





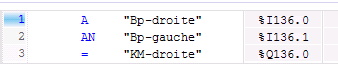


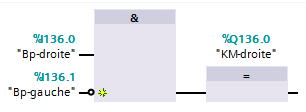
Dossier ressource

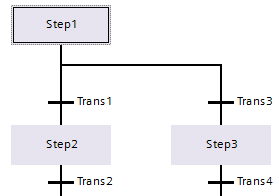
Automates SIEMENS

Logiciel TIA PORTAL

STEP 7







**SOMMAIRE**

[Introduction 2](#_Toc168241458)

[Lancement du logiciel 2](#_Toc168241459)

[Présentation de l’environnement TIA PORTAL 3](#_Toc168241460)

[ Vue du portail 3](#_Toc168241461)

[ Vue du projet 4](#_Toc168241462)

[ Navigateur de projet 5](#_Toc168241463)

[ Task Cards 6](#_Toc168241464)

[ Fenêtre d’inspection 7](#_Toc168241465)

[Création d’un projet TIA PORTAL 7](#_Toc168241466)

[ Sélection des paramètres de sécurité 7](#_Toc168241467)

[ Créer une station matérielle 7](#_Toc168241473)

[ Configurer la station matérielle 9](#_Toc168241474)

[**Adresses Ethernet de l’API** 9](#_Toc168241475)

[**Paramétrage protection et sécurité** 9](#_Toc168241476)

[**Paramétrage mise en route de la CPU** 9](#_Toc168241480)

[**Paramétrage Mémento de cadence** 10](#_Toc168241487)

[**Configuration des Entrées (%I) et Sorties (%Q) dans la CPU et dans les modules** 10](#_Toc168241488)

[ Compiler la configuration matérielle 11](#_Toc168241489)

[ Charger la configuration matérielle 12](#_Toc168241492)

[ Création d’un programme 12](#_Toc168241493)

[ Compiler les blocs programme 17](#_Toc168241494)

[ Charger les blocs dans la CPU 18](#_Toc168241495)

[ Visualiser les blocs 19](#_Toc168241496)

[ Visualiser / forcer des variables : table de visualisation 19](#_Toc168241497)

[Programmation Temporisations 21](#_Toc168241500)

[ Langage CONT 21](#_Toc168241501)

[ Langage SCL 22](#_Toc168241502)

[Programmation Compteurs 23](#_Toc168241503)

[ Par incrémentation d’un mot en langage CONT 23](#_Toc168241507)

[ Par utilisation des blocs compteurs intégrés en langage CONT 24](#_Toc168241521)

[ Par utilisation des blocs compteurs intégrés en langage SCL 25](#_Toc168241522)

[Programmation langage GRAPH 26](#_Toc168241523)

Introduction

La plateforme **T**otally **I**ntegrated **A**utomation **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 :

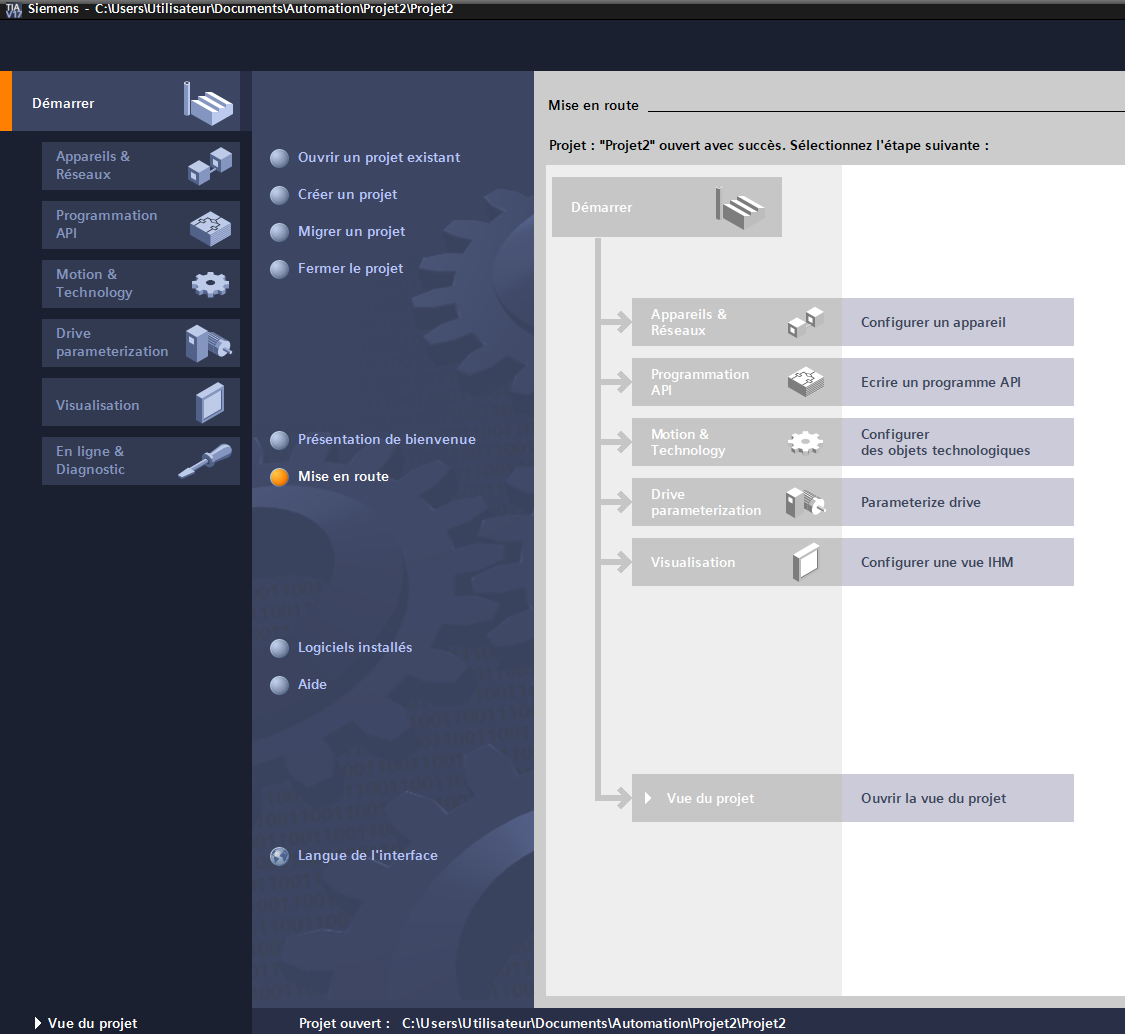
* **SIMATIC STEP 7 V17** permettant de programmer les automates programmables des gammes :
* **S7-1200**
* **S7-1500**
* **S7-300**
* **SIMATIC WinCC V17** permettant de programmer les **I**nterfaces **H**omme **M**achine

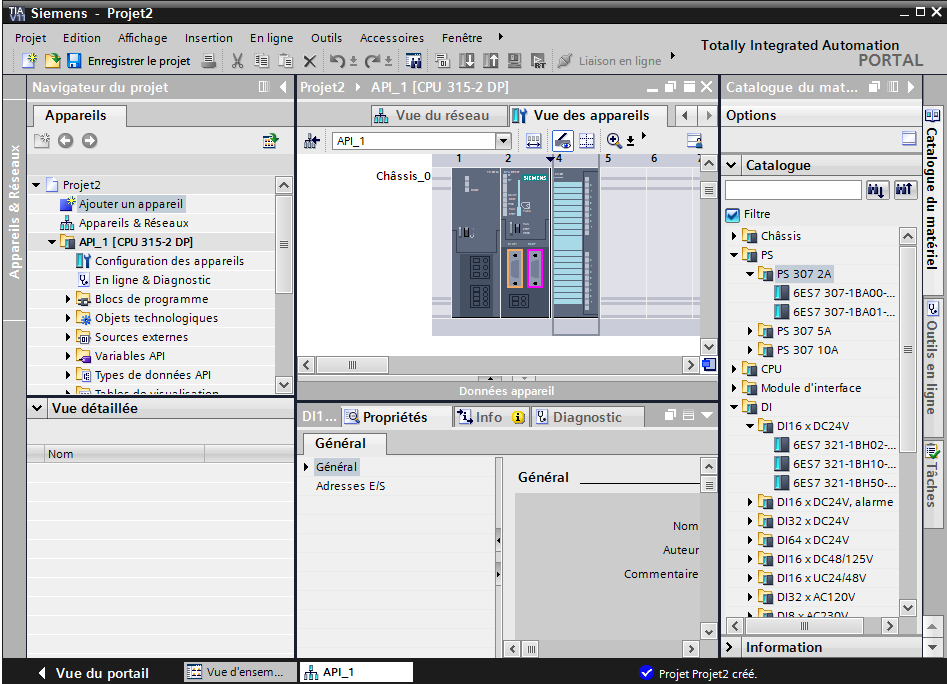
Lancement du logiciel



**Cliquer sur l’icône**

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



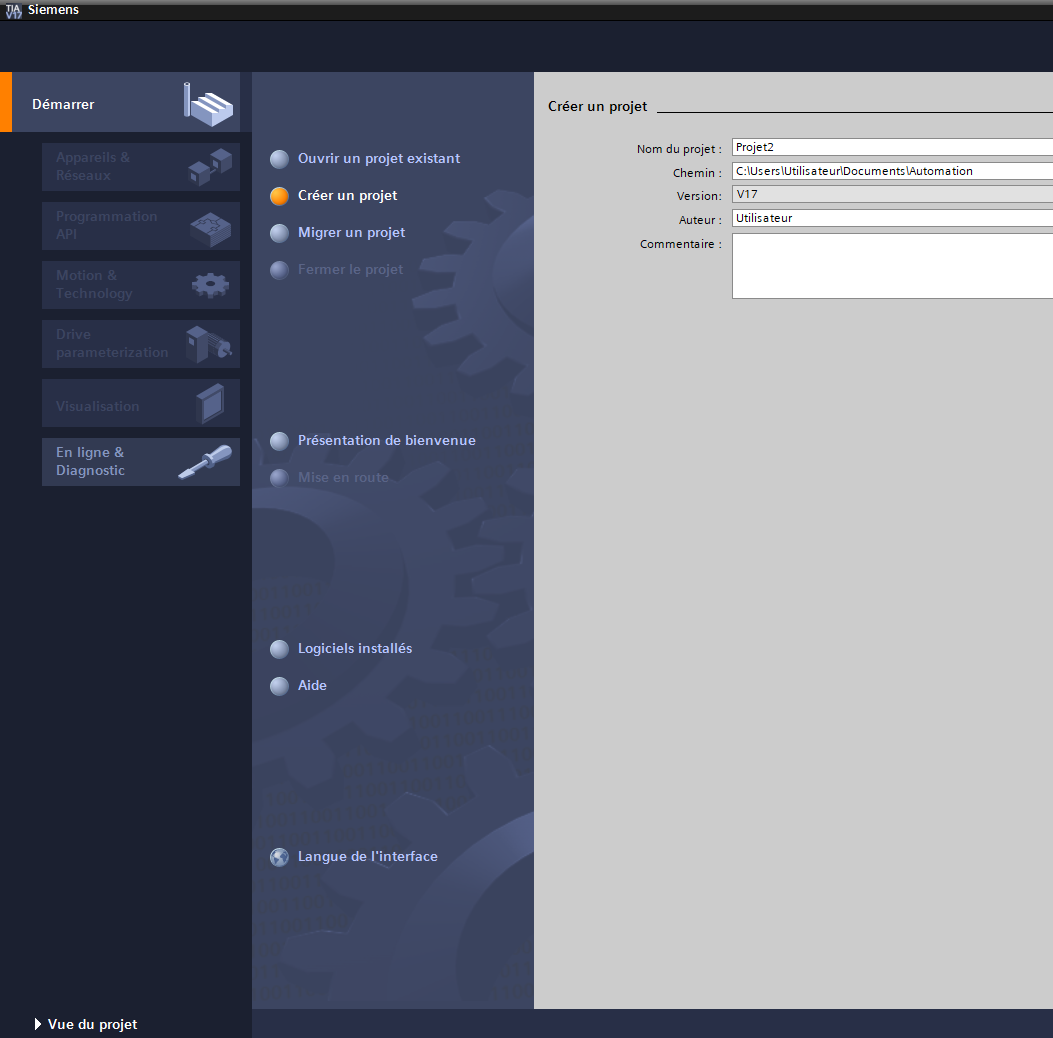


Présentation de l’environnement TIA PORTAL

* Vue du portail

Elle permet :

