

Post-processeur PB

Pour AUTOMGEN V6 et automates PB



CHAPITRE N°1.	1
INTRODUCTION	
CHAPITRE N°2.	2
INSTALLATION	2
PARAMÉTRAGE	2
CONNEXION ENTRE LE PC ET LES AUTOMATES PB	4
CHAPITRE N°3.	5
PRISE EN MAIN	5
CHAPITRE N°4.	7
GÉNÉRALITÉS	7
PREMIÈRE PHASE DE TRADUCTION DU GÉNÉRATEUR DE CODE	7
DEUXIÈME PHASE DE TRADUCTION DU GÉNÉRATEUR DE CODE	7
MODULE DE DIALOGUE	7
CHAPITRE N°5.	8
LES QUATRE FICHIERS TEXTES ASSOCIÉS À UNE APPLICATION	8
LA DÉCLARATION UNITAIRE	8
LA DÉCLARATION DE TABLE LINÉAIRE	9
LA DÉCLARATION DE TABLE POUR AFFECTATION AUTOMATIQUE	9
LES TYPES DE VARIABLES AUTOMGEN	9
LE FICHIER MODÈLE « PB.VUS »	10
LES FICHIERS « .SRT » ET « .END »	11
CHAPITRE N°6.	13
LES DIRECTIVES DE GÉNÉRATION DE CODE	13
;£E : GESTION DE L'ÉTAT DES ÉTAPES DANS LE TEMPS À LA CHARGE DU PROGRAMMEUR	13
CHAPITRE N°7.	14
PARAMÉTRAGE DE L'AUTOMATE	14
DIRECTIVE DE SÉLECTION DU TYPE D'AUTOMATE	14
DIRECTIVES D'ASSEMBLAGE	14
CHAPITRE N°8.	15
LE MODULE DE DIALOGUE	15
PARAMÉTRAGE	15
CHAPITRE N°9.	16

TECHNIQUES AVANCÉES	16
INSERTION DE CODE PB DANS LES APPLICATIONS	16
<u>CHAPITRE N°10.</u>	<u>17</u>
LIMITE DE COMPATIBILITÉ	17
<u>CHAPITRE N°11.</u>	<u>18</u>
EXEMPLE COMPLET	18

CHAPITRE N°1.

Introduction

Le post-processeur PB est un module logiciel composé de plusieurs fichiers exécutables.

Il traduit les applications développées avec l'atelier logiciel AUTOMGEN en fichiers binaires PB et les télécharge dans les automates PB 15, PB 80, PB 100 et PB400.

PB est une marque déposée 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 des variables et de modification du mode de marche.

CHAPITRE N°2.

Installation

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

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

Paramétrage

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

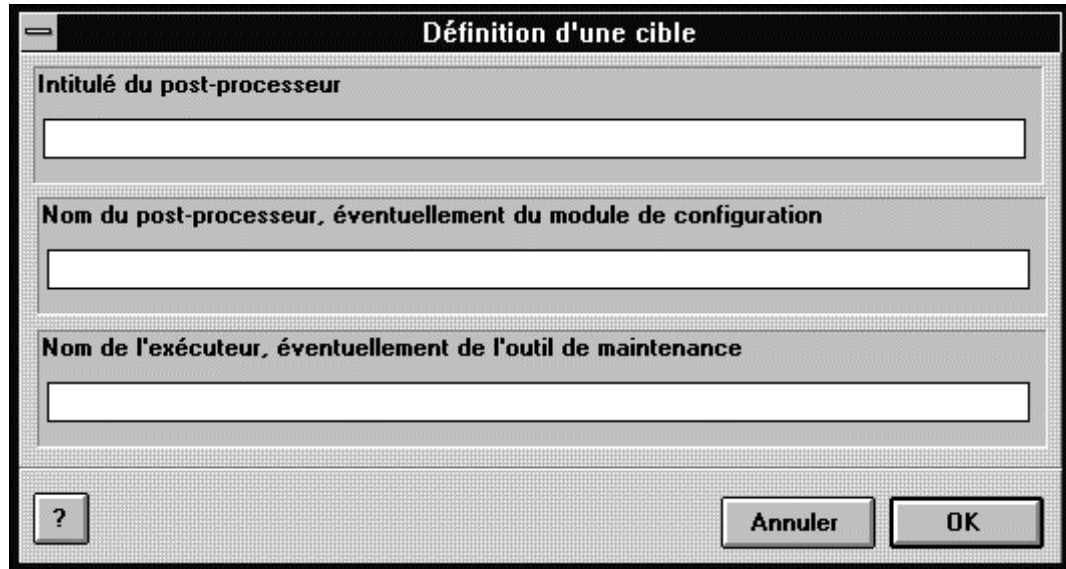


1. Sélectionnez la commande « Cibles ... » du menu « Option »,
2. Choisissez la ligne « PB » puis cliquez sur « OK »,
3. Choisissez le menu « Options », « Paramètres par défaut », le bouton poussoir « PBDIAL32.EXE »,
4. La boîte de dialogue de paramétrage du module de dialogue s'ouvre .



