1
R _bx1030_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi3_
AND _bi1_
AND _bi5_
ST _bu99_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x1020_
AND _bu99_
S _bx1030_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi5_
R _bx140_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi1_
AND _bi3_
ST _bu99_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x130_
AND _bu99_
S _bx140_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi7_
ST _b110_
' (*END_PHRASE*)
' (*PHRASE*)
LD [_m200_=30]
OR [_m200_<30]
R _bx150_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x140_
AND _bi5_
S _bx150_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi8_
ST _b113_

```

```

' (*END_PHRASE*)
' (*PHRASE*)
LD 1
R _bx160_
' (*END_PHRASE*)
' (*PHRASE*)
LD _x150_
AND [_m200_=30]
S _bx160_
' (*END_PHRASE*)
' (*PHRASE*)
LD 1
R _bx200_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b0_
S _bx200_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bi0_
ANDN _u1000_
AND _bx100_
ST _b117_
' (*END_PHRASE*)
' (*PHRASE*)
LDN _t0_
AND _bx1010_
ST _b119_
' (*END_PHRASE*)
' (*PHRASE*)
LD _t0_
AND _bx1010_
ST _b120_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b118_
R _bx140_
R _bx130_
R _bx120_
R _bx110_
R _bx100_
R _bx160_
R _bx150_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b121_
R _bx140_
R _bx130_
R _bx120_
R _bx110_
S _bx100_
R _bx160_
R _bx150_
; ***** Module : (ACTION)
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx100_
ST _o0_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b117_
ST _o6_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx110_
ST _o2_
' (*END_PHRASE*)

```

```

' (*PHRASE*)
LD _bx1010_
ST _b118_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b119_
ST _bt0_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b120_
JMP CN %L_label10_
' (*END_PHRASE*)
' (*PHRASE*)
LDN _o7_
ST _o7_
' (*END_PHRASE*)
' (*PHRASE*)
@label10

' (*END_PHRASE*)
' (*PHRASE*)
LD _bx120_
ST _o4_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1020_
OR _bx140_
ST _o5_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1020_
OR _bx130_
ST _o3_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1020_
OR _bx130_
ST _o1_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1020_
OR _b113_
OR _bx160_
R _bu1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1030_
ST _b121_
R _o7_
' (*END_PHRASE*)
' (*PHRASE*)
LD _b110_
S _bu1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx150_
[ INC _m200_ ]
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx160_
OR _bx200_
[ _m200_:=0 ]
; ***** Evolution des bits d'étapes
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx100_
ST _x100_

```

```

' (*END_PHRASE*)
' (*PHRASE*)
LD _bx110_
ST _x110_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx120_
ST _x120_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx130_
ST _x130_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx140_
ST _x140_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx150_
ST _x150_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx160_
ST _x160_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx200_
ST _x200_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1000_
ST _x1000_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1010_
ST _x1010_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1020_
ST _x1020_
' (*END_PHRASE*)
' (*PHRASE*)
LD _bx1030_
ST _x1030_
; ***** Evolution des bits utilisateurs
' (*END_PHRASE*)
' (*PHRASE*)
LD _bu1000_
ST _u1000_
; ***** Evolution des temporisations
' (*END_PHRASE*)
' (*PHRASE*)
LD 1
[%TM_tempo0_.P:=_tconsi0_]
[_tcompt0_.:=%TM_tempo0_.V]
' (*END_PHRASE*)
' (*PHRASE*)
LD _bt0_
IN %TM_tempo0_
' (*END_PHRASE*)
' (*PHRASE*)
LD %TM_tempo0_.Q
ST _t0_
' (*END_PHRASE*)
; ***** Programme de fin ***** END
; pas de fichier pl7.END pour l'application, utilisation du fichier
C:\AUTOMV6\defpl7.END

```

```
; Fichier .END pour TSX-37&57 - AUTOMGEN V6  
'END_PROGRAM  
'[EOF]
```

Fichier « .LAP »

