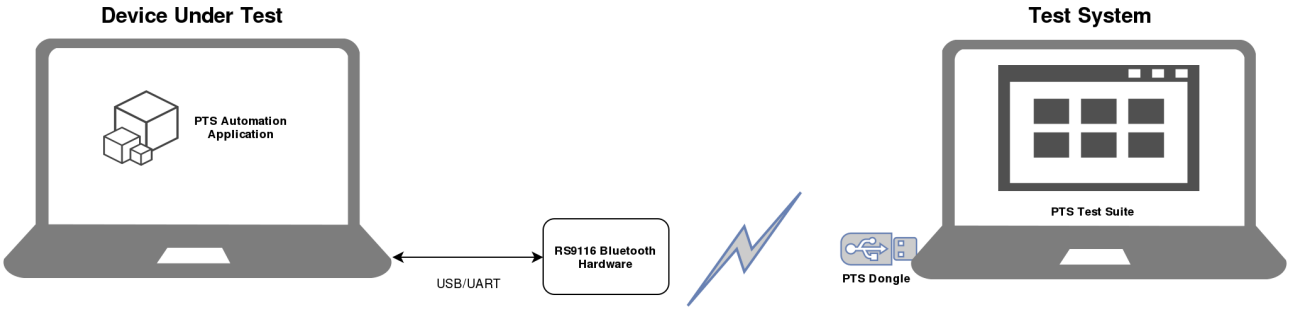
PTS Profile Test Automation

RS9116 Firmware and Software Releases

Exported on 05/24/2019

Table of Contents

Description of Test Setup



- The Test System PTS for Bluetooth is running on a PC System
- The test system communicates with the Bluetooth Hardware (PTS Dongle) via an HCI connection.
- The tests are performed as remote tests and all communication between the Test System and the DUT is done via the radio interface.
- The PTS automation application willl run on a linux PC system, and the communication between application and RS9116 Bluetooth hardware is done through USB/UART interface

Prequisites

1. Connect RS9116 EVK to linux PC with either USB/UART Interface.
2. Flash RS9116_WC_SI.rps using kermit (Refer to PRM documents in Doc folder)
   [File path:- RSI_SDK_xxx/resources/firmware/RS9116_WC_SI.rps into RS9116 EVK].
3. Please refer the web link for Bluetooth PTS setup - https://wiki.mozilla.org/B2G/QA/Bluetooth_PTS

PTS IXIT(Implementation extra Information for Testing) Settings

The following IXIT settings should be configured for running automation cases. [In PTS software, Open the IXIT setting window from menu View→ Tools→ ICS/IXT Tool Window]

```
TSPX_bd_addr_iut                = Redpine module address
TSPX_delete_link_key            = TRUE
TSPX_pin_code                   = 0000

// For Testing Absolute Volume feature
TSPX_tester_av_role             = SNK

// For RFCOMM layer
TSPX_server_channel_iut         = 1
```

Applications Overview

1. **a2dp_source_avrcp_pts:-** This application used to test BT profiles A2DP, AVRCP, AVDTP AVCTP test cases
2. **spp_slave:-** This application used to test BT profiles RFCOMM, SDP, SPP test cases
3. **smp_test:-** This application used to test BLE SM (Security Manager) master/slave test cases
4. **gatt_server:-** This application used to test BLE GATT Server test cases
5. **gatt_client:-** This application used to test BLE GATT Client test cases
6. **gap_test:-** This application used to test the GAP layer Master/Slave test cases
7. **dual_mode:-** This application used to test the dual mode test cases
8. **l2cap_slave**:- This application used to test the BT L2CAP slave test cases
9. **l2cap_ble_peripheral**:- This application used to test the BLE L2CAP slave test cases

Configuring the Application

- Open ***rsi_a2dp_source_avrcp_pts.c*** file and update/modify following macros,

  RSI_BT_LOCAL_NAME refers name of the Redpine module to appear during scanning by remote devices.

```
#define RSI_BT_LOCAL_NAME                                   "A2DP_AVRCP_SOURCE"
```

  RSI_BT_REMOTE_BD_ADDR refers BD address of the PTS Test device to which redpine device has to connect.

```
#define RSI_BT_REMOTE_BD_ADDR                               "00:1E:7C:25:E9:6D"
```

  PIN_CODE refers four bytes string required for pairing process.

```
#define PIN_CODE                                            "0000"
```

  INIT_TC_ID refers to initial test case id from which application need to run

```
#define INIT_TC_ID                                           TC_SRC_AS_BV_01_I
```