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



**Actions**

**Portails pour les différentes tâches à exécuter**

**Fenêtre de sélection de l’action à exécuter**

**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.

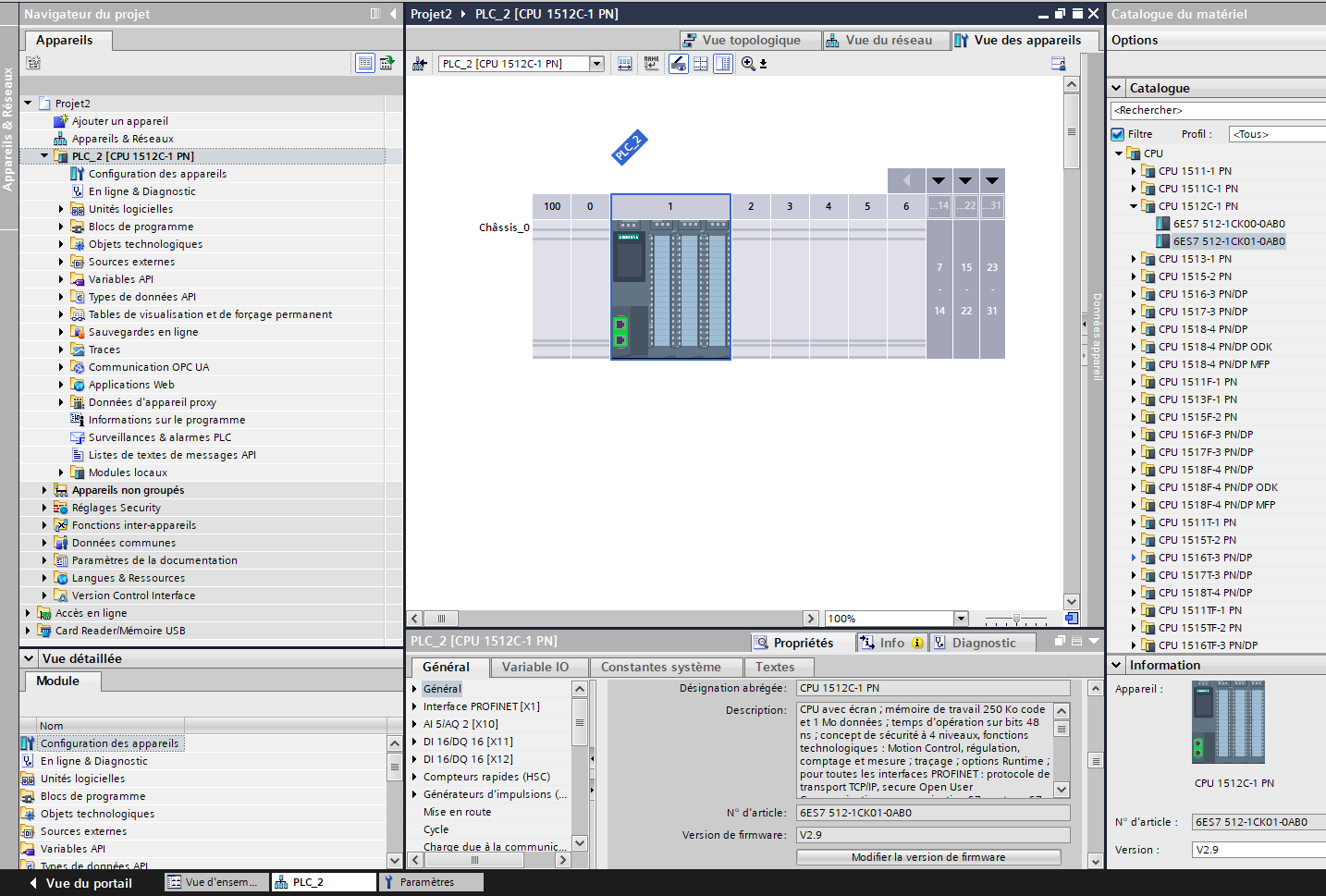
**Fenêtre de sélection**

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**

**Fenêtre de travail**

**Task Cards :**

**Onglet de sélection des tâches**

**Vue détaillée**

**Fenêtre d’inspection**

**Barre des tâches**

**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.

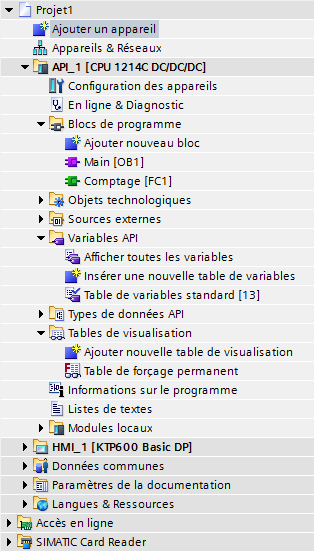
**Remarque** : Il est possible de partager la fenêtre de travail en utilisant différentes icônes.

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



* Navigateur de projet

**Projet**



**CPU**

**Programme CPU**

**Variables API**

**Langues et ressources (pour le projet dans son ensemble)**

**Modules locaux**

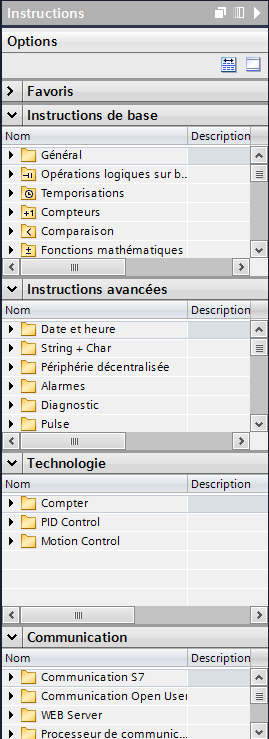
**(périphérie décentralisée)**

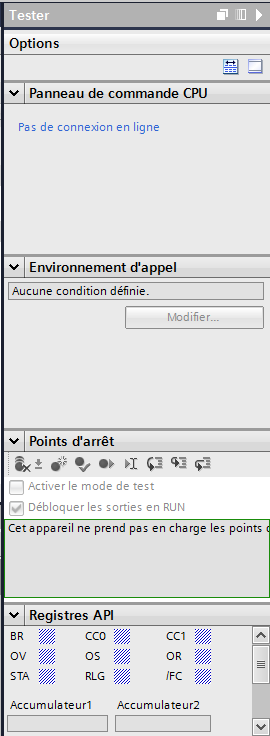
**Contrôle commande Interface Homme/Machine**

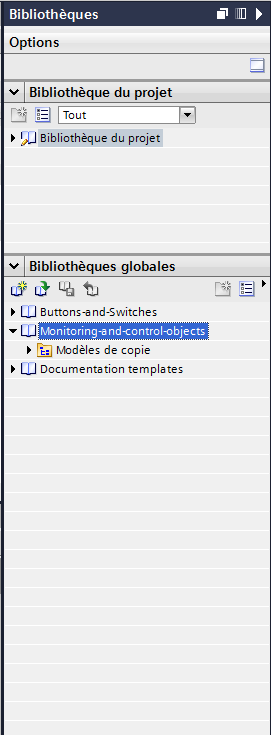
**Accès en ligne**

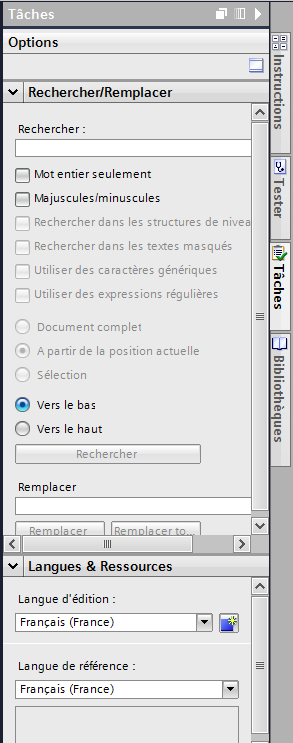
**Lecteur de carte SIMATIC**

* 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

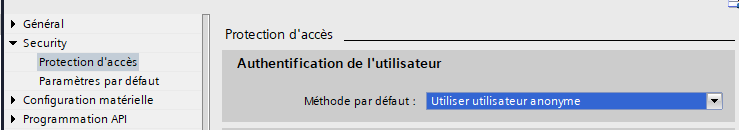
Un projet contient l’ensemble des programmes et des données nécessaires pour effectuer une tâche d’automatisation. Ce projet peut contenir un ou plusieurs programmes utilisés par une ou plusieurs CPU.

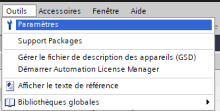
Après avoir créé un nouveau projet suivre la démarche décrite ci-dessous.

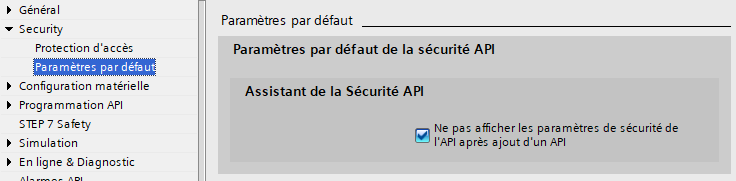
* Sélection des paramètres de sécurité

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.

