pbs-RPI-BBB

pbssoftLogic runtime for RPI and BeagleBoneBlack

Ver 1.0

Dec-2024

1. Installation

This document describes how to use pbsSoftLogic for the Raspberry PI and BeagleBone Blackboard Core CPU Board.

pbsSoftLogic can use the following RPI-BBB resources :

- GPIOs
- Watch Dog
- Serial Ports
- CAN IO Card with pbsCAN Protocols

Launch the Linux terminal and install the following software:

```
apt-get update
apt-get install openssh-server
```

edit /etc/ssh/sshd_config by nano utility and change PermitRootLogin to yes:

```
PermitRootLogin yes
```

Save the file and restart the RPI-BBB.

You can connect to the RPI-BBB using an SSH client software such as Putty and Filezilla.
If you want to use Email Publishing Driver, SQLServer connection by TDS and MQTT in pbsSoftLogic, connect to RPI-BBB as root and install the following software, otherwise you do not need to install them.

for Email Publishing Driver install following modules on the RPI-BBB:

```
apt-get install  curl
apt-get install  python3-pycurl
apt-get install  libcurl4-openssl-dev
```

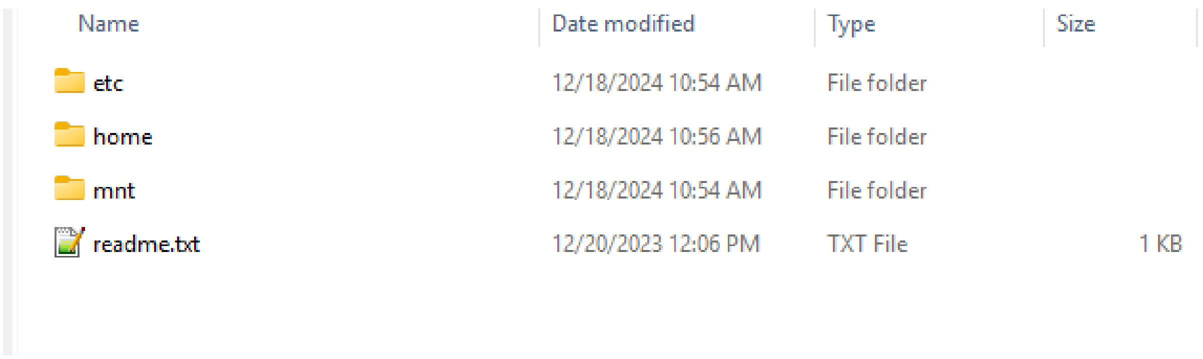
for SQLite Driver install following modules on the RPI-BBB:

```
apt-get install   freetds-dev
apt-get install   freetds-bin
```

for MQTT Driver install following modules on the RPI-BBB:

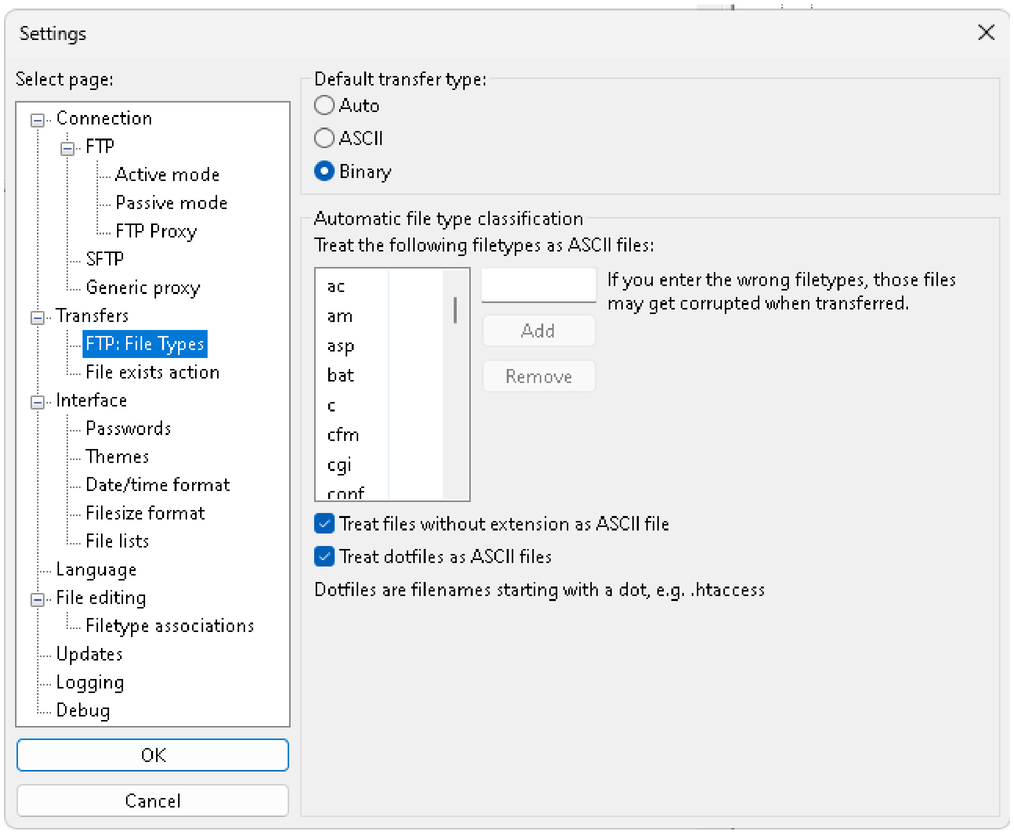
```
apt-get install mosquitto-clients
apt-get install mosquitto-dev
apt-get install mosquitto
apt-get install libmosquitto-dev
```

Unzip the pbsSoftLogic_ RPI64.zip folder, which is the pbsSoftLogic runtime kernel for RPI 64 Bit , and transfer it to the controller as follows:



Name	Date modified	Type	Size
etc	12/18/2024 10:54 AM	File folder	
home	12/18/2024 10:56 AM	File folder	
mnt	12/18/2024 10:54 AM	File folder	
readme.txt	12/20/2023 12:06 PM	TXT File	1 KB

Transfer content of the home folder to home folder of RPI-BBB:. Use filezilla Client for transferring to the controller.
When using filezilla, make sure the transfer type is set to binary, otherwise it will damage the transferred files. It is on Auto by default.



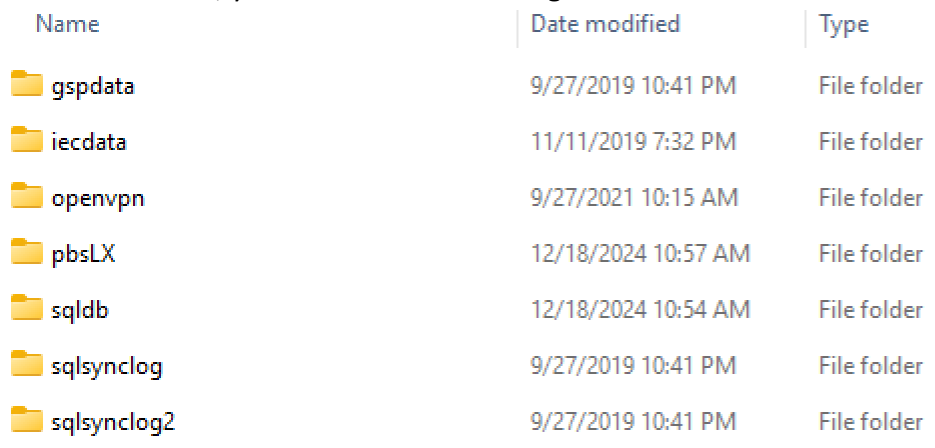
Default transfer type:
Auto
ASCII
[✓ Binary]

Automatic file type classification
Treat the following filetypes as ASCII files:
ac
am
asp
bat
c
cfm
cgi
conf

If you enter the wrong filetypes, those files may get corrupted when transferred.

