

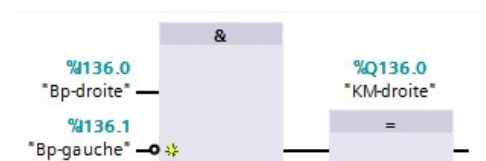


Dossier ressource

Automates SIEMENS

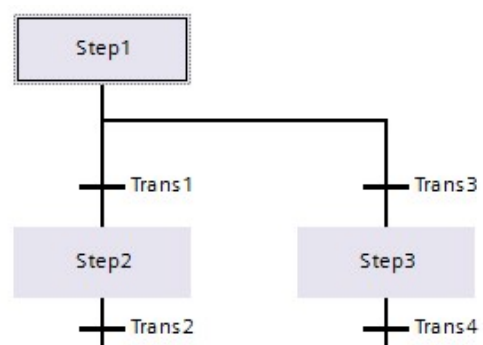
Logiciel TIA PORTAL

STEP 7



"KM-droite" := "Bp-droite" AND NOT "Bp-gauche";

1	A	"Bp-droite"	%I136.0
2	AN	"Bp-gauche"	%I136.1
3	=	"KM-droite"	%Q136.0



SOMMAIRE

<u>Introduction</u>	<u>2</u>
<u>Lancement du logiciel</u>	<u>2</u>
<u>Présentation de l'environnement TIA PORTAL</u>	<u>3</u>
☐ Vue du portail	3
☐ Vue du projet	4
☐ Navigateur de projet	5
☐ Task Cards	6
☐ Fenêtre d'inspection	7
<u>Création d'un projet TIA PORTAL</u>	<u>7</u>
☐ Sélection des paramètres de sécurité	7
☐ Créer une station matérielle	7
☐ Configurer la station matérielle	9
Adresses Ethernet de l'API	9
Paramétrage protection et sécurité	9
Paramétrage mise en route de la CPU	9
Paramétrage Mémento de cadence	10
Configuration des Entrées (%I) et Sorties (%Q) dans la CPU et dans les modules	10
☐ Compiler la configuration matérielle	11
☐ Charger la configuration matérielle	12
☐ Création d'un programme	12
☐ Compiler les blocs programme	17
☐ Charger les blocs dans la CPU	18
☐ Visualiser les blocs	19
☐ Visualiser / forcer des variables : table de visualisation	19
<u>Programmation Temporisations</u>	<u>21</u>
☐ Langage CONT	21
☐ Langage SCL	22
<u>Programmation Compteurs</u>	<u>23</u>
☐ Par incrémentation d'un mot en langage CONT	23
☐ Par utilisation des blocs compteurs intégrés en langage CONT	24
☐ Par utilisation des blocs compteurs intégrés en langage SCL	25
<u>Programmation langage GRAPH</u>	<u>26</u>

Introduction

La plateforme **Totally Integrated Automation Portal** est l'environnement de travail Siemens permettant de mettre en œuvre des solutions d'automatisation avec un système d'ingénierie intégré comprenant les logiciels :

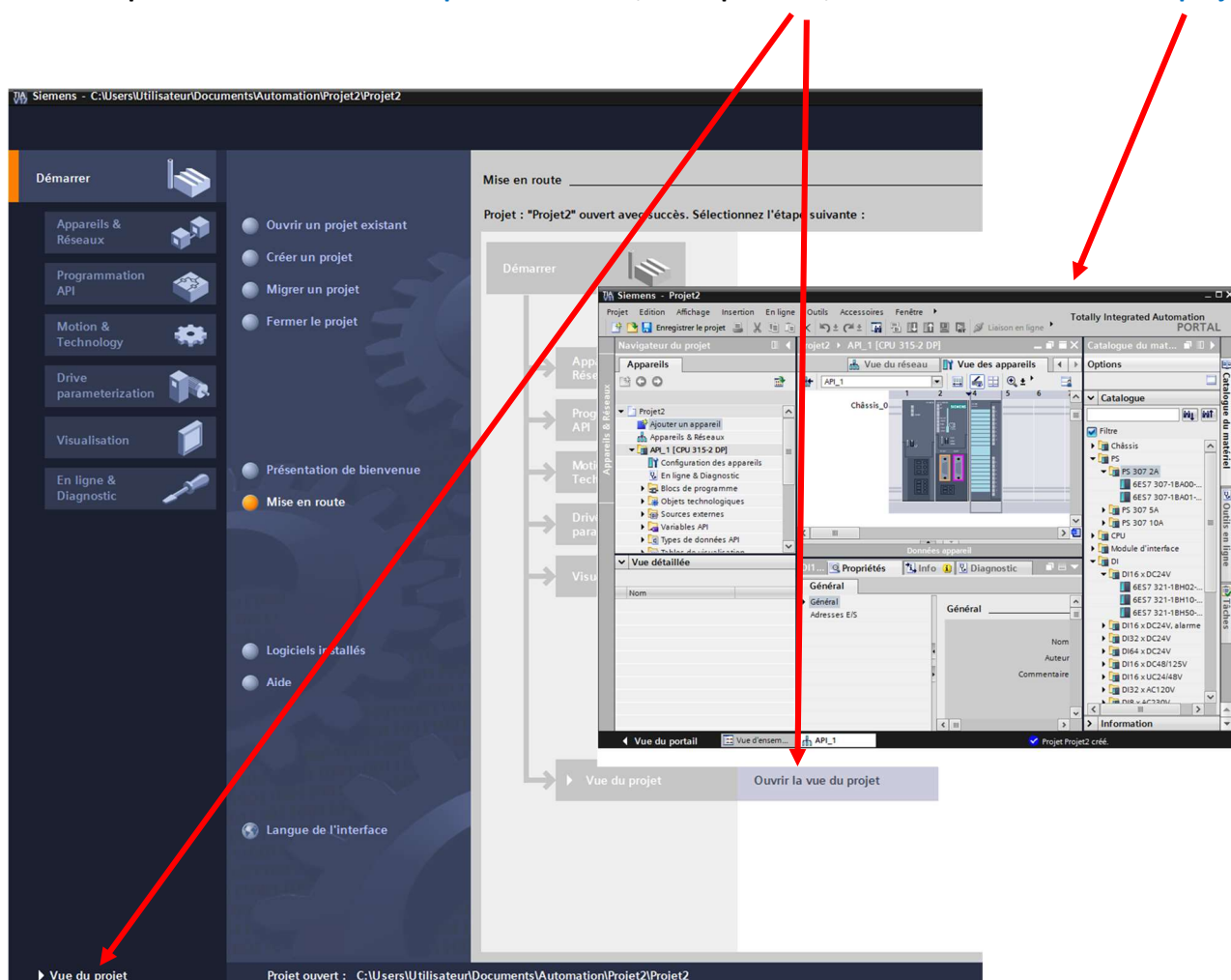
- **SIMATIC STEP 7 V17** permettant de programmer les automates programmables des gammes :
 - ✓ S7-1200
 - ✓ S7-1500
 - ✓ S7-300
- **SIMATIC WinCC V17** permettant de programmer les Interfaces Homme Machine

Lancement du logiciel

Cliquer sur l'icône



Une première fenêtre « **vue du portail** » s'ouvre, en cliquant sur, on ouvre la fenêtre « **vue du projet** »

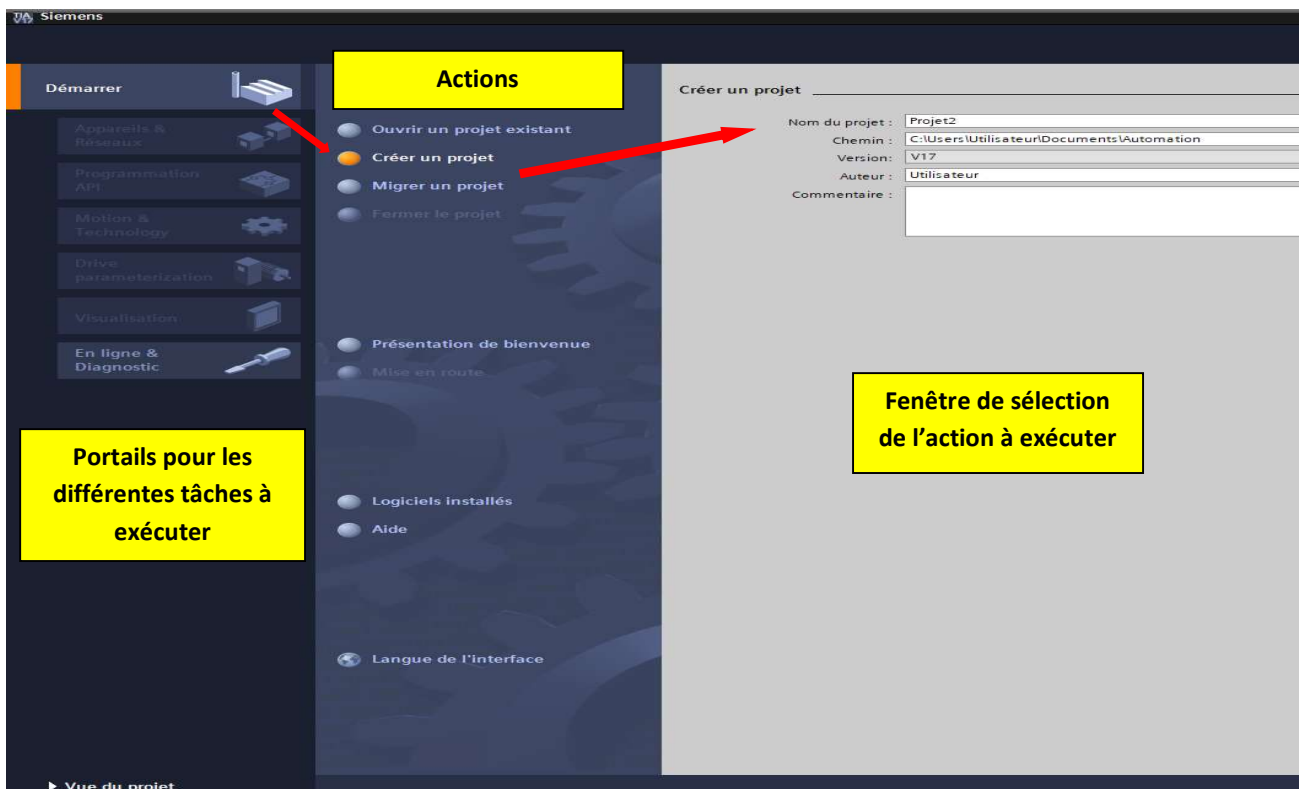


Présentation de l'environnement TIA PORTAL

❑ Vue du portail

Elle permet :

- Une approche axée sur les tâches à exécuter
- Une prise en main rapide par un guidage de l'utilisateur



Présentation de la vue du portail

Chaque portail permet de traiter une catégorie de tâches

La fenêtre affiche la liste des actions pertinentes pour la tâche sélectionnée

Une fenêtre de sélection propose l'ensemble des actions à exécuter

Actions

Les actions pouvant être exécutées dans le portail sélectionné sont affichées. Le portail est doté d'une aide contextuelle.

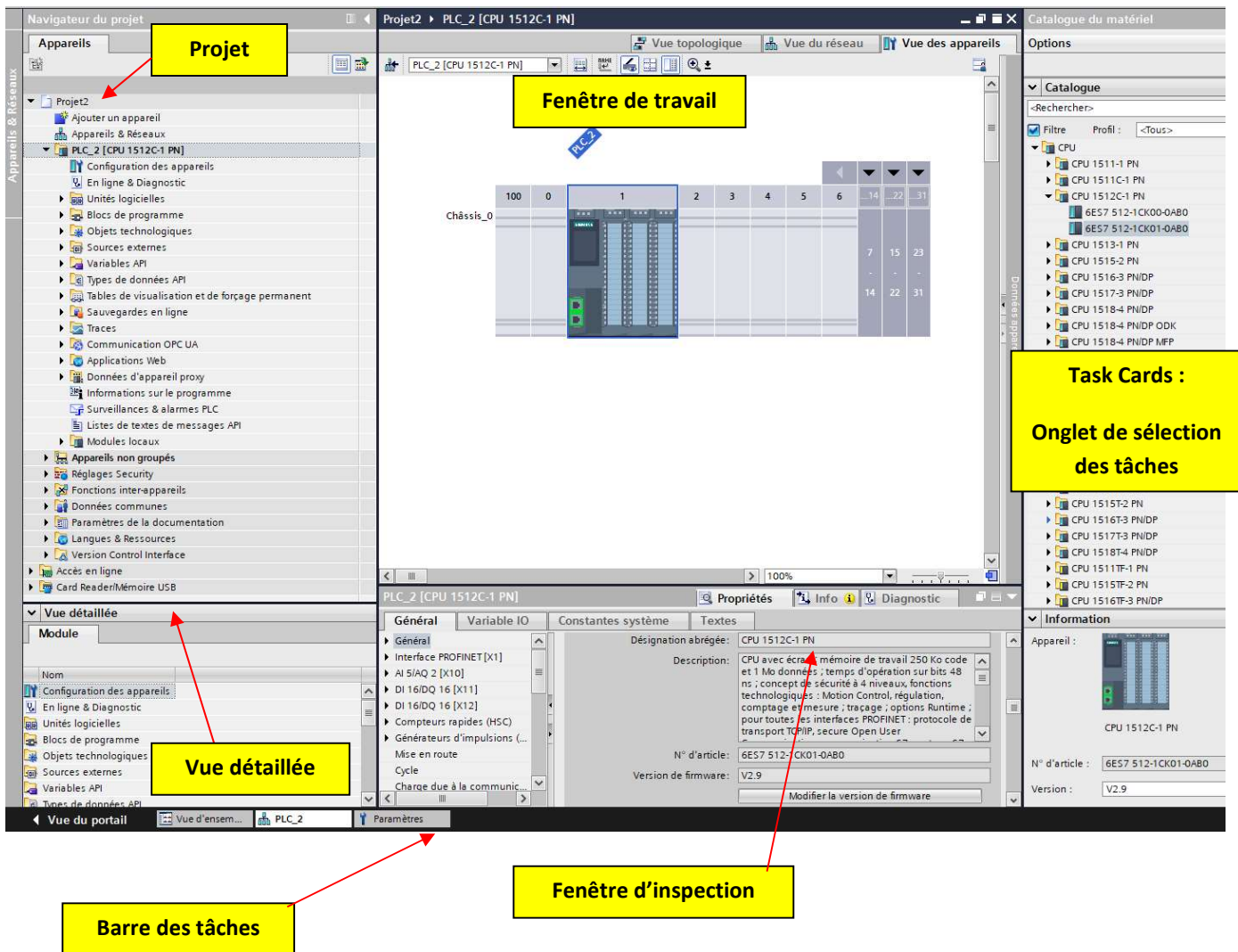
Fenêtre de sélection

Elle est disponible sur chacun des portails. Le contenu de la fenêtre est fonction de sélection courante.

❑ Vue du projet

Elle permet :

- Partage hiérarchique du projet
- Les éditeurs requis s'ouvrent en fonction des tâches à réaliser
- Les éditeurs, les paramètres et les données sont visualisés dans une seule et même vue



Projet

Il contient l'ensemble des éléments et des données requis pour mettre en œuvre la solution d'automatisation souhaitée. Dans la vue du projet, on peut ajouter de nouveaux éléments ou traiter / interroger / éditer les éléments existants.

Fenêtre de travail

Elle permet de visualiser les objets qui ont été ouverts pour être traités. Il peut s'agir de composants matériels, de blocs, de tables de variables pour les API ou encore de vues pour des pupitres opérateurs (HMI), ...

Fenêtre d'inspection

Elle affiche des informations complémentaires sur un objet sélectionné ou sur les actions en cours d'exécution.