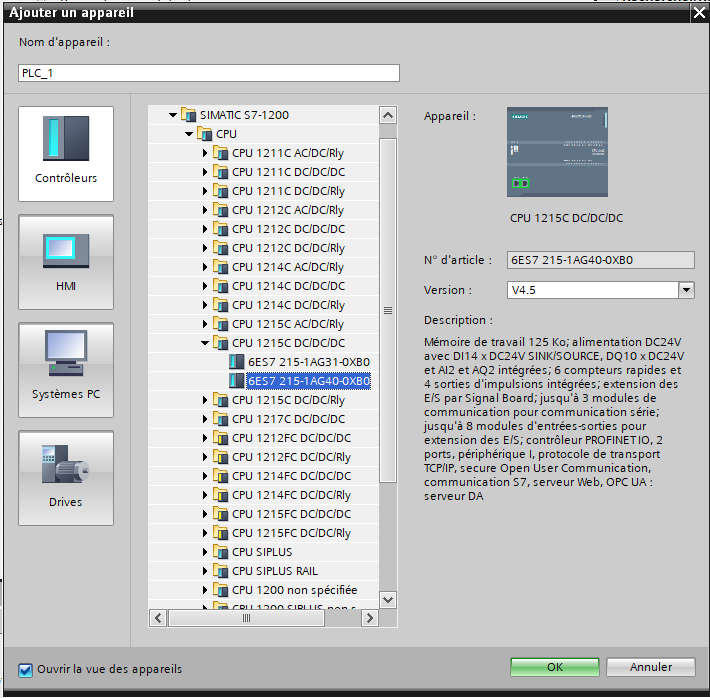


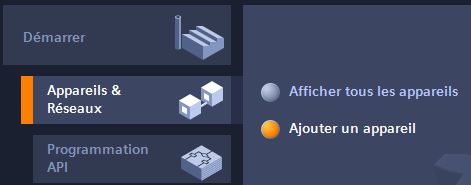


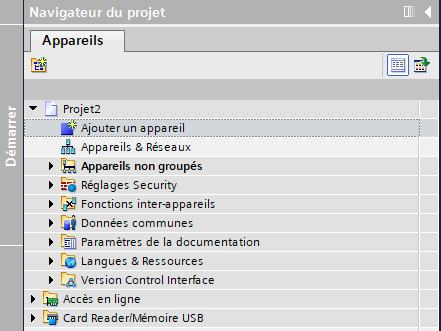


* Créer une station matérielle

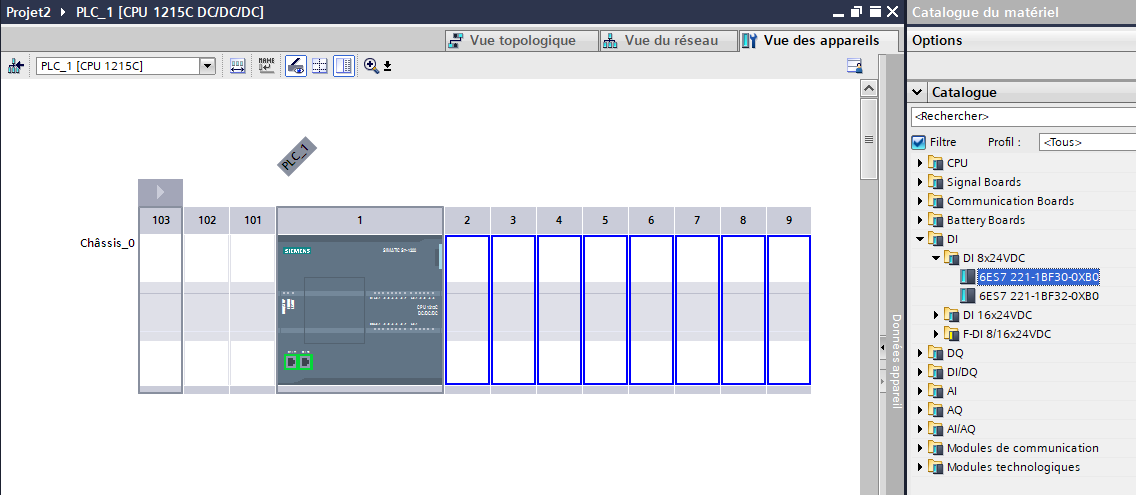
Lors de la création d’un nouvel appareil (SIMATIC S7-1200, SIMATIC S7-1500), un châssis adapté est automatiquement créé. L’appareil sélectionné est enfiché au premier emplacement autorisé du châssis. Indépendamment du chemin choisi, l’appareil ajouté est visible dans la vue des appareils et la vue du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

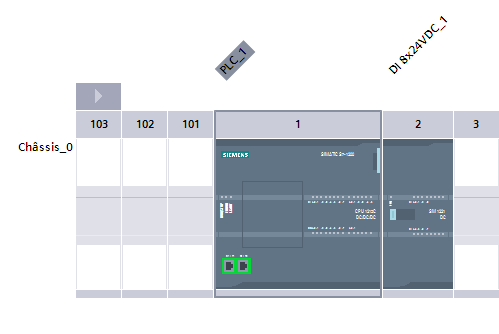
Il est possible de supprimer des appareils complets ou des stations entières dans la vue du réseau, et des composants individuels d’une station dans la vue des appareils. Les composants matériels supprimés sont effacés du système et les adresses occupées sont libérées.

**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.

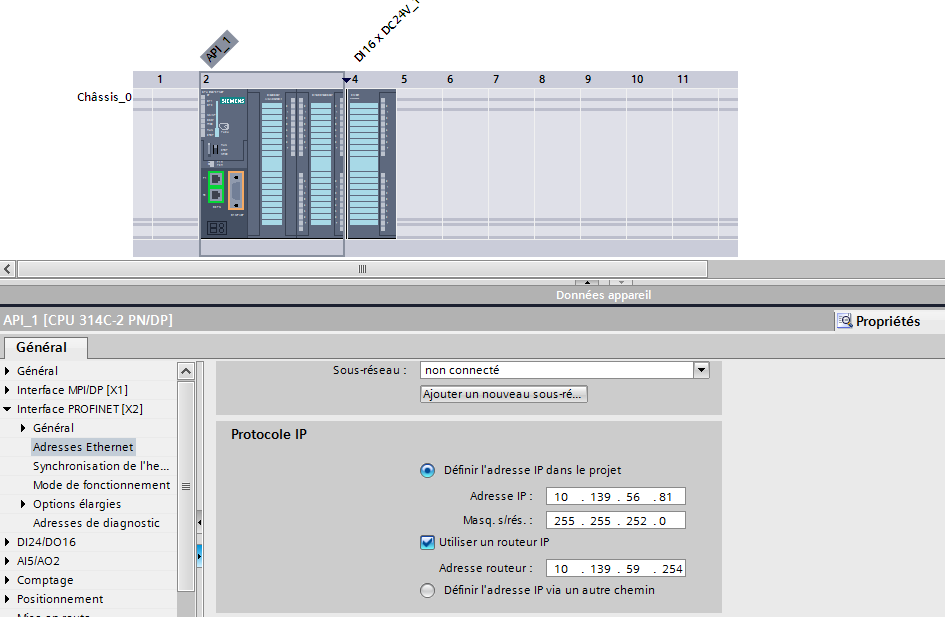
* Configurer la station matérielle

**Adresses Ethernet de l’API**



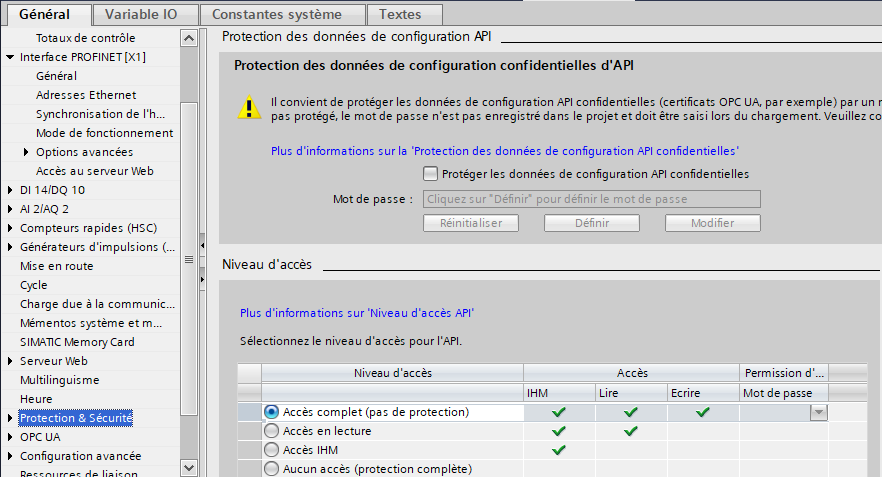
Il est nécessaire d’indiquer dans la fenêtre Protocole IP l’adresse Ethernet de l’API

* 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.

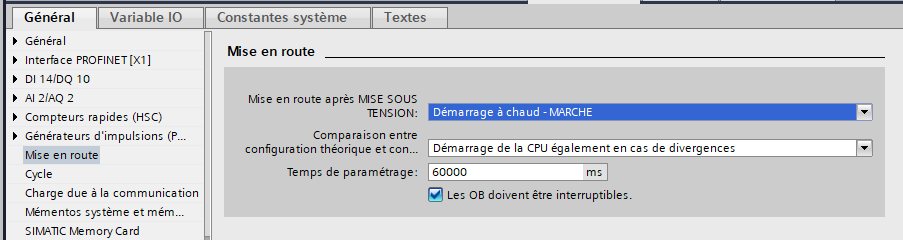


**Paramétrage mise en route de la CPU**

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.

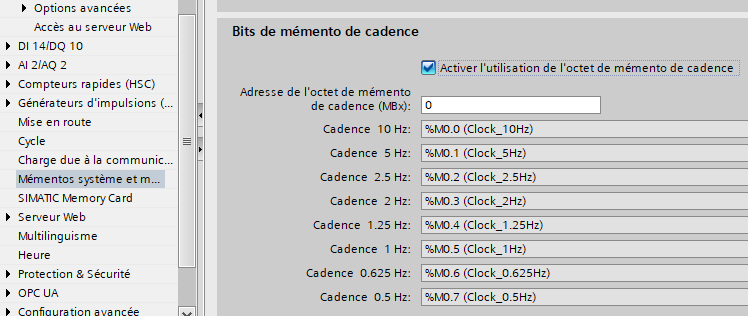


**Paramétrage Mémento de cadence**

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.

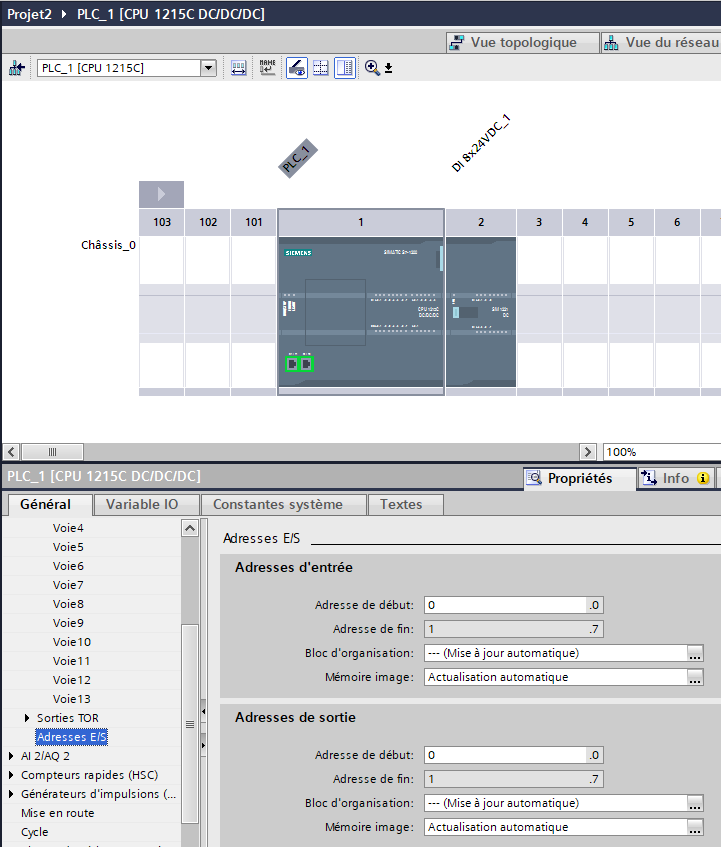
Attention de ne pas utiliser le même memento pour une autre variable du programme.

