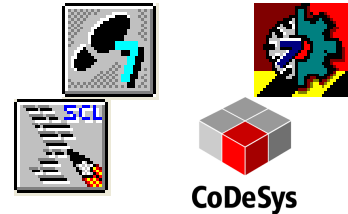


Normalisation de la symbolique

Approche et découverte de l'utilisation d'une symbolique.



Généralités

Dans les travaux de programmation, les variables sont écrites suivant une symbolique particulière. Cette démarche est faite dans un but de standardisation et de ligne de conduite. La symbolique nous apporte plusieurs informations, elle nous donne des renseignements sur le type de variables, une courte désignation de sa fonction et son genre.

La programmation en langage « texte structuré » est une suite de « phrases » représentant différentes instructions, des variables, des déclarations etc. Sa clarté de lecture peut en être affaiblie si la symbolique des variables est trop longue ou mal formulée. Il est donc important d'établir certaines règles.

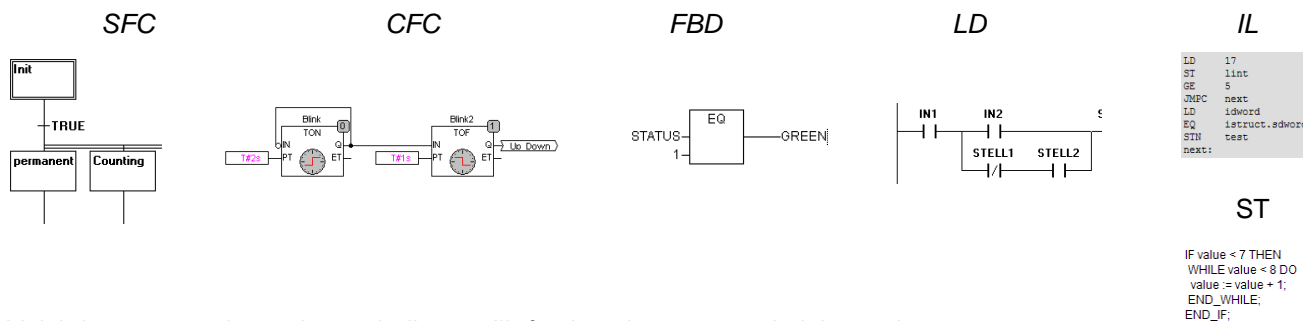
Cette symbolique s'applique également à d'autres langages que le « texte structuré » :

Langages graphiques.

- SFC (Sequential Function Chart) Diagramme Fonctionnel en Séquence
- CFC (Continuous Function Chart) Blocs fonctionnels graphiques
- FBD (Function Block Diagram) Blocs Fonctionnels
- LD (Ladder Diagram) Langage à Contacts

Langages textes.

- IL (Instruction List) Liste d'Instruction
- ST (Structured Text) Littéral Structuré

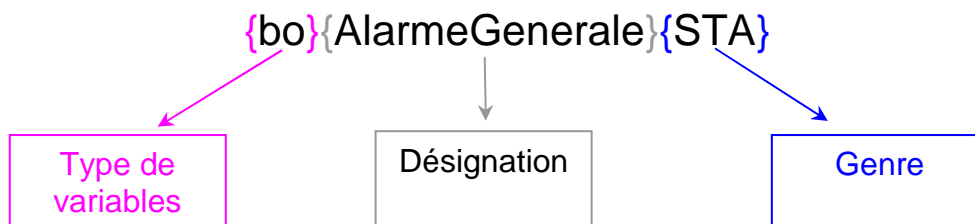


Voici donc une variante de symbolique utilisée dans les travaux de laboratoire.

Exemples :

```

boALM_CFn1_GLO
inVitesseActuelleMotMEM
fbTimerStartRequis
boBP_StartGIN
...
  
```



Type de variables.

