

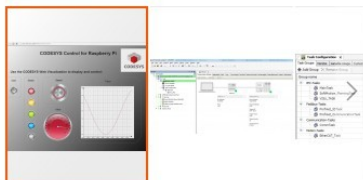
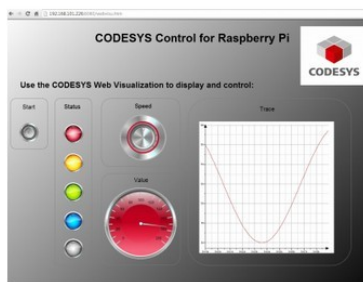
# Codesys sur Raspberry Pi 4

## Le Runtime Codesys pour Raspberry Pi

### Présentation

Codesys fournit un Runtime pour cible Raspberry Pi afin de le transformer en Automate exécutant du code Ladder / FBD / CFC et sur lequel nous pouvons piloter les GPIO.

- Le plugin peut se télécharger [ici](#)
- Il est nécessaire d'avoir un compte Codesys (gratuit)
- Le Runtime est **gratuit** mais il stoppe au bout de 2h ce qui nécessite un redémarrage (pas gênant pour un TP, inutilisable en production)
- Si vous souhaitez le Runtime sans la limite des 2h, il vous coûtera 100€ pour une clé de licence à demander directement chez Codesys



#### CODESYS Control for Raspberry Pi MC SL



[7 Bewertungen](#)

[Fügen Sie Ihre Bewertung hinzu](#)

CODESYS Control for Raspberry Pi MC SL ist eine angepasstes CODESYS Control Laufzeitsystem für den Raspberry Pi mit mehreren Kernen auf einer CPU.

Aktuelle Version: 4.5.0.0

Artikelnummer: 2302000032

[Download](#)

100,00 €

Zzgl. Mehrwertsteuer

Kaufen Sie 4 für jeweils 95,00 € und **sparen Sie 5%**  
Kaufen Sie 10 für jeweils 91,00 € und **sparen Sie 9%**  
Kaufen Sie 25 für jeweils 85,00 € und **sparen Sie 15%**  
Kaufen Sie 50 für jeweils 81,00 € und **sparen Sie 19%**

[Anmelden und in den Warenkorb legen](#)

## Avis perso :

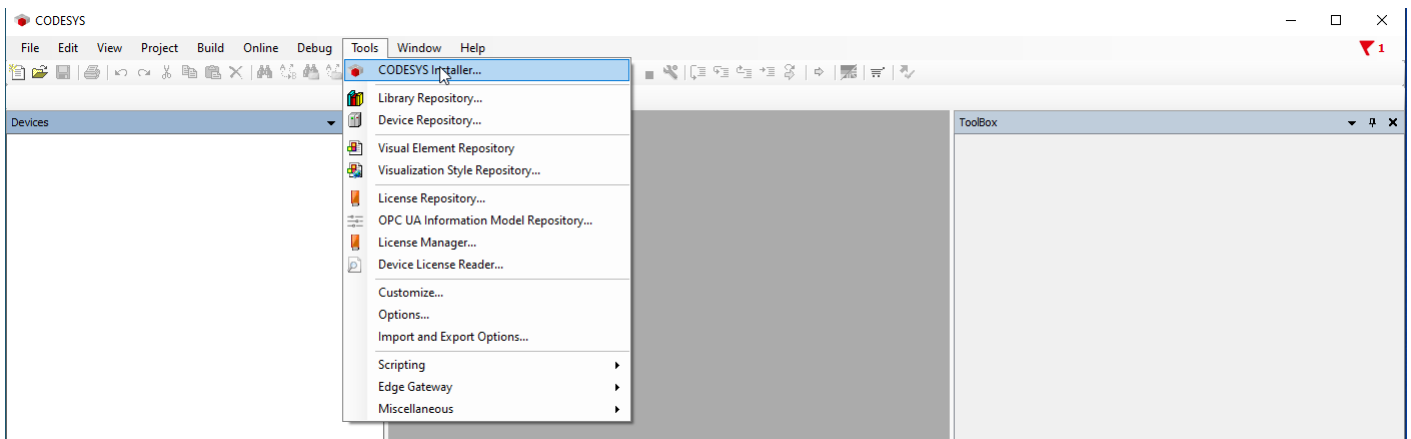
Il est tentant de se dire, qu'avec ce plugin à 100€ + un Raspberry Pi 4 à 50€, on peut atteindre des performances équivalentes à un système IPC à 2k€ chez Siemens ou autre sauf que :

- Un Raspberry Pi n'est pas durci pour résister à un environnement industriel (pas de garantie)
- Pas de possibilité de connecter des cartes I/O simplement, il faut passer par un bus de terrain de type Ethernet ou alors mettre des adaptateurs 0/24V sur les GPIO.
- De base, les performances temps réels ne sont pas garanties. (L'OS de la Raspberry Pi n'est pas patché RealTime ou Xenomai)
- Par contre, on peut intégrer facilement en plus des outils de type MQTT, NodeRed, Grafana car le Raspi possède une forte puissance de calcul face à un automate classique.

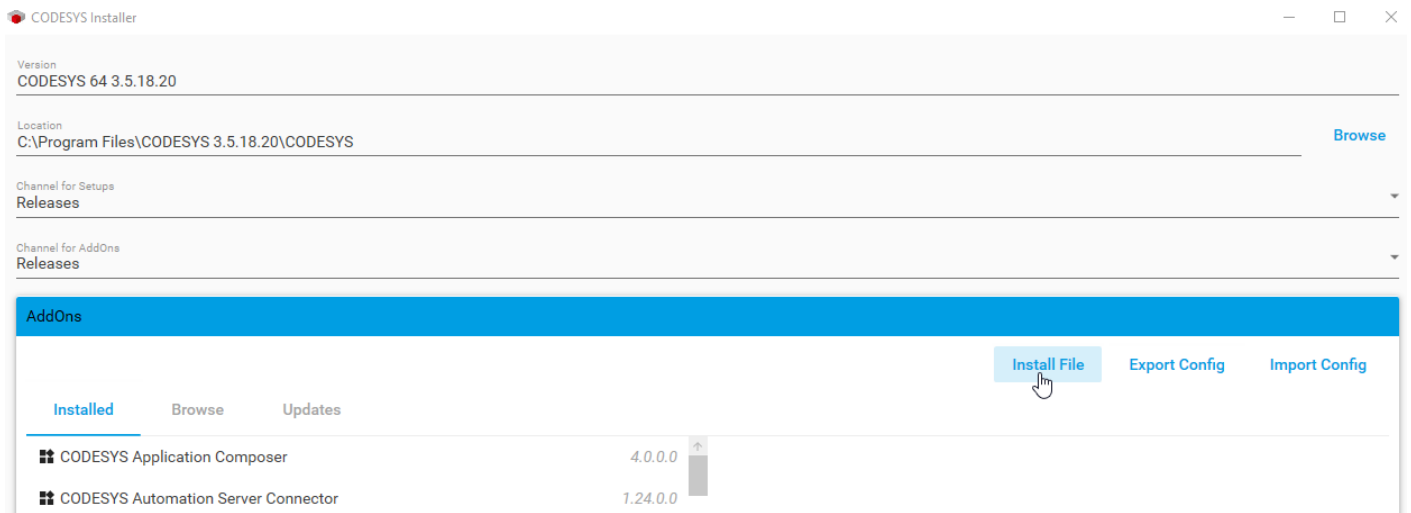
Malgré les limitations, des industriels parient sur ce type d'outils en production avec un applicatif qui nécessiterait normalement du matériel beaucoup plus coûteux. Dans le cadre d'un projet étudiant ou d'un POC, il s'agit d'un outil adapté et c'est toujours valorisant pour un étudiant de BUT GEII de pouvoir dire à un entretien qu'on a réalisé son propre IPC basé sur un Runtime Codesys sur cible Raspberry Pi 4.

