QNX 6.6.0
Setup Manual
For Raspberry Pi2 Board

User's Manual: Software

BCM 2836

www.swhwc.com

Rev.1.3 December 15, 2016

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1. Overview

1.1 Features

In the case of V.1.2.0 BSP, following drivers/libraries/utilities are supported:

1) Startup
2) ARM timer
3) Mailbox
4) Serial driver
5) SD driver
6) Display driver
7) USB driver
8) Network driver
9) I2C driver
10) GPIO driver
11) CAN driver
12) i2c_test, gpio_test, spi_test utilities

1.2 Scope

This document scope applies to the evaluation of reproducing QNX6.6.0 image and testing QNX BSP drivers for Raspberry Pi2 platform.

1.3 Target System

1) Target platform: Raspberry Pi2 platform.
2) Target software: QNX SDP 6.6.0

1.4 List of Abbreviations and Acronym

Abbreviation	Full Form
BSP	Board Support Package
Mailbox	BCM2836 Mail box
SDP	Software Development Platform - A software is used to build BSP
Momentics IDE	QNX Momentics Integrated Development Environment

1.5 Environmental Requirement

Table 1.1 Environment Requirement

Equipment	Explanation
Windows Host PC	Windows 7 or higher
Terminal software	TeraTerm (version 4.75 or newer)

2. Building QNX6.6.0 OS image

2.1 Prerequisites

1) QNX6.6.0 SDP installed on Windows Host PC.

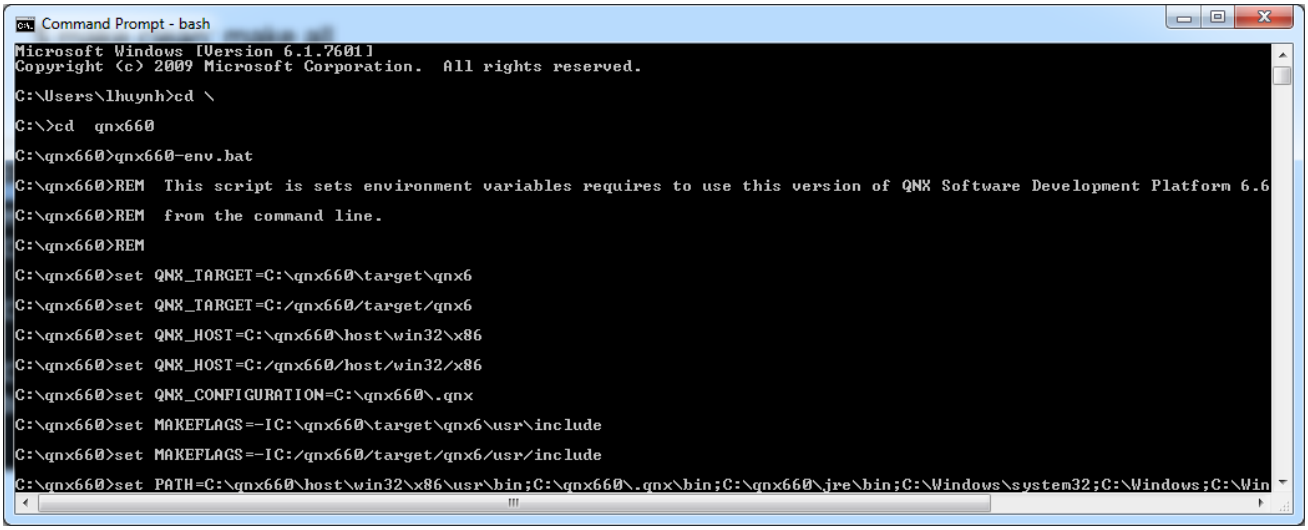
2) QNX6.6.0 BSP for Raspberry Pi2 board project folder
Download bsp-ntotrunk-broadcom-bcm2836-trunk.zip and extract it into a folder on Windows Host PC (ex: C:\bsp-ntotrunk-broadcom-bcm2836-trunk)

3) Valid active key (bcm2836.build)
Replace (overwrite) valid active key (bcm2836.build) onto bsp-ntotrunk-broadcom-bcm2836-trunk\prebuilt\armle-v7\boot\build\bcm2836.build
A valid active key is not required to build QNX6.6 image from bsp-ntotrunk-broadcom-bcm2836-trunk.zip, but you can't successfully boot QNX6.6 with an invalid active key.
To obtain a valid active please get your Raspberry Pi2 board's serial number and send this serial number to SHC.

2.2 Build QNX 6.6.0 BSP

To generate the QNX image for Raspberry Pi2 board, enter following commands from the command prompt console:

```
$ cd QNX 6.6.0 install
$ qnx660-env.bat
$ cd <QNX_BSP_Root_Directory>
$ bash
$ make clean; make all
```



QNX 6.6.0 image is created at: bsp-ntotrunk-broadcom-bcm2836-trunk/images/ifs-bcm2836.bin

3. Boot up QNX 6.6.0

3.1 Prerequisites

- A SD Card which the first partition is formatted as FAT32.
- ifs-bcm2836.bin: a QNX 6.6 image for Raspberry Pi2 board, which is built by a valid active key as guide in 2.2 section Build QNX 6.6.0 BSP.
- SD_boot_binaries.zip which is patched in the release package.
- Tera Term software installed on Windows PC.

3.2 Boot up steps

Please follow the steps:

3.2.1 Step 1: Copy the necessary files to SD card.

- Put file ifs-bcm2836.bin which located at bsp-ntotrunk-broadcom-bcm2836-trunk/images/ifs-bcm2836.bin to the root folder of the SD card.
- Extract files in SD_boot_binaries.zip to the root folder of the SD card. After this step, in the root folder of SD card should have these following files:
    - config.txt
    - bootcode.bin
    - start.elf
    - start_cd.elf
    - start_db.elf
    - start_x.elf
    - fixup.dat
    - fixup_cd.dat
    - fixup_db.dat
    - fixup_x.dat
    - ifs-bcm2836.bin

3.2.2 Step 2: Insert SD to Raspberry Pi2 board

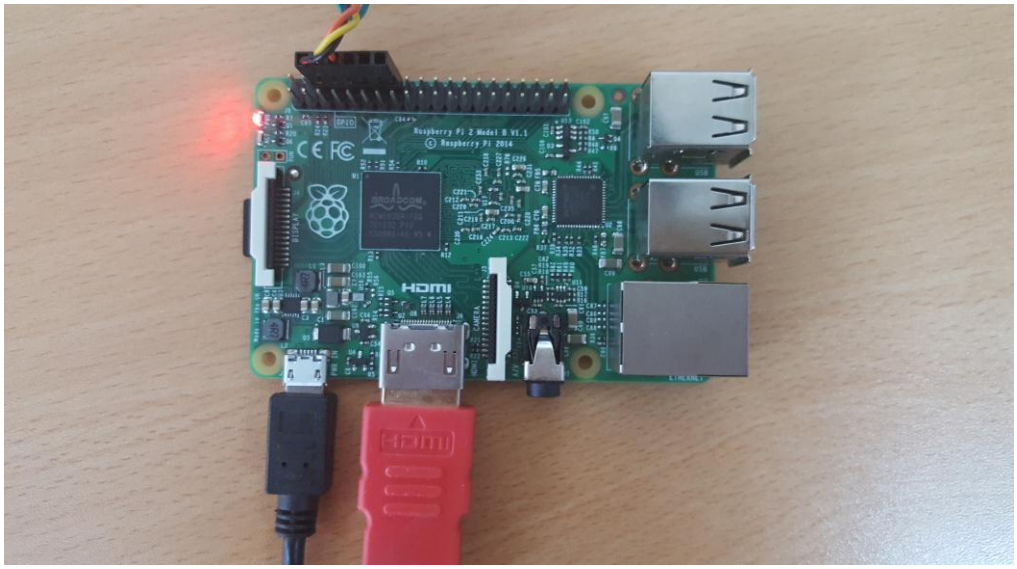
Insert SD card which has the necessary files into SD slot of Raspberry P2 board.

3.2.3 Step 3: Connect serial port of Raspberry Pi2 to Window PC

Please use TTL-232R-3V3 (http://www.ftdichip.com/Products/Cables/USBTTLSerial.htm) to connect Raspberry Pi2 board to Window PC as following:

GND:	TTL-232R-3V3 pin1 (Black)	<-------->	RPi2 J8 pin 6
Rx :	TTL-232R-3V3 pin5 (Yellow)	<-------->	RPi2 J8 pin 8
Tx :	TTL-232R-3V3 pin4 (Orange)	<-------->	RPi2 J8 pin 10

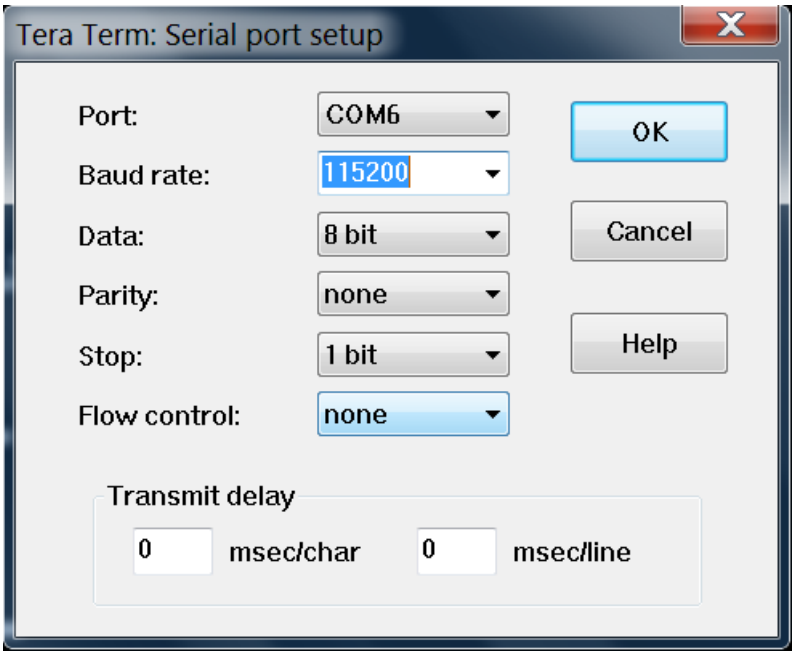
Please note: you must use the TTL-232R-3V3 version. The version TTL-232R-5V could damage your Raspberry Pi GPIO because of 5V logic.



