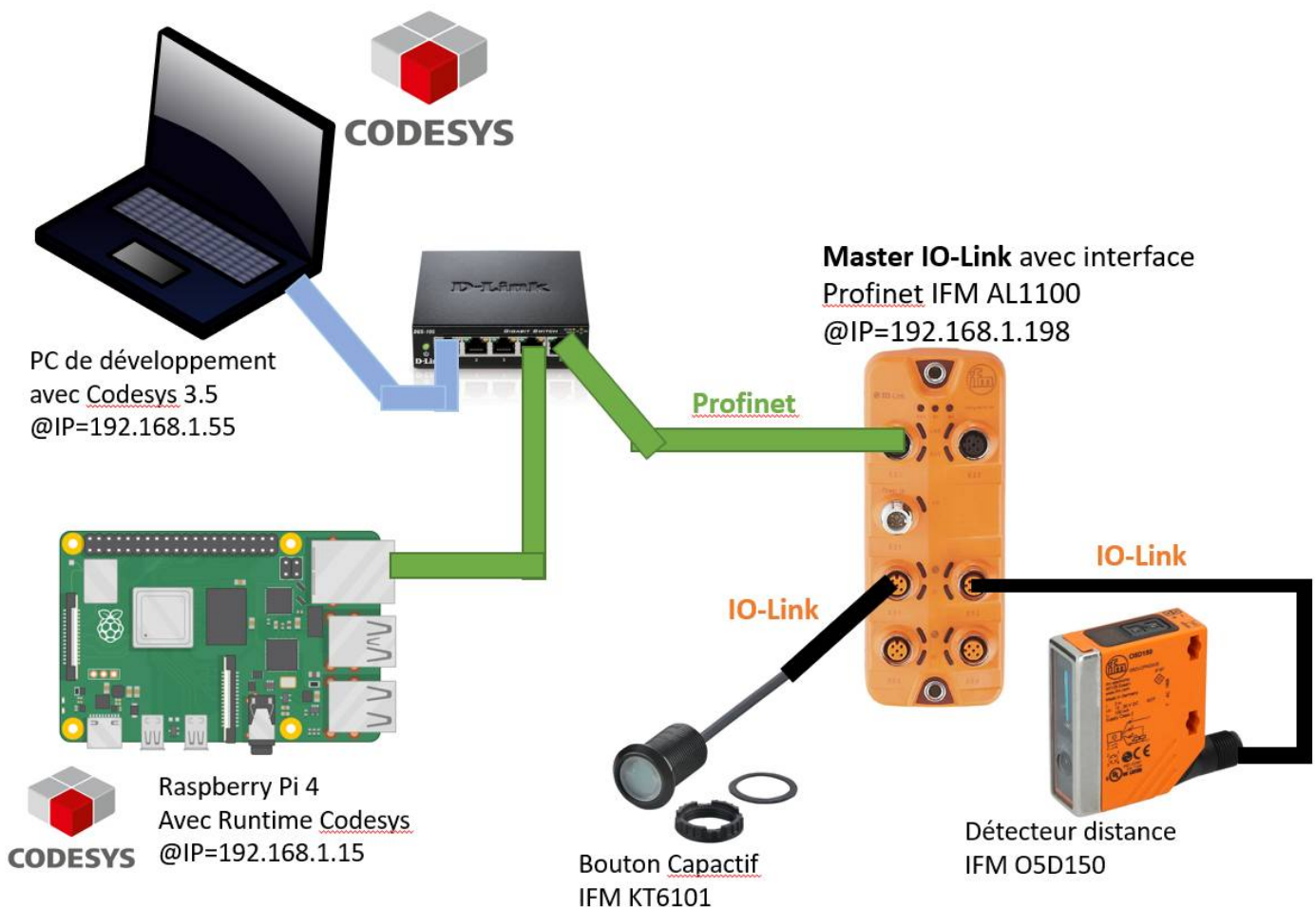


# Master IO-Link sur Raspberry Pi

## Master IO-Link Raspberry Pi

Cet article fait suite à la communication entre TwinCAT 3 et un Master IO-Link Profinet. Il est également possible de connecter un Master IO-Link Profinet sur un Raspberry Pi 4 avec le Runtime Codesys. Le schéma du montage utilisé est représenté ci-dessous :



- Le Raspberry Pi va constituer notre PLC. Le Runtime Codesys est installé dessus. Il va gérer la communication Profinet avec le Master IO-Link Profinet AL1100 d'IFM.
  - @IP du Raspberry Pi est 192. 168. 1. 15
- Le Master IO-Link Profinet AL1100 d'IFM va être associé à deux capteurs IO-Link



- Détecteur de distance O5D150
- Bouton capacitif KT6101
- @IP du Master IO-Link est `192. 168. 1. 198`
- Le PC de développement va permettre de générer le code PLC et le transférer sur le Raspberry Pi. Contrairement à TwinCAT où le Runtime est exécuté sur le PC, dans cette application, le Runtime est sur le Raspberry Pi.
  - @IP du PC de dev : `192. 168. 1. 55`

### Prérequis :

- Codesys 3.5 installé (et fonctionnel) sur un PC de développement -> cf article
- Runtime Codesys installé sur un Raspberry Pi -> cf article

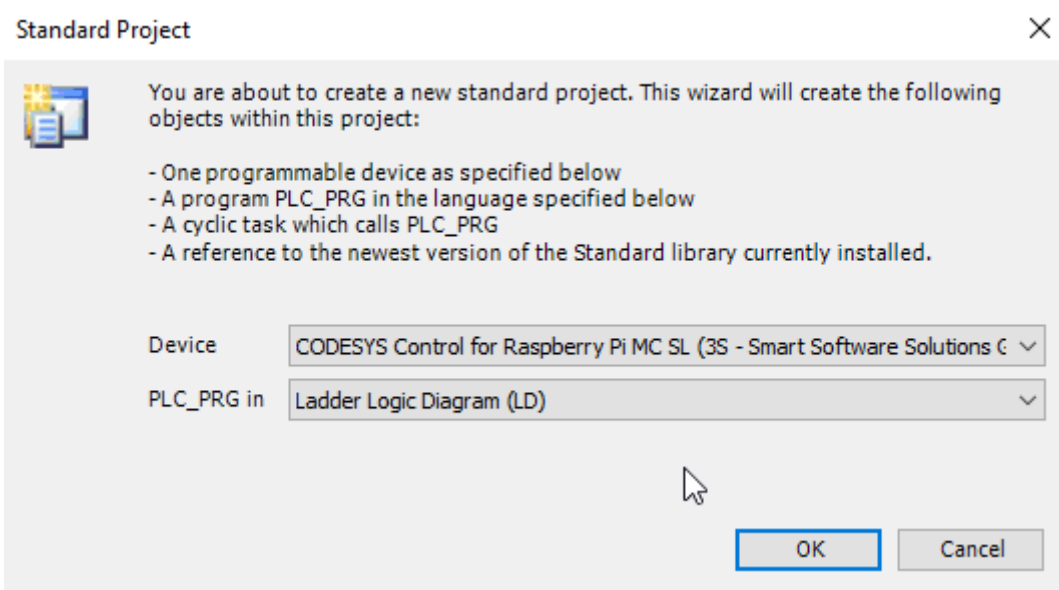
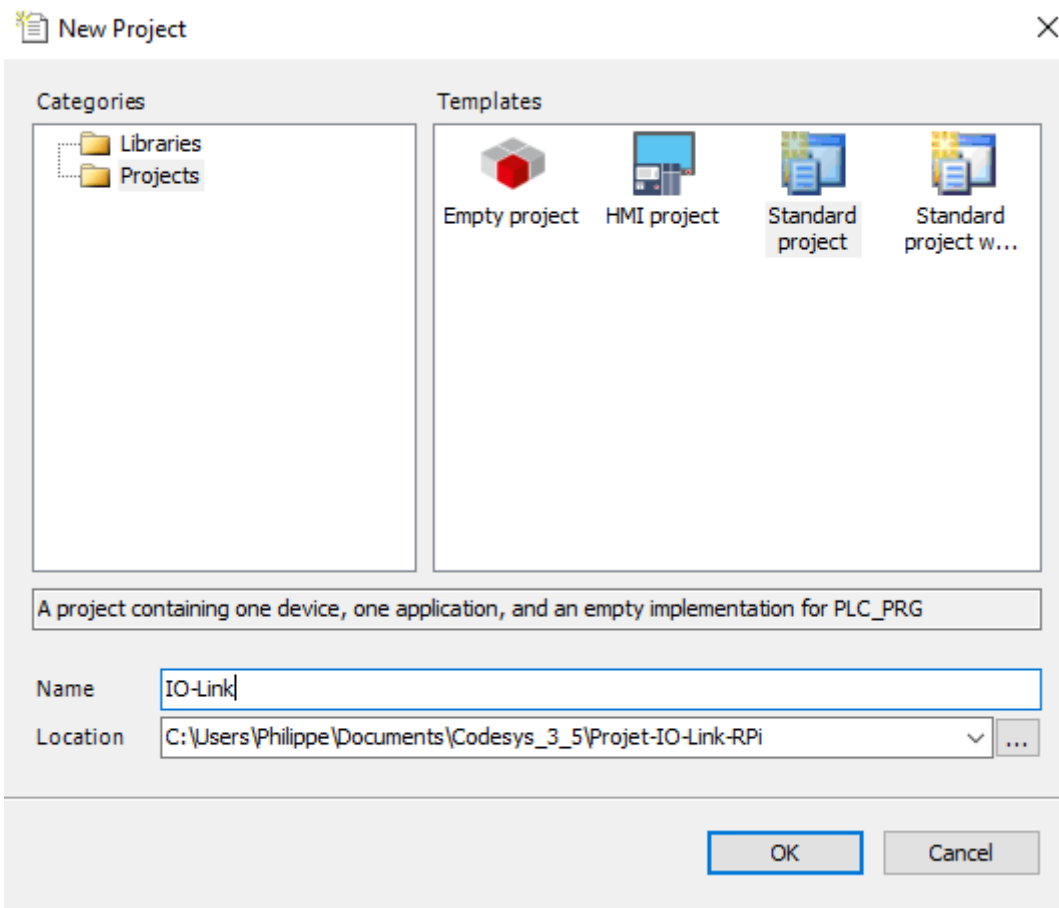
Il est également nécessaire d'avoir lu l'article sur la communication entre le Master IO-Link Profinet et TwinCAT car plusieurs aspects vont se retrouver ici :

- les fonctions en langage ST pour le décodage des capteurs
- l'association des variables

# Projet Codesys

- Créer un Standard New Project
- Choisir pour le Device Codesys Control for Raspberry PI MC SL
- Choisir Ladder Logic pour PLC\_PRG



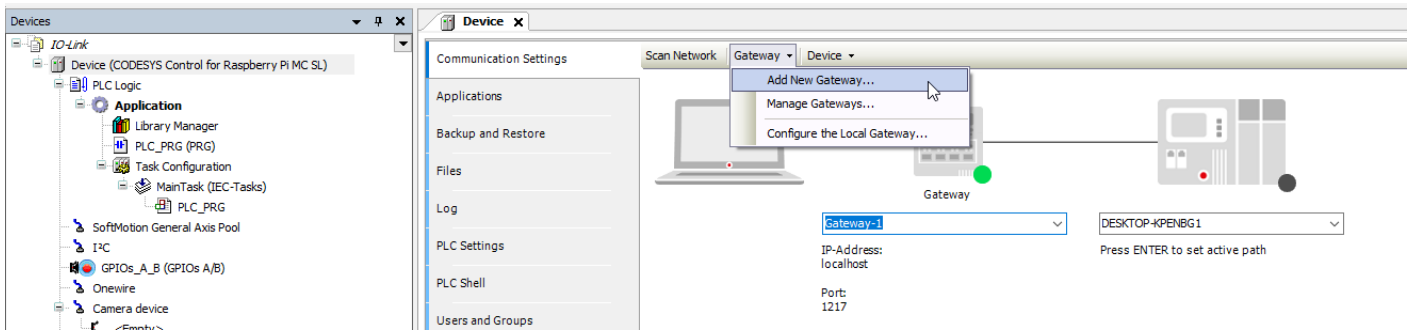


## Gateway et Raspberry Pi

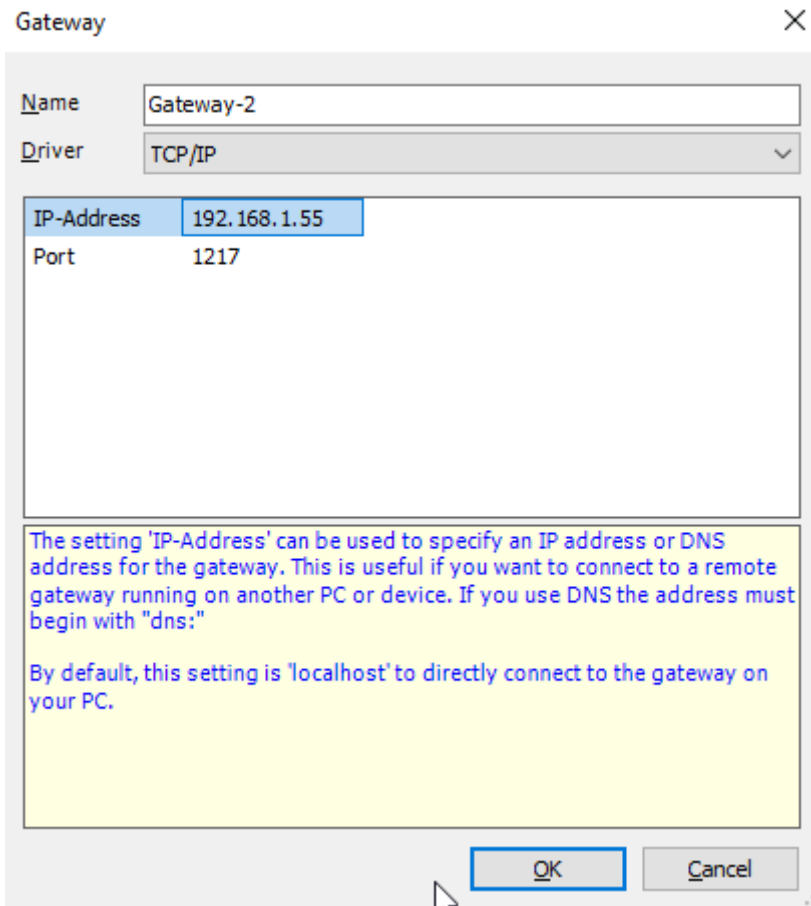
En double cliquant sur Device, dans les Communications Settings:

- Gateway -> Add New Gateway





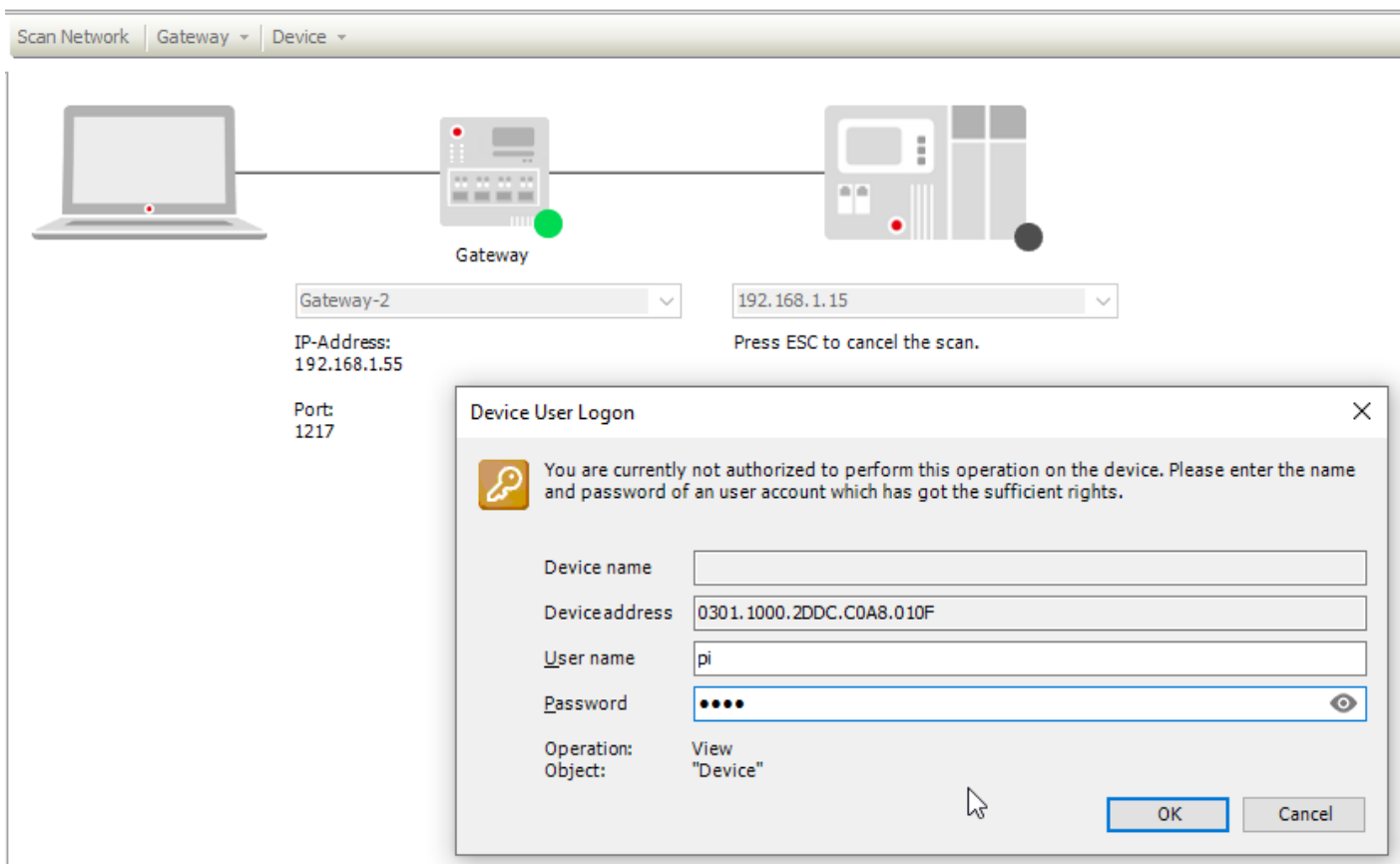
- Placer l'adresse IP du PC développement (@IP=192.168.1.55 dans mon cas)