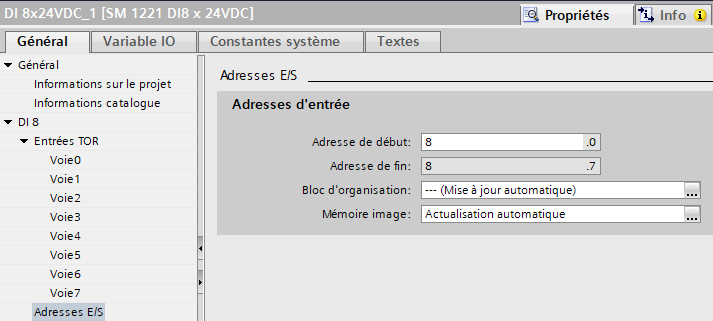


**Configuration des Entrées (%I) et Sorties (%Q) dans la CPU et dans les modules**

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 valeures.





**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,

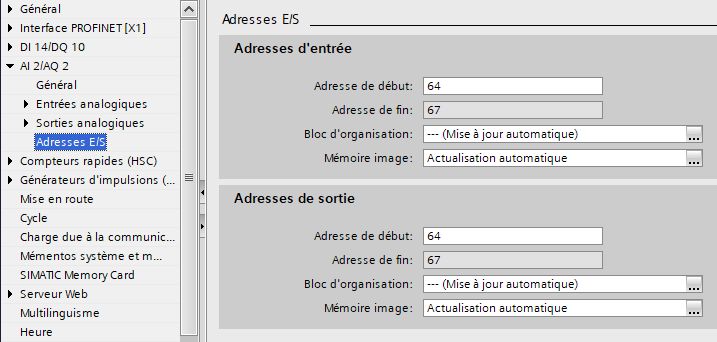
**7** l'adresse de bit.

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.

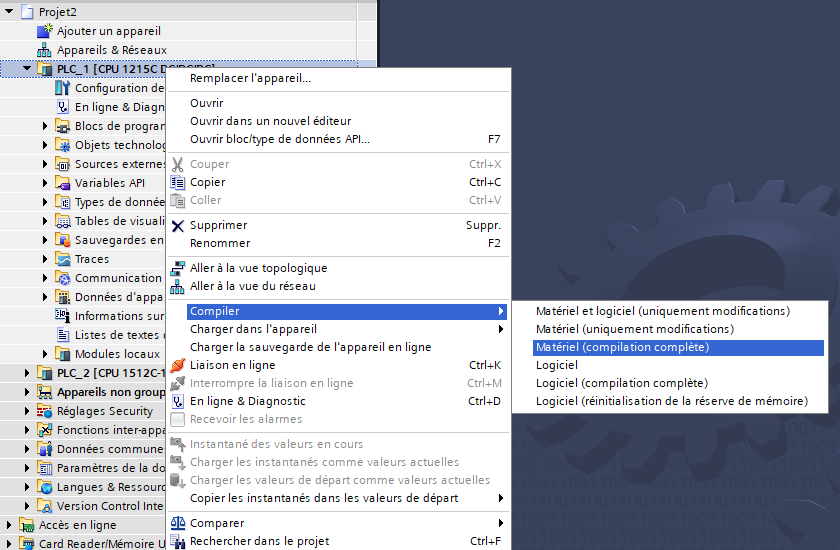
Les entrées et sorties analogiques sont affectées par groupes de deux entrées ou deux sorties.

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

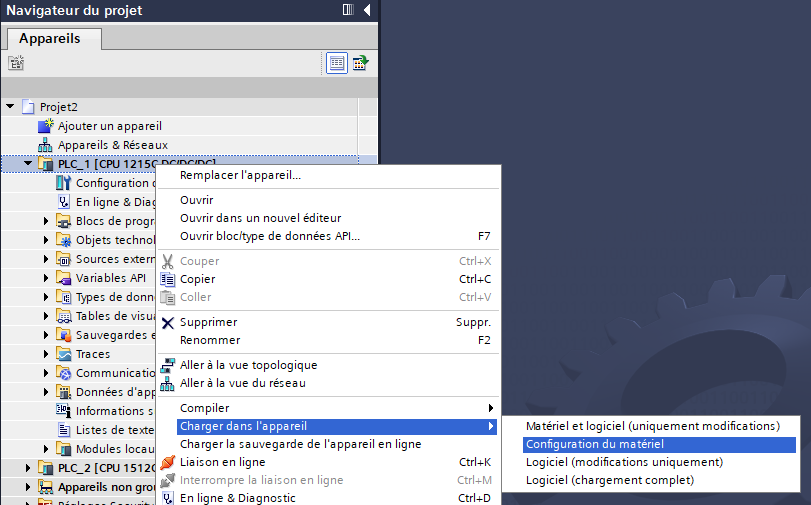


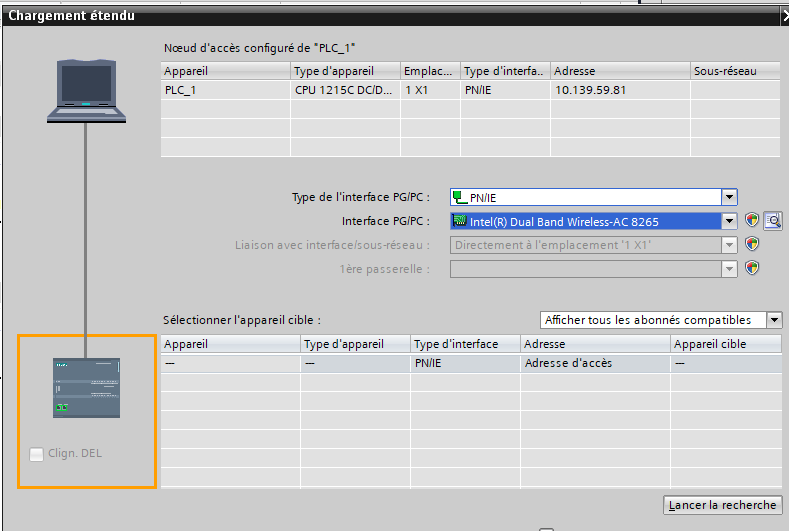
Après compilation, vérifier dans la fenêtre d’inspection si la compilation a été réalisée sans erreur. En cas d’erreurs, vérifier le type d’erreur dans la fenêtre d’inspection et corriger. Il est impossible de charger une configuration, avec des erreurs, dans l’automate.

* 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.

Clic droit





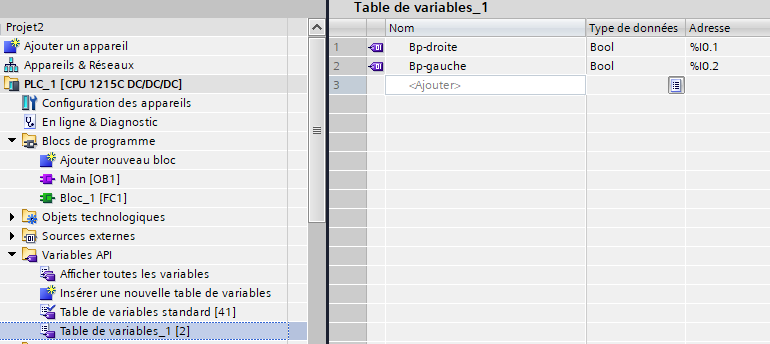
Choisir la carte Ethernet

Permet de se connecter à l’automate désiré

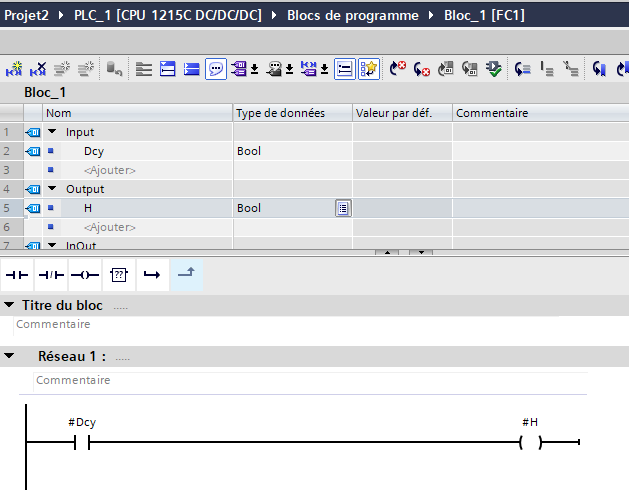
* Création d’un programme

**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.



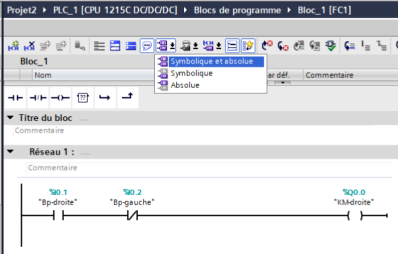
**Variables locales *:*** Elles sont définies dans la partie déclarative d’un bloc. Elles ne peuvent être utilisées que dans ce même bloc. On peut donc réutiliser le même nom symbolique dans la partie déclarative d’un autre bloc.

* **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.

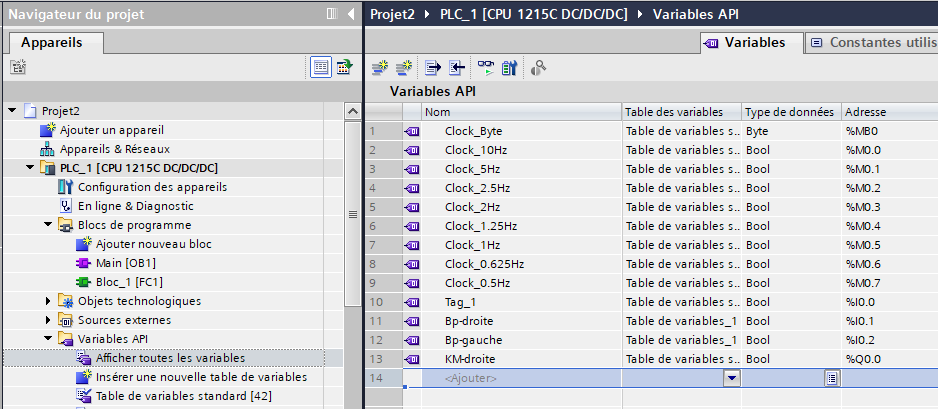


* **Table de variables API**

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.



**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)**

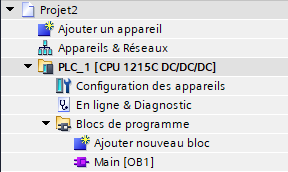
Ils constituent l’interface entre le système d’exploitation et le programme utilisateur. Les blocs d'organisation permettent de structurer votre programme. Une fois le traitement des OB de cycle de programme achevé, la CPU réexécute immédiatement les OB de cycle de programme. Ce traitement cyclique est 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 crée 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**

Bloc Fonction

**FC1**

Contient une partie du programme pour la commande du système.

