Formation

Logiciel

AUTOMGEN 7

Support de cours

Parc d'Activités des Peupliers - Bâtiment L - 25, rue des Peupliers - 92000 NANTERRE
Tél : 01 47 82 80 94 - Fax : 01 47 82 57 61
www.pos-industry.com

24/09/2002

SOMMAIRE

1 INSTALLATION DU LOGICIEL AUTOMGEN 7

1.1 CONFIGURATION MATERIELLE ET LOGICIELLE REQUISE

- un ordinateur avec UC de type Pentium ou équivalent,
- une mémoire RAM de 32 Mo (minimum),
- carte graphique 8Mo
- un espace disponible sur le disque dur de 30 Mo,
- un lecteur de CD ROM,
- un lecteur de disquettes 3"$^{1/2}$,
- un écran couleur SVGA (17" recommandé),
- une souris Microsoft ou compatible,
- MS Windows 95, 98, 2000, ME ou XP.

1.2 PROCEDURE D'INSTALLATION

- Le CD-ROM contient AUTOMGEN 7, ACROBAT READER (pour l'accès à la documentation en ligne), CROSSROADS (un utilitaire de conversion 3D) et DIRECTX 8 (pour la gestion de l'affichage 3D).
- Placer le CD ROM d'AUTOMGEN dans le lecteur approprié,
- L'installation démarre automatiquement (autorun), sinon lancez l'exécutable « setup.exe » qui se trouve à la racine sur le CD-ROM.
- Suivez les instructions qui apparaissent à l'écran.

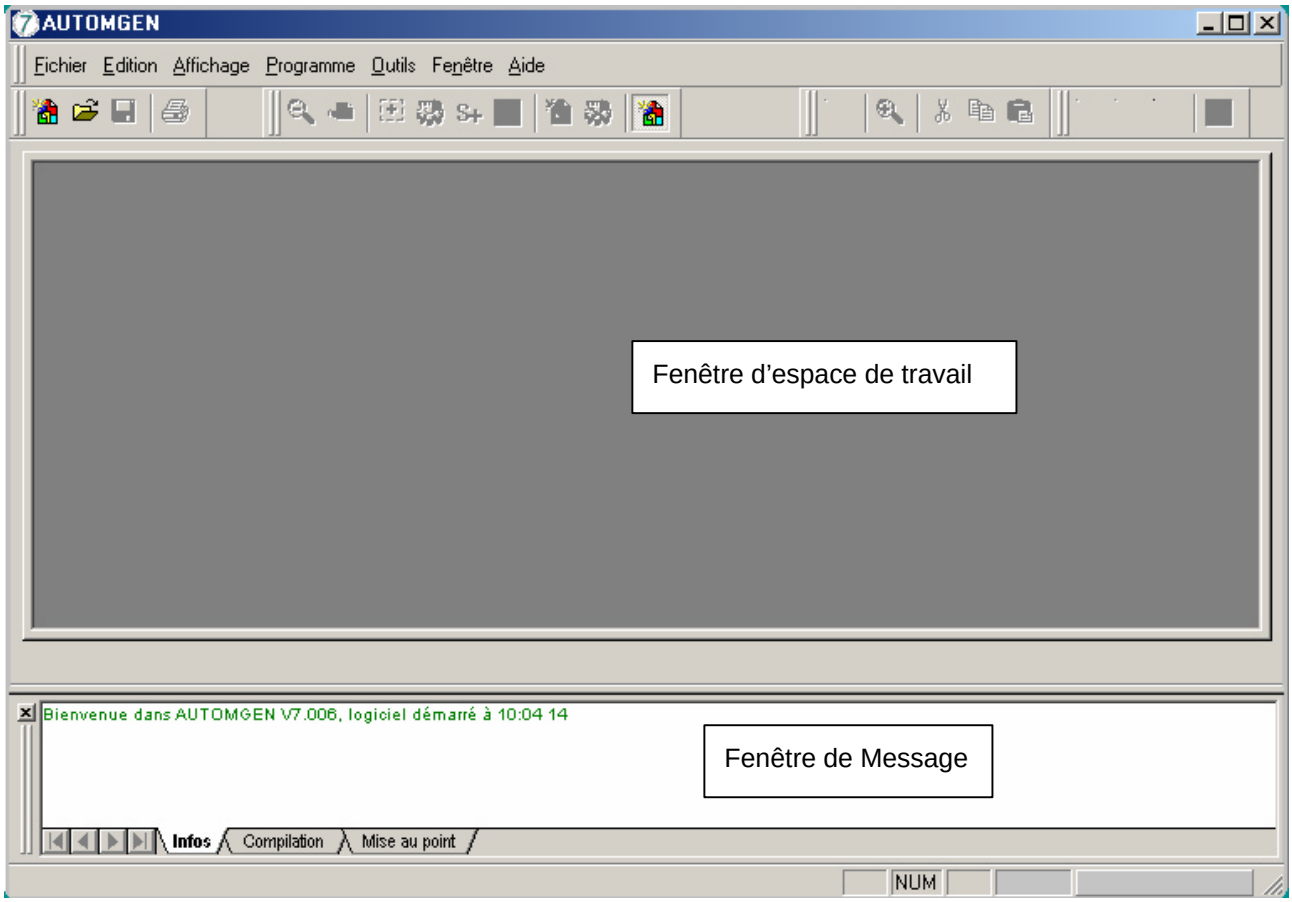
A la fin de l'installation, un groupe de programmes est créé, contenant l'icône de démarrage du logiciel AUTOMGEN, une icône ouvrant un fichier de texte contenant les notes d'information du logiciel et une icône ouvrant le gestionnaire de licence.

2 LANCEMENT AUTOMGEN

Pour démarrer AUTOMGEN, cliquez deux fois sur l'icône 

du groupe de programme du même nom.

La fenêtre principale apparaît :



Cet écran contient 2 fenêtres :

- la fenêtre **"Espace de travail"**,

- la fenêtre **"Messages"**,

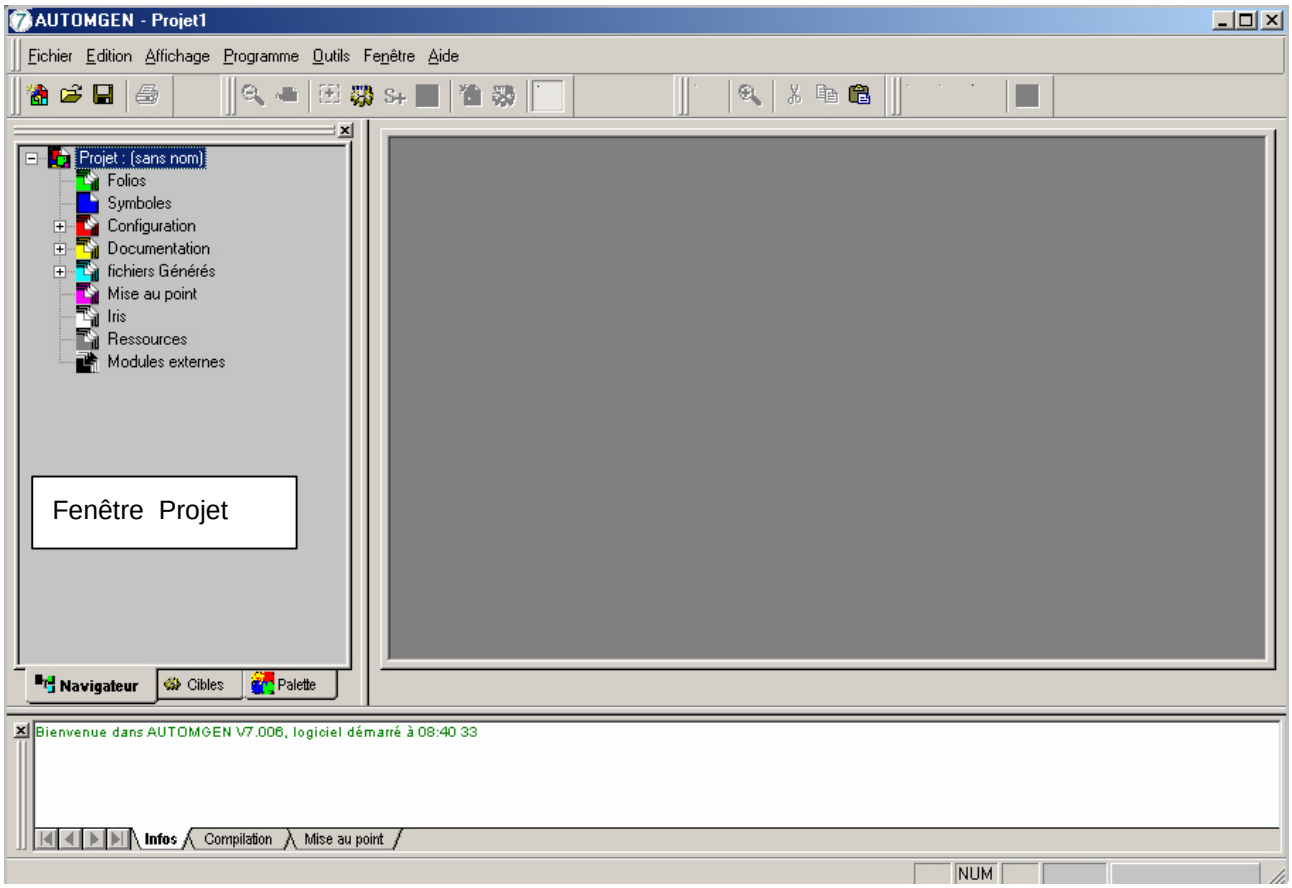
Pour accéder à une de ces 2 fenêtres, cliquez directement dans la fenêtre désirée.

3 OUVRIR UN NOUVEAU PROJET

Pour ouvrir un nouveau projet :

- Sélectionnez la commande "**Nouveau**" du menu "**Fichier**"
- Ou appuyez sur l'icône 

La fenêtre principale apparaît avec une troisième fenêtre « **Projet** » :

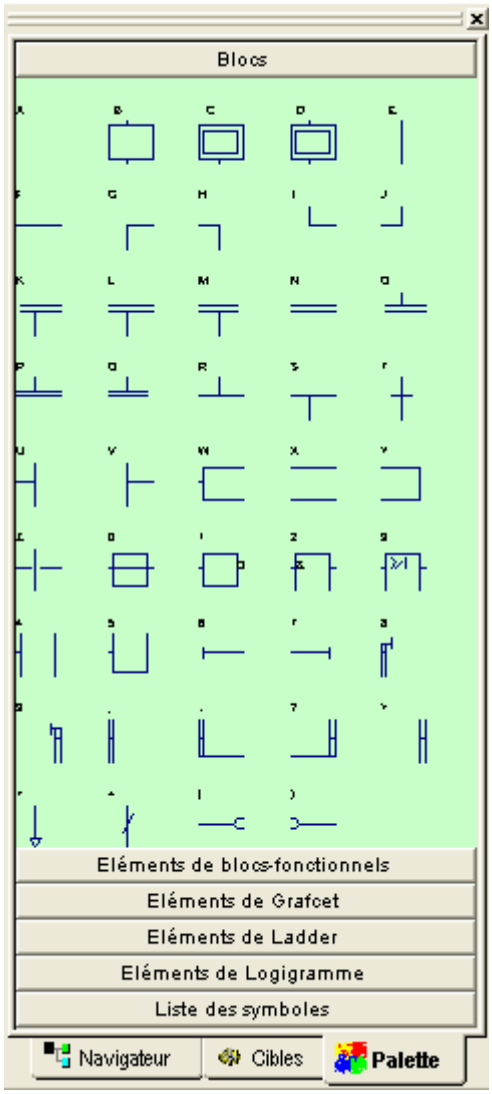


La fenêtre projet contient :

- L'onglet **Navigateur**, qui permet de créer les différents éléments d'une application et d'y accéder rapidement :
  - ✓ Folios,
  - ✓ Table de symbole,
  - ✓ Configuration du Post-processeur (adressage, ..),
  - ✓ Configuration matérielle,
  - ✓ Monitoring (Mise au point),
  - ✓ Modules IRIS 2D et 3D,
  - ✓ Ressources (objets IRIS 2D, objets 3D, ….),
  - ✓ Modules externes.

- L'onglet **Cibles** permet de sélectionner le type de traducteur (exécuteur PC ou automate programmable) avec laquelle on va effectuer la compilation du projet :

- L'onglet **Palette** permet d'accéder à des éléments de logigrammes, de blocs fonctionnels, de grafcet, de ladder, de bloc et la liste des symboles à utiliser lors de la création des programmes dans les folios :



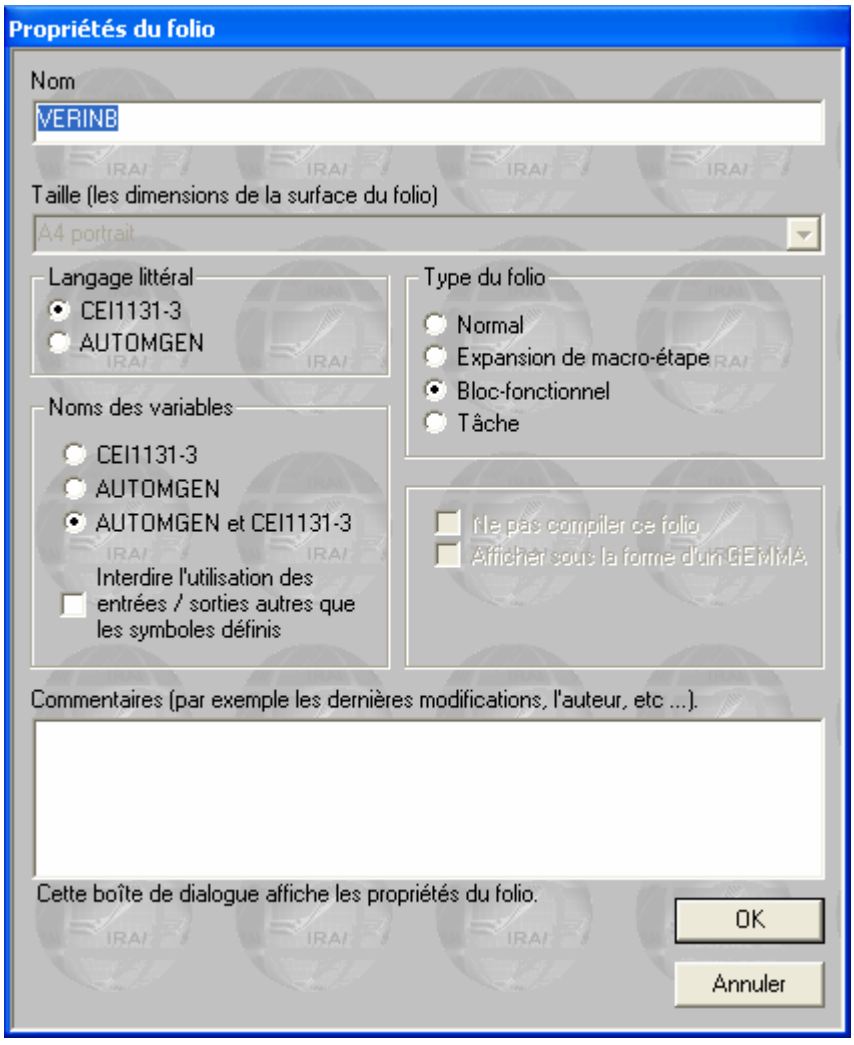
4 LE NAVIGATEUR

4.1 FOLIOS

Un folio est une page sur laquelle est dessiné un programme ou une partie de programme.

