ProviewR

OPEN SOURCE PROCESS CONTROL

ProviewR on Raspberry Pi

2021 10 18

Copyright © 2005-2022 SSAB EMEA AB

Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

Table of Contents

Introduction....5
Development....5
Install the cross compiler....5
Install ProviewR....5
Create a project....5
Configure the project....6
Build the project....9
RPI installation....9
Distribute the project....9
Start ProviewR runtime....10

Introduction

Raspberry PI is a small single board computer developed to promote the the teaching of basic computer science in schools. This document describes how build a ProviewR project for raspbian on Raspberry PI. The reader expects to have some knowledge of how to create an build projects in ProviewR.

Development

There are two ways to develop a system for Raspberry PI. On way is to install the ProviewR development package on an RPI board with raspbian installed. The installation and creation of a projects follows the same procedure as for an ordinary debian system. It will work for small projects but is not ideal for larger projects.

The other way is to use an ordinary pc with ubuntu or debian, and to install the rpi cross compiler to generate code for the Raspberry PI. In this case you have to have access to the ProviewR runtime archives, cross compiled for RPI, and they are available in the pwrrpi package. This document will describe how to install the cross compiler, and configure a ProviewR project for Raspberry PI.

Install the cross compiler

On the development station, the cross compiler for RPI should be installed.

```
> sudo apt-get install gcc-arm-linux-gnueabihf
                       g++-arm-linux-gnueabihf
```