[✓] Treat files without extension as ASCII file
[✓] Treat dotfiles as ASCII files
Dotfiles are filenames starting with a dot, e.g. .htaccess

In the home folder, you can see the following folders:

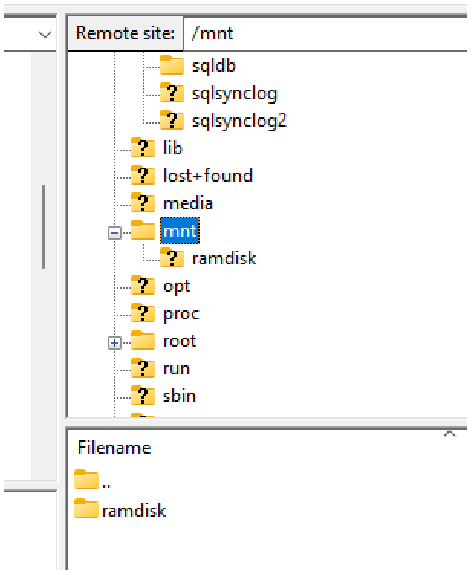


Name	Date modified	Type
gspdata	9/27/2019 10:41 PM	File folder
iecdata	11/11/2019 7:32 PM	File folder
openvpn	9/27/2021 10:15 AM	File folder
pbsLX	12/18/2024 10:57 AM	File folder
sqldb	12/18/2024 10:54 AM	File folder
sqlsynclog	9/27/2019 10:41 PM	File folder
sqlsynclog2	9/27/2019 10:41 PM	File folder

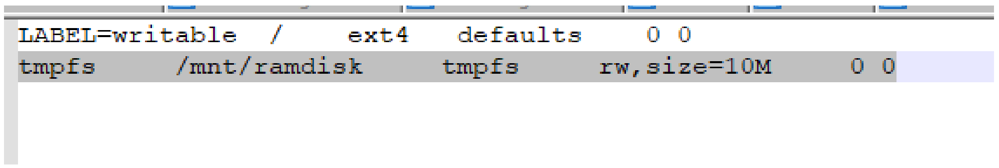
The runtime core is the pbsLX folder. Other folders are for storing data if you are using some drivers.

For an explanation of the pbsLX folder, refer to the pbsSoftLogic user guide.

pbssoftLogic runtime requires ramdisk to run. So please create a folder in /mnt named ramdisk.



Using filezilla, edit the /etc/fstab file on RPI-BBB and add the following line to it:



```
LABEL=writable  /     ext4    defaults     0  0
tmpfs       /mnt/ramdisk     tmpfs       rw,size=10M      0  0
```

This command will convert the /mnt/ramdisk folder as a real ramdisk in Linux.

In the unzipped pbsSoftLogic_ RPI64.zip folder, you can see the etc/init.d folder. Copy the /etc/init.d/xpsle file to the same path on the RPI-BBB.

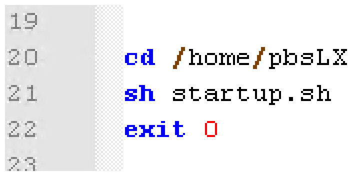
Connect to the controller by putty utility by root user and execute following command :

chmod +x /etc/init.d/xpsle

ln -s /etc/init.d/xpsle /etc/rc5.d/S97xpsle

This command executes pbsSLKLX, which is the pbsSoftLogic runtime kernel, at boot time.

You can run pbsSoftLogic kernel by adding these lines to /etc/rc.local file :



```

cd /home/pbsLX
sh startup.sh
exit 0

```

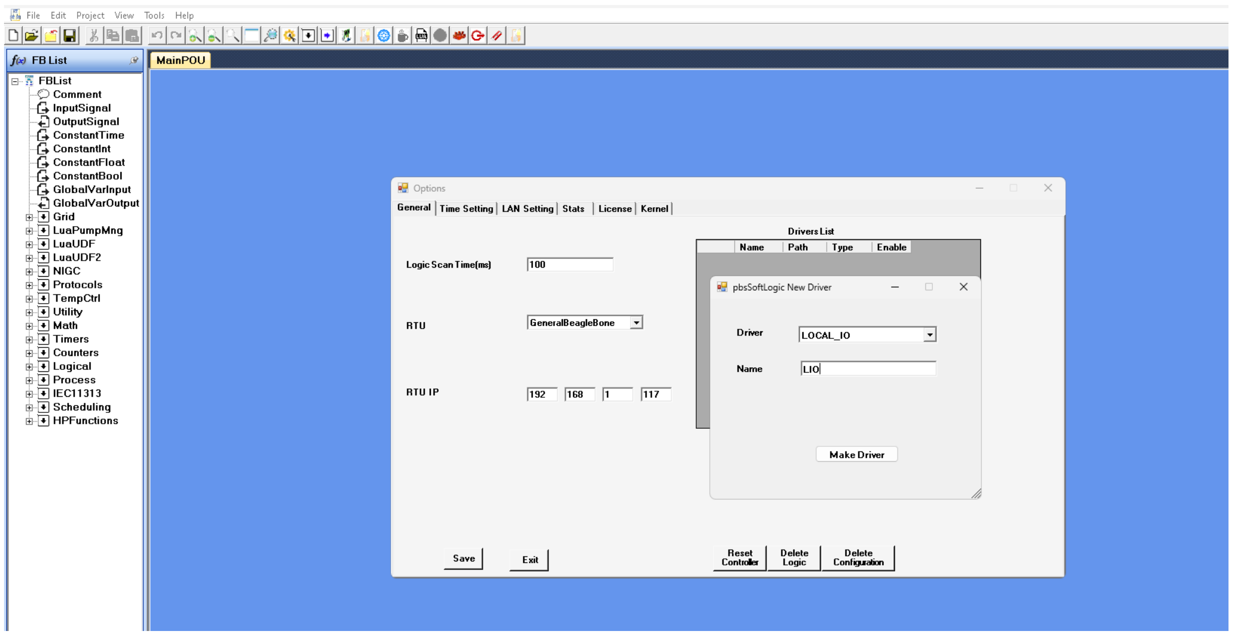
You can restart the RPI-BBB. The pbsSoftLogic runtime is now ready to use.

2 – Programming

To use pbsSoftLogic IDE, please refer to pbsSoftLogic User Guide. In this section, we will explain the local IO for RPI-BBB.

In pbsSoftLogic, Local IO is a driver that manages all the resources placed on the main CPU, such as LED, GPIO, Modem, Watch dog, etc. If you want to use RPI or BBB to test and learn pbsSoftLogic, you don't need to define Local IO for your project.

Run pbsSoftLogic and create a new project and open the project settings.



