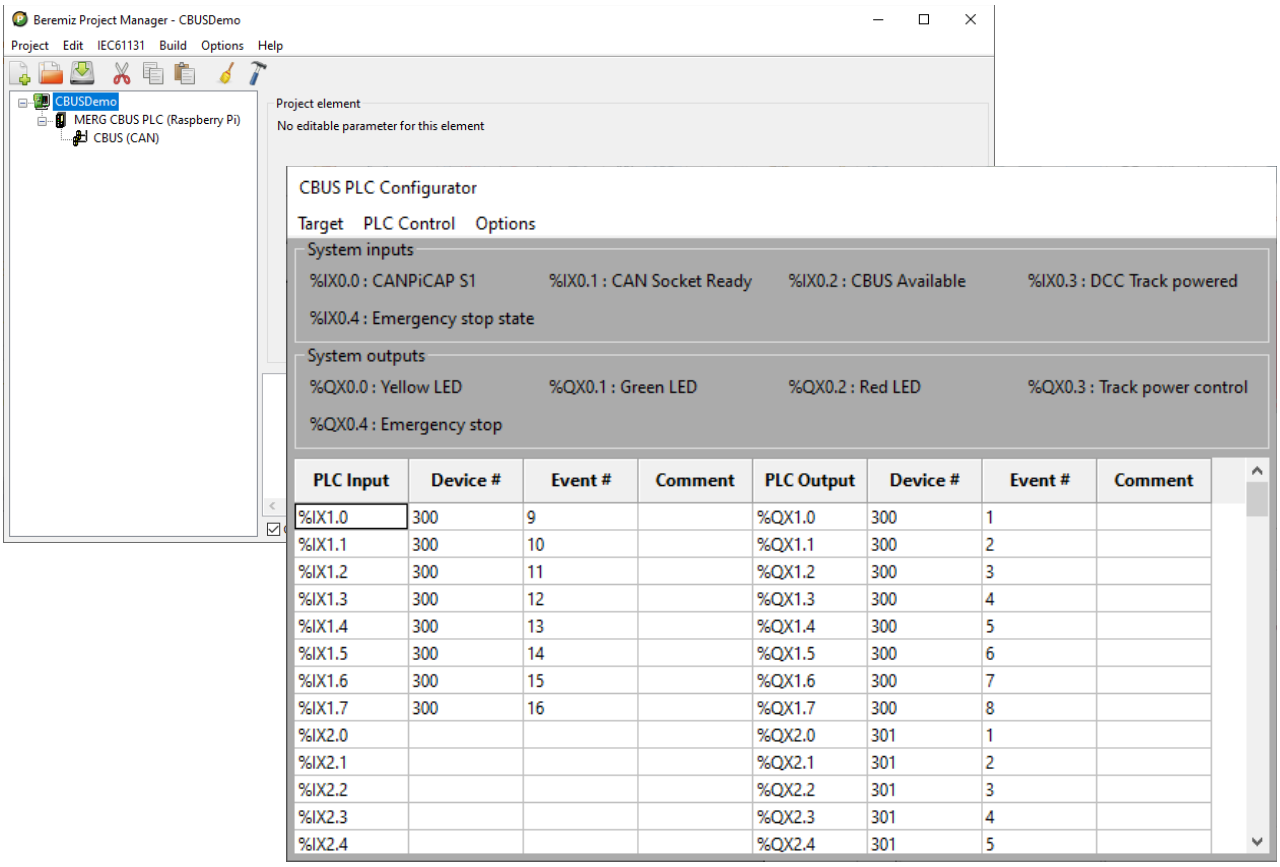
BEREMIZ PROJECT MANAGER

MERG CBUS PLC TARGET

CONFIGURATION / INSTALLATION MANUAL



PLC Input	Device #	Event #	Comment	PLC Output	Device #	Event #	Comment
%IX1.0	300	9		%QX1.0	300	1	
%IX1.1	300	10		%QX1.1	300	2	
%IX1.2	300	11		%QX1.2	300	3	
%IX1.3	300	12		%QX1.3	300	4	
%IX1.4	300	13		%QX1.4	300	5	
%IX1.5	300	14		%QX1.5	300	6	
%IX1.6	300	15		%QX1.6	300	7	
%IX1.7	300	16		%QX1.7	300	8	
%IX2.0				%QX2.0	301	1	
%IX2.1				%QX2.1	301	2	
%IX2.2				%QX2.2	301	3	
%IX2.3				%QX2.3	301	4	
%IX2.4				%QX2.4	301	5	

Contents

1 - Introduction

Beremiz Project Manager is a software suite, built around Beremiz, the Open Source IDE for IEC61131-3 based machine automation. It acts as a "top level" management allowing you to create powerful Programmable Logic Controller applications on multiple targets, programmed in any of the IEC61131-3 languages :

- Ladder Diagram
- Function Block Diagram
- Stuctured Text
- Instruction List
- Sequential Function Chart

The software suite is operated from Beremiz Project Manager Manager application, which acts as a control tower. The Manager is in charge of project generation and configuration (including I/O configuration), which is then send into Beremiz IDE where you can write the PLC programs.

Once IEC61131-3 code has been written, Beremiz Project Manager Manager will generate all project files needed by the compiler. With a simple click, your PLC program will be compiled and sent to your target, wtihout needing to write a single line of code within Arduino IDE.