Les deux premières lettres sont réservées pour le type de variables. Le type de variables nous renseigne sur le format utilisé. Il est important de connaître le format des variables lorsque l'on fait des opérations mathématiques ou de transfert car elles doivent être compatibles les unes aux autres. L'importance de cette information se fait sentir lorsque vous travaillez avec un nombre considérable de variables globales et locales et qu'il n'est pas toujours facile de retrouver l'endroit de leur déclaration. Le type s'écrit en minuscule. Voici une liste des différents types possibles :

Type symbolique	Type	Valeur min	Valeur max	Mémoire requise
bo	BOOL	FALSE ou « 0 »	TRUE ou « 1 »	1/8 bit
by	BYTE	0	255	8 bits
wo	WORD	0	65535	16 bits
dw	DWORD	0	4294967295	32 bits
si	SINT	-128	127	8 bits
us	USINT	0	255	8 bits
in	INT	-32768	32767	16 bits
ui	UINT	0	65535	16 bits
ud	UDINT	0	4294967295	32 bits
di	DINT	-2147483648	2147483647	32 bits
re	REAL	~ -3.402823 x 10 ³⁸	~ 3.402823 x 10 ³⁸	32 bits
lr	LREAL	~ -1.79769313486231E308	~ 1.79769313486232E308	64 bits
st	STRING		Max 80 caractères	80 + 1 bytes
da	DATE	D#1970-01-01	D#2106-02-06	32 bits
dt	DATE_AND_TIME	DT#1970-01-01-00 :00	DT#2106-02-06-06 :28 :15	32 bits
ti	TIME	T#0ms	T#71582m47s295ms	32 bits
td	TIME_OF_DAY	TOD#00 :00	TOD#1193 :02 :47.295	32 bits

		Caractéristiques	Remarques	
su	STRUCT	Type de données utilisateur		
ar	ARRAYS	Type de données utilisateur	Jusqu'à 3 dimensions	
po	POINTER	Type de données utilisateur	Accès direct par adresse	
en	ENUM	Type de données utilisateur	Déclaration en tant que nom (caractères)	
al	ALIAS	Type de données utilisateur	Déclaration en tant que caractères	
fc	FUNCTION	Fonction		
fb	FUNCTION_BLOCK	Bloc fonctionnel		
ac	ACTION			
pr	PROGRAMME			

s5	S5_TIME	Valeur de temporisation	<i>types ne concernant que les symboliques en programmation Step7</i>	
ut	UDT	Type de données utilisateur		
db	DBx	Bloc de données		
sd	SDBx	Bloc de données système		
ob	OBx	Bloc opérationnel		
tr	TIMER	Temporisation		
cr	COUNTER	Compteur		
an	ANY	Paramètre ANY		

Désignation.

La désignation représente la définition de la variable. Il faut qu'elle soit la plus claire possible. N'hésitez pas à utiliser le « _ » pour séparer des chiffres. L'espace n'est pas autorisé, certains éditeurs ne l'acceptent pas. La symbolique d'une variable ne doit pas atteindre une longueur trop importante, c'est pourquoi un nombre maximum de caractères doit être imposé. Ce choix se porte à 24 caractères au total, y compris le type et le genre, (une tolérance de +2 est possible pour les cas ambigus). Chaque mot commence par une majuscule de manière à éviter le « _ » qui tend à diminuer le nombre de caractères significatifs.

Les « ^ » ne sont pas acceptés par le compilateur en SCL.

Exemples de désignations :
 ..ALM_CFn1_...
 ..VitesseActuelleMot...
 ..TimerStartRequis
 ...BP_Start...

A cette désignation vient s'ajouter une liste d'abréviations de termes souvent utilisés. Cela permet de raccourcir la symbolique et d'utiliser les mêmes lettres lors d'appellations d'éléments.

Exemple : Pour nommer un bouton poussoir, on utilisera toujours l'abréviation « BP ».

<i>Abréviation</i>	<i>terme</i>
ALM	Alarme (en)
AS	Alarme sonore
AU	Arrêt d'urgence
BP	Bouton poussoir
CPI	Capteur de proximité inductif
CPC	Capteur de proximité capacitif
CPO	Capteur de proximité optique
CPL	Capteur de proximité laser
CO	Contacteur
CF	Convertisseur de fréquence
DEF	Défaut (en)
DI	Disjoncteur
EM	Electro-aimant
IP	Interrupteur de position
IR	Interrupteur rotatif
JO	Joystick
MO	Moteur
PO	Pompe
PT	Pressostat
RE	Régulateur
RL	Relais
TH	Thermique
TT	Thermostat
SO	Sonde de mesure
VO	Voyant
VA	Vanne
VE	Vérin pneumatique ou hydraulique
VT	Vacuostat

Il est parfois intéressant d'utiliser des termes anglais, cela raccourci certaines symboliques (Vitesse => Speed ou Démarrage => Start etc).