# Installation du Runtime Raspberry Pi sur Codesys

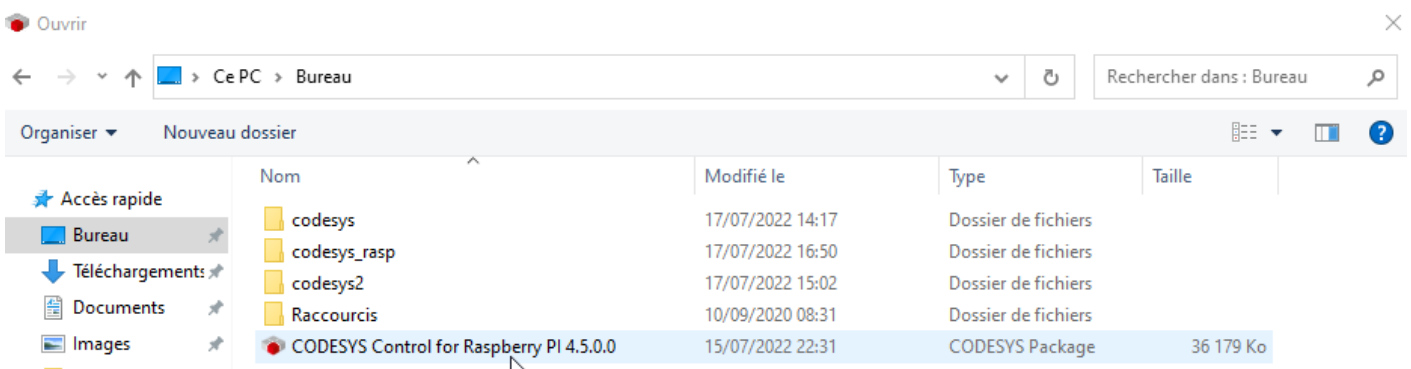
- Pour installer l'Addon, faire Tools -> CODESYS Installer



- Cliquer sur Install File



- Sélectionner le fichier téléchargé précédemment



- On ferme le logiciel Codesys en arrière Plan et l'on fait OK |

Confirmation Required

The following operations will be performed.  
Click OK to proceed with the changes listed below.

==== Install Packages ====

Install Packages (Silent=False, IncludeDependencies=False)

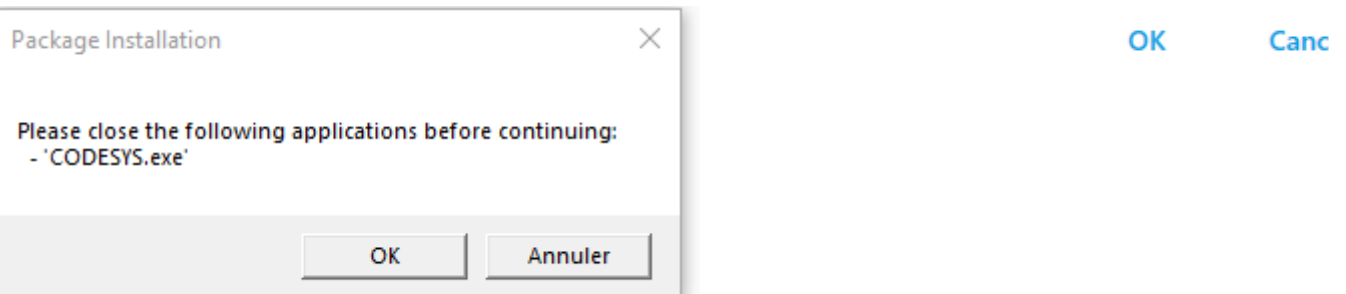
InstallationPath: 'C:\Program Files\CODESYS 3.5.18.20\CODESYS'

- CODESYS Control for Raspberry PI, Version 4.5.0.0 ('C:\Users\install\Desktop\CODESYS Control for Raspberry PI 4.5.0.0.packa