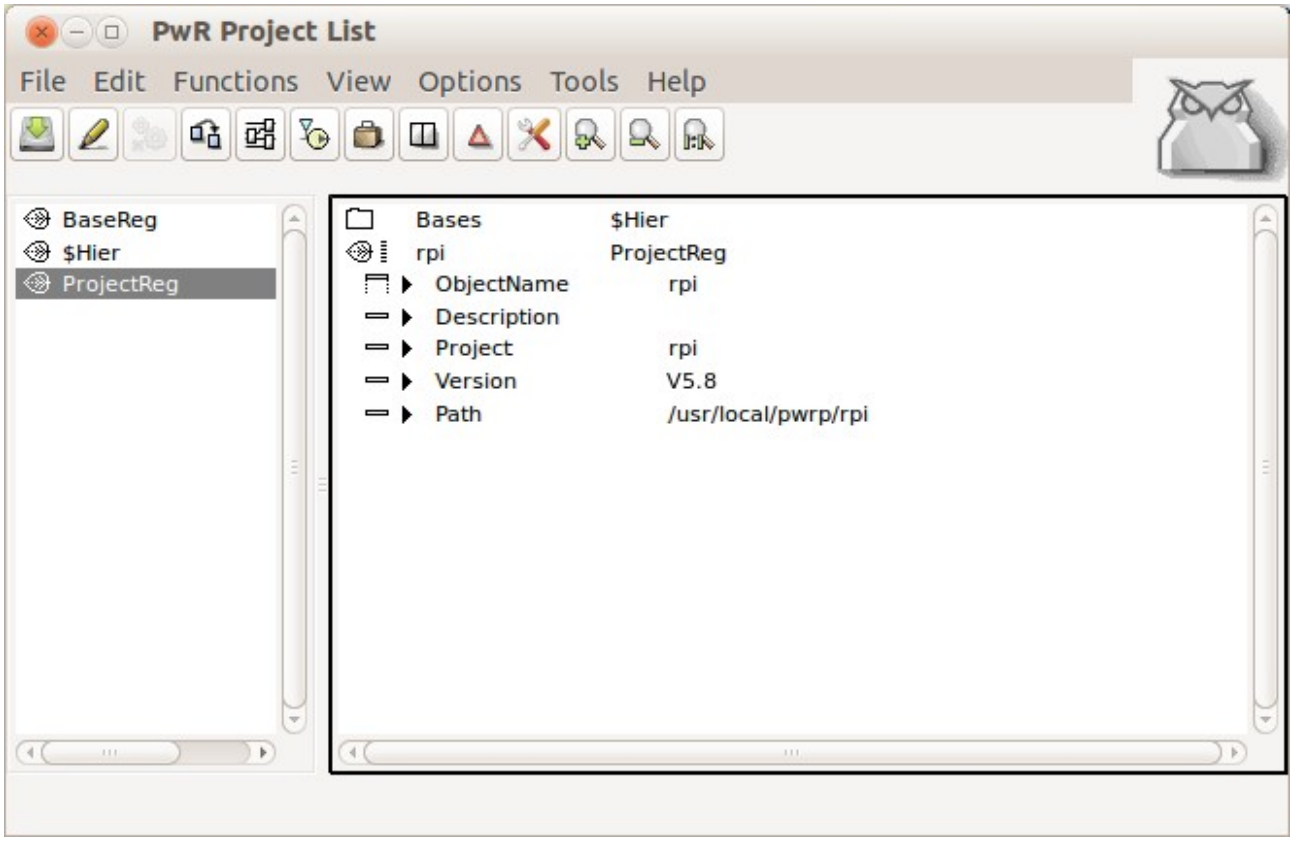
Install ProviewR

Download and install the ProviewR development package, pwr58, and the corresponding ProviewR RPI development package pwrrpi58.

Create a project

Log in as user pwrp and start the administrator to create an rpi project.

In this example the project name is set to rpi.



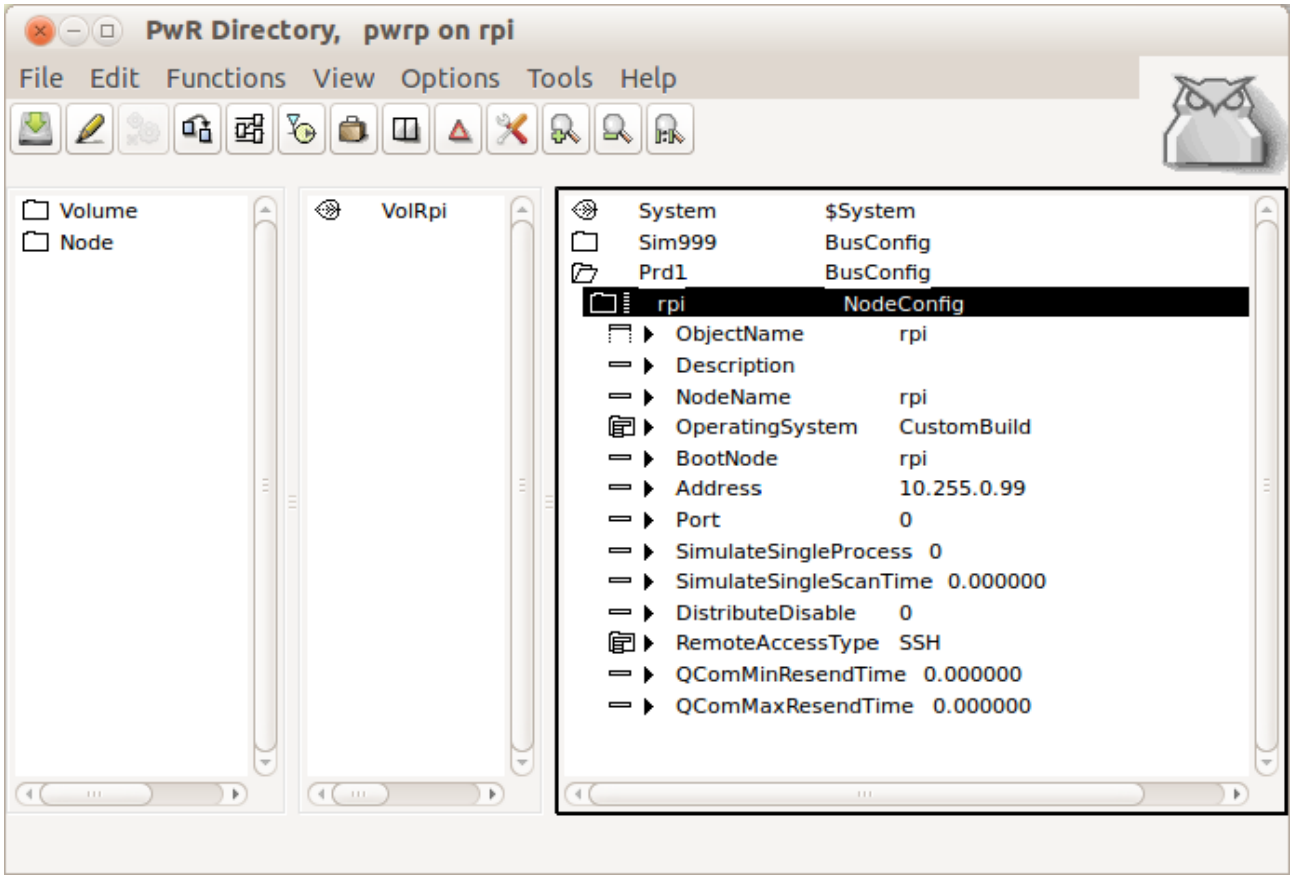
Open the new project by leaving edit mode and activate 'Open Project' in the popup menu for the ProjectReg object.