Remarque :

Dans la plupart des entreprises, les commentaires de la programmation se font en anglais. C'est, malheureusement pour nous (francophone), la langue la mieux acceptée pour ce type d'activité, ceci en raison de l'exportation réalisée par l'industrie en Suisse. Et c'est une langue bien acceptée, par les clients, dans le monde (que ce soit pour des raisons de compréhension ou de respect des normes).

Genre.

Le genre sert à clarifier la catégorie de la variable. Par exemple, elle peut être une variable locale ou globale, elle peut être utilisée comme paramètre d'entrée ou de sortie d'une fonction etc.

Le genre s'écrit en majuscule et termine la symbolique. Il n'est autorisé qu'un maximum de trois caractères. Il est possible qu'une variable n'ait pas de genre déterminé, dans ce cas, la symbolique n'en comportera pas (exemple : fcGraphEtats).

Pour le dissocier de la désignation, le genre doit être écrit en majuscule.

Exemples de genre pour la symbolique PLC Codesys (TwinCat, Wago, Rexroth, Elau, Festo, ...)

	Genre	Type	Commentaires	Remarques
Locales	PIN	VAR_INPUT	Paramètre d'entrée local	
	POU	VAR_OUTPUT	Paramètre de sortie local	
	PIO	VAR_IN_OUT	Paramètre d'entrée/sortie local	
	LOC	VAR	Variable locale	
	LIN	VAR (Local IN => AT %I*)	Variable d'assignation I/O (locale)	Valable pour un bloc fonctionnel
	LOU	VAR (Local OUT => AT %Q*)	Variable d'assignation I/O (locale)	Valable pour un bloc fonctionnel
	LRE	RETAIN	Variable rémanente locale	Perd sa rémanence dans une fonction !
	LPE	PERSISTENT	Variable persistante locale	
Globales	LCO	CONSTANT	Variable constante locale	
	LEX	EXTERNAL	Variable globale pouvant être importée dans un programme	
	GIN	VAR_GLOBAL (IN => AT %I*)	Variable globale d'assignation I/O	
	GOU	VAR_GLOBAL (OUT => AT %Q*)	Variable globale d'assignation I/O	
	GLO	VAR_GLOBAL	Variable globale	
	GRE	VAR_GLOBAL RETAIN	Variable globale rémanente	
	GPE	VAR_GLOBAL PERSISTENT		
	GCO	VAR_GLOBAL CONSTANT	Variable constante globale	

Remarque concernant les variables rémanentes et persistantes.

Lors de l'arrêt de TwinCat, les valeurs de ces variables sont sauvegardées dans deux fichiers sur le disque dur du système. Le répertoire peut être spécifié dans le système TwinCat en l'indiquant dans ces propriétés (PLC tab). Le réglage standard est : « Drive »:\TwinCatBoot. Les fichiers ont un nom fixe et une extension définie.

File name	Description
TCPLC_P_x.wbp	Boot project (x = number of the run-time system)
TCPLC_R_x.wbp	RETAIN variables (x = number of the run-time system)
TCPLC_T_x.wbp	PERSISTENT variables (x = number of the run-time system)
TCPLC_R_x.wb~	Backup copy of the RETAIN variables (x = number of the run-time system)
TCPLC_T_x.wb~	Backup copy of the PERSISTENT variables (x = number of the run-time system)

Exemples de genre pour la symbolique PLC Step7

Un « MWxx, MBxx ou Mxx » sera considéré comme une globale de genre MEM.

	Genre	Type	Commentaires	Remarques
Locales	LST	VAR (STAT)	Variable statique (instance) (locale)	N'existe que dans les blocs fonctionnels
	LTP	VAR_TEMP	Variable temporaire (locale)	
	PIN	VAR_INTPUT	Paramètre d'entrée (local)	
	POU	VAR_OUTPUT	Paramètre de sortie (local)	
	PIO	VAR_IN_OUT	Paramètre d'entrée/sortie (local)	
Globales	LCT	CONSTANT	Variable constante (locale)	
	GIN	I	Variable globale d'assignation I/O (globale)	
	GOU	Q	Variable globale d'assignation I/O (globale)	
	GME	MEMENTO	Variable globale memento (globale)	
	GPI	PI	Entrée périphérique directe (globale)	
	GPQ	PQ	Sortie périphérique directe (globale)	
	GDB	Data	Variable de données (globale)	
GDS	Data système	Variable de données systèmes (globale)		