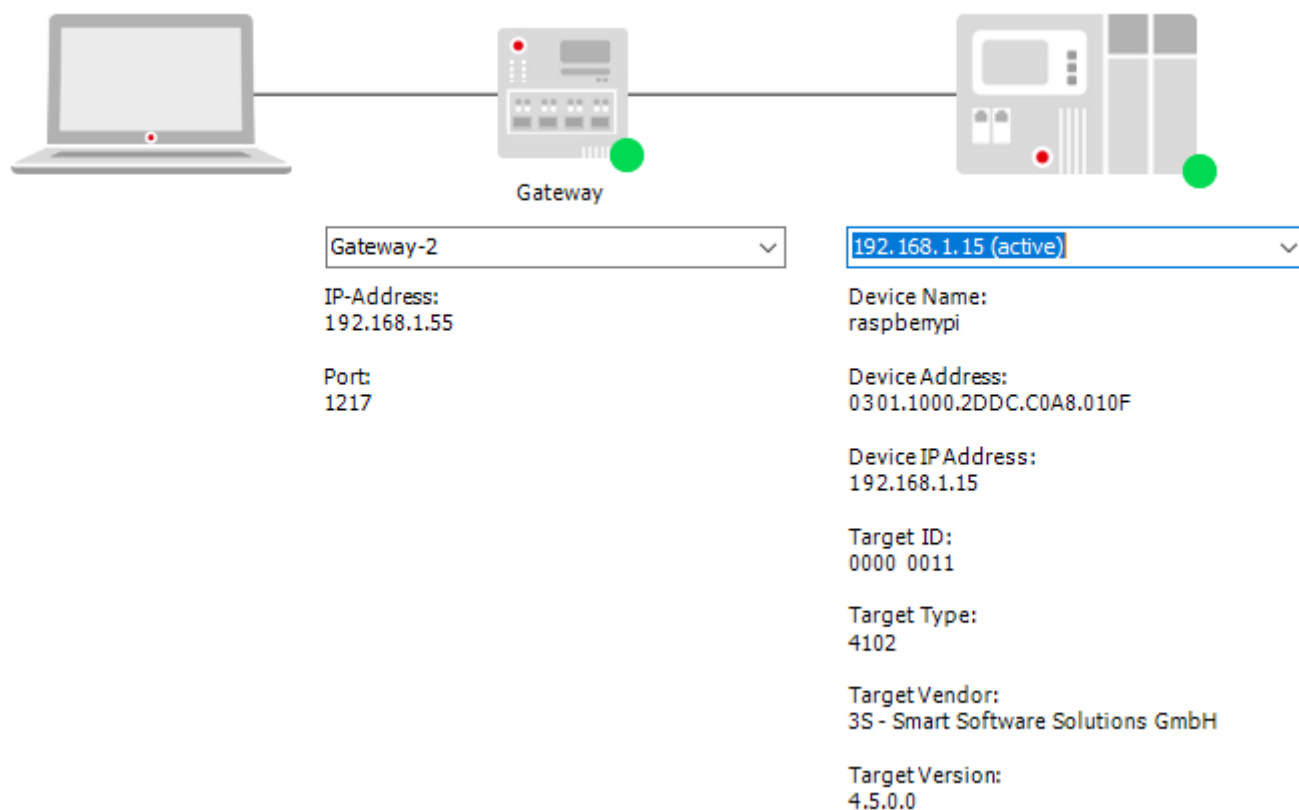
À droite de la Gateway,

- placer l'adresse IP du Raspberry Pi
- en appuyant sur Entrée, un fenêtr de Device User Logon apparaît :
  - mettre le User Name du Raspberry pi -> pi
  - mettre le Password -> 3.14 dans mon cas





La liaison avec le Raspberry Pi doit passer au vert et indiquer (active)



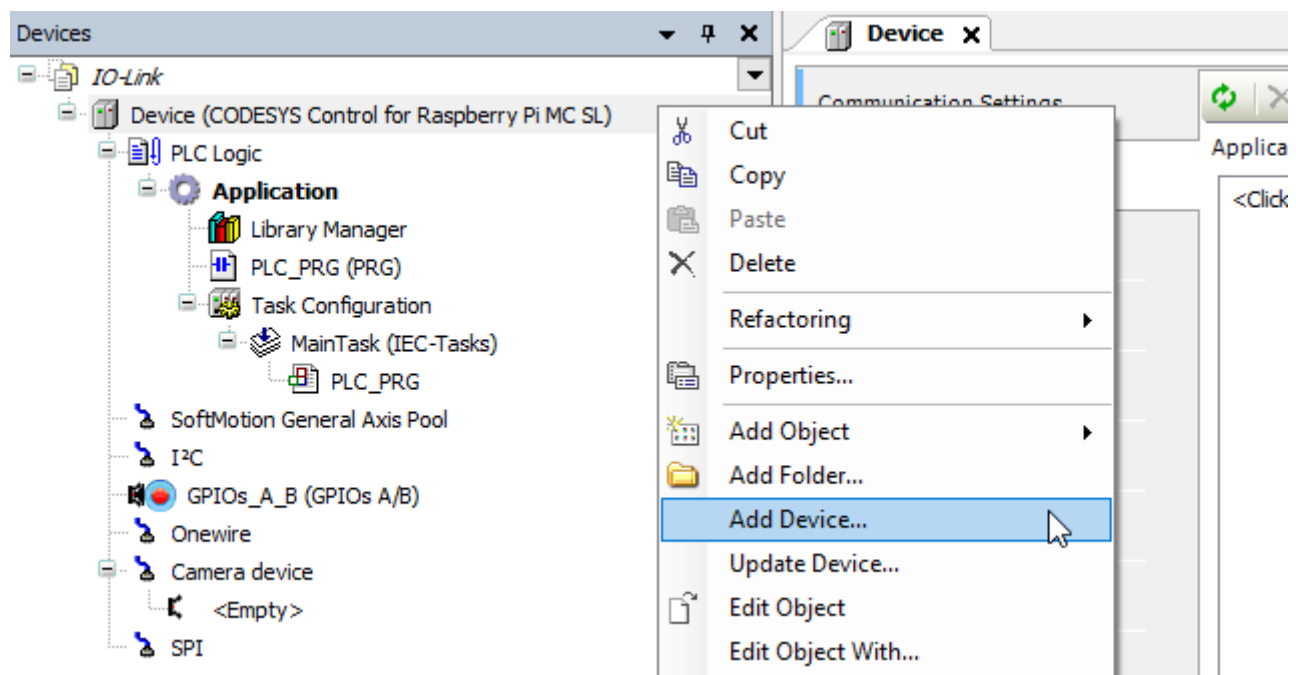
Si tout est au vert, on peut passer à la liaison Profinet



# Configuration de la liaison Profinet

Faire un clic droit sur Device :

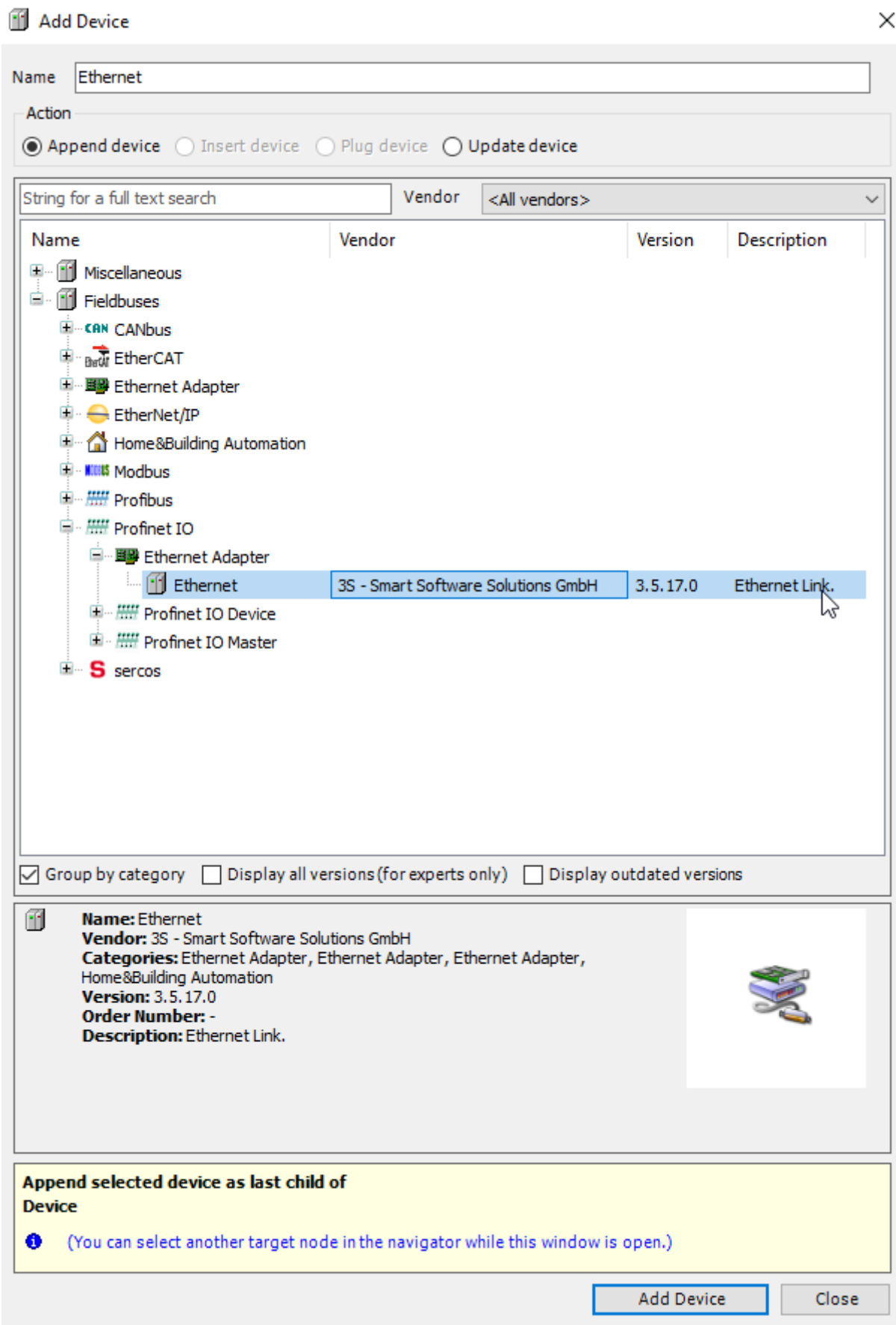
- Add Device



Dans l'arborescence Profinet IO :

- Choisir Ethernet
- et faire Add Device

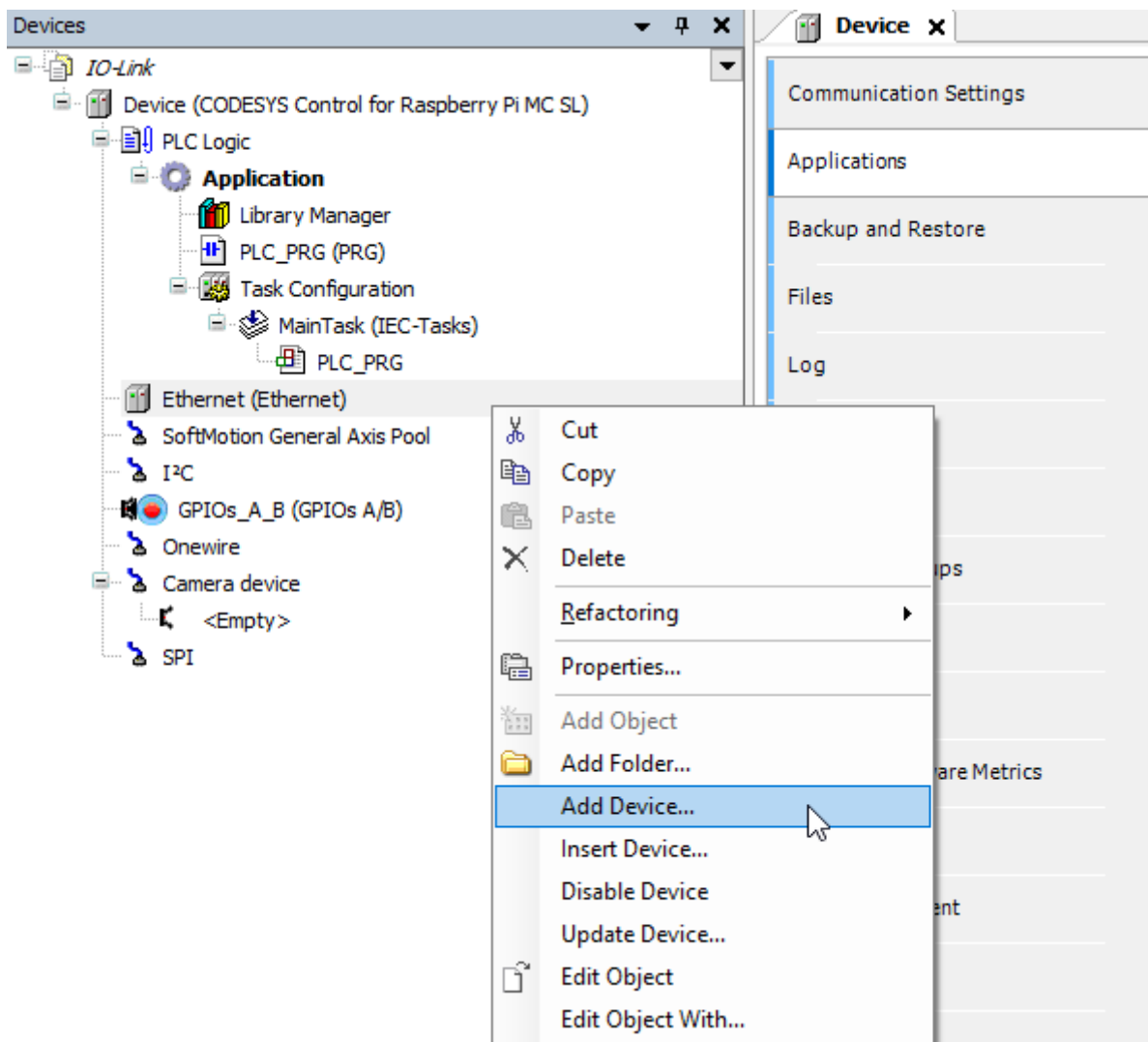




Sur Ethernet(Ethernet), faire un clic droit :