1.1 - IMPORTANT REMINDER

It must be understood that applications written using Beremiz and/or Beremiz Project Manager shall not be used in any safety-critical application without a full and competent review, from a certified authority.

Beremiz Project Manager is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY, and without even the implied warranty of merchantability or fitness for a particular purpose.

2 - Software installation

Three set of files are needed in order to build Beremiz PLC application on the Raspberry Pi for the MERG CBUS PLC :

- The target files, which are source files and project files needed along the PLC application code. These files will be compiled and linked automatically under control of Beremiz Project Manager
- The target plugin, which is loaded automatically in Beremiz Project Manager when the Raspberry Pi Pico target is selected when creating a new project
- The compiler tools and libraries, which must be installed on the Raspberry Pi target

The two first sets of files (target files and target plugin) are part of the Beremiz Project Manager package and they don't need to be installed separately, as they are copied automatically when you install Beremiz Project Manager. Please refer to Beremiz Project Manager documentation for details about the installation processus.

The PLC compiler tools must be installed and configured separately from the Beremiz Project Manager. Please follow instructions from the following chapter before using Beremiz Project Manager to build a CBUS PLC project.

2.1 - Preparation of Raspberry Pi target

We assume that you know how to use Raspberry Pi OS for the following steps. The CBUS PLC requires the MERG Kit 86 to installed on it before the CBUS PLC can be configured successfully.

- Prepare a SD card with Raspberry Pi OS (32-bit version is recommended as the CBUS PLC has not yet been tested on 64-bit OS). You don't the complete image with all "recommended software", just make sure that GCC compiler chain is installed on it.
- Write down your user name and the password, you will need them later to configure Beremiz Project Manager
- Create a new directory on the SD card. This directory will receive the various files needed to compile your CBUS PLC project. **Make sure there is no space in the directory name**.
- In this directory, create a sub-directory. As for the parent one, make sure there is no space in its name.

  (For our examples below, we will use ~/CBUSPLC/PLC)
- Write down the resulting path, you will need it to configure the CBUS PLC plugin in the Beremiz Project Manager
- Open the Raspberry menu on desktop (using the raspberry icon)
- Open "Raspberry Pi configuration" in Preferences
- Click on Interfaces
- Activate the SSH toggle switch (the button shall be on the rightmost side)
- Save the configuration

2.1.1 - Configuring CAN driver for the MERG Kit86

CBUS PLC uses a CAN driver which is part of the Raspberry Pi OS. However, this driver is not active by default and requires a configuration file to be modified in order to have the MERG Kit 86 board to be recognized.

- Open a terminal window

- Enter the following command **sudo nano /boot/config.txt**
- Add the following lines to the file which opened (first line is a comment)

  #Enable CANPiCAP interface

  dtparam=spi=on

  dtoverlay=mcp2515-can0,oscillator=16000000,interrupt=25

- Save the file and exit nano, then reboot the Raspberry Pi

  Once the computer has rebooted, enter the following line in terminal

  ls /sys/bus/spi/devices/spi0.0/net

The computer shall display **can0** in return. If not, either you have made a mistake in the config.txt file or the CANPiCAP board is not working/is not connected properly.

You have to make sure that the board is recognized properly as **can0** otherwise the CBUS PLC can not work at all.

2.1.2 - WiringPi installation

The library WiringPi is required to give acccess to the LEDs and pushbutton located on the CANPiCAP board. It was delivered with early versions of Raspberry Pi OS, but latest versions do not include it anymore as the original author stopped working on it.

There exists two possibilities to install WiringPi :

- get it from the non-official github which maintains it (by non official, we mean not the github from the author)
- get it on the Beremiz Project Manager folder on Github

The best solution is to try to download the library from github first using the following command :

git clone https://github.com/WiringPi/WiringPi.git

Go into the folder " Files for targets / RPi CBUS PLC" and transfer the WiringPi folder content into the Raspberry Pi computer you want to use for the CBUS PLC (place the content in a directory named "WiringPi")

Once the files are downloaded on the Raspberry Pi, type the following commands in a terminal :

cd WiringPi

chown +x build (only needed if you downloaded the files from Google Drive)

./build

Wait until the script indicates "All done". The WiringPi library is now installed on your system.

You can use **gpio readall** command to check that everything is now installed properly (the console shall display a table with GPIO available on your system)

2.1.3 - Installation of CBUS PLC Supervisor and scripts

The CBUS PLC is intented to run without screen, mouse and keyboard. It is controlled remotely from Beremiz Project Manager using a tool, called "supervisor", which communicates with the development computer over Ethernet or WiFi.

Download files located in Files for Targets / RPi CBUS PLC / Supervisor on your Raspberry, in the base directory you have created in chapter 2.1 (~/CBUSPLC in our example)

You should now see the following files in this directory :

- compile_plc.sh
- plc_supervisor
- start_cbus.sh
- start_plc.sh

Enter the following command in the terminal

cd ~/CBUSPLC (use directory name you have created here)

sudo chown +x *.sh

sudo chown +x plc_supervisor

The last step is to check that the scripts and remote supervisor are now working as expected.

Enter the following command in the terminal :

./start_cbus.sh

You should normally see the can0 interface (created previously) being displayed.