Logic Scan Time(ms): 100

RTU: GeneralBeagleBone

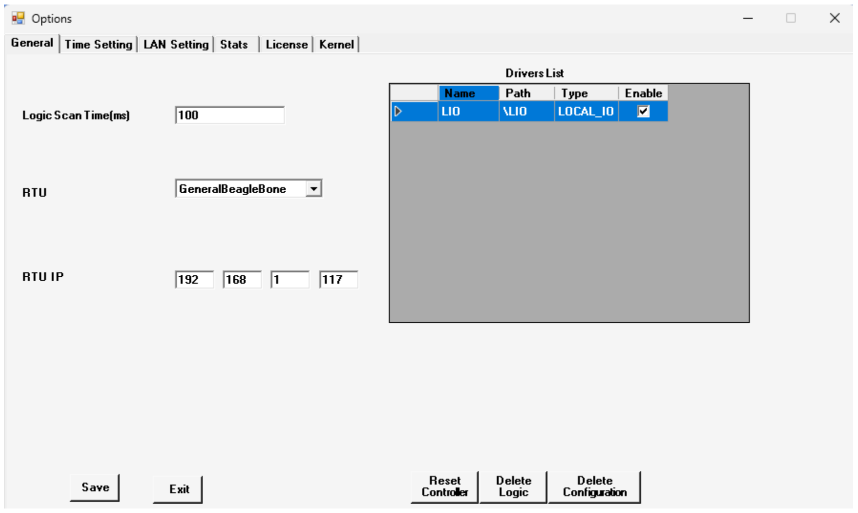
RTU IP: 192 168 1 117

Driver: LOCAL_IO

Name: LIO

Select GeneralBeagleBone as the RTU type and enter the controller IP address.

Right-click on the list of drivers and add a new Local_IO to the project and name it LIO.



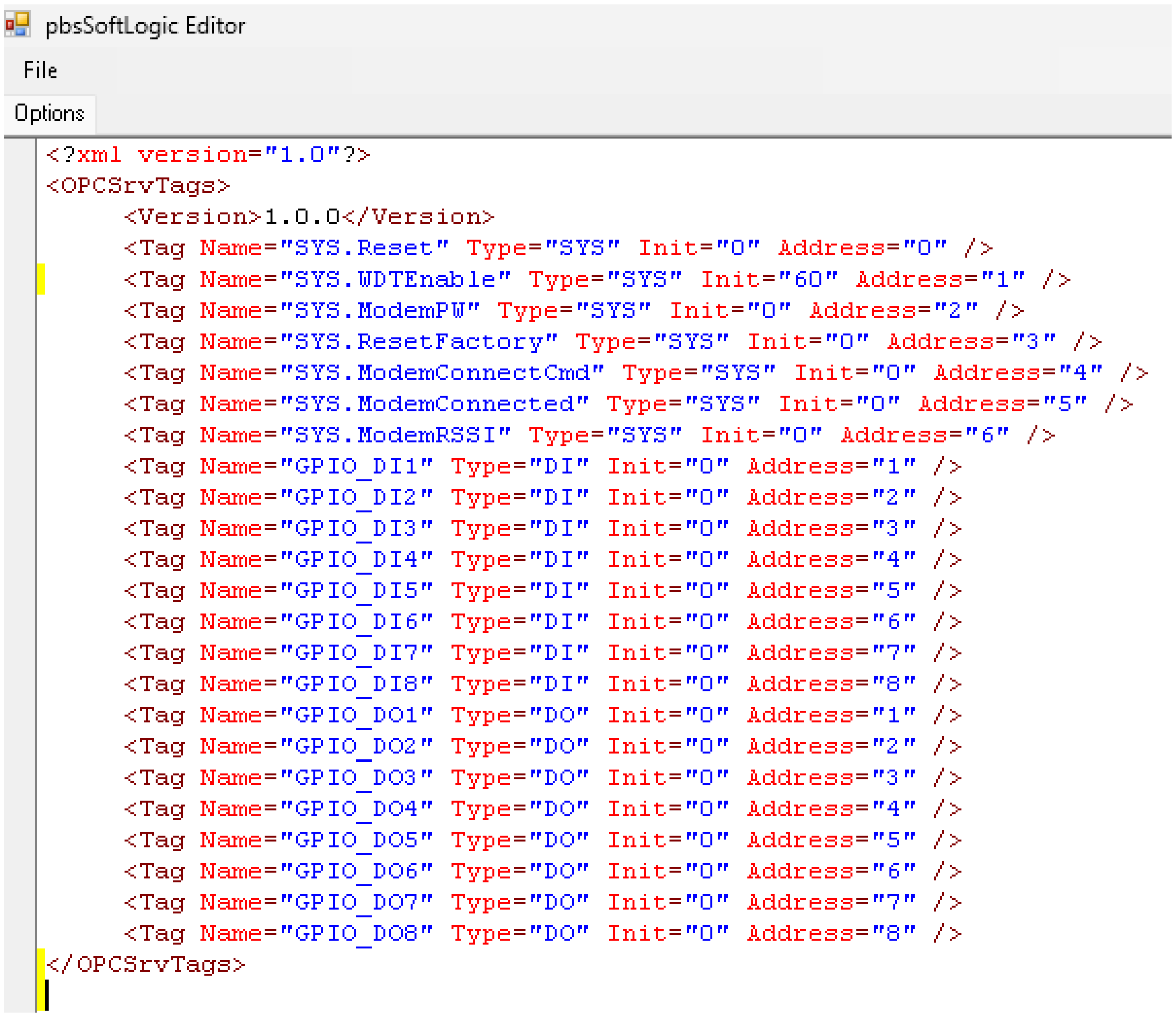
Logic Scan Time(ms): 100

RTU: GeneralBeagleBone

RTU IP: 192 168 1 117

Name	Path	Type	Enable
LIO	\LIO	LOCAL_IO	✓

Double-click on LIO Driver and you will see the following screen:



```
<?xml version="1.0"?>
<OPCSrvTags>
     <Version>1.0.0</Version>
     <Tag Name="SYS.Reset" Type="SYS" Init="0" Address="0" />
     <Tag Name="SYS.WDTEnable" Type="SYS" Init="60" Address="1" />
     <Tag Name="SYS.ModemPW" Type="SYS" Init="0" Address="2" />
     <Tag Name="SYS.ResetFactory" Type="SYS" Init="0" Address="3" />
     <Tag Name="SYS.ModemConnectCmd" Type="SYS" Init="0" Address="4" />
     <Tag Name="SYS.ModemConnected" Type="SYS" Init="0" Address="5" />
     <Tag Name="SYS.ModemRSSI" Type="SYS" Init="0" Address="6" />
     <Tag Name="GPIO_DI1" Type="DI" Init="0" Address="1" />
     <Tag Name="GPIO_DI2" Type="DI" Init="0" Address="2" />
     <Tag Name="GPIO_DI3" Type="DI" Init="0" Address="3" />
     <Tag Name="GPIO_DI4" Type="DI" Init="0" Address="4" />
     <Tag Name="GPIO_DI5" Type="DI" Init="0" Address="5" />
     <Tag Name="GPIO_DI6" Type="DI" Init="0" Address="6" />
     <Tag Name="GPIO_DI7" Type="DI" Init="0" Address="7" />
     <Tag Name="GPIO_DI8" Type="DI" Init="0" Address="8" />
     <Tag Name="GPIO_DO1" Type="DO" Init="0" Address="1" />
     <Tag Name="GPIO_DO2" Type="DO" Init="0" Address="2" />
     <Tag Name="GPIO_DO3" Type="DO" Init="0" Address="3" />
     <Tag Name="GPIO_DO4" Type="DO" Init="0" Address="4" />
     <Tag Name="GPIO_DO5" Type="DO" Init="0" Address="5" />
     <Tag Name="GPIO_DO6" Type="DO" Init="0" Address="6" />
     <Tag Name="GPIO_DO7" Type="DO" Init="0" Address="7" />
     <Tag Name="GPIO_DO8" Type="DO" Init="0" Address="8" />
</OPCSrvTags>
```

You can use the above tags in your logic:

SYS.Reset : when SYS.Reset set to 1 , for value more than watch dog time , will reset RPI-BBB.