- Add Device

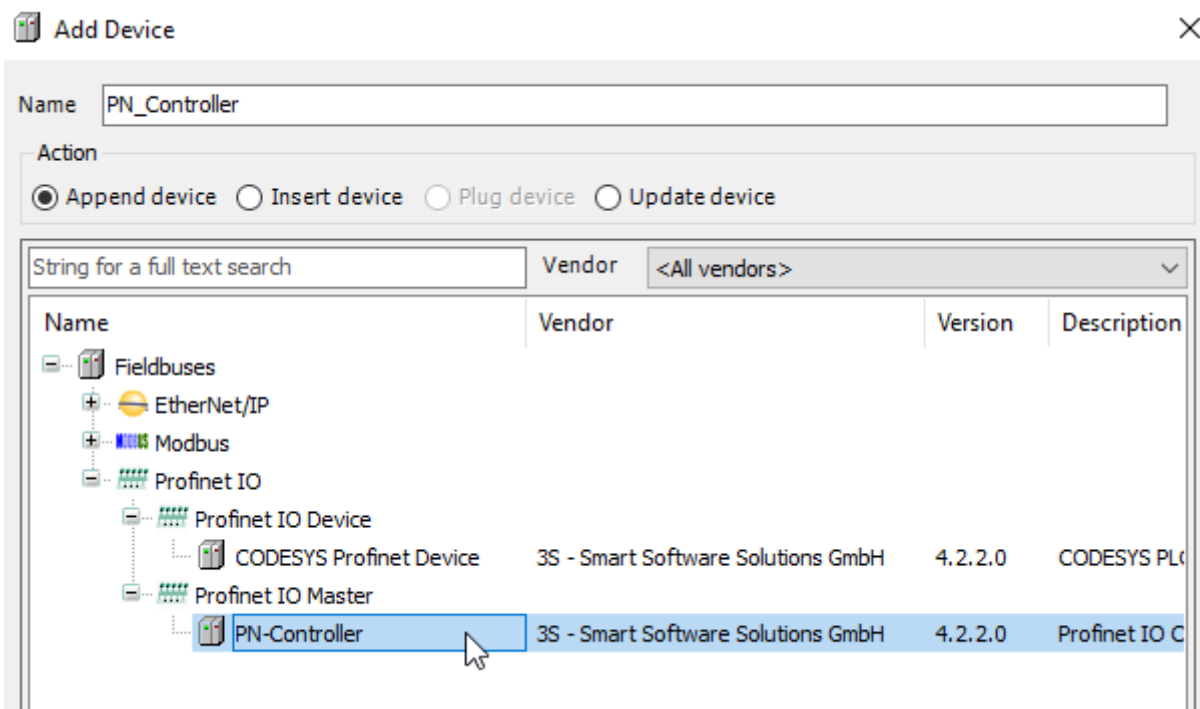




Dans l'arborescence Profinet IO Master :

- Choisir PN-Controller



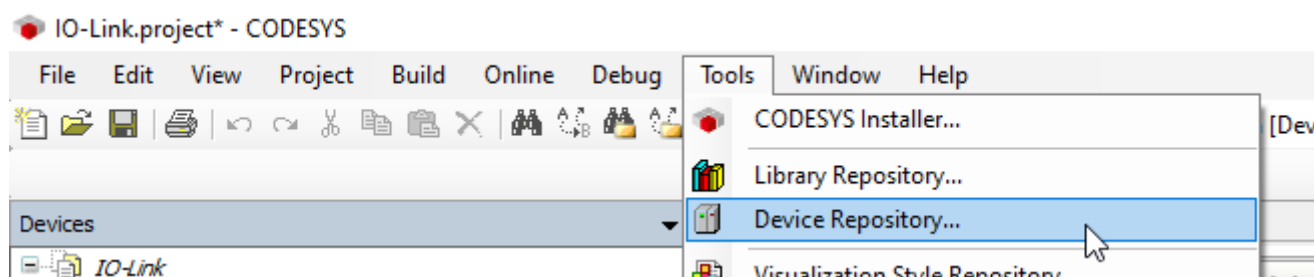


## GSDML du Master IO-Link Profinet d'IFM

Comme pour TIA Portal, le Master IO-Link Profinet n'est pas installée de base dans Codesys. Il est nécessaire d'ajouter le fichier GSDML correspondant dans le Device Repository.

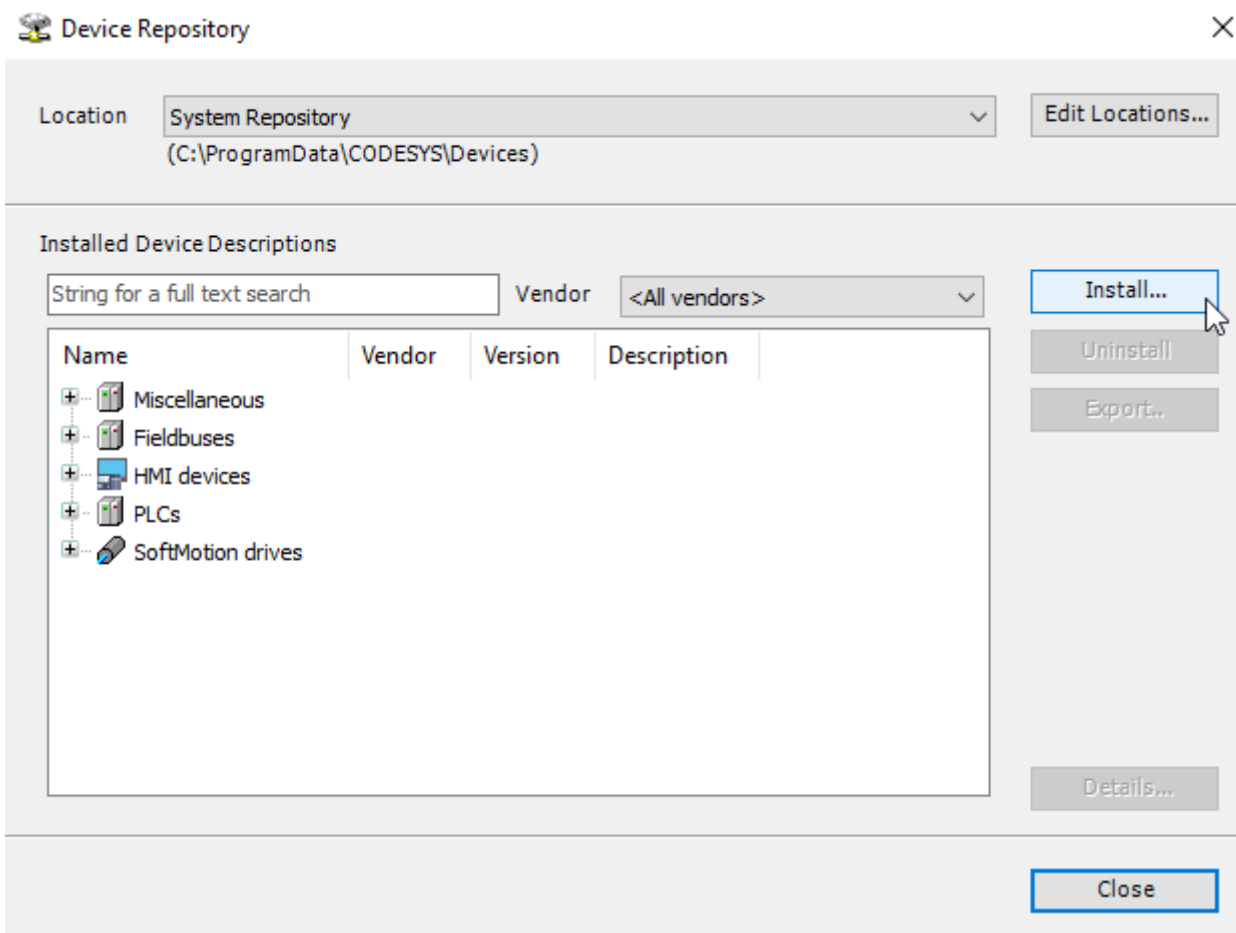
Dans l'onglet Tools, faire :

- Device Repository



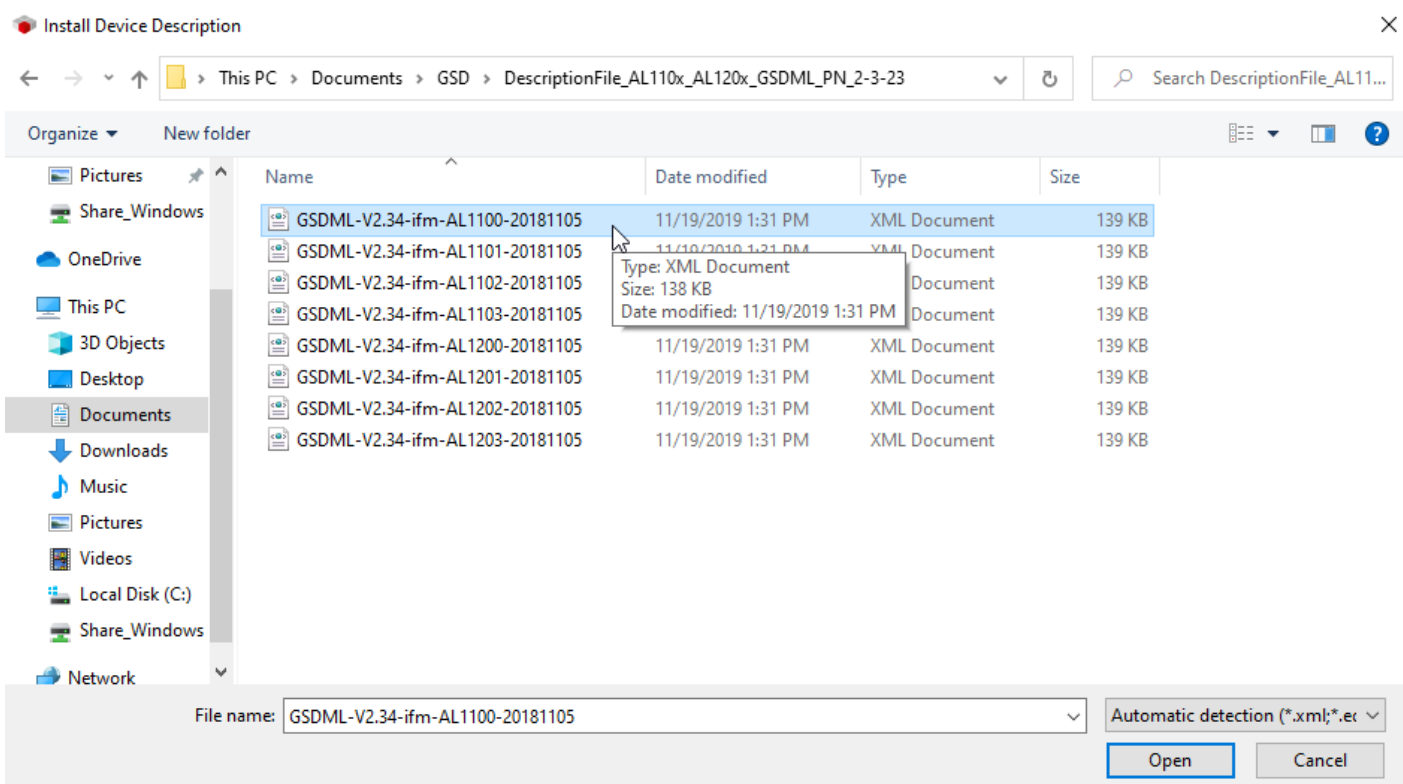
Cliquer sur Install





et sélectionner le GSDML correspondant à votre Master IO-Link. Pour rappel, le fichier GSDML du Master IO-Link Profinet AL1100 se télécharge directement chez IFM.

- après avoir sélectionné le fichier, faire open.





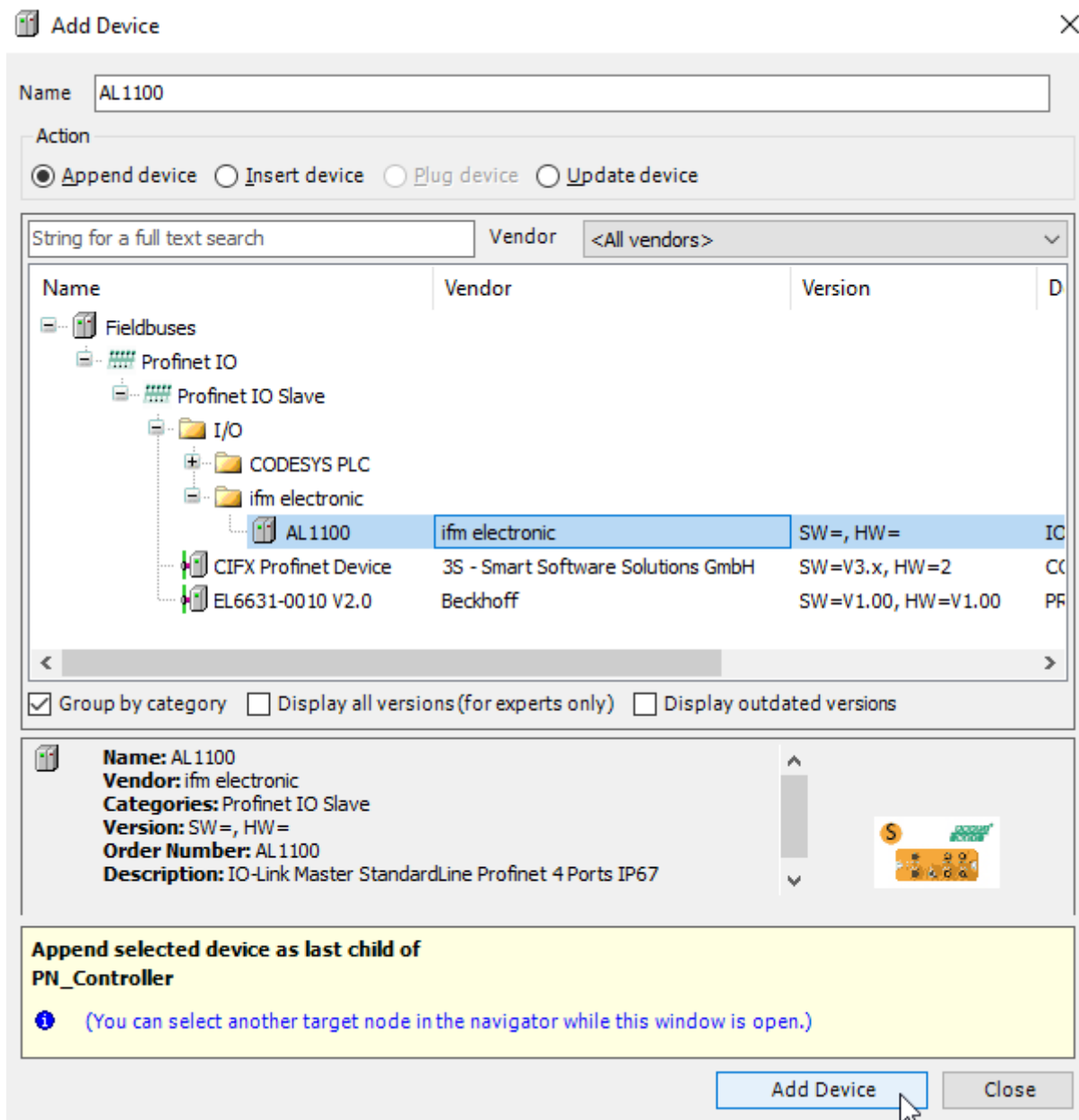
# Ajout du Master IO-Link

En cliquant sur PN\_Controller (PN-Controller), faire

- Add Device

Dans l'arborescence Profinet IO -> IO -> ifm electronic, choisir

- AL1100, ifm electronic
- faire Add Device

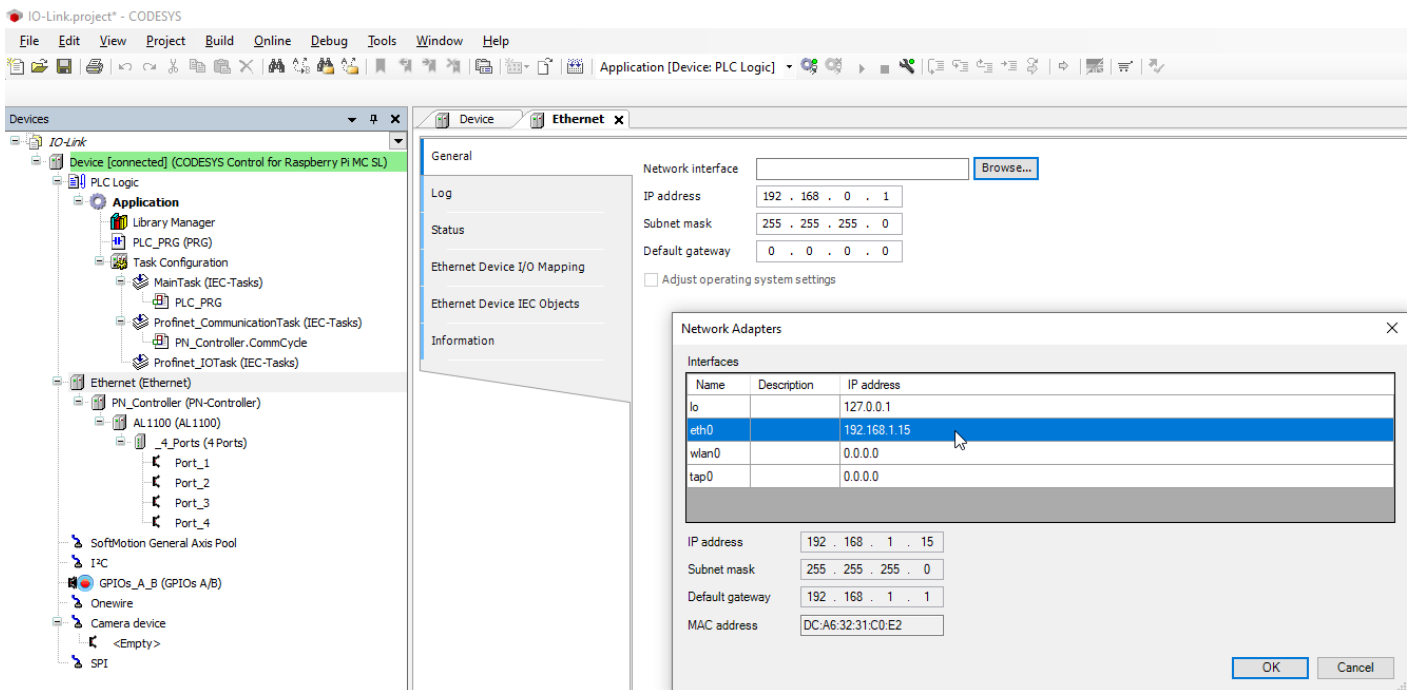


## Configuration Profinet et IO-Link

Double cliquer sur Ethernet (Ethernet)