- Following are the **non-configurable** macros in the application. **BT_GLOBAL_BUFF_LEN** refers to the number of bytes required by the application and the driver.

```
#define BT_GLOBAL_BUFF_LEN                                  10000
```

Open ***rsi_bt_config.h*** file and update/modify following macros

The following are the test case ID sequence from which application will automatically shift to next test case after successful completion of previous test case.

```
//! Need to define Either one of the below macro for running PTS test cases
#define PTS_A2DP_AVRCP_1_3  //! This Macro should be defined for PTS A2DP & AVRCP automation test cases
#define PTS_ABS_VOL_FEAT   //! This Macro should be defined for PTS AVRCP Absolute volume feature test
cases

 enum test_case_id_e {
  TC_NORMAL_MODE = 1,

  /* A2DP SRC */
  TC_SRC_AS_BV_01_I,
  TC_SRC_CC_BV_09_I,
  TC_SRC_CC_BV_10_I,
  TC_SRC_REL_BV_01_I,
  TC_SRC_REL_BV_02_I,
  TC_SRC_SDP_BV_01_I,
  TC_SRC_SET_BV_01_I,
  TC_SRC_SET_BV_02_I,
  TC_SRC_SET_BV_03_I,
  TC_SRC_SET_BV_04_I,
  TC_SRC_SET_BV_05_I,
  TC_SRC_SET_BV_06_I,
  TC_IOPT_CL_A2DP_SRC_SFC_BV_01_I,

  /* AVCTP TG */
  TC_TG_NFR_BI_01_C,               //15
  TC_TG_NFR_BV_03_C,

  /* AVDTP SRC ACP */
  TC_SRC_ACP_SIG_SMG_BV_06_C,      //17
  TC_SRC_ACP_SIG_SMG_BV_08_C,
  TC_SRC_ACP_SIG_SMG_BV_10_C,
  TC_SRC_ACP_SIG_SMG_BV_12_C,
  TC_SRC_ACP_SIG_SMG_BV_16_C,
  TC_SRC_ACP_SIG_SMG_BV_18_C,
  TC_SRC_ACP_SIG_SMG_BV_20_C,
  TC_SRC_ACP_SIG_SMG_BV_22_C,
  TC_SRC_ACP_SIG_SMG_BV_24_C,
  TC_SRC_ACP_SIG_SMG_BI_02_C,
  TC_SRC_ACP_SIG_SMG_BI_05_C,
  TC_SRC_ACP_SIG_SMG_BI_08_C,
  TC_SRC_ACP_SIG_SMG_BI_11_C,
  TC_SRC_ACP_SIG_SMG_BI_17_C,
  TC_SRC_ACP_SIG_SMG_BI_20_C,
  TC_SRC_ACP_SIG_SMG_BI_23_C,
  TC_SRC_ACP_SIG_SMG_BI_26_C,
  TC_SRC_ACP_TRA_BTR_BI_01_C,

  /* AVDTP SRC INT */
  TC_SRC_INT_SIG_SMG_BV_05_C,     //35
  TC_SRC_INT_SIG_SMG_BV_07_C,
  TC_SRC_INT_SIG_SMG_BV_09_C,
  TC_SRC_INT_SIG_SMG_BV_11_C,
  TC_SRC_INT_SIG_SMG_BV_15_C,
  TC_SRC_INT_SIG_SMG_BV_17_C,
```

```
  TC_SRC_INT_SIG_SMG_BV_19_C,
  TC_SRC_INT_SIG_SMG_BV_21_C,
  TC_SRC_INT_SIG_SMG_BV_23_C,
  TC_SRC_INT_SIG_SMG_BV_31_C,
  TC_SRC_INT_SIG_SMG_BI_30_C,
  TC_SRC_INT_TRA_BTR_BV_01_C,

  /* AVRCP TG */
  TC_TG_CEC_BV_01_I,              //47
  TC_TG_CEC_BV_02_I,
  TC_TG_CFG_BI_01_C,
  TC_TG_CFG_BV_02_C,
  TC_TG_CRC_BV_01_C,
  TC_TG_CRC_BV_02_C,
  TC_TG_ICC_BV_01_I,
  TC_TG_ICC_BV_02_I,
  TC_TG_INV_BI_01_C,
  TC_TG_MDI_BV_02_C,
  TC_TG_MDI_BV_04_C,
  TC_TG_MDI_BV_05_C,
  TC_TG_NFY_BI_01_C,
  TC_TG_NFY_BV_02_C,
  TC_TG_NFY_BV_04_C,
  TC_TG_NFY_BV_05_C,
  TC_TG_NFY_BV_08_C,
  TC_TG_PTT_BV_01_I,
  TC_TG_PTT_BV_02_I,
  TC_TG_RCR_BV_02_C,
  TC_TG_RCR_BV_04_C,
  TC_IOCT_CL_AVRCP_TG_SFC_BV_04_I,

  /* AVRCP CT Role ABS Volume */
  TC_CT_VLH_BI_03_C,
  TC_CT_VLH_BI_04_C,
  TC_CT_VLH_BV_01_C,
  TC_CT_VLH_BV_03_C,
  TC_CT_VLH_BV_01_I,
  TC_CT_VLH_BV_02_I,

  /* AVRCP CT Role ABS Volume */
  TC_TG_VLH_BI_01_C,
  TC_TG_VLH_BI_02_C,
  TC_TG_VLH_BV_02_C,
  TC_TG_VLH_BV_04_C,
  TC_TG_VLH_BV_01_I,
  TC_TG_VLH_BV_02_I,

 TC_TG_MAX
} TC_ID;
```