SYS.WDTEnable : if set Init to 0 , WDT is disabled , when set to 60 for defining 60 sec WDT . If you need more WDT time, increase this value.

SYS.ModemPW: You can consider a 4G modem for your board and connect its power to GPIO and turn the modem on/off from the program. If set to 0, it means you don't have a modem on the board, otherwise the GPIO number should be set to SYS.ModemPW for the initial value.

SYS.ResetFactory : You can consider a GPIO as a factory reset button to delete the pbsSoftLogic logic and configuration files. Every time you press the Factory Reset button and hold it for more than 10 seconds, the LocalIO Driver deletes the configuration and logic files.

If the initial value is set to 0, it means you do not have the Factory Reset button; otherwise the GPIO number should be set as the initial value.

SYS.ModemConnectCMD : Changing from 0 to 1 turns the modem on and tries to connect to the mobile network. Changing from 1 to 0 turns the modem off.

SYS.ModemConnected: When it changes to 1, it indicates that the modem is properly connected to the network.

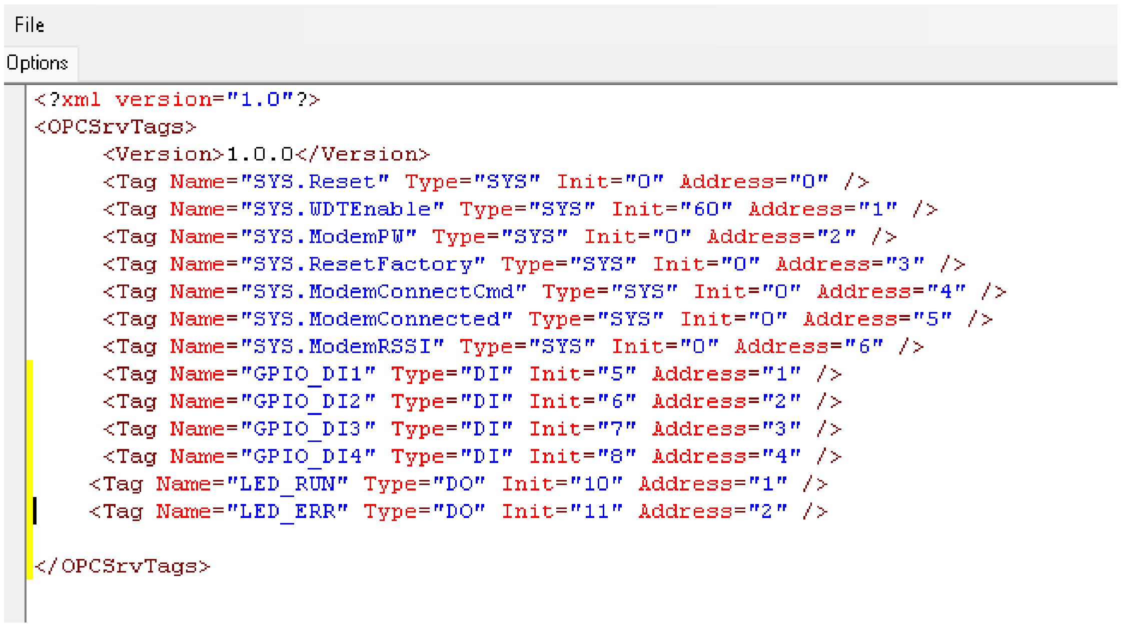
SYS.ModemRSSI : Shows Modem RSSI signal .

You can design up to 64 GPIOs as digital inputs or outputs on your board. To configure the GPIO software on your RPi or BBB, you should refer to their manual. We assume that you have configured the GPIOs and have one number for each GPIO.

If you don't have GPIO on your board that you want to be controlled by pbsSoftLogic, set all initial values for signals of type DI/DO to 0.

Otherwise you can add or remove tags from the GPIO list. You can change the name and set the initial value to the GPIO number. The tag address must be unique for each DI or DO type.

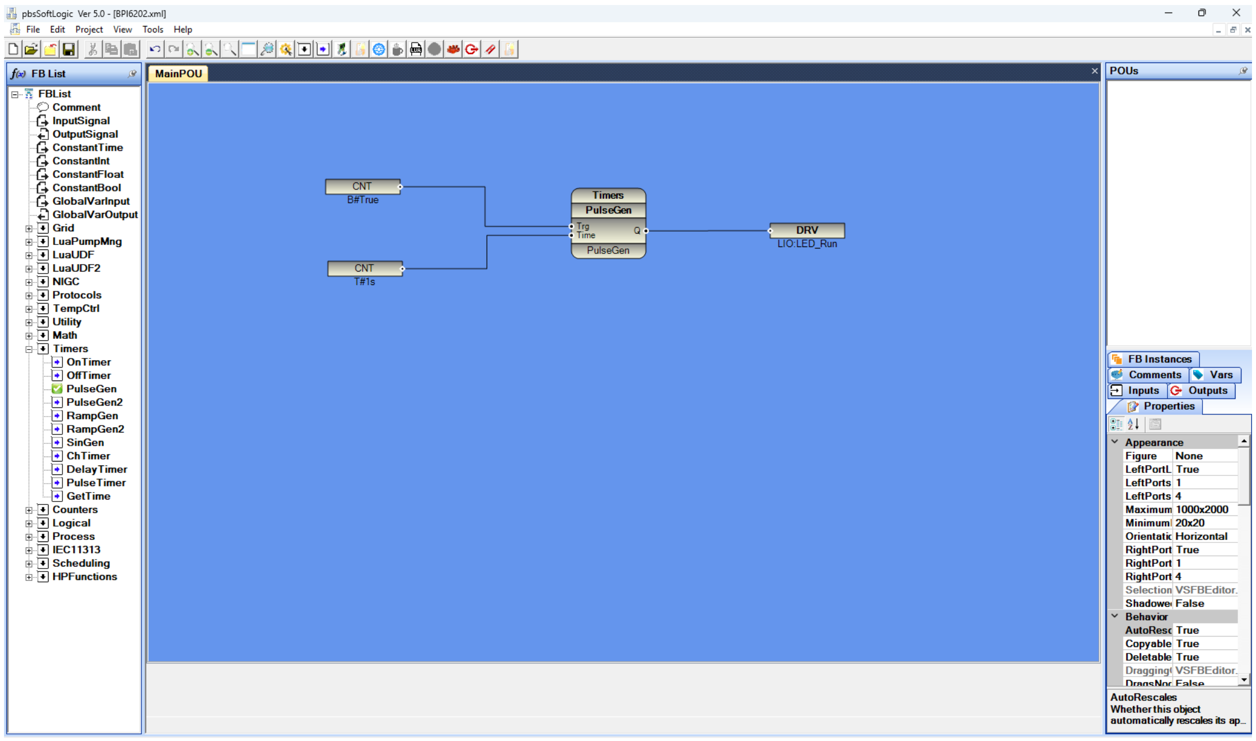
Suppose you have four GPIOs as DI with GPIO numbers 5,6,7,8 and two LEDs with GPIO numbers 10,11. The GPIO list will be as follows:



```
<?xml version="1.0"?>
<OPCSrvTags>
     <Version>1.0.0</Version>
     <Tag Name="SYS.Reset" Type="SYS" Init="0" Address="0" />
     <Tag Name="SYS.WDTEnable" Type="SYS" Init="60" Address="1" />
     <Tag Name="SYS.ModemPW" Type="SYS" Init="0" Address="2" />
     <Tag Name="SYS.ResetFactory" Type="SYS" Init="0" Address="3" />
     <Tag Name="SYS.ModemConnectCmd" Type="SYS" Init="0" Address="4" />
     <Tag Name="SYS.ModemConnected" Type="SYS" Init="0" Address="5" />
     <Tag Name="SYS.ModemRSSI" Type="SYS" Init="0" Address="6" />
     <Tag Name="GPIO_DI1" Type="DI" Init="5" Address="1" />
     <Tag Name="GPIO_DI2" Type="DI" Init="6" Address="2" />
     <Tag Name="GPIO_DI3" Type="DI" Init="7" Address="3" />
     <Tag Name="GPIO_DI4" Type="DI" Init="8" Address="4" />
    <Tag Name="LED_RUN" Type="DO" Init="10" Address="1" />
    <Tag Name="LED_ERR" Type="DO" Init="11" Address="2" />

</OPCSrvTags>
```