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

La fenêtre suivante est alors affichée sous WINDOWS :



Définition d'une cible

Intitulé du post-processeur

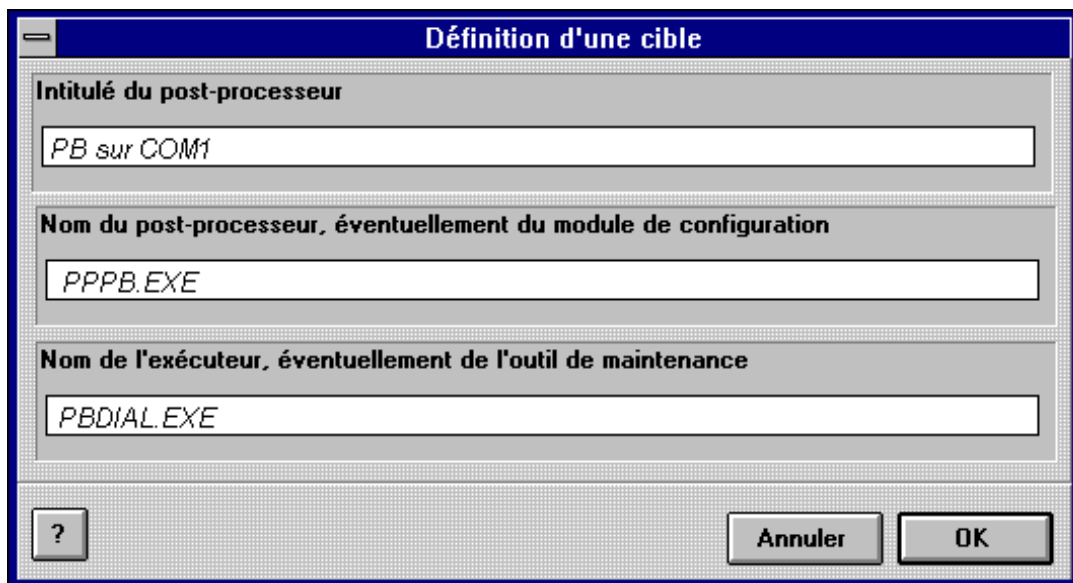
Nom du post-processeur, éventuellement du module de configuration

Nom de l'exécuteur, éventuellement de l'outil de maintenance

? Annuler 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 (« **PB** » par exemple) suivi du port de communication utilisé (« **COM2** » par exemple). Entrez « **PB sur COM1** » ou « **PB sur COM2** »,
4. Entrez sur la deuxième ligne le nom du générateur de code PB. Entrez « **PPPB.EXE** »,
5. Entrez sur la troisième ligne le nom du module de communication. Entrez « **PBDAL.EXE** » pour utiliser le port COM1 ou entrez « **PBDIAL.EXE -C2** » pour utiliser le port COM2. Il faut un et un seul espace entre PBDIAL.EXE et -C2.

Exemple d'installation du post-processeur PB utilisé sur COM1.



Définition d'une cible

Intitulé du post-processeur

PB sur COM1

Nom du post-processeur, éventuellement du module de configuration

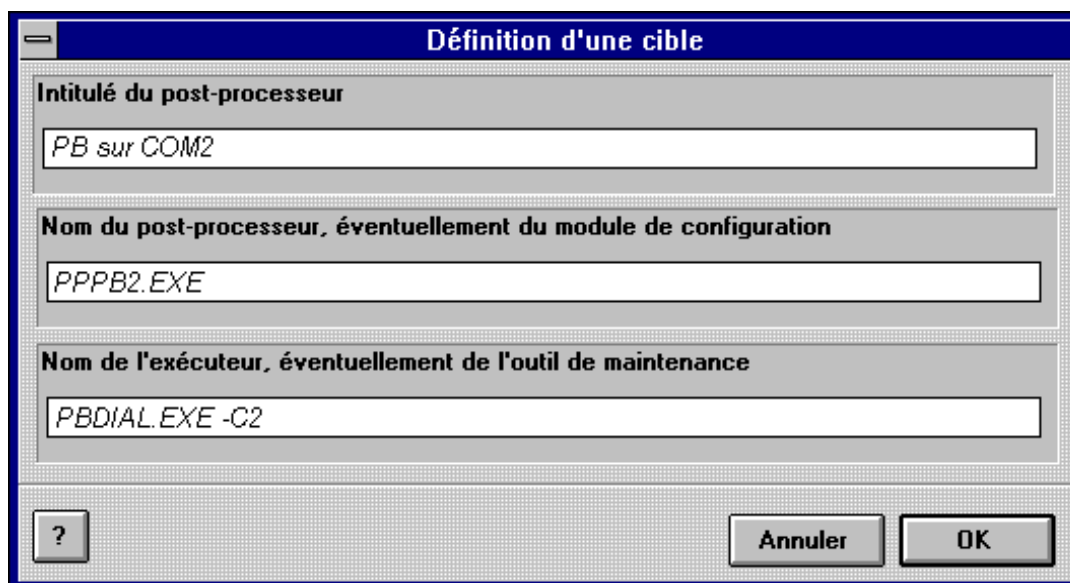
PPPB.EXE

Nom de l'exécuteur, éventuellement de l'outil de maintenance

PBDIAL.EXE

? Annuler OK

Exemple d'installation sur COM2.:



Définition d'une cible

Intitulé du post-processeur
PB sur COM2

Nom du post-processeur, éventuellement du module de configuration
PPPB2.EXE

Nom de l'exécuteur, éventuellement de l'outil de maintenance
PBDIAL.EXE -C2

? Annuler 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 PB

Un boîtier de conversion RS232 vers boucle de courant doit être utilisé. Le câble fourni avec le logiciel SCOLA7 peut également être utilisé avec l'automate PB15.

CHAPITRE N°3.

Prise en main

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

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 « PB.GR7 » qui se trouve dans le sous-répertoire « EXAPIPB » du répertoire où est installé AUTOMGEN,
4. Cliquez sur l'option « Cibles ... » du menu « Option », sélectionnez la ligne « PB sur COM1 » ou « PB sur COM2 »¹ puis le bouton « OK »,
5. Si votre automate est un PB15 passez directement au point numéro 9,
6. Si votre automate est autre qu'un PB15 choisissez la commande « Editeur de texte » du menu « Boîte à outils »,
7. Ouvrez le fichier « DEFPB.VSY » qui se trouve dans le répertoire où est installé AUTOMGEN. Dans ce fichier, modifiez la ligne « \$PB=15 » en « \$PB=80 », « \$PB=100 » ou « \$PB=400 » suivant le type de votre automate.
8. Sauvez le fichier « DEFPB.VSY »,
9. Cliquez sur l'option « Compiler » du menu « Compiler »,

La compilation est lancée : elle se termine par l'appel du post-processeur PB.