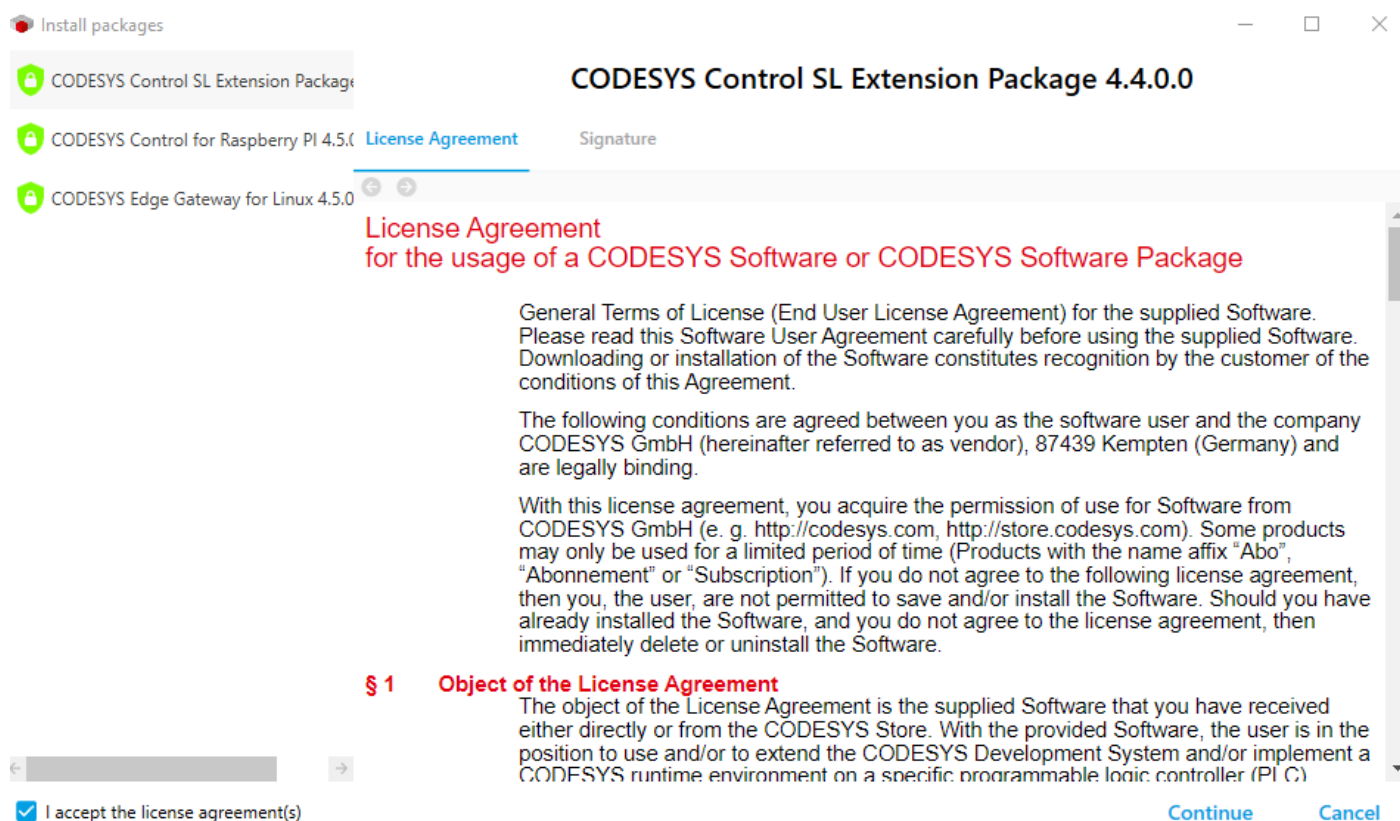
Dependencies

- CODESYS Edge Gateway for Linux, Version 4.5.0.0

- CODESYS Control SL Extension Package, Version 4.4.0.0



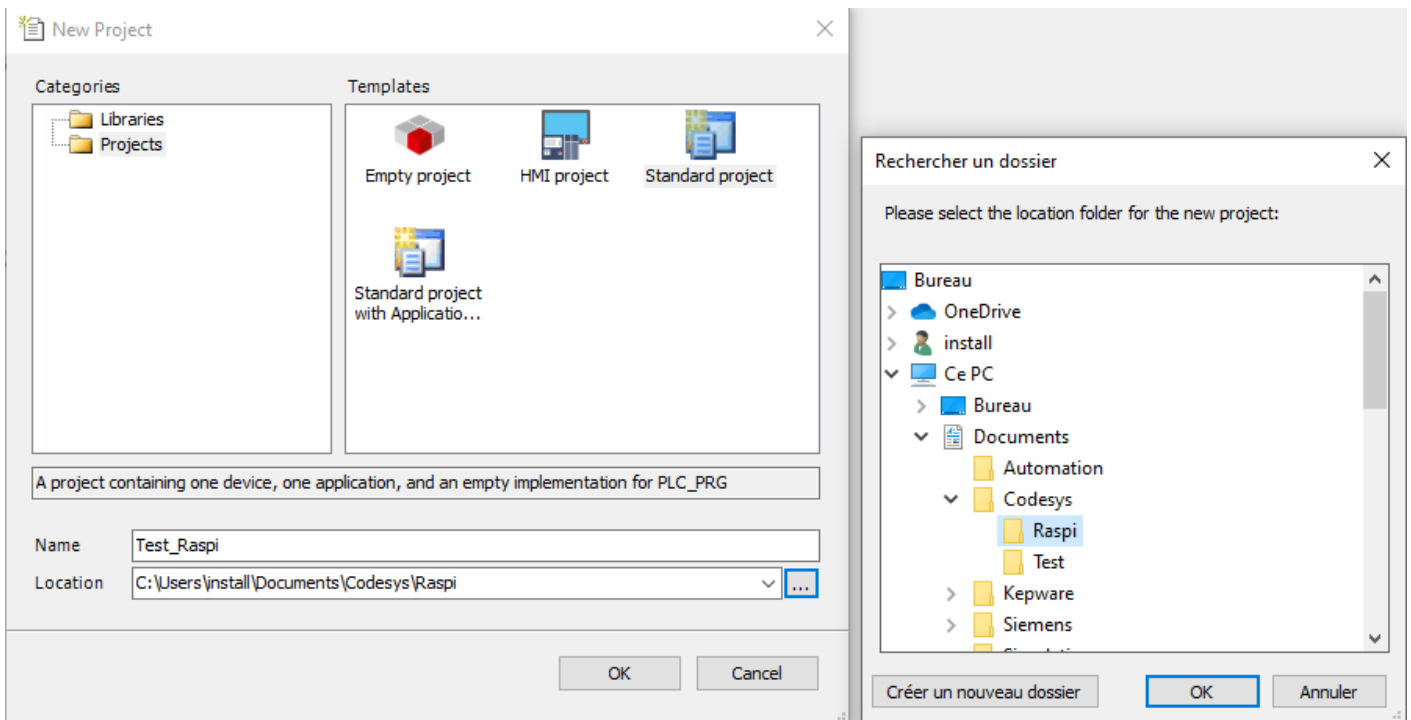
- |:-----|-----:|
- On coche I accept the license agreements et normalement l'AddOn est correctement installé. On peut passer à la création d'un Projet pour le Raspi.