Now suppose you want to create a pulse generator and connect it to the Run LED. From the Timer Function Blocks, drag and drop a PulseGen function and place it on the MainPOU.

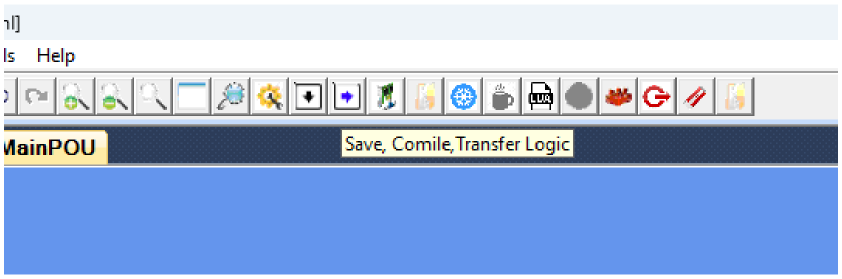


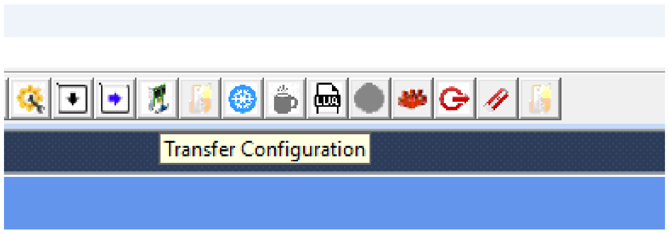
Drag a Constant Bool and connect to Trg input of PulseGen FB .

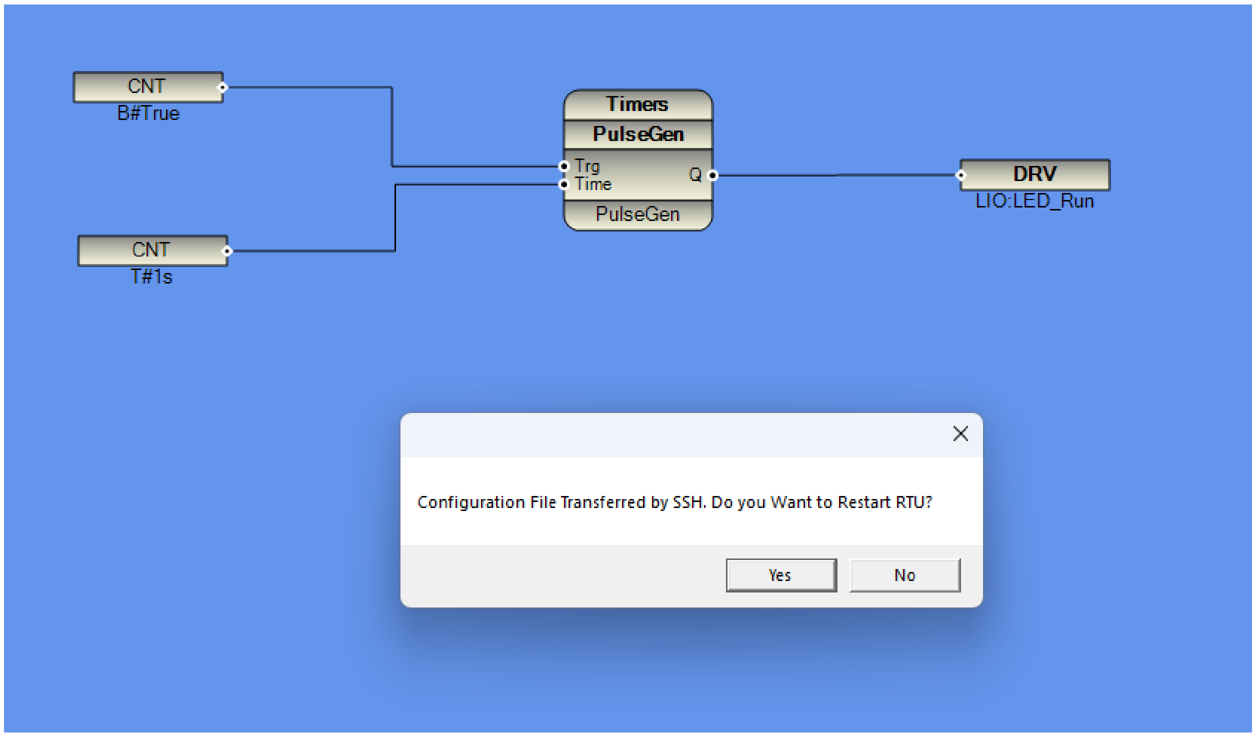
Drag a constant time and connect to Time input of Pulsegen FB.

Drag an OutputSignal and connect to Q Output of FB .

Save the project and transfer the Logic and Configuration to the RPI-BBB and restart it.