Configure the project

Run the configurator wizard and apply the default configuration.

Before leaving the directory volume, make following modifications.

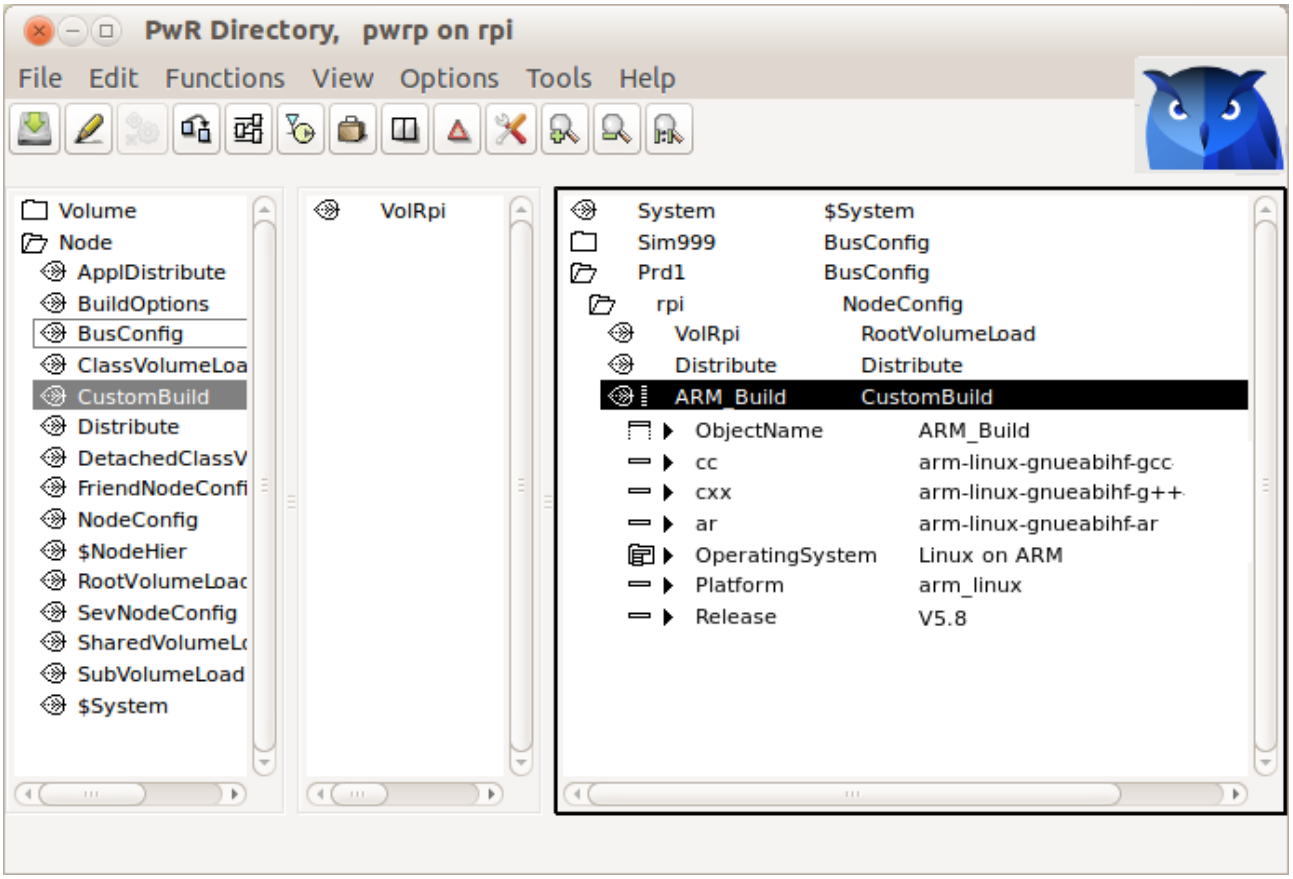
Change the OperatingSystem of the NodeConfig object for the Raspberry PI node to *CustomBuild*. Also insert the IP Address if this is not previously done, and the node name of the RPI in NodeName. Note that the default nodename in the raspbian distribution is 'raspberrypi', not rpi as in the figure below. As BootNode you can also set the IP address if the nodename isn't known on the development station.