Task Cards

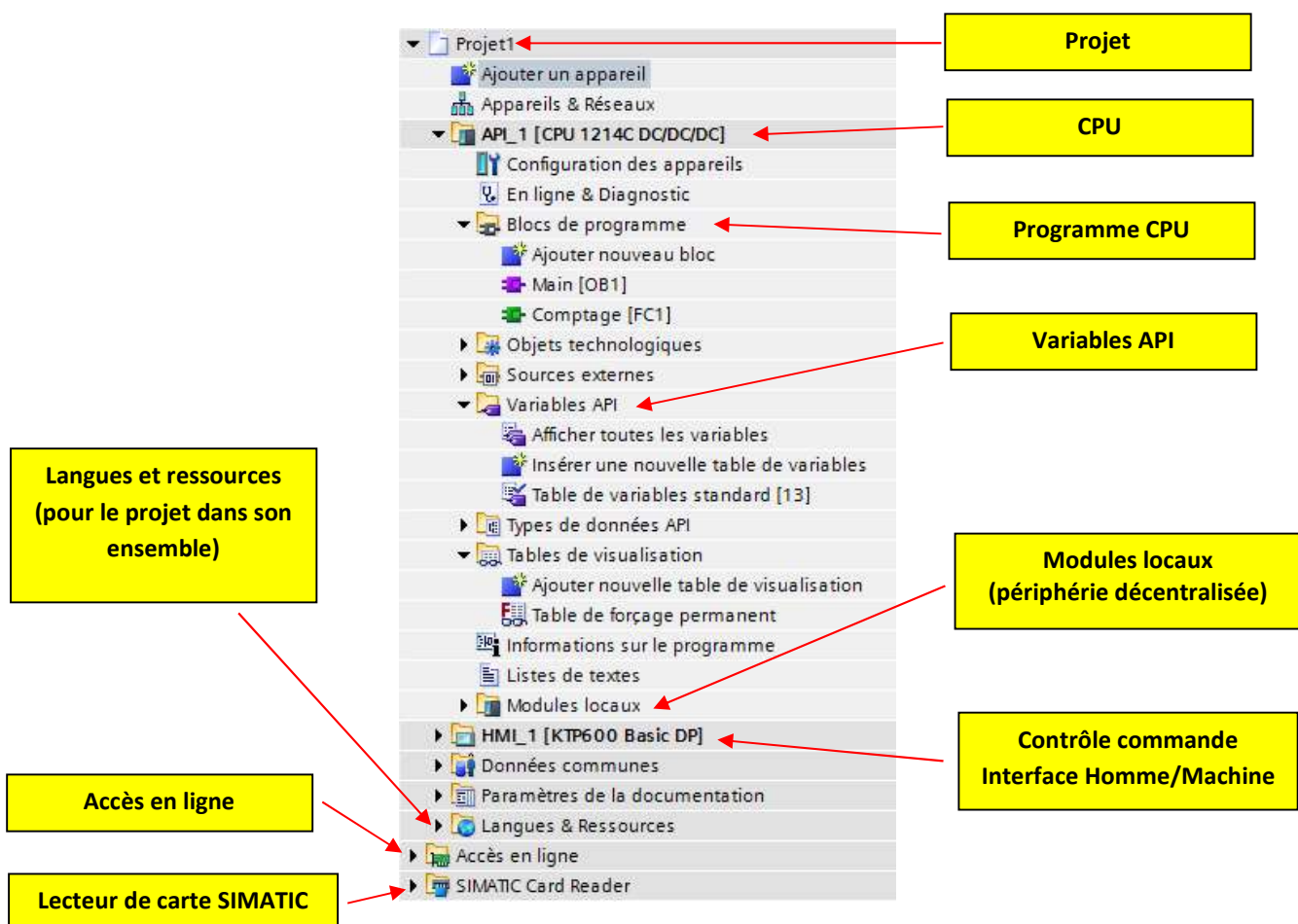
Le contenu des menus affichés dans la barre d'onglets est fonction de l'objet sélectionné dans la fenêtre de travail. S'il s'agit par exemple d'une station matérielle, le catalogue du matériel est disponible en onglet. S'il s'agit d'un bloc de programme, l'utilisateur pourra visualiser les instructions.

Remarque : Il est possible de partager la fenêtre de travail en utilisant différentes icônes.

Exemple : fractionner la fenêtre dans le sens horizontal ou dans le sens vertical

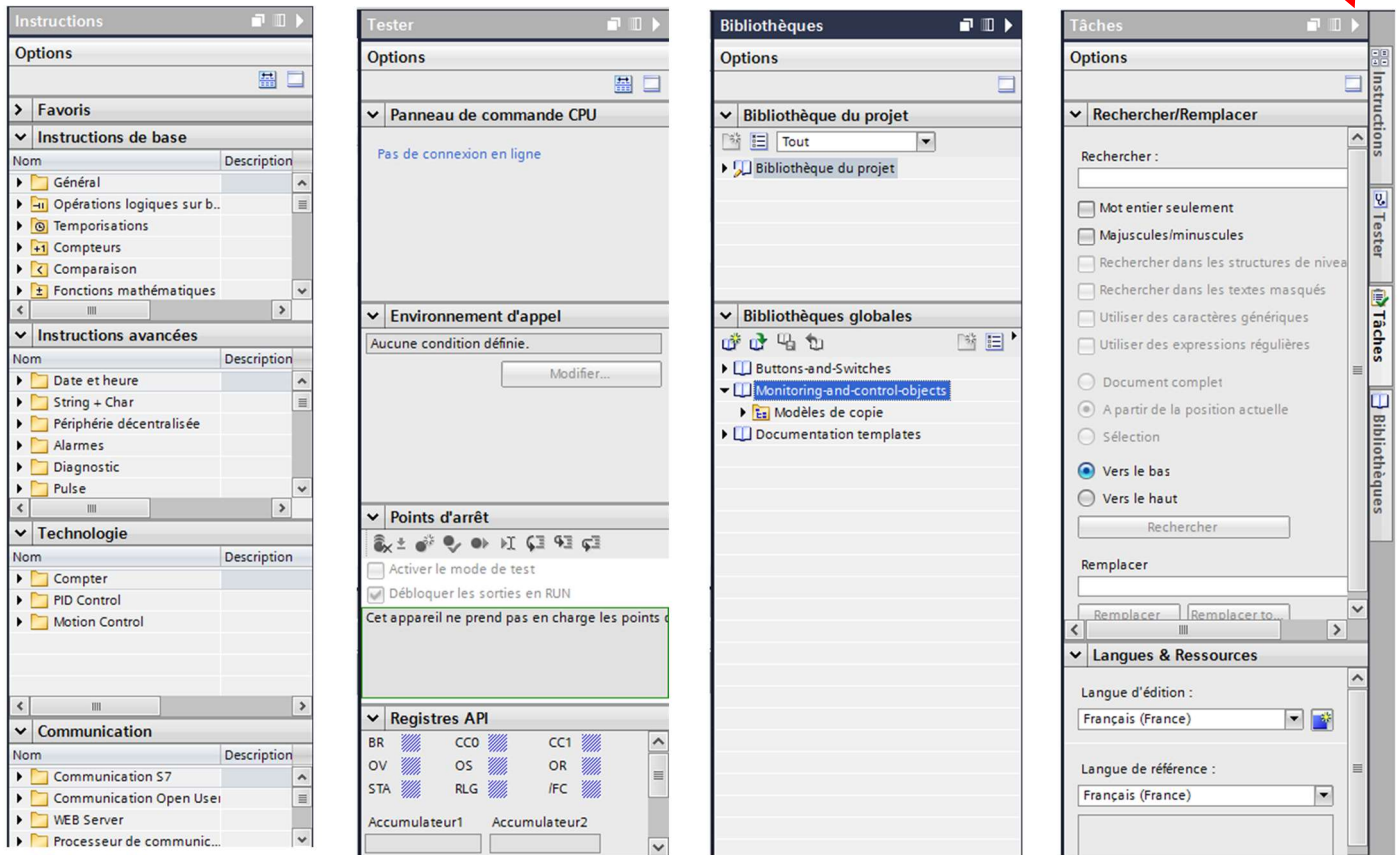


📁 Navigateur de projet



❑ Task Cards

Ouvrir / Fermer la barre d'onglets



Les onglets affichés sur la droite de l'écran peuvent être ouverts ou fermés à tout moment. Les onglets disponibles à l'écran sont fonction des produits installés et de l'objet ouvert dans la fenêtre de travail ou en cours de traitement. Ces onglets permettent de 'exécuter différentes actions :

- **Catalogue du matériel** lors de l'ajout d'un appareil ou **liste d'instructions** lors de l'écriture du programme.
- **Outils en ligne**
Lorsqu'une liaison en ligne est établie, des informations en ligne et des données de diagnostic peuvent être consultées (temps de cycle actuel de la CPU, ...). Par ailleurs, la CPU peut être basculée en mode STOP et RUN.
- **Tâches**
Cet onglet permet d'accéder aux fonctions classiques des différents éditeurs : chercher/remplacer des variables, des instructions, ...
- **Bibliothèques**
Pour enregistrer des objets susceptibles d'être réutilisés, il faut utiliser des bibliothèques locales (relatives au projet) et globales. Elles permettent de mémoriser et de réutiliser (copier) des objets de tout type (stations complètes, blocs, listes de variables, ...).

❑ Fenêtre d'inspection

La fenêtre d'inspection permet d'afficher des informations supplémentaires sur un objet sélectionné ou sur une action en cours d'exécution. Elle comprend les éléments suivants :

● Propriétés

Cet onglet permet de visualiser les propriétés de l'objet sélectionné et de modifier les propriétés éditables.

● Info

Cet onglet permet d'afficher les informations complémentaires sur l'objet sélectionné. Par ailleurs, des messages sur les actions en cours d'exécution sont affichés (par ex. sur la compilation ou le chargement de blocs dans la CPU).

● Diagnostic

Cet onglet permet d'afficher des informations sur le diagnostic système et les événements d'alarmes configurés.

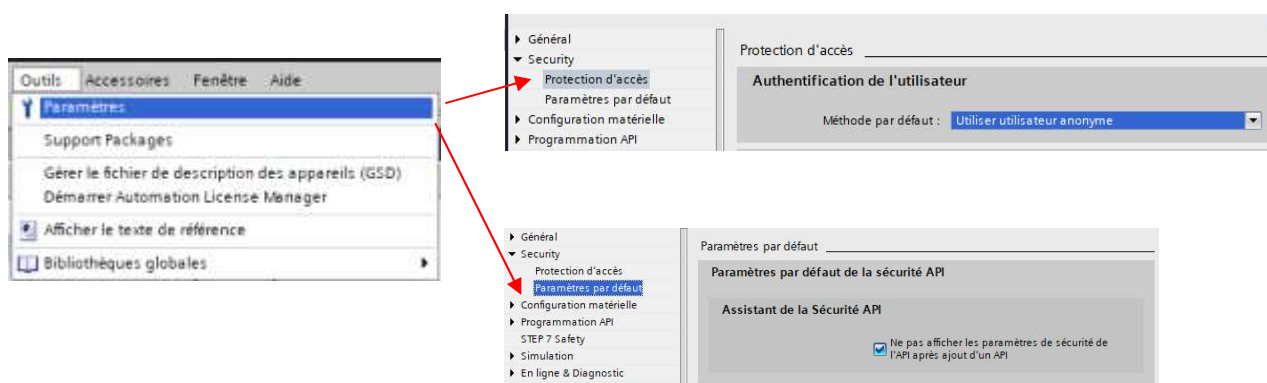
Création d'un projet TIA PORTAL

Un projet contient l'ensemble des programmes et des données nécessaires pour effectuer une tâche d'automatisation. Ce projet peut contenir un ou plusieurs programmes utilisés par une ou plusieurs CPU. Après avoir créé un nouveau projet suivre la démarche décrite ci-dessous.

❑ Sélection des paramètres de sécurité

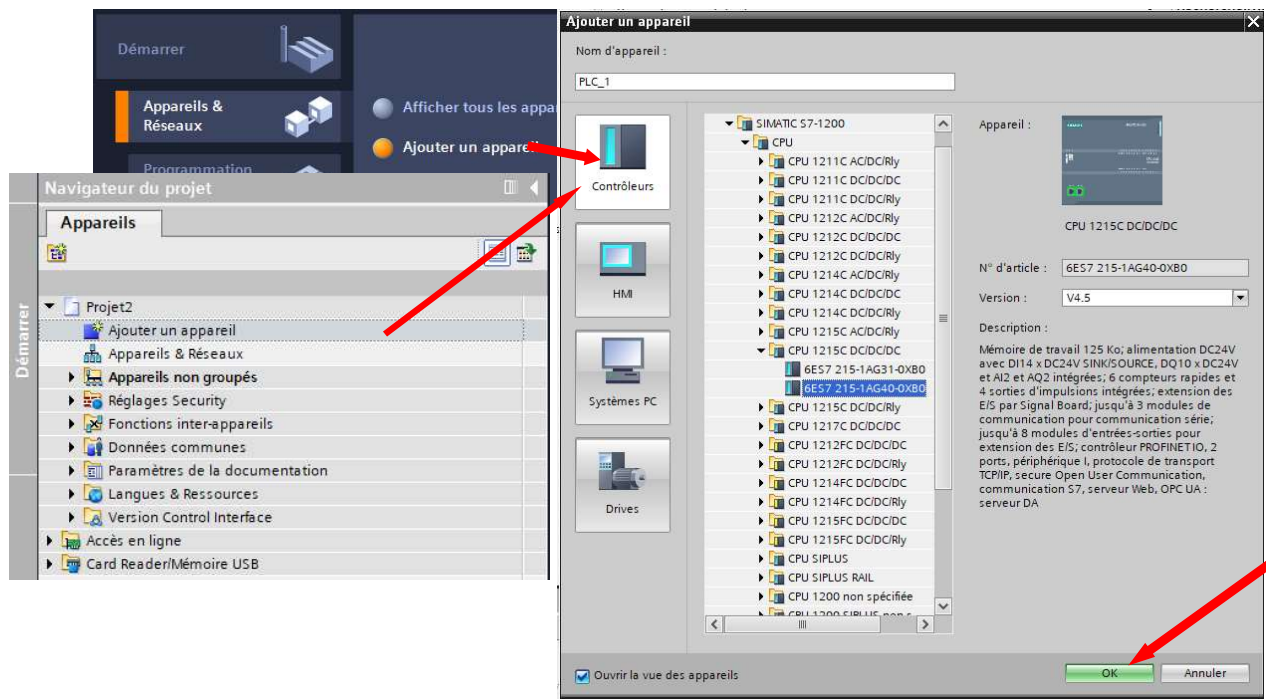
Avec les nouvelles normes de sécurité, il est possible de mettre des mots de passe pour accéder à la configuration et à la programmation des automates. Ceci n'est pas utile dans notre cas.

Modifier les paramètres de sécurité selon les éléments ci-dessous. **A effectuer une seule fois ; à l'ouverture du logiciel, il reprendra le paramétrage effectué.**

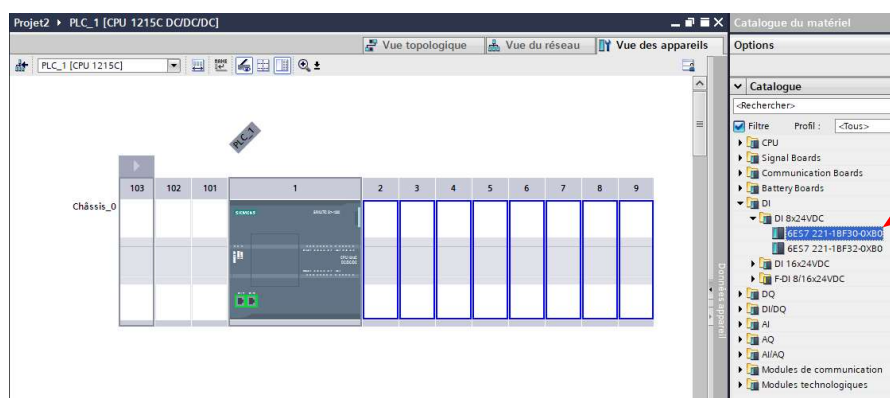


❑ Créer une station matérielle

Lors de la création d'un nouvel appareil (SIMATIC S7-1200, SIMATIC S7-1500), un châssis adapté est automatiquement créé. L'appareil sélectionné est enfiché au premier emplacement autorisé du châssis. Indépendamment du chemin choisi, l'appareil ajouté est visible dans la vue des appareils et la vue du réseau de l'éditeur d'appareils et de réseaux.



Avec la configuration matérielle à votre disposition, positionner l'ensemble des différents modules dans le châssis (rack). Dans la Task Cards, ouvrir l'onglet [catalogue de matériel](#).



Remarque : Il est possible d'insérer, de supprimer et de remplacer un module.

Insérer

Un nouveau module peut être enfiché par simple glisser-déposer, même entre des modules déjà enfichés. Faire glisser pour ce faire le nouveau module entre les modules déjà enfichés ; une marque d'insertion apparaît et le nouveau module peut être inséré. Tous les modules sont décalés d'un emplacement vers la droite.

Supprimer

Il est possible de supprimer des appareils complets ou des stations entières dans la vue du réseau, et des composants individuels d'une station dans la vue des appareils. Les composants matériels supprimés sont effacés du système et les adresses occupées sont libérées.