Check now the supervisor :

./plc_supervisor

You should see a message telling you that the PLC supervisor is now running. The supervisor is now waiting for commands waiting from Beremiz Project Manager.

Configuration is now finished on the Raspberry side !

2.1.4 - Network configuration

Communication between Beremiz Project Manager and the CBUS PLC is done using network (Ethernet or WiFi). It is highly recommended that you use static IP addresses for the CBUS PLC (even if DHCP would work) as it makes the network management much easier.

In order to set a static IP address on the PLC, open a terminal and enter **sudo nano /etc/dhcpcd.conf** (take care : if you make a mistake entering the filename – which is quite easy, you have been warned – you will see an empty file! In that case, close nano and check what you have entered)

In the file, add the following lines (you can also uncomment and changes the lines provided as an exemple)

interface eth0 (use wlan0 for WiFi)

static ip_address=192.168.0.41/24 (enter the IP address you want, with the correct mask)

static routers=192.168.0.1 (use your router IP address if you have one)

Save the file, close the nano editor and restart the Raspberry Pi to take the changes into account

2.1.5 - Making the supervisor start automatically

TODO

2.2 - Preparation of Beremiz Project Manager

Installation of Beremiz Project Manager is described in the User's Manual (which is mostly copying the whole application directory on a Windows machine and install eventually the C++ runtimes needed by the application)

In order to communicate with CBUS PLC, WinSCP must be installed on the Windows PC running Beremiz Project Manager.

The Windows installer is available here : https://winscp.net/eng/downloads.php#putty

We have also put a copy of this installer in the release on Github (in "Files for targets / MERG CBUS PLC" folder)

Always download the installer from one of these sources. Don't trust any other source, as it exists fake copies of WinSCP which install malware and/or unwanted applications on your system.

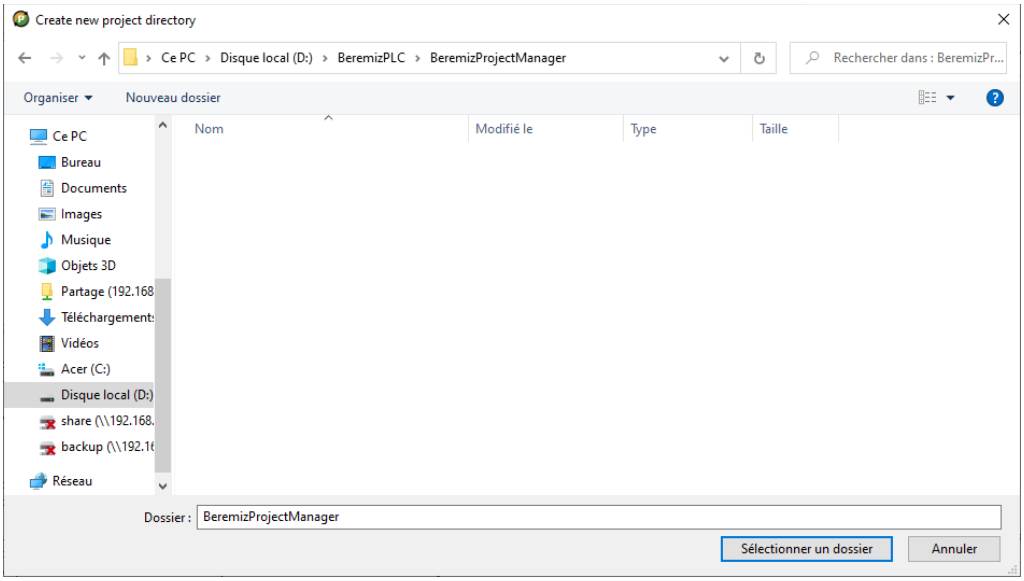
The version installed from the Microsoft Store is a pain to use with Beremiz Project Manager (as the executables are hidden in user directory), so don't use this one.

3 - CBUS PLC project creation

3.1 - Creating a new project

A Beremiz project is a set of files located in a folder (folder's name being the the project name). Basically, nothing useful can be done until a project is created or loaded in the manager software.

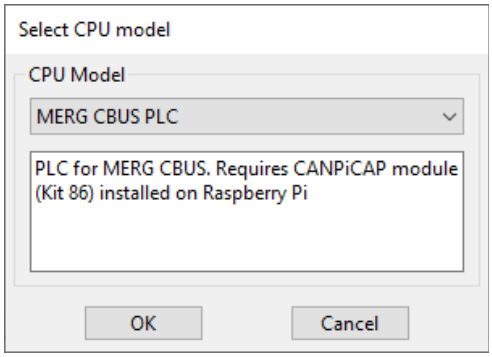
Click on "New project" command in File menu in order to open the dialog below.