3.2.4 Step 4: Connect HDMI LCD to Raspberry Pi2 board

3.2.5 Step 5: Setup serial port for Window PC

From Window PC open Tera Term program and set up as following:



3.2.6 Step 6: Power up Raspberry Pi2 board

Raspberry Pi2 board is powered up by plug the board to an USB power source. You could power it by your PC USB port if it could supply more than 800mA.

3.2.7 Step 7: Image QNX 6.6.0 will be loaded

Image will be loaded and the [Tera Term] terminal will output as bellows:

```
CPU0: L1 Icache: 1024x32
CPU0: L1 Dcache: 512x64 WB
CPU0: L2 Dcache: 8192x64 WB
CPU0: VFP-d32 FPSID=41023075
```

```
CPU0: NEON MVFR0=10110222 MVFR1=11111111
CPU0: 410fc075: Cortex A7 rev 5 600MHz
Loading IFS...decompressing...done
CPU1: L1 Icache: 1024x32
CPU1: L1 Dcache: 512x64 WB
CPU1: L2 Dcache: 8192x64 WB
CPU1: VFP-d32 FPSID=41023075
CPU1: NEON MVFR0=10110222 MVFR1=11111111
CPU1: 410fc075: Cortex A7 rev 5 600MHz
CPU2: L1 Icache: 1024x32
CPU2: L1 Dcache: 512x64 WB
CPU2: L2 Dcache: 8192x64 WB
CPU2: VFP-d32 FPSID=41023075
CPU2: NEON MVFR0=10110222 MVFR1=11111111
CPU2: 410fc075: Cortex A7 rev 5 600MHz
CPU3: L1 Icache: 1024x32
CPU3: L1 Dcache: 512x64 WB
CPU3: L2 Dcache: 8192x64 WB
CPU3: VFP-d32 FPSID=41023075
CPU3: NEON MVFR0=10110222 MVFR1=11111111
CPU3: 410fc075: Cortex A7 rev 5 600MHz
alloc_syspage_memory: syspage size:00000dc8 _syspage_ptr:00def000
callout_io_map: mapping paddr:3f00b200 returns:fc419200
callout_io_map: mapping paddr:40000000 returns:fc41a000
callout_io_map: mapping paddr:3f00b200 returns:fc41b200
callout_io_map: mapping paddr:40000000 returns:fc41c000
callout_io_map: mapping paddr:3f00b200 returns:fc41d200
callout_io_map: mapping paddr:40000000 returns:fc41e000
callout_io_map: mapping paddr:3f00b200 returns:fc41f200
callout_io_map: mapping paddr:40000000 returns:fc420000
callout_io_map: mapping paddr:00000000 returns:fc421000
callout_io_map: mapping paddr:3f100000 returns:fc422000
callout_io_map: mapping paddr:3f00b400 returns:fc423400
callout_io_map: mapping paddr:3f00b400 returns:fc425400
callout_io_map: mapping paddr:3f00b400 returns:fc427400
callout_io_map: mapping paddr:3f201000 returns:fc429000
callout_io_map: mapping paddr:3f201000 returns:fc42a000
callout_io_map: mapping paddr:3f201000 returns:fc42b000
cpu_startnext: cpu1 -> fc410b50
cpu_startnext: cpu2 -> fc410b50
cpu_startnext: cpu3 -> fc410b50

System page at phys:00def000 user:fc410000 kern:fc410000
Starting next program at vfe05ae3c
cpu_startnext: cpu0 -> fe05ae3c
VFPv3: fpsid=41023075
coproc_attach(10): attach fe076c34 (fe078b68)
coproc_attach(11): attach fe076c34 (fe078b68)
Welcome to Neutrino 6.6.0 on the RaspBerry P2 Board(ARMv7 Cortex-
A7 core)
Starting pl011 driver...
Starting slogger and pipe servers...
Starting USB driver...
Starting SPI driver for SPI...
Starting GPIO driver...
Starting I2C driver
Starting graphics driver...
Starting SD memory card driver...
Path=0 - rpi
 target=0 lun=0     Direct-Access(0) - SDMMC: 00000 Rev: 0.1
Starting network driver...
```

```
Starting DHCP...
RPi2#
```

3.2.8 Step 8: Boot up finish.

Boot up is finished with prompt “RPi2#” and user could enter command from this prompt.

4. Test driver

4.1 Serial driver

Make sure can type some commands.

4.2 USB driver

Notes: Mouse isn't supported yet.

1) Plug in USB device to USB port (Upper or Lower) of Raspberry board.
2) Make sure io-usb and devb-umas is running as below.

```
# io-usb -d rpi ioport=0x3f980000,irq=73

# devb-umass cam pnp
```

3) From Terminal console, run below commands

```
# mount -tdos  /dev/hdXtY /usb_flash     (Replace hdXtY with the actual USB
                                         device node created in /dev/)

# cp -V /usb_flash/file_name /tmp        (Copy data from USB)

# cp -V /tmp/filename /usb_flash/        (Copy data to USB)
```

4.3 Network driver

4.3.1 Test Overview

Connect the Raspberry Pi2 board and Windows PC via a hub by using LAN cables. Make sure that data is sent and received between PC and the Raspberry Pi2 board through Ethernet.

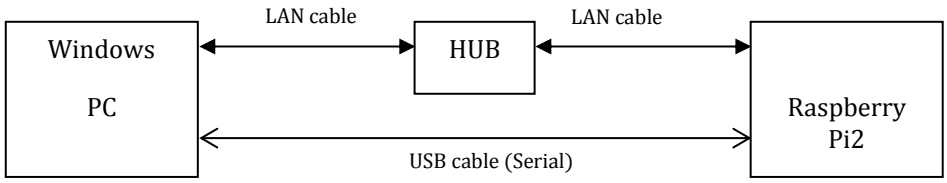


Figure 4-1 Connect the Raspberry Pi2 board and Windows PC

4.3.2 Equipment and Software Used

(1) LAN cables (straight through)
(2) USB cable
(3) Windows PC
(4) HUB 100 Mbps
(5) Test programs and data files:

No.	Name	Description	Remarks
1	ttermpro.exe	Terminal software	Windows PC
2	TELNET.EXE	TELNET client program	Windows PC
3	ping.exe	Ping program	Windows PC and Raspberry Pi2
4	qde.exe	QNX Momentics IDE 5.0	Windows PC

4.3.3 Test Items

No.	Major Test Item	Test Outline
1	Verify link connectivity	Execute ping command on the target and Windows PC to verify link connectivity.
2	Connect to target using TELNET	Start the TELNET on the Window PC and make sure that command transmission and data reception are possible.
3	File transmission	Start QNX Momentics IDE 5.0 on the Window PC and make sure that a file can be transferred between Window PC and Raspberry Pi2 board

4.3.4 Test Procedures

- Start QNX Momentics IDE on the Windows PC. Add Raspberry Pi2 to the target list. Use the File System Navigator to copy data to the Windows PC.
- Start Ethernet driver on Raspberry Pi2 target: Make sure the followings were started already at time of starting QNX image by pidin a as below:

 # pidin a

….

28690 io-pkt-v4 -dsmsc9514 mac=2e990a002e28 -ptcpip

32787 dhcp.client -l10 -d -ien0

36885 devc-pty

36886 inetd

40980 qconn port=8000

- To check ip of the board use “ifconfig” command.

4.3.5 Test Link connectivity using ping command

(1) On Raspberry Pi2 board

+ ping to host with packet size 64 bytes (64=56+8)

 # ping –h <host IP address> -s 56 -c 5

Expected result: successful with 5 packets transmitted, 5 received, 0% packet loss

+ Ex: # ping –h 192.168.1.103 –s 56 –c 5

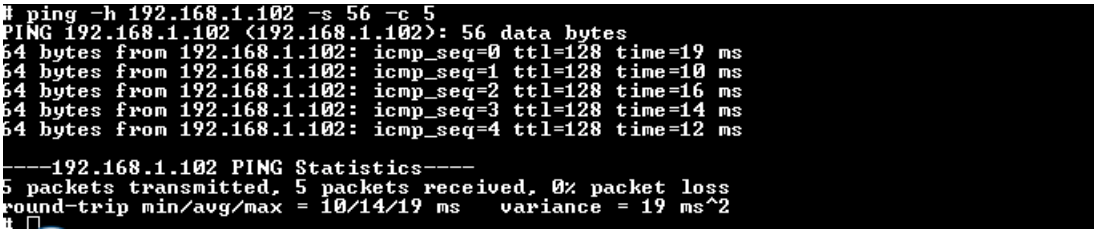


Figure 4-2 Test ping command on Raspberry Pi2 board

(2) On Windows PC Host

+ ping to board with packet size 64 bytes

#ping <board IP address> -l 64 -n 5

Expected result: successful with 5 packets transmitted, 5 received, 0% packet loss

+ Ex: # ping 192.168.1.102 –l 64 –n 5

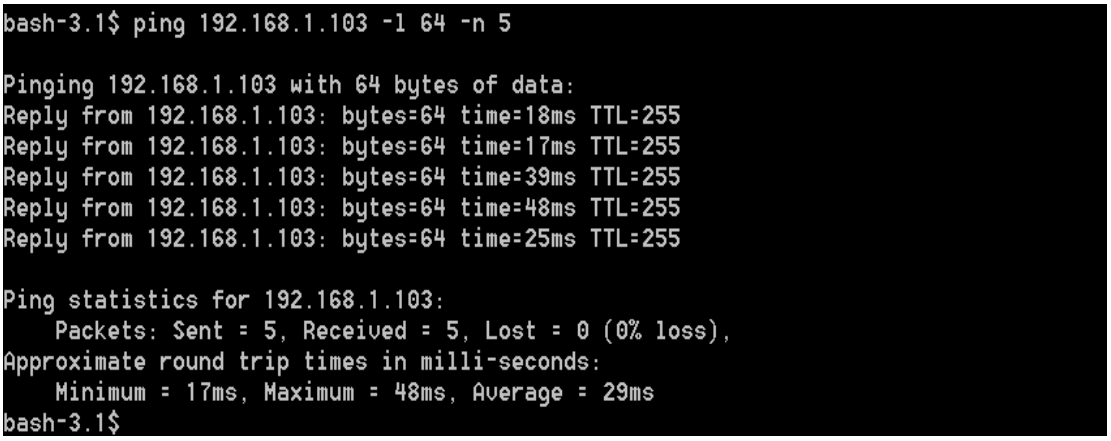


Figure 4-3 Test ping command on PC

4.3.6 Testing Command Control by a TELNET Client

(1) From command line on Window PC:

- telnet 192.168.1.103 (IP of Raspberry Pi2 Board)

(2) Login message is displayed on the Windows PC.