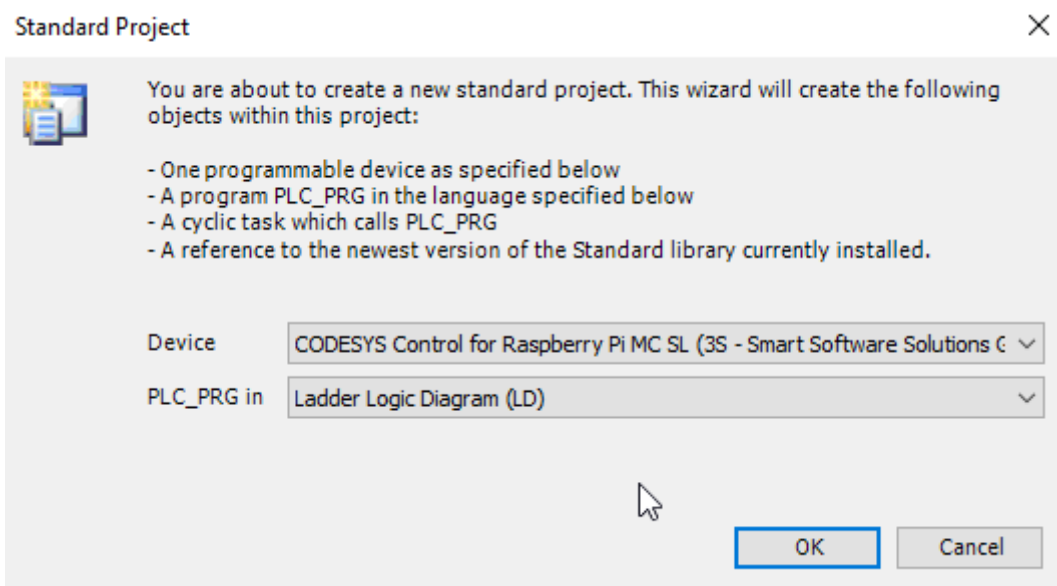
# Projet Codesys sur Raspberry Pi 4

## Arborescence de projet

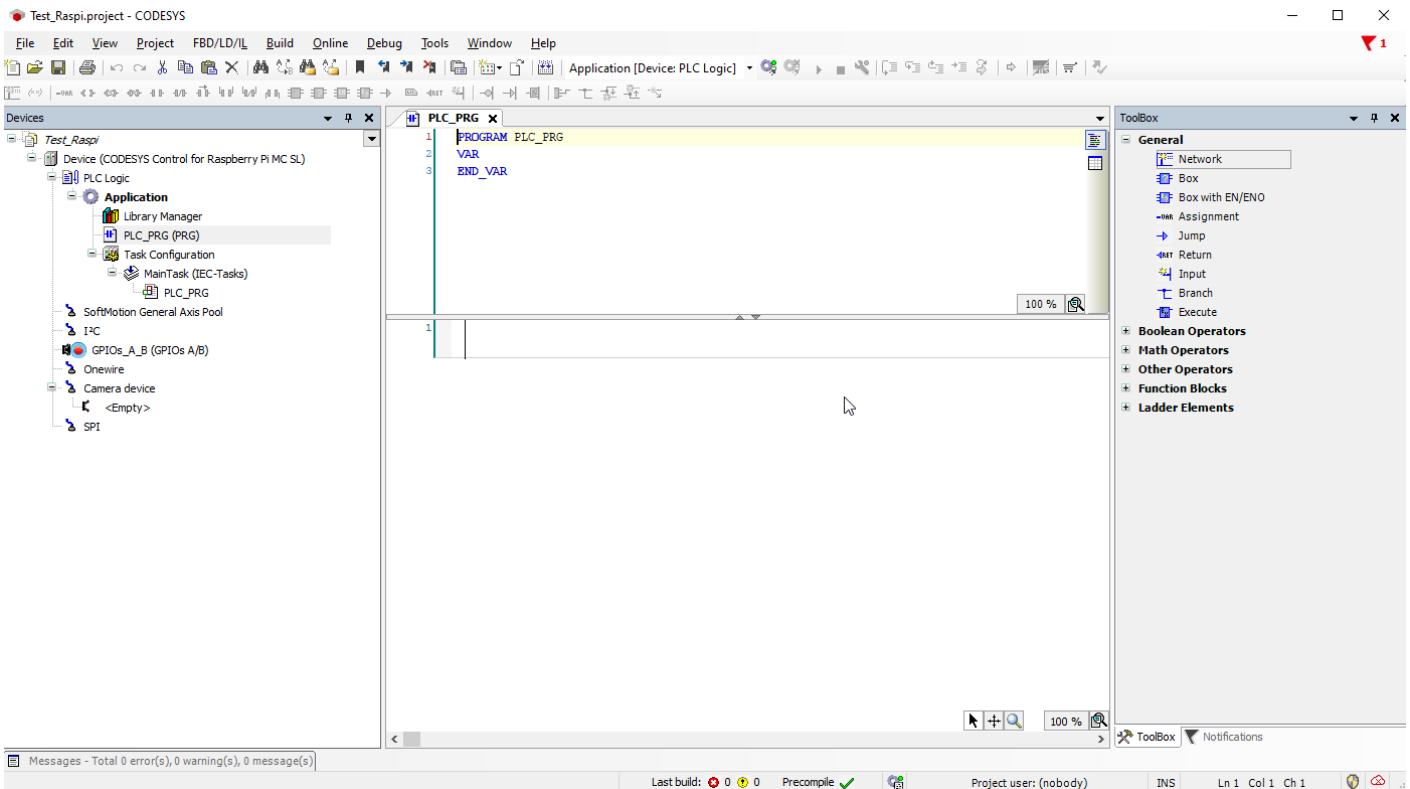
- On crée un Standard Project nommé Test\_Raspi
- Enregistré dans le dossier Codesys\Raspi à créer



- Pour Device, on choisit CODESYS Control for Raspberry Pi MC SL
- PLC\_PRG : en Ladder (LD)

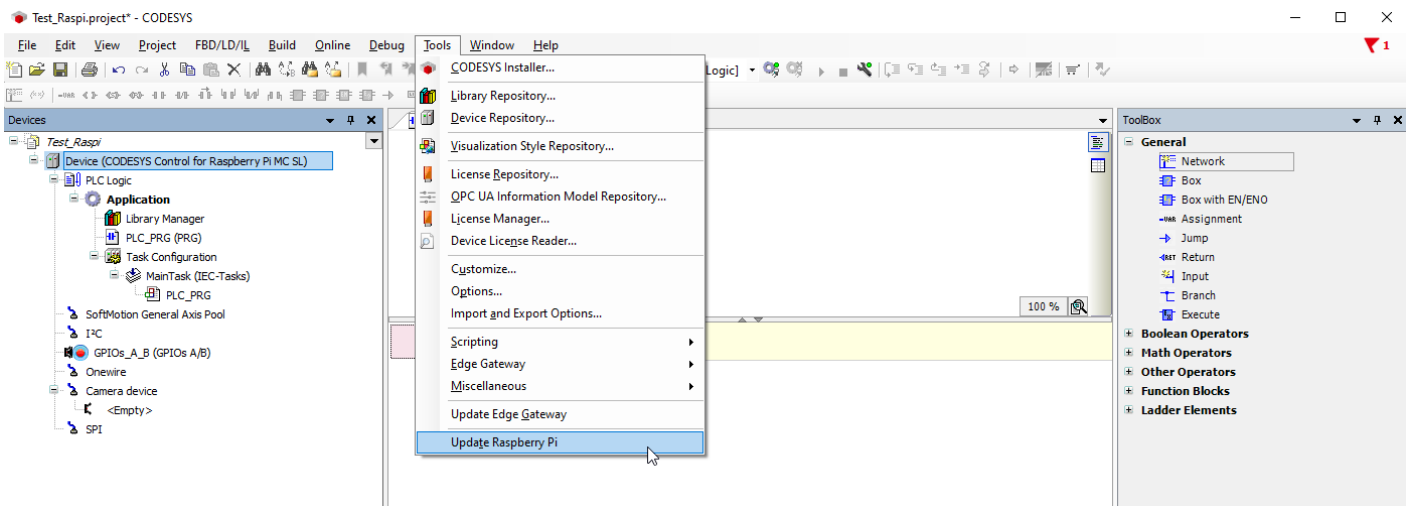


- On obtient l'arborescence de projet suivante :