Remplacer

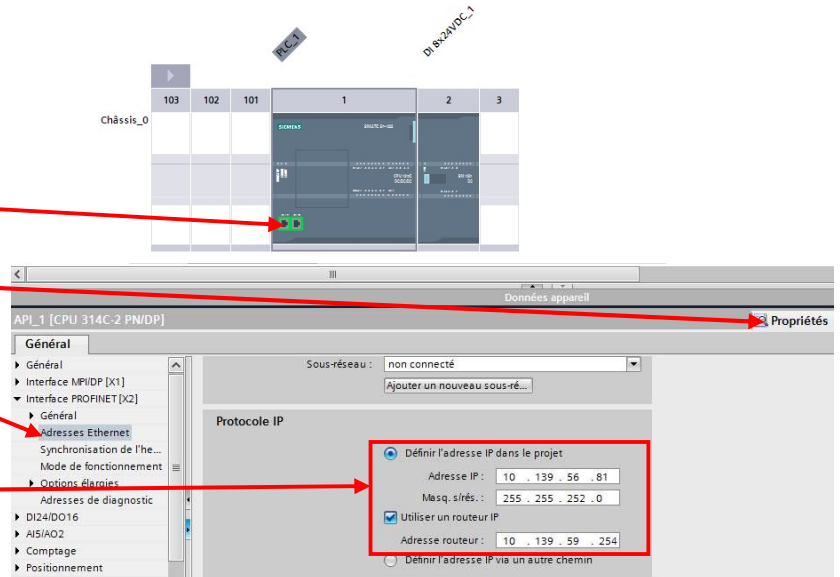
L'ensemble des paramètres de l'ancien module sont récupérés par le nouveau module. Un remplacement de module peut être nécessaire par ex. lorsqu'on doit adapter la configuration prévue (projet hors ligne) à une configuration réelle (CPU, en ligne) après une mise à jour du firmware.

❑ Configurer la station matérielle

Adresses Ethernet de l'API

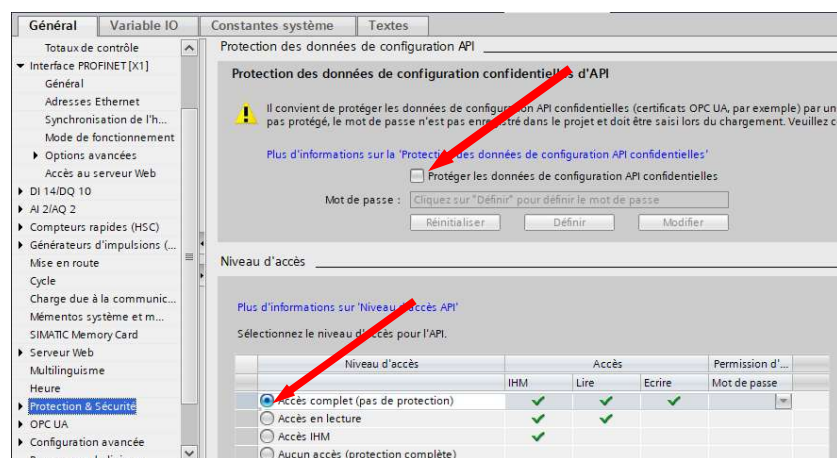
Il est nécessaire d'indiquer dans la fenêtre Protocole IP l'adresse Ethernet de l'API

- Cliquer sur la CPU
- Cliquer sur Propriétés
- Cliquer sur adresses Ethernet
- Indiquer l'adresse de l'API sans oublier le routeur (si celui-ci est utilisé)



Paramétrage protection et sécurité

Il est déconseillé de mettre des mots de passe pour configurer l'API et pour accéder à l'API. Paramétrer Protection et sécurité suivant figure ci-dessous.

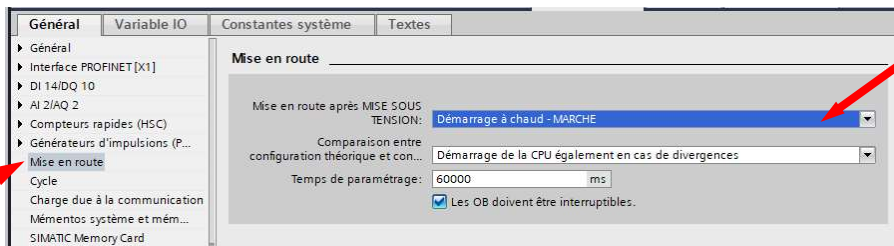


Paramétrage mise en route de la CPU

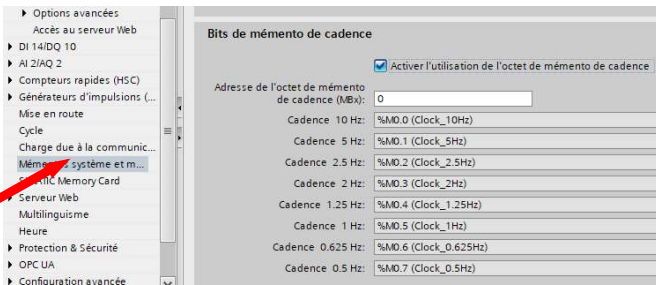
La CPU peut passer à l'état ARRET suite à des défauts réparables, tels que la défaillance d'un module d'entrées-sorties remplaçable, ou à des défauts temporaires, tels qu'une perturbation sur la ligne d'alimentation ou un événement erratique à la mise sous tension.

Si la CPU a été configurée à "Démarrage à chaud - mode de fonctionnement avant la mise hors tension", elle ne repassera pas à l'état MARCHE une fois le défaut réparé ou éliminé tant qu'elle n'aura pas reçu de STEP 7 une nouvelle commande de passage à l'état MARCHE. En l'absence de nouvelle commande, l'état ARRET sera conservé comme état de fonctionnement avant la mise hors tension.

Typiquement, les CPU destinées à fonctionner indépendamment d'une liaison STEP 7 doivent être configurées à "Démarrage à chaud - MARCHE" afin qu'elles puissent repasser à l'état MARCHE lors d'une mise hors tension puis sous tension suivant l'élimination de défauts.



Paramétrage Mémento de cadence



Dans la fenêtre d'inspection, il suffit de cocher la case **Mémento de cadence** puis d'indiquer l'adresse d'un octet (dans l'exemple %M0).

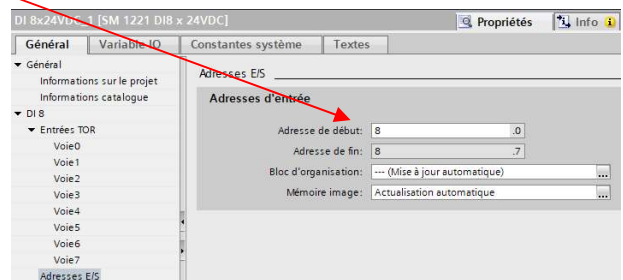
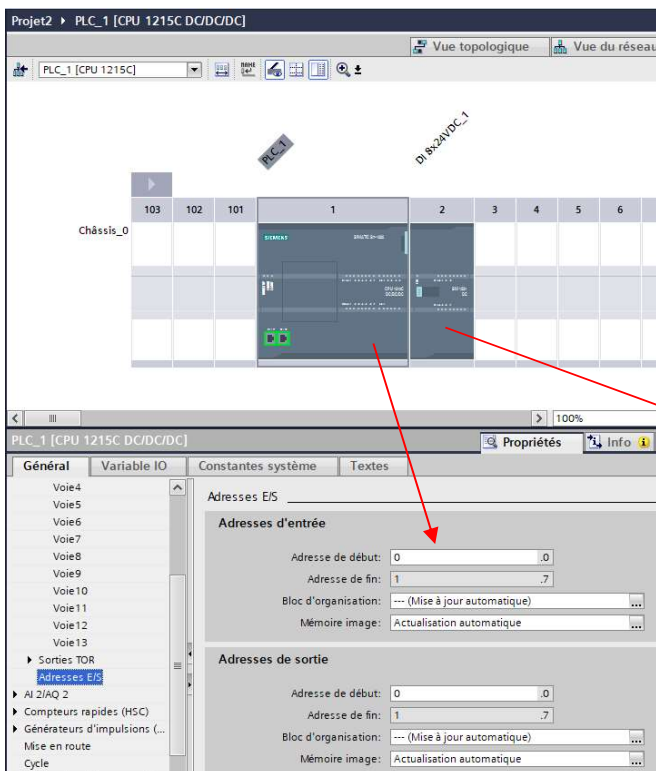
Les mémentos de cadence s'utilisent pour déclencher des opérations périodiques ou pour faire clignoter par ex. un feu de signalisation. Ce sont des mémentos dont la valeur binaire est modifiée périodiquement par le système d'exploitation de la CPU.

Attention de ne pas utiliser le même memento pour une autre variable du programme.

Configuration des Entrées (%) et Sorties (%Q) dans la CPU et dans les modules

Lorsque vous ajoutez une CPU et des modules d'E/S dans votre écran de configuration, les adresses I et Q sont automatiquement affectées. Cet adressage est visible dans la fenêtre d'inspection lors de la configuration des appareils.

Il est possible de modifier l'adressage par défaut en sélectionnant le champ d'adresse dans l'écran de configuration et en saisissant de nouvelles valeurs.



Adresses bit et octet

Les API Siemens travaille en octal.

Afin de permettre l'adressage d'une entrée ou d'une sortie à l'intérieur d'un octet, chaque octet est divisé en huit **bits**. Ces derniers sont numérotés de 0 à 7. On obtient ainsi l'**adresse du bit**.

Pour adresser la troisième entrée, il faut entrer l'adresse suivante

%I 0. 2

I désigne le type d'adresse entrée,
0 l'adresse de l'octet,
2 l'adresse du bit.

L'adresse de l'octet et l'adresse du bit sont toujours séparées par un point.

Pour adresser la dernière sortie d'un octet, il faut entrer l'adresse suivante

%Q 5. 7

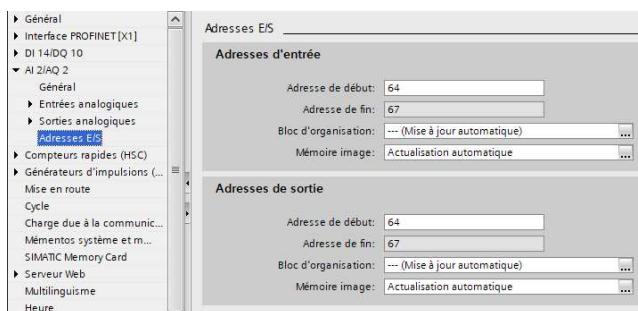
Q désigne le type d'adresse sortie,
5 l'adresse de l'octet,
7 l'adresse de bit.

L'adresse de l'octet et l'adresse du bit sont toujours séparées par un point.

Les entrées et sorties numériques sont attribuées dans des groupes de 8 points (1 octet), que le module utilise tous les points ou non.

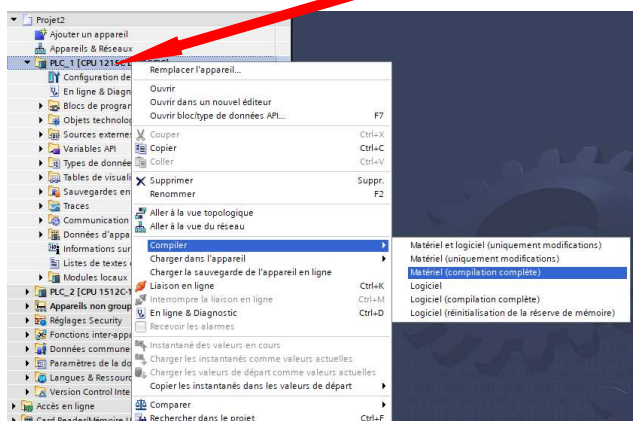
Les entrées et sorties analogiques sont affectées par groupes de deux entrées ou deux sorties.

Chaque entrée ou sortie est codée sur 2 octets (octet 64 et octet 65 ou octet 66 et octet 67)



❑ Compiler la configuration matérielle

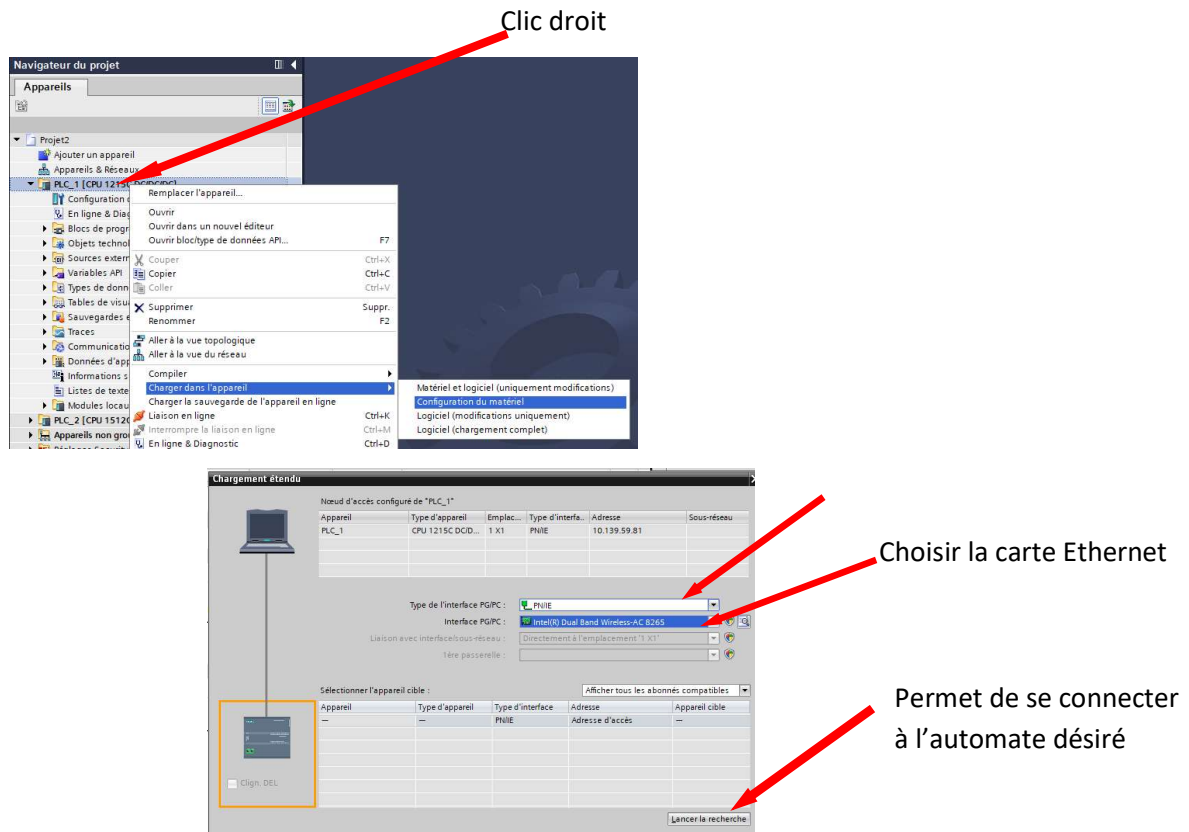
Clic droit



Après compilation, vérifier dans la fenêtre d'inspection si la compilation a été réalisée sans erreur. En cas d'erreurs, vérifier le type d'erreur dans la fenêtre d'inspection et corriger. Il est impossible de charger une configuration, avec des erreurs, dans l'automate.

❑ Charger la configuration matérielle

Les données matérielles du projet sont générées lors de la configuration de la station matérielle, des réseaux et des liaisons. Au premier chargement du projet, l'ensemble des données matérielles du projet sont chargées. Lors des opérations de chargement ultérieures, seules les modifications opérées dans la configuration sont chargées.



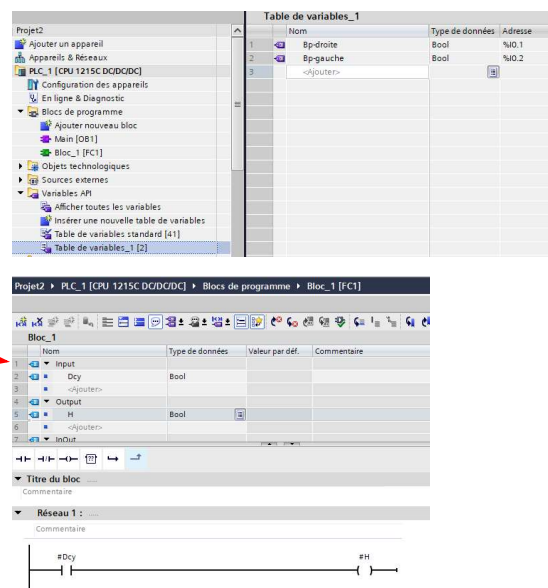
❑ Création d'un programme

Variables API

Il existe deux types de variables :

Variables globales : Elles sont déclarées dans la table de variables et ont la propriété de pouvoir être employées dans tous les blocs de programme. Le nom doit être univoque c'est-à-dire qu'un nom symbolique ne doit pas être utilisé plusieurs fois.