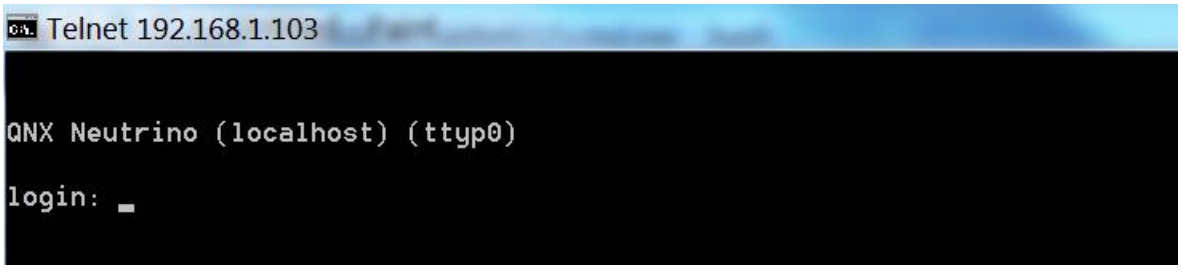


Figure 4-4 Telnet login console

(3) Enter root. You can login to Raspberry Pi2 board

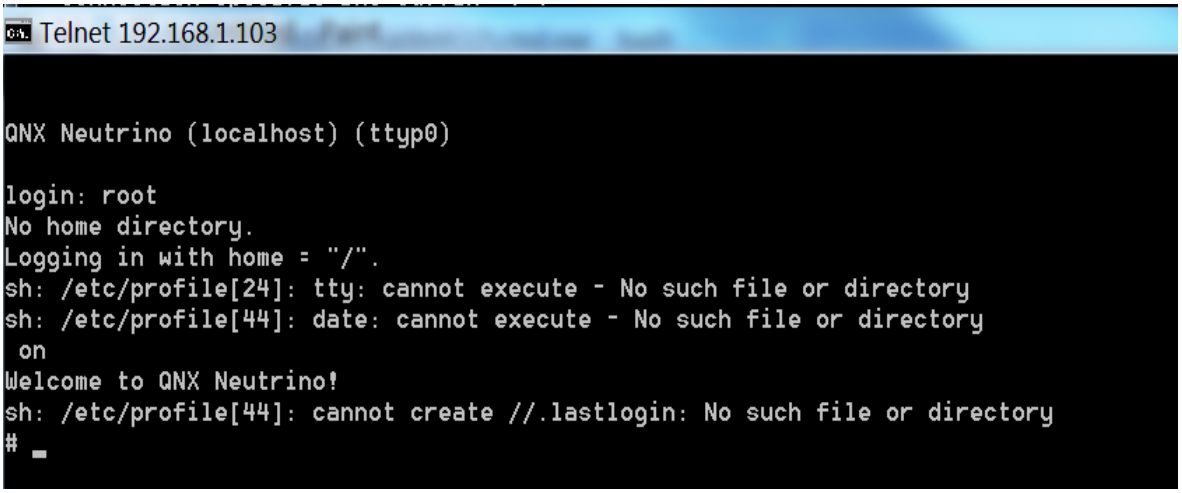


Figure 4-5 Raspberry Pi2 QNX 6.6 console

4.3.7 File transmission test between Window PC and Raspberry P2 board

(1) Start QNX Momentics IDE 5.0 on the Window PC.

(2) Add Raspberry Pi2 platform to the target list.

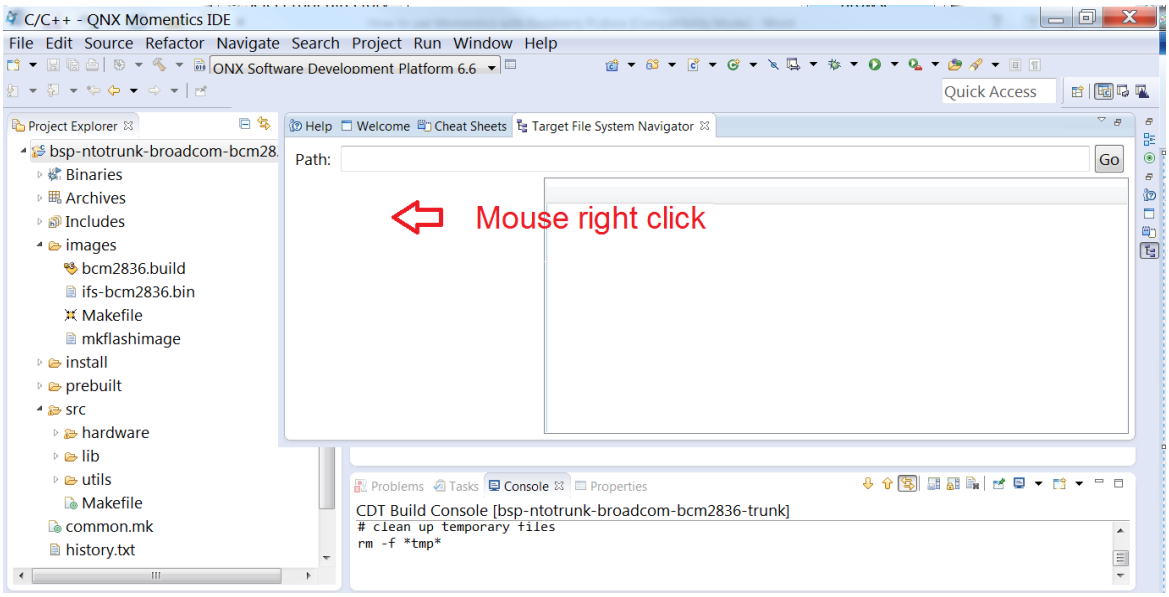


Figure 4.6 Open New QNX target

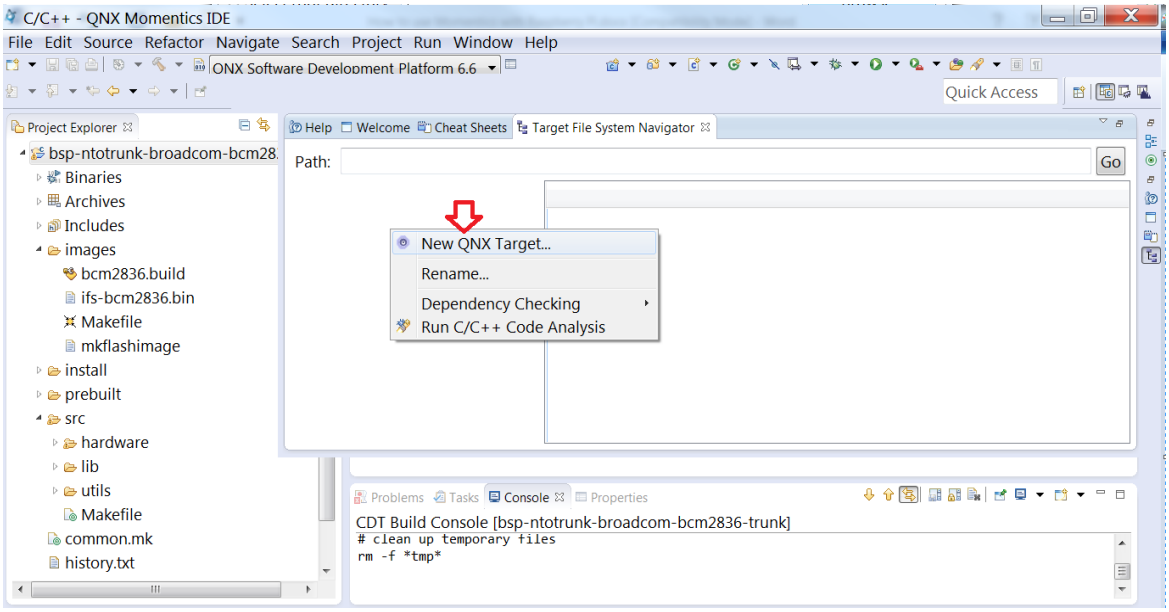


Figure 4.7 Select New QNX target

Enter IP of Raspberry Pi2 platform:

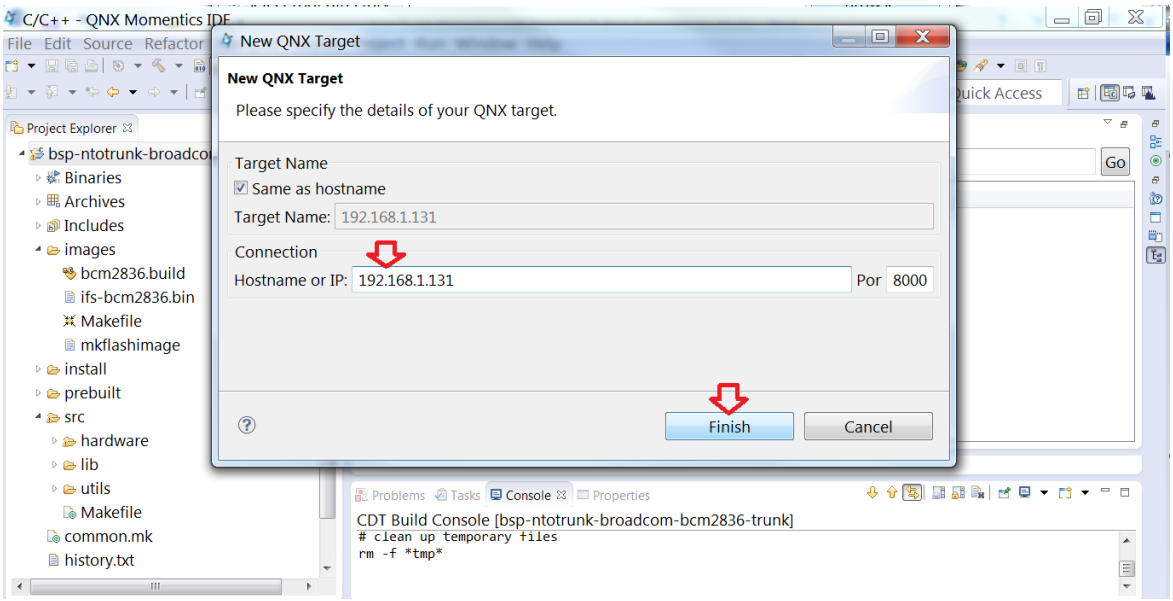


Figure 4.8 Add Raspberry Pi2 QNX 6.6 target

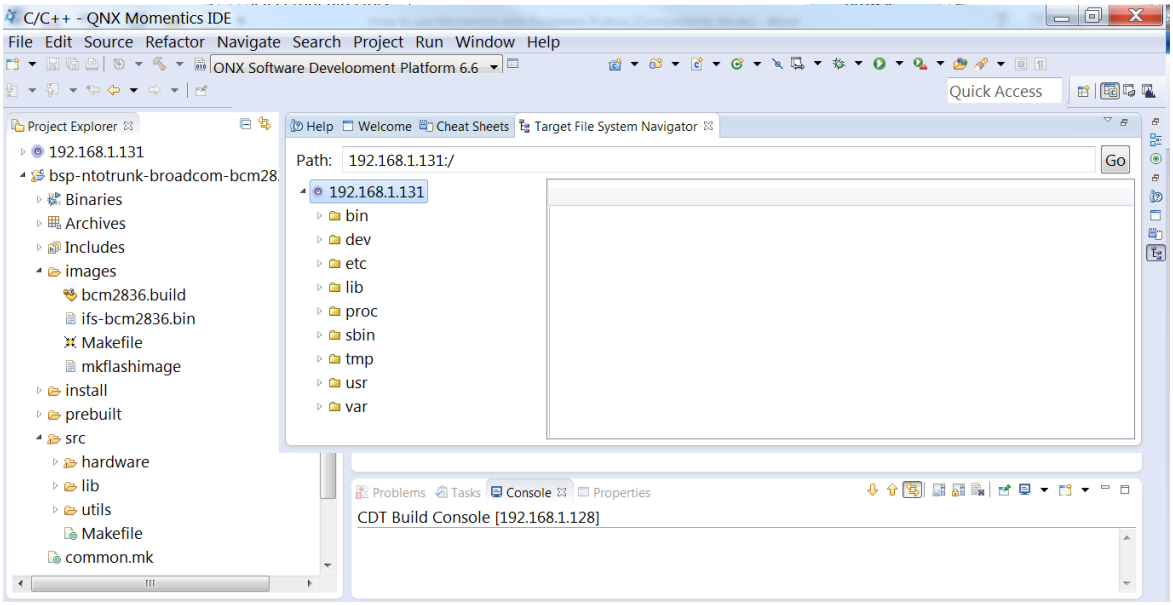


Figure 4.9 Finish adding Raspberry Pi2 QNX 6.6 target

(3) Open /tmp folder

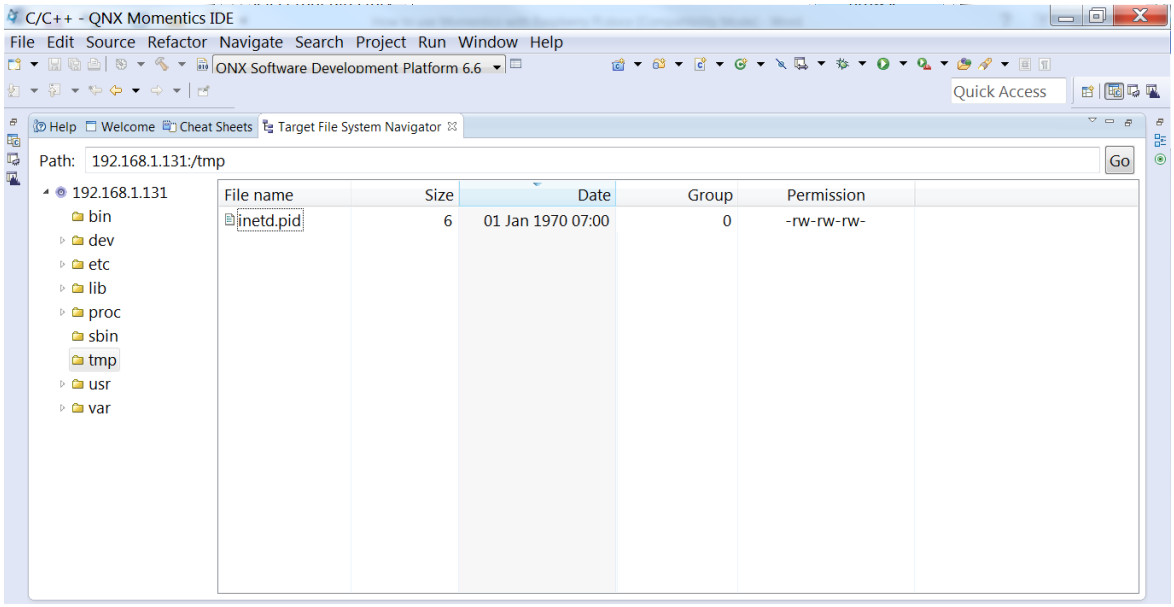


Figure 4.10 Raspberry Pi2 QNX 6.6 target tmp folder list

(4) Copy a file from Windows PC to QNX 6.6 Raspberry Pi2

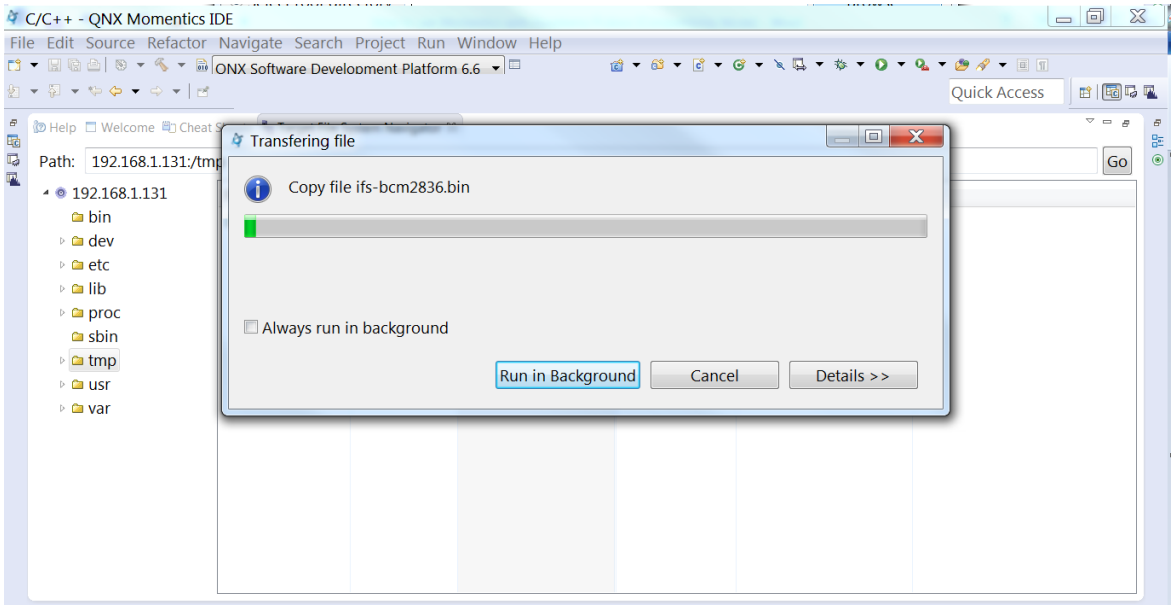


Figure 4.11 Send a files from PC to Raspberry Pi2 QNX 6.6 target

(5) Make sure the size of the target file in /tmp and the source file is match

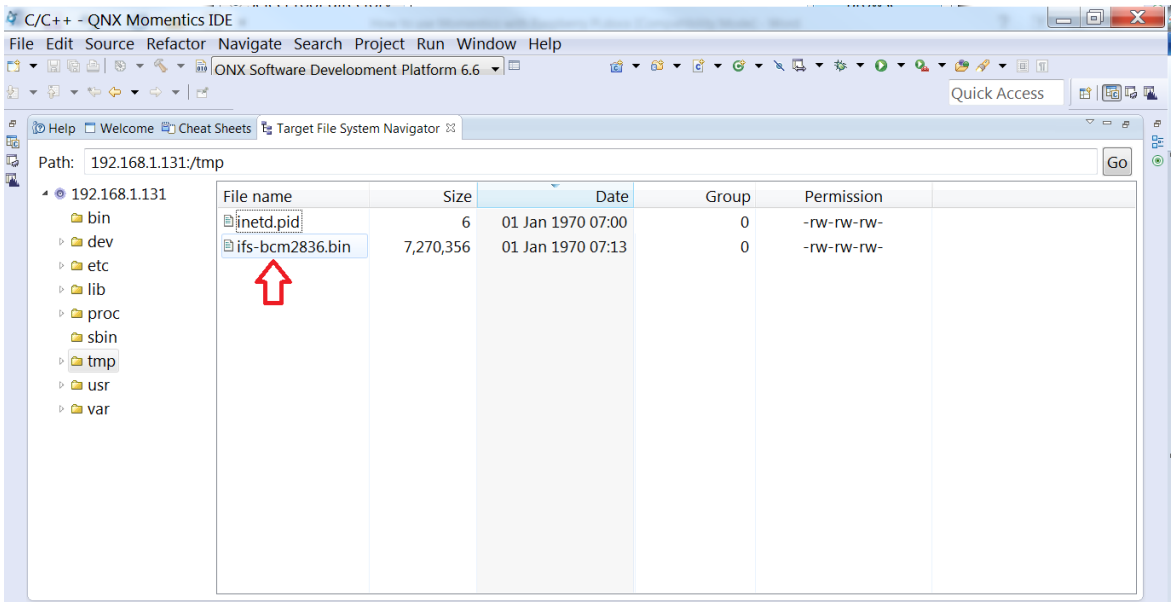


Figure 4.12 Finish sending a files from PC to Raspberry Pi2 QNX 6.6 target

(6) Copy a file from /tmp to window PC.