Il faut cliquer avec le bouton droit de la souris pour créer un nouveau folio ou importer un ou plusieurs folios existants. Il est possible d'importer des folios au format AUTOMGEN (*.GR7) ou des GRAFCET issus du logiciel CADEPA (*.GIG, format d'export de l'éditeur Grafcet CADEPA Windows).

Lorsque le folio est ouvert, il est alors possible de définir ses propriétés : pour cela, sélectionner le folio puis cliquer sur le bouton droit de la souris et choisir dans le menu la commande <**propriétés**> pour faire apparaître la fenêtre suivante :



Les différents types de folios proposés sont les suivants :

- folio normal : pour la création de grafcet, de ladder, etc…
- folio contenant une expansion de macro-étape
- folio type bloc-fonctionnel
- folio contenant une tâche (par exemple tâche évènementiel pour les automates télémécanique)

Cette fenêtre permet également de choisir le type de syntaxe AUTOMGEN ou CEI 1131-3 pour le langage littéral et les noms de variables de l'application.

4.2 SYMBOLES

La liste des symboles donne la correspondance entre des noms « symboliques » et des noms de variables. Un projet ne peut contenir qu'une seule table de symboles.

Il faut cliquer avec le bouton droit de la souris pour avoir la possibilité de créer une table de symbole.

4.3 CONFIGURATION

4.3.1 POST-PROCESSEURS

Sous cette rubrique se trouve tous les éléments de configuration des post-processus, c'est à dire les différents types d'automates.

4.3.2 OPTIONS DU COMPILATEUR

Il permet de modifier le réglage des options du compilateur.

4.4 DOCUMENTATION

Il permet d'accéder à la fonction d'impression du dossier.

4.5 FICHIERS GENERES

4.5.1 CODE PIVOT

C'est un langage littéral de bas niveau résultant de la compilation du projet.

4.5.2 REFERENCES CROISEES

Il affiche la liste des variables utilisées dans l'application avec leurs éventuelles variables automates associées ainsi que le nom du ou des folios où elles sont utilisées.

4.5.3 POST-PROCESSEURS

Il contient la compilation du projet dans le langage correspondant à l'automate sélectionné.

4.6 MISE AU POINT

Il regroupe des outils permettant la visualisation et la modification en dynamique de l'état des variables.

4.7 IRIS

IRIS 2D permet de créer des pupitres, des applications de supervision et des applications de simulation de parties opératives 2D.

IRIS 3D permet de créer des applications de simulations de parties opératives 3D.

4.8 RESSOURSES

Il permet d'ajouter tout type de fichier au projet.

Les fichiers ainsi ajoutés feront partie intégrante du projet et seront sauvegardés avec les autres éléments.

4.9 MODULES EXTERNES

Ils sont réservés à des modules exécutables développés par des tiers et interfacés avec automgen.

5 LES VARIABLES

5.1 VARIABLES BOOLEENNES

Type	Syntaxe AUTOMGEN	Syntaxe CEI 1131-3	Commentaire
Entrées	I0 à I9999	%I0 à %I9999	Peut correspondre ou non à des entrées physiques
Sorties	O0 à O9999	%Q0 à %Q9999	Peut correspondre ou non à des sorties physiques
Bits Systèmes	U0 à U99	%M0 à%M99	Réserver à l'exécuteur PC
Bits Utilisateurs	U100 à U9999	%M100 à %M9999	Bits internes à usage général
Etapes grafcet	X0 à X9999	%X0 à %X9999	Bits d'étapes grafcet
Bits de mots	M0#0 à M9999#15	%MW0:X0 à %MW9999 :X15	0 est le poids faible 15 est le poids fort

5.2 VARIABLES NUMERIQUES

Type	Syntaxe AUTOMGEN	Syntaxe CEI 1131-3	Commentaire
Compteurs	C0 à C9999	%C0 à %C9999	Compteur de 16 bits
Mots systèmes	M0 à M199	%MW0 à %MW199	Réserver à l'exécuteur PC
Mots Utilisateurs	M200 à M 9999	%MW200 à%MW9999	Mots de 16 Bits
Longs	L100 à L4998	%MD100 à %MD4998	Valeurs entières sur 32 bits
Flottants	F100 à F4998	%MF100 à %MF4998	Valeurs réelles sur 32 bits

5.3 COMPTEUR, MOT OU LONG

Les actions pouvant être effectuer sur les compteurs, mots ou long sont :

- Mise à zéro : R
- Incrémentation de un : +
- Décrémentation de un : -

5.4 TEMPORISATIONS

Type	Syntaxe AUTOMGEN	Syntaxe CEI 1131-3	Commentaire
Tempo	T0 à T9999	%T0 à %T9999	

La durée s'exprime par défaut en dixième de seconde.

Elle peut être exprimée en jours, heures, minutes, secondes et millisecondes avec les opérateurs « d », « h », « m », « s » et « ms ».

Les actions sur les temporisateurs sont les suivantes :

- Mise à zéro ou initialisation : R
- Lancement tempo : S ou action dans une étape

6 LES ACTIONS

Les actions sont utilisées dans les rectangles d'action du langage grafcet, du langage logigramme et les bobines du langage ladder.

Les syntaxes sont :

- Affectation complémentée d'une variable booléenne : « N variable booléenne »
- Mise à 1 d'une variable booléenne : « S variable booléenne »
- Mise à 0 d'une variable booléenne : « R variable booléenne »
- Inversion d'une variable booléenne : « I variable booléenne »
- Temporisation limité dans le temps : « LTn/durée »
- Temporisée : « DTn/durée »
- Impulsion sur front montant : « P1 »
- Impulsion sur front descendant : « P0 »
- Mémorisé et temporisé : « SDTn/durée »
- Temporisé et mémorisé : « DSTn/durée »
- Mémorisé et limité dans le temps : « SLTn/durée »
- Forçage d'un grafcet selon une liste d'étapes actives : « F<grafcet> :{<liste d'étape active>} » ou « F/<nom de folio> :{<liste d'étapes actives>} »
- Mémorisation de l'état d'un grafcet : « G<grafcet> :n° de bit » ou « G/<nom de folio> :<n° de bit> »
- Forçage d'un grafcet à partir d'un état mémorisé : « F<grafcet> :<n° de bit> » ou « F/<nom de folio> :<n° de bit> »
- Figeage d'un grafcet : « F<grafcet> » ou « F/<nom de folio> »

7 LES TESTS

Un test est une équation booléenne composée de une ou de n variables séparées par des opérateurs « + »(ou) ou « . »(et).

Les différentes tests sont :

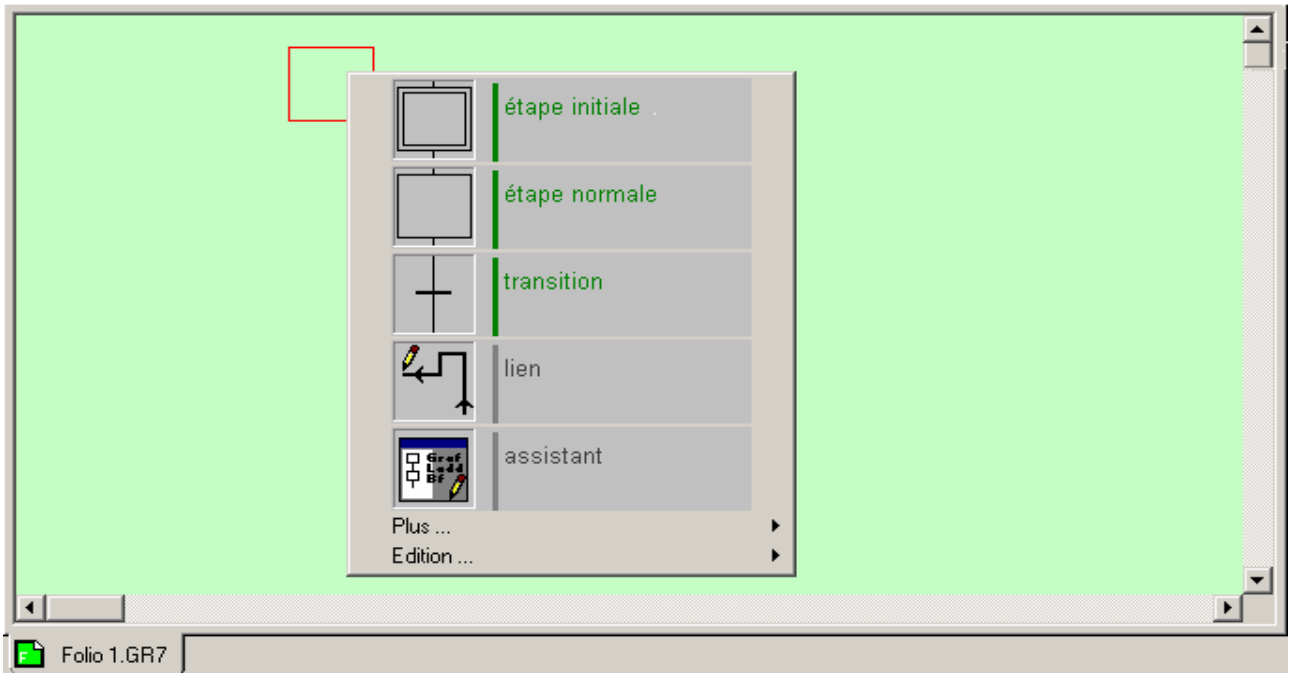
- « / » : test l'état complémenté
- « u » ou « ↑ » : test le front montant
- « d » ou « ↓ » : test le front descendant
- « » (néant) ou « =1 » : test toujours vrai
- « = » : égal
- « ! » ou « <> » : différent
- « < » : inférieur (non signé)
- « > » : supérieur (non signé)
- « << » : inférieur (signé)
- « >> » : supérieur (signé)
- « <= » : inférieur ou égal (non signé)
- « >= » : supérieur ou égal (non signé)
- « <<= » : inférieur ou égal (signé)
- « >>= » : supérieur ou égal (signé)

8 FOLIOS

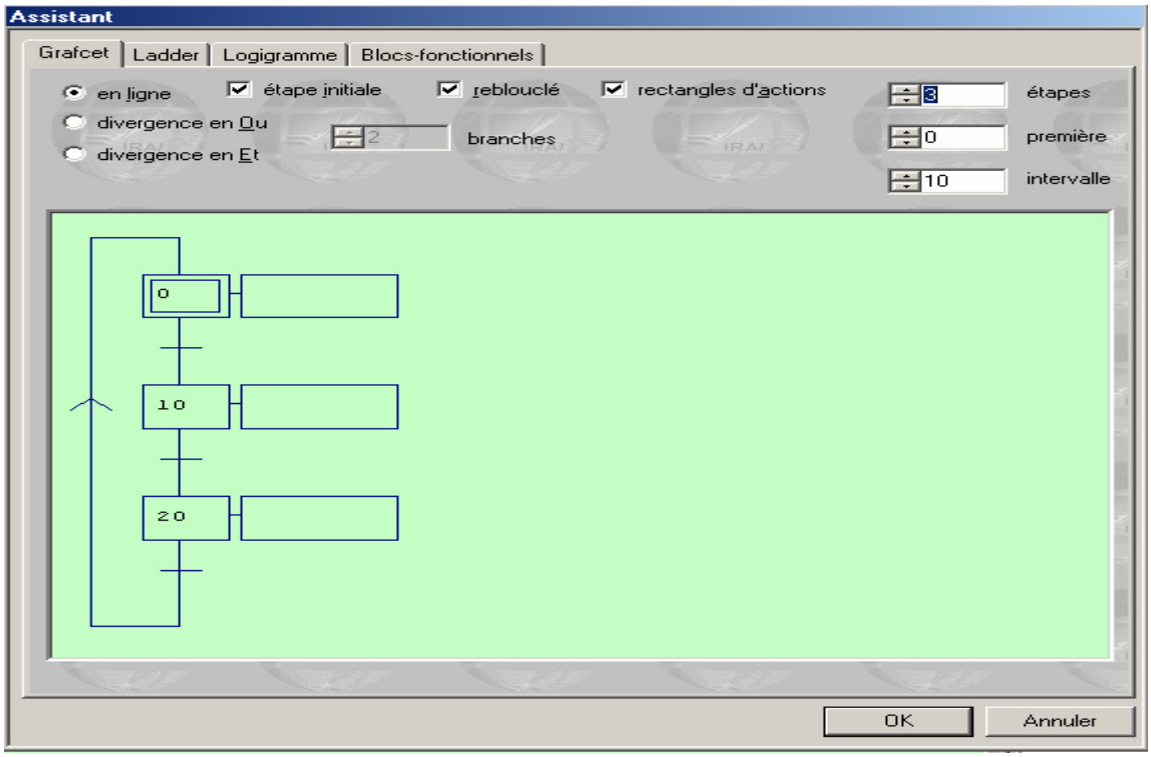
Une fois, qu'on a crée un nouveau folio, on peut le programmer.