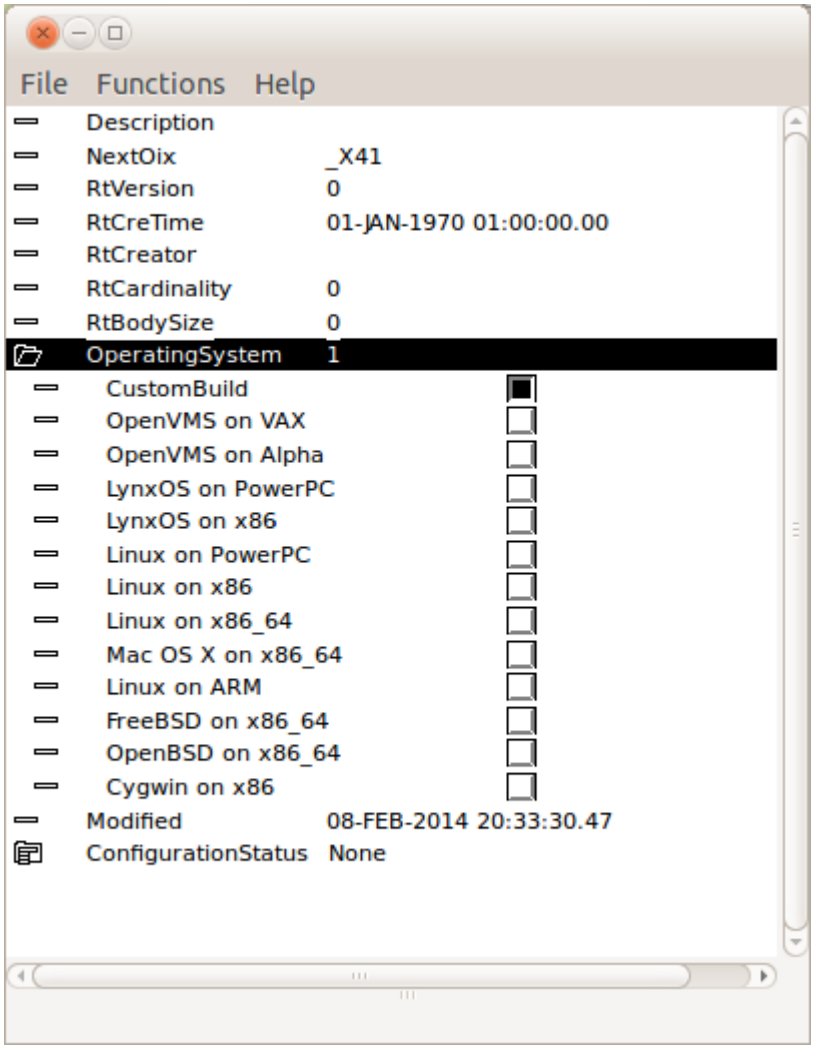
Add a CustumBuild object below the NodeConfig object.

Set

- OperatingSystem to *Linux on ARM*
- Platform to *arm_linux*
- Relase to the current ProviewR release.
- cc to *arm-linux-gnueabihf-gcc*
- cxx to *arm-linux-gnueabihf-g++*
- ar to *arm-linux-gnueabihf-ar*



Open the root volume, in the example VolRpi, execute the basic configuration with the wizard, and then open volume attributes from File/Volume Attributes in the menu. Set OperatingSystem to *CustomBuild.*



Build the project

Build the RPI node by activating the build button in the configurator tool bar, and select the rpi node in the list.

RPI installation

Install Raspbian Pi OS on the RPI board, and then download and install the pwrrt package. See ProviewR Installation Guide.

Distribute the project

Distribute the RPI project from the development station to the RPI board from Functions/Distribute in the configurator. Select the rpi node in the list. Note that the distributor will prompt for a password in the terminal window for the configurator. Enter the password for user pwrp, pwrp, two times.

Start ProviewR runtime

Login as user pwrp with password pwrp, on the RPI. Start ProviewR with

```
> pwr start
```

Start rt_xtt and checkout the runtime environment.

```
> rt_xtt
```