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 « PB.COD » a été généré dans le même répertoire que le fichier « PB.GR7 ». C'est ce fichier qui sera transféré dans l'automate.

10. Cliquez sur la commande « Charger » du menu « Exécuter »,
11. Une fenêtre « Module de dialogue PB » s'ouvre et la connexion est réalisée²,
12. Une boîte de dialogue propose de télécharger le programme, cliquez sur « Oui », le téléchargement est alors effectué³,

¹si aucune de ces lignes n'apparaît, l'installation n'a pas été réalisée correctement, retournez au chapitre « Installation ».

²si ce n'est pas le cas, vérifiez que l'installation est en accord avec le port de communication utilisé (voir le chapitre « Installation »).

³ si un message d'erreur « le fichier ne correspond pas au type d'Api » apparaît, c'est que l'automate ne correspond pas au type inscrit dans le fichier « DEFPB.VSY ». Reprenez au point numéro 7.

13. Une boîte de dialogue propose de vérifier le programme, cliquez sur « Oui », la vérification est alors effectuée⁴,
14. Cliquez sur la commande « Exécuter » du menu « Exécuter », l'automate passe en RUN,
15. Cliquez sur la commande « Visualiser » du menu « Debug », la visualisation dynamique est active sur le folio « PB ».

Ceci termine le chapitre « Prise en Main ».

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

CHAPITRE N°4.

Généralités

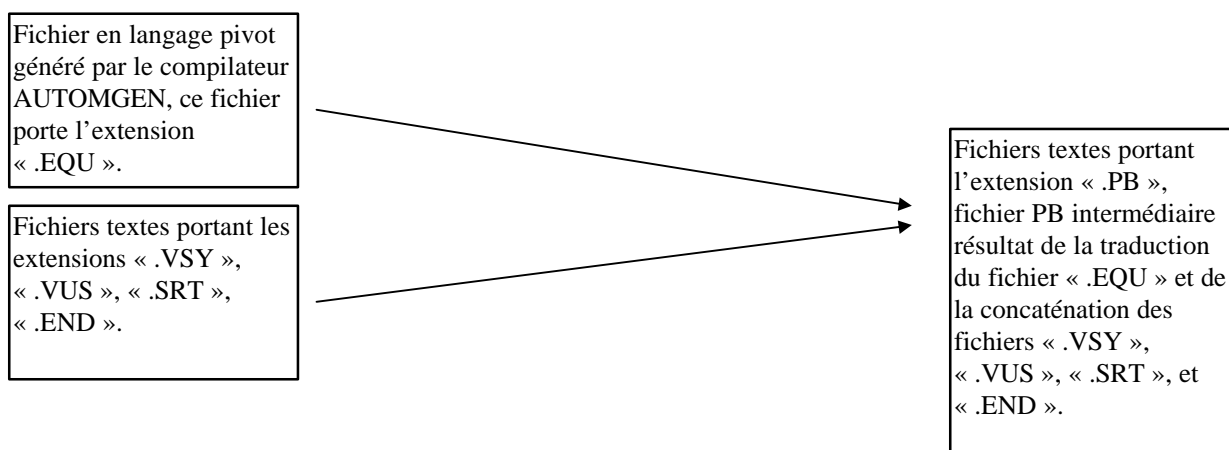
Le post-processeur se compose de différents modules détaillés ci-dessous. Pour rendre les explications plus compréhensibles, seuls les fichiers principaux et accessibles au programmeur seront mentionnés.

Première phase de traduction du générateur de code

« PPPB.EXE » ou « PPPB32.EXE » sont les exécutables qui réalisent la première passe de traduction. Ils traduisent le langage pivot d'AUTOMGEN représenté par un fichier portant l'extension « .EQU » en fichier PB intermédiaire qui porte l'extension « .PB ». En outre, ce générateur de code cherche des fichiers textes qui portent les extensions « .VSY », « .VUS », « .SRT », « .END » et les recopie dans le fichier de sortie s'ils existent.

La création de ces fichiers est à votre charge et sera détaillée dans le chapitre suivant.

Le schéma ci-dessous donne une description de la tâche accomplie par « PPPB.EXE » ou « PPPB32.EXE » :



Deuxième phase de traduction du générateur de code

« PBCOMP.EXE » (16 bits) ou « PBCOMP32.EXE » (32 bits) sont les exécutables qui réalisent la deuxième phase de traduction. Ils traduisent le fichier « .PB » en un fichier « .COD ». Ces traducteurs effectuent deux tâches : remplacer le nom des variables AUTOMGEN par le nom des variables PB et résoudre les sauts dans le programme. Un fichier « .LAP » est également généré, il contient la liste des instructions en langage PB.

Module de dialogue

« WPBDIAL.EXE » (16 bits) ou « PBDIAL32.EXE » (32 bits) sont les modules de dialogue appelés par l'environnement AUTOMGEN. Ils 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°5.

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 PB, 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 PB), vous pourrez étudier ce chapitre ainsi que le chapitre « Techniques avancées ».

Le générateur de code cherche les fichiers portant l'extension « .VSYS », « .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 « DEFBS » 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 « .VSYS » et « .VUS » sont des fichiers de déclaration de variables. Ils donnent la correspondance entre les variables AUTOMGEN et les variables PB.

Le fichier « .VSYS » donne la correspondance pour les variables dites « Système ». Ce fichier contient également des directives de compilation (voir chapitre « Paramétrage » et « Techniques avancées »). Le fichier « DEFBS.VSYS » 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 « DEFBS.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=800 ; le mot AUTOMGEN 200 est attribué au mot PB 800

#i10=a ; l'entrée 10 d'AUTOMGEN est attribuée à l'entrée PB 0A

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 :

##10:m100=80a ; les mots AUTOMGEN 100 à 109 sont attribués aux mots PB
; 800 à 809

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 PB, dernière variable PB

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=a00,a3f ; affecte automatiquement les bits d'étapes d'AUTOMGEN dans les
; bits PB A00 à A3F

###m&c=800,83f ; affecte les mots et les compteurs d'AUTOMGEN dans les mots
; PB 800 à 83f

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 « Compilateur ».