8.1 CREATION DE GRAFCET

En cliquant sur le bouton droit de la souris dans la fenêtre du folio, on obtient une fenêtre de sélection d'objets graphiques :

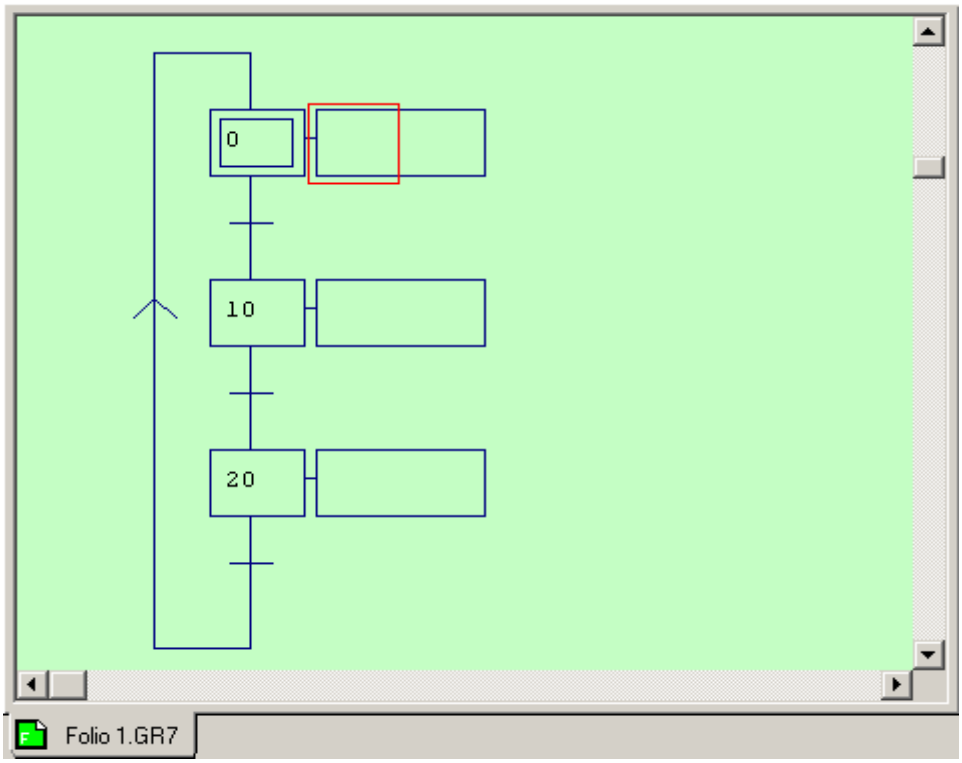


Si on sélectionne l'assistant, on obtient une aide à la création de grafcet, de ladder, de logigramme ou de blocs-fonctionnels :

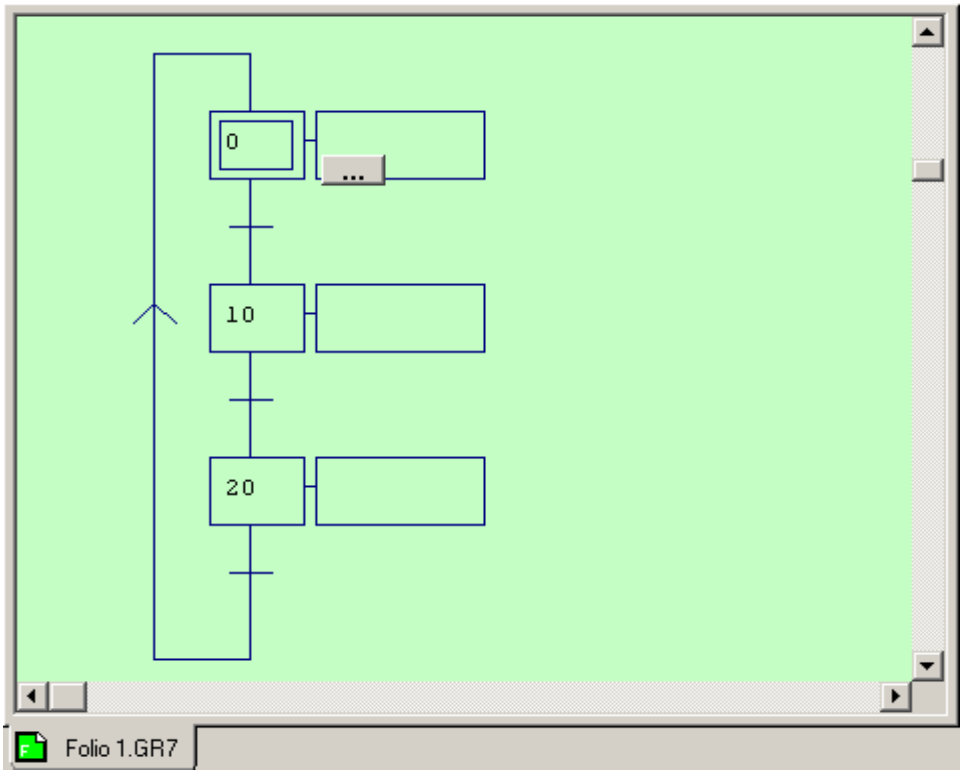


8.1.1 ETAPES

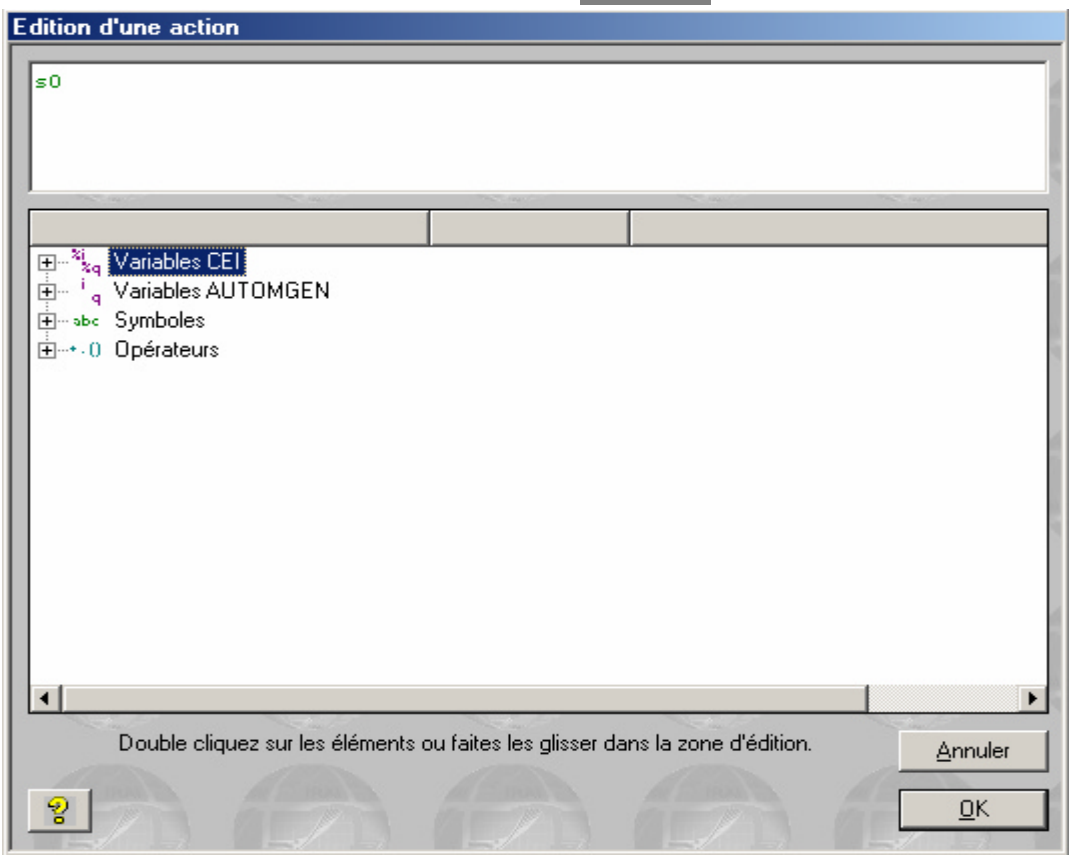
Pour insérer les actions à effectuer sur un étape, on vient cliquer sur le rectangle lié à l'étape.



Une fois qu'on a cliqué, on obtient :

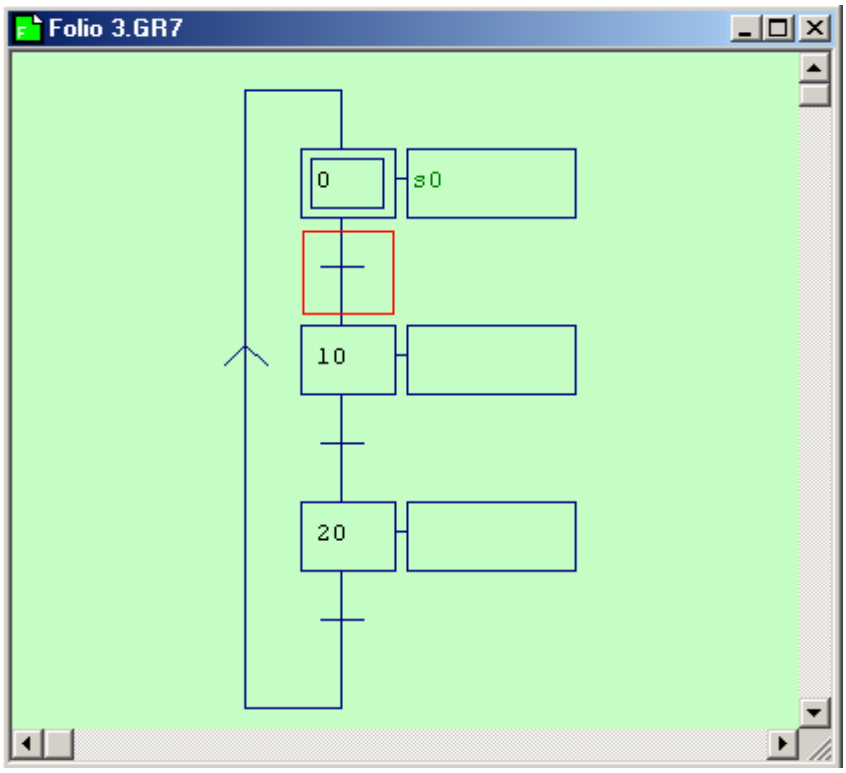


On peut écrire directement l'action et la variable ou en cliquant sur ..., on obtient une fenêtre contenant toutes les actions et les variables qu'on peut effectuer sur cette étape.

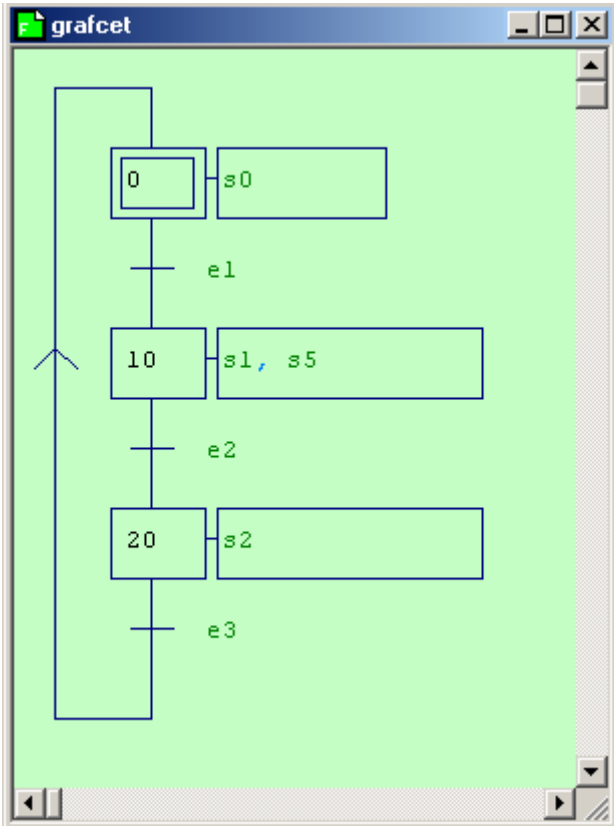


8.1.2 TRANSITIONS

On clique sur la transition qu'on veut renseigner :



Ensuite, on suit la même démarche que pour une étape.



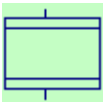
Remarque :

- L'onglet Palette permet de modifier le grafcet, en rajoutant d'autre étape ou des divergences et des convergences.
- Sur les étapes, on peut effectuer plusieurs actions en les séparant par « , ».
- Pour ajouter des commentaires, il ne faut pas se positionner sur le grafcet.

8.1.3 MACRO-ETAPES

Une macro-étape (ME) est l'unique représentation d'un ensemble unique d'étapes et de transitions nommé « expansion de ME ».

Règles d'une ME :

- Une macro-étape est un grafcet se trouvant dans un folio distinct.
- Il doit comporter une étape d'entrée et une étape de sortie.
- L'étape d'entrée devra porter le numéro 0 ou le repère Exxx, avec xxx = numéro quelconque.
- L'étape de sortie devra porter le numéro 9999 ou le repère Sxxx, avec xxx=numéro quelconque.
- Symbole : 

8.2 CREATION DE GEMMA

AUTOMGEN implémente la description de grafcet de gestion des modes de marche sous forme de Gemma.

Il est possible de passer d'un mode d'édition Gemma au mode d'édition grafcet et inversement si la structure du grafcet n'est pas modifié.

Marche à suivre :

- Ajouter un nouveau folio avec le choix « gemma » dans la liste des tailles.
- Cliquez sur la propriété du folio et sélectionner « afficher sous la forme d'un gemma ». si on désélectionne cette case, on obtient un affichage en grafcet des états validés en gemma.

La fenêtre contient alors un gemma dont tous les rectangles et tous les liens sont grisés.

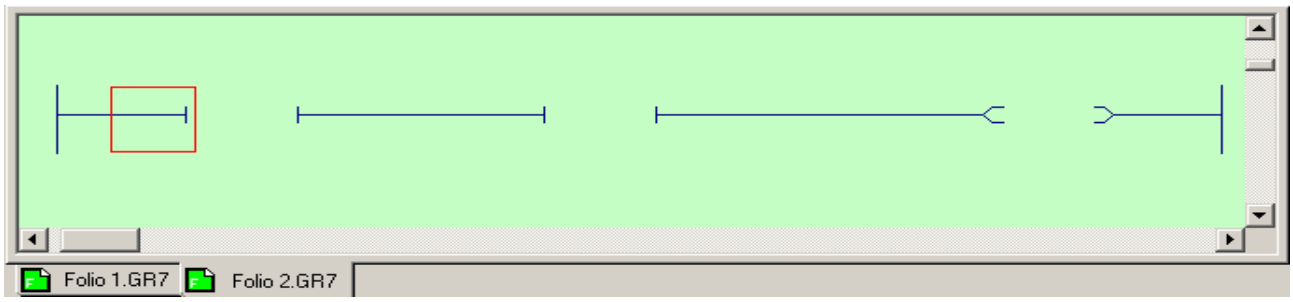
Pour valider un rectangle ou une liaison, il faut cliquer dessus avec le bouton droit de la souris.