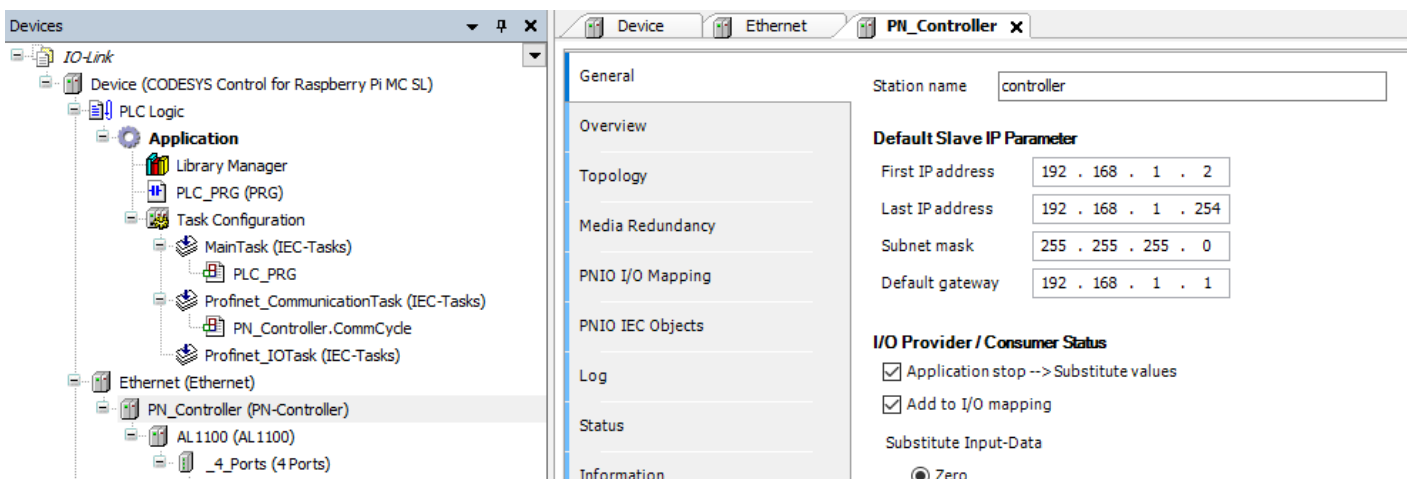


- Dans Network interface, faire Browse
- une fenêtre Network Adapters apparaît
  - choisir l'interface `eth0` qui correspond à l'interface ethernet du Raspberry Pi



Double cliquer sur PN\_Controller et modifier la plage d'adresses pour les Slave

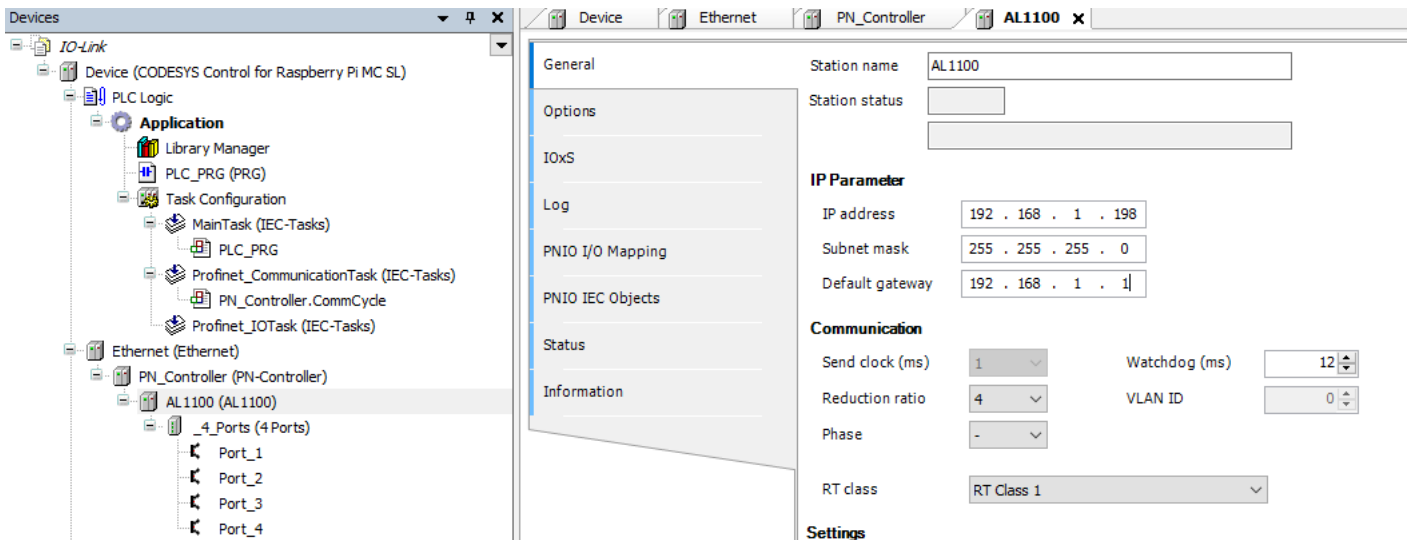
- First IP Address : 192.168.1.2
- Last IP Address : 192.168.1.254
- Subnet Mask : 255.255.255.0
- Default Gateway : 192.168.1.1



Double cliquer sur AL1100 (AL1100) et modifier les IP Parameter

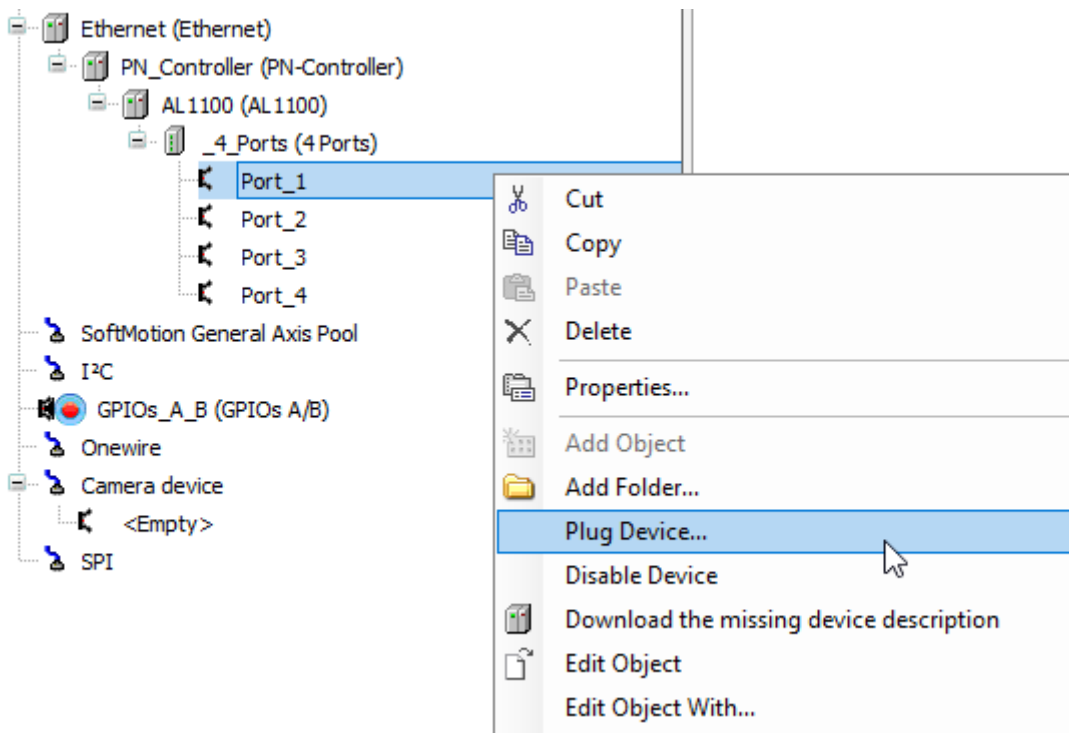
- IP Address : 192.168.1.198
- Subnet Mask : 255.255.255.0
- Default Gateway : 192.168.1.1





Dans \_4\_Ports (4 Ports) , sur le Port\_1, faire

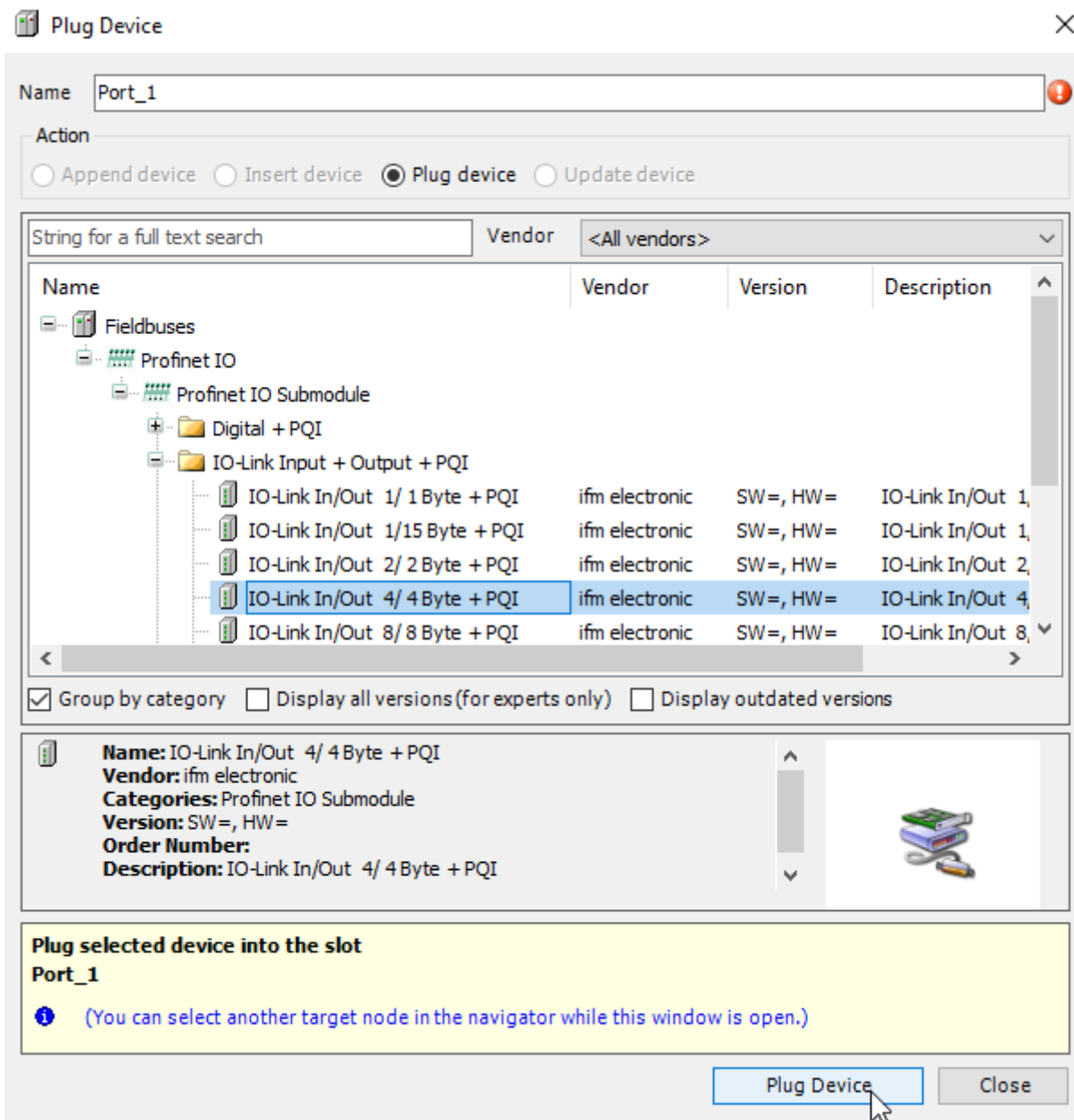
- Plug Device



Comme nous avons branché le bouton capacitif KT6101 sur le port 1, il faut lui associer la taille de IO-Link Input + Output adapté au message. Dans notre cas, choisir :

- IO-Link In/Out 4/4 Byte + PQI
- et faire Plug Device

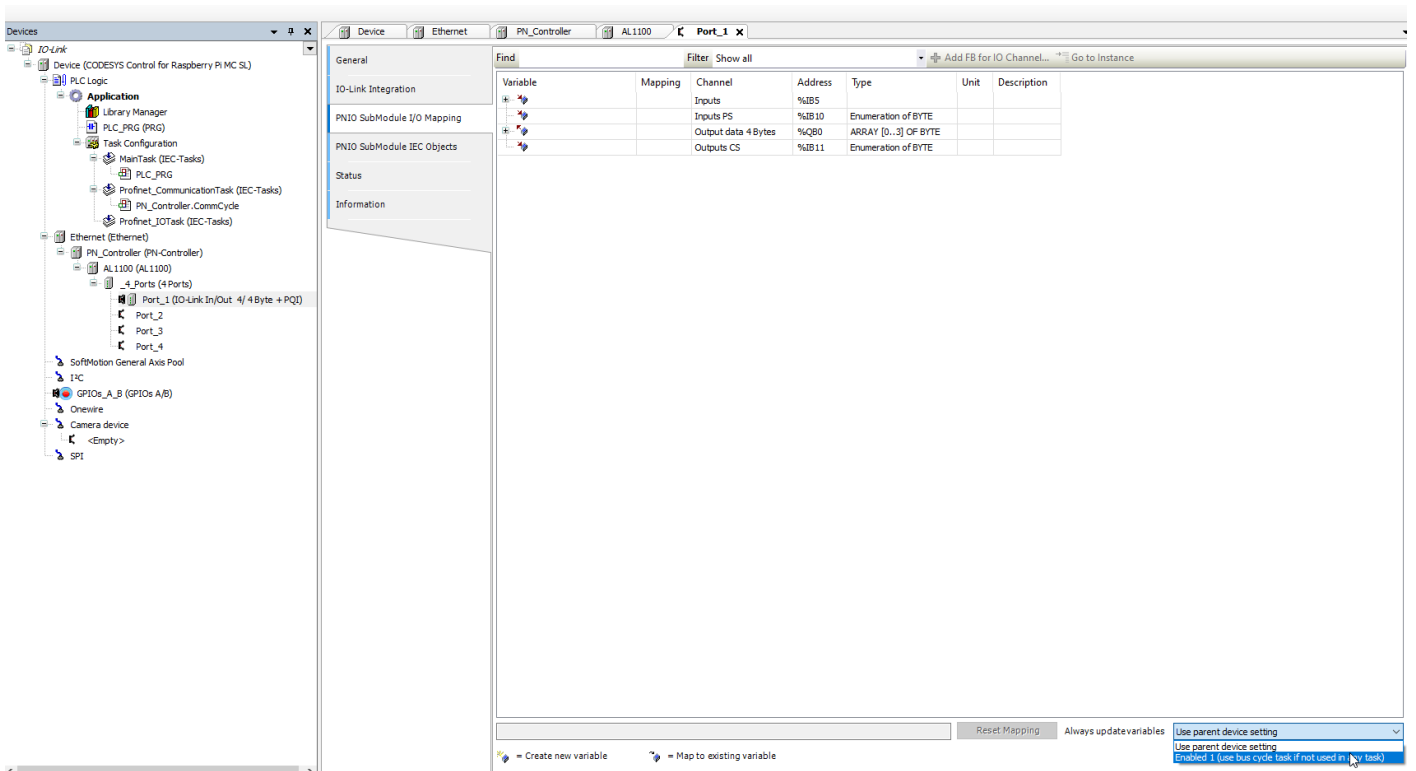




Avant d'aller plus loin, nous allons tout de suite configurer le rafraichissement automatique des données :