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
t	Temporisation, état de fin
bt	Temporisation, état de lancement
tconsi	Mot de consigne d'une temporisation
tcompt	Mot de comptage d'une temporisation

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

Le fichier modèle « PB.VUS »

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

```
; Fichier .VUS standard pour api PB
; AUTOMGEN V6 - (C)opyright 1999 IRAI

; Définition des variables bits
; Etendre cette zone si nécessaire et en fonction des capacités de votre automate
$IF PB=15
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,a3f
$ENDIF
$IF PB=80
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,aff
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=b00,bff
$ENDIF
$IF PB=OTHER
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,aff
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=b00,bff
$ENDIF

; Définition des mots
$IF PB=15
###m&c&tconsi&tcompt=800,80b
$ENDIF
$IF PB=80
###m&c&tconsi&tcompt=800,89f
$ENDIF
$IF PB=OTHER
###m&c&tconsi&tcompt=700,8df
$ENDIF

; Configuration E/S (à modifier suivant cartes installées)

$IF PB=15
##16:bi0=000
##8:bi16=010
##16:o0=020
$ENDIF
$IF PB=80
; Par exemple 3 cartes de 16 entrées en 030,040 et 050 associées de i0 à i47
; et 2 cartes de 16 sorties en 060 et 070 associées de o0 à o31
##16:bi0=030
##16:bi16=040
##16:bi32=050
##16:o0=060
##16:o16=070
```

Définition des variables booléennes

*Définition des mots ,des compteurs
des consignes et compteurs de
temporisations*

Définition des E/S (PB15)

Définition des E/S (PB80)


```

$ENDIF
$IF PB=OTHER
; Par exemple 3 cartes de 16 entrées en 030,040 et 050 associées de i0 à i47
; et 2 cartes de 16 sorties en 060 et 070 associées de o0 à o31
##16:bi0=030
##16:bi16=040
##16:bi32=050
##16:o0=060
##16:o16=070
$ENDIF

```

Définition des E/S (PB100 et PB40)

Les fichiers « .SRT » et « .END »

Ces fichiers permettent d'insérer du code PB 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 « DEFPB.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 une syntaxe de type assembleur équivalente à celle du langage PB. Les compléments de syntaxe suivants sont à prendre en compte :

- la directive « \$ORG=xxxx » permet de définir le début de l'adresse d'assemblage, au départ l'adresse d'assemblage est fixée à 0C30,

Exemple :

\$ORG=1C30

- la directive « \$TOP=xxx » définit l'adresse maximale pour le saut de page. Elle précise les trois digits de poids faible des adresses au dessus desquelles un saut de page sera automatiquement généré par l'assembleur.
- la directive « \$CONST=xxxx,yyyy » définit l'adresse de départ et de fin pour le stockage des constantes. Les constantes sont logées dans une table en dehors du programme,
- la directive « WORD xxxx » insère la valeur xxxx (quatre digits en hexadécimal) dans le programme,
- la directive « ADR xxxx » insère l'adresse de la variable xxxx (quatre digits en hexadécimal) dans le programme,
- la syntaxe #nnnn permet de faire référence à une valeur constante.

Par exemple :

apl #1234 ; place la constante 1234 (hexadécimal) dans l'accumulateur.

- pour les sauts, il faut utiliser la syntaxe suivante : « @nom de label » marque la destination,
« SAUT _nom de label_ » marque le saut, Exemple :

SAUT _fin_

...

@fin

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

CHAPITRE N°6.

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 copie des bits bx vers x plus efficace (copie de table de bits).

CHAPITRE N°7.

Paramétrage de l'automate

Pour paramétrer l'automate, il faut modifier le fichier texte « .VSX » de l'application. Le fichier « DEFPB.VSX » est utilisé s'il n'y a pas de fichier « .VSX » associé à l'application. Des directives de compilation conditionnelle permettent de sélectionner des sections de programme qui seront compilées en fonction du type d'automate. La syntaxe du début de condition est : « \$IF PB=type d'automate » et se termine par « \$ENDIF ».

Directive de sélection du type d'automate

\$PB=

Valeurs possibles : 15, 80, 100 ou 400.

Directives d'assemblage

\$CONST=<début>,<fin>

Définit les adresses où seront logées les constantes du programme.

\$TOP=<adresse maximale>

Définit les 12 bits de poids faibles de l'adresse au delà de laquelle un saut de page automatique sera généré par le compilateur.

\$ORG=<adresse>

Définit une nouvelle adresse courante.

\$PUSHORG

Sauve l'adresse courante.

\$POPORG

Restaure l'adresse courante.

CHAPITRE N°8.

Le module de dialogue

« WPBDIAL.EXE » (16 bits) et « PBDIAL32.EXE » (32 bits) sont les modules de dialogue 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 mode pas à pas n'est pas supporté par le module de dialogue.

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

Par défaut le module de dialogue utilise le port COM1.

Des paramètres peuvent être ajoutés à la suite du nom du module de communication sur la troisième ligne de définition de la cible dans l'environnement AUTOMGEN. Les paramètres doivent être écrits à la suite d'un espace. Aucun espace ne doit ensuite apparaître, les paramètres sont écrits à la suite les uns des autres.

Modification du port de communication

Le paramètre « -C2 » permet d'utiliser le port COM2 du PC.

Exemples de configuration du module de communication sur le port COM2 :

« PBDIAL.EXE -C2 »

CHAPITRE N°9.