Pour modifier le contenu d'un rectangle ou d'une liaison, il faut cliquer dessus avec le bouton gauche de la souris.

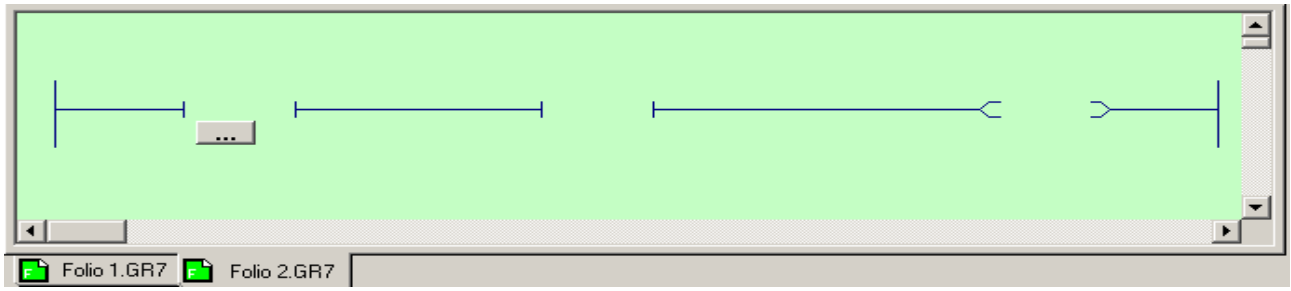
8.3 CREATION DE LADDER

De la même façon que le grafcet, on sélectionne le ladder dans l'assistant.

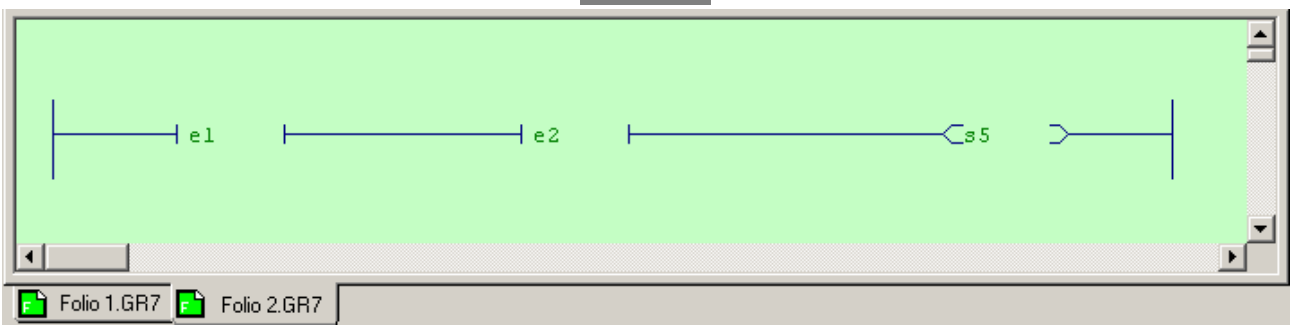
Pour renseigner le contact, on se positionne comme suit et on clique sur le bouton gauche de la souris :



Une fois qu'on a cliqué, on obtient :

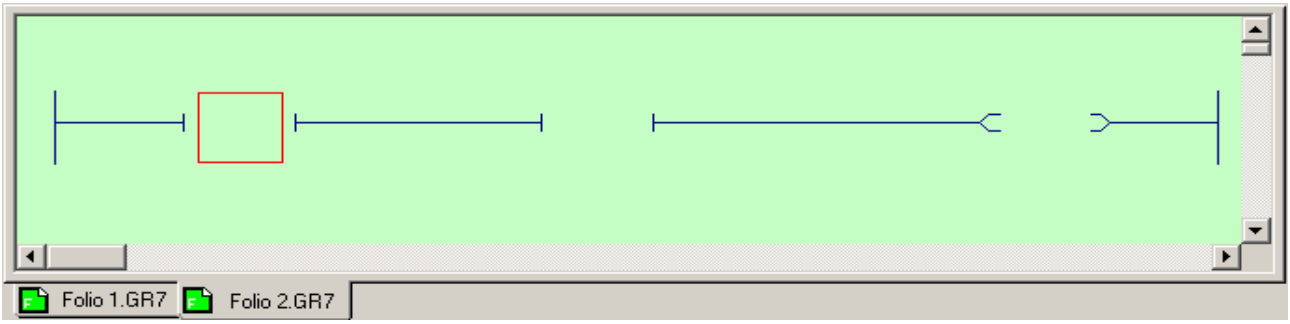


En cliquant sur les ..., on obtient la même fenêtre que pour le grafcet contenant tous les tests et les variables qu'on peut effectuer sur cette contact.



Remarque :

- L'onglet Palette permet de modifier le ladder, en rajoutant d'autre contact ou bobines.
- Pour ajouter des commentaires, il ne faut pas se positionner sur le ladder. Par contre pour mettre le commentaire sur le contact, il faut se mettre comme suit :

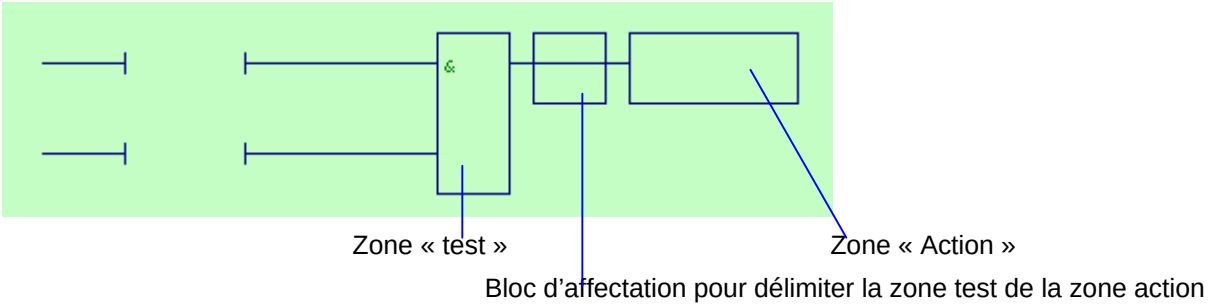


8.4 CREATION DES LOGIGRAMMES

De la même façon que le grafcet, on sélectionne le logigramme dans l'assistant.

Le langage logigramme permet d'écrire graphiquement des équations booléennes.

Le logigramme se présente de la forme :



Le paramétrage se fait de la même façon que le ladder et le grafcet.

8.5 CREATION DE BLOCS FONCTIONNELS

Les blocs fonctionnels permettent d'associer à un élément graphique un ensemble d'instructions écrites en langage littéral.

Ils sont surtout utilisées pour éviter d'écrire le même bout de programme qui se répète plusieurs fois dans le programme par exemple pour des vannes, des moteurs, ….

Les blocs fonctionnels s'utilisent à l'intérieur de schémas de types logigramme ou ladder, ils possèdent de une à n entrées booléenne et de une à n sorties booléennes.

8.5.1 BLOCS FONCTIONNELS SIMPLES

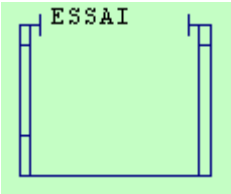
Un bloc fonctionnel est composé de deux fichiers distincts. Un fichier portant l'extension « .ZON » qui contient le dessin du bloc fonctionnel et un fichier portant l'extension « .LIB » qui contient une suite d'instructions écrites en langage littéral définissant le fonctionnement du bloc fonctionnel.

Les fichiers « .ZON » et « .LIB » doivent être écrite en majuscule et porter le nom du bloc fonctionnel.

8.5.1.1 CREATION FICHIER « .ZON »

La création du fichier « .ZON » se fait dans les folios avec l'assistant.

On crée un bloc fonctionnel à partir de l'assistant avec le nombre d'entrée et de sortie et son nom en majuscule.



Ensuite on sélectionne le bloc pour le sauvegardé par l'intermédiaire de la commande « copier vers.. » du menu « édition » avec le nom en majuscule.

Une fois qu'on a crée l'enveloppe, on va crée les instruction du bloc.

8.5.1.2 CREATION FICHIER « .LIB »

La création du fichier « .LIB » se fait par un éditeur de texte de Windows (Notepad,…) en dehors d'AUTOMGEN. C'est un fichier texte contenant les instructions en langage littéral.

Syntaxe des variables du bloc fonctionnel:

- Entrée : « {Ix} » ou x est le numéro de l'entrée booléenne exprimé en Hexadécimal.
- Sortie : « {Ox} » ou x est le numéro de la sortie booléenne exprimé en Hexadécimal.
- Référence à un paramètre du bloc : {?x} ou x est le numéro du paramètre en Hexadécimal.

Exemple :

{O0}=(1) ({I1})

{O0}=(0) ({I0})

Une fois qu'on a crée le fichier « .LIB », on doit faire l'appel de ce fichier par AUTOMGEN.

Dans AUTOMGEN, on insère le fichier « .LIB » par l'intermédiaire de la « Ressource ».

Remarque :

- Si on modifie le fichier « .LIB », on doit l'effacer de la ressource et la rajouter dans la ressource pour que AUTOMGEN prenne en compte les modifications.

8.5.2 BLOCS FONCTIONNELS EVOLUES

Les blocs fonctionnels évolués sont des folios qu'on utilise comme des blocs fonctionnels.

L'appel de ces blocs fonctionnel se fait avec l'extension « .GR7 ».

Pour référencer une variable externe d'un bloc fonctionnel, il faut utiliser un mnémonique incluant le texte suivant :

- « {In} » pour référencer l'entrée booléenne n.
- « {On} » pour référencer la sortie booléenne n.
- « {?x} » pour référencer le paramètre n.

8.6 LANGAGES LITTERAUX

Il y a 3 formes de langages littéraux sur AUTOMGEN :

- le langage littéral bas niveau
- le langage littéral étendu
- le langage littéral St de la norme CEI 1131-3

8.6.1 LE LANGAGE LITTERAL BAS NIVEAU

Le langage littéral de bas niveau est un langage de type assembleur, aussi connu sous le nom code pivot.

Il utilise une notion d'accumulateur pour les traitements numériques.

8.6.1.1 LES DRAPEAUX

Les drapeaux sont des variables booléennes positionnées en fonction du résultat des opérations numériques.

Indicateur de :

- Retenue C : - 1 une opération a engendré une retenue
  - 0 une opération n'a pas engendré une retenue
- Zéro Z : - 1 une opération a engendré un résultat nul
  - 0 une opération a engendré un résultat non nul
- Signe S : - 1 une opération a engendré un résultat négatif
  - 0 une opération a engendré un résultat positif
- Débordement O : - 1 une opération a engendré un dépassement de capacité
  - 0 une opération n'a pas engendré un dépassement de capacité

8.6.1.2 LES ACCUMULATEURS

Les accumulateurs sont des registres internes au système qui exécute le programme final et permettent d'accueillir temporairement des valeurs.

Il existe 3 accumulateurs :

- AAA : accumulateur 16 bits
- AAL : accumulateur entiers 32 bits
- AAF : accumulateur flottant

Pour charger l'accumulateur, on utilise l'instruction : LDA

Pour transférer le contenu de l'accumulateur dans une variable, on utilise l'instruction : STA

8.6.1.3 MODES D'ADRESSAGE

Les différents mode d'adressage sont :

TYPE	SYNTAXE	EXEMPLE
Immédiat 16 bits	{constante}	100
Immédiat 32 bits	{constante}L	100000L
Immédiat flottants	{constante}R	3.14R
Absolu	{variable}{repère de variable}	O540
Accumulateur 16 bits	AAA	AAA
Accumulateur 32 bits	AAL	AAL
Accumulateur flottants	AAF	AAF
Indirect	{variable}{(repère de mot)}	O(220)
Label	: {nom de label} :	:boucle:

8.6.1.4 LES TESTS

Si un test est composé d'une seule variable, un opérateur AND doit quand même lui être associé.

Opérateurs :

- AND : et
- ORR : ou
- EOR : fin de ou
- / : pas
- # : front montant
- * : front descendant
- @ : état immédiat
- = : égal
- ! : différent
- < : inférieur non signé
- > : supérieur non signé
- << : inférieur signé
- >> : supérieur signé

Les commentaires doivent débuter par le caractère « ; » (point virgule), tous les caractères venant à la suite sont ignorés.

8.6.1.5 NUMEROTATIONS

Les variables peuvent être écrites sous formes :

- Décimal
- Hexadécimal : préfixe « $ » ou « 16# »
- Binaire : préfixe « % » ou « 2# »
- Ascii : le caractère « " » suivi de 1 ou 2 caractères suivis de « " »

La syntaxe des réels est la suivante : [-]i[[.d] Esx]

- i est la partie entière
- d est la partie décimale
- s est le signe de l'exposant
- x est l'exposant

8.6.1.6 PREDISPOSITIONS

Une prédisposition permet de fixer la valeur d'une variable au démarrage de l'application.

La syntaxe est la suivante : « $(variable)=constante{,constante{,constante…}} »

Exemple :

$t1=100 fixe la consigne de la temporisation 25 à 100

$mw100=10,20,30 place les valeurs 10, 20, 30 dans les mots 100, 101, 102.

8.6.1.7 ADRESSAGE INDIRECT

L 'adressage indirect permet d'effectuer une opération sur une variable pointée par un index.

Ce sont les variables M(les mots) qui servent d'index.

La syntaxe est la suivante : « variable (index) »

8.6.1.8 ADRESSAGE D'UNE VARIABLE

Le caractère « ? » permet de spécifier l'adresse d'une variable.

Exemple : LDA ?O10 ; place la valeur 10 dans l'acumulateur.

8.6.1.9 SAUT ET LABEL

Les sauts doivent faire référence à un label.

La syntaxe d'un label est : « :nom du label: »

Exemple :

Jmp :suite :

…

:suite :

...

8.6.1.10 MACRO-INSTRUCTION

Les macro-instructions sont des nouvelles instructions du langage littéral derrière lesquelles se cache un ensemble d'instructions de base.

La syntaxe d'appel d'une macro-instruction est : « %<nom de la macro-instruction>{paramètres…} »

La syntaxe de déclaration d'une macro-instruction est :

#MACRO

<programme>

#ENDM

cette déclaration se trouve dans un fichier portant le nom de la macro-instruction et possédant l'extension « .M ».