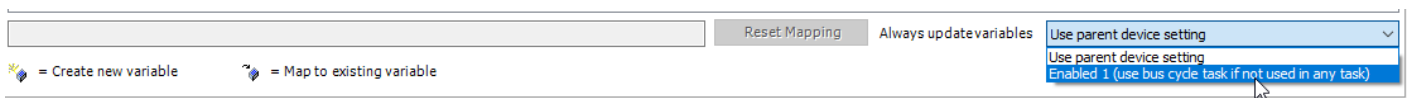
En double cliquant sur Port 1, en bas à droite, dans le champs Always Update Variables :





Faire :

- Enabled 1 (uses bus cycle task if not used in any task)



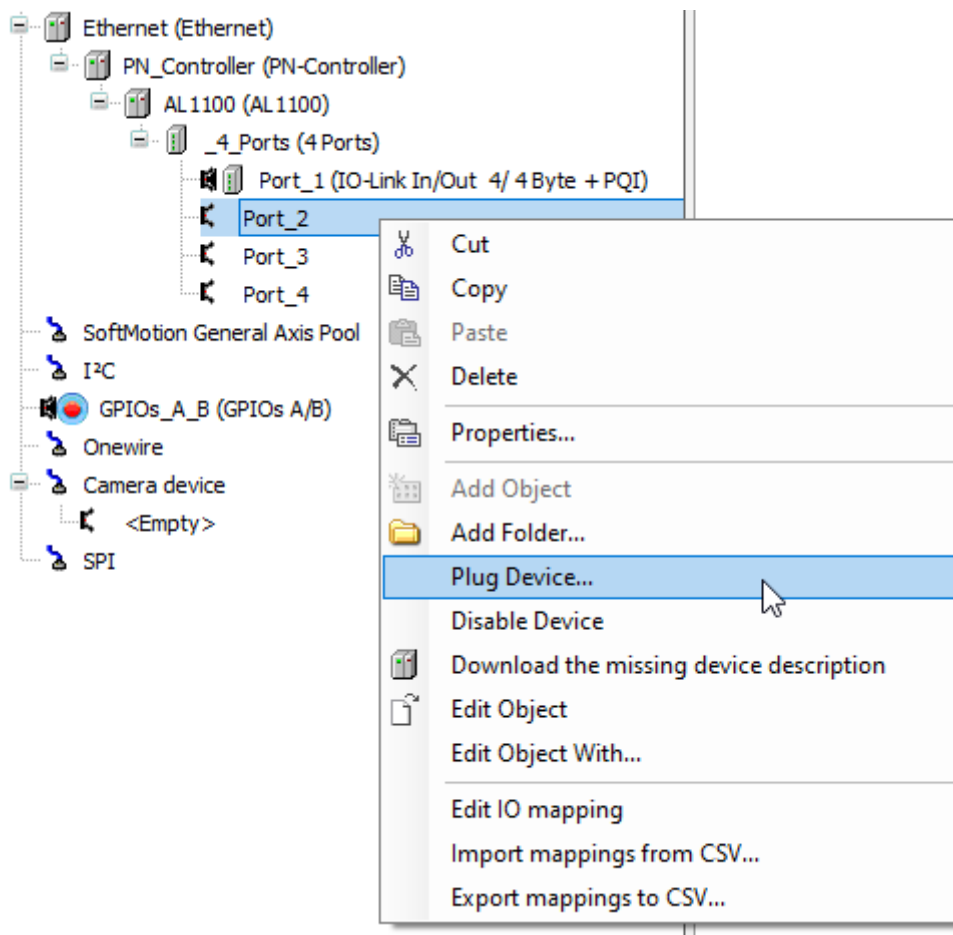
Vérifier que cela soit bien pris en compte sinon les valeurs des capteurs n'apparaîtrons pas !



On va procéder de manière analogue pour Port\_2 :

- on clic sur Plug Device

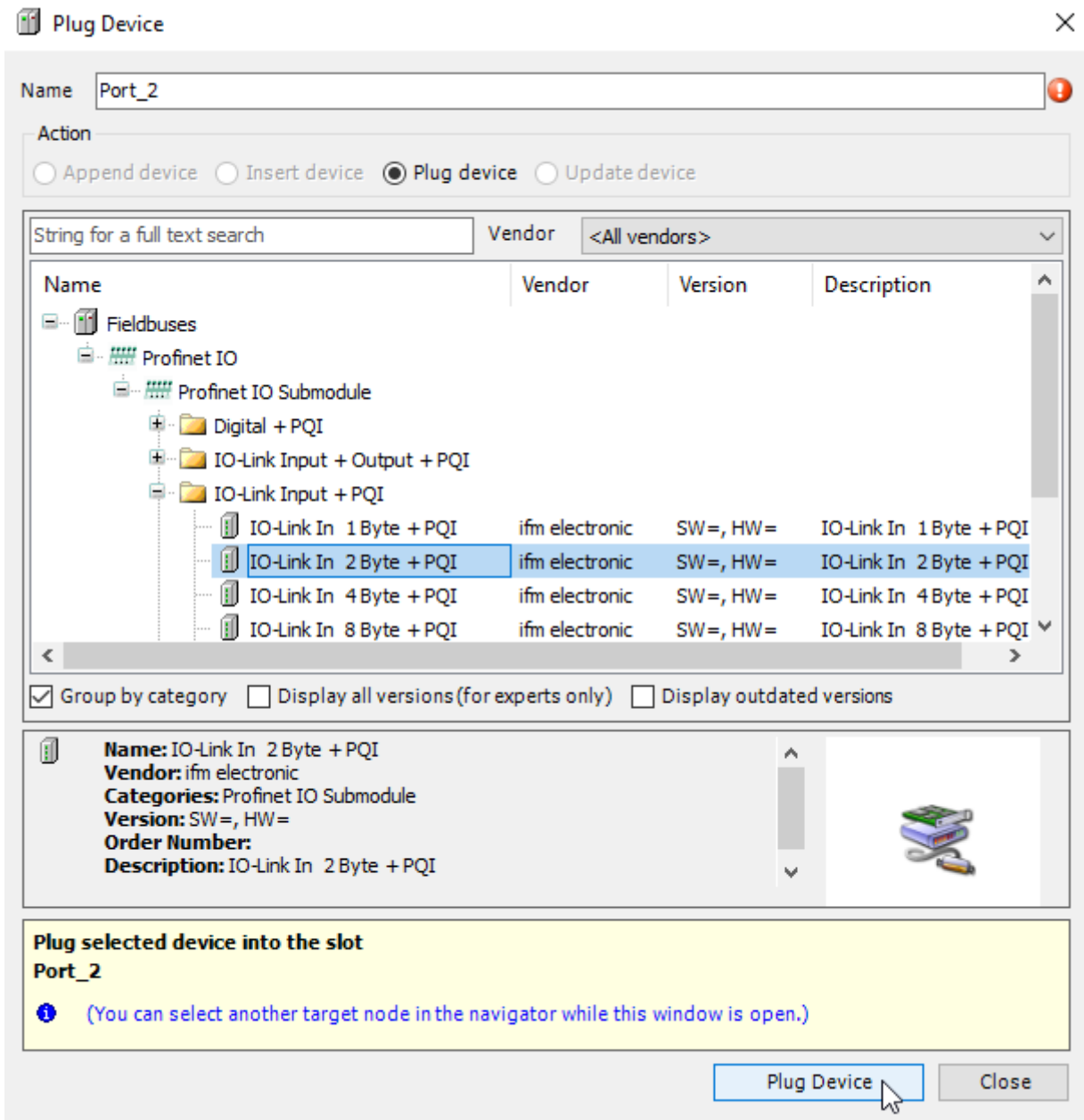




Il s'agit du détecteur de distance O5D150 qui est branché sur Port\_2, celui ci nécessite 2 Bytes d'Input pour le message :

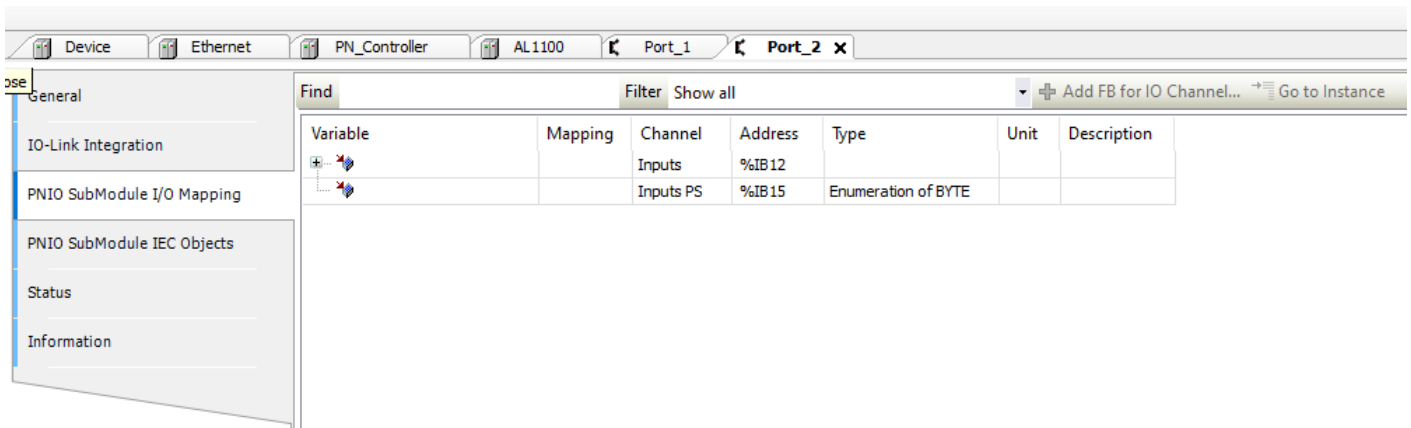
- IO-Link 2 Bytes + PQI
- et Plug Device





Pareil que précédemment, on fait attention à autoriser Always update variables avec l'option :

- Enabled 1 (use bus cycle task if not used in any task)





Reset Mapping
Always update variables
Enabled 1 (use bus cycle task if not used in any task) ▼

🔍 = Create new variable
🔍 = Map to existing variable

Pour les Port\_3 et Port\_4, comme aucun capteur n'est branché dessus, nous placerons Disabled

Plug Device
✕

Name  !

**Action**  
☐ Append device
☐ Insert device
☒ Plug device
☐ Update device

String for a full text search 
Vendor <All vendors> ▼

Name	Vendor	Version	Description
Fieldbuses			
Profinet IO			
Profinet IO Submodule			
Digital + PQI			
IO-Link Input + Output + PQI			
IO-Link Input + PQI			
IO-Link Output + PQI			
Disabled	ifm electronic	SW =, HW =	Disabled

☒ Group by category
☐ Display all versions (for experts only)
☐ Display outdated versions

**Name:** Disabled  
**Vendor:** ifm electronic  
**Categories:** Profinet IO Submodule  
**Version:** SW =, HW =  
**Order Number:**  
**Description:** Disabled

**Plug selected device into the slot**  
Port\_3  
! (You can select another target node in the navigator while this window is open.)

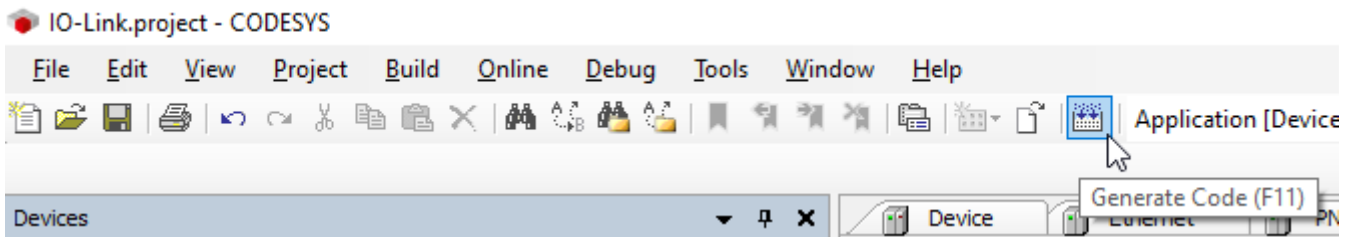
Plug Device Close

# Test rapide du fonctionnement du Master IO-Link

Sans oublier de faire Save, cliquer sur

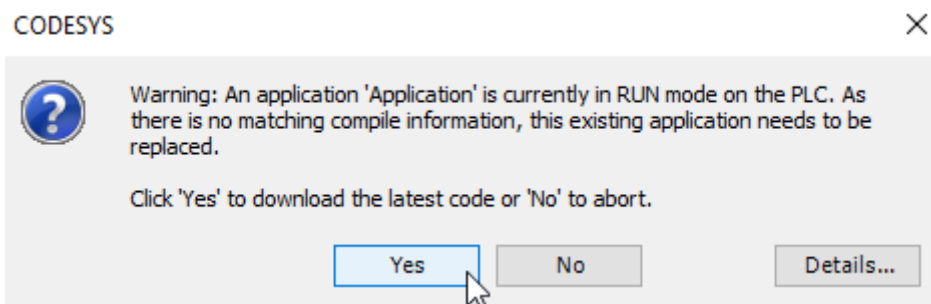
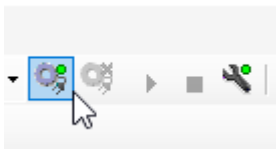
- Generate Code (F11)





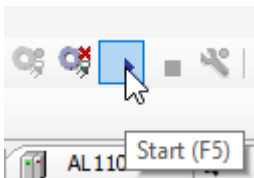
Pour se connecter sur le Raspberry Pi et y transférer le code Automate, cliquer sur la petite prise de courant (Login)

- Codesys peut indiquer ce message en indiquant qu'une application est déjà en Run sur le Raspberry Pi, confirmer avec Yes le transfert.



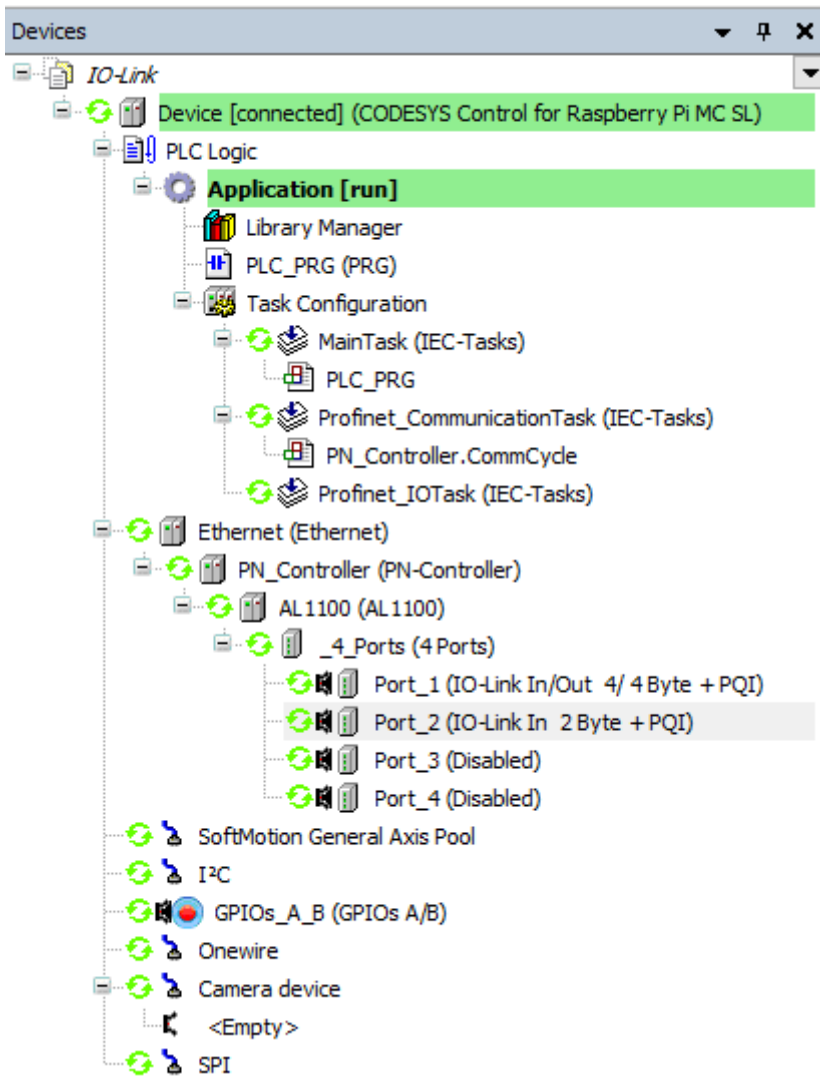
On clique sur :

- Start (F5)



Normalement, tous les éléments passent au vert.