Techniques avancées

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

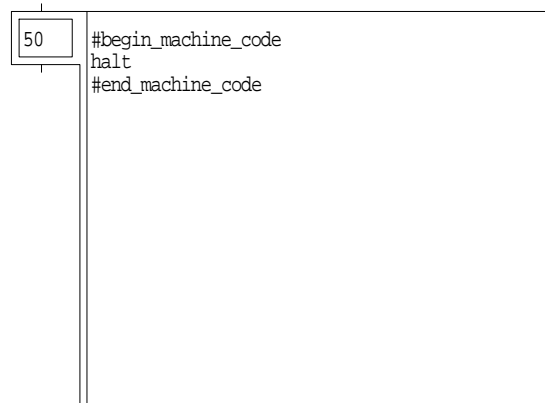
Insertion de code PB dans les applications

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 PB.

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

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

Exemple :



```
50 #begin_machine_code
    halt
    #end_machine_code
```

CHAPITRE N°10.

Limite de compatibilité

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

- l'adressage indirect n'est pas utilisable,
- les sous-programmes ne sont pas utilisables,
- les calculs en virgule flottante ne sont pas utilisables,
- les calculs sur variables 32 bits ne sont pas utilisables,
- les instructions numériques sont réduites à celles disponibles sur l'automate cible,
- certaines instructions du langage littéral bas niveau ne sont pas utilisables : RFZ, RFS ainsi que toutes les instructions relatives aux calculs en virgule flottante.

Si le générateur de code PB 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.EXE⁵ » pour obtenir un listing du fichier « .EQU ».

⁵le chapitre « Techniques avancées » du manuel de l'utilisateur associé au module principal d'AUTOMGEN décrit l'utilisation de l'utilitaire « CODELIST.EXE »

CHAPITRE N°11.

Exemple complet

Ce chapitre contient le dossier de documentation de l'exemple « PB » généré avec la version d'AUTOMGEN sous WINDOWS.

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

PB



I R A I

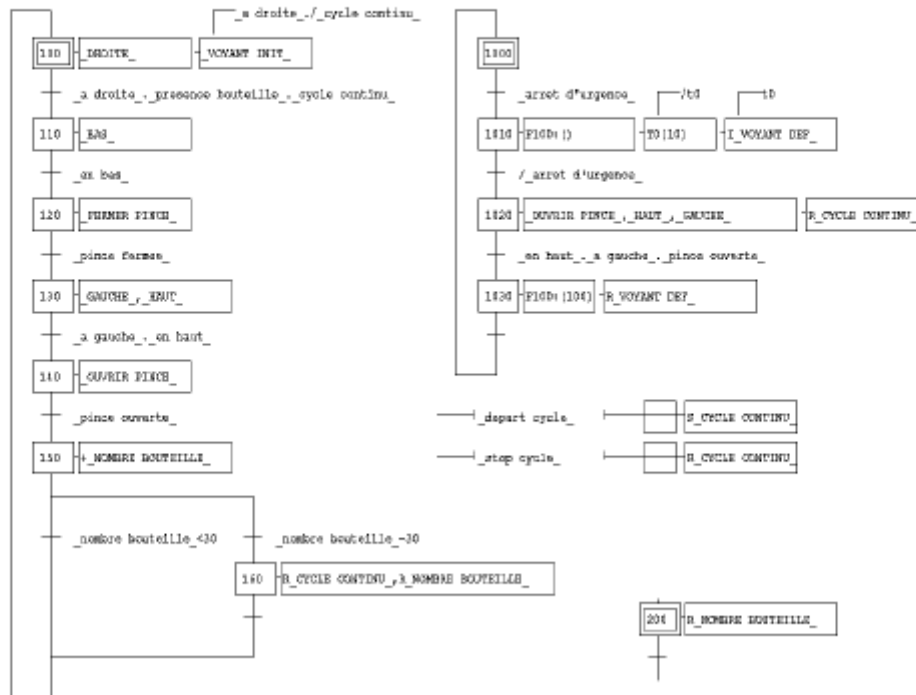
11 rue des Eglantiers - 30110 LA GRAND COMBE

PB

Cible : PB 15

Date de création : 20/08/1999 10:06:16

/* Programme d'exemple */



AUTOMGEN	APT N°1	APT N°2	SCHEMALE	COMMENTAIRES OU NOM DES FEUILLES OU LA VARIABLE EST UTILISEE
I0010		0	a droite	capteur varia 1 sorti
I0011		1	a gauche	capteur varia 1 rentre
I0012		2	en bas	capteur varia 2 sorti
I0013		3	en haut	capteur varia 2 rentre
I0014		4	place fermee	capteur place fermee
I0015		5	place ouverte	capteur place ouverte
I0016		6	presence bouteille	capteur bouteille en attente
I0017		7	depart cycle	bp depart de cycle
I0018		8	stop cycle	bp stop
I0019		9	arrêt d'urgence	bp arrêt d'urgence
N0210	N01		nombre bouteille	nombre de bouteilles passees
O0010	20		sortir	sortir le varia 1
O0011	21		rentrer	rentrer le varia 1
O0012	22		sortir	sortir le varia 2
O0013	23		rentrer	rentrer le varia2
O0014	24		FERMER PENCE	fermer la place
O0015	25		OUVRIR PENCE	ouvrir la place
O0016	26		VAVANT VERT	voyant vert
O0017	27		VAVANT ROGE	voyant rouge
P0010	A23	A25		
U0111	A18	A19		
U1010	A18	A19	cycle cochina	etat cycle cochina
X0110	A03	A01		
X0111	A02	A03		
X0112	A04	A03		
X0113	A06	A07		
X0114	A08	A09		
X0115	A0A	A0A		
X0116	A0C	A00		
X0210	A0E	A0F		
X1010	A10	A11		
X1011	A12	A13		
X1012	A14	A15		
X1013	A16	A17		
	A16	A17		