Est appelé par l'OB1.

Bloc d'organisation

**OB1**

Bloc appelé cycliquement par le système d'exploitation.

C'est dans celui-ci que sont appelés les blocs fonction FC1, FC2, …

**CALL FC1**

**CALL FC2**

Bloc Fonction

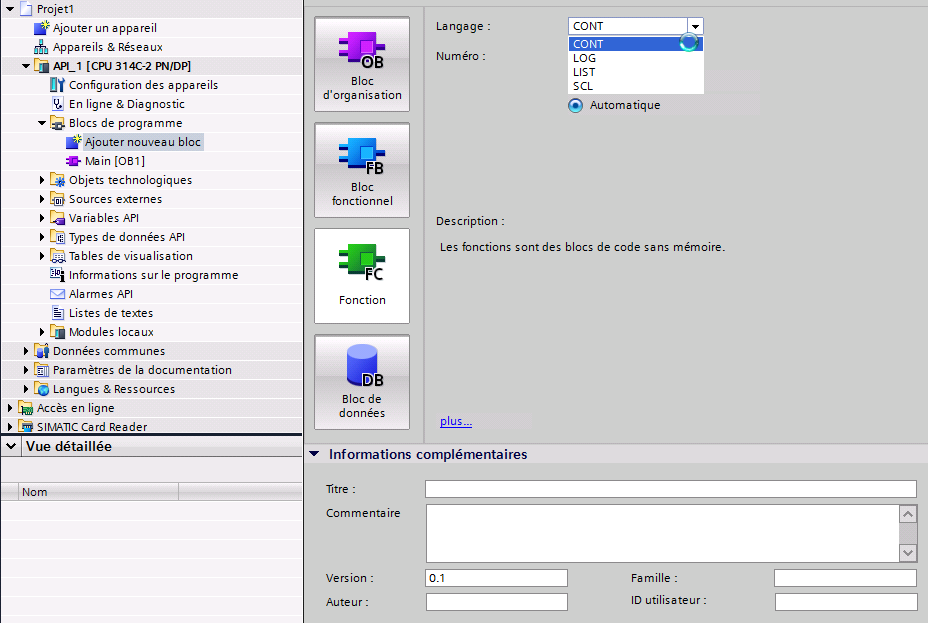
**FC2**

Contient une partie du programme pour la commande du système.

Est appelé par l'OB1.

* **Ajouter un nouveau bloc**

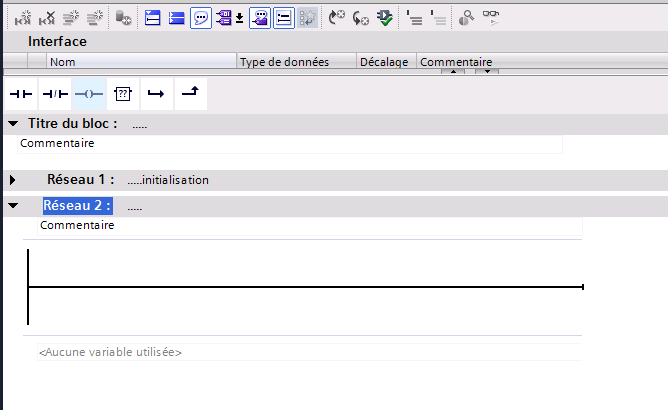
Pour ajouter un nouveau bloc, procédez comme ci-dessous. Lors de la création du nouveau bloc, il faut déterminer le type de bloc désiré, le langage de programmation, le nom symbolique et le numéro. Le numéro du bloc peut être attribué manuellement ou automatiquement. La fenêtre des « informations complémentaires » 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.



**Activer / Désactiver commentaires réseaux**

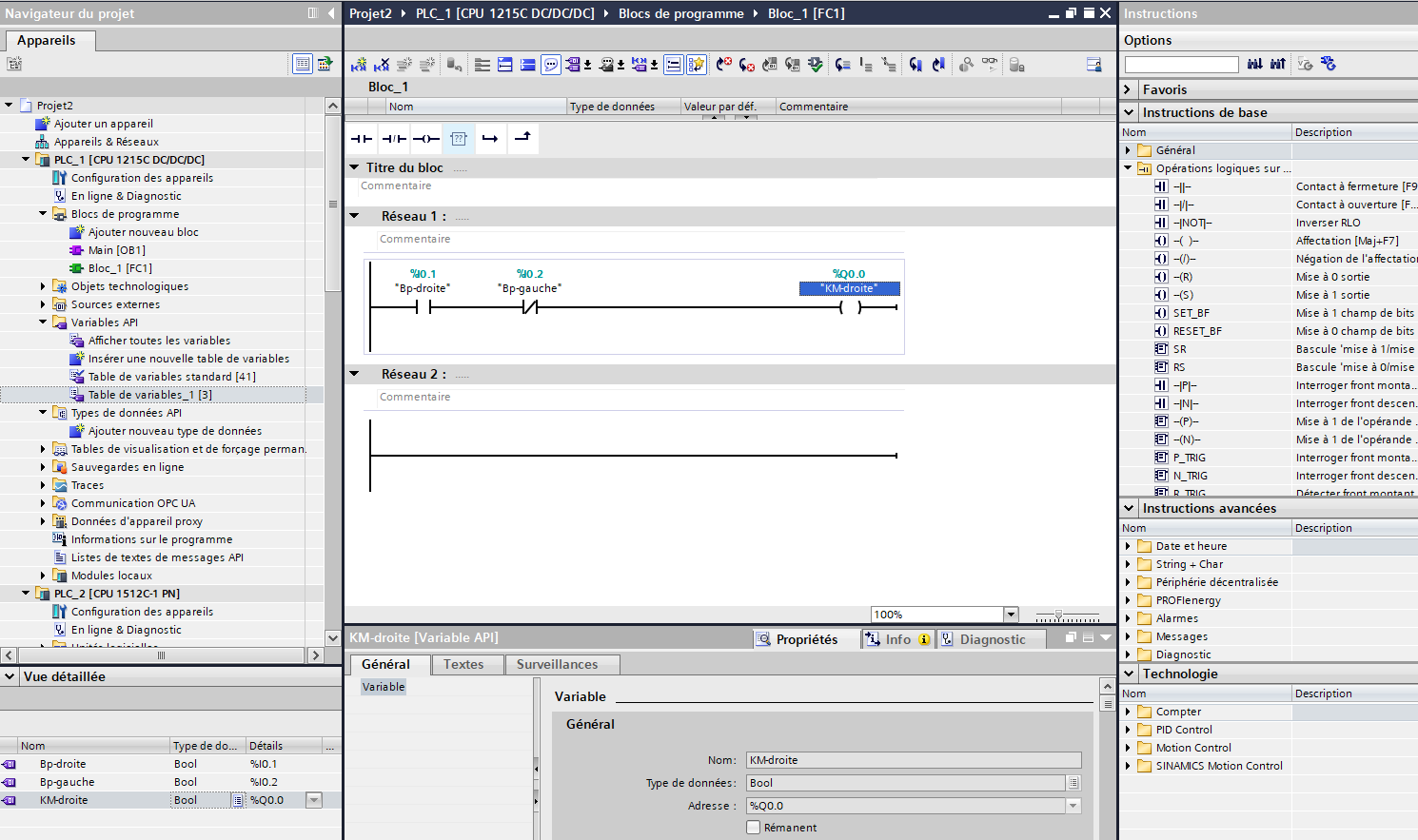
**Ouvrir / Fermer tous les réseaux**

**Activer / Désactiver commentaires instructions**

**Ajouter / Supprimer un réseau**

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

dessous.



**Ajouter un réseau**

**Afficher / Masquer**

**les favoris**

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)

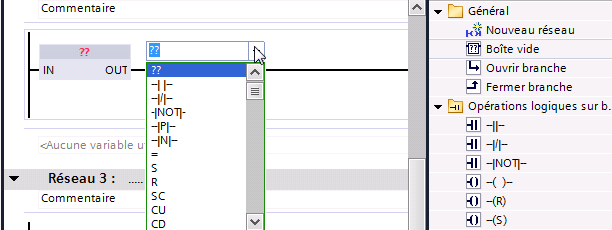
Les opérandes peuvent être saisis avec les adresses absolues ou les noms symboliques. Si la table des variables est sélectionnée (pas ouverte) dans le navigateur de projet, les variables peuvent également être insérées dans le programme à l’emplacement souhaité par glisser & déposer depuis la vue détaillée.

**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.

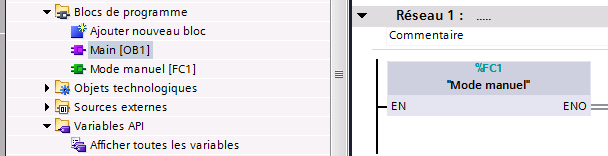
**Remarque**

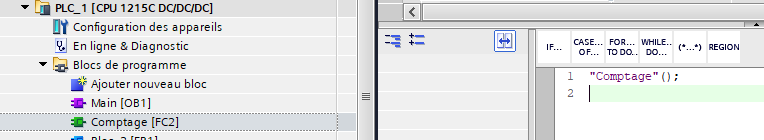
Une instruction peut également être programmée via une boîte vide. Il suffit d’insérer dans le réseau une boîte vide par glisser & déposer à partir de l’onglet « instructions de base » puis sélectionner dans la boîte vide l’instruction et le type de données que vous souhaitez programmer.



* **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

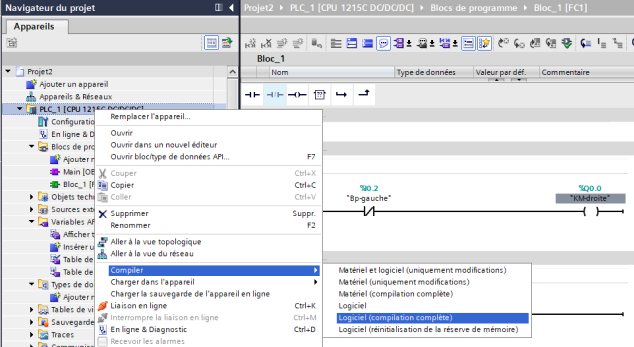




* Compiler les blocs programme

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** uniquementles blocs modifiés sont compilés.