En cliquant sur le Port\_2, dans

- PN/IO SubModule I/O Mapping, en développant l'arborescence, on peut voir les données de distance dans :
  - Input data 2 Bytes[0] %IB12
  - Input data 2 Bytes[0] %IB13

Ces données varient avec la distance mesurée. Comme pour TwinCAT, il faudra mapper ces données aux variables du programme Automate.

Variable	Mapping	Channel	Address	Type	Current Value
Inputs			%IB 12	ARRAY [0..1] OF BYTE	Only subelements up...
Input data 2 Bytes			%IB 12	BYTE	1
Input data 2 Bytes[0]			%IB 13	BYTE	97
Input data 2 Bytes[1]			%IB 14	USINT	160
Port Status			%IB 15	Enumeration of BYTE	GOOD
Inputs PS					



En cliquant sur le Port\_1, dans

- PN/IO SubModule I/O Mapping, en développant l'arborescence, on peut voir les données de distance dans :
  - Input datat 4 Bytes[0] %IB5
  - Input datat 4 Bytes[1] %IB6
  - ... Ces données varient avec l'appui sur le bouton capacitif. Pareillement, nous devrons mapper ces variables capteur au programme automate.

Variable	Mapping	Channel	Address	Type	Current Value
Inputs			%IB5		Only subelements up...
Input data 4 Bytes			%IB5	ARRAY [0..3] OF BYTE	Only subelements up...
Input data 4 Bytes[0]			%IB5	BYTE	11
Input data 4 Bytes[1]			%IB6	BYTE	184
Input data 4 Bytes[2]			%IB7	BYTE	0
Input data 4 Bytes[3]			%IB8	BYTE	1
Port Status			%IB9	USINT	160
Inputs PS			%IB10	Enumeration of BYTE	GOOD
Output data 4 Bytes			%QB0	ARRAY [0..3] OF BYTE	Only subelements up...
Output data 4 Bytes[0]			%QB0	BYTE	0
Output data 4 Bytes[1]			%QB1	BYTE	0
Output data 4 Bytes[2]			%QB2	BYTE	0
Output data 4 Bytes[3]			%QB3	BYTE	0
Outputs CS			%IB11	Enumeration of BYTE	GOOD

Pour se déconnecter, on clique sur Stop pour Logout.

## Bilan

Cette première étape nous a permis de valider la bonne communication avec le Master IO-Link. Les prochaines étapes seront:

- création des fonctions de décodage pour les capteurs
- mapping des variables automate avec les variables IO-Link
- création d'une IHM permettant d'afficher ces résultats
- test de l'IHM sur une tablette

# Programme Automate et fonctions capteurs

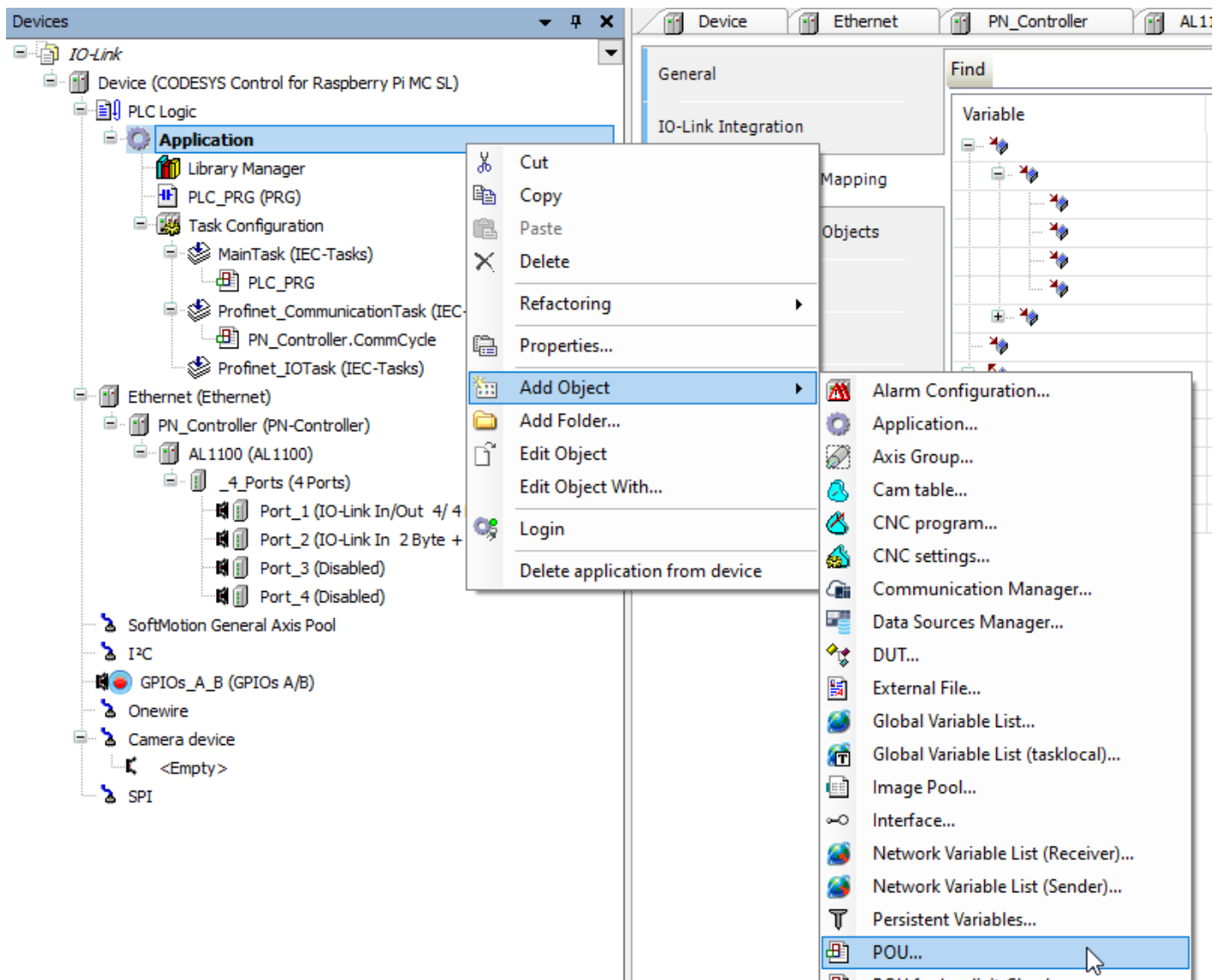
Les différentes étapes présentées ci-dessous seront très similaires à celles présentées pour l'association d'un Master IO-Link à TwinCAT.

## Fonction capteur O5D150



Dans Application faire :

- Add Object -> POU...



Dans Add POU :

- Name : FB\_O5D150
- Type : Function block
- Implementation Language : ST





Create a new POU (Program Organization Unit)

Name  
FB\_O5D150

Type

☐ Program

☒ **Function block**

☐ Extends  ...

☐ Implements  ...

☐ Final ☐ Abstract

Access specifier  
 ▾

Method implementation language  
Continuous Function Chart (CFC) ▾

☐ **Function**

Return type  ...

Implementation language  
Structured Text (ST) ▾

Add Cancel

Le programme de la fonction FB\_O5D150 est identique à celui présenté dans TwinCAT :



The screenshot shows the CODESYS IDE interface. On the left, the 'Devices' tree is expanded to 'IO-Link', showing the hierarchy: Device (CODESYS Control for Raspberry Pi MC SL) > PLC Logic > Application > Library Manager > FB\_05D150 (FB). The main editor displays the ladder logic for the FB\_05D150 block. The code is as follows:

```

1  FUNCTION_BLOCK FB_05D150
2  VAR_INPUT
3  END_VAR
4  VAR_OUTPUT
5      nCurrentDistance : INT;
6      bSwitchState     : BOOL;
7  END_VAR
8  VAR
9      aInputBytes      : ARRAY[0..1] OF BYTE;
10     nWord0            : WORD;
11     nWord_temp0       : WORD;
12     nWord_temp1       : WORD;
13 END_VAR
14
15 nWord_temp0 := TO_WORD(aInputBytes[0]);
16 nWord_temp1 := TO_WORD(aInputBytes[1]);
17
18 nWord_temp0 := SHL(nWord_temp0,8);
19
20 nWord0 := nWord_temp0 OR nWord_temp1;
21
22 bSwitchState := nWord0.0;
23
24 nWord_temp0 := SHR(nWord0,4);
25 nCurrentDistance := TO_INT(nWord_temp0);

```

De même pour l'instanciation et l'appel de fonction, seule variante, le PLC\_PRG est dans notre cas écrit en Ladder, ce qui signifie qu'il est nécessaire d'ajouter un bloc à nommer fb05D150 pour réaliser l'appel de fonction.

The screenshot shows the CODESYS IDE interface. On the left, the 'Devices' tree is expanded to 'IO-Link', showing the hierarchy: Device (CODESYS Control for Raspberry Pi MC SL) > PLC Logic > Application > Library Manager > PLC\_PRG (PRG). The main editor displays the ladder logic for the PLC\_PRG program. The code is as follows:

```

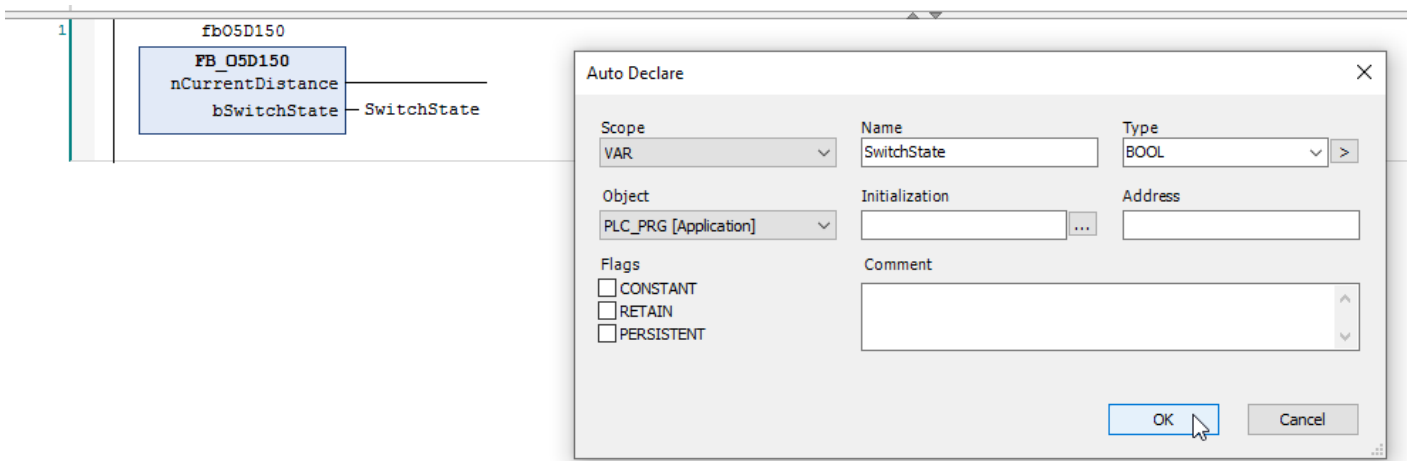
1  PROGRAM PLC_PRG
2  VAR
3      fb05D150 : FB_05D150;
4  END_VAR

```

Below the code, a ladder logic diagram shows the instantiation of the FB\_05D150 block. The block is labeled 'fb05D150' and has two outputs: 'nCurrentDistance' and 'bSwitchState'. The output 'bSwitchState' is connected to a red '???' symbol, indicating a missing or undefined output.

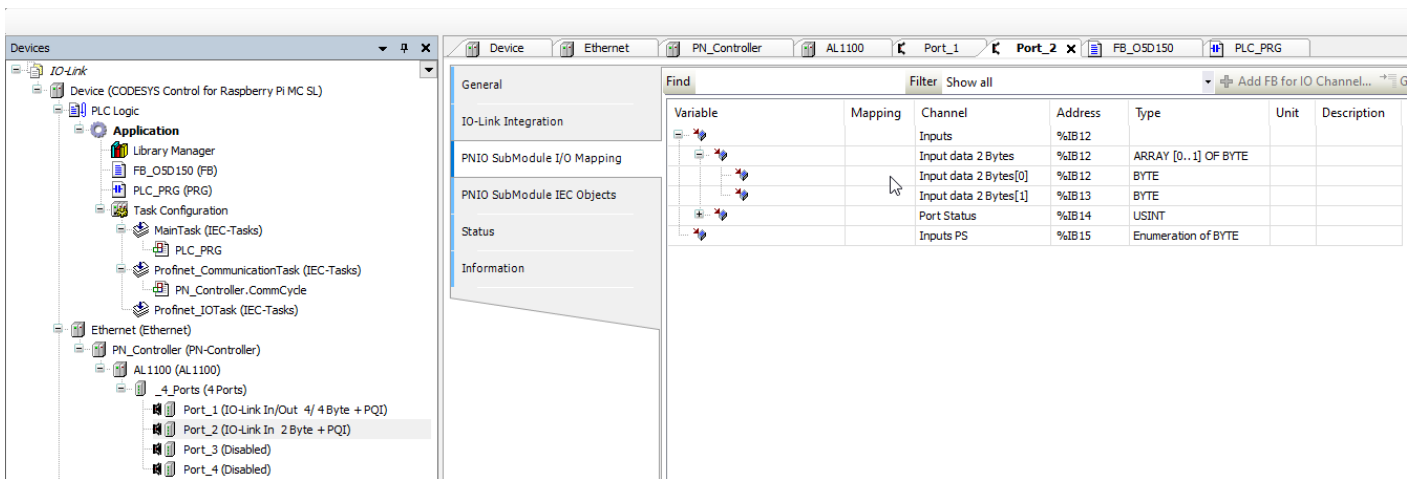
On peut ajouter une variable SwitchState de type dans le PLC\_PRG simplement en écrivant SwitchState en sortie du bloc.