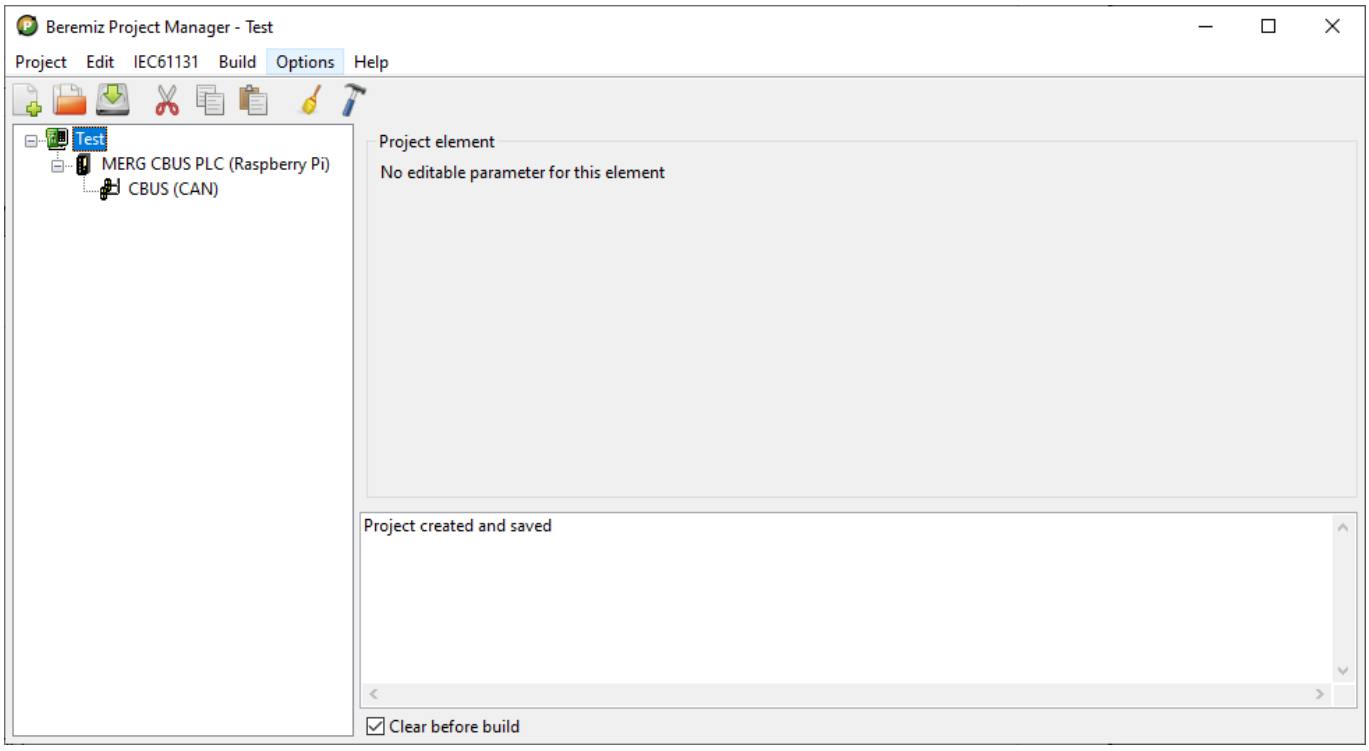
Projects can be created anywhere on the hard disk (you don't have to use the same location as on the previous screenshot). Using this dialog, select the folder in which you want to create the project. Click then on "New folder" button and enter the desired name for the project. Make sure that selection bar is on the newly created project.

Click then on "Select folder" button (bottom right of the dialog) to confirm project creation and close the dialog.

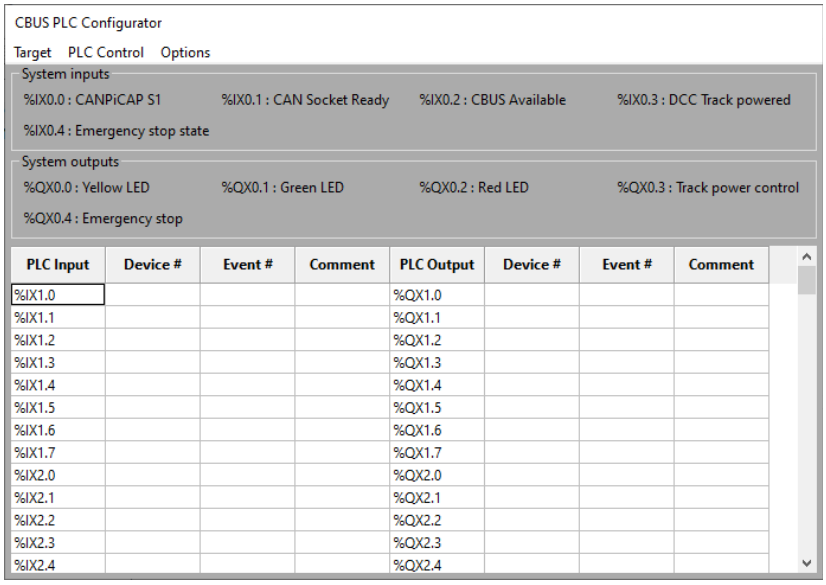
Once the project is created, the following dialog appears. Select "MERG CBUS PLC" for the CPU model.



Click on OK to confirm the CPU model. The project will then appear in the main tree of Beremiz Project Manager.