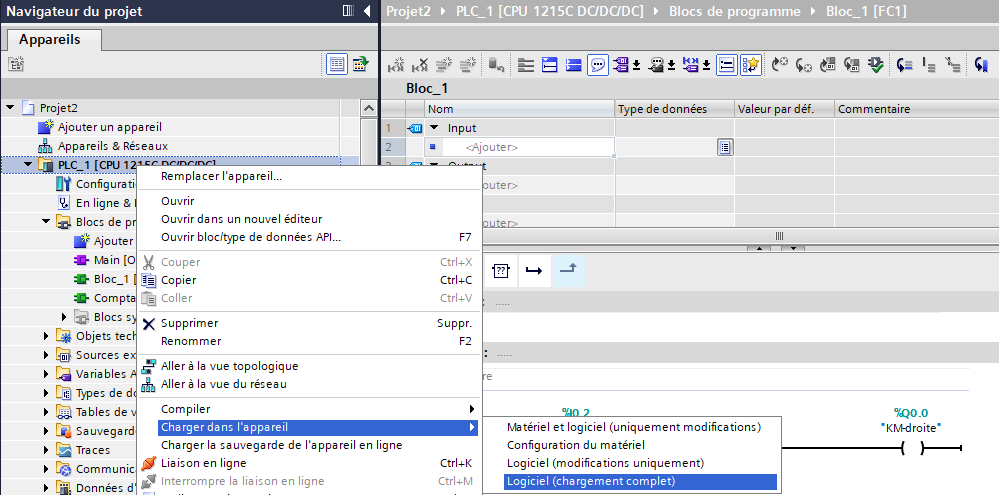


Clic droit

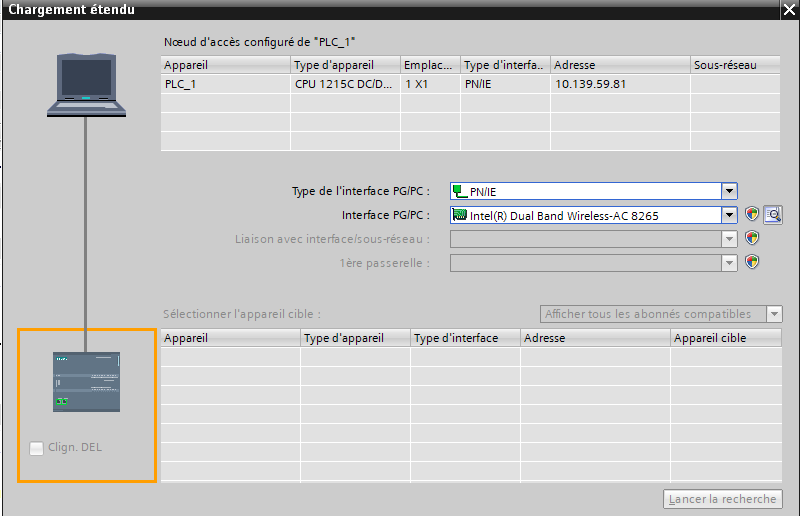
* Charger les blocs dans la CPU

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



Permet de se connecter à l’automate désiré

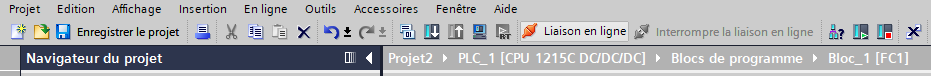


Choisir la carte Ethernet

* 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

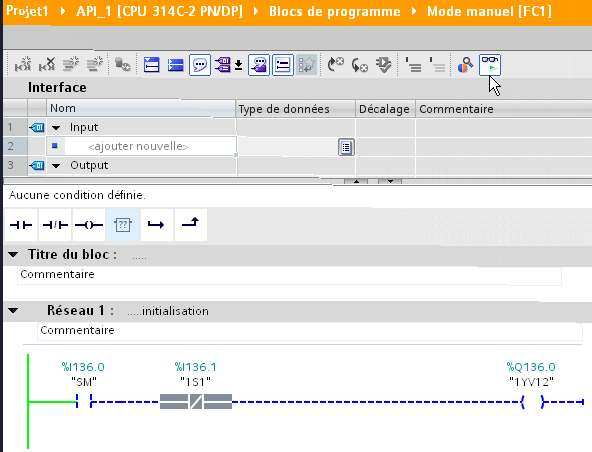


En mode test, l’état des opérandes et les éléments CONT / LOG sont représentés avec des codes couleur.

* 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

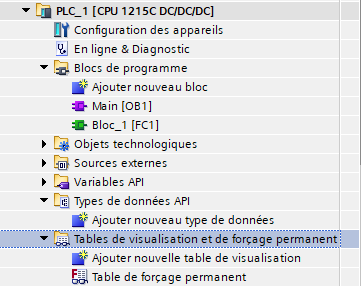


**Remarque**

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.

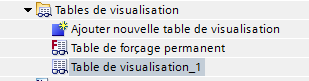
* Visualiser / forcer des variables : table de visualisation

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***

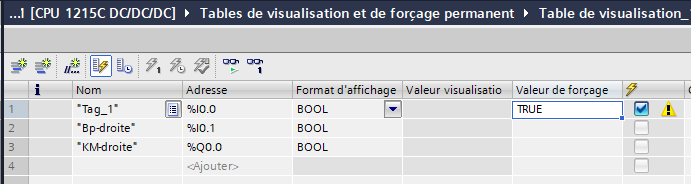
***sortie = 0 : erreur logique du programme (double affectation par exemple)***

* **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

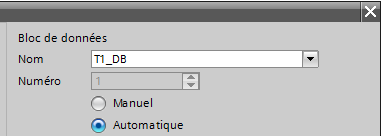
* Saisissez les adresses et valeurs de forçage souhaitées dans la table de visualisation.
* Sélectionnez l'adresse à forcer en activant la case à cocher pour le forçage dans la colonne située après la "Valeur de forçage".   
  Un triangle jaune apparaît derrière la case à cocher activée et indique que 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.



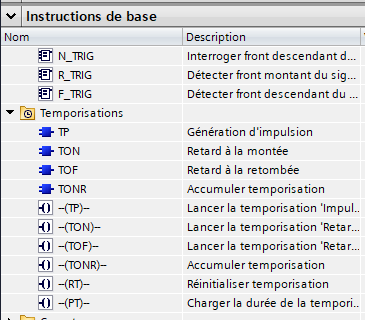
Programmation Temporisations

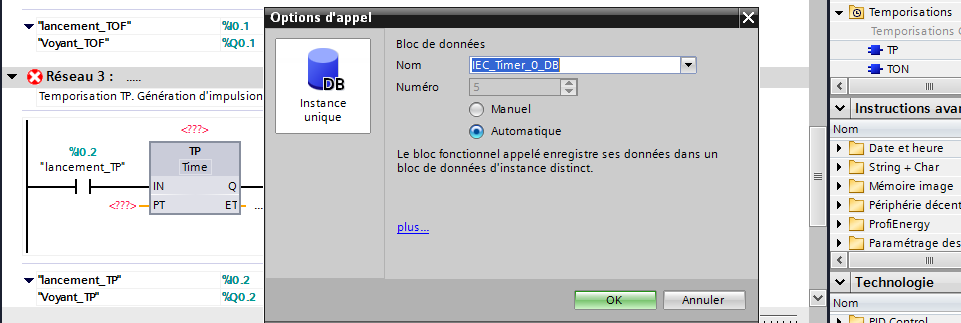
Le logiciel possède les 4 temporisations de la norme CEI

* TP (génération d’impulsions)
* TON (retard à la montée)
* TOF (retard à la remontée)
* TONR (accumuler temporisation)



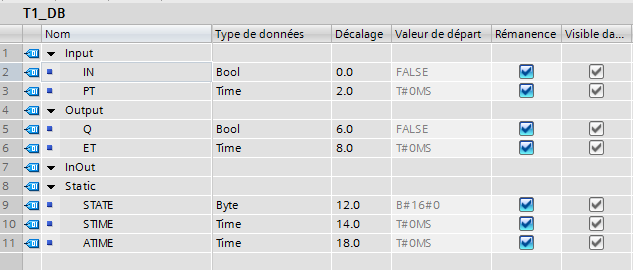
* Langage CONT

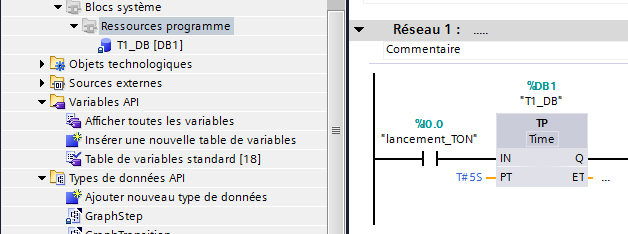
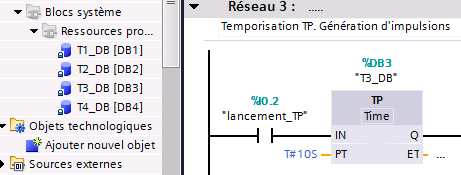


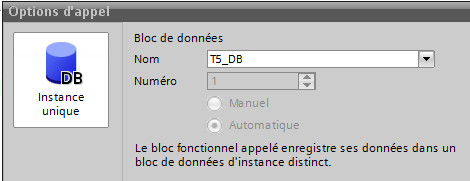


* Se positionner sur le réseau
* Choisir la temporisation désirée puis par glisser & déposer, la positionner sur le réseau
* Une fenêtre s’ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre temporisation (bien laisser \_DB à la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

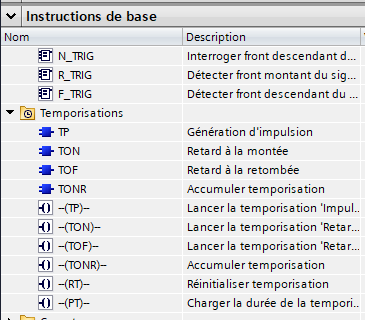
Une fois la temporisation insérée, un bloc de données apparaît dans le navigateur sous « ressources programme ». Si on clique dessus, on a accès à l’ensemble des données de la temporisation.

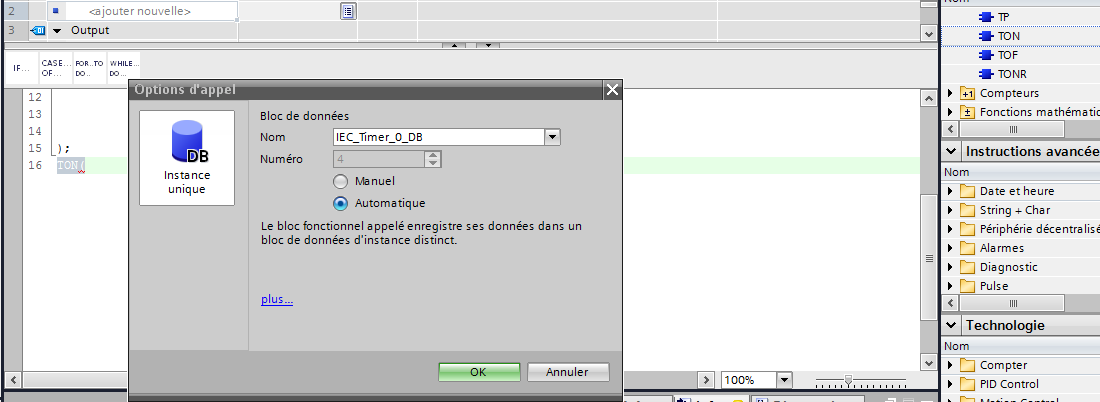






* Langage SCL

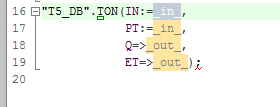




* Se placer sur une ligne
* Choisir la temporisation désirée puis par glisser & déposer, la positionner sur le réseau
* Une fenêtre s’ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre temporisation (bien laisser \_DB à la fin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

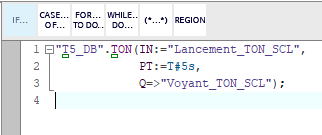
Dans la fenêtre de travail, on obtient les lignes ci-contre.

* 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 ; ne pas oublier d’effacer également la virgule qui précède.