Fichier « PB.PB »

```
; ***** Variables systèmes ***** VSY
; pas de fichier C:\AUTOMV6\EXAPIPB\PB.VSY pour l'application,
utilisation du fichier C:\AUTOMV6\defpb.VSY
; Déclaration des variables systèmes pour api PB
; AUTOMGEN V6 (C)opyright 1999 IRAI

$PB=15

#egal=9f0
#inf=9f1
#sup=9f2

$IF PB=15
$CONST=080c,817
$ENDIF

$IF PB=80
$CONST=08a0,8ff
#iw=800 ; mot implicite pour les instructions de calcul
$ENDIF

$IF PB=OTHER
$CONST=500,6ff
$TOP=c40
#iw=910 ; mot implicite pour les instructions de calcul
$ENDIF
; ***** Variables utilisateur ***** VUS
; pas de fichier C:\AUTOMV6\EXAPIPB\PB.VUS pour l'application,
utilisation du fichier C:\AUTOMV6\defpb.VUS
; Fichier .VUS standard pour api PB
; AUTOMGEN V6 - (C)opyright 1999 IRAI

; Définition des variables bits
; Etendre cette zone si nécessaire et en fonction des capacités de votre
automate
$IF PB=15
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,a3
f
$ENDIF
$IF PB=80
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,af
f
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=b00,bf
f
$ENDIF
$IF PB=OTHER
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=a00,af
f
###i&bo&x&bx&bb&bu&b&u&t&bt&ux&dx&ui&di&uo&do&du&uu&ub&db&tt&ut&dt=b00,bf
f
$ENDIF

; Définition des mots
$IF PB=15
###m&c&t&consi&t&compt=800,80b
$ENDIF
$IF PB=80
###m&c&t&consi&t&compt=800,89f
$ENDIF
$IF PB=OTHER
###m&c&t&consi&t&compt=700,8df
$ENDIF

; Configuration E/S (à modifier suivant cartes installées)
```

```

$IF PB=15
##16:bi0=000
##8:bi16=010
##16:o0=020
$ENDIF
$IF PB=80
; Par exemple 3 cartes de 16 entrées en 030,040 et 050 associées de i0 à
i47
; et 2 cartes de 16 sorties en 060 et 070 associées de o0 à o31
##16:bi0=030
##16:bi16=040
##16:bi32=050
##16:o0=060
##16:o16=070
$ENDIF
$IF PB=OTHER
; Par exemple 3 cartes de 16 entrées en 030,040 et 050 associées de i0 à
i47
; et 2 cartes de 16 sorties en 060 et 070 associées de o0 à o31
##16:bi0=030
##16:bi16=040
##16:bi32=050
##16:o0=060
##16:o16=070
$ENDIF
; ***** 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_
; ***** Programme de démarrage ***** SRT
; pas de fichier C:\AUTOMV6\EXAPIPB\PB.SRT pour l'application,
utilisation du fichier C:\AUTOMV6\defpb.SRT

$IF PB=15
; raz bits internes
apl #0
rgr c00
rgr c01
rgr c02
rgr c03
mu _b0_
saut a02

```

```

@debut
mz _b0_
$ENDIF

$IF PB=80
$ORG=1C00
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0
word 0

$org=0C30
pred
mu _b0_
saut a02
@debut
mz _b0_
$ENDIF

$IF PB=OTHER
de a00
mz aff
de b00
mz bff
de 1a00
mz laff
de 1b00
mz lbff
; Etendre les initialisations suivant la configuration de votre automate
mu _b0_
saut _cycle_
@debut
$IF PB=400
rafr
mu 010
$ENDIF
mz _b0_
@cycle
$ENDIF

; ***** Prédispositions
$PUSHORG
$ORG=_tconsi0_
word 000a
$POPORG
; ***** Code généré par le post-processeur *****

; ***** Module : C:\AUTOMV6\EXAPIPB\PB.GR7
si _bi0_
si _bi6_
si _u1000_
mz _bx100_
si _b0_

```

```

mu _bx100_
si _x160_
et _bu99_
apl _m200_
cpl #001e
si _inf_
et _b4_
si _x150_
si _b4_
et _bu98_
si _bu99_
si _bu98_
ou _b2_
si _b2_
mu _bx100_
si _bi9_
mz _bx1000_
si _b0_
mu _bx1000_
si _x1030_
mu _bx1000_
si _bi2_
mz _bx110_
si _x100_
si _bi0_
si _bi6_
si _u1000_
mu _bx110_
si/_ _bi9_
mz _bx1010_
si _x1000_
si _bi9_
mu _bx1010_
si _bi4_
mz _bx120_
si _x110_
si _bi2_
mu _bx120_
si _bi3_
si _bi1_
si _bi5_
mz _bx1020_
si _x1010_
si/_ _bi9_
mu _bx1020_
si _bi1_
si _bi3_
mz _bx130_
si _x120_

si _bi4_
mu _bx130_
mz _bx1030_
si _x1020_
si _bi3_
si _bi1_
si _bi5_
mu _bx1030_
si _bi5_
mz _bx140_
si _x130_
si _bi1_
si _bi3_
mu _bx140_
si _bi7_
et _b110_
apl _m200_

```

```

cpl #001e
si _inf_
et _b5_
apl _m200_
cpl #001e
si _egal_
et _b4_
si _b4_
si _b5_
ou _b2_
si _b2_
mz _bx150_
si _x140_
si _bi5_
mu _bx150_
si _bi8_
et _b113_
mz _bx160_
apl _m200_
cpl #001e
si _egal_
et _b4_
si _x150_
si _b4_
mu _bx160_
mz _bx200_
si _b0_
mu _bx200_
si _bi0_
si/ _u1000_
et _b118_
si/ _t0_
et _b121_
si _t0_
et _b123_
si/ _b119_
saut _line99_
mz _bx140_
mz _bx130_
mz _bx120_
mz _bx110_
mz _bx100_
mz _bx160_
mz _bx150_
@line99
si/ _b124_
saut _line108_
mz _bx140_
mz _bx130_
mz _bx120_
mz _bx110_
mu _bx100_
mz _bx160_
mz _bx150_
@line108
; ***** Module : (ACTION)
si _b118_
si _bx100_
et _b117_
si _b121_
si _bx1010_
et _b120_
si _b123_
si _bx1010_
et _b122_
si _bx100_
et _o0_