Flashing the Firmware

Using Kermit

Make sure to have .kermrc file with the below configuration in /root folder

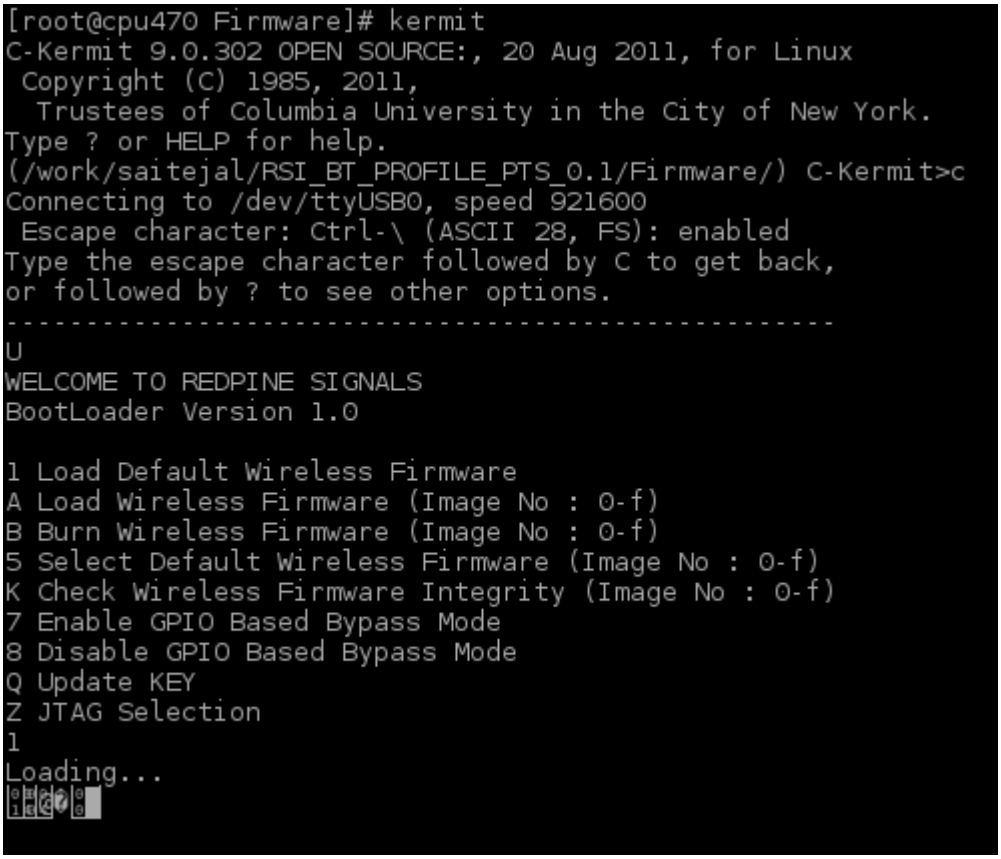
```
set modem type none
set line /dev/ttyUSB0
set speed 921600
set carrier-watch off
set handshake none
set flow-control none
robust
set file type bin
set file name lit
set rec pack 100
set send pack 4000
set window 10
```