Après renseignement des paramètres

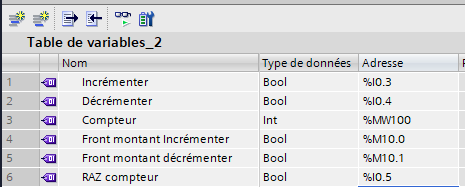


Programmation Compteurs

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

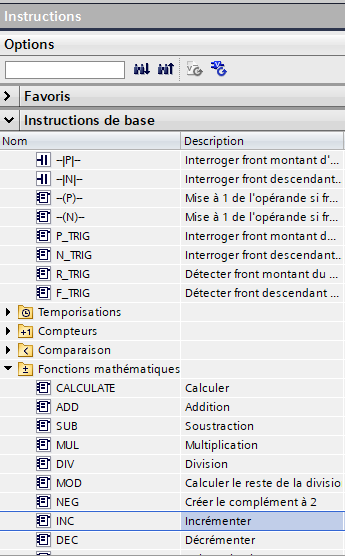
* Incrémentation d’un mot (plus souple)
* Utilisation des blocs compteurs intégrés dans le logiciel

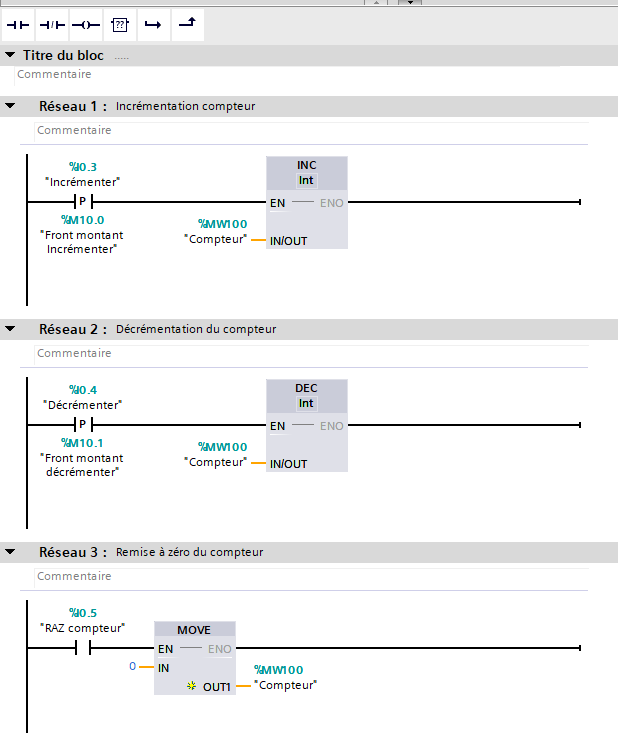
* Par incrémentation d’un mot en langage CONT



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’incrémentation et la décrémentation se font sur un front montant de la variable sinon le compteur s’incrémente ou se décrémente tant que la variable est active.

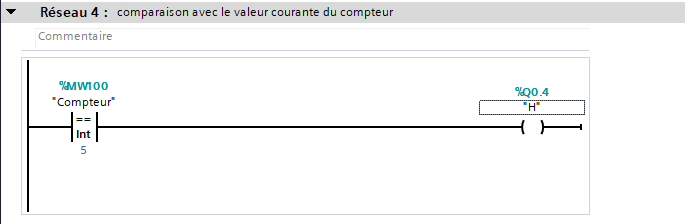
Ci-dessous le programme avec les différents réseaux. On retrouve les différents symboles dans les menus de la Task kards



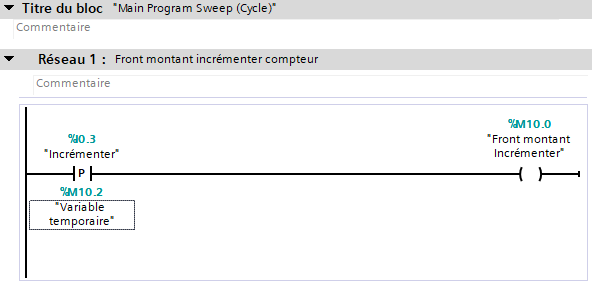








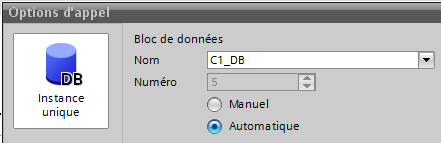
Remarque fronts montants

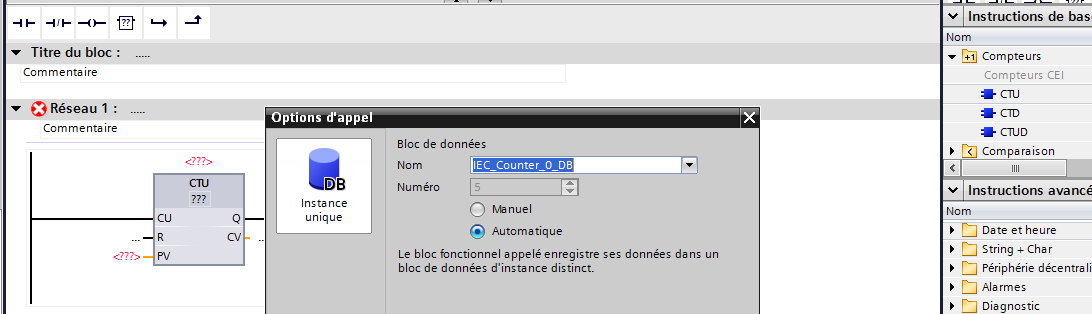


Si le front montant d’une variable est présent à différents endroits du programme, il est impératif de choisir des bits « front montant » différents.

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.

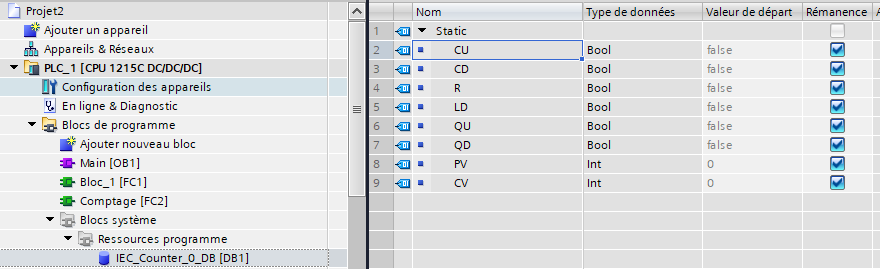
* Par utilisation des blocs compteurs intégrés en langage CONT



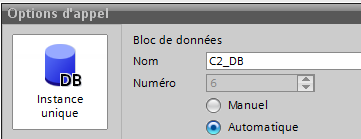


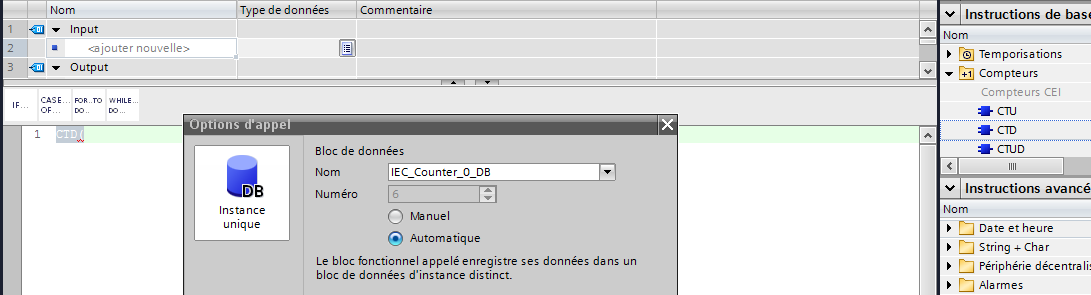
* Se positionner sur le réseau
* Choisir le compteur désiré puis par glisser & déposer, le positionner sur le réseau
* Une fenêtre s’ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre compteur (bien laisser \_DB à la fin du nom)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

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.



* Par utilisation des blocs compteurs intégrés en langage SCL

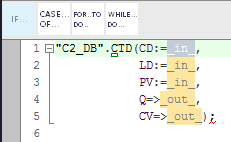




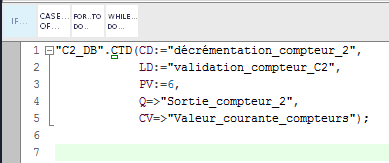
* Se placer sur une ligne
* Choisir le compteur désiré puis par glisser & déposer, le positionner sur le réseau
* Une fenêtre s’ouvre pour indiquer le bloc de données (permettant de stocker les données), modifier le nom pour indiquer le numéro de votre compteur (bien laisser \_DB à la fin du nom)

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



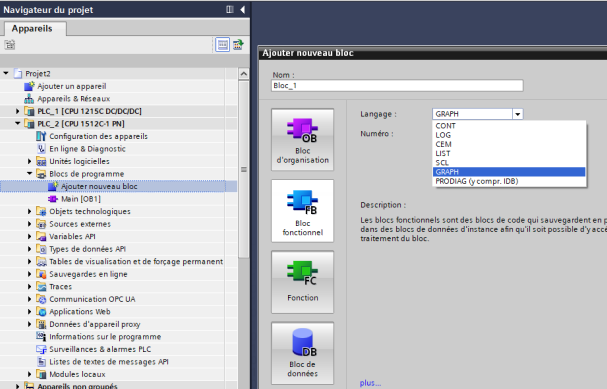
Programmation langage GRAPH

Le langage GRAPH permet de programmer directement des systèmes séquentiels représentés sous forme de grafcet. 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.

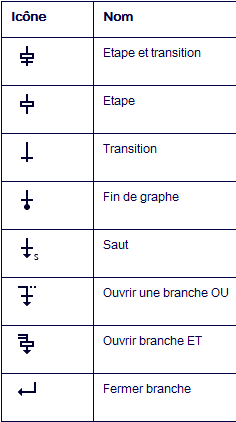
* **Ajout d’un bloc GRAPH**

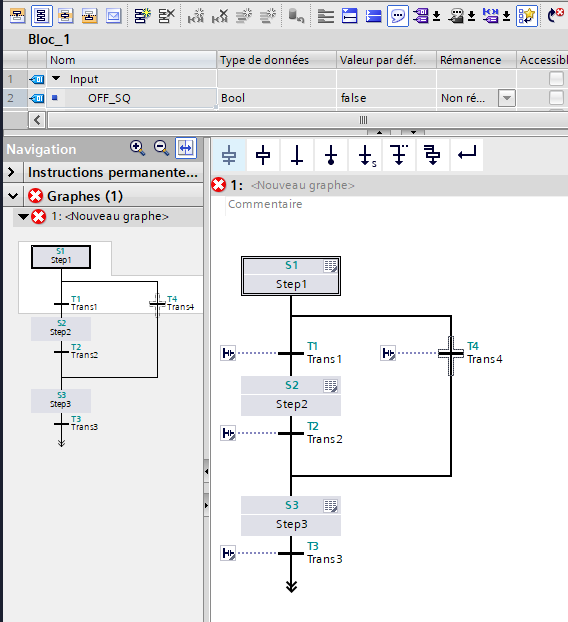


* **Vue d’ensemble de l’interface de programmation**

**Vue étape unique**

**Vue graphe**





**Glisser & Déplacer**

**les éléments**

* **Exemple de programmation**

Soit le grafcet ci-dessous à programmer en langage GRAPH

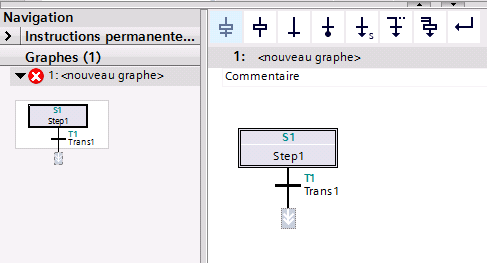
**Grafcet point de vue PC Grafcet point de vue API**