```



```

si _b117_
et _o6_
si _bx110_
et _o2_
si _bx1010_
et _b119_
si _b120_
et _bt0_
si _b122_
et _b1_
si/ _b1_
saut _label0_
si/ _o7_
et _o7_
@label0
si _bx120_
et _o4_
si _bx1020_
si _bx140_
ou _b2_
si _b2_
et _o5_
si _bx1020_
si _bx130_
ou _b2_
si _b2_
et _o3_
si _bx1020_
si _bx130_
ou _b2_
si _b2_
et _o1_
si _bx1020_
si _b113_
si _bx160_
ou _b2_
si _b2_
mz _bu1000_
si _bx1030_
et _b124_
si _bx1030_
mz _o7_
si _b110_
mu _bu1000_
si _bx150_
ibe _m200_
si _bx160_
si _bx200_
ou _b2_
si _b2_
et _b1_
si/ _b1_
saut _label1_
apl #0000
rgr _m200_
@label1
; ***** Evolution des bits d'étapes
si _bx100_
et _x100_
si _bx110_
et _x110_
si _bx120_
et _x120_
si _bx130_
et _x130_
si _bx140_
et _x140_

```

```

si _bx150_
et _x150_
si _bx160_
et _x160_
si _bx200_
et _x200_
si _bx1000_
et _x1000_
si _bx1010_
et _x1010_
si _bx1020_
et _x1020_
si _bx1030_
et _x1030_
; ***** Evolution des bits utilisateurs
si _bu1000_
et _u1000_
; ***** Evolution des temporisations
si _bt0_
tp _t0_
adr 9fe
adr _tconsi0_
adr _tcompt0_
; ***** Programme de fin ***** END
; pas de fichier C:\AUTOMV6\EXAPIPB\PB.END pour l'application,
utilisation du fichier C:\AUTOMV6\defpb.END
saut _debut_

```

Fichier « PB.LAP »

0C30 : 110C
 0C31 : E400
 0C32 : E401
 0C33 : E402
 0C34 : E403
 0C35 : 431A
 0C36 : F302
 0C37 : 531A
 0800 : 000A

APL #0 (080C)
 RGR C00
 RGR C01
 RGR C02
 RGR C03
 MU A1A
 SAUT A02
 MZ A1A
 WORD 000A

; MODULE : C:\AUTOMV6\EXAPIPB\PB.GR7

0C38 : 8000
 0C39 : 8006
 0C3A : 8318
 0C3B : 5301
 0C3C : 831A
 0C3D : 4301
 0C3E : 830C
 0C3F : 331B
 0C40 : 1101
 0C41 : A10D
 0C42 : 82F1
 0C43 : 331C
 0C44 : 830A
 0C45 : 831C
 0C46 : 331D
 0C47 : 831B
 0C48 : 831D
 0C49 : 231E
 0C4A : 831E
 0C4B : 4301
 0C4C : 8009
 0C4D : 5311
 0C4E : 831A
 0C4F : 4311
 0C50 : 8316
 0C51 : 4311
 0C52 : 8002
 0C53 : 5303
 0C54 : 8300
 0C55 : 8000
 0C56 : 8006
 0C57 : 8318
 0C58 : 4303
 0C59 : 7009
 0C5A : 5313
 0C5B : 8310
 0C5C : 8009
 0C5D : 4313
 0C5E : 8004
 0C5F : 5305
 0C60 : 8302
 0C61 : 8002
 0C62 : 4305
 0C63 : 8003
 0C64 : 8001
 0C65 : 8005
 0C66 : 5315
 0C67 : 8312
 0C68 : 7009
 0C69 : 4315
 0C6A : 8001
 0C6B : 8003
 0C6C : 5307

SI 0
 SI 6
 SI A18
 MZ A01
 SI A1A
 MU A01
 SI A0C
 ET A1B
 APL 801
 CPL #001E (080D)
 SI 9F1
 ET A1C
 SI A0A
 SI A1C
 ET A1D
 SI A1B
 SI A1D
 OU A1E
 SI A1E
 MU A01
 SI 9
 MZ A11
 SI A1A
 MU A11
 SI A16
 MU A11
 SI 2
 MZ A03
 SI A00
 SI 0
 SI 6
 SI A18
 MU A03
 SI/ 9
 MZ A13
 SI A10
 SI 9
 MU A13
 SI 4
 MZ A05
 SI A02
 SI 2
 MU A05
 SI 3
 SI 1
 SI 5
 MZ A15
 SI A12
 SI/ 9
 MU A15
 SI 1
 SI 3
 MZ A07

0C6D : 8304
 0C6E : 8004
 0C6F : 4307
 0C70 : 5317
 0C71 : 8314
 0C72 : 8003
 0C73 : 8001
 0C74 : 8005
 0C75 : 4317
 0C76 : 8005
 0C77 : 5309
 0C78 : 8306
 0C79 : 8001
 0C7A : 8003
 0C7B : 4309
 0C7C : 8007
 0C7D : 331F
 0C7E : 1101
 0C7F : A10D
 0C80 : 82F1
 0C81 : 3320
 0C82 : 1101
 0C83 : A10D
 0C84 : 82F0
 0C85 : 331C
 0C86 : 831C
 0C87 : 8320
 0C88 : 231E
 0C89 : 831E
 0C8A : 530B
 0C8B : 8308
 0C8C : 8005
 0C8D : 430B
 0C8E : 8008
 0C8F : 3321
 0C90 : 530D
 0C91 : 1101
 0C92 : A10D
 0C93 : 82F0
 0C94 : 331C
 0C95 : 830A
 0C96 : 831C
 0C97 : 430D
 0C98 : 530F
 0C99 : 831A
 0C9A : 430F
 0C9B : 8000
 0C9C : 7318
 0C9D : 3322
 0C9E : 7323
 0C9F : 3324
 0CA0 : 8323
 0CA1 : 3325
 0CA2 : 7326
 0CA3 : F308
 0CA4 : 5309
 0CA5 : 5307
 0CA6 : 5305
 0CA7 : 5303
 0CA8 : 5301
 0CA9 : 530D
 0CAA : 530B
 0CAB : 7327
 0CAC : F308
 0CAD : 5309
 0CAE : 5307
 0CAF : 5305