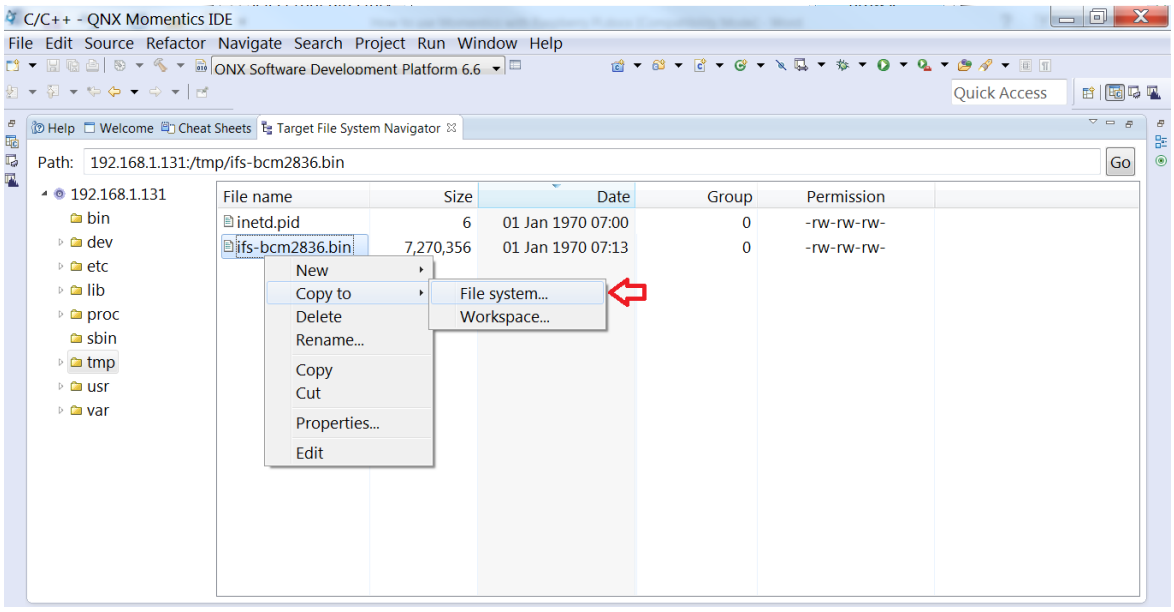


Figure 4.13 Send a files from Raspberry Pi2 QNX 6.6 target to PC

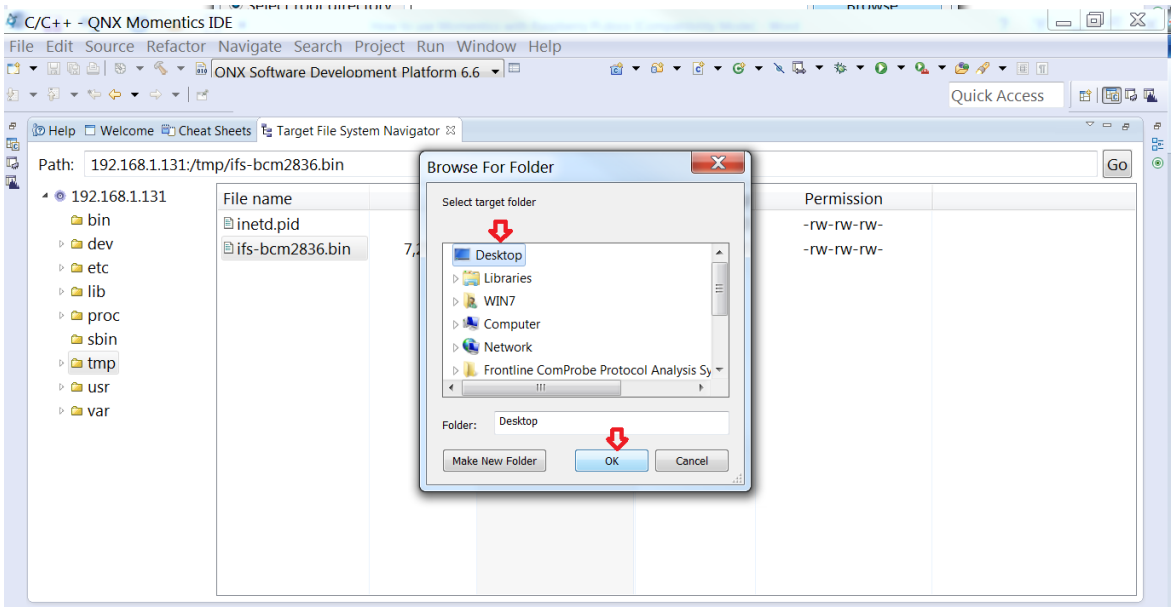


Figure 4.14 Locate file on PC

(7) Make sure the size of the target file in Window PC and source file is match

4.4 SD driver

Notes: Driver isn't supported mode auto detect device insert/remove yet.

Make sure /dev/sd10tX is created in file system by command:

```
RPi2# ls /dev/
```

4.5 Graphics driver

The display driver supports:

- HDMI: Displayed via pipeline 1.

4.5.1 Start/stop the driver

If display driver has not run yet, run it by

```
RPi2#GRAPHICS_ROOT=/usr/lib/graphics/bcm2836

RPi2#LD_LIBRARY_PATH=/usr/lib:/lib:/lib/dll:$LD_LIBRARY_PATH

RPi2#screen &
```

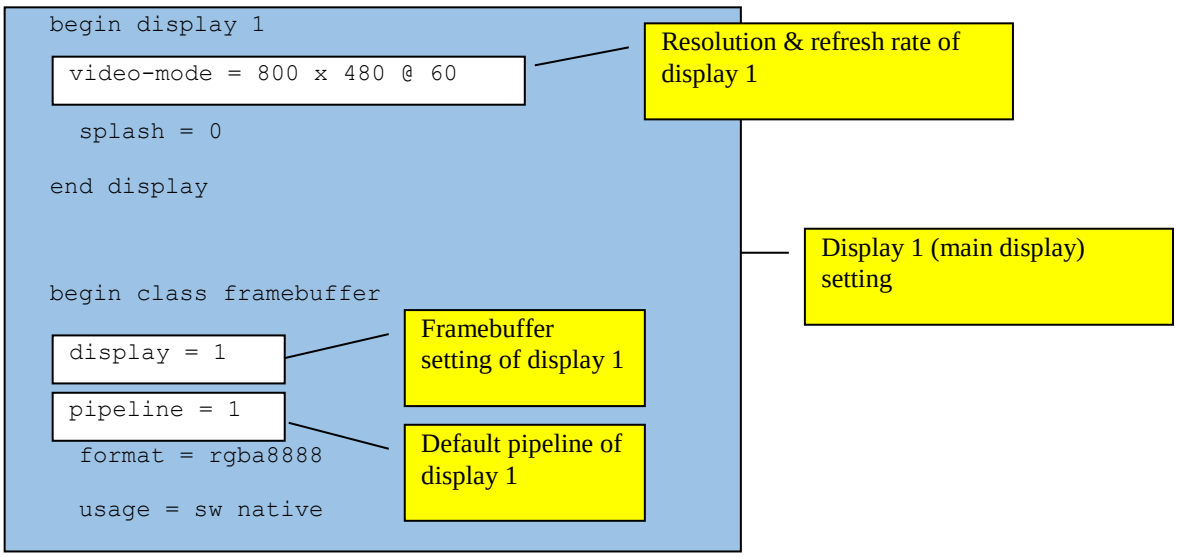
To stop the driver, use "kill <screen process ID>" or "slay screen" then make sure screen process has been terminated.

4.5.2 Display configuration

This section describes how to configure resolution, which display is used, which is main display.

With default setting, HDMI monitor is configured as main display at 800x480.

Display setting is configured by ...\prebuilt\armle-v7\usr\lib\graphics\bcm2836 \graphics.conf. Below is a part of display default setting "graphics.conf"



The resolution in "graphics.conf" file above is used to allocate frame buffer of Video Core. In order to take effect on real monitor resolution, your display has to support that resolution and "config.txt" in SD card memory needs to be configured too. There are two options to be set:

```
hdmi_group = 2
hdmi_mode = x (x is the mode of resolution)
```

Refer to config.txt - Raspberry Pi Documentation:
https://www.raspberrypi.org/documentation/configuration/config-txt.md

- For example: The resolution in “graphics.conf” file is “video-mode = 1920 x 1080 @ 60”, “config.txt” will be set as follow:

```
hdmi_group = 2
hdmi_mode = 82
```

❖ **Note:** In order to detect HDMI hot plug, place this attribute “hdmi_force_hotplug=1” in “config.txt” file in SD card. This will make Raspberry board appears that a HDMI display is attached. In other words, HDMI output mode will be used, even if no HDMI monitor is detected. In addition, this will solve the problem when Raspberry board is started and there are no display is connected to it. If you connect a display to the board that time, the display will have no input.

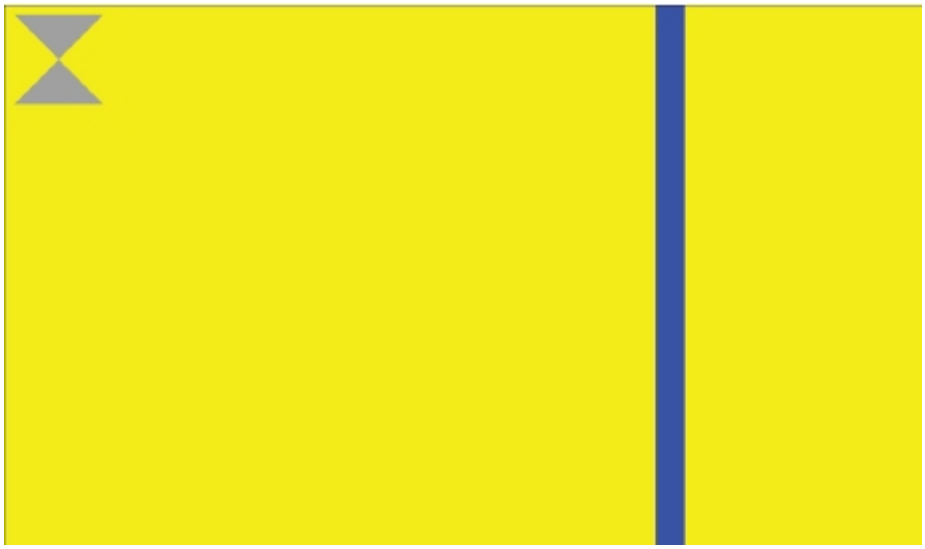
4.5.3 Test driver

- Connect HDMI monitor to port HDMI.
- Make sure that screen has been started.