Nous allons mapper les variables de la fonction O5D150 aux variables IO-Link :

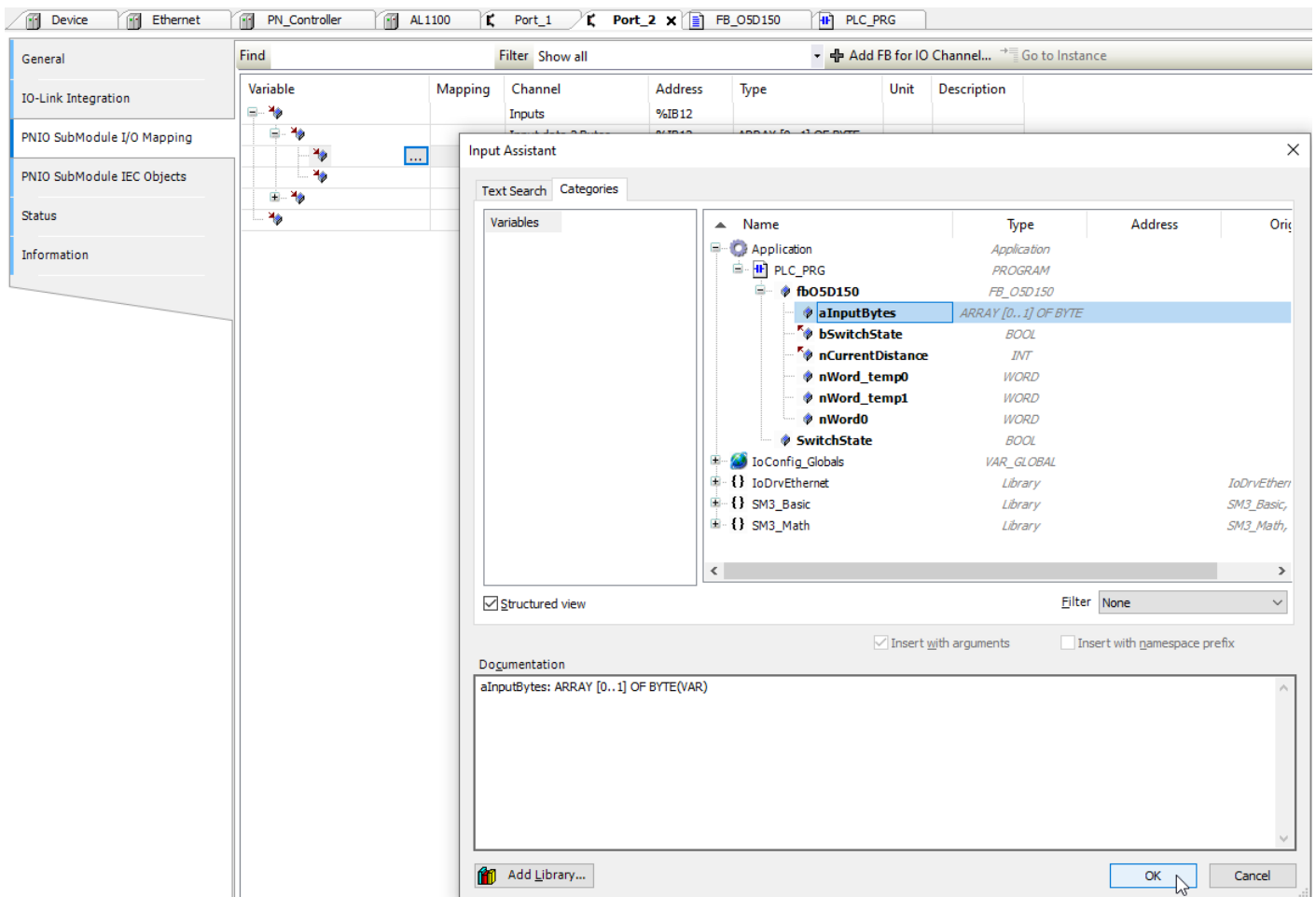
- dans Port\_2 -> PN I/O SubModule Mapping
- cliquer à côté de la petite boîte bleue dans variable au niveau de la ligne %IB12 BYTE



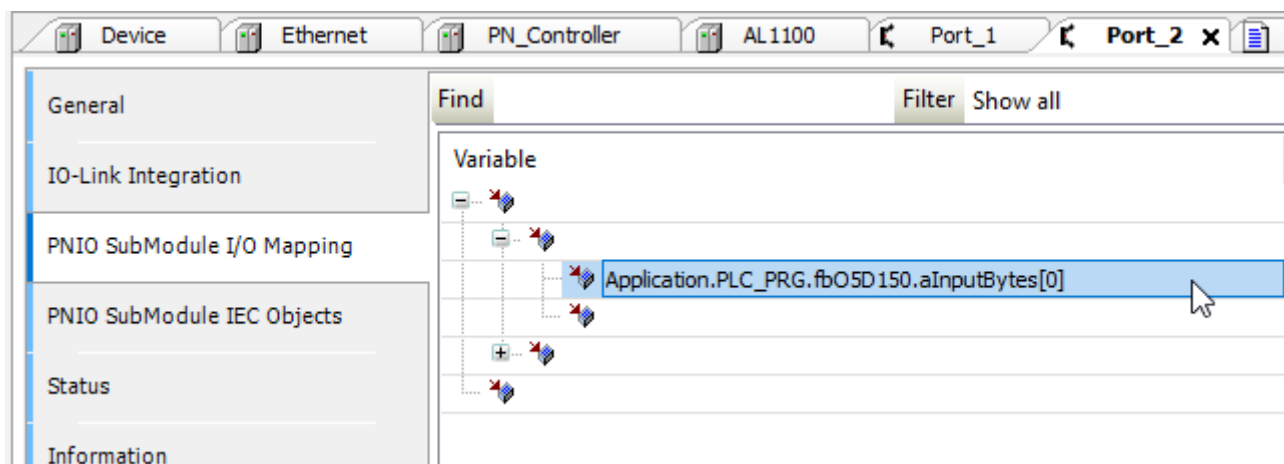
Trois petits points doivent apparaître et lancer l'Input Assistant:

- choisir dan PLC\_PRG -> fbO5D150 -> alnputBytes

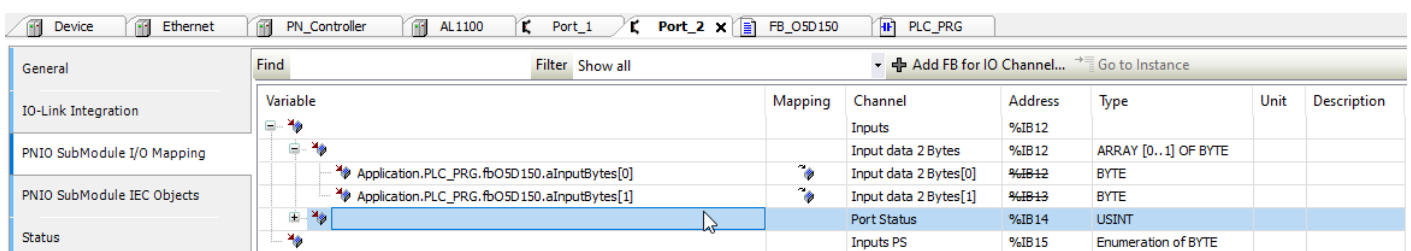




Placer à la fin de la variable [0] pour indiquer qu'il s'agit de l'élément 0 de l'array of Byte.



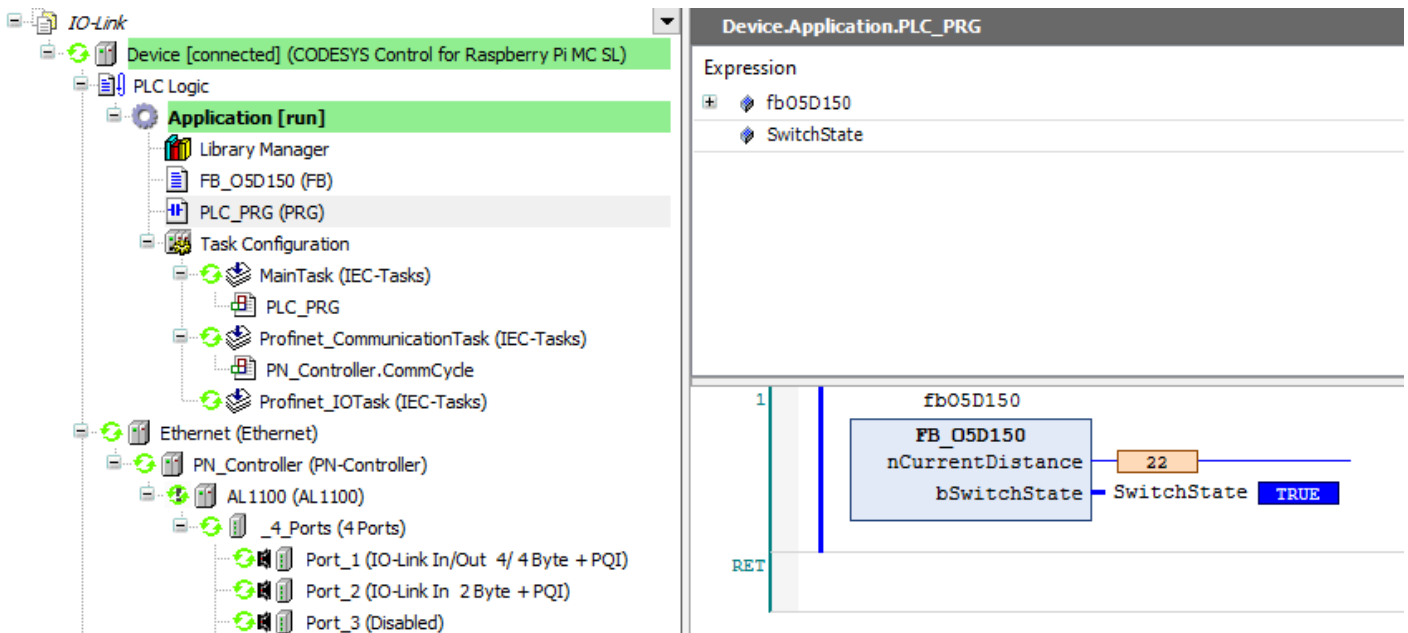
On fait de même pour la ligne %IB13. N'oubliez pas de placer à la fin de la variable [1] pour indiquer qu'il s'agit de l'élément 1 de l'array of Byte.





On effectue un Save, suivi d'un Generate, Login et Start

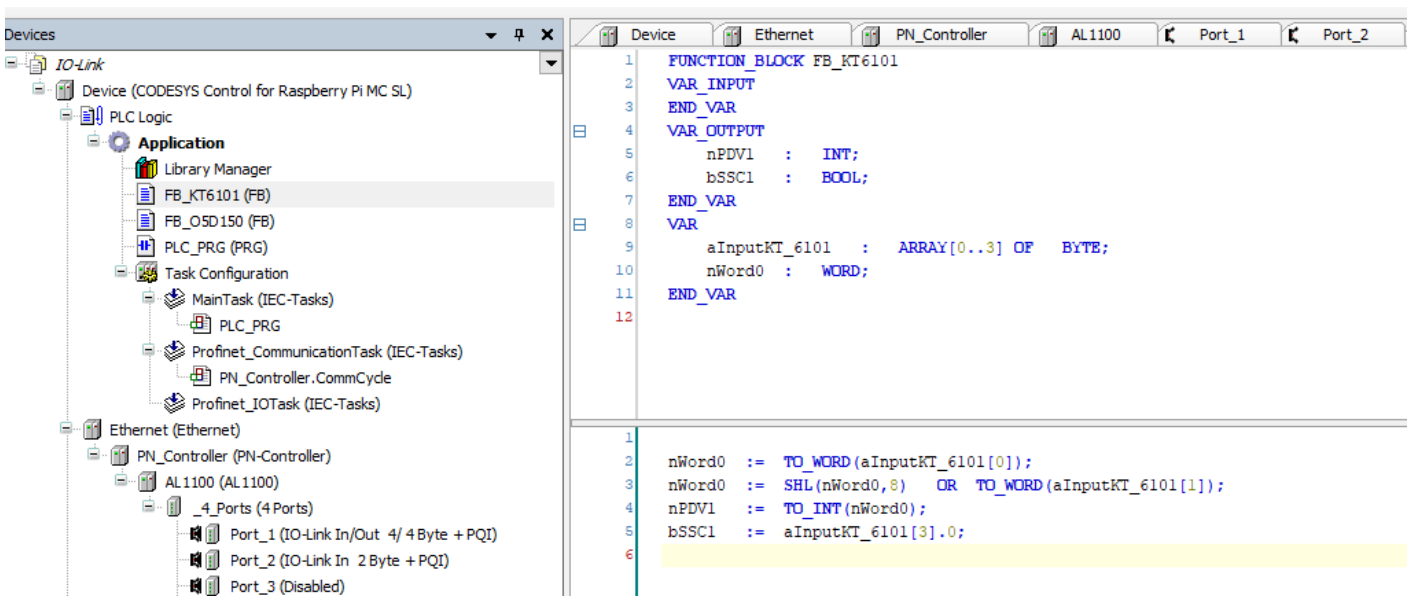
- et normalement le PLC\_PRG indique la valeur de distance mesurée par le capteur O5D150.



On n'oublie pas de faire Stop et Logout pour faire des modification du programme.

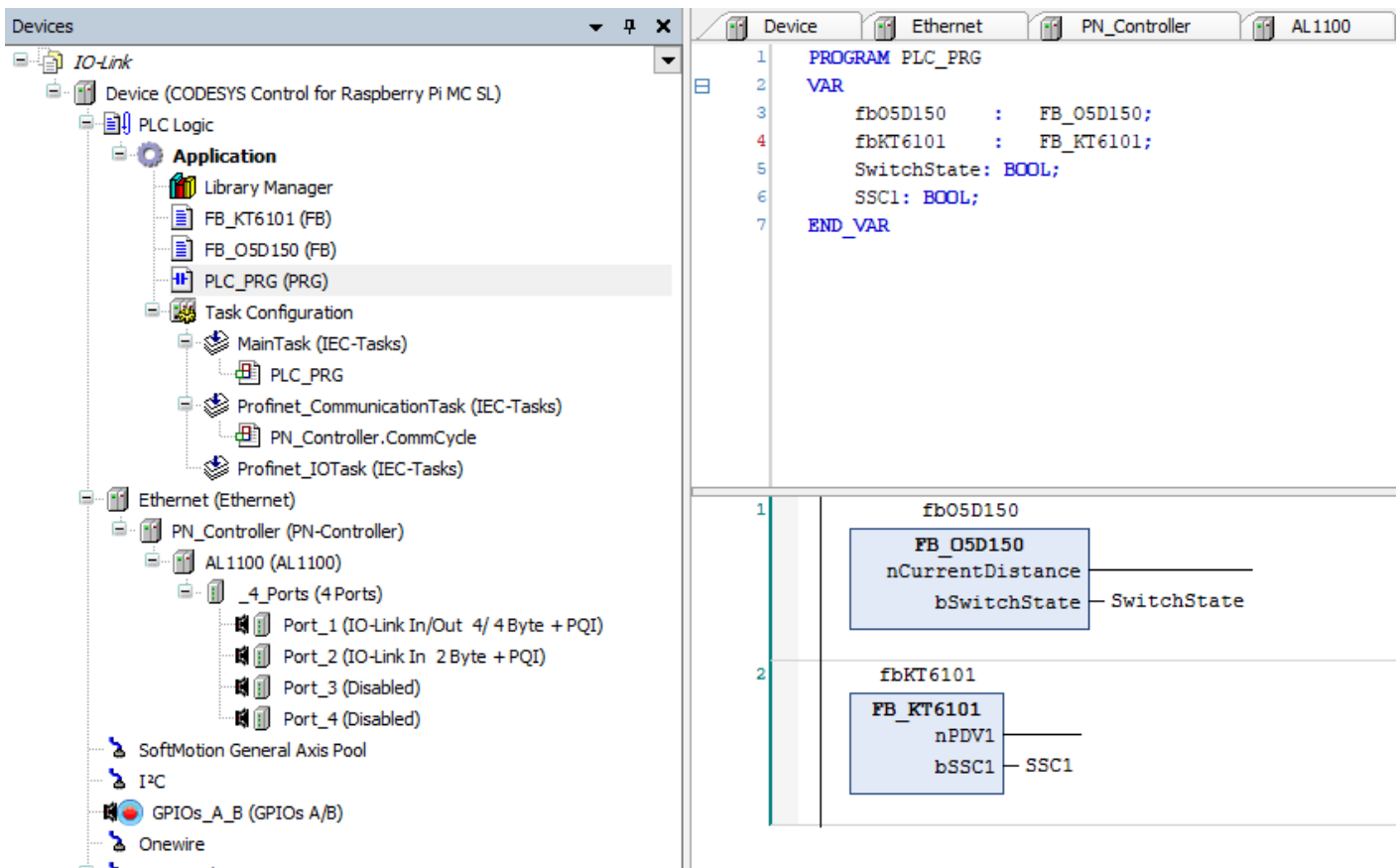
## Fonction bouton capacitif KT6101

Comme pour le capteur O5D150, le code pour le bouton capacitif KT6101 est identique à celui développé pour TwinCAT.