# Update du Raspberry Pi

- On fait un update du Raspberry Pi dans Tools -> Update Raspberry Pi



- On met le username du Raspi : pi
- On met le mdp : xxxx
- On met l'adresse IP du Raspberry Pi : 192.168.1.15 (on peut également faire un scan)
- On clique sur Install
- On vérifie le System Info que des données apparaissent (bug sinon)
- On clique sur Configure
- La création d'un User peut être demandé avec un login et un mdp pour avoir l'autorisation de transférer du code sur l'automate. Perso, j'ai choisi le même login et mdp que pour le Raspi.

Raspberry Pi

▲ Login credentials

Username

Password

☐ SSH login based on key

▲ Select target

IP Address

▲ CODESYS Runtime Package

Version

Package directory

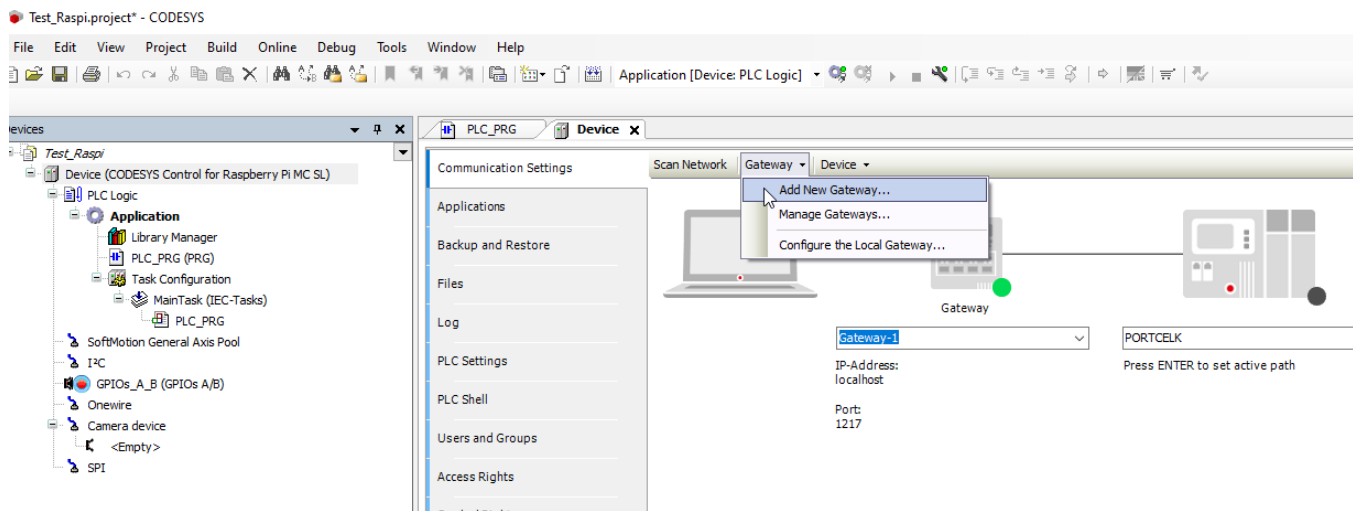
▲ Additional Packages

▲ System

▲ Runtime

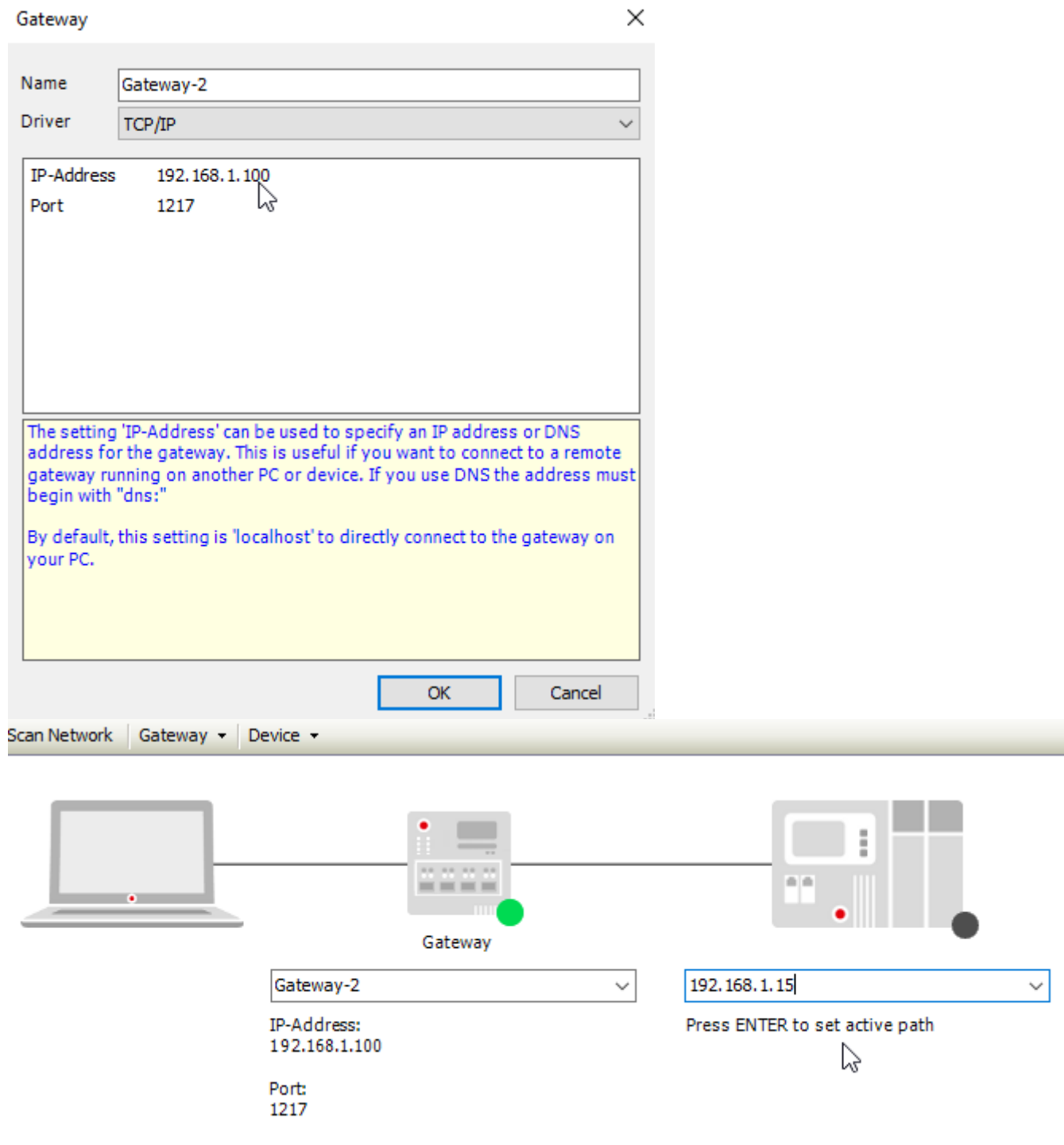
## Connexion et Gateway

- On double clique sur Device