Dix paramètres peuvent être passés à la macro-instruction. A l'appel, ces paramètres seront placés sur la même ligne que la macro-instruction et seront séparés par un espace.

Dans le programme de la macro-instruction, la syntaxe « { ?n} » fait référence au paramètre n.

8.6.1.11 LIBRAIRIE

La notion de librairie permet la définition de ressources qui seront compilées une seule fois dans une application, quel que soit le nombre d'appels à ces ressources.

Syntaxe :

#LIBRARY <nom de la librairie>

<programme>

#ENDL

8.6.2 LANGAGE LITTERAL ETENDU

C'est un sur ensemble du langage littéral bas niveau.

Il permet d'écrire plus simplement et sous une forme plus concise des équations booléennes et numériques.

8.6.2.1 EQUATIONS BOOLEENNES

Les opérateurs sont :

- « (/) » : Affectation complémentée
- « (0) » : Mise à zéro
- « (1) » : Mise à 1
- « . » : Et
- « + » : Ou

8.6.2.2 EQUATIONS NUMERIQUES

Les opérateurs sont :

- « + » : Addition
- « - » : Soustraction
- « * » : Multiplication
- « / » : Division
- « < » : Décalage à gauche
- « > » : Décalage à droite
- « & » : ET binaire
- « | » : OU binaire
- « ^ » : OU exclusif

8.6.2.3 STRUCTURE

- Conditionnelles : IF... THEN… ELSE…

Syntaxe :

```
IF (test)
    THEN
    Action si vrai
    ENDIF
```

ELSE
Action si faux
ENDIF

- Tant que : WHILE… ENDWHILE

Syntaxe :

WHILE (test)
Action à répéter tant que le test est vrai
ENDWHILE

8.6.3 LANGAGE LITTERAL ST

Le langage littéral ST est un langage littéral structuré défini par la norme CEI1131-3.

Pour définir des sections en langage littéral ST, il faut :

- « #BEGIN_ST » : marque le début d'une section en langage ST.
- « #END_ST » : marque la fin d'une section en langage ST.

Pour le langage ST, les commentaires doivent débuter par « (* » et se termine par « *) ».
Tous les instructions en langage ST sont terminées par le caractère « ; ».
Pour affecter une valeur à une variable, on utilise « := »

8.6.3.1 STRUCTURES DE PROGRAMMATIONS

La syntaxe des structures est :

- Test SI ALORS SINON : IF condition THEN action ELSE action ENDIF.
- Boucle TANT QUE : WHILE condition DO action ENDWHILE.
- Boucle JUSQU'A CE QUE : REPEAT action UNTIL condition ; ENDREPEAT ;
- Boucle DEPUIS JUSQU'A :
  FOR variable :=valeur de départ TO valeur de fin BY pas DO action ENDFOR ;
- SORTIE de boucle : EXIT

9 CONFIGURATION

9.1 POST-PROCESSEUR

Les post-processeurs sont des modules logicielles permettant de traduire les fichiers de code pivot générés par le compilateur AUTOMGEN en fichiers exécutables sur une cible ainsi que d'assurer la connexion dynamique à la cible.

Le mot « cible » désigne de façon générique un système programmable capable d'exécuter une application.

9.1.1 LES FICHIERS DE CONFIGURATION

Quatre éléments de configuration sont utilisés par chaque post-processeur.

9.1.1.1 SYSTEME

Le système contient la configuration matérielle de la cible, la configuration logicielle, des options permettant de modifier la façon dont le post-processeur génère le code.

9.1.1.2 CORRESPONDANCES DE VARIABLES

Lorsque le post-processeur traduit un fichier du langage pivot d'AUTOMGEN vers un langage cible spécifique, il doit attribuer les variables d'AUTOMGEN à des variables de la cible. Cet élément contient la description précise de l'attribution des variables.

9.1.1.3 CODE CONSTRUCTEUR DEMARRAGE

Cet élément contient du langage machine propre à la cible qui sera placé au début du code exécutable généré par le post-processeur.

9.1.1.4 CODE CONSTRUCTEUR FIN

Cet élément contient du langage machine propre à la cible qui sera placé à la fin du code exécutable généré par le post-processeur.

9.2 OPTIONS DU COMPILATEUR

Cet élément permet de choisir les options de compilation de l'application.

9.3 EXEMPLE : EXECUTEUR PC

Si on choisit la cible « Exécuteur PC », on a alors un automate embarqué sur le PC.

Il contient les éléments suivantes :

- Variables :

Il permet de configurer le nombre de variable pour l'application.

- Exécution :

Il permet de modifier la période d'exécution sur PC.

- Drivers E/S :

En ajoutant les drivers d'E/S, on peut piloter les entrées/sorties connecté directement à l'ordinateur.

9.4 EXEMPLE : POST-PROCESSEUR PL7 (TSX37 ET TSX57)

Il contient les éléments suivantes :

- Système
- Correspondances des variables
- Code constructeur démarrage
- Code constructeur fin
- Option de connexion
- Module de communication

On peut définir les quatres premiers éléments comme des éléments de configuration par défaut.

Remarque :

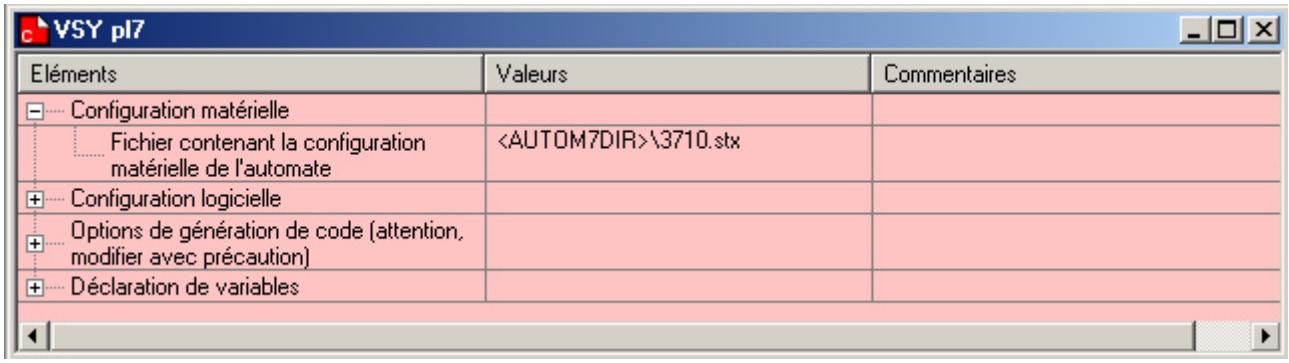
Le fait de définir un élément comme configuration par défaut, entraîne que seule la réinstallation du post-processeur permet de restaurer l'élément de configuration.

9.4.1 SYSTEME

Le système est décomposé de 4 éléments :

- Configuration matérielle
- Configuration logicielle
- Options de génération de code
- Déclaration de variables

9.4.1.1 CONFIGURATION MATERIELLE

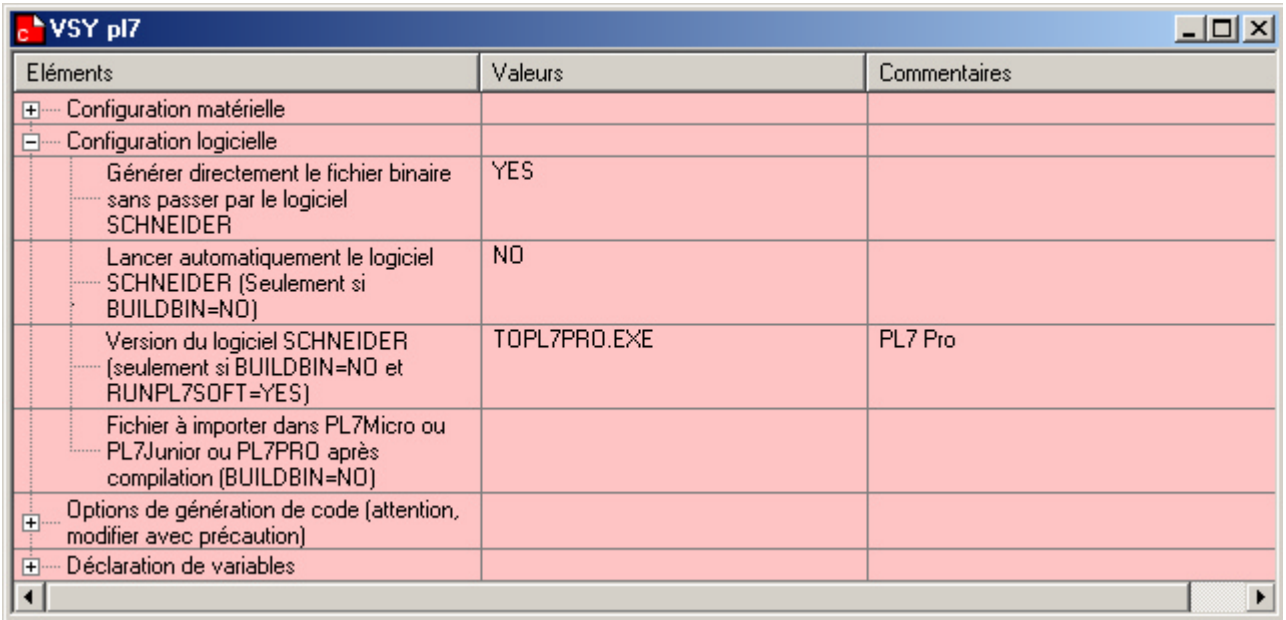


Pour changer de fichier, il faut cliquer 2 fois sur le fichier.

Remarque :

Le fichier de configuration doit être crée avec les outils de programmation SCHNEIDER. Et ensuite, on le sauvegarde dans le répertoire d'installation d'AUTOMGEN ou on l'intègre dans les ressources du projet d'AUTOMGEN.

9.4.1.2 CONFIGURATION LOGICIELLE

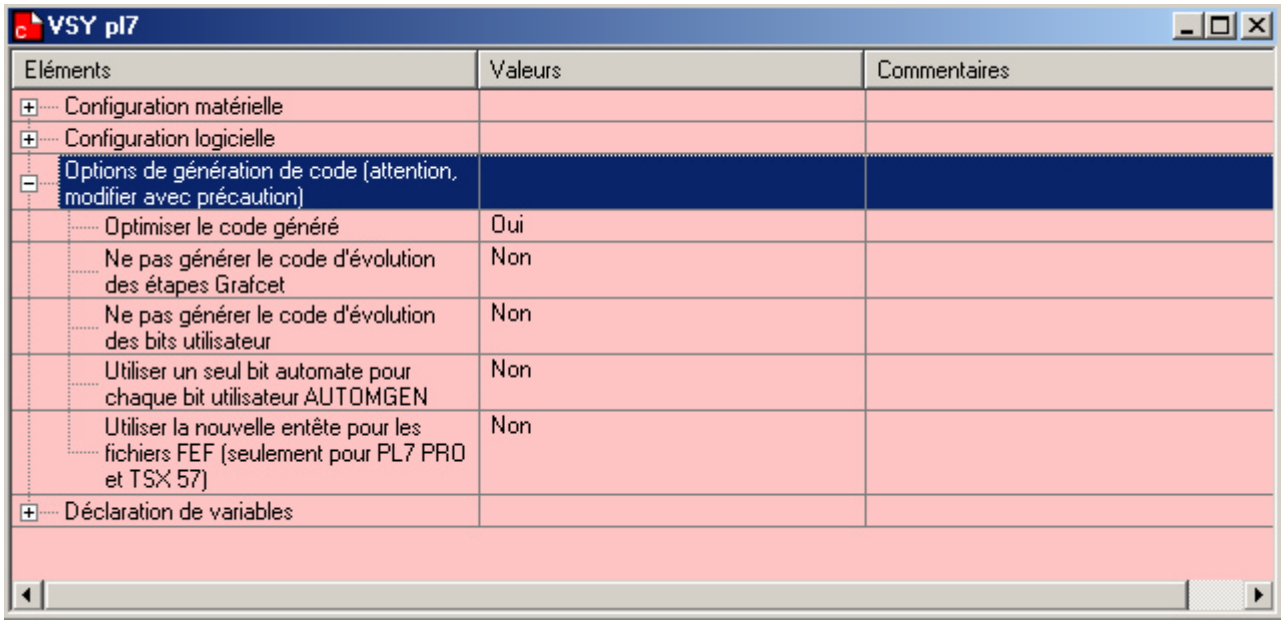


- Générer directement le fichier binaire sans passer par le logiciel SCHNEIDER :

Si on choisit « non », on est dans le mode d'import manuel et on est obligé de choisir un nom de fichier dans l'élément « fichier à importer dans PL7Micro ou PL7Junior ou PL7PRO après compilation » pour l'export vers l'atelier logiciel SCHNEIDER.

Si on choisit « oui », on est dans le mode d'import automatique et on doit rentrer la version du logiciel utilisé dans l'élément « Version du logiciel SCHNEIDER »

9.4.1.3 OPTION DE GENERATION DE CODE



- Optimiser le code généré :

Généralement réglé sur « oui ». le réglage sur « non » peut permettre une analyse plus aisée du code généré.

- Ne pas générer le code d'évolution des étapes grafcet :

Si réglé sur « oui », on doit écrire dans l'élément « code constructeur fin » les instructions permettant la recopie des états immédiats de variables booléennes vers les états passés.

L'état immédiat, c'est l'état écrit par la dernière instruction exécutée par le cible se reportant à cette variable, ou, à défaut celui qu'avait la variable à la fin du dernier cycle d'exécution, ou, à défaut, si c'est le premier cycle d'exécution l'état d'initialisation de la variable.

L'état passé de la variable, c'est l'état qu'avait la variable à la fin du dernier cycle d'exécution.