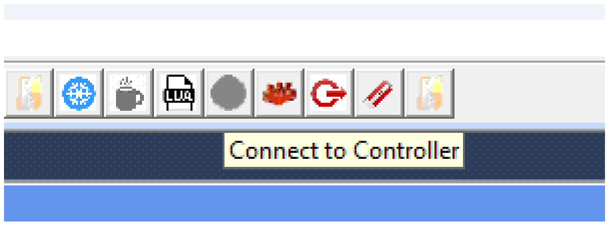


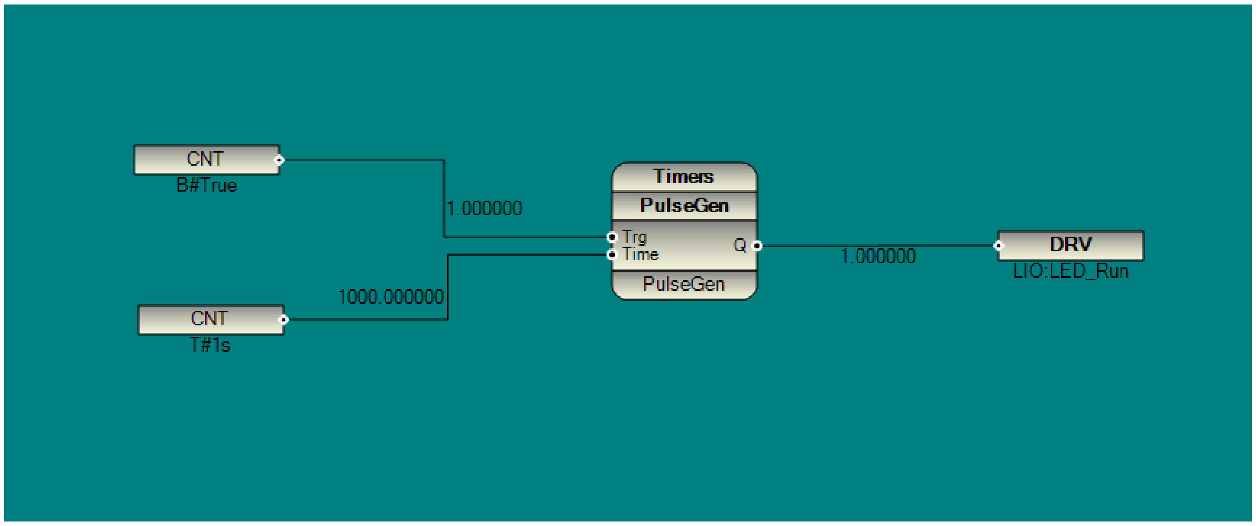


Configuration File Transferred by SSH. Do you Want to Restart RTU?

It takes 40 seconds for the RPI-BBB to boot and run the pbsSoftlogic runtime kernel. After that, you can connect to it and monitor your logic.

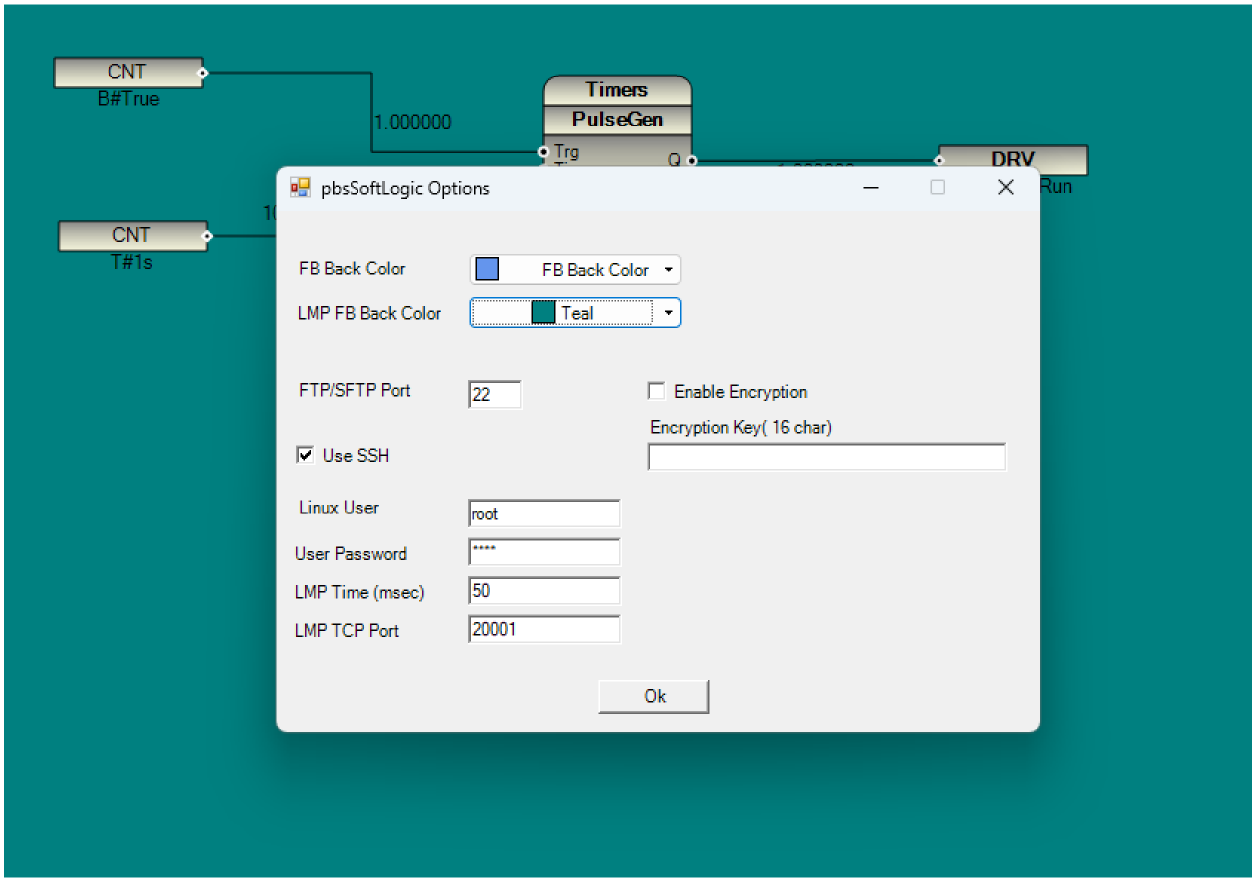
Click on connect to controller button and it will monitor your logic.





And the RPI_BB if you setup GPIOs properly , RUN LED flashes on and off every second.

In pbsSoftLogic, the default password for root is root. But if you change it, you need to change it in the tools option menu.



FB Back Color: FB Back Color
LMP FB Back Color: Teal
FTP/SFTP Port: 22
☐ Enable Encryption
Encryption Key(16 char):
☑ Use SSH
Linux User: root
User Password: ****
LMP Time (msec): 50
LMP TCP Port: 20001

After changing the password, close and re-run pbsSoftLogic IDE.

3- Protocol Configuration

To use protocols such as Modbus, DNP3, IEC104, MQTT, ... you need to add the appropriate driver to your project. For detailed explanations, please refer to the pbsSoftLogic user guide. In This section shows the serial port settings.