Variables locales : Elles sont définies dans la partie déclarative d'un bloc. Elles ne peuvent être utilisées que dans ce même bloc. On peut donc réutiliser le même nom symbolique dans la partie déclarative d'un autre bloc.

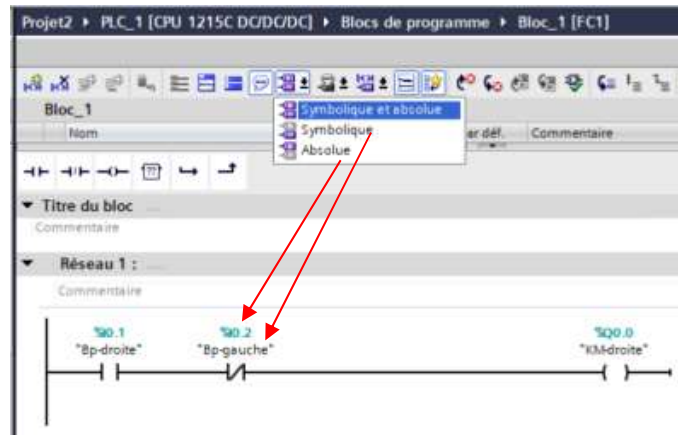


● Variables globales : adressage absolu et symbolique

Toutes les variables globales : les entrées (%I), les sorties (%Q), les mémentos (%M) possèdent une adresse absolue et une adresse symbolique. L'utilisateur peut déterminer l'adresse qu'il veut visualiser à l'écran et le type d'adressage qu'il souhaite utiliser pour la programmation.

Si une adresse symbolique (par ex. Bp-droite) est utilisée dans le programme sans qu'aucune adresse absolue ne lui ait été affectée, le bloc pourra être enregistré, mais la compilation et le chargement dans l'automate seront impossibles.

Si l'utilisateur programme directement avec une adresse absolue (par ex. %I0.0), une adresse symbolique lui sera automatiquement affectée (par ex. Tag_1), qui pourra être modifiée ultérieurement.



● Table de variables API

La table de variables API, générée en même temps que la création de la CPU, contient la déclaration des variables et des constantes utilisées (par exemple le memento de cadence ci-dessous)

Les variables sont des opérandes utilisés dans le programme dont le contenu peut varier, les constantes sont des opérandes dont le contenu ne varie pas.

Lors de la création d'une variable ou d'une constante dans une table de variables API, il faut définir le nom symbolique (par ex. Bp_droite), le type de donnée (par ex. Bool) et l'adresse absolue (par ex. %I0.0). Pour les constantes, il n'est pas nécessaire de définir une adresse absolue.

	Nom	Table des variables	Type de données	Adresse
1	Clock_Byte	Table de variables s...	Byte	%M0.0
2	Clock_10Hz	Table de variables s...	Bool	%M0.1
3	Clock_5Hz	Table de variables s...	Bool	%M0.2
4	Clock_2.5Hz	Table de variables s...	Bool	%M0.3
5	Clock_1Hz	Table de variables s...	Bool	%M0.4
6	Clock_0.625Hz	Table de variables s...	Bool	%M0.5
7	Clock_0.5Hz	Table de variables s...	Bool	%M0.6
8	Tag_1	Table de variables s...	Bool	%M0.7
9	Bp-droite	Table de variables_1	Bool	%I0.1
10	Bp-gauche	Table de variables_1	Bool	%I0.2
11	KM-droite	Table de variables s...	Bool	%Q0.0
12	<Ajouter>			

Remarques

- M (mémentos) : Utilisez la zone des mémentos (mémoire M) pour les données afin de stocker l'état intermédiaire d'une opération ou d'autres informations de commande.
- Lors de l'adressage de mémentos, attention de ne pas faire chevaucher les variables. Si vous utiliser le mot %MW10 pour une variable, ne pas prendre %MB11 pour une autre variable ; le mot %MW10 est constitué des octets %MB10 et %MB11.
- Il est possible de copier & coller de et vers Excel.
- En cliquant sur le caractère de remplissage dans le coin inférieur droit de la cellule, puis en glissant vers le bas, on peut créer automatiquement des variables (comme sur Excel)
- La mémoire rémanente est une mémoire vive non volatile (RAM) permettant de stocker en mémoire tampon des mémentos, des temporisations, des compteurs et des blocs de données, même en l'absence d'une pile de sauvegarde. Les zones devant faire l'objet d'une sauvegarde en mémoire tampon doivent être définies.

Différents types de bloc du programme

Selon les exigences du processus à automatiser, le programme peut être structuré en différents blocs, de blocs de données, de commentaires et de mnémoniques liés à une application. C'est ce qu'on appelle la programmation structurée. Elle permet de décomposer une tâche en plusieurs tâches partielles qui sont ainsi plus faciles à résoudre et à tester dans leur fonctionnalité.

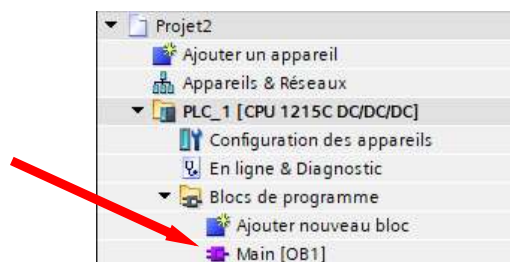
Blocs d'organisation (OB)

Ils constituent l'interface entre le système d'exploitation et le programme utilisateur. Les blocs d'organisation permettent de structurer votre programme. Une fois le traitement des OB de cycle de programme achevé, la CPU réexécute immédiatement les OB de cycle de programme. Ce traitement cyclique est le type "normal" de traitement utilisé par les automates programmables. Il existe plusieurs types de blocs OB. Deux exemples :

- **Bloc OB1** qui va appeler les différents blocs du programme et qui est lu à chaque cycle automate.
- **Bloc OB START/UP** qui va initialiser des valeurs à la mise en RUN de l'automate et qui est lu une seule fois à la mise en RUN.

Remarque

Un bloc d'organisation OB1 vide est automatiquement créé dans le programme utilisateur après avoir effectué la configuration de l'appareil.



Blocs fonctions (FC)

Ils assurent une fonctionnalité spécifique dans une séquence de programme. Les blocs fonctions peuvent être paramétrables. Dans ce cas, les paramètres requis sont transmis à la fonction lorsqu'elle est appelée. Les blocs fonctions conviennent donc pour la programmation de fonctionnalités récurrentes et complexes, par exemple pour des opérations de calcul.

Blocs fonctionnels (FB)

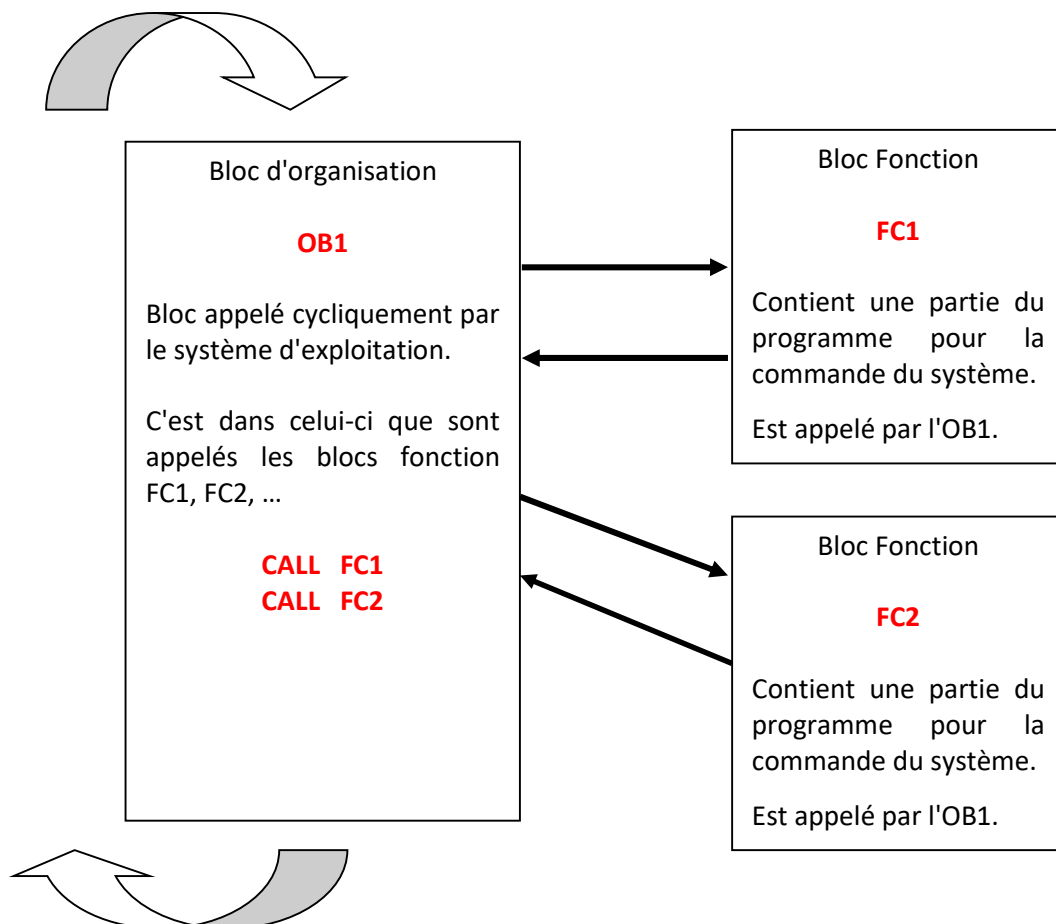
Ils s'apparentent à des fonctions qui disposent en plus de zones mémoire spécifiques sous la forme de blocs de données d'instance. Ces blocs conviennent donc pour la programmation de fonctionnalités récurrentes encore plus complexes, par exemple pour des opérations de régulation ou de programmation en SFC (programmation grafcet).

Blocs de données (DB)

Ils sont utilisés dans le programme utilisateur pour sauvegarder les données des blocs de code. Tous les blocs de programme dans le programme utilisateur peuvent accéder aux données d'un DB global.

Les données sauvegardées dans un DB ne sont pas effacées à la fin de l'exécution du bloc de code associé. Les OB, FB et FC peuvent accéder aux données dans un DB global.

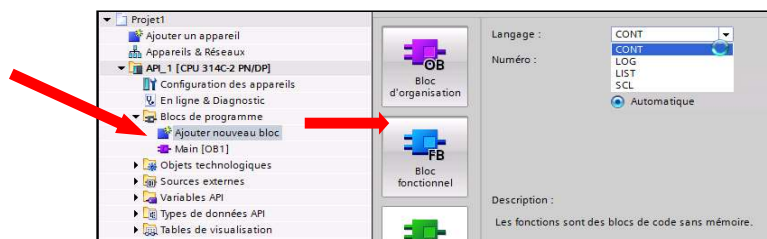
Structure du programme



Ajouter un nouveau bloc

Pour ajouter un nouveau bloc, procédez comme ci-dessous. Lors de la création du nouveau bloc, il faut déterminer le type de bloc désiré, le langage de programmation, le nom symbolique et

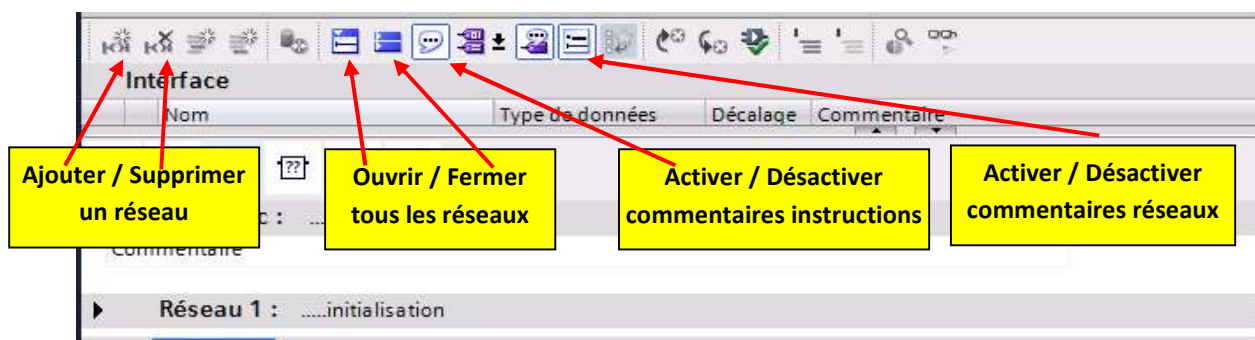
le numéro. Le numéro du bloc peut être attribué manuellement ou automatiquement. La fenêtre des « informations complémentaires » permet de documenter plus en détail le bloc.



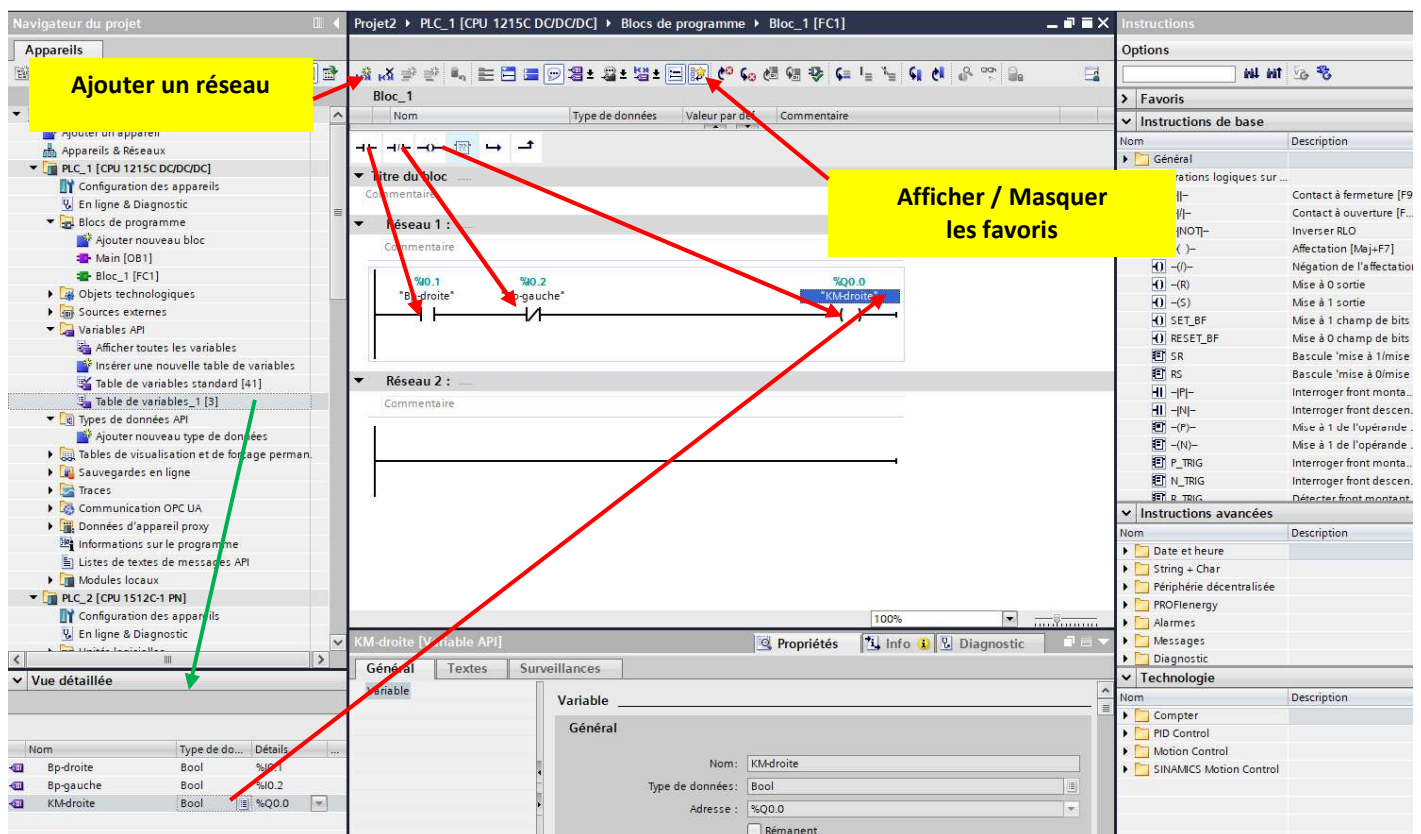
Programmation d'un bloc

Chaque bloc peut comprendre différents segments appelés réseaux. La subdivision d'un bloc en réseaux est déterminée par l'utilisateur.