- Il faut créer une nouvelle Gateway correspond à la carte réseau sur laquelle est branchée votre Raspberry Pi




- L'adresse IP de la carte Ethernet était dans cet exemple de 192.168.1.100 (à adapter selon votre config)
- On place l'adresse IP de votre Raspberry Pi à côté de la Gateway et l'on appuie sur la touche entrée





- On remet le login et mot de passe qui permet de transférer le code PLC sur la cible Raspberry Pi
- La cible passe au vert avec du texte en-dessous
- On est OK pour la suite

Device User Logon



You are currently not authorized to perform this operation on the device. Please enter the name and password of an user account which has got the sufficient rights.

Device name

Deviceaddress0364.1000.2DDC.C0A8.010F

User namephilippe

Password●●●●●●●●

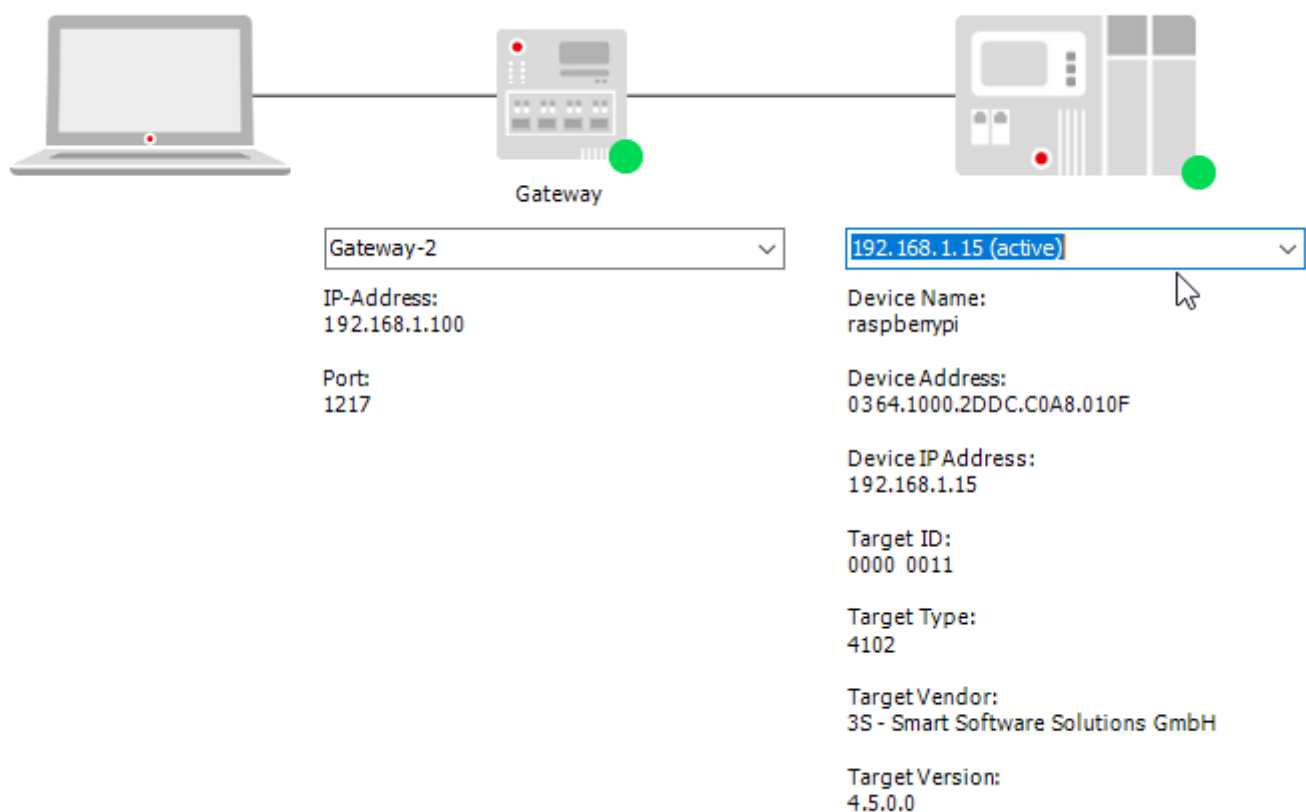
Operation:View

Object:"Device"