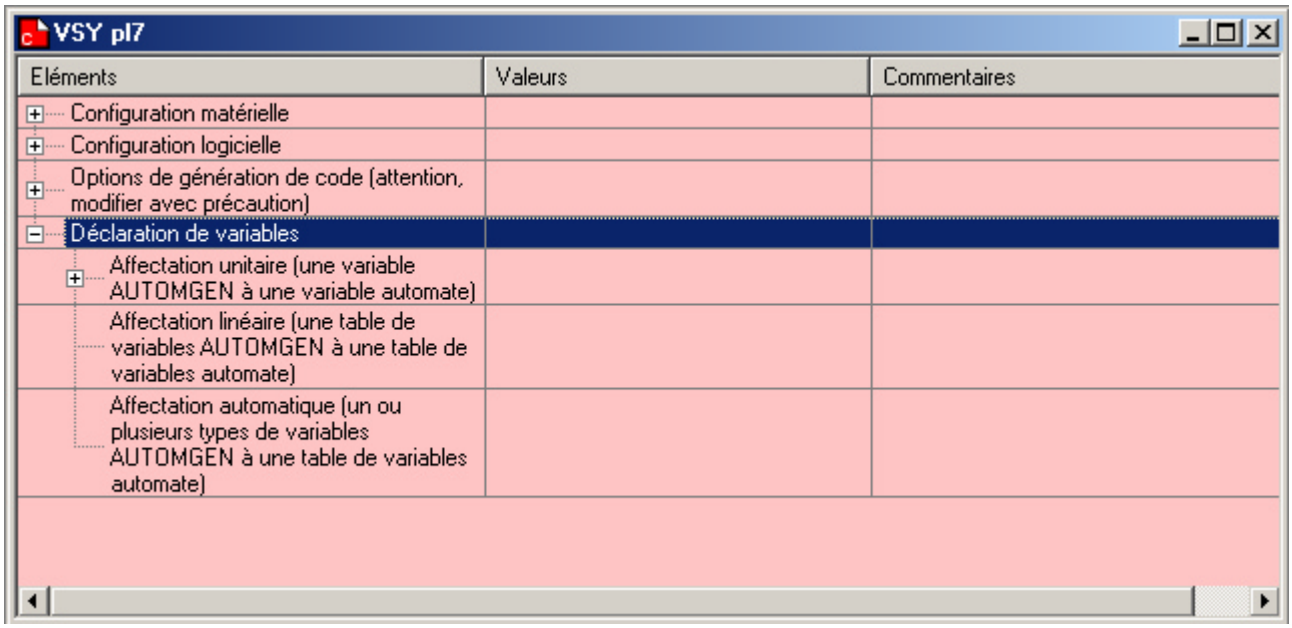
De ce faite, le code généré par le compilateur AUTOMGEN assume ce qui suit :

- Une affectation de variable booléenne se fait sur son état immédiat
- Un test de variable booléenne se fait sur son état passé.

- Ne pas générer le code d'évolution des bits utilisateurs :

Identique à l'option précédente mais appliquée au bits utilisateurs.

9.4.1.4 DECLARATION DE VARIABLES SYSTEME



- Affectation unitaire :

Elle permet d'associer une variable AUTOMGEN à une variable de la cible. Elle doit être utilisée uniquement si une seule déclaration est nécessaire.

Cette déclaration utilise deux informations : le nom de la variable AUTOMGEN et le nom de la variable de la cible.

- Affectation linéaire :

Elle permet d'associer une série de variables consécutives d'AUTOMGEN à une série de variables consécutives de la cible.

Cette affectation est typiquement utilisée pour :

  - o la déclaration des variables d'entrées/sorties
  - o la déclaration de tables de bits ou de mots devant avoir une adresse fixe

Cette déclaration utilise trois informations : le nom de la première variable AUTOMGEN, le nom de la première variable de la cible et la dimension de la table en nombre de variable.

- Affectation automatique :

Elle permet d'associer un ou plusieurs types de variables AUTOMGEN à une plage de variables de la cible.

Cette affectation laisse le soin au compilateur de trouver une affectation à chaque varaiable présenta dans le code généré de la déclaration.

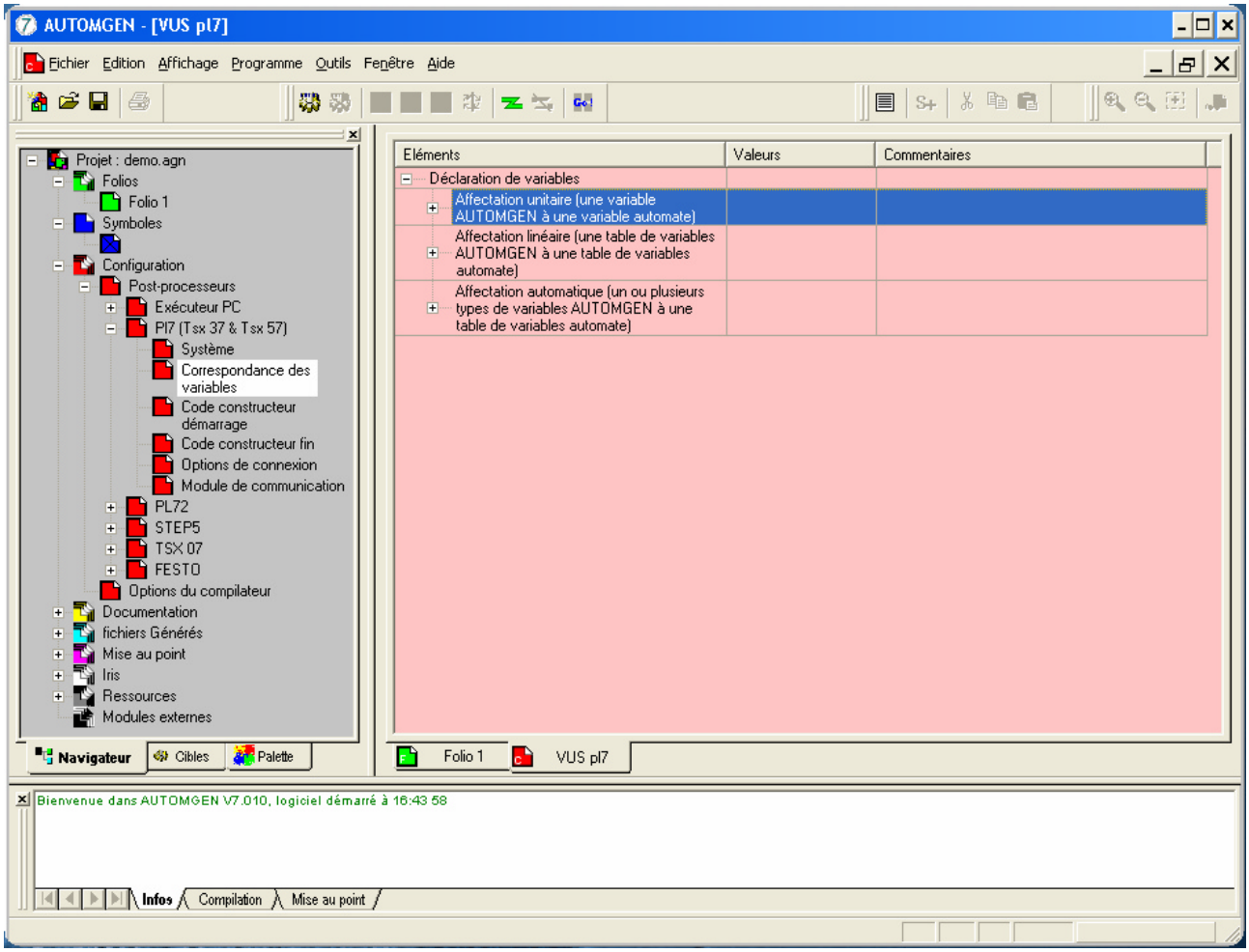
Cette déclaration utilise trois informations : le type de la variable AUTOMGEN, le nom de la première variable de la cible et le numéro de la dernière variable incluse de la plage de la cible.

9.4.2 CORRESPONDANCES DE VARIABLES

La correspondances de variables permet d'attribuer des variables AUTOMGEN à des variables de la cible.

La déclaration se fait de la même façon que dans l'élément « système ».

Sélectionner le document <***Correspondance des variables***> dans le dossier <***Configuration/Post-processeurs/Pl7 (Tsx37 & Tsx 57)***>

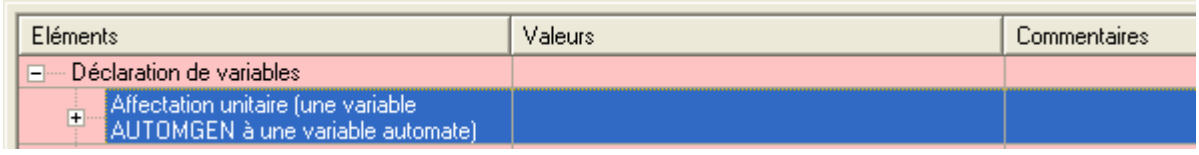


AUTOMGEN propose 2 types d'adressage :

- l'***Affectation unitaire*** : une variable AUTOMGEN est affectée à une adresse de l'automate,
- l'***Affectation linéaire*** : les variables AUTOMGEN sont affectées automatiquement et de façon linéaire aux adresses de l'automate.

9.4.2.1 l'Affectation unitaire :

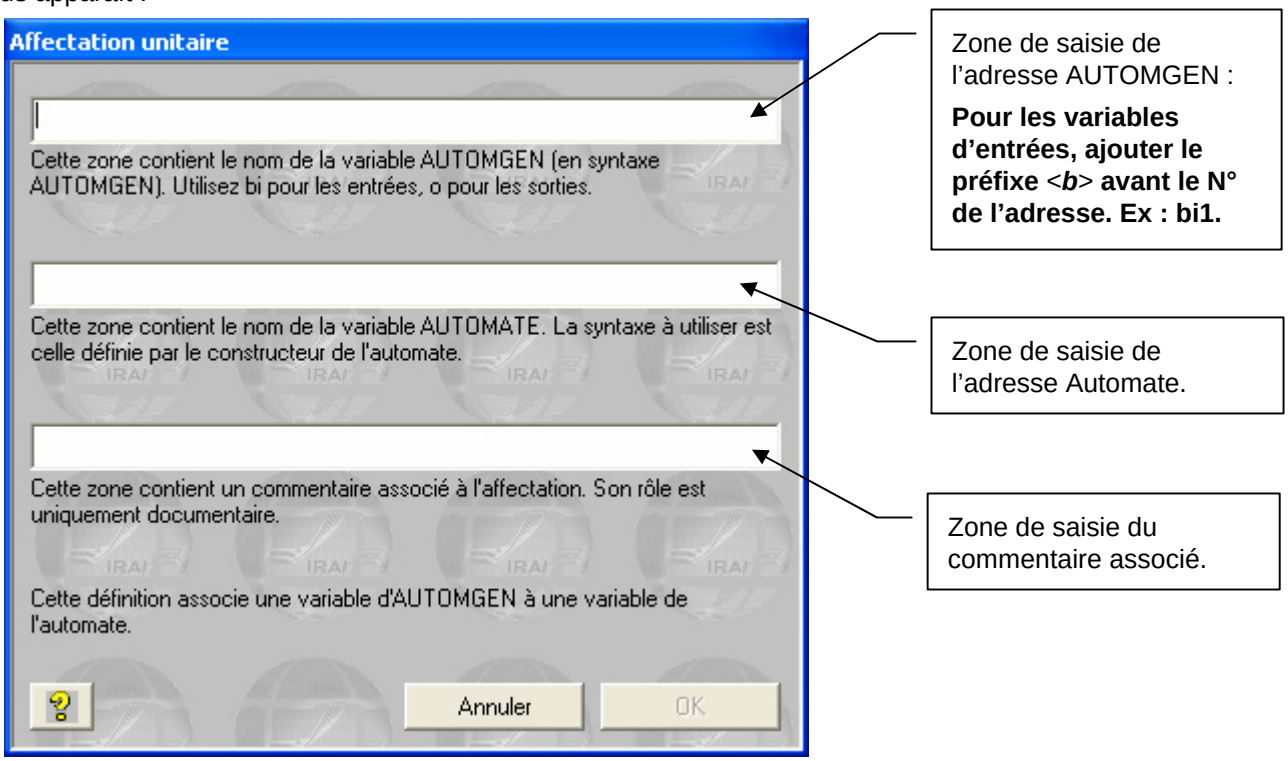
Pour affecter unitairement les variables AUTOMGEN aux adresses API, cliquer avec le **bouton droit** de la souris sur <***Affectation unitaire***>. Sélectionner <***Ajouter***> dans le menu qui apparaît.



Sélectionner <***Ajouter***> dans le menu qui apparaît.

Si plusieurs types de cibles sont gérés par le post-processeur, la boîte de dialogue ci-dessous permet de déterminer si la nouvelle affectation est seulement pour un type en particulier ou pour tous les types.

Sélectionner le <***Domaine de validité de l'affectation***> puis valider, la fenêtre <***Affectation unitaire***> ci-dessous apparaît :



Après avoir complété les zones de saisie, valider. L'adressage apparaît dans la liste ci-dessous.

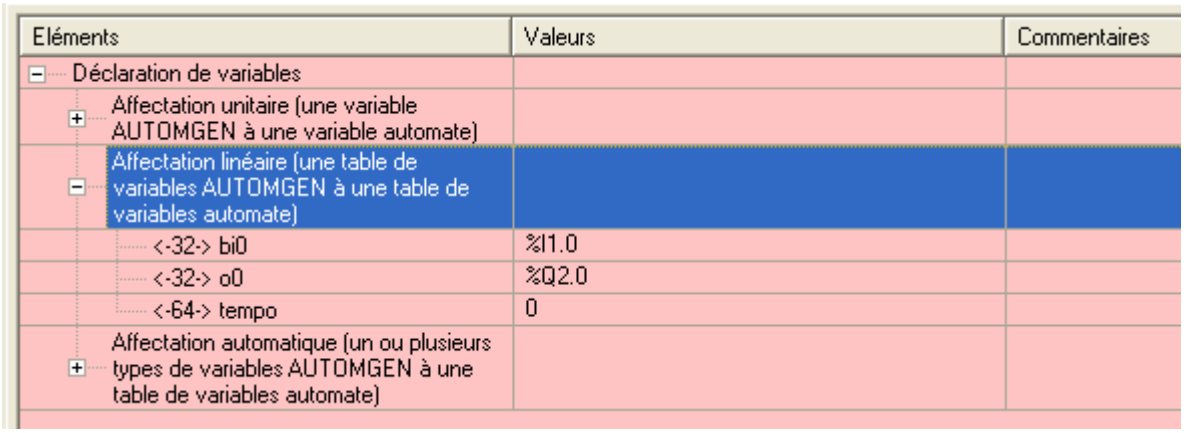
Eléments	Valeurs	Commentaires
Déclaration de variables		
Affectation unitaire (une variable AUTOMGEN à une variable automate)		
m62	%SW30	temps de cycle en ms
b8	%S11	débordement chien de garde
bi1	%i0.5	BP départ cycle
bi2	%i0.8	
o1	%q1.2	
o11	%q1.14	
Affectation linéaire (une table de variables AUTOMGEN à une table de variables automate)		
Affectation automatique (un ou plusieurs types de variables AUTOMGEN à une table de variables automate)		

Une fois l'adressage unitaire terminé, il faut supprimer les zones d'affectation définies dans l'affectation linéaire.