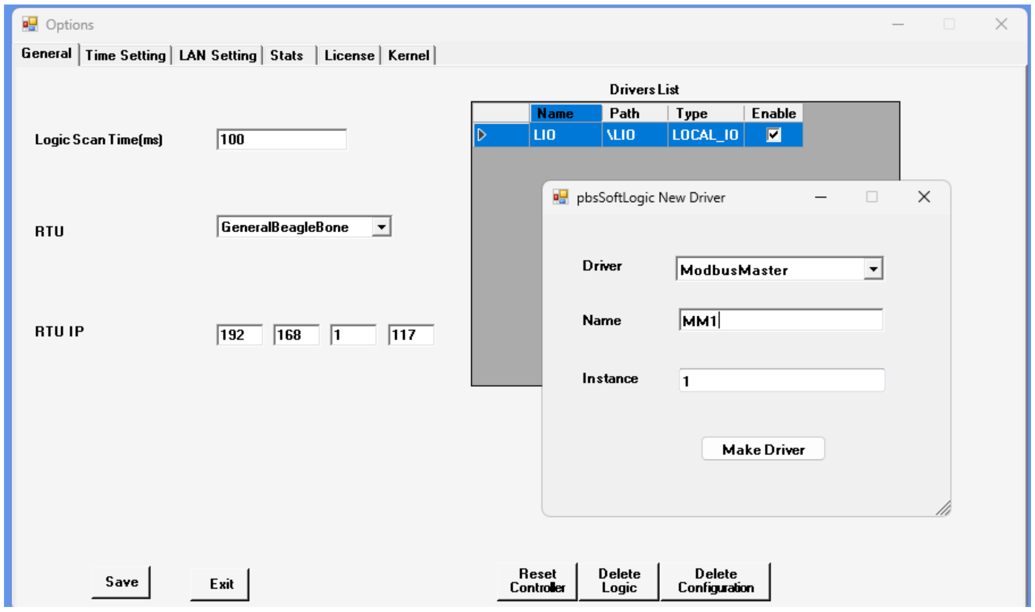
When you want to use serial ports in pbsSoftLogic, you need to use the serial port names.

From BeagleBone Black Manual :

The BeagleBone Black has four serial port UARTs, which are named /dev/ttyO1, /dev/ttyO2, /dev/ttyO4, and /dev/ttyO5. Each UART connects to a receiver (RX) and transmitter (TX) .

Suppose you want to define a Modbus Master for the UART1 of BBB for the project.

Right-click in the driver list and add Modbus Master Driver. For example, name it MM1.



Logic Scan Time(ms): 100

RTU: GeneralBeagleBone

RTU IP: 192 168 1 117

Drivers List

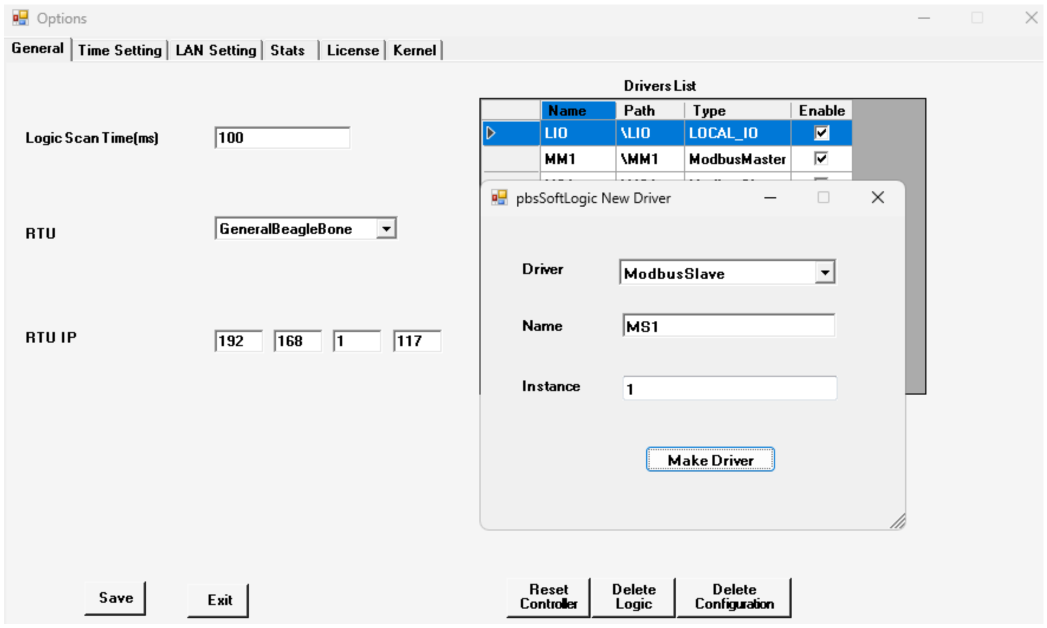
Name	Path	Type	Enable
LIO	\LIO	LOCAL_IO	✓

Driver: ModbusMaster

Name: MM1

Instance: 1

Add One Modbus Slave Driver and name it MS1.



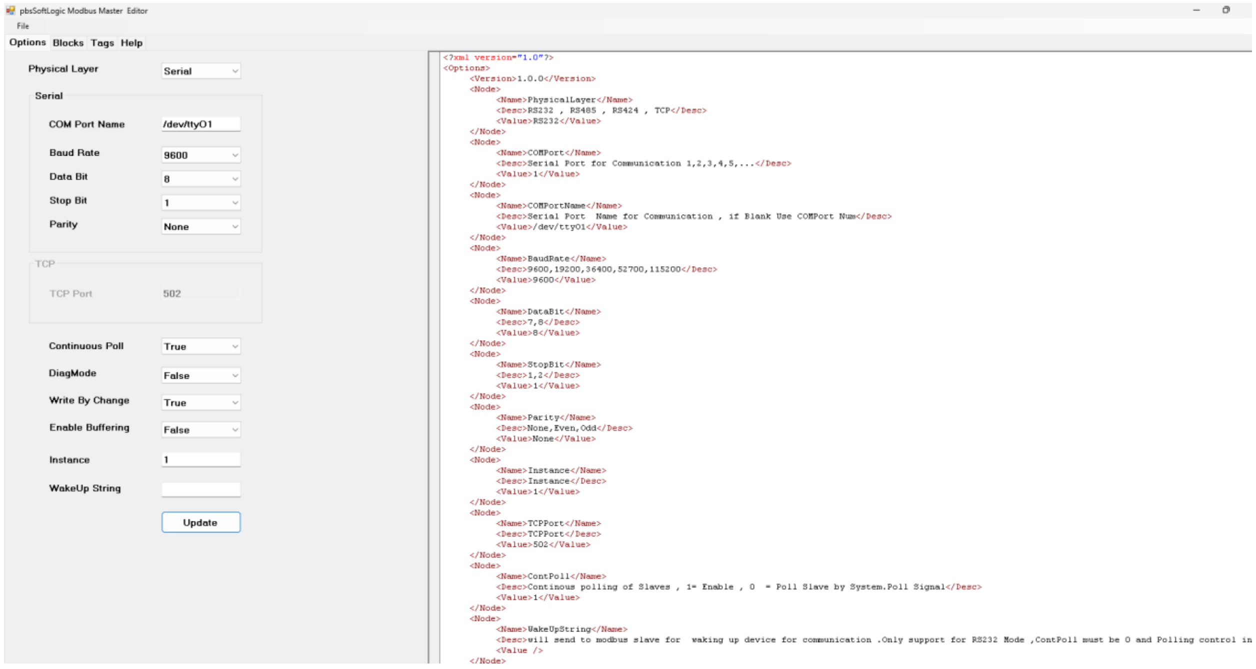
Logic Scan Time(ms): 100
RTU: GeneralBeagleBone
RTU IP: 192 168 1 117

Name	Path	Type	Enable
LIO	\LIO	LOCAL_IO	✓
MM1	\MM1	ModbusMaster	✓

Driver: ModbusSlave
Name: MS1
Instance: 1

Keep the instance number 1 for both drivers. The instance number must be unique for a driver type. If you want to add another Modbus Master to the project, you must change the instance to 2 for the second driver.