OK

Cancel

Scan Network | Gateway | Device

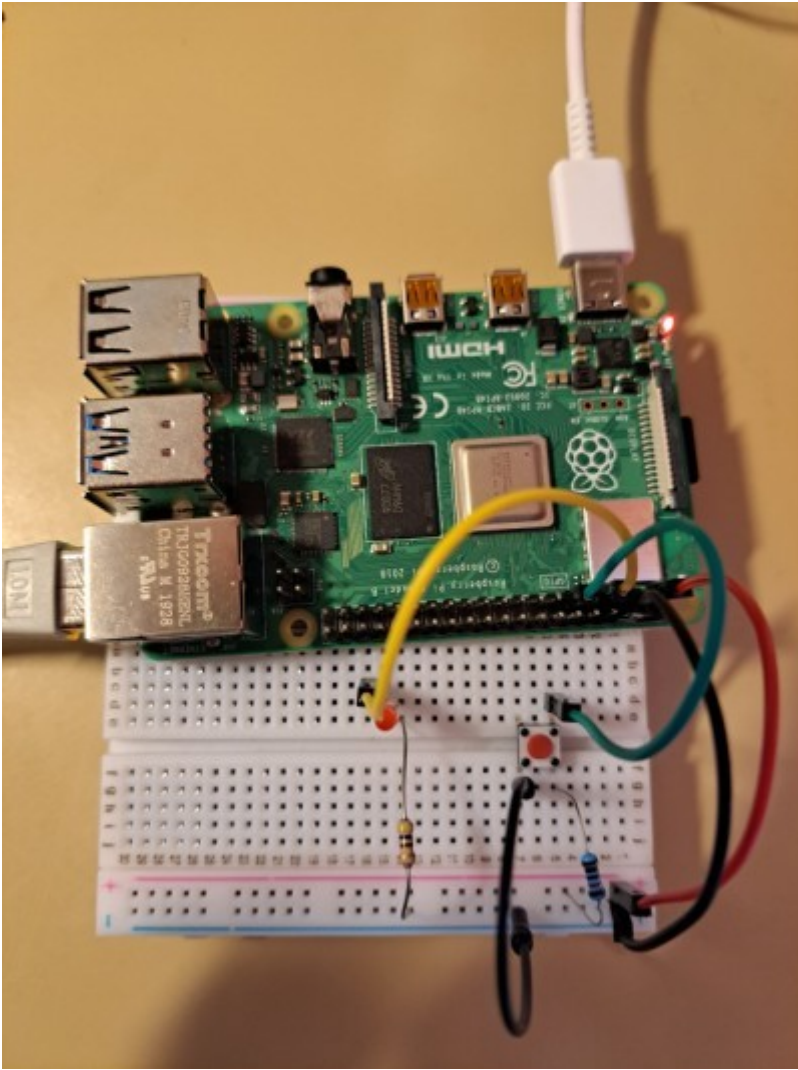


# Programme de test Raspberry Pi

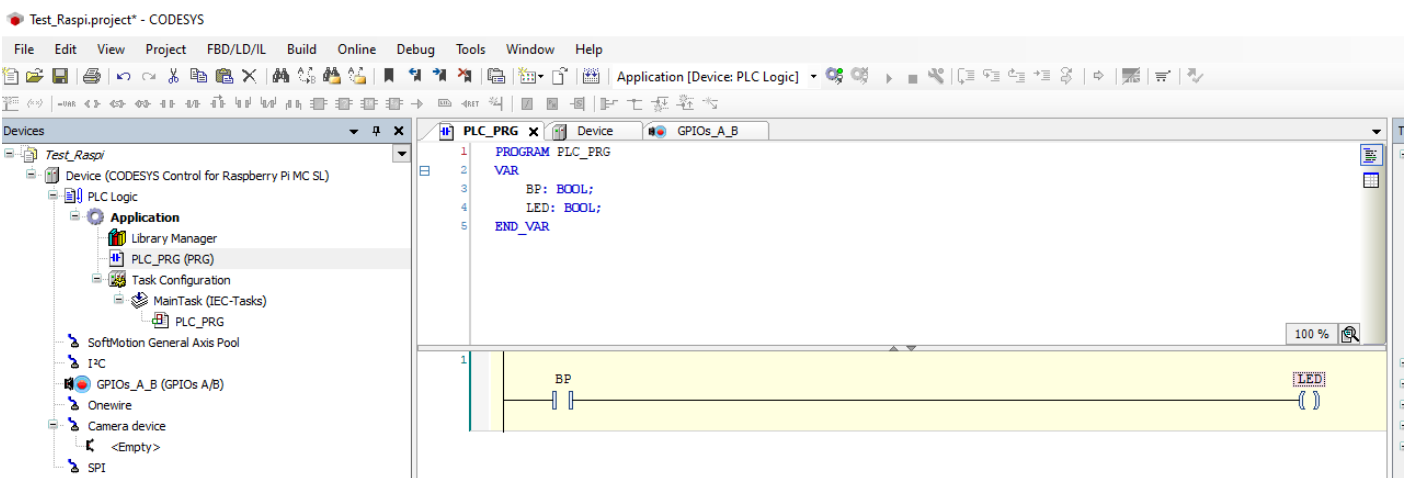
Nous souhaitons faire dans ce programme de test un pilotage de LED à partir d'un Bouton Poussoir :

- Le BP est branché en Pull-Up ce qui signifie qu'à l'état de repos, il envoie 3.3V, à l'état actif, il envoie 0V
- Le BP est branché sur la GPIO 17

- La Led est branchée sur la GPIO 4



- Voici le programme crée pour piloter la LED
- Comme le BP est actif à l'état de repos, on aura la LED qui sera allumée, si on appuie sur le BP, la LED s'éteindra



- Il faut maintenant affecter les GPIO aux variables BP et LED
- On configure GPIO 4 comme Output pour la LED

- On configure GPIO 17 comme Input pour le BP

Parameters						
Parameter	Type	Value	Default Value	Unit	Description	
GPIO4	Enumeration of BYTE	Output	not used		configuration of GPIO4	
GPIO17	Enumeration of BYTE	Input	not used		configuration of GPIO17	
GPIO18	Enumeration of BYTE	not used	not used		configuration of GPIO18	
GPIO22	Enumeration of BYTE	not used	not used		configuration of GPIO22	
GPIO23	Enumeration of BYTE	not used	not used		configuration of GPIO23	

- Dans I/O Mapping on affecte la variable BP au Bit 17 pour Digital Input

Test\_Raspi.project\* - CODESYS

File Edit View Project Build Online Debug Tools Window Help

Application [Device: PLC Logic]

Devices

Test\_Raspi

Device (CODESYS Control for Raspberry Pi MC SL)

PLC Logic

Application

Library Manager

PLC\_PRG (PRG)

Task Configuration

MainTask (IEC-Tasks)

PLC\_PRG

SoftMotion General Axis Pool

I²C

GPIOs\_A\_B (GPIOs A/B)

Onewire

Camera device

<Empty>

SPI

Parameters

I/O Mapping

IEC Objects

ation

Find

Filter Show all

Add FB for IO Channel...

Variable	Mapping	Channel	Address	Type	Unit
		digital inputs (GPIO0..GPIO31)	%ID0	DWORD	
		Bit4	%IX0.4	BOOL	
		Bit17	%IX2.1	BOOL	
		Bit18	%IX2.2	BOOL	
		Bit22	%IX2.6	BOOL	