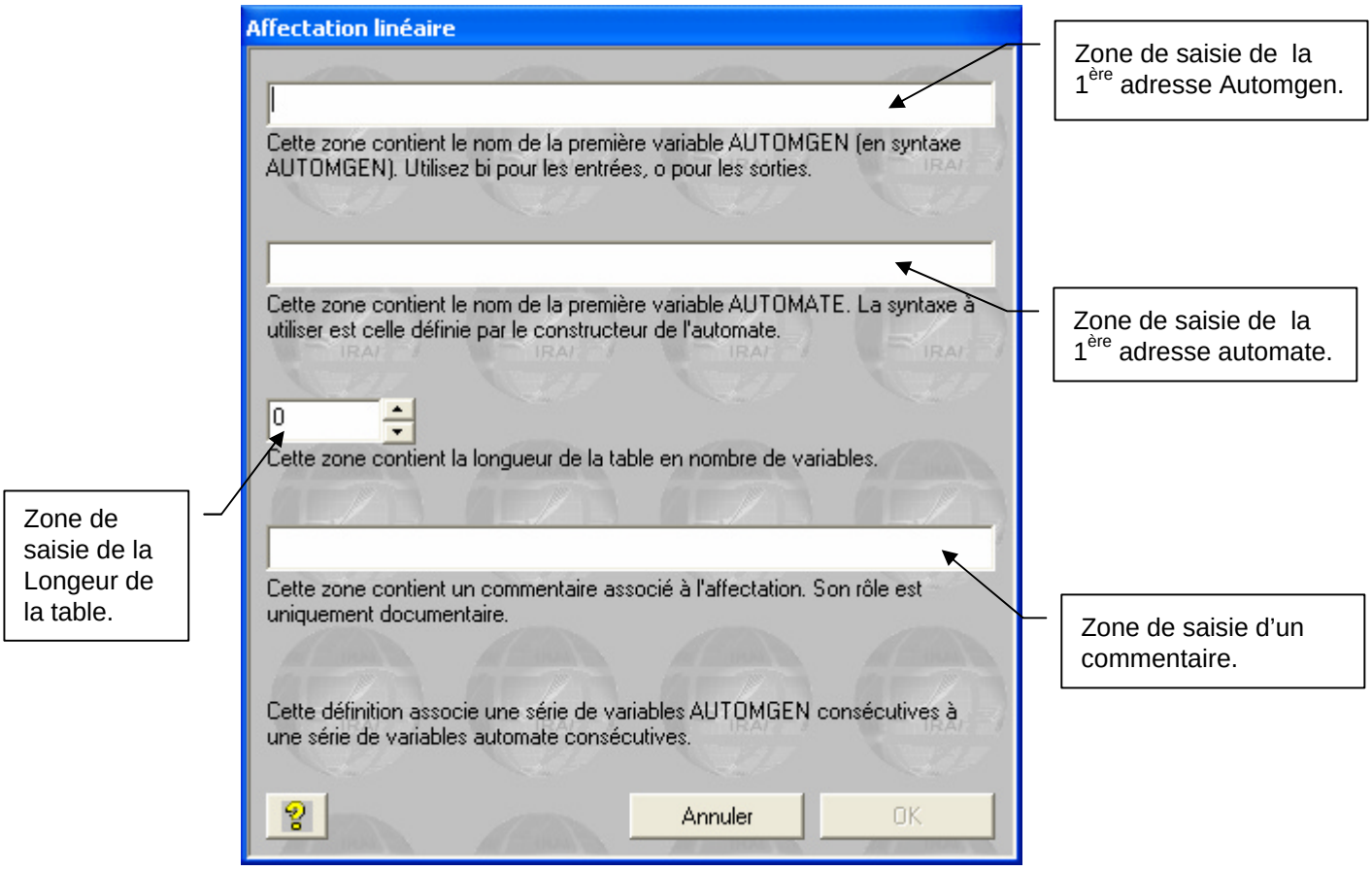
9.4.2. l'Affectation linéaire :

L'affectation linéaire permet d'affecter automatiquement une table de variables AUTOMGEN à une table d'adresses API. Pour cela l'utilisateur doit définir des zones d'affectation.

Pour ajouter une zone d'affectation, cliquer avec le **bouton droit** de la souris sur <***Affectation unitaire***>. Sélectionner <***Ajouter***> dans le menu qui apparaît.

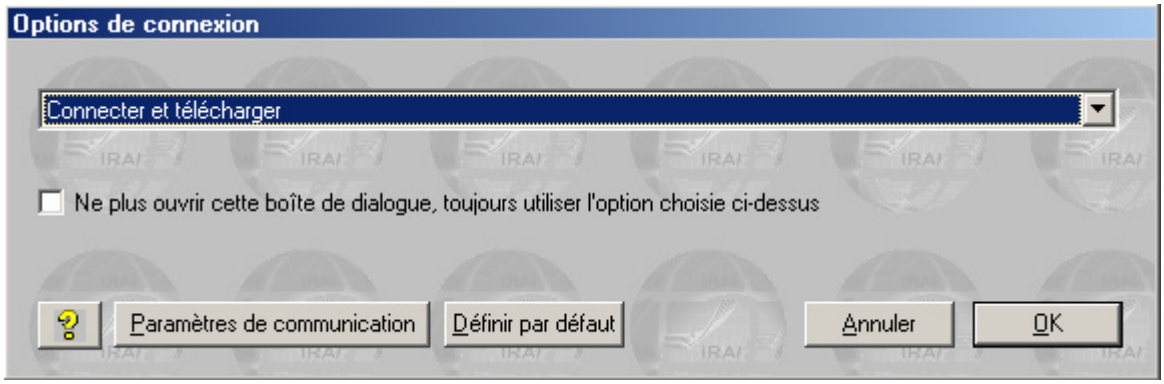


Compléter la boîte de dialogue suivante :



9.4.3 OPTION DE CONNEXION

Cet élément permet de choisir le mode de connexion et de définir les paramètres de communication.



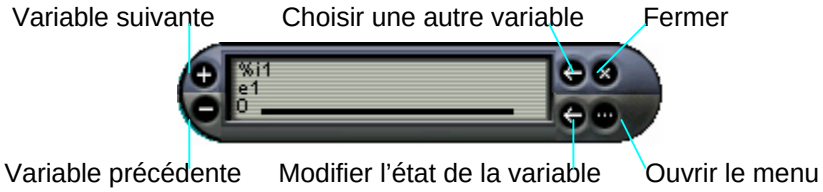
Cette boîte de dialogue s'ouvre automatiquement lorsqu'une connexion à une cible est demandée. En cochant la case « ne plus ouvrir… » , cette ouverture n'est plus automatique. Pour l'ouvrir de nouveau, laissez enfoncée la touche [SHIFT] du clavier en lançant la commande de connexion ou la commande « GO ».

9.4.4 MODULE DE COMMUNICATION

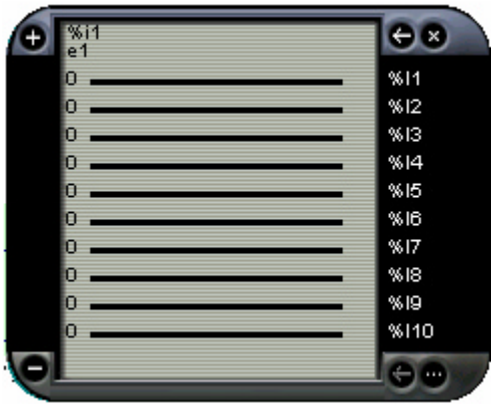
Cet élément permet de définir le port de communication et le paramétrage du port.

10 MISE AU POINT

Cette outil permet de créer des « fenêtres de monitoring » qui permettent de visualiser ou de forcer l'état des variables.

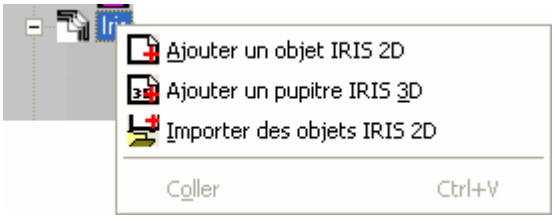


Si on choisit dans le menu du monitoring une table de variable, on obtient le monitoring suivant :



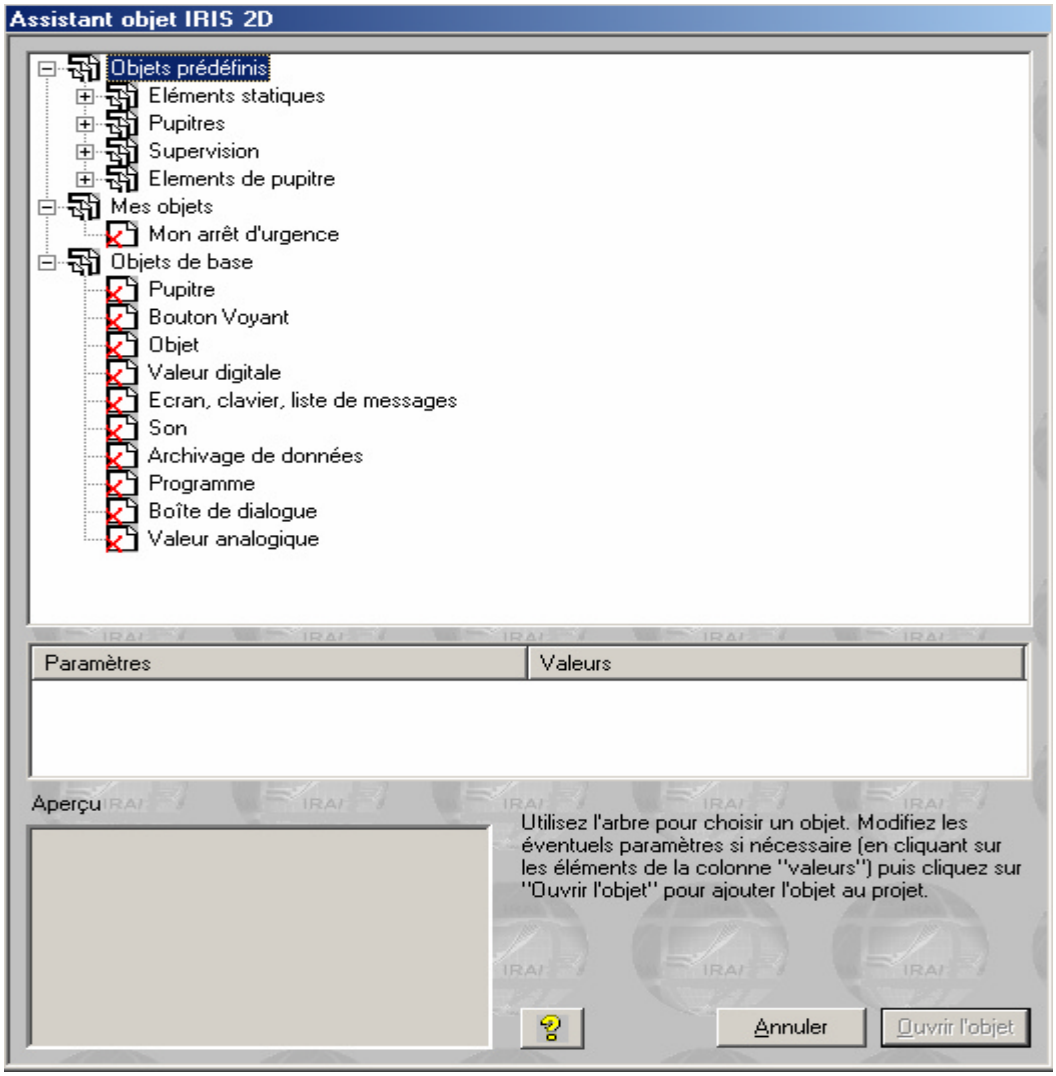
11 - MODULES IRIS

Pour ajouter un pupitre ou un objet IRIS 2D ou 3D, sélectionner le dossier <***Iris***> dans le navigateur puis cliquer sur le bouton droit de la souris.



11.1 IRIS 2D

Pour ajouter un pupitre ou un objet IRIS 2D, sélectionner <***Ajouter un objet IRIS 2D***>. La fenêtre <***Assistant objet IRIS 2D***> apparaît à l'écran :



On choisit d'abord un pupitre.

11.1.1 PUPITRE

- Mode Utilisateur :

- Mode Configuration :

En cliquant à droite de la souris dans le pupitre, on obtient ses caractéristiques c'est à dire le mode configuration :

: Effacement d'objet

: Passer l'objet en mode Utilisateur

: Modifier les caractéristiques de l'objet

... : Ce bouton n'existe que pour l'objet « pupitre », car il permet de gérer les objets sur sa surface : Ajout, Suppression, ….