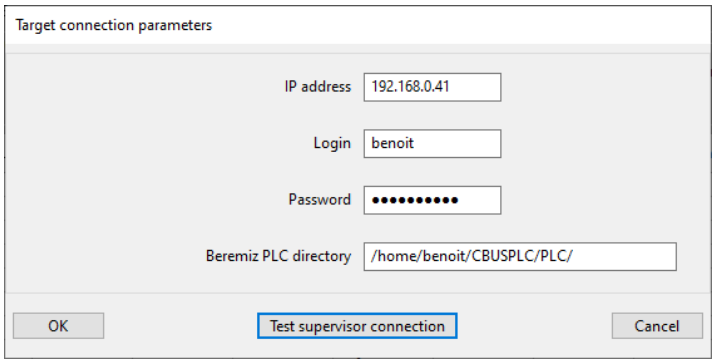
3.2 - Remote target configuration



Click on Target then choose Connection Properties to open the following dialog. Enter in this dialog the various informations needed by Beremiz Project Manager to locate the CBUS PLC target. These informations are the one created in §2.1. The login and password are the one created when you configure Raspberry Pi OS at the first startup.

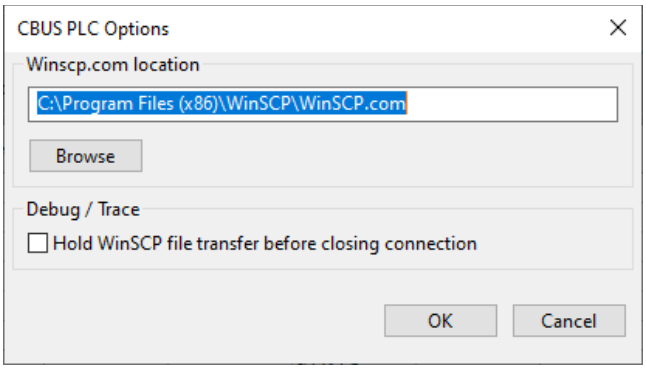
Note that the "/" is required at the end of the Beremiz PLC directory.

You can also use this dialog to check the communication with the PLC supervisor running on CBUS PLC target. Click on "Test supervisor connection" : if the supervisor is running and can be contacted at the address given in "IP address" field, a confirmation message box will be displayed.



The configuration editor is also used to define the location of WinSCP.com. The location of the file must be entered before any operation on the target is performed, including the framework generation.

Note that you must be point on WinSCP.com file, not WinSCP.exe.

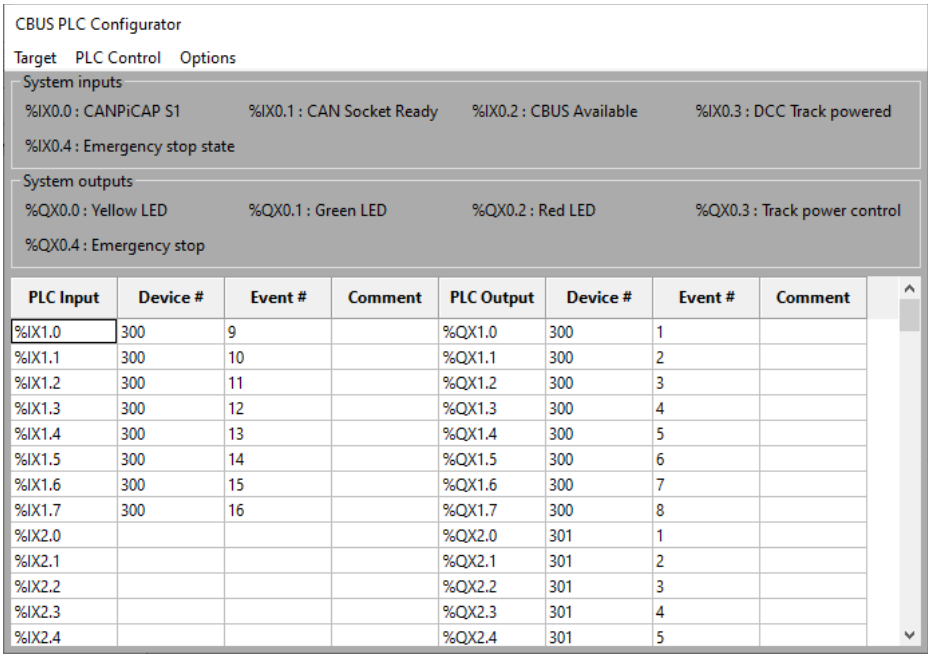


The "Hold WinSCP transfer" option is used mainly for debugging the communication with the target, in case file transfer (for framework generation or before build) fails. When this box is checked, the file transfer operation will stop after completion or error, allowing you to see any possible error messages.

The parameters given in these two dialogs should normally be entered only one time, just after the project has been created.

3.3 - Inputs/Outputs configuration

The MERG CBUS PLC allows up to 128 digital inputs and 128 digital outputs located on CBUS modules. These I/Os are related to CBUS "events" associated with a CBUS "Device ID".



PLC Input	Device #	Event #	Comment	PLC Output	Device #	Event #	Comment
%IX1.0	300	9		%QX1.0	300	1	
%IX1.1	300	10		%QX1.1	300	2	
%IX1.2	300	11		%QX1.2	300	3	
%IX1.3	300	12		%QX1.3	300	4	
%IX1.4	300	13		%QX1.4	300	5	
%IX1.5	300	14		%QX1.5	300	6	
%IX1.6	300	15		%QX1.6	300	7	
%IX1.7	300	16		%QX1.7	300	8	
%IX2.0				%QX2.0	301	1	
%IX2.1				%QX2.1	301	2	
%IX2.2				%QX2.2	301	3	
%IX2.3				%QX2.3	301	4	
%IX2.4				%QX2.4	301	5	

The 128 inputs and 128 outputs are refreshed automatically by the CBUS PLC engine, so they are used directly using their address (%IX or %QX) in the PLC application.