L'utilisateur peut attribuer à chaque réseau un titre et un commentaire.



Après avoir inséré un réseau, écrire le programme en utilisant les instructions développées ci-dessous.



Les instructions au sein d'un bloc peuvent être programmées comme suit :

- Par glisser & déposer dans le programme à partir de la barre des favoris ou à partir du catalogue des instructions (à droite de l'écran)
- En sélectionnant tout d'abord l'emplacement souhaité dans le programme et en double cliquant ensuite sur l'instruction souhaitée dans la barre des favoris ou dans le catalogue des instructions (à droite de l'écran)

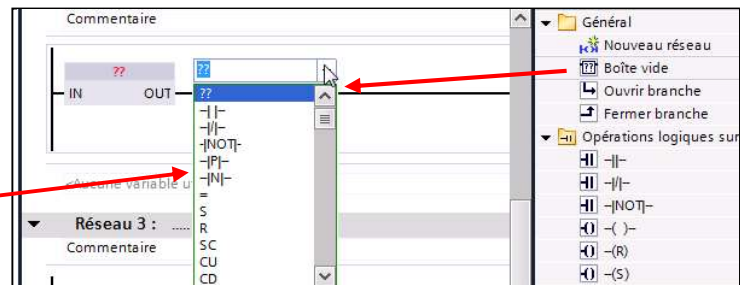
Les opérandes peuvent être saisis avec les adresses absolues ou les noms symboliques. **Si la table des variables est sélectionnée (pas ouverte) dans le navigateur de projet, les variables peuvent également être insérées dans le programme à l'emplacement souhaité par glisser & déposer depuis la vue détaillée.**

Barre des favoris

Les éléments CONT fréquemment utilisés sont disponibles dans la barre des favoris qui peut être étendue individuellement par glisser & déposer à partir du catalogue.

Remarque

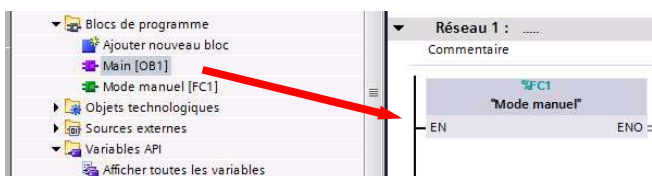
Une instruction peut également être programmée via une boîte vide. Il suffit d'insérer dans le réseau une boîte vide par glisser & déposer à partir de l'onglet « instructions de base » puis sélectionner dans la boîte vide l'instruction et le type de données que vous souhaitez programmer.



Programmation du bloc OB1 afin d'appeler le bloc FC1

- Ouvrir le bloc organisation **OB1** qui appellera le bloc FC1 dans le programme.
- A l'aide de la fonction glisser & déposer prendre le bloc FC1 du navigateur de projet et le placer sur le réseau de l'OB1

Langage CONT



Langage SCL



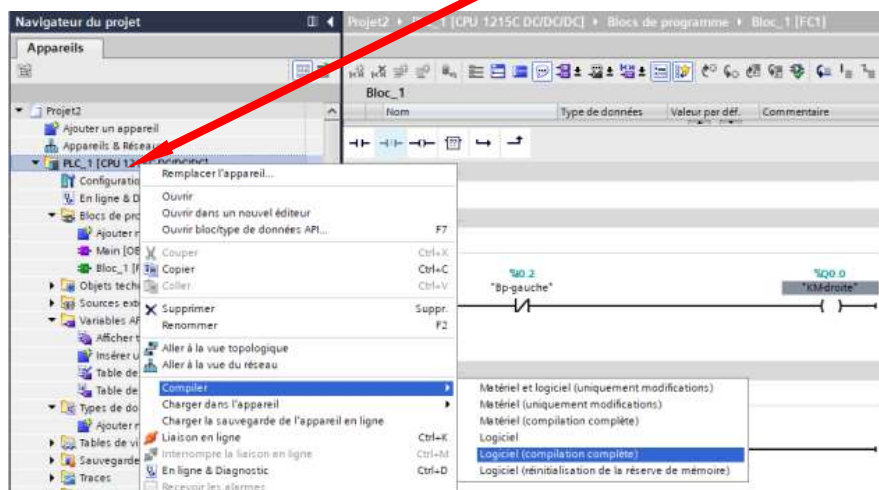
❑ Compiler les blocs programme

Les données logicielles du projet comprennent les blocs du programme. L'utilisateur peut opter pour une compilation complète des données logicielles ou pour une compilation des seules modifications logicielles :

- **Logiciel (compilation complète des blocs)** : tous les blocs du programme utilisateur sont compilés

 **Logiciel** uniquement les blocs modifiés sont compilés.

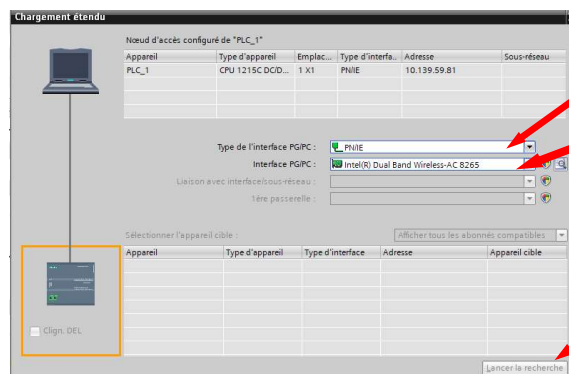
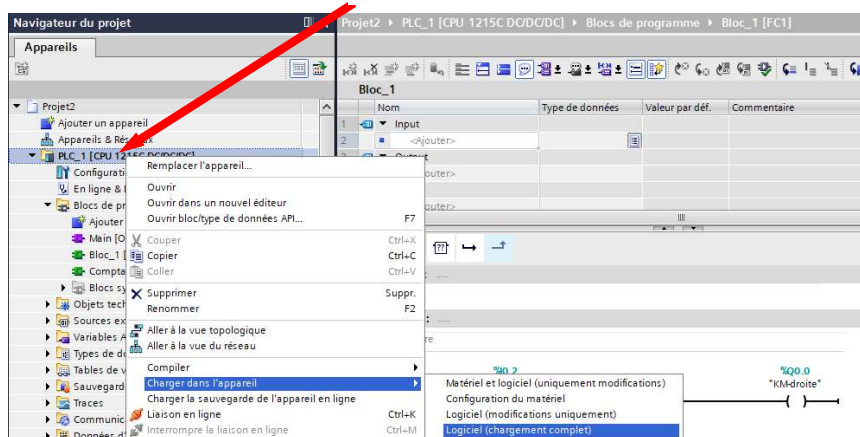
Clic droit



❑ Charger les blocs dans la CPU

Les données logicielles du projet comprennent les blocs du programme. Au premier chargement du projet, l'ensemble des données logicielles du projet sont chargées. Lors des opérations de chargement ultérieures, l'utilisateur peut opter pour un chargement complet des données logicielles ou pour un chargement des seules modifications logicielles :

Clic droit



Choisir la carte Ethernet

Permet de se connecter à l'automate désiré

❑ Visualiser les blocs

La fonction de test « Visualiser un bloc » permet de suivre l'exécution du programme au sein d'un bloc. Les blocs ne peuvent être visualisés que si une liaison en ligne a été établie avec la CPU. Par ailleurs, le bloc hors ligne doit être identique au bloc en ligne.

Vérifier que l'automate est bien mode RUN

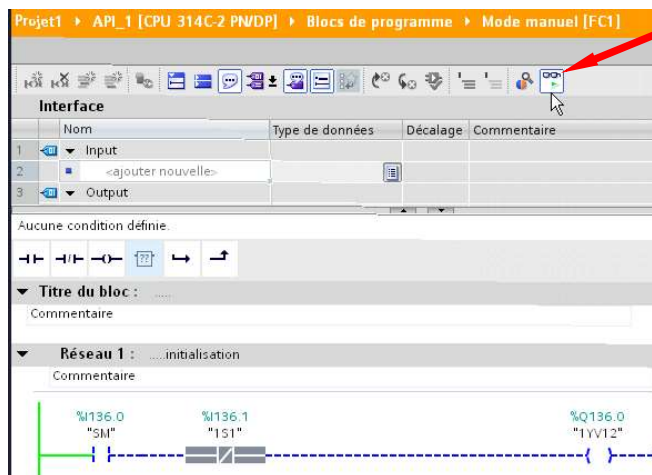


En mode test, l'état des opérandes et les éléments CONT / LOG sont représentés avec des codes couleur.

- Condition remplie → l'élément est représenté en vert
- Condition non remplie → l'élément est représenté en bleu

Exemple de visualisation en langage CONT

Pour visualiser, cliquer sur



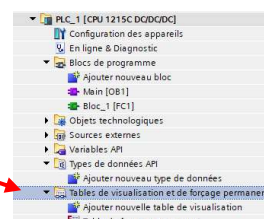
Remarque

Si le réseau reste d'une couleur grise : soit le réseau n'est pas appelé par l'OB1 soit l'API n'est pas en mode RUN.

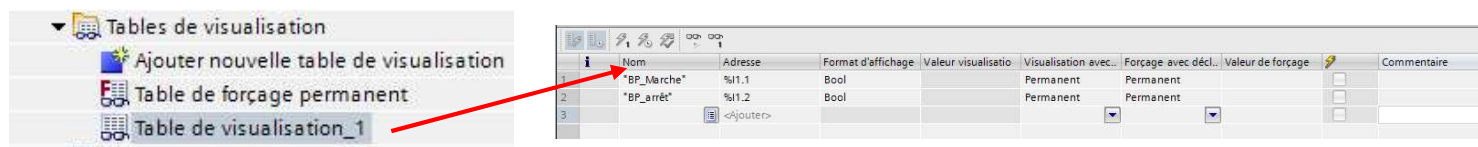
❑ Visualiser / forcer des variables : table de visualisation

La fonction de test « Visualiser et forcer une variable » permet de visualiser et / ou de forcer des variables dans un format quelconque. Les variables souhaitées doivent être déclarées dans une table de visualisation. L'ensemble des variables et opérandes, à l'exception des variables locales et temporaires, peuvent être visualisées et / ou forcées.

Pour créer une table de visualisation cliquer sur



On obtient la table ci-dessous. Il est possible de renommer la table (clic droit et renommer)



En choisissant les points et les conditions de déclenchement appropriées avec les icônes, il est possible de réaliser les tests suivants :

Forcer toutes les valeurs actives de manière unique et immédiate

Forcer toutes les valeurs actives par « Forçage avec condition de déclenchement »



✓ **Test de câblage des entrées : Visualiser les variables**

Point de déclenchement : **début de cycle** ; condition de déclenchement : **cyclique**

✓ **Simuler l'état des entrées** (affecter un état indépendamment du processus) : **Forcer des variables**

Point de déclenchement : **début de cycle** ; Condition de déclenchement : **cyclique**

✓ **Distinction erreur logicielle / défaut matériel** (un actionneur qui devrait être activé dans le processus n'est pas enclenché) : **Visualiser les variables** pour visualiser la sortie correspondante

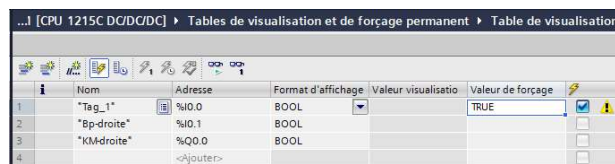
Point de déclenchement : **fin de cycle** ; Condition de déclenchement : **cyclique**
sortie = 1 : logique du programme ok ; défaut matériel
sortie = 0 : erreur logique du programme (double affectation par exemple)

✓ **Forcer des sorties** (indépendamment de la logique du programme) : **Forcer les variables**

Point de déclenchement : **fin de cycle** ; Condition de déclenchement : **cyclique**

Exemple : forçage de valeurs de manière unique et immédiate

- Saisissez les adresses et valeurs de forçage souhaitées dans la table de visualisation.
- Sélectionnez l'adresse à forcer en activant la case à cocher pour le forçage dans la colonne située après la "Valeur de forçage". Un triangle jaune apparaît derrière la case à cocher activée et indique que l'adresse est à présent sélectionnée pour le forçage, mais que celui-ci n'a pas encore été réalisé.
- Choisissez la commande "En ligne > Forcer > Forcer immédiatement" pour forcer les adresses sélectionnées de manière unique et immédiate avec les valeurs prédéfinies.

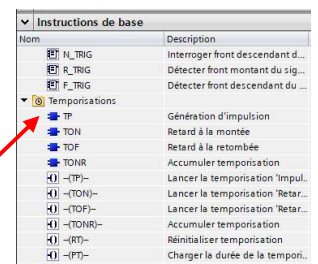
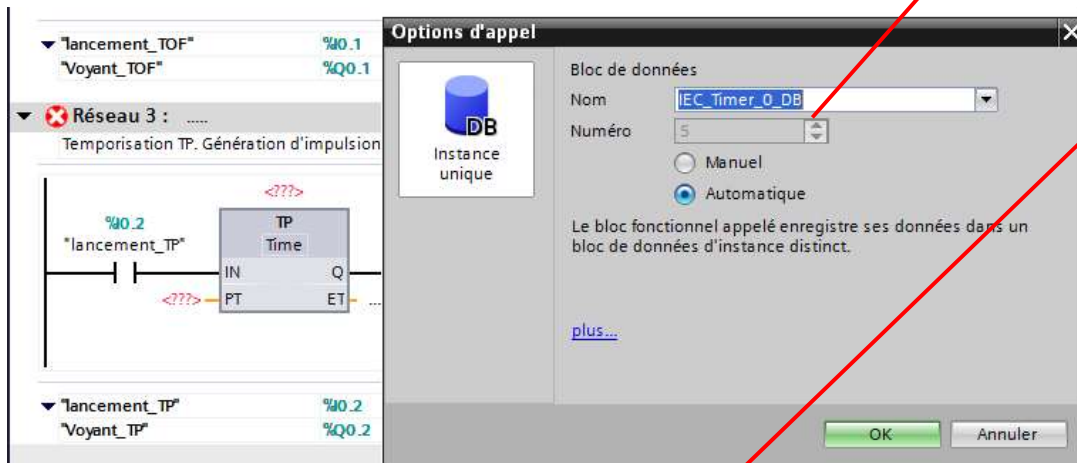
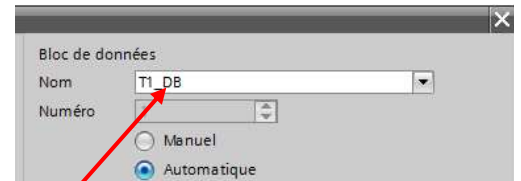


Programmation Temporisations

Le logiciel possède les 4 temporisations de la norme CEI

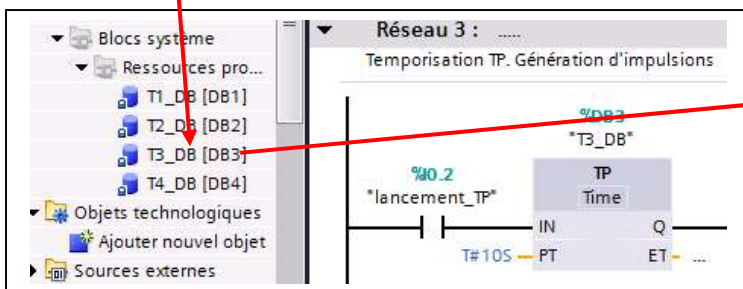
- ✓ TP (génération d'impulsions)
- ✓ TON (retard à la montée)
- ✓ TOF (retard à la remontée)
- ✓ TONR (accumuler temporisation)

□ Langage CONT



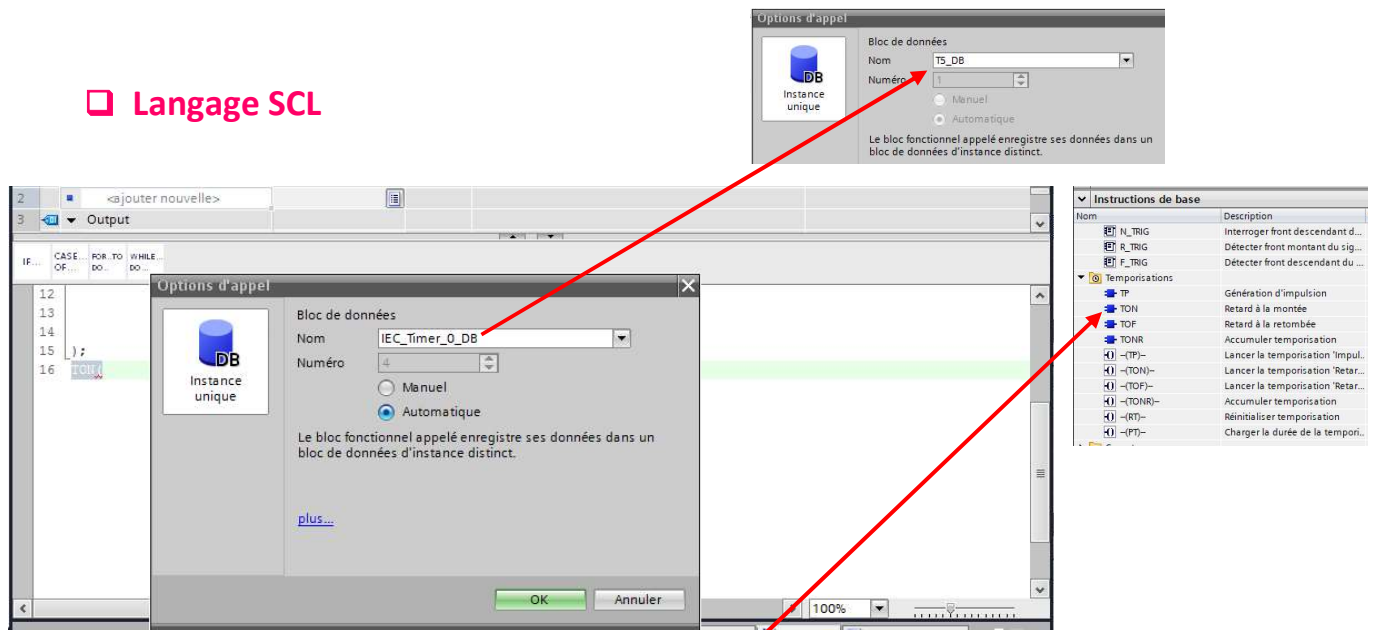
- Se positionner sur le réseau
- Choisir la temporisation désirée puis par glisser & déposer, la positionner sur le réseau
- Une fenêtre s'ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre temporisation (**bien laisser _DB à la fin du nom**)
- Indiquer devant l'entrée PT la valeur de la temporisation (par exemple : 5s ou 50ms)

Une fois la temporisation insérée, un bloc de données apparaît dans le navigateur sous « ressources programme ». Si on clique dessus, on a accès à l'ensemble des données de la temporisation.