SI A04
 SI 4
 MU A07
 MZ A17
 SI A14
 SI 3
 SI 1
 SI 5
 MU A17
 SI 5
 MZ A09
 SI A06
 SI 1
 SI 3
 MU A09
 SI 7
 ET A1F
 APL 801
 CPL #001E (080D)
 SI 9F1
 ET A20
 APL 801
 CPL #001E (080D)
 SI 9F0
 ET A1C
 SI A1C
 SI A20
 OU A1E
 SI A1E
 MZ A0B
 SI A08
 SI 5
 MU A0B
 SI 8
 ET A21
 MZ A0D
 APL 801
 CPL #001E (080D)
 SI 9F0
 ET A1C
 SI A0A
 SI A1C
 MU A0D
 MZ A0F
 SI A1A
 MU A0F
 SI 0
 SI/ A18
 ET A22
 SI/ A23
 ET A24
 SI A23
 ET A25
 SI/ A26
 SAUT 0CAB (A08)
 MZ A09
 MZ A07
 MZ A05
 MZ A03
 MZ A01
 MZ A0D
 MZ A0B
 SI/ A27
 SAUT 0CB4 (A08)
 MZ A09
 MZ A07
 MZ A05

0CB0 : 5303
 0CB1 : 4301
 0CB2 : 530D
 0CB3 : 530B

MZ A03
 MU A01
 MZ A0D
 MZ A0B

; MODULE : (ACTION)

0CB4 : 8322
 0CB5 : 8301
 0CB6 : 3328
 0CB7 : 8324
 0CB8 : 8313
 0CB9 : 3329
 0CBA : 8325
 0CBB : 8313
 0CBC : 332A
 0CBD : 8301
 0CBE : 3020
 0CBF : 8328
 0CC0 : 3026
 0CC1 : 8303
 0CC2 : 3022
 0CC3 : 8313
 0CC4 : 3326
 0CC5 : 8329
 0CC6 : 332B
 0CC7 : 832A
 0CC8 : 332C
 0CC9 : 732C
 0CCA : F303
 0CCB : 7027
 0CCC : 3027
 0CCD : 8305
 0CCE : 3024
 0CCF : 8315
 0CD0 : 8309
 0CD1 : 231E
 0CD2 : 831E
 0CD3 : 3025
 0CD4 : 8315
 0CD5 : 8307
 0CD6 : 231E
 0CD7 : 831E
 0CD8 : 3023
 0CD9 : 8315
 0CDA : 8307
 0CDB : 231E
 0CDC : 831E
 0CDD : 3021
 0CDE : 8315
 0CDF : 8321
 0CE0 : 830D
 0CE1 : 231E
 0CE2 : 831E
 0CE3 : 5319
 0CE4 : 8317
 0CE5 : 3327
 0CE6 : 8317
 0CE7 : 5027
 0CE8 : 831F
 0CE9 : 4319
 0CEA : 830B
 0CEB : C101
 0CEC : 830D
 0CED : 830F
 0CEE : 231E
 0CEF : 831E

SI A22
 SI A01
 ET A28
 SI A24
 SI A13
 ET A29
 SI A25
 SI A13
 ET A2A
 SI A01
 ET 20
 SI A28
 ET 26
 SI A03
 ET 22
 SI A13
 ET A26
 SI A29
 ET A2B
 SI A2A
 ET A2C
 SI/ A2C
 SAUT 0CCD (A03)
 SI/ 27
 ET 27
 SI A05
 ET 24
 SI A15
 SI A09
 OU A1E
 SI A1E
 ET 25
 SI A15
 SI A07
 OU A1E
 SI A1E
 ET 23
 SI A15
 SI A07
 OU A1E
 SI A1E
 ET 21
 SI A15
 SI A21
 SI A0D
 OU A1E
 SI A1E
 MZ A19
 SI A17
 ET A27
 SI A17
 MZ 27
 SI A1F
 MU A19
 SI A0B
 IBE 801
 SI A0D
 SI A0F
 OU A1E
 SI A1E

0CF0 : 332C
 0CF1 : 732C
 0CF2 : F303
 0CF3 : 110C
 0CF4 : E101
 0CF5 : 8301
 0CF6 : 3300
 0CF7 : 8303
 0CF8 : 3302
 0CF9 : 8305
 0CFA : 3304
 0CFB : 8307
 0CFC : 3306
 0CFD : 8309
 0CFE : 3308
 0CFF : 830B
 0D00 : 330A
 0D01 : 830D
 0D02 : 330C
 0D03 : 830F
 0D04 : 330E
 0D05 : 8311
 0D06 : 3310
 0D07 : 8313
 0D08 : 3312
 0D09 : 8315
 0D0A : 3314
 0D0B : 8317
 0D0C : 3316
 0D0D : 8319
 0D0E : 3318
 0D0F : 832B
 0D10 : 9323
 0D11 : 02FE
 0D12 : 0100
 0D13 : 0102
 0D14 : F437

Constantes

080C : 0000
 080D : 001E

ET A2C
 SI/ A2C
 SAUT 0CF5 (A03)
 APL #0000 (080C)
 RGR 801
 SI A01
 ET A00
 SI A03
 ET A02
 SI A05
 ET A04
 SI A07
 ET A06
 SI A09
 ET A08
 SI A0B
 ET A0A
 SI A0D
 ET A0C
 SI A0F
 ET A0E
 SI A11
 ET A10
 SI A13
 ET A12
 SI A15
 ET A14
 SI A17
 ET A16
 SI A19
 ET A18
 SI A2B
 TP A23
 ADR 9FE
 ADR 800
 ADR 802
 SAUT 0C37