To test display on HDMI monitor:

- Using “sw-vsync”:

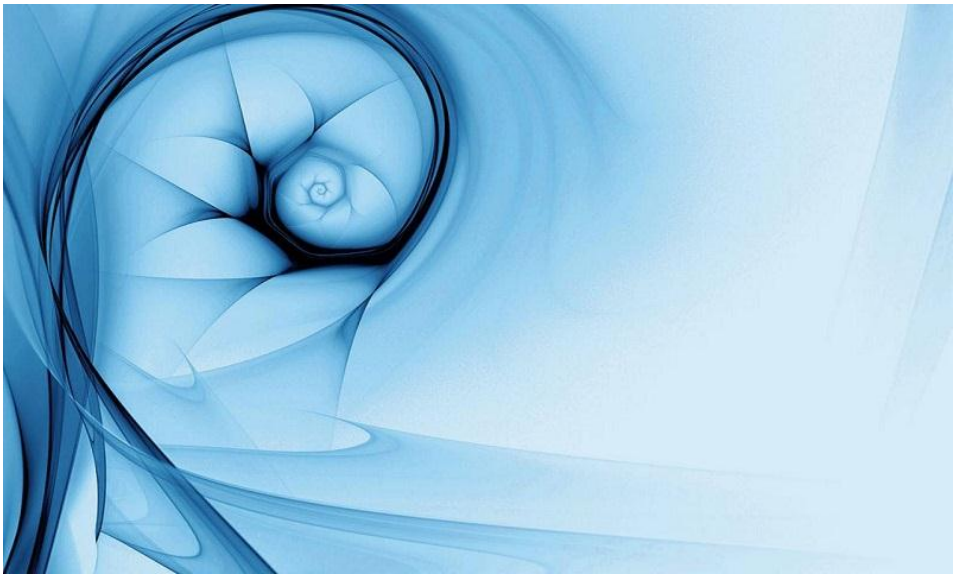
```
RPi2#sw-vsync -size=[width]x[height] &
```



To stop it, use “kill <screen process ID>” or “slay sw-vsync” then make sure screen process has been terminated.

- Using “display_image”:

```
RPi2#display_image -file=/usr/images/[width]x[height]/default.jpg -display=1 -
pos=[x],[y]
```



To stop the driver, use “kill <screen process ID>” or “slay display_image” then make sure screen process has been terminated.

4.5.4 Configuring driver in source code

Driver developers can configure display driver by modify the driver source code. By that way, they can add more resolutions, allocate different pipelines for each display from default.

4.6 I2C driver

4.6.1 Test Overview

Make sure I2C driver are running with entry point /dev/i2c1

- Connect I2C1 pin on Raspberry Pi2 with HDC1000 I2C device.

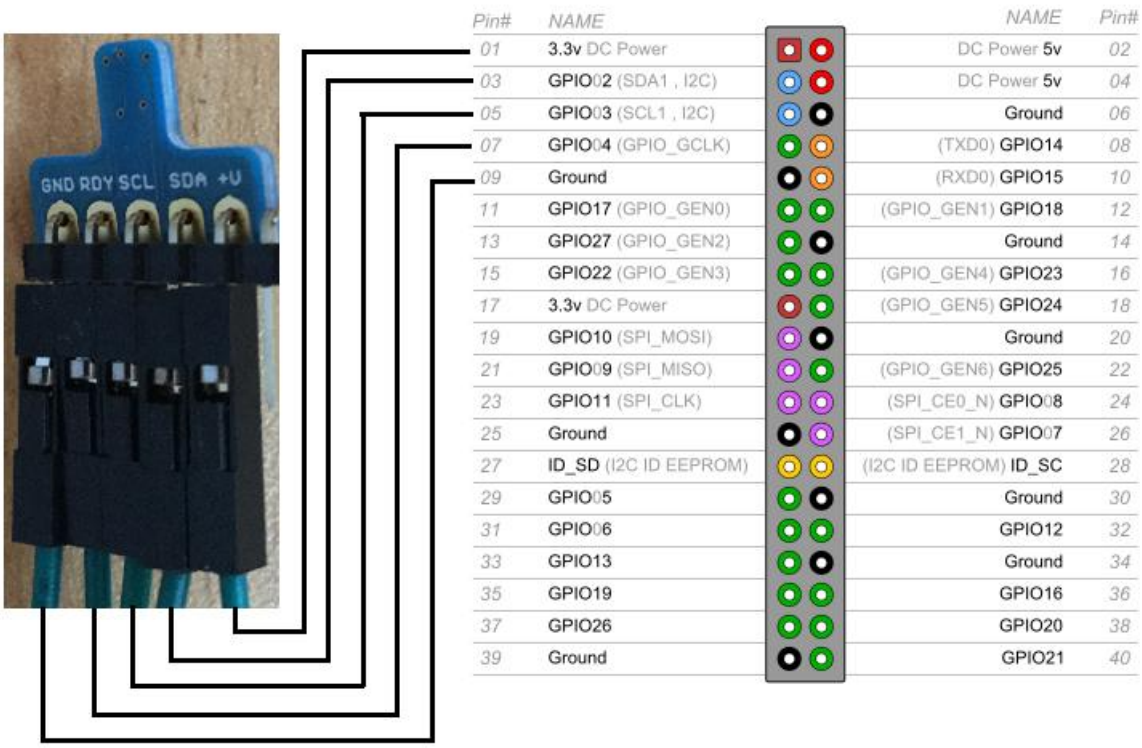


Figure 4-15 Connect I2C1 pin of Raspberry Pi2 with HDC1000

4.6.2 Test Procedures

- Run command as below:

```
RPi2#i2c_test <channel>, <slave>, <speed>
```

Note: There must be a space after a comma.

4.6.3 Test I2C with HDC1000

- Run command as below:

```
RPi2#i2c_test channel=1, slave=0x40, speed=100000
```

- If the I2C run normally, you can see the result in the terminal as below figure.

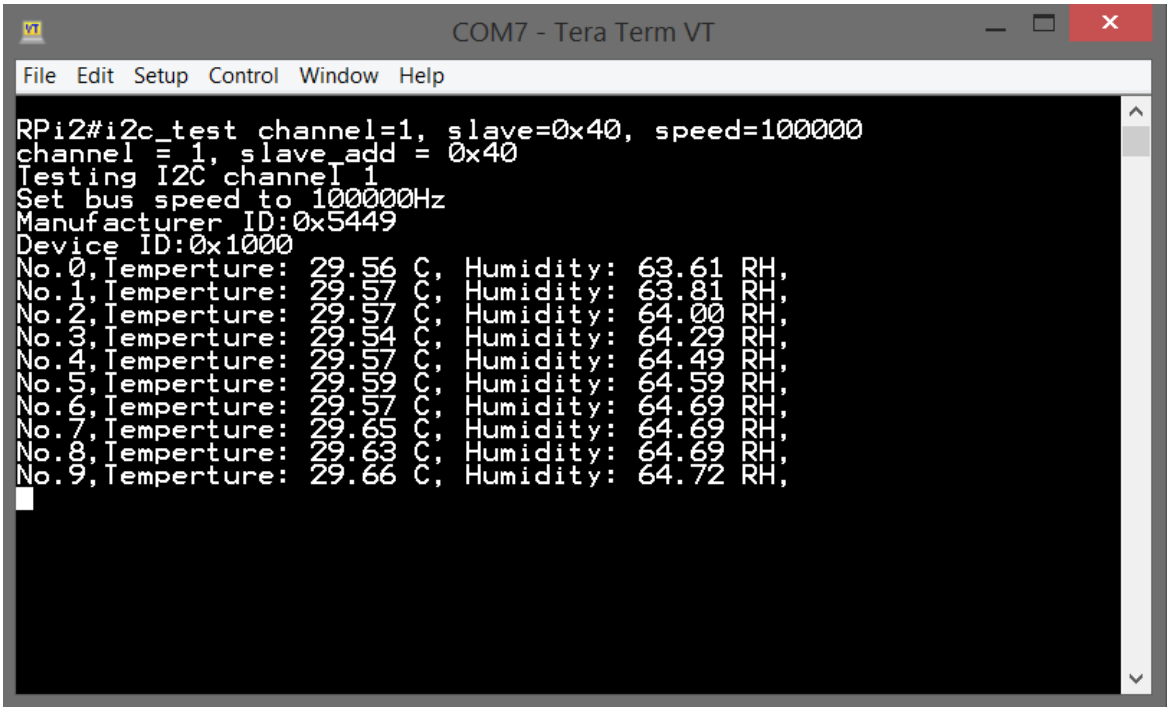


Figure 4-16 Test I2C1 with HDC1000, speed=100000 and speed=400000

4.7 SPI driver

4.7.1 Test Overview

Make sure SPI driver are running with entry point /dev/spi0

- Connect SPI pin on Raspberry Pi2 with SPI device.