T1_DB						
	Nom	Type de données	Décalage	Valeur de départ	Rémanence	Visible da...
1	Input					
2	IN	Bool	0.0	FALSE	☒	☒
3	PT	Time	2.0	T#0MS	☒	☒
4	Output					
5	Q	Bool	6.0	FALSE	☒	☒
6	ET	Time	8.0	T#0MS	☒	☒
7	InOut					
8	Static					
9	STATE	Byte	12.0	B#16#0	☒	☒
10	STIME	Time	14.0	T#0MS	☒	☒
11	ATIME	Time	18.0	T#0MS	☒	☒

□ Langage SCL



- Se placer sur une ligne
- Choisir la temporisation désirée puis par glisser & déposer, la positionner sur le réseau
- Une fenêtre s'ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre temporisation (**bien laisser _DB à la fin du nom**)

Dans la fenêtre de travail, on obtient les lignes ci-contre.

- Cliquer (sans ouvrir) sur table de variables standard afin qu'elle soit visible dans vue détaillée
- Remplacer par glisser & déposer les variables à la place des **_in_** de **IN ; Q ; ET** si nécessaire
- Remplacer la valeur de **_in_** de **PT** par la valeur de la temporisation
- Si certaines valeurs ne sont pas utiles les supprimer (par exemple : effacer **ET=>_out** ; ne pas oublier d'effacer également la virgule qui précède.

```

16 "T5_DB".TON(IN:=in,
17               PT:=in,
18               Q=>out,
19               ET=>out);
20

```

Après renseignement des paramètres

```

IF... CASE... FOR... WHILE... (*...*) REGION
OF... TO DO.. DO...

1 "T5_DB".TON(IN:="Lancement_TON_SCL",
2             PT:=T#5s,
3             Q=>"Voyant_TON_SCL");
4

```

Programmation Compteurs

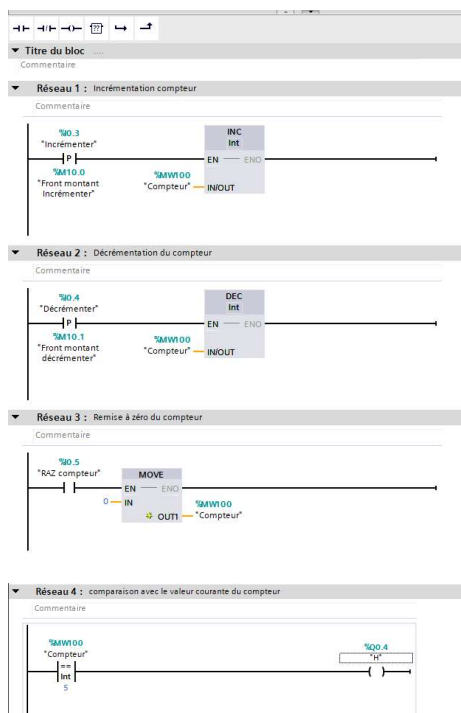
La fonction de comptage peut être réalisée de deux méthodes différentes :

- ✓ Incréméntation d'un mot (plus souple)
- ✓ Utilisation des blocs compteurs intégrés dans le logiciel

❑ Par incréméntation d'un mot en langage CONT

Nous avons besoin de variables TOR pour incrémenter, pour décrémenter et pour la remise à zéro du compteur ; d'une variable INT pour la valeur courante du compteur. L'incréméntation et la décréméntation se font sur un front montant de la variable sinon le compteur s'incréménte ou se décréménte tant que la variable est active.

Ci-dessous le programme avec les différents réseaux. On retrouve les différents symboles dans les menus de la Task kards

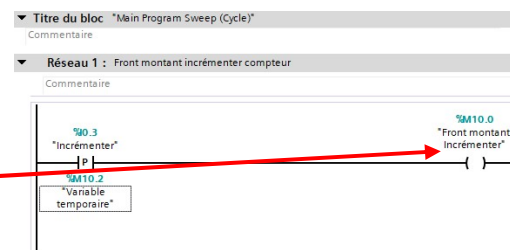


Instructions	
Options	
Favoris	
Instructions de base	
Nom	Description
HI -P-	Interroger front montant
HI -P-	Interroger front descendant
HI -P-	Mise à 1 de l'opérande si
HI -P-	Mise à 1 de l'opérande si
HI -P-	Interroger front montant
HI -P-	Interroger front descendant
HI -P-	Déclencher front montant di
HI -P-	Déclencher front descendant
Temponisations	
Compteurs	
Comparaison	
Fonctions mathématiques	
CALCULATE	Calculer
ADD	Addition
SUB	Soustraction
MUL	Multiplication
DIV	Division
MOD	Calculer le reste de la div
NEG	Créer le complément à 2
INC	Incréménter
DEC	Décréménter
Comparaison	
CMP ==	Egal à
CMP <>	Différent de
CMP >=	Supérieur ou égal à
CMP <=	Inférieur ou égal à
CMP >	Supérieur à
CMP <	Inférieur à
IN_Range	Valeur dans la plage
OUT_Range	Valeur en dehors de la pl
OK	Contrôler validité
NOT_OK	Contrôler invalidité

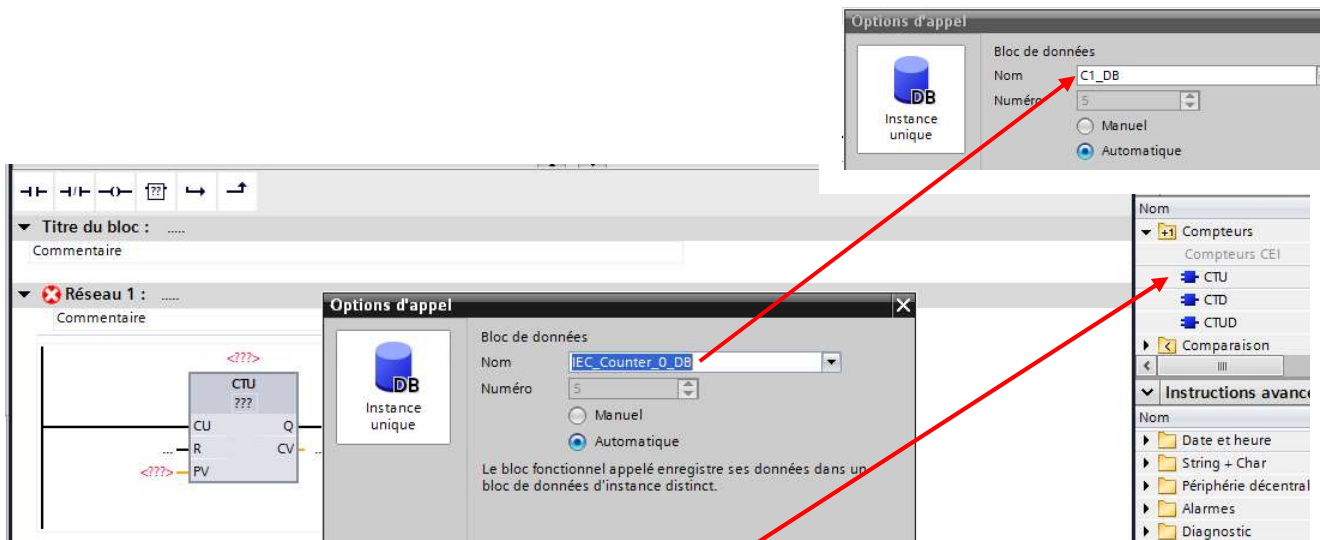
Transfert	
MOVE	Copier valeur
Deserialize	Désérialiser
Serialize	Sérialiser
MOVE_BLK	Copier zone
MOVE_BLK_VARIANT	Copier zone
UMOVE_BLK	Copier zone contiguë

Remarque fronts montants

Si le front montant d'une variable est présent à différents endroits du programme, il est impératif de choisir des bits « front montant » différents. Une autre solution est peut-être utilisée : écrire le front montant dans l'OB1 et l'associer à une variable ; ensuite, on peut utiliser cette variable dans différents blocs du programme.



❑ Par utilisation des blocs compteurs intégrés en langage CONT

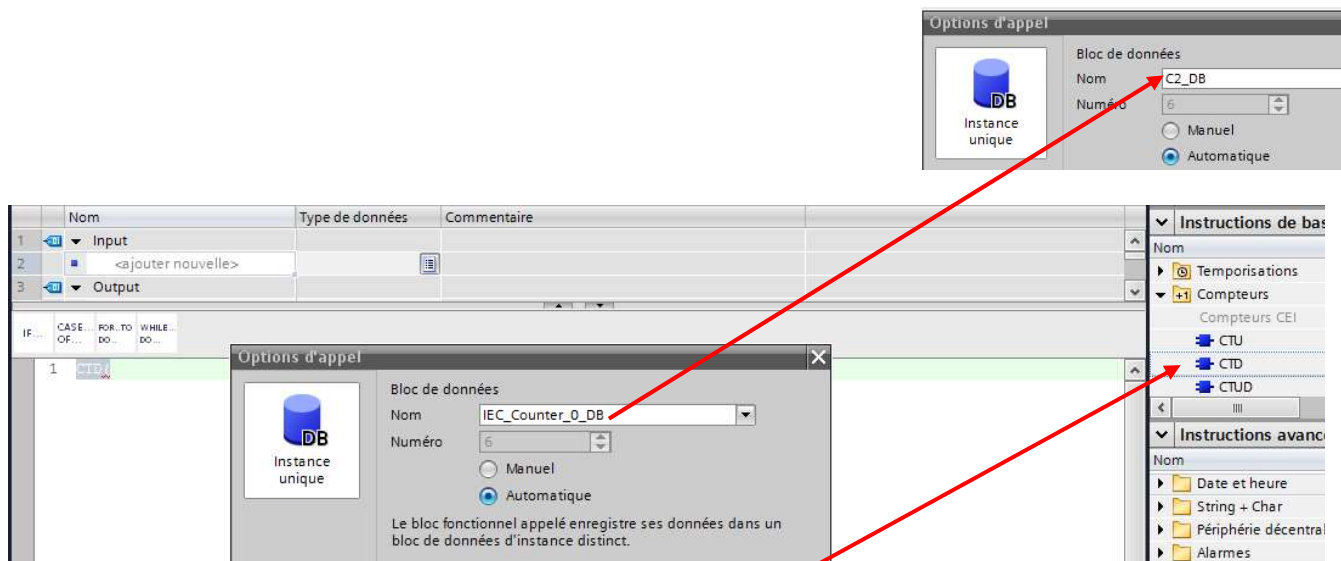


- Se positionner sur le réseau
- Choisir le compteur désiré puis par glisser & déposer, le positionner sur le réseau
- Une fenêtre s'ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre compteur (**bien laisser _DB à la fin du nom**)
- Indiquer devant l'entrée **PV** la valeur de la consigne à atteindre
- Indiquer devant l'entrée **R** l'entrée permettant une remise à zéro
- Indiquer devant l'entrée **CU** l'entrée permettant l'incrémement

Une fois le compteur inséré, un bloc de données apparaît dans le navigateur sous « ressources programme ». Si on clique dessus, on a accès à l'ensemble des données de la temporisation.

	Nom	Type de données	Valeur de départ	Rémanence
1	Static			
2	CU	Bool	false	☒
3	CD	Bool	false	☒
4	R	Bool	false	☒
5	LD	Bool	false	☒
6	QU	Bool	false	☒
7	QD	Bool	false	☒
8	PV	Int	0	☒
9	CV	Int	0	☒

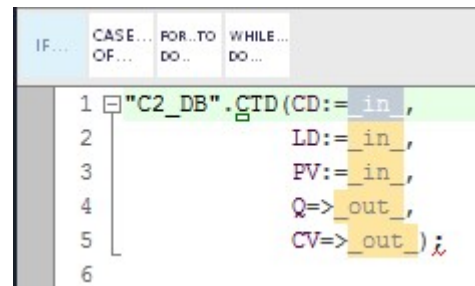
❑ Par utilisation des blocs compteurs intégrés en langage SCL



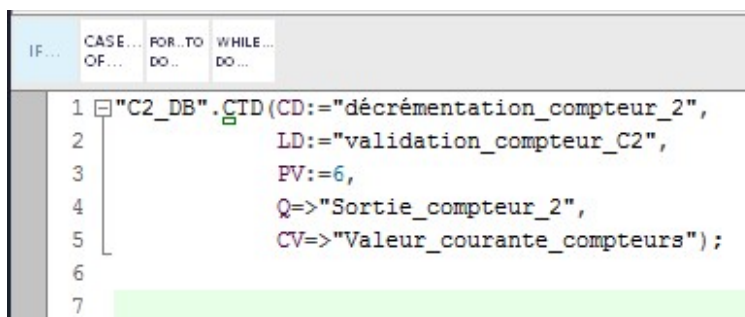
- Se placer sur une ligne
- Choisir le compteur désiré puis par glisser & déposer, le positionner sur le réseau
- Une fenêtre s'ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre compteur (**bien laisser _DB à la fin du nom**)

Dans la fenêtre de travail, on obtient les lignes ci-contre.

- Cliquer (sans ouvrir) sur table de variables standard afin qu'elle soit visible dans vue détaillée
- Remplacer par glisser & déposer les variables à la place des **_in_** de **CD** ; **LD** ; **Q** ; **CV** si nécessaire
- Remplacer la valeur de **_in_** de **PT** par la valeur de la temporisation



Après renseignement des paramètres



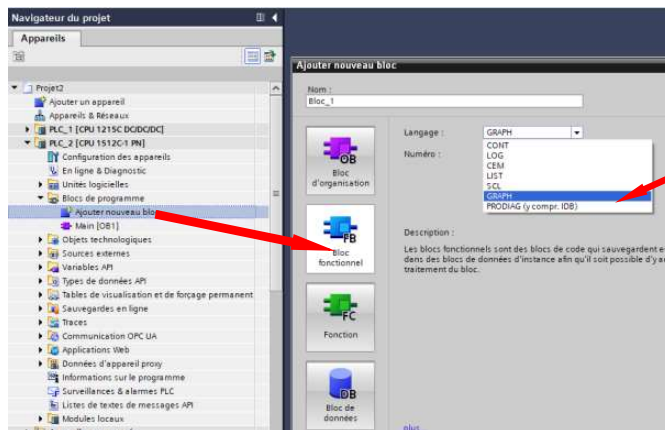
Programmation langage GRAPH

Le langage GRAPH permet de programmer directement des systèmes séquentiels représentés sous forme de grafset. Ils sont écrits dans un bloc fonctionnel de **type FB**.