On réalise l'instanciation et l'appel de la fonction fbKT6101 dans PLC\_PRG





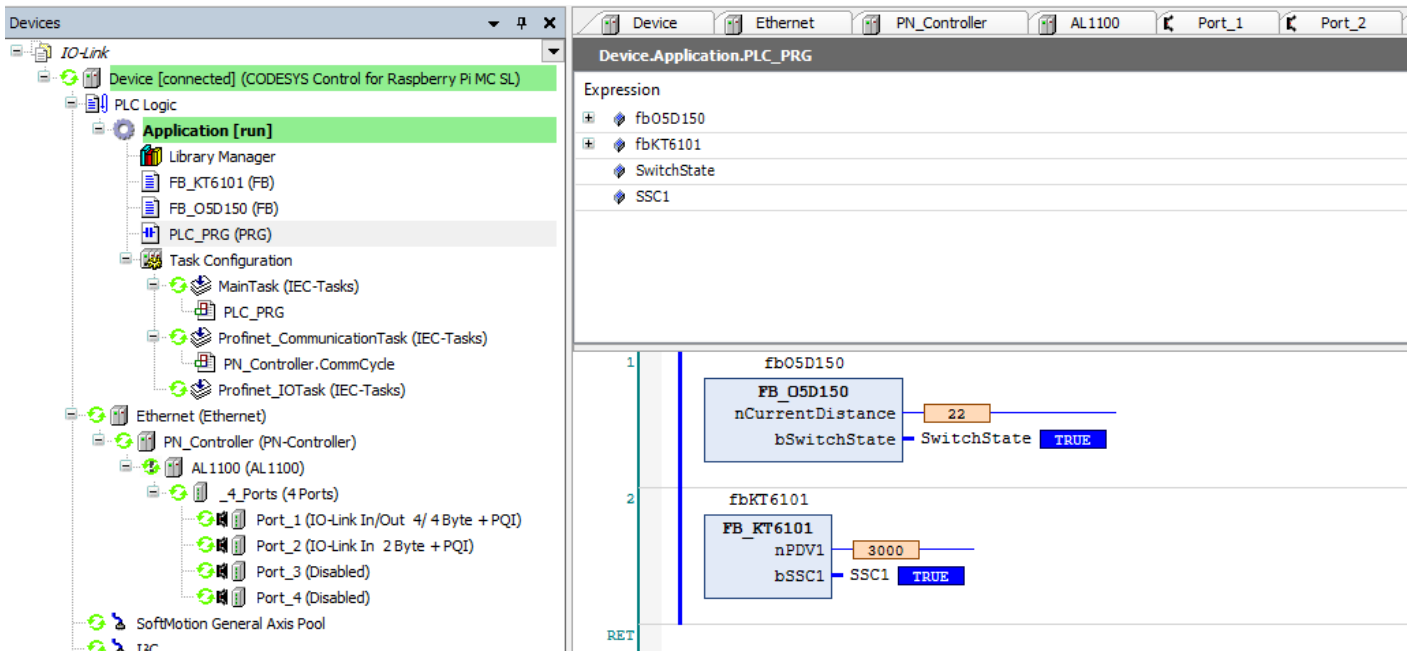
Pour le mapping des variables IO-Link, on procède de la même manière que le capteur O5D150

Variable	Mapping	Channel	Address	Type
Application.PLC_PRG.fbKT6101.aInputKT_6101[0]		Inputs	%IB5	ARRAY [0..3] OF BYTE
Application.PLC_PRG.fbKT6101.aInputKT_6101[1]		Input data 4 Bytes[0]	%IB5	BYTE
Application.PLC_PRG.fbKT6101.aInputKT_6101[2]		Input data 4 Bytes[1]	%IB6	BYTE
Application.PLC_PRG.fbKT6101.aInputKT_6101[3]		Input data 4 Bytes[2]	%IB7	BYTE
		Input data 4 Bytes[3]	%IB8	BYTE
		Port Status	%IB9	USINT
		Inputs PS	%IB10	Enumeration of BYTE
		Output data 4 Bytes	%QB0	ARRAY [0..3] OF BYTE
		Output data 4 Bytes[0]	%QB0	BYTE
		Output data 4 Bytes[1]	%QB1	BYTE
		Output data 4 Bytes[2]	%QB2	BYTE
		Output data 4 Bytes[3]	%QB3	BYTE
		Outputs CS	%IB11	Enumeration of BYTE

On fait Generate, Login et Start :

- Le capteur de distance est toujours fonctionnel
- Le bouton capacitif réagit en fonction de l'appui





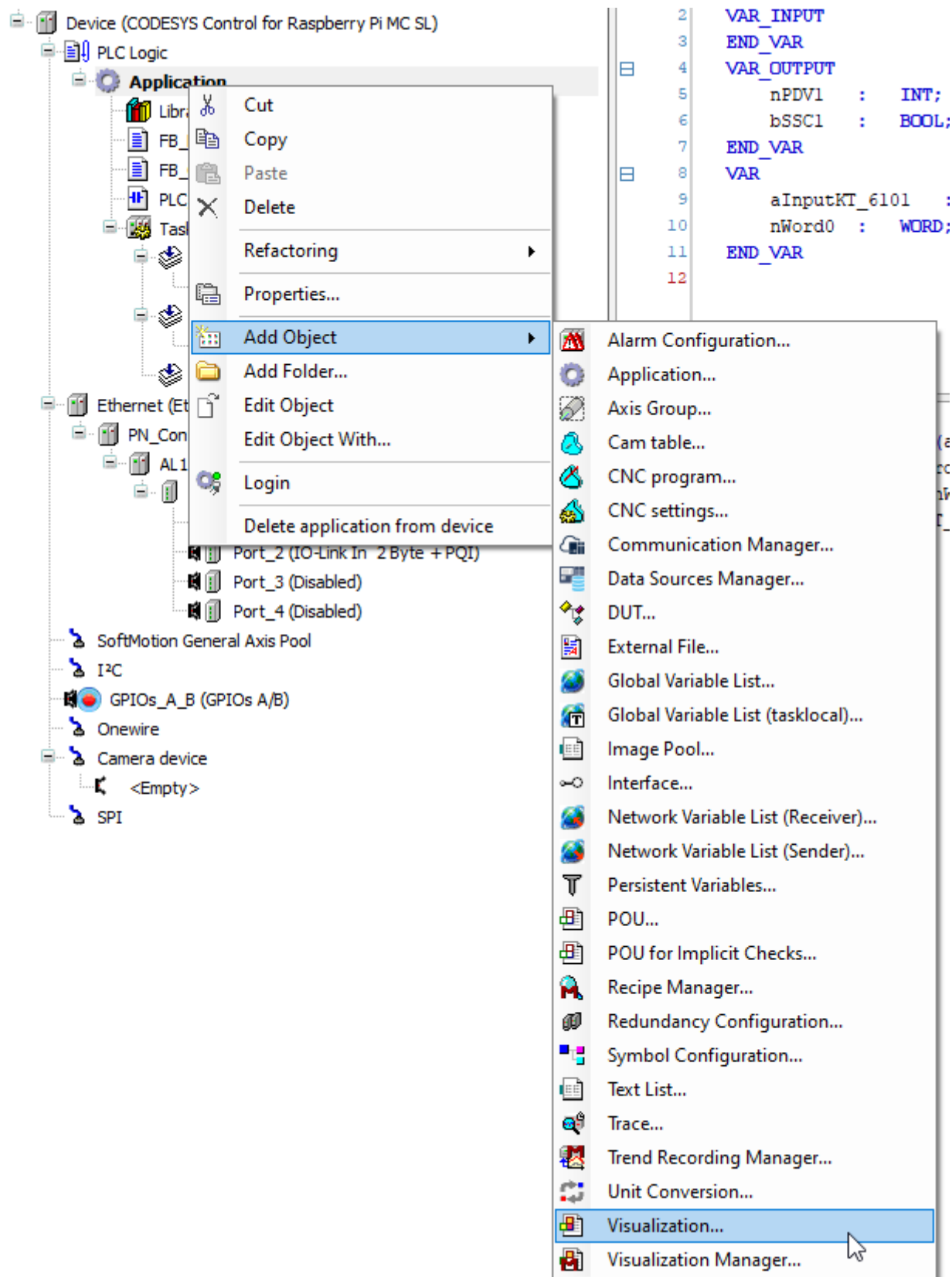
Pour finaliser cette démonstration, nous allons intégrer une IHM et réaliser une WebVisu sur tablette.

# Visualisation IHM et WebVisu

Pour intégrer une visualisation :

- Clic droit sur Application -> Add Object -> Visualization...
- Dans la fenêtre Add Visualization, laisser le Name : Visualization et faire Add






Pour simuler les deux couleurs lumineuses Bleue et Verte, j'utilise une astuce sur la visualisation :

- Je crée un voyant Bleu que j'associe à la variable PLC\_PRG.fbKT6101.bSSC1 que j'inverse avec NOT().
- ce voyant passera au premier plan avec la même variable dans Bring to foreground
- -> quand le voyant bleu est allumé, il passe au premier plan





Property	Value
Element name	GenElemInst_1
Type of element	Lamp
Position	
X	519
Y	133
Width	70
Height	70
Variable	NOT(PLC_PRG.fbKT6101.bSSC1)
Image settings	
Transparent	<input type="checkbox"/>
Transparent color	 Black
Isotropic type	Isotropic
Horizontal align...	Left
Vertical alignment	Top
Texts	
Tooltip	
State variables	
Invisible	
Center	
X	554
Y	168
Absolute movement	
Movement	
X	
Y	
Rotation	
Scaling	
Interior rotation	
Animation duration	0
Bring to foreground	NOT(PLC_PRG.fbKT6101.bSSC1)
Background	
Image	Blue

Pour le voyant vert, je procède de la même manière sans inverser la variable PLC\_PRG.fbKT6101.bSSC1





Element name	GenElemInst_2
Type of element	Lamp
Position	
X	614
Y	132
Width	70
Height	70
Variable	PLC_PRG.fbKT6101.bSSC1
Image settings	
Transparent	
Transparent color	Black
Isotropic type	Isotropic
Horizontal align...	Left
Vertical alignment	Top
Texts	
Tooltip	
State variables	
Invisible	
Center	
X	649
Y	167
Absolute movement	
Movement	
X	
Y	
Rotation	
Scaling	
Interior rotation	
Animation duration	0
Bring to foreground	PLC_PRG.fbKT6101.bSSC1
Background	
Image	Green

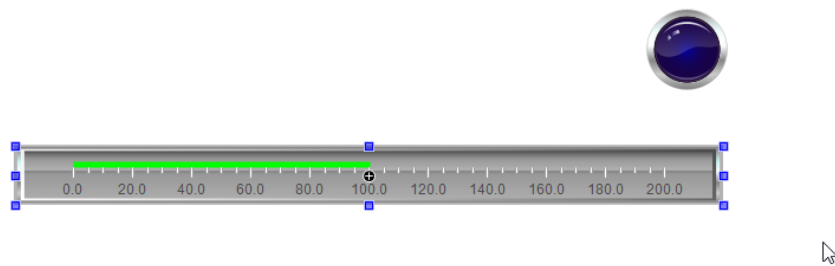
Je superpose les deux voyants sur l'IHM :

- si le voyant Bleu est actif, il passera au premier plan
- si le voyant Vert est actif, il passera au premier plan

Une jauge de distance est également placée :

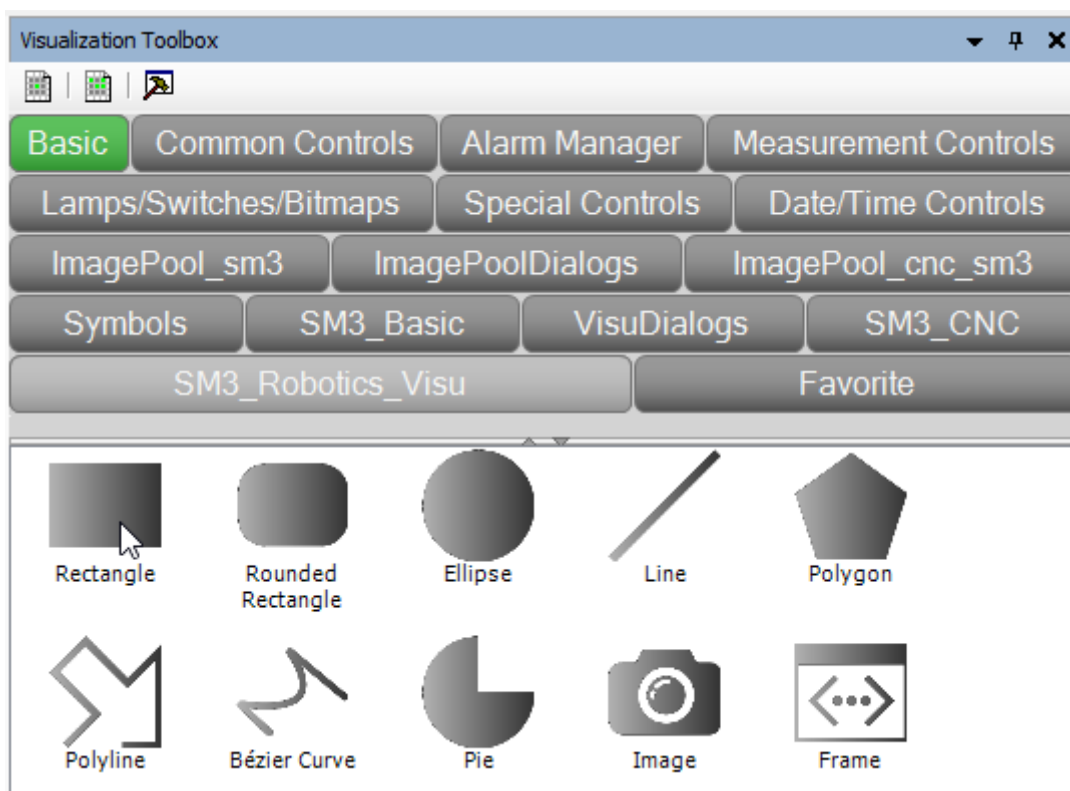
- la variable associée à la jauge est `PLC_PRG.fb05D150.nCurrentDistance`
- Scale End est configuré à 200 pour la distance max du capteur en cm
- Main Scale est placée à 20





Property	Value
Element name	GenElemInst_3
Type of element	Bar Display
Value	PLC_PRG.fbOSD150.nCurrentDistance
Center	
X	463
Y	274
Absolute movement	
Movement	
X	
Y	
Rotation	
Scaling	
Interior rotation	
Animation duration	0
Bring to foreground	
Position	
X	164
Y	249
Width	599
Height	50
Background	
Image color	Gray
Own image	
Bar	
Diagram type	Scale besides bar
Orientation	Horizontal
Running direction	Left to right
Optimum size f...	<input type="checkbox"/>
Scale	
Scale start	0
Scale end	200
Main scale	20
Subscale	5
Scale line width	1
Scale color	<input type="text"/> Scale color barelement
Scale in 3D	<input checked="" type="checkbox"/>
Element frame	<input type="checkbox"/>

On souhaite également afficher la distance dans un champ, nous placerons ainsi un rectangle qui servira de champ de visualisation :



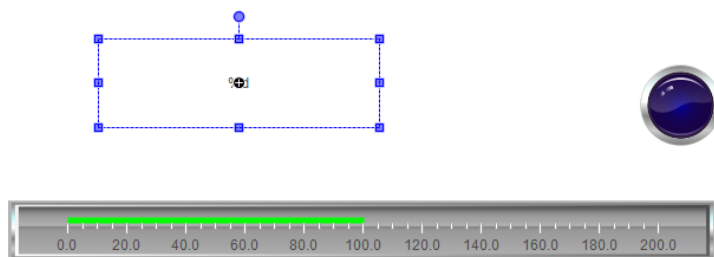
Dans le champ Text, nous mettrons :

- %d pour la valeur à afficher en Int



Dans le Champ Text variable, nous mettrons :

- PLC\_PRG. fb05D150. nCurrentDistance

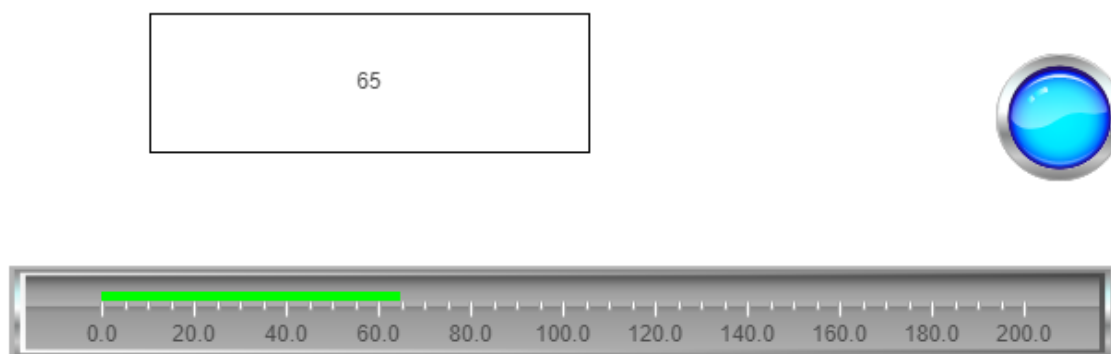


Property	Value
Element name	GenElemInst_4
Text ID	569
Type of element	Rectangle
Position	
X	239
Y	111
Width	238
Height	75
Angle	0
Center	
X	358
Y	148
Colors	
Use gradient color	<input type="checkbox"/>
Gradient setting	linear, Black, White
Appearance	
Texts	
Text	%d
Tooltip	
Text properties	
Absolute movement	
Movement	
X	
Y	
Rotation	
Scaling	
Interior rotation	
Use REAL values	<input type="checkbox"/>
Relative movement	
Text variables	
Text variable	PLC_PRG.fb05D150.nCurrentDistance
Tooltip variable	

Il reste à faire la connexion au Raspberry Pi et lancer le Runtime :

- Generate, Login, Start

Et l'IHM s'anime en fonction des mesures et appuis sur les capteurs.





# WebVisu

Il est possible de récupérer cette IHM sur une page Web à travers la WebVisu.

- On place l'adresse IP du Raspberry Pi dans le navigateur de la tablette suivi du numéro de port 8080 et l'on charge la page webvisu.htm
- 192.168.1.15:8080/webvisu.htm
- On peut visualiser la mesure de distance faite par le capteur ainsi que les appuis sur le bouton capacitif

---

Revision #2

Created 4 July 2023 09:28:01 by Philippe Celka

Updated 4 July 2023 09:51:21 by Philippe Celka