Input Assistant

Text Search Categories

Variables

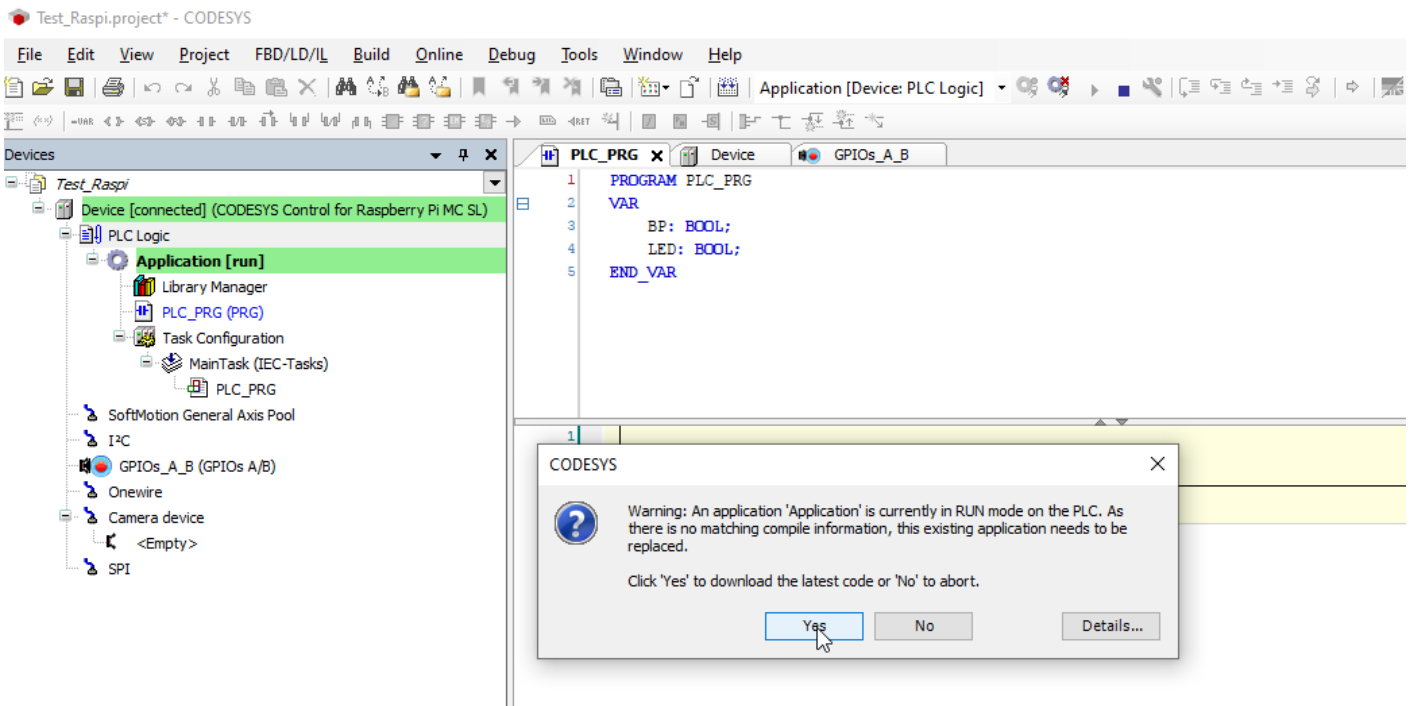
Name	Type	Address	Origin
Application	Application		
PLC_PRG	PROGRAM		
BP	BOOL		
LED	BOOL		
IoConfig_Glob...	VAR_GLOBAL		
SM3_Basic	Library		SM3_Basic, 4.11.0.0...
SM3_Math	Library		SM3_Math, 4.11.0.0...

- De manière équivalente on affecte la variable LED au Bit 4 pour Digital Output

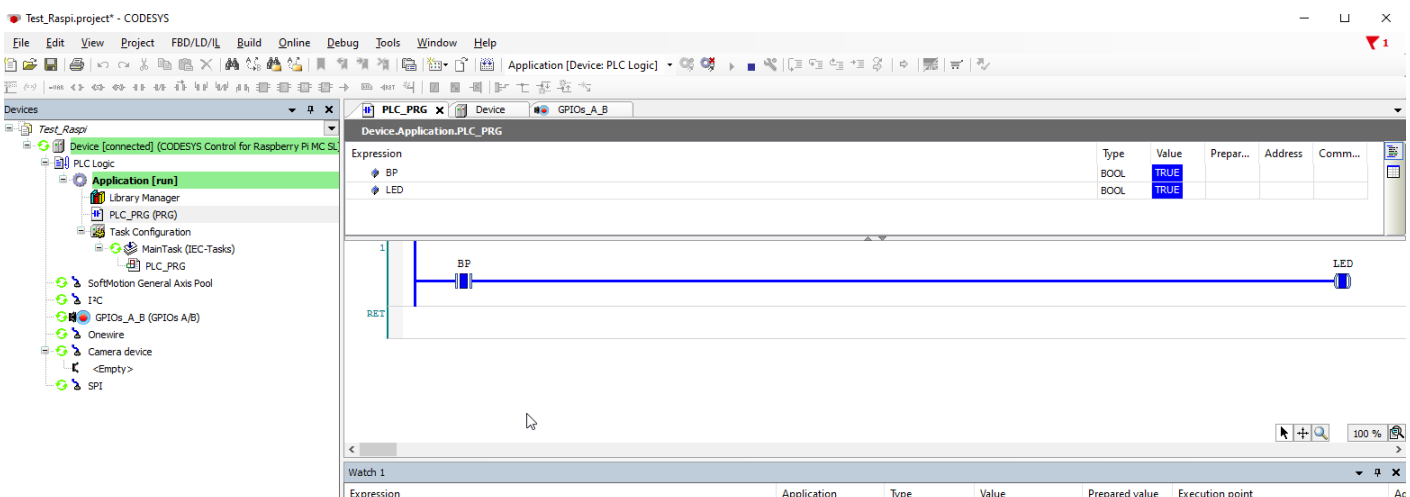
Parameters						
Variable	Mapping	Channel	Address	Type	Unit	
		digital inputs (GPIO0..GPIO31)	%ID0	DWORD		
		Bit4	%IX0.4	BOOL		
Application.PLC_PRG.BP		Bit17	%IX2.1	BOOL		
		Bit18	%IX2.2	BOOL		
		Bit22	%IX2.6	BOOL		
		Bit23	%IX2.7	BOOL		
		Bit24	%IX3.0	BOOL		
		Bit25	%IX3.1	BOOL		
		Bit27	%IX3.3	BOOL		
		Bit28	%IX3.4	BOOL		
		Bit29	%IX3.5	BOOL		
		Bit30	%IX3.6	BOOL		
		Bit31	%IX3.7	BOOL		
		digital outputs (GPIO0..GPIO31)	%QD0	DWORD		
Application.PLC_PRG.LED		Bit4	%QX0.4	BOOL		
		Bit17	%QX2.1	BOOL		
		Bit18	%QX2.2	BOOL		

# Test

- On fait Generate Code (F11)
- On fait Login (Alt+F8)



- On fait START (F5)
- On remarque que la LED est automatiquement allumée, ce qui est conforme au comportement attendu
- Quand on appuie sur le BP, la LED s'éteint (conforme aussi)
- Vous venez de créer un Automate avec un Raspberry Pi -> Bravo !



Revision #3

Created 29 June 2023 17:11:02 by Philippe Celka

Updated 4 July 2023 09:17:11 by Philippe Celka