Un bloc fonctionnel (FB) est une fonction ou une séquence de fonctions dans un bloc logique auquel est affectée une mémoire permettant de stocker des variables. Un FB a besoin de cette mémoire supplémentaire sous forme de « bloc de données d'instance » appelé DB.

Rappel : Le module GRAPH est disponible uniquement sur les automates S7-1500, impossible de faire du GRAPH sur les automates S7-1200.

Ajout d'un bloc GRAPH



Vue d'ensemble de l'interface de programmation

Vue graphe

Vue étape unique

Glisser & Déplacer les éléments

Icône	Nom
	Etape et transition
	Etape
	Transition
	Fin de graphe
	Saut
	Ouvrir une branche OU
	Ouvrir branche ET
	Fermer branche

● Exemple de programmation

Soit le grafcet ci-dessous à programmer en langage GRAPH

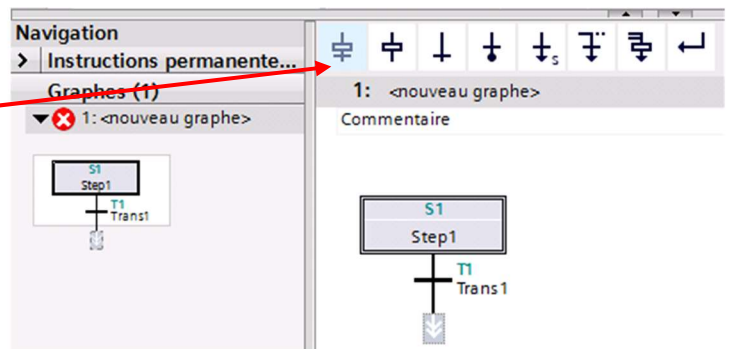
Grafcet point de vue PC

Grafcet point de vue API

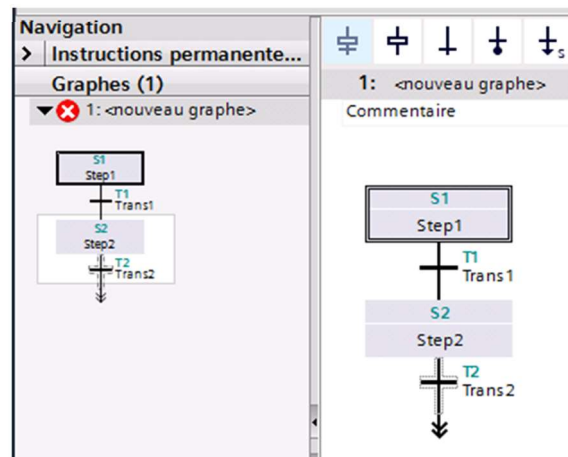
Pour la programmation du compteur C1, prendre une variable API nommée %MW100 et de type INT. Incrémenter cette variable à l'étape STEP2 ; mise à zéro de cette variable à l'étape STEP4 et tester la valeur de cette variable à l'aide d'un bloc COMPARAISON

➤ Ecrire la structure du graphe

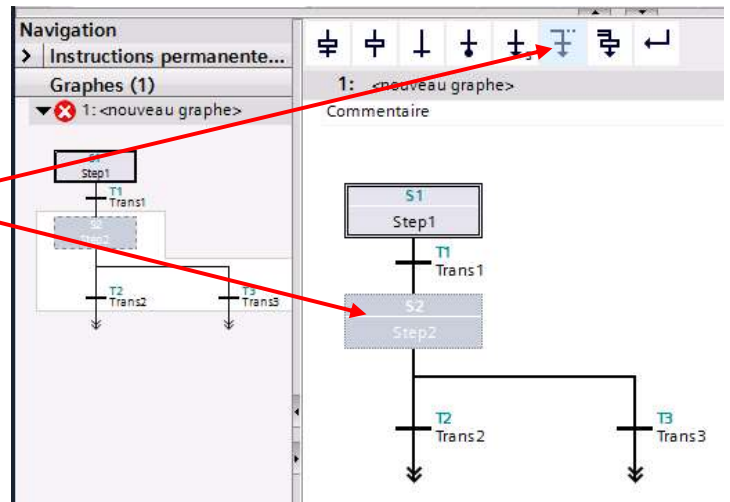
- Cliquer sur la transition T1
- Insérer une étape + transition



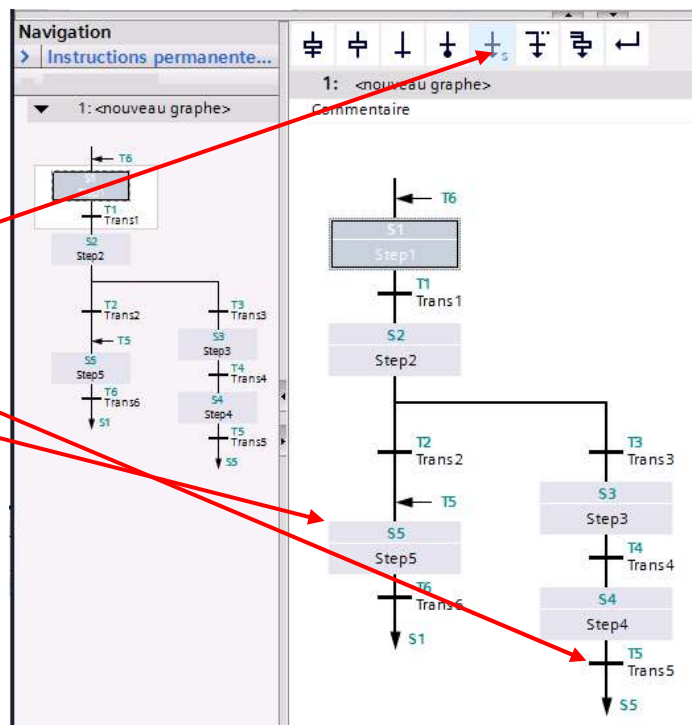
La deuxième étape et la deuxième transition sont insérées



- Cliquer sur **l'étape 2**
- Insérer une **branche OU**

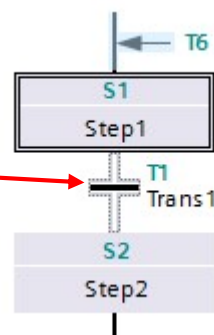


- Continuer à écrire la structure du grafcet.
- Pour « boucler » le grafcet cliquer sur la **transition T5**.
- **Insérer un saut.**
- Positionner le pointeur de la souris sur **l'étape destinataire S5** puis valider.
- Faire de même pour la liaison Transition T6 Etape S1

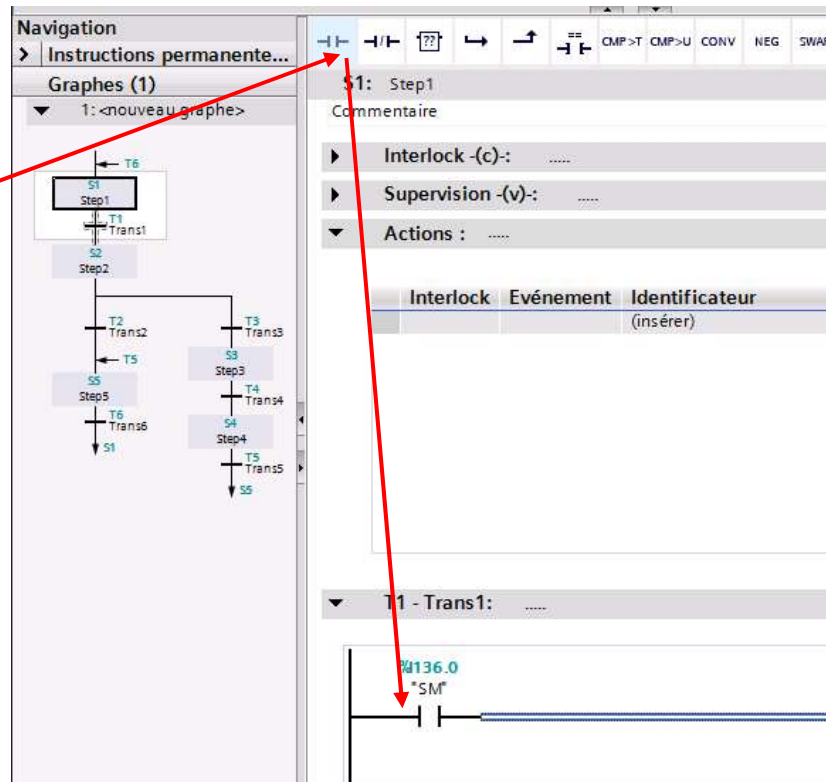


➤ Ecrire les réceptivités associées aux transitions

- Cliquer 2 fois sur **la transition**.

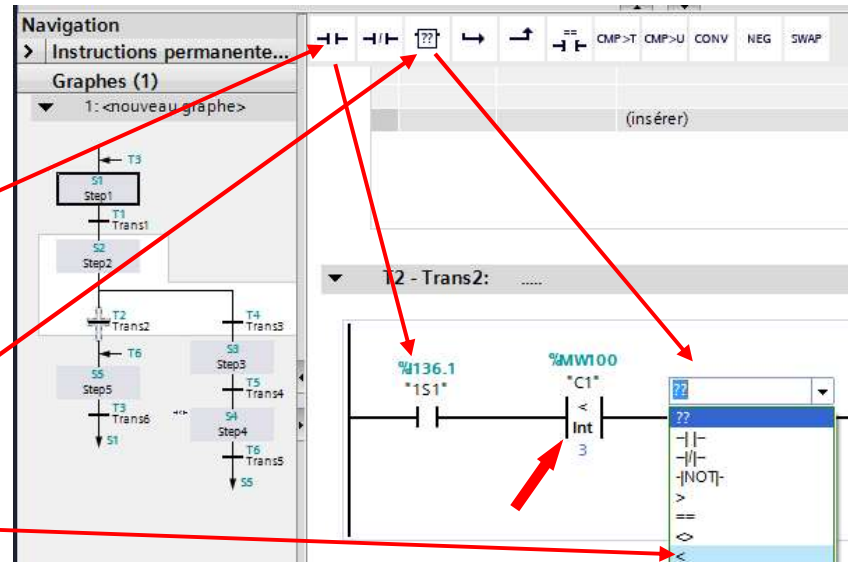


- Insérer **un contact** par glisser&déplacer et lui donner une adresse.



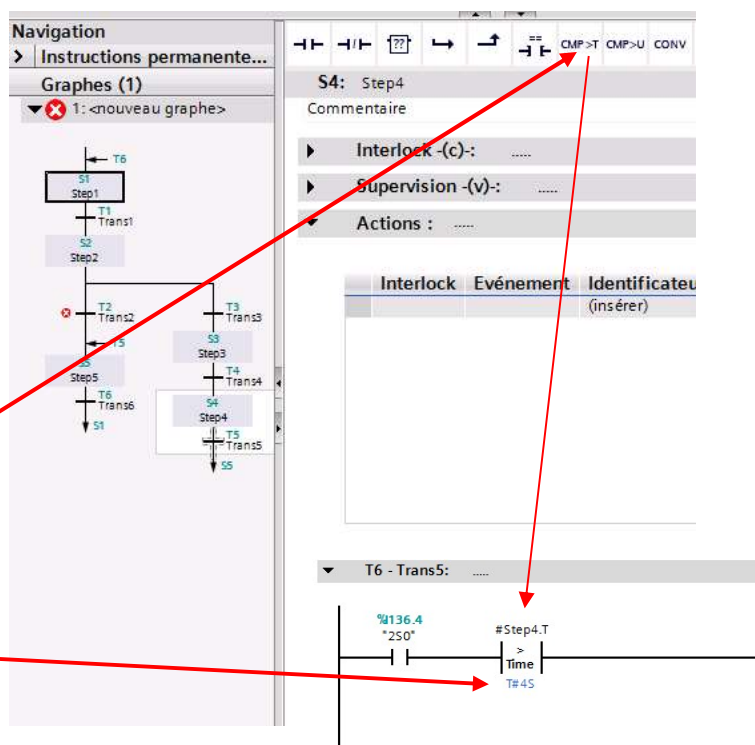
Transition entre l'étape 2 et l'étape 5

- Insérer **un contact** par glisser&déplacer et lui donner une adresse.
- Insérer un **bloc boîte vide**,
- Cliquer dessus et choisir <
- Compléter les opérandes



Transition entre l'étape 3 et l'étape 5

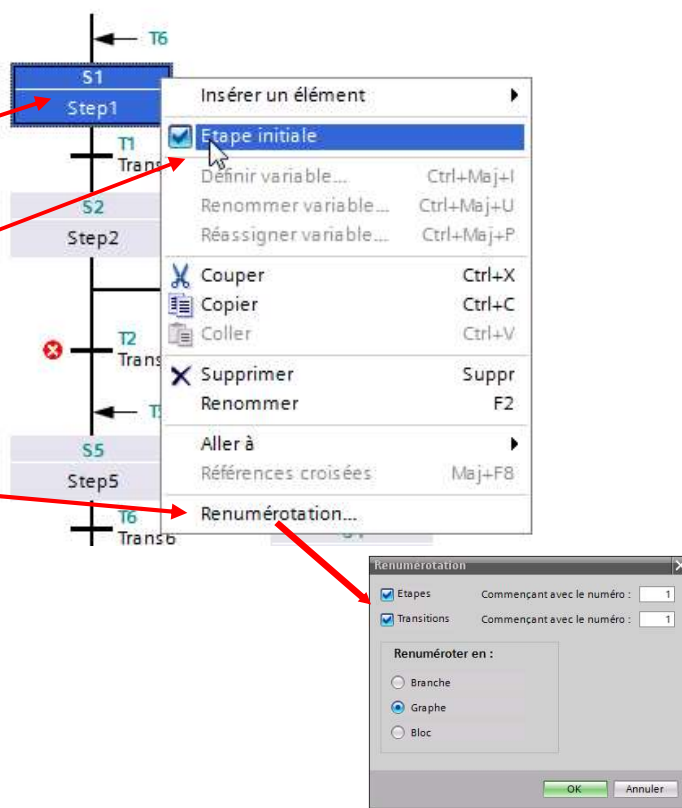
- Insérer un **contact** par glisser&déplacer et lui donner une adresse.
- Insérer un **bloc supérieur au temps d'activité d'étape**
- Modifier la valeur de la variable temps



Continuer de la même manière afin d'écrire toutes les réceptivités associées aux transitions

➤ Modifier les propriétés d'une étape

- Cliquer sur le repère et le numéro d'étape pour les modifier
- Cliquez droit pour définir une étape initiale
- Cliquez droit pour renuméroter l'ensemble des étapes ou des transitions



➤ Ecrire les actions associées aux étapes

Un tableau récapitulatif des différentes actions se situe page 36 et page 37, il n'est pas complet ; se reporter à l'aide en ligne TIA PORTAL.

Chaque action est définie par un identificateur, une adresse et éventuellement un évènement et ou une condition d'interlock (qui permet de programmer une action conditionnelle).