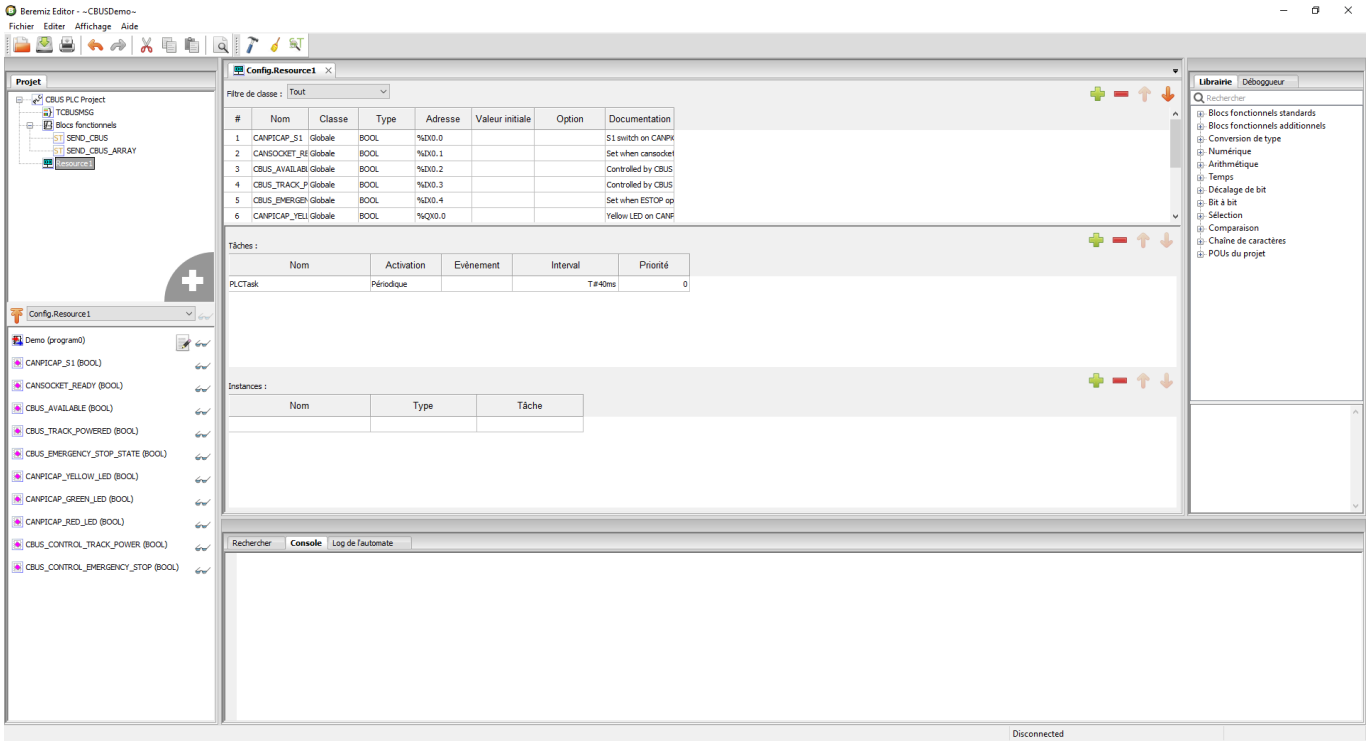
Device ID and Event numbers are entered into CBUS modules using the MERG CBUS Configuration Utility. Note that CBUS PLC requires the CBUS modules to operate in FLiM mode. Please refer to MERG CBUS documentation and tutorials for detailed information about the CBUS configuration process.

CBUS PLC also allows direct CBUS communication using dedicated Function Blocks, which allow to send any CBUS message directly on the CBUS.

Note that you can change I/O configuration at any moment, even after you have started to write the PLC program. The configuration parameters for the PLC program are generated automatically each time a Build command is executed on Beremiz Project Manager.

3.4 - Writing the PLC program

Once your PLC project is created or loaded in Beremiz Project Manager you can open the Beremiz programming environment. To do that, click on "Open Beremiz Editor" command in "IEC61131" menu. This will launch Beremiz (note that first startup of Beremiz after installation can take a few minutes).



When you create a new CBUS PLC project, you will notice that a few elements are created automatically :

- global variables for the CBUS system I/Os
- two function blocks
  - SEND_CBUS
  - SEND_CBUS_ARRAY
- a variable type called TCBUSMSG

VERY IMPORTANT : never remove / delete these two function blocks or the TCBUSMSG type from your project! Once deleted, there is no way to restore them in a project. If they are deleted, you will be unable to use user defined CBUS messages in your PLC program.

Please refer to Beremiz Project Manager manual for details about the project configuration and writing the PLC code.

3.4.1 - Generation of PLC application code

Once you have written your PLC program, you will need to generate the target code. This step transforms your PLC program in a set of C source code files.