```
[HEADER]
DATE = date #1996-8-3
STANDARD = 'PLCOpen v0.1 1993'
SENDER = 'IRAI'
[APPLICATION]
NAME = 'IRAI'
DATE = date #1996-8-2
VERSION = ''
[SOURCE_UNIT]
SU_TYPE = PROG
NAME = 'MAST_MAIN'
LANGUAGE = IL
BODY =
ADDRESS = MAST MAIN
VAR_GLOBAL
TEMPO0 AT %TM0:=(TON,100 MS,10,YES);
END_VAR
PROGRAM
(*PHRASE*)
LD %S13
[%M0:32:=0]
[%M32:32:=0]
[%M64:32:=0]
[%M96:32:=0]
[%M128:32:=0]
[%M160:32:=0]
[%M192:32:=0]
[%M224:32:=0]
(*END_PHRASE*)
(*PHRASE*)
LD %S13
[%MW1:=10]

; MODULE : PL7.GR7

(*END_PHRASE*)
(*PHRASE*)
LD %I1.0
AND %I1.6
AND %M24
R %M1
(*END_PHRASE*)
(*PHRASE*)
LD %S13
S %M1
(*END_PHRASE*)
(*PHRASE*)
LD %M12
ST %M26
(*END_PHRASE*)
(*PHRASE*)
LD %M10
AND [%MW2<30]
ST %M27
(*END_PHRASE*)
(*PHRASE*)
LD %M26
OR %M27
S %M1
(*END_PHRASE*)
(*PHRASE*)
LD %I1.9
R %M17
(*END_PHRASE*)
(*PHRASE*)
```

```

LD %S13
S %M17
(*END_PHRASE*)
(*PHRASE*)
LD %M22
S %M17
(*END_PHRASE*)
(*PHRASE*)
LD %I1.2
R %M3
(*END_PHRASE*)
(*PHRASE*)
LD %I1.0
AND %I1.6
AND %M24
ST %M26
(*END_PHRASE*)
(*PHRASE*)
LD %M0
AND %M26
S %M3
(*END_PHRASE*)
(*PHRASE*)
LDN %I1.9
R %M19
(*END_PHRASE*)
(*PHRASE*)
LD %M16
AND %I1.9
S %M19
(*END_PHRASE*)
(*PHRASE*)
LD %I1.4
R %M5
(*END_PHRASE*)
(*PHRASE*)
LD %M2
AND %I1.2
S %M5
(*END_PHRASE*)
(*PHRASE*)
LD %I1.3
AND %I1.1
AND %I1.5
R %M21
(*END_PHRASE*)
(*PHRASE*)
LD %M18
ANDN %I1.9
S %M21
(*END_PHRASE*)
(*PHRASE*)
LD %I1.1
AND %I1.3
R %M7
(*END_PHRASE*)
(*PHRASE*)
LD %M4
AND %I1.4
S %M7
(*END_PHRASE*)
(*PHRASE*)
LD 1
R %M23
(*END_PHRASE*)
(*PHRASE*)
LD %I1.3

```

```

AND %I1.1
AND %I1.5
ST %M26
(*END_PHRASE*)
(*PHRASE*)
LD %M20
AND %M26
S %M23
(*END_PHRASE*)
(*PHRASE*)
LD %I1.5
R %M9
(*END_PHRASE*)
(*PHRASE*)
LD %I1.1
AND %I1.3
ST %M26
(*END_PHRASE*)
(*PHRASE*)
LD %M6
AND %M26
S %M9
(*END_PHRASE*)
(*PHRASE*)
LD %I1.7
ST %M28
(*END_PHRASE*)
(*PHRASE*)
LD [%MW2=30]
OR [%MW2<30]
R %M11
(*END_PHRASE*)
(*PHRASE*)
LD %M8
AND %I1.5
S %M11
(*END_PHRASE*)
(*PHRASE*)
LD %I1.8
ST %M29
(*END_PHRASE*)
(*PHRASE*)
LD 1
R %M13
(*END_PHRASE*)
(*PHRASE*)
LD %M10
AND [%MW2=30]
S %M13
(*END_PHRASE*)
(*PHRASE*)
LD 1
R %M15
(*END_PHRASE*)
(*PHRASE*)
LD %S13
S %M15
(*END_PHRASE*)
(*PHRASE*)
LD %I1.0
ANDN %M24
AND %M1
ST %M30
(*END_PHRASE*)
(*PHRASE*)
LDN %M31
AND %M19

```