Interlock	Evénement	Identificateur
- Pas de condition -(C)- - Interlock	- Pas d'événement A1 - Acquiescement de message L0 - Condition Interlock apparaissant L1 - Condition Interlock disparaissant R1 - Enregistrement activé S0 - Etape désactivée S1 - Etape activée V0 - Erreur de surveillance disparaissant V1 - Erreur de surveillance apparaissant	- Pas d'opération CD - Décomptage CR - Réinitialiser le compteur CS - Définir valeur initiale du compteur CU - Comptage D - Retard à la montée L - Activer pour une durée limitée N - Mise à 1 tant que l'étape est active ON - Activer l'étape OFF - Désactiver l'étape R - Mettre à 0 S - Mettre à 1

Ecriture de l'action associée à l'étape 2

- Cliquer deux fois sur l'étape pour ouvrir la fenêtre des actions
- Choisir l'évènement (front montant) **S1**
- Choisir l'identificateur **N** (l'action est vraie tant que l'étape est active)
- Ecrire l'action **C1 := C1 + 1**
- Choisir l'identificateur **N** (l'action est vraie tant que l'étape est active)
- Indiquer la variable associée à l'action **1YV14**

Interlock	Evénement	Identificateur	Action
	S1	N - Mettre à 1 tant que l'étape est active	"C1":="C1"+16#1
		N - Mettre à 1 tant que l'étape est active	"1YV14"
		<ajouter>	

Ecriture de l'action associée à l'étape 4

- Cliquer deux fois sur l'étape pour ouvrir la fenêtre des actions
- Choisir l'identificateur **N** (l'action est vraie tant que l'étape est active)
- Indiquer la variable associée à l'action **2YV14**
- Sélectionner **(C)** dans Interlock
- Sélectionner **S1** dans Evènement
- Choisir l'identificateur **N** (l'action est vraie tant que l'étape est active)
- Ecrire l'action **C1 := 0**
- Ouvrir le réseau Interlock
- Ecrire la condition de verrouillage

Interlock -(c)-:			
Supervision -(v)-:			
Actions :			
-(C)-	Interlock	Evénement	Identificateur
	-(C)-	S1	N - Mise à 1 tant que l'étape est active
			N - Mise à 1 tant que l'étape est active
			"2YV14"
			"C1":=0

S4: Step4	
Commentaire	
Interlock -(c)-:	
%I36.5	"SP"
	Verrouillage
	(C)

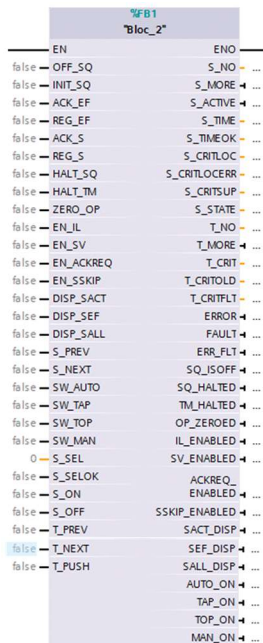
Continuer de la même manière afin d'écrire toutes les actions associées aux étapes

➤ Paramètres d'appel du bloc

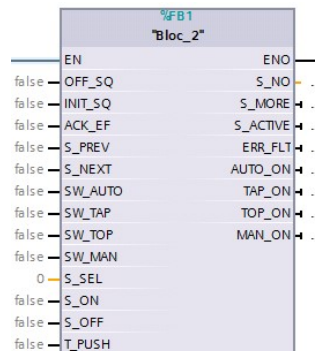
Comme pour un bloc FC, le bloc FB doit être appelé dans le bloc OB1. Lorsqu'on appelle un bloc FB, le logiciel crée également un DB d'instance servant de zone mémoire.

Pour appeler le bloc FB1, vous disposez des deux possibilités de sélection suivantes :

Paramètres d'interface maximaux



Paramètres d'interface par défaut



Cette sélection vous permet de définir le nombre de paramètres d'entrée et de sortie. Le jeu de paramètres standard permet le fonctionnement du graphe séquentiel dans différents modes de fonctionnement et d'acquiescer des messages.

Le jeu de paramètres maximum fournit des possibilités diagnostiques supplémentaires. Vous pouvez supprimer ou ajouter, à la main, différents paramètres dans les deux jeux de paramètres.

Les différents paramètres FB et leur signification se situent aux pages 34 et 35

Modifier le jeu de paramètres d'un bloc ouvert

Dans le menu "Edition" sélectionnez la commande "Paramètres d'interface maximaux" ou la commande "Paramètres d'interface par défaut".

Présentation des paramètres d'entrée d'un bloc GRAPH

Le tableau suivant montre un aperçu des paramètres d'entrée d'un bloc GRAPH :

Paramètres	Type de données	Description	Jeu de paramètres standard	Jeu de paramètres maximum
EN	BOOL	Entrée de validation Si EN n'est pas connecté, le FB continue à être traité.	X	X
OFF_SQ	BOOL	OFF_SEQUENCE: Désactiver le graphe séquentiel, autrement dit désactiver toutes les étapes	X	X
INIT_SQ	BOOL	INIT_SEQUENCE: Activer les étapes initiales, réinitialiser le graphe séquentiel	X	X
ACK_EF	BOOL	ACKNOWLEDGE_ERROR_FAULT: Acquittement d'une anomalie, forcer le passage à l'étape suivante	X	X
REG_EF	BOOL	REGISTRATE_ERROR_FAULT: Enregistrer toutes les erreurs et anomalies	-	X
ACK_S	BOOL	ACKNOWLEDGE_STEP: Acquitter l'étape affichée dans S_NO	-	X
REG_S	BOOL	REGISTRATE_STEP: Enregistrer l'étape affichée dans S_NO	-	X
HALT_SQ	BOOL	HALT_SEQUENCE: Arrêter/réactiver la chaîne séquentielle	-	X
HALT_TM	BOOL	HALT_TIMES: Arrêter/réactiver tous les temps d'activation d'étape et les opérations (L et D) dépendantes du temps du graphe	-	X
ZERO_OP	BOOL	ZERO_OPERANDS: Remettre à zéro tous les opérandes ayant l'identification N, D, L dans les étapes actives et ne pas exécuter les instructions CALL dans les actions/réactiver les opérandes et les instructions CALL	-	X
EN_IL	BOOL	ENABLE_INTERLOCKS: Désactiver/réactiver les verrouillages d'étape (Interlock) (le graphe séquentiel se comporte comme si le verrouillage d'étape était satisfait)	-	X
EN_SV	BOOL	ENABLE_SUPERVISIONS: Désactiver/réactiver les verrouillages d'étape (Supervision) (le graphe séquentiel se comporte comme si le verrouillage d'étape n'était pas satisfait)	-	X
EN_ACKREQ	BOOL	ENABLE_ACKNOWLEDGE_REQUIRED: Activer l'acquittement obligatoire	-	X
EN_SSKIP	BOOL	ENABLE_STEP_SKIPPING: Activer Ignorer une étape	-	X
DISP_SACT	BOOL	DISPLAY_ACTIVE_STEPS: Activer uniquement des étapes actives	-	X
DISP_SEF	BOOL	DISPLAY_STEPS_WITH_ERROR_OR_FAULT: Afficher uniquement les étapes erronées et anormales	-	X
DISP_SALL	BOOL	DISPLAY_ALL_STEPS: Afficher toutes les étapes	-	X
S_PREV	BOOL	PREVIOUS_STEP: Fonctionnement automatique : Revenir en arrière dans les étapes actuellement actives, affichage des numéros d'étape dans S_NO Mode manuel : Afficher l'étape précédente dans S_NO (numéro plus petit)	X	X
S_NEXT	BOOL	NEXT_STEP: Fonctionnement automatique : Feuilleter dans les étapes actuellement actives, affichage du numéro d'étape dans S_NO Mode manuel : Afficher l'étape suivante dans S_NO (numéro plus grand)	X	X
SW_AUTO	BOOL	SWITCH_MODE_AUTOMATIC: Changement de mode de fonctionnement : Mode automatique	X	X
SW_TAP	BOOL	SWITCH_MODE_AUTOMATIC: Changement de mode de fonctionnement : Mode semi-automatique	X	X
SW_TOP	BOOL	SWITCH_MODE_AUTOMATIC: Changement de mode de fonctionnement : Mode automatique ou semi-automatique	X	X
SW_MAN	BOOL	SWITCH_MODE_MANUAL: Changement de mode de fonctionnement : Mode manuel, aucun déroulement autonome n'est lancé	X	X
S_SEL	INT	STEP_SELECT: Sélectionner un numéro d'étape pour les paramètres de sortie S_NO en mode manuel, activer/désactiver avec S_ON, S_OFF	X	X
S_SELOK	BOOL	STEP_SELECT_OK: Reprendre la valeur dans S_SEL pour S_NO	-	X
S_ON	BOOL	STEP_ON: Mode manuel : Activer l'étape affichée	X	X
S_OFF	BOOL	STEP_OFF: Mode manuel : Désactiver l'étape affichée	X	X
T_PREV	BOOL	PREVIOUS_TRANSITION: Afficher la transition précédente valable dans T_NO	-	X
T_NEXT	BOOL	NEXT_TRANSITION: Afficher la transition suivante valable dans T_NO	-	X
T_PUSH	BOOL	PUSH_TRANSITION: La transition commute lorsque la condition est satisfaite et T_PUSH (front) Condition : Mode automatique ou mode manuel	X	X

Présentation des paramètres de sortie d'un bloc GRAPH

Le tableau suivant montre un aperçu des paramètres de sortie d'un bloc GRAPH :

Paramètres	Type de données	Description	Jeu de paramètres standard	Jeu de paramètres maximum
ENO	BOOL	Sortie de validation lorsque le FB est actif, ENO a la valeur 1 en l'absence d'erreur, sinon 0	x	x
S_NO	INT	STEP_NUMBER: Affichage numéro d'étape	x	x
S_MORE	BOOL	MORE_STEPS: D'autres étapes sont actives	x	x
S_ACTIVE	BOOL	STEP_ACTIVE: L'étape affichée est active	x	x
S_TIME	TIME	STEP_TIME: Temps d'activation d'une étape	-	x
S_TIMEOK	TIME	STEP_TIME_OK: Temps d'activation d'une étape sans erreur	-	x
S_CRITLOC	DWORD	STEP_CRITERIA_INTERLOCK: Bits de critères Interlock	-	x
S_CRITLOC ERR	DWORD	S_CRITERIA_IL_LAST_ERROR: Bits de critères Interlock pour l'événement L1	-	x
S_CRITSUP	DWORD	STEP_CRITERIA_SUPERVISION: Bits de critères de supervision	-	x
S_STATE	WORD	STEP_STATE: Bits d'état d'une étape	-	x
T_NO	INT	TRANSITION: Numéro de transition valable	-	x
T_MORE	BOOL	MORE_TRANSITIONS: Autres transitions variables prêtes à être affichées	-	x
T_CRIT	DWORD	TRANSITION_CRITERIA: Bits de critères de transition	-	x
T_CRITOLD	DWORD	T_CRITERIA_LAST_CYCLE: Bits de critères de transition du dernier cycle	-	x
T_CRITFLT	DWORD	T_CRITERIA_LAST_FAULT: Bits de critères de transition pour l'événement V1	-	x
ERROR	BOOL	INTERLOCK_ERROR: Erreur Interlock (n'importe quelle étape)	-	x
FAULT	BOOL	SUPERVISION_FAULT: Erreur de supervision (n'importe quelle étape)	-	x
ERR_FLT	BOOL	IL_ERROR_OR_SV_FAULT: Anomalie groupée	x	x
SQ_ISOFF	BOOL	SEQUENCE_IS_OFF: Le graphe séquentiel est désactivé (aucune étape active)	-	x
SQ_HALTED	BOOL	SEQUENCE_IS_HALTED: Graphe séquentiel arrêté	-	x
TM_HALTED	BOOL	TIMES_ARE_HALTED: Temps arrêtés	-	x
OP_ZEROED	BOOL	OPERANDS_ARE_ZEROED: Opérandes réinitialisés	-	x
IL_ENABLED	BOOL	INTERLOCK_IS_ENABLED: Le verrouillage d'étape est pris en compte	-	x
SV_ENABLED	BOOL	SUPERVISION_IS_ENABLED: La surveillance d'étape est prise en compte	-	x
ACKREQ_ENABLED	BOOL	ACKNOWLEDGE_REQUIRED_IS_ENABLED: L'acquiescement obligatoire est activé	-	x
SSKIP_ENABLED	BOOL	STEP_SKIPPING_IS_ENABLED: La fonction ignorer une étape est activée	-	x
SACT_DISP	BOOL	ACTIVE_STEPS_WERE_DISPLAYED: Afficher uniquement les étapes actives dans S_NO	-	x
SEF_DISP	BOOL	STEPS_WITH_ERROR_FAULT_WERE_DISPLAYED: Afficher uniquement les étapes erronées et anormales dans S_NO	-	x
SALL_DISP	BOOL	ALL_STEPS_WERE_DISPLAYED: Afficher toutes les étapes dans S_NO	-	x
AUTO_ON	BOOL	AUTOMATIC_IS_ON: Affichage Mode automatique	x	x
TAP_ON	BOOL	T_AND_PUSH_IS_ON: Affichage mode semi-automatique	x	x
TOP_ON	BOOL	T_OR_PUSH_IS_ON: Affichage mode semi-automatique	x	x
MAN_ON	BOOL	MANUAL_IS_ON: Affichage mode manuel	x	x

Présentation

Le tableau suivant présente des actions indépendantes d'un événement avec et sans Interlock :

Interlock	Evénement	Identification	Action	Condition	Etat logique
		N		Une étape est active.	L'opérande transmet le signal 1.
		S			L'opérande est mis à 1.
		R			L'opérande est mis à 0 et reste ensuite à 0.
		N	CALL		Le bloc indiqué est appelé.
		L	<Zeitwert> (Valeur de temporisation)		L'opérande transmet le signal 1 pendant n secondes.
		D	<Valeur de temporisation>		n secondes après l'activation de l'étape, l'opérande transmet le signal 1 pour la durée de l'activation de l'étape. Ceci ne s'applique pas lorsque la durée de l'activation de l'étape est inférieure à n secondes.
		TF	<Valeur de temporisation>		Avec l'activation de l'étape, l'opérande transmet le signal 1. Avec la désactivation de l'étape, la temporisation s'écoule et l'opérande est mis sur 0 une fois seulement que la temporisation est écoulée.
-(C)-		N		Une étape est active et la condition Interlock est remplie.	L'opérand transmet le signal 1.
-(C)-		S			L'opérande est mis à 1 et reste ensuite à 1.
-(C)-		R			L'opérande est mis à 0 et reste ensuite à 0.
-(C)-		N	CALL		Le bloc indiqué est appelé.
-(C)-		L	<Valeur de temporisation>		L'opérande transmet le signal 1 pendant n secondes. Si l'étape n'est pas active, il transmet le signal 0.
-(C)-		D	<Valeur de temporisation>		L'opérande transmet le signal 1 n secondes après l'activation de l'étape et tant que l'étape est active et que la condition est remplie. Si l'étape n'est pas active, l'opérande transmet le signal 0.

Actions qui dépendent d'un événement avec et sans Interlock

Ces actions sont exécutées une fois lorsque l'événement survient et si l'étape correspondante est active.

Lors de la poursuite de l'exécution de la chaîne séquentielle, une action qui dépend d'un événement n'est réexécutée que si l'événement survient une nouvelle fois.

Le tableau ci-dessous montre des actions reliées à une étape apparaissante.

Interlock	Evénement	Identification	Action	Condition	Etat logique
	S1	N		Une étape est activée (apparaissante).	L'opérande transmet une seule fois le signal 1.
	S1	S			L'opérande est mis à 1.
	S1	R			L'opérande est mis à 0.
	S1	N	CALL		Le bloc indiqué est appelé.
	S1	ON	Nom d'étape		L'étape est activée.
	S1	OFF	Nom d'étape		L'étape est désactivée.
	S1	OFF	S_ALL		Toutes les étapes sont désactivées, hormis l'étape dans laquelle se trouve l'action.
-(C)-	S1	N		Une étape est activée et la condition Interlock est remplie.	L'opérande transmet une seule fois le signal 1.
-(C)-	S1	S			L'opérande est mis à 1.
-(C)-	S1	R			L'opérande est mis à 0.
-(C)-	S1	N	CALL		Le bloc indiqué est appelé.
-(C)-	S1	ON	Nom d'étape		L'étape est activée.
-(C)-	S1	OFF	Nom d'étape		L'étape est désactivée.
-(C)-	S1	OFF	S_ALL		Toutes les étapes sont désactivées, hormis l'étape dans laquelle se trouve l'action.

Le tableau ci-dessous montre des compteurs dans les actions reliés à une étape apparaissante :

Interlock	Événement	Identification	Action	Condition	Etat logique
	S1	CS	Opérande, <valeur de compteur>	Une étape est active.	La valeur initiale du compteur est chargée dans le compteur dès que l'étape devient active (apparaissante).
	S1	CU	Opérande		Dès que l'étape est activée, le compteur avance de 1.
	S1	CD	Opérande		Dès que l'étape est activée, le compteur recule de 1.
	S1	CR	Opérande		Dès que l'étape est activée, le compteur est remis à 0.
-(C)-	S1	CS	Opérande, <valeur de compteur>	Une étape est active et la condition Interlock est remplie.	La valeur d'état du compteur est chargée dans le compteur dès que l'étape est activée et que la condition est remplie.
-(C)-	S1	CU	Opérande		Dès que l'étape est activée et que la condition est remplie, le compteur avance de 1.
-(C)-	S1	CD	Opérande		Dès que l'étape est activée et que la condition est remplie, le compteur recule de 1.
-(C)-	S1	CR	Opérande		Dès que l'étape est activée et que la condition est remplie, le compteur est remis à 0.

Le tableau ci-dessous montre des compteurs dans les actions reliés à une étape disparaissante :

Interlock	Événement	Identification	Action	Condition	Etat logique
	S0	CS	Opérande, <valeur de compteur>	Une étape est active.	La valeur initiale du compteur est chargée dans le compteur dès que l'étape est désactivée (disparaît).
	S0	CU	Opérande		Dès que l'étape est désactivée, le compteur avance de 1.
	S0	CD	Opérande		Dès que l'étape est désactivée, le compteur recule de 1.
	S0	CR	Opérande		Dès que l'étape est désactivée, le compteur est remis à 0.