THIS STEP MUST BE EXECUTED EACH TIME THE PLC PROGRAM HAS BEEN MODIFIED, BEFORE IT CAN BE COMPILED !!

If you do not generate the PLC code, it will either be impossible to compile your PLC project or the changes done in the PLC code will be ignored.

To generate the PLC code, click on the "Build" button on top toolbar :

Note that Beremiz generates PLC code in two steps :

- All programs in the project are translated into Structured Text language, whatever the language used (Ladder Diagram, Function Block Diagram, etc...)
- The global program obtained at previous step is translated into C code using MatIEC tool

When code generation is fisnihed, make sure that no error is being reported by Beremiz. You must see "C code generated successfully" in the Beremiz console before you can compile the PLC program. If you see the message "PLC code generation failed" in the console, you have to check your program syntax against IEC61131-3 rules.

Do not hesitate to use the "Build" button regularly to check the syntax of your program, especilly if you are a beginner in PLC and/or IEC61131-3. Most of the errors are very easy to understand if you read carefully errors messages from Beremiz.

Keep in mind that IEC61131-3 languages are less permissive than Python or even C. One of most common problem when you begin in PLC programming is to think that the code generator will understand what you want to do because it is clear in your mind.

IEC61131-3 uses quite strict syntax to be sure that what is written describes precisely what it is intended to be done. For example : don't try to copy a BYTE variable inside a SINT variable. The code generator will report an error (not a warning), as the two variables dont have the same size and the system does not know what it has to do to generate the missing part (in that case, it does not assume it should use a null field. You have to write that YOU want to do it, using a typecast operator for example)

The "Show IEC code" button allows to see your whole PLC program wirtten entirely in Structured Text :

The "Clean target" button erases all files created previously (needed only if you want to force a rebuild) :

4 - Target files generation

4.1 - Framework generation

A PLC application requires two sets of files in order to run correctly :

- the PLC application (written in Beremiz editor)
- the PLC framework, which contains a huge set of functions required by the IEC61131-3 norm

The PLC framework is tightly associated with the target itself (in other terms, each target can require a different framework).

The framework for Raspberry Pi Pico is provided with Beremiz Project Manager and you have nothing special to do to install it on your machine.

However, the framework must be copied along with your PLC application, since this last one requires the PLC framework to run properly. Moreover, the framework may have to be modified depending on options related to the target itself (these options are defined by the plugin editor).

The biggest difference between the PLC framework and the PLC application is that the framework is not supposed to be modified once the target properties have been defined. That's why the framework is copied

independantly of the PLC application itself : it is normally copied only one time in your project.

The PLC framework is generated by calling the "Generate framework" from the "IEC61131".

Don't forget to call this command at least one time before the first compilation, otherwise, the application will be impossible to compile !

In most case, the PLC framework can be generated at any moment, as long as it has been generated BEFORE the first build. The best practice is however to generate it before starting to work on the PLC code in Beremiz editor (as the PLC project may need informations from the framework), just after the project has been created.

You can call this command as many time as you want during project creation and edition, as it will never affect the PLC source code itself.

5 - Building project

Once the PLC application code and framework have been generated, it becomes possible to build the PLC application.

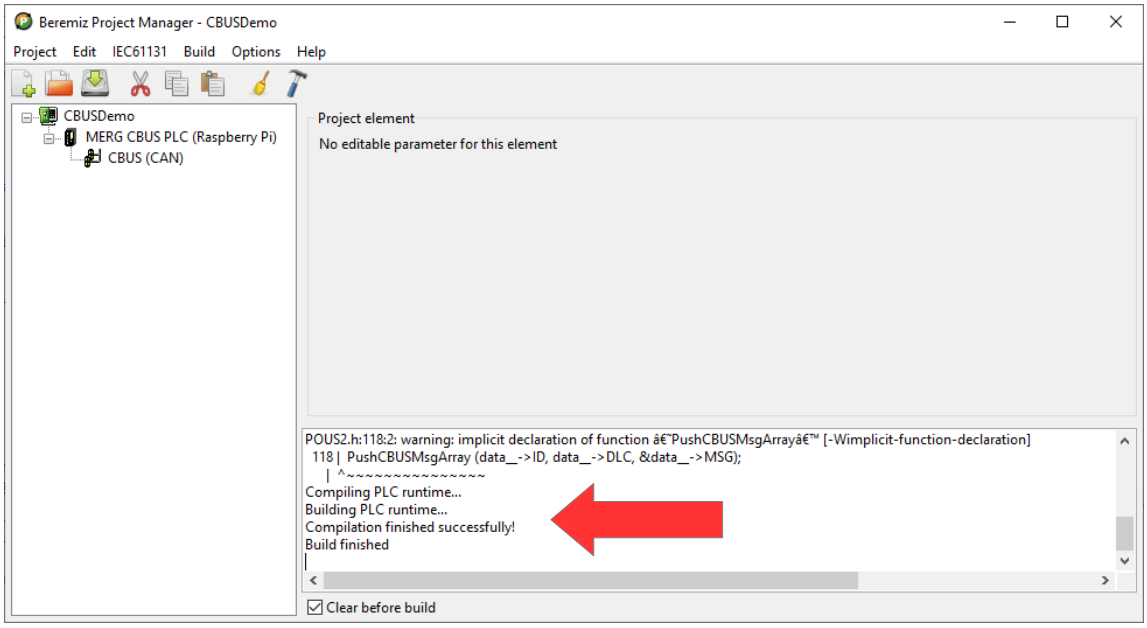
The "Build" command performs the following operations :