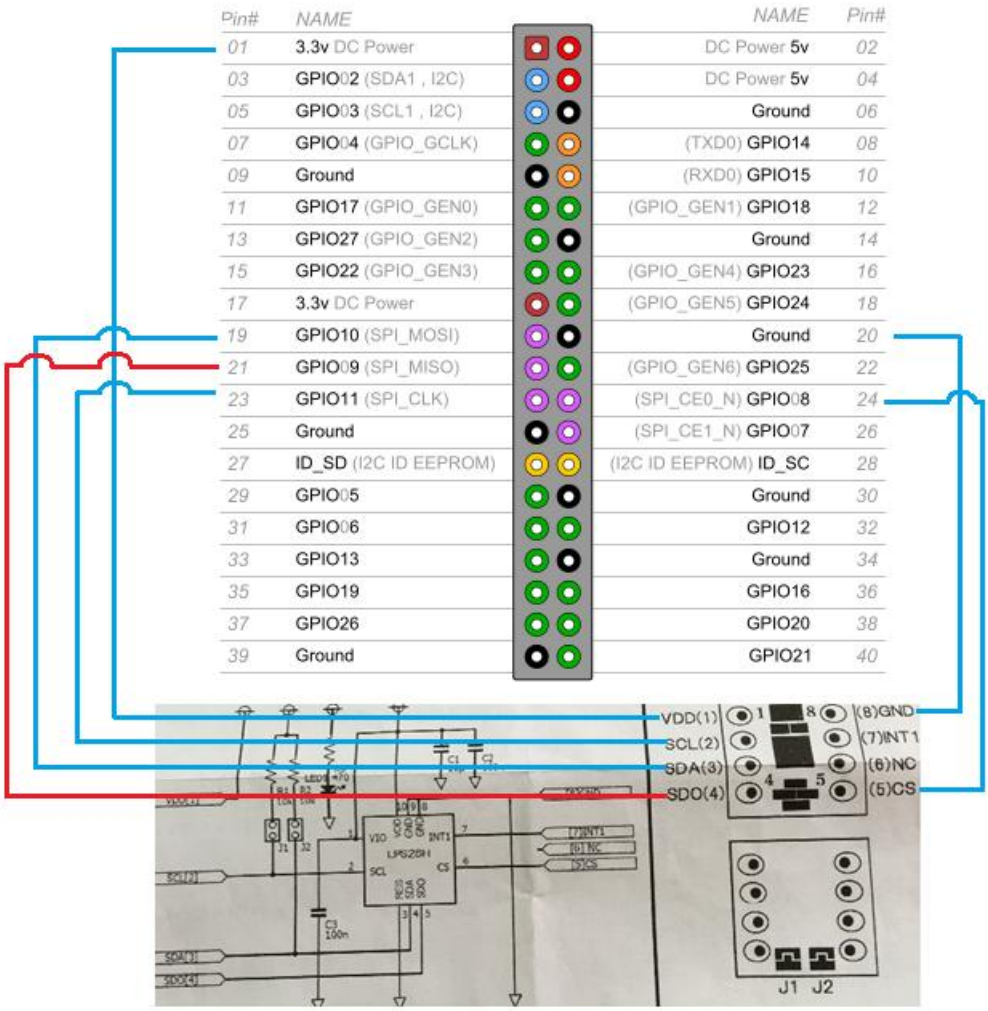


Figure 4-17 Connect SPI pin of Raspberry Pi2 with LPS25H

4.7.2 Test Procedures

- Run command as below:

```
RPi2#spi_test
```

4.7.3 Test SPI with LPS25H

- Run command as below:

```
RPi2#spi_test
```

- If the SPI run normally, you can see the result in the terminal as below figure.

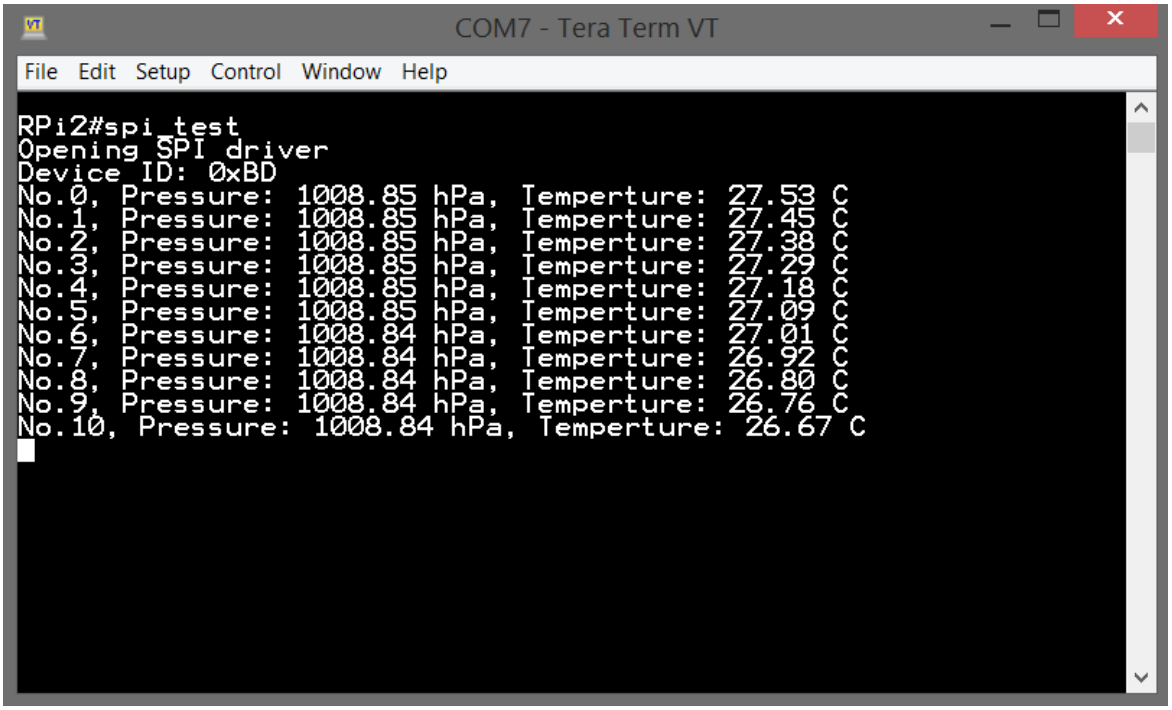


Figure 4-18 Test SPI with LS25H

4.8 GPIO driver

4.8.1 Test Overview

Make sure GPIO driver are running with entry point /dev/gpio

4.8.2 Test Procedures

- Run command as below:

```
RPi2#gpio_test
```

- If the GPIO run normally, you will see active LED (green LED) on the board blinking and the terminal will show up as below figure

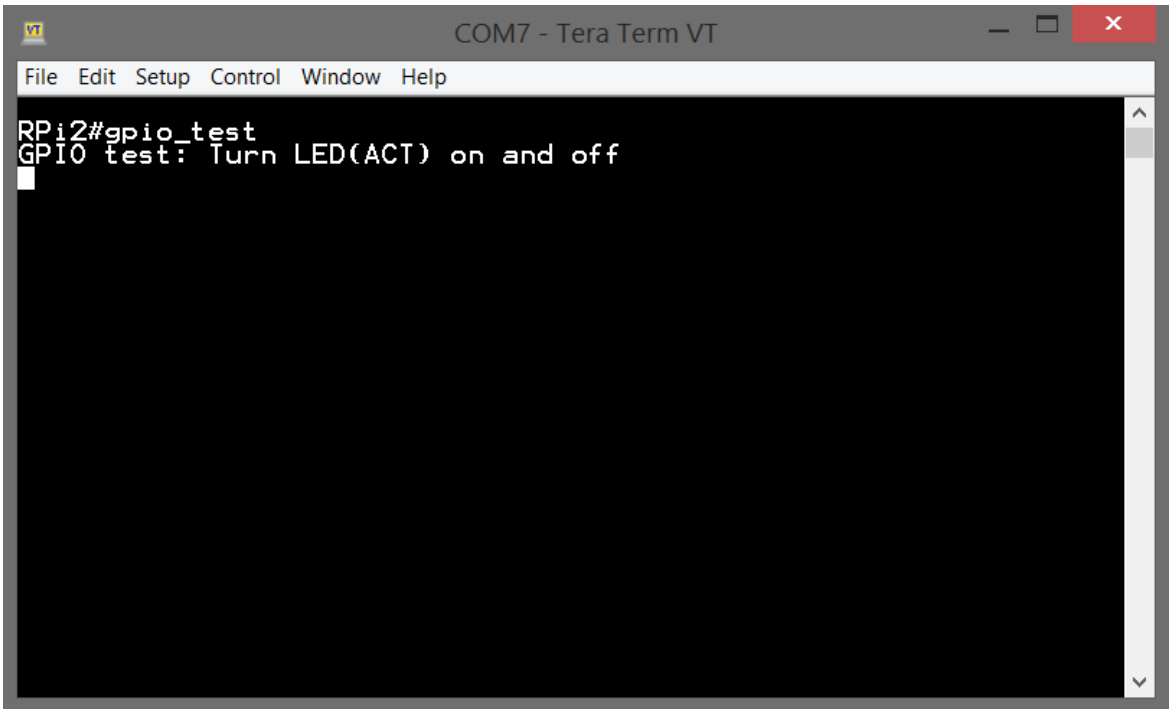


Figure 4-19 Test GPIO

4.9 CAN driver

4.9.1 Test Overview

- With current version, these following baud rate are supported: 25K, 50K, 100K, 125K, 250K, 500K.
- CAN driver use spi bus to access to MCP2515 CAN module (use 3.3 V power). To test this driver, make sure SPI driver are running with entry point /dev/spi0
- In this test, we will use two Raspberry Pi2 boards drive two MCP2515 CAN modules to communicate with each other via can bus.
- We also leverage the QNX utility canctl to explorer our MCP2515 Can module driver. We could consult manual of canctl as below link:
    - http://www.qnx.com/developers/docs/660/index.jsp?topic=%2Fcom.qnx.doc.neutrino.utilities%2Ftopic%2Fc%2Fcanctl.html

- Our connection for this test is illustration as below diagrams:

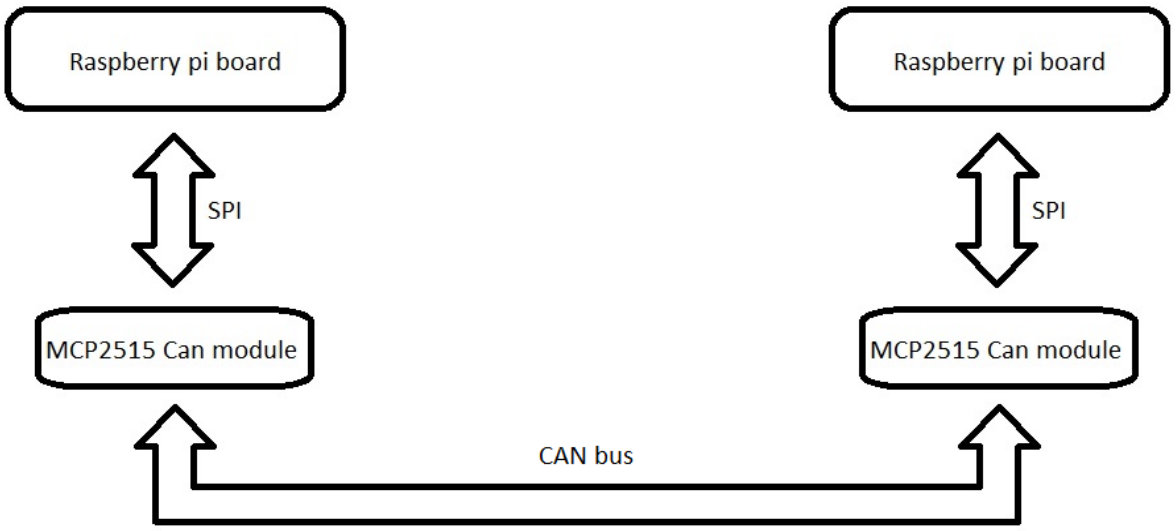


Figure 4-20 System test block connection

- Connect Raspberry Pi board SPI pins to MCP2515 module as below diagram:

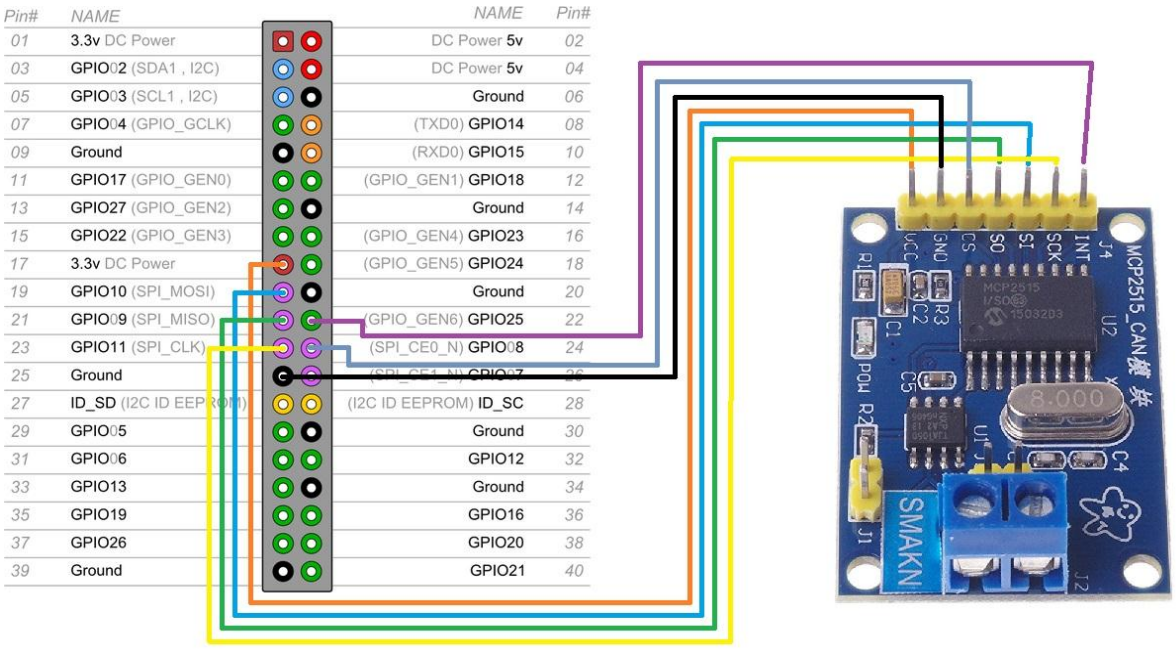


Figure 4-21 MCP2515 can module SPI connect to Raspberry Pi2 board

- Connect Can port of MCP2515 Can module of two node together as below diagram:

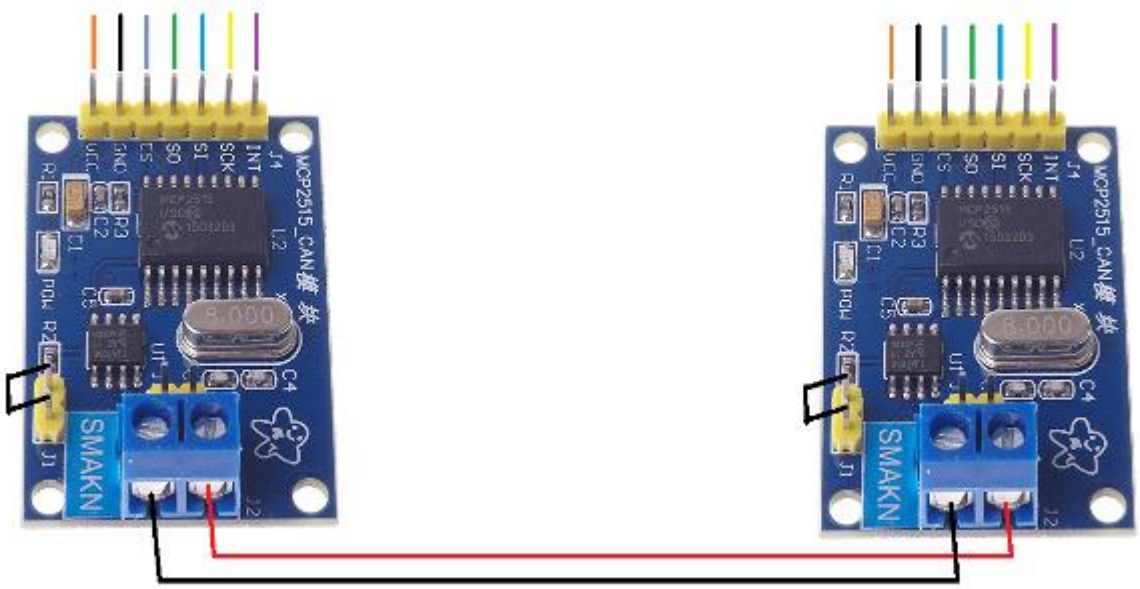


Figure 4-22 MCP2515 can module Can bus connection

- We will have a wired up system something as below:

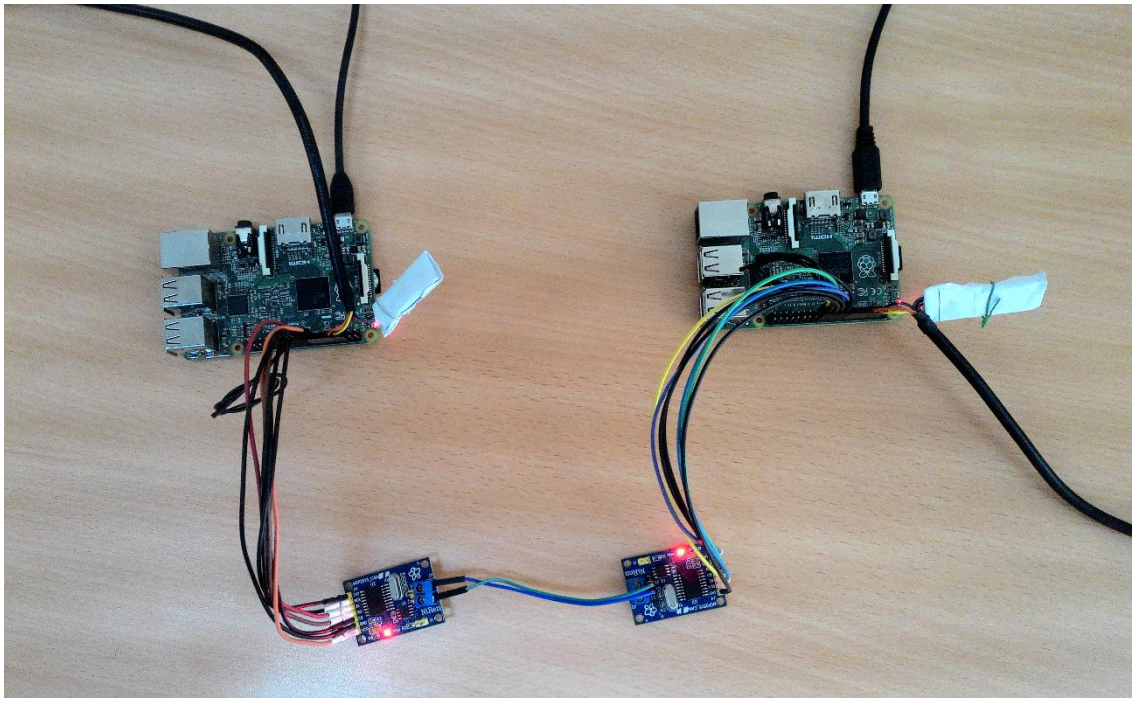


Figure 4-23 Connected system overview

4.9.2 Test Procedures

- Please make sure CAN modules is connected properly.
- Start CAN driver on each board use default setting.
- Send CAN message from Raspberry Pi2 board #1
- Get CAN receive message on Raspberry Pi2 board #2
- Change baud rate on each board accordingly and repeat above test

4.9.3 Test CAN driver use default setting

Step 1: Start CAN driver on Raspberry Pi2 each board
CAN driver is ran on the board as default. If not please run as below:
- Board #1: # can-bcm2836 &
- Board #2: # can-bcm2836 &

Step 2: Send CAN message from Raspberry Pi2 board #1
- Board #1:
echo hi > /dev/can1/tx33

Step 3: Get CAN receive message on Raspberry Pi2 board #2
- Board 2: # cat /dev/can1/rx1
- Make sure “hi” message is received.

4.9.4 Test CAN driver with various baud rates

Following is example for testing with baud rate 250K. Repeat test for other baud rate.

Please kill can-bcm2836 driver before start with new baud rate.

Step 1: Start CAN driver on Raspberry Pi2 each board
- Board #1: # can-bcm2836 –b250K &
- Board #2: # can-bcm2836 –b250K &

Step 2: Send CAN message from Raspberry Pi2 board #1
- Board #1:
echo hi > /dev/can1/tx33

Step 3: Get CAN receive message on Raspberry Pi2 board #2
- Board 2: # cat /dev/can1/rx1
- Make sure “hi” message is received.

REVISION HISTORY	QNX 6.6.0 Setup Manual For Raspberry Pi2 Board

Rev.	Date	Description	
		Page	Summary
1.0	September 09, 2016	-	First release
1.1	September 14, 2016	P.7 to P.12	- Added 4.2 USB driver - Added 4.3 Network driver
1.2	September 27, 2016	P.13 to P.20	- Added 4.4 SD driver - Added 4.5 Graphics driver - Added 4.6 I2C driver - Added 4.7 SPI driver - Added 4.8 GPIO driver
1.3	December 15, 2016	P.12 P.27 to P.30	- Removed note for Insert/Remove issue on USB - Added 4.9 CAN driver