```

ST %M32
( *END_PHRASE* )
( *PHRASE* )
LD %M31
AND %M19
ST %M33
( *END_PHRASE* )
( *PHRASE* )
LD %M34
R %M9
R %M7
R %M5
R %M3
R %M1
R %M13
R %M11
( *END_PHRASE* )
( *PHRASE* )
LD %M35
R %M9
R %M7
R %M5
R %M3
S %M1
R %M13
R %M11

; MODULE : (ACTION)

( *END_PHRASE* )
( *PHRASE* )
LD %M1
ST %Q2.0
( *END_PHRASE* )
( *PHRASE* )
LD %M30
ST %Q2.6
( *END_PHRASE* )
( *PHRASE* )
LD %M3
ST %Q2.2
( *END_PHRASE* )
( *PHRASE* )
LD %M19
ST %M34
( *END_PHRASE* )
( *PHRASE* )
LD %M32
ST %M36
( *END_PHRASE* )
( *PHRASE* )
LD %M33
JMPCN %L0
( *END_PHRASE* )
( *PHRASE* )
LDN %Q2.7
ST %Q2.7
( *END_PHRASE* )
( *PHRASE* )
%L0:
( *END_PHRASE* )
( *PHRASE* )
LD %M5
ST %Q2.4
( *END_PHRASE* )
( *PHRASE* )
LD %M21

```

```

OR %M9
ST %Q2.5
(*END_PHRASE*)
(*PHRASE*)
LD %M21
OR %M7
ST %Q2.3
(*END_PHRASE*)
(*PHRASE*)
LD %M21
OR %M7
ST %Q2.1
(*END_PHRASE*)
(*PHRASE*)
LD %M21
OR %M29
OR %M13
R %M25
(*END_PHRASE*)
(*PHRASE*)
LD %M23
ST %M35
R %Q2.7
(*END_PHRASE*)
(*PHRASE*)
LD %M28
S %M25
(*END_PHRASE*)
(*PHRASE*)
LD %M11
[ INC %MW2 ]
(*END_PHRASE*)
(*PHRASE*)
LD %M13
OR %M15
[ %MW2:=0 ]
(*END_PHRASE*)
(*PHRASE*)
LD %M1
ST %M0
(*END_PHRASE*)
(*PHRASE*)
LD %M3
ST %M2
(*END_PHRASE*)
(*PHRASE*)
LD %M5
ST %M4
(*END_PHRASE*)
(*PHRASE*)
LD %M7
ST %M6
(*END_PHRASE*)
(*PHRASE*)
LD %M9
ST %M8
(*END_PHRASE*)
(*PHRASE*)
LD %M11
ST %M10
(*END_PHRASE*)
(*PHRASE*)
LD %M13
ST %M12
(*END_PHRASE*)
(*PHRASE*)
LD %M15

```

```

ST %M14
( *END_PHRASE* )
( *PHRASE* )
LD %M17
ST %M16
( *END_PHRASE* )
( *PHRASE* )
LD %M19
ST %M18
( *END_PHRASE* )
( *PHRASE* )
LD %M21
ST %M20
( *END_PHRASE* )
( *PHRASE* )
LD %M23
ST %M22
( *END_PHRASE* )
( *PHRASE* )
LD %M25
ST %M24
( *END_PHRASE* )
( *PHRASE* )
LD 1
[ %TM0.P:=%MW1 ]
[ %MW3:=%TM0.V ]
( *END_PHRASE* )
( *PHRASE* )
LD %M36
IN %TM0
( *END_PHRASE* )
( *PHRASE* )
LD %TM0.Q
ST %M31
( *END_PHRASE* )
END_PROGRAM
[ EOF ]

```