Double-click on MM1 Driver and you will see the following screen:

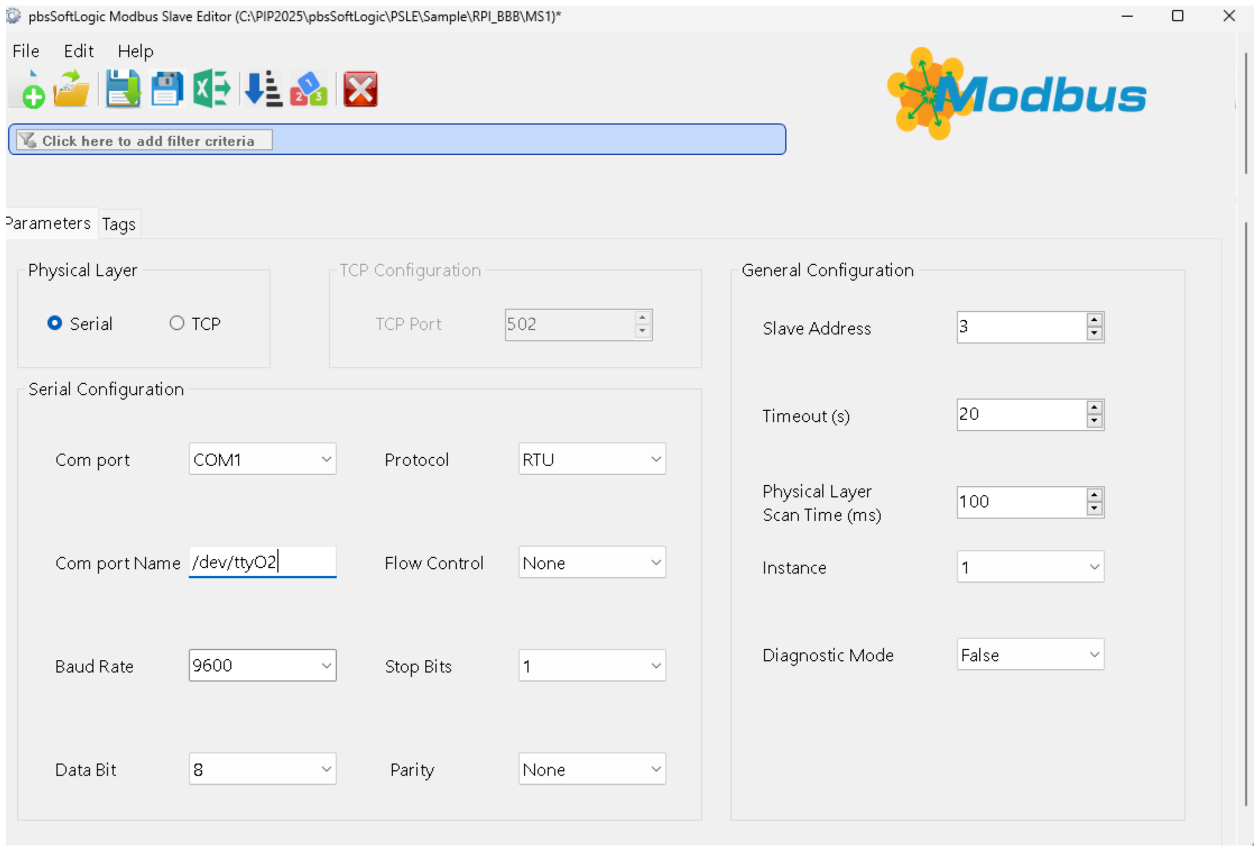


Physical Layer: Serial
COM Port Name: /dev/ttyO1
Baud Rate: 9600
Data Bit: 8
Stop Bit: 1
Parity: None
TCP Port: 502
Continuous Poll: True
DiagMode: False
Write By Change: True
Enable Buffering: False
Instance: 1
WakeUp String:

```xml
<?xml version="1.0"?>
<Options>
    <Version>1.0.0</Version>
    <Node>
        <Name>PhysicalLayer</Name>
        <Desc>RS232 , RS485 , RS424 , TCP</Desc>
        <Value>RS232</Value>
    </Node>
    <Node>
        <Name>COMPort</Name>
        <Desc>Serial Port for Communication 1,2,3,4,5,...</Desc>
        <Value>1</Value>
    </Node>
    <Node>
        <Name>COMPortName</Name>
        <Desc>Serial Port  Name for Communication , if Blank Use COMPort Num</Desc>
        <Value>/dev/ttyO1</Value>
    </Node>
    <Node>
        <Name>BaudRate</Name>
        <Desc>9600,19200,36400,52700,115200</Desc>
        <Value>9600</Value>
    </Node>
    <Node>
        <Name>DataBit</Name>
        <Desc>7,8</Desc>
        <Value>8</Value>
    </Node>
    <Node>
        <Name>StopBit</Name>
        <Desc>1,2</Desc>
        <Value>1</Value>
    </Node>
    <Node>
        <Name>Parity</Name>
        <Desc>None,Even,Odd</Desc>
        <Value>None</Value>
    </Node>
    <Node>
        <Name>Instance</Name>
        <Desc>Instance</Desc>
        <Value>1</Value>
    </Node>
    <Node>
        <Name>TCPPort</Name>
        <Desc>TCPPort</Desc>
        <Value>502</Value>
    </Node>
    <Node>
        <Name>ContPoll</Name>
        <Desc>Continous polling of Slaves , 1= Enable , 0  = Poll Slave by System.Poll Signal</Desc>
        <Value>1</Value>
    </Node>
    <Node>
        <Name>WakeUpString</Name>
        <Desc>will send to modbus slave for  waking up device for communication .Only support for RS232 Mode ,ContPoll must be 0 and Polling control in
        <Value />
    </Node>
```

You need to use the serial port name as the COM port name and set the other parameters.

Likewise, you should use the serial port name as the COM port name for modbus Slave Driver.


pbsSoftLogic Modbus Slave Editor (C:\PIP2025\pbsSoftLogic\PSLE\Sample\RPI_BBB\MS1)*
File
Edit
Help
Click here to add filter criteria
Modbus
Parameters
Tags

Physical Layer: [✓ Serial] TCP

TCP Configuration: TCP Port 502

Serial Configuration:

Com port	COM1	Protocol	RTU
Com port Name	/dev/ttyO2	Flow Control	None
Baud Rate	9600	Stop Bits	1
Data Bit	8	Parity	None

General Configuration:

Slave Address	3
Timeout (s)	20
Physical Layer Scan Time (ms)	100
Instance	1
Diagnostic Mode	False

For protocol configuration details, please refer to the pbsSoftLogic user guide for each protocol.

If you want to connect a modem to your board, you need to rename the modem serial port to /dev/ttyModem . You can do this by editing the /etc/udev/rules.d folder and adding a new rule to rename the modem serial port like the following example:

```
SUBSYSTEM=="tty" ACTION=="add", KERNEL=="ttyUSB[0-9]*", ATTRS{idVendor}=="05c6",
ATTRS{idProduct}=="9090", ENV{ID_USB_INTERFACE_NUM}=="03", SYMLINK+="ttyModem "
```