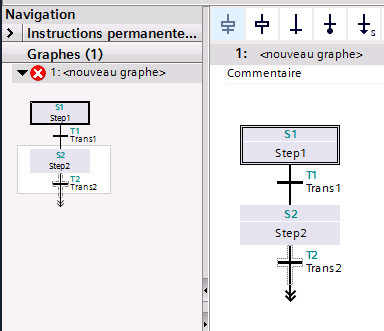


Pour la programmation du compteur C1, prendre une variable API nommée %MW100 et de type INT. Incrémenter cette variable à l’étape STEP2 ; mise à zéro de cette variable à l’étape STEP4 et tester la valeur de cette variable à l’aide d’un bloc COMPARAISON

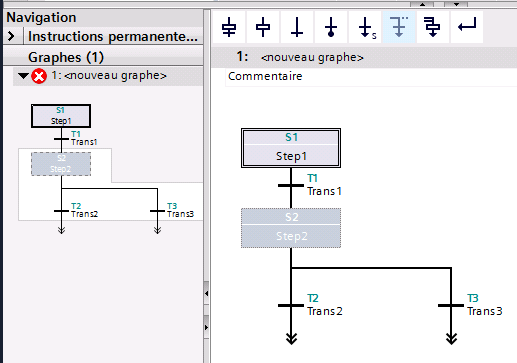
* **Ecrire la structure du graphe**



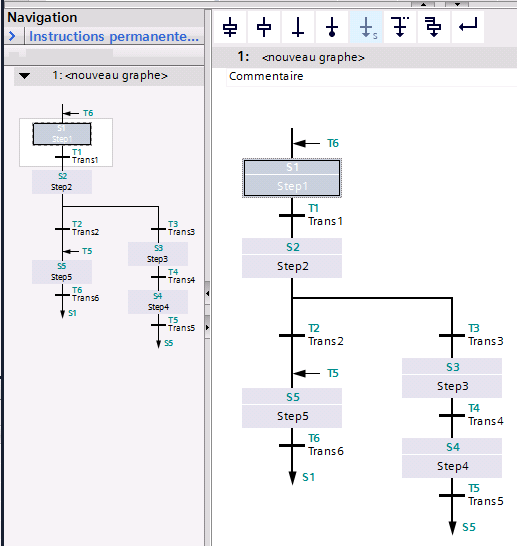
* Cliquer sur la transition T1
* Insérer une étape + transition



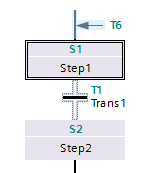
La deuxième étape et la deuxième transition sont insérées



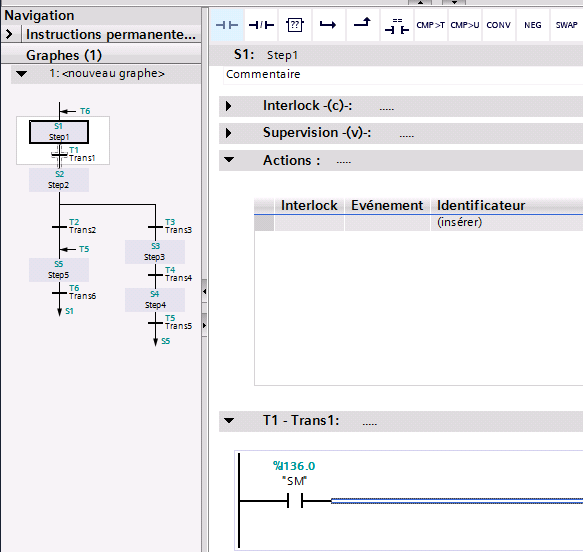
* Cliquer sur **l’étape 2**
* Insérer une **branche OU**

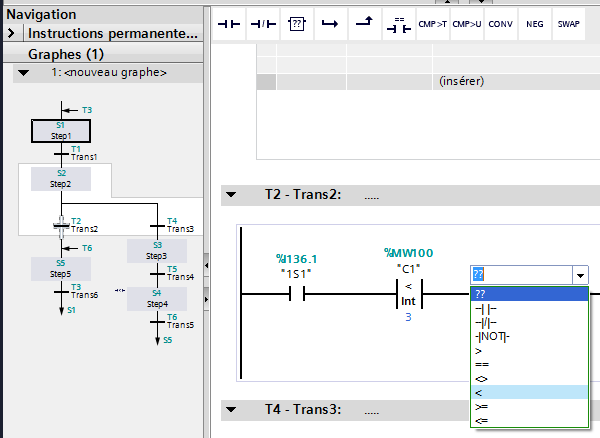


* Continuer à écrire la structure du grafcet.
* Pour « boucler » le grafcet cliquer sur la **transition T5.**
* **Insérer un saut.**
* Positionner le pointeur de la souris sur **l’étape destinataire S5** puis valider.
* Faire de même pour la liaison Transition T6 Etape S1
* **Ecrire les réceptivités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



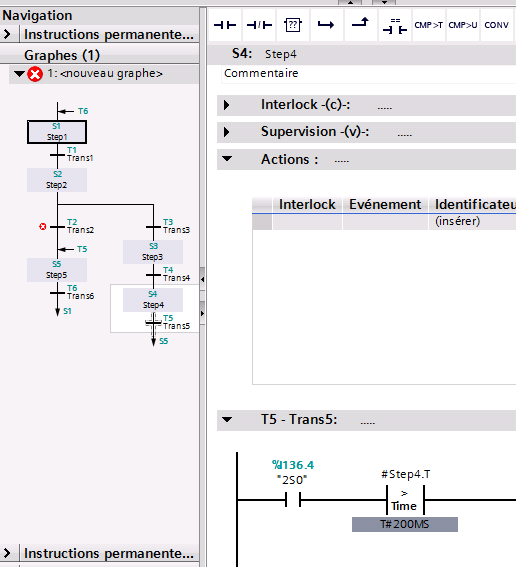
* Cliquer 2 fois sur **la transition**.
* Insérer **un contact** par glisser&déplacer et lui donner une adresse.





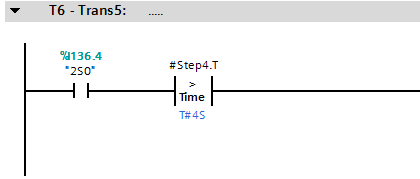
**Transition entre l’étape 2 et l’étape 5**

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

****

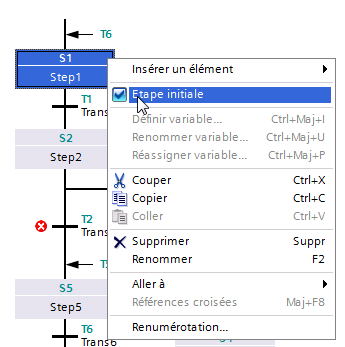
**Transition entre l’étape 3 et l’étape 5**

* 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

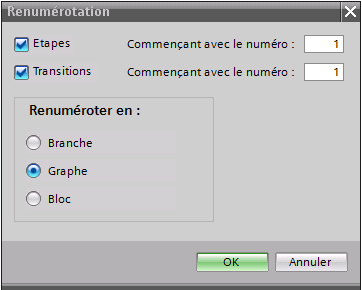


**Continuer de la même manière afin d’écrire toutes les réceptivités associées aux transitions**

* **Modifier les propriétés d’une étape**



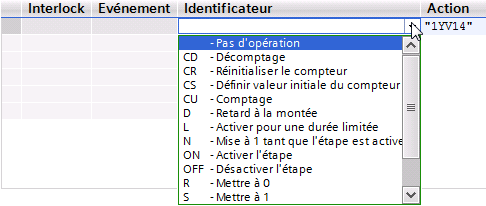
* Cliquer sur le repère et le numéro d’étape pour les modifier
* Clique droit pour définir une étape initiale
* Clique droit pour renuméroter l’ensemble des étapes ou des transitions

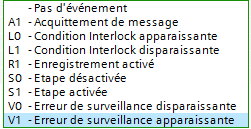


* **Ecrire les actions associées aux étapes**

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).

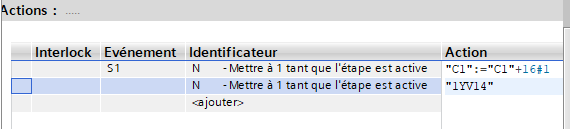




****

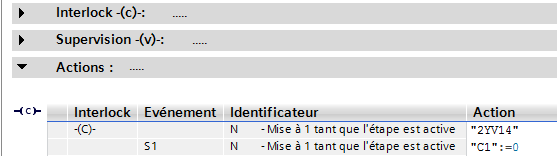
**Ecriture de l’action associée à l’étape 2**

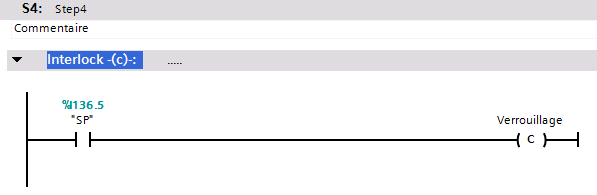
* Cliquer deux fois sur l’étape pour ouvrir la fenêtre des actions
* Choisir l‘évènement (front montant) **S1**
* Choisir l’identificateur **N** (l’action est vraie tant que l’étape est active)
* Ecrire l’action **C1 := C1 +1**
* Choisir l’identificateur **N** (l’action est vraie tant que l’étape est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



**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





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

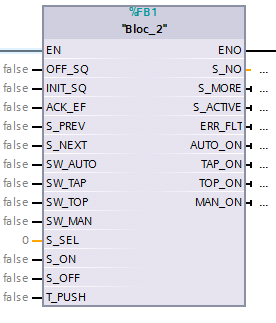
* **Paramètres d’appel du bloc**

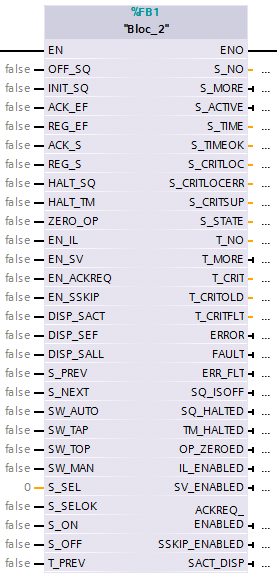
Comme pour un bloc FC, le bloc FB doit être appelé dans le bloc OB1. Lorsqu’on appelle un bloc FB, le logiciel crée également un DB d’instance servant de zone mémoire.

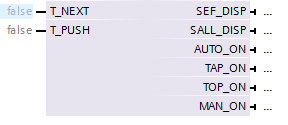
Pour appeler le bloc FB1, vous disposez des deux possibilités de sélection suivantes :

Paramètres d'interface par défaut

Paramètres d'interface maximaux







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'acquitter 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**".