1. Run kermit, and then press c to continue.
2. Enter the pipe key |. Kermit should echo back a U. Enter a capital U. This will make the boot loader menu appear. This process is called Auto Baud Rate Detection (ABRD) and is used set the baud rate of the RS9116
3. Choose option B and select image 0, then enter send command with firmware binary file to flash firmware binary as shown in screenshot below
4. After flashing the firmware, kermit will show the message as "Upgradation Successfull"

```
[root@cpu470 Firmware]# kermit
C-Kermit 9.0.302 OPEN SOURCE:, 20 Aug 2011, for Linux
 Copyright (C) 1985, 2011,
  Trustees of Columbia University in the City of New York.
Type ? or HELP for help.
(/work/saitejal/RSI_BT_PROFILE_PTS_0.1/Firmware/) C-Kermit>c
Connecting to /dev/ttyUSB0, speed 921600
 Escape character: Ctrl-\ (ASCII 28, FS): enabled
Type the escape character followed by C to get back,
or followed by ? to see other options.
----------------------------------------------------
U
WELCOME TO REDPINE SIGNALS
BootLoader Version 1.0

1 Load Default Wireless Firmware
A Load Wireless Firmware (Image No : 0-f)
B Burn Wireless Firmware (Image No : 0-f)
5 Select Default Wireless Firmware (Image No : 0-f)
K Check Wireless Firmware Integrity (Image No : 0-f)
7 Enable GPIO Based Bypass Mode
8 Disable GPIO Based Bypass Mode
Q Update KEY
Z JTAG Selection
B
Enter Wireless Image No(0-f)
0
Send RS9116.NBZ.WC.GENR.x.x.x.rps

(Back at cpu470)
----------------------------------------------------
(/work/saitejal/RSI_BT_PROFILE_PTS_0.1/Firmware/) C-Kermit>send ./RS9116_WC_SI.rps
(/work/saitejal/RSI_BT_PROFILE_PTS_0.1/Firmware/) C-Kermit>c
Connecting to /dev/ttyUSB0, speed 921600
 Escape character: Ctrl-\ (ASCII 28, FS): enabled
Type the escape character followed by C to get back,
or followed by ? to see other options.
----------------------------------------------------

Upgradation Successful

Enter Next Command
```

Loading the Firmware

1. Run kermit, and then press c to continue.
2. Enter the pipe key |. Kermit should echo back a U. Enter a capital U. This will make the boot loader menu appear. This process is called Auto Baud Rate Detection (ABRD) and is used set the baud rate of the RS9116
3. Select option 1, then firmware will be loaded as shown in screenshot below



Compiling & Loading the USB driver kernel module (Required only for USB interface)

1. In case of USB interface, go to the below path and compile & load usb driver as shown below

```
#! Change directory to below path
cd /work/RSI_SDK_xxx/platforms/linux/Driver/usb/src

#! Compile usb driver as below
make clean;make

#! Load usb driver kernel module (Required only once)
insmod rpsusb.ko
```

Compiling & Running the Application

1. Select either USB/UART interface in Makefile as shown below, to run application either through USB/UART interface.
2. Driver Makefile will be available in below path

```
#! Change directory to below path
cd /work/RSI_SDK_xxx/sapis/build

#! Open makefile
gvim Makefile
```

3. Open Makefile and make below changes accordingly for selecting USB/UART interface

```
#! Uncomment below line for using UART interface
CFLAGS+= -D RSI_UART_INTERFACE
#! Uncomment below line for using USB interface
CFLAGS+= -D RSI_USB_INTERFACE
```

4. In case of UART interface, load the firmware using kermit every time before running an application. (Refer to PRM documents in Doc folder)
5. Application will be available in below path.

```
RSI_SDK_xxx/apps/pts_apps/a2dp_source_avrcp_pts
```

6. Follow the below steps for compiling the rsi_a2dp_source_avrcp_pts.c application.
    a. Go to application folder

```
cd RSI_SDK_xxx/apps/pts_apps/a2dp_source_avrcp_pts
```

    b. Compile application using below commands. Then executable file 'rsi_wc_app' will be generated.

```
make clean; make
```

    c. To run an application.

```
./rsi_wc_app
```

Executing the Application

1. After successful firmware loading, run the a2dp source avrcp pts application binary (rsi_wc_app).
2. RS9116 device will be in required mode (either initiator or acceptor) as per the test case specified.
3. PTS test device will run test cases according to the IXIT file and application will corresponds to the test case sequence.

Note

- This can be run in wise-connect mode in linux pc with USB/UART Interface.