- Generation of I/O configuration files (based on the configuration declared in the I/O configuration editor)
- Copy of C generated files generated by Beremiz editor
- Copy of C "extensions" (Function Blocks written by user in C)

Before calling the Build command, it is **MANDATORY** to be sure that C files have been generated correctly (no errors reported) by Beremiz, and that PLC framework has been generated too (see previous chapters)

The Build command in the Build menu will perform automatically the tasks described herebefore. That's why you need to declare first the network location of CBUS PLC on the netwrok using the dialog described in §3.2.

Once the files have been copied on the CBUS PLC target, the supervisor will launch the compilation process on the target itself. You can see the progress of the compilation on the log window of the Beremiz Project Manager.



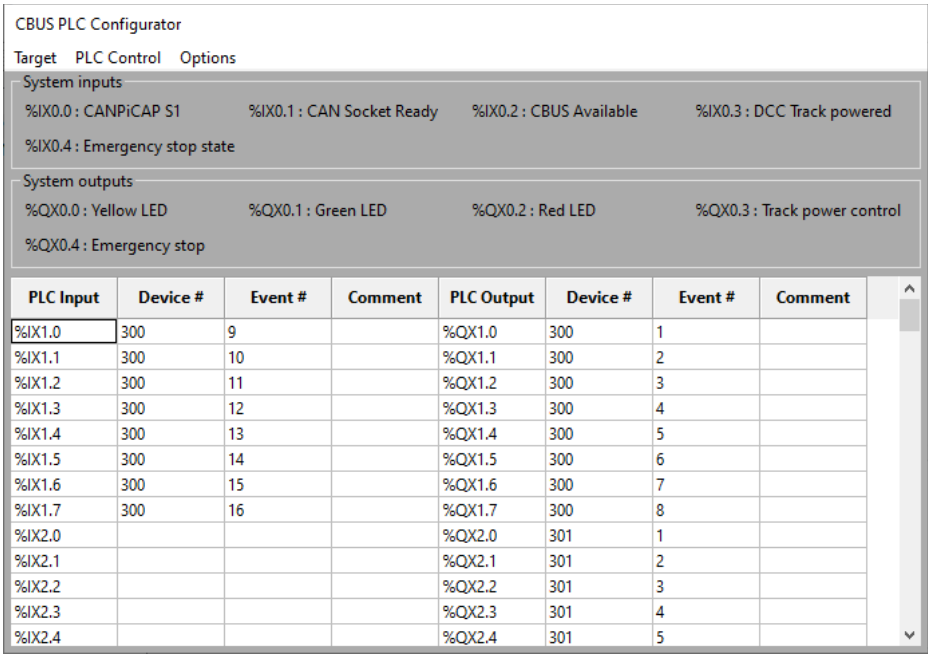
The compilation shall end with the message "Compilation finished successfully". If you don't see this message, it means that an error has been found and compilation has been aborted. Check the log window to see the error messages and identify the cause of the problem.

6 - Start / Stop PLC application

Once your PLC application has compiled without errors, the PLC runtime can be started. Note that you can perfectly compile a new PLC application while the PLC is running.

The PLC application execution is controlled by the supervisor. It can be started or stoppped at any moment. It is also possible to make the PLC application start automatically when the Raspberry Pi is started.

To control PLC application, open the CBUS PLC configuration dialog and use the PLC Control menu.



PLC Input	Device #	Event #	Comment	PLC Output	Device #	Event #	Comment
%IX1.0	300	9		%QX1.0	300	1	
%IX1.1	300	10		%QX1.1	300	2	
%IX1.2	300	11		%QX1.2	300	3	
%IX1.3	300	12		%QX1.3	300	4	
%IX1.4	300	13		%QX1.4	300	5	
%IX1.5	300	14		%QX1.5	300	6	
%IX1.6	300	15		%QX1.6	300	7	
%IX1.7	300	16		%QX1.7	300	8	
%IX2.0				%QX2.0	301	1	
%IX2.1				%QX2.1	301	2	
%IX2.2				%QX2.2	301	3	
%IX2.3				%QX2.3	301	4	
%IX2.4				%QX2.4	301	5	

The supervisor will check that PLC runtime has started/stopped successfully and will report the status to Beremiz Project Manager.

7 - CBUS PLC target limitations

The Beremiz Project Manager suite fully complies with IEC61131-3 specifications. However, because of hardware limitations related to the Raspberry Pi platform, it is not possible to use the following features in PLC code :

- "Time of day" related function blocks
- RETAIN variables
- No "live debugging" between Beremiz and physical target (for now...)

8 - Document revisions

Date	Auteur	Version	Description
23/10/2022	B.Bouchez	1.0	First version

06/11/2022	B.Bouchez	1.1	Added correct path to dhcpcd.conf (chapter 2.1.4)
06/01/2024	B.Bouchez	1.2	Update after release of Beremiz Project Manager on Github

Prepared with OpenOffice Writer software.