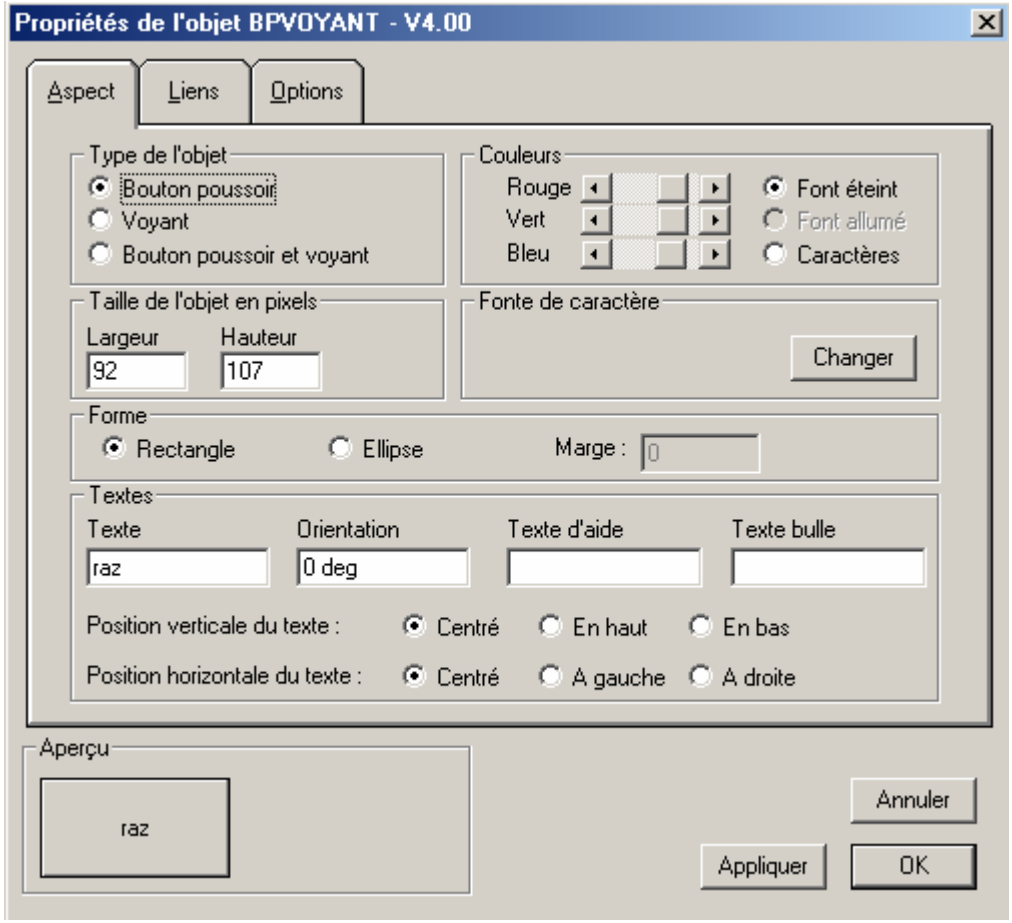
Remarque : Les 3 premières boutons se retrouvent sur tous les objets qu'on créera dans le pupitre. L'obtention du mode configuration se fait toujours par clic du bouton droit de la souris sur l'objet correspondant.

11.1.2 INSERTION OBJET DANS LE PUPITRE

L'ajout d'un objet dans le pupitre se fait par le bouton ... , soit par le bouton droit de la souris sur l'objet pupitre du navigateur.

En choisissant par exemple un bouton voyant, on a :

En cliquant sur , on a les caractéristiques du bouton :



- Onglet « Aspect » : permet de définir la forme de l'objet.
- Onglet « liens » : permet de définir les actions à effectuer
- Onglet « Options » : permet de définir le type du bouton

11.2 IRIS 3D

IRIS 3D permet de créer des applications de simulation de parties opératives 3D.

Il permet d'animer des objets 3D crées dans des modeleurs standards : 3DStudio, SOLIDWORKS, SOLIDCONCEPT,…

La forme native des fichiers traité par IRIS 3D est celui des fichiers DIRECTX 8 « .X » de Microsoft.

Un convertisseur du format « .3DS » vers « .X » est intégré à l'environnement.

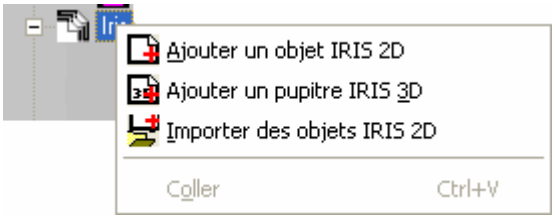
De plus l'utilitaire CROSSROADS fourni sur le CD-ROM d'installation permet de convertir le format 3D vers « .3DS ».

Remarque :

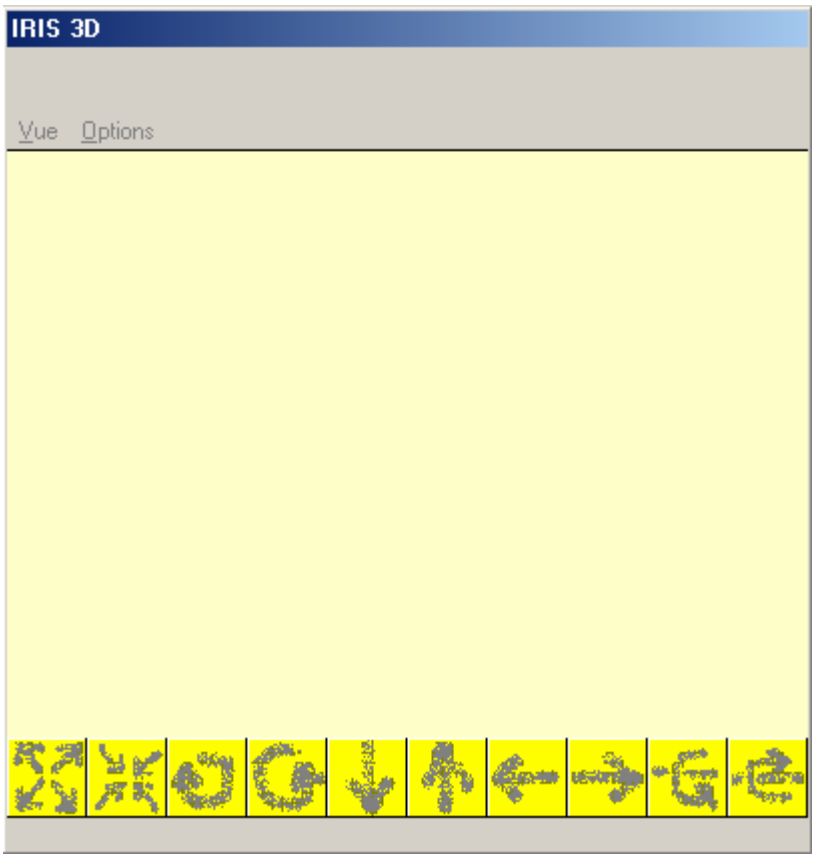
Avant de commencer à créer des animations en 3D, il faut importer les images dans la ressources sous format « .X ».

11.2.1 PUPITRE

Pour ajouter un pupitre ou un objet IRIS 3D, cliquer avec le bouton droit de la souris sur le dossier <***Iris***> dans le navigateur d'AUTOMGEN, puis sélectionner <***Ajouter un objet IRIS 3D***> dans le menu suivant :

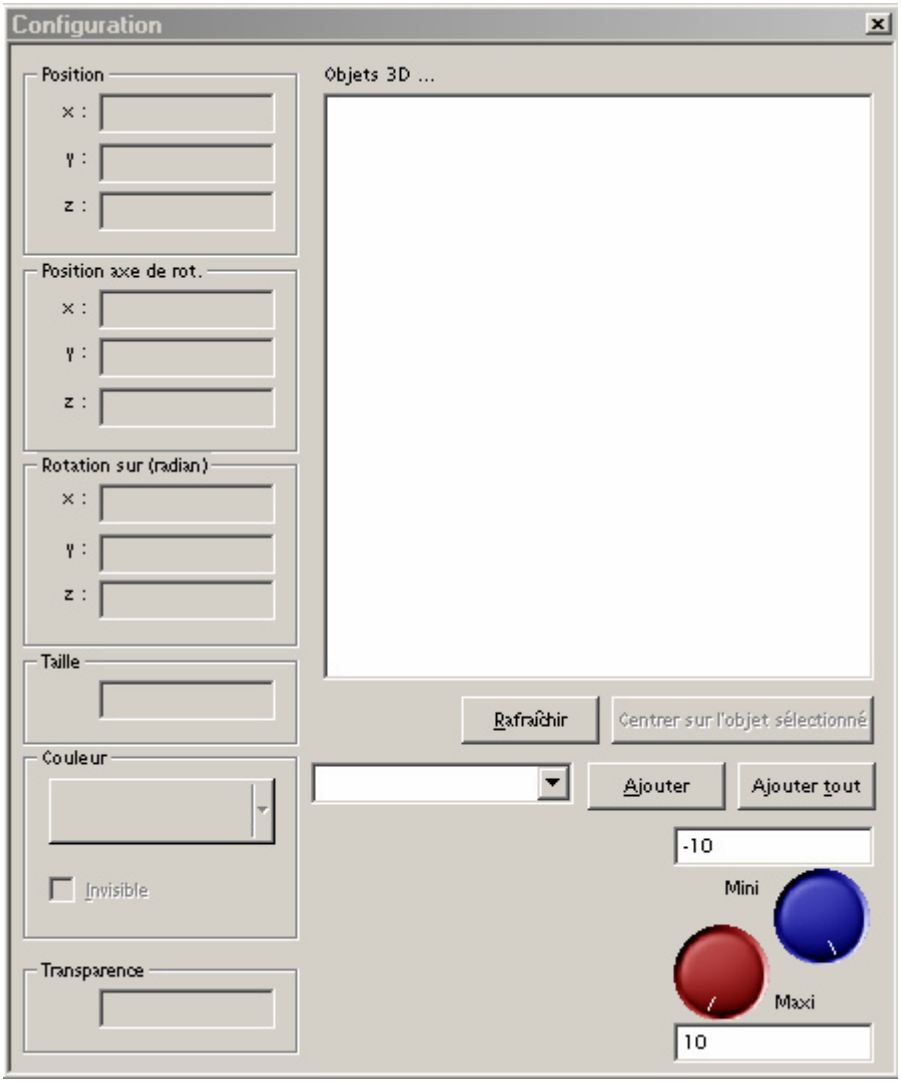


Le pupitre 3D se présente de la façon suivantes :



11.2.2 INSERTION OBJET DANS LE PUPITRE

Il faut sélectionner « Ouvrir la fenêtre de configuration » dans le menu « Options » de la fenêtre IRIS 3D.



Pour ajouter un objet 3D présents dans la ressources, on clique sur l'élément  et tous les éléments présents dans la ressources apparaissent.

Par exemple :

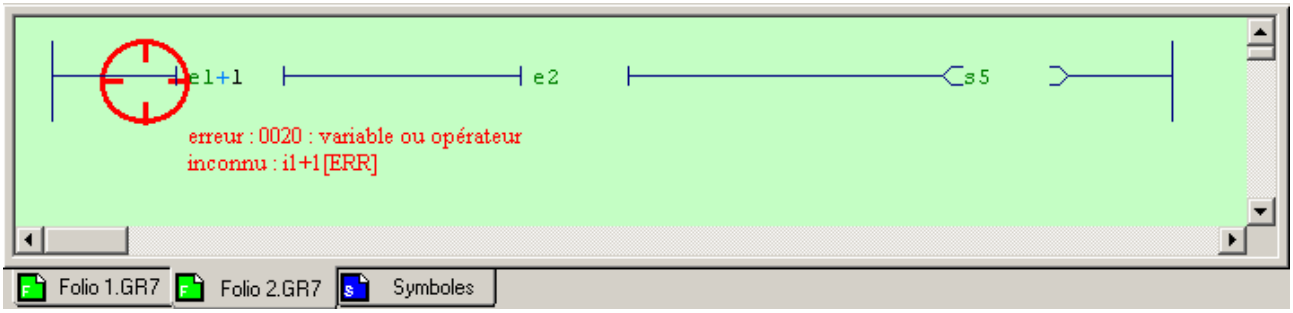
Ensuite on paramètre l'objet pour son animation.

12 EXECUTER UNE APPLICATION

12.1 COMPILATEUR

Le compilateur  traduit les folios en un ensemble d'équations de langage pivot.

Pour localiser une erreur, il faut faire un double clic sur le message d'erreur et le logicielle renvoie sur l'erreur.



12.2 EXECUTION

Le bouton GO représente la méthode la plus rapide pour observer le résultat de l'exécution d'une application.

Il active les mécanisme suivants :

- Compilation de l'application si elle n'est pas à jour
- Installation du module d'exécution
- Passage de la cible en RUN
- Activation de la visualisation dynamique

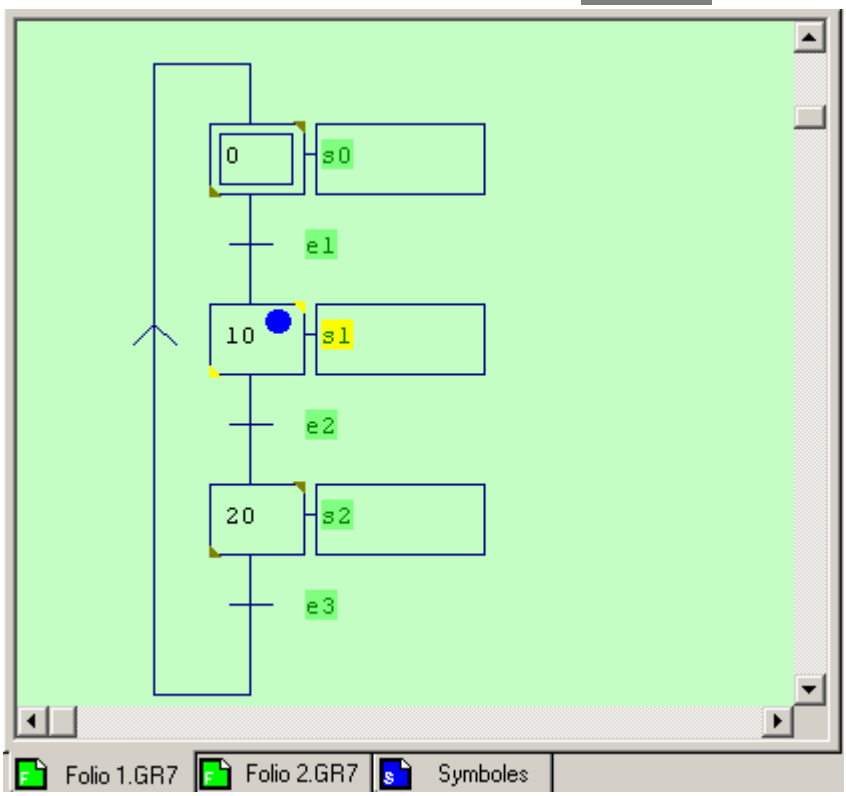
Sinon on peut lancer par le bouton RUN, une fois qu'on a compilé le projet sans erreurs.

Remarque :

L'activation de la visualisation dynamique se fait par le bouton 

Au niveau de la visualisation en dynamique sur les folios :

- Vert correspond à l'état 0
- Jaune correspond à l'état 1



12.3 SUR PC

Dans la fenêtre « projet », on sélectionne l'onglet « cibles ».

Ensuite on sélectionne l'option « Exécuteur PC » pour exécuter l'application sur PC.

12.4 SUR AUTOMATE

Dans la fenêtre « projet », on sélectionne l'onglet « cibles ».

Ensuite on sélectionne l'automate correspondant.

On compile l'application et on lance la connexion avec laquelle on doit définir le type de liaison avec l'automate. Ceci permet de transférer le programme dans l'automate.