APPLICATION NOTE:
SX-ULPGN-BTZ
Hosted Mode

Silex Technology America
201 East Sandpointe, Suite 245
Santa Ana, CA 92707
Revision 1.0, August 5, 2019

Revision History

Rev. No.	Date	Revision by	Description
1.0	Aug 05, 2019	T.Nakase	Initial draft

Table of Contents

1. Scope

The purpose of this document is to provide a guideline for setting up and evaluating UART and SPI in hosted mode with the SX-ULPGN-BTZ EVK. This document uses Raspberry Pi 3 Model B+ as the host processor board.

Note: The SPI interconnect is not confirmed to be working as of this writing.

2. References

2.1 QDN Document

2.1.1 QCA402x (CDB2x) Development Kit User Guide, 80-YA121-140 Rev.C

2.2 SX-ULPGN-BTZ QSG

2.2.1 SX-ULPGN-BTZ Development Quick Start Guide, 140-00217-100 v1.4

2.3 Raspberry Pi 3 Model B+

2.3.1 https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/

2.4 NOOBS Installer

2.4.1 https://www.raspberrypi.org/downloads/noobs/

2.5 Raspbian

2.5.1 https://www.raspberrypi.org/downloads/raspbian/

3. Equipment

3.1 Hardware

- The EVK board, SX-ULPGN-BTZ EVK (WCBN3516A_EVB V01)
- Host PC
- USB 2.0 Cable (Type A male – Type B male) x2
- USB Gender Changer (Type B female – Type A male) x2
- Jumper Cap x9
- Jumper Cable (female – female) x3

3.2 Host Processor Board

- Raspberry Pi 3 Model B+ Starter Kit (Element 14 23-20181RK, USB port x 4, HDMI port x 1)
- HDMI monitor
- HDMI cable
- USB keyboard
- USB mouse
- Jumper Cable (female – female) x 7 (you can share it with Jumper Cable in Section 3.1)

3.3 Host PC Configuration

- Intel Core i7-4790 Processor @ 3.60 GHz
- 8 GB RAM

- 160GB HDD
- USB 2.0/3.0 x2
- Gigabyte Ethernet Port x1
- Windows 10 Professional
- Username: silex, Account type: Administrator

4. EVK Board Setup

4.1 SX-ULPGN-BTZ Development Environment

Follow Sections 4 through 6 listed in the SX-ULPGN-BTZ QSG document to setup the development environment to your Host PC.

4.2 Update DevCfg File

HostedMode_demo is preconfigured to SPI hosted mode. If you are using the UART interconnect, the DevCfg file must be updated before building the firmware. For details, see Section 9.1.8 of the **QDN Document**.

<SDK_source>\target\quartz\demo\HostedMode_demo\src\export\DevCfgmaster_devcfg_out_cdb.xml

Note: If you don't have this file in the export folder above, run the following command with using the Command Prompt. The files in the export folder will not be populated if you have never built firmware in this project directory.

```
> build.bat prepare
```

4.3 Build HostedMode_demo image

1. Open Command Prompt and go to <SDK_source>\target\quartz\demo\HostedMode_demo\build\gcc.

2. Run build.bat

```
> build.bat t 4020 cdb
```

Note: To build with Eclipse IDE, see Section 7 in the **SX-ULPGN-BTZ QSG document**.

4.4 Program HostedMode_demo image with JTAG

1. Confirm that JTAG is enabled (see Section 6.1.2 in the **SX-ULPGN-BTZ QSG document** for details.)

2. Open the Command Prompt and go to <SDK_source>\target\quartz\demo\HostedMode_demo\build\gcc.

3. Connect the USB1 and USB2 port on the EVK board to the Host PC with the USB cables (if you haven't already connected the EVK board).

4. Run flash_openocd.bat

```
> flash_openocd.bat
```

5. Wait until you see a message that says "Flash Operation Completed Successfully…" on the Command Prompt.

Note: To program with Eclipse IDE, see Section 8 of the **SX-ULPGN-BTZ QSG document**.

6. Unplug the USB Cables from USB1 and USB2 on the EVK board.

7. Remove the Jumper cables that were installed in this section.

5. Host Processor Board Setup

5.1 Install Raspbian

1. Program the NOOBS installer to your mini SD card.

Note: Generally, the Raspberry Pi Starter Kit includes a mini SD card with pre-programmed NOOBS. For more information, refer to the document from the Starter Kit manufacturer.

2. Insert the micro SD card into the host processor board and power it up.

3. Wait until you see the OS install dialog for NOOBS.

4. Select and install Raspbian Lite.

Note: This document only requires the use of a CLI console. A desktop environment is optional.

5. The system will automatically reboot after successful installation.

6. Login using the default user credentials (user: pi, password: raspberry)

7. Enable the SPI interface with raspi-config. (Select "Interfacing Options" -> "SPI")

5.2 Build iotd and NB_QCLI_demo

Build IoT daemon, iotd and userland application, NB_QCLI_demo on Raspbian. See Section 9.1.5, 9.1.6 and 9.1.7 in the **QDN Document** for details *(before building these, see the below note).*

```
$ cd <SDK_source>/target/exthost/Linux/daemon; make
$ cd <SDK_source>/target/exthost/Linux/qapi; make
$ cd <SDK_source>/target/exthost/Linux/app/NB_QCLI_demo/build; make
```

Note: The Host Processor Board GPIO pinout can be matched to the BTZ board CHIP_PWD_L pinout by editing the iotd_config.ini file, but the current SDK release has a bug that will not reflect the setting of this file. To work around this issue, change the QZ_WAKE_GPIOn macro as defined in <SDK_source>/target/exthost/Linux/hif/spi/spi_regs.h as follows:

```
#define QZ_WAKE_GPIOn "24"
```

5.3 Connect EVK board to Host Processor Board

1. There are two ways to connect the EVK board to the Host Processor Board depending on the interconnect you are using: 1.a for UART interconnect or 1.b for SPI interconnect.

1.a Connect USB2 on the EVK board to the USB port of the Host Processor Board with the USB Cable.

1.b Connect the SPI bus with the Jumper Cable as follows:

EVK board	SPI Signal	Host Processor Board
J16.8	CLK	J8.23
J16.10	CS	J8.24
J102.2	MISO	J8.21
J101.2	MOSI	J8.19
J117.2	INT	J8.22 (GPIO25)
J117.1	GND	J8.20
J114.9	CHIP_PWD_L	J8.18 (GPIO24)

2. Connect USB1 and USB2 on the EVK board to the USB port on the Host Processor Board (the position doesn't matter) using the USB Cable.

6. Run Application

6.1 Configure IoT Daemon

1. Login to the Host Processor Board.

2. Edit <SDK_source>/target/exthost/Linux/daemon/iotd_config.ini as indicated with yellow-hatching.

```
[SYSTEM]
num_device=1        # Number of Quartz Devices
num_interface=2     # Number of bus interface
num_clients=5       # Max number of clients
num_buffer=20       # Max buffer count
pwd_gpio=24         # PWD GPIO pin number
force_reset=1
heart_beat_enable=1     #
heart_beat_interval=15   # heart beat interval in second
throughput_test_enable=0    #
throughput_test_mode=0      # 0-send only, 1:loopback
throughput_test_interval=5  # test interval in seconds
```

```
throughput_test_packet_len=1000 # packet length
dbg_level=1                  #debug verbosity level


[INTERFACE]
enable=0                                          # 0:Disable 1:Enable
type=0                                            # 0:UART, 1:SPI, 2:SDIO
speed=115200                                      # Baud/frequency
flow_control=0                                    # UART Flow Control: 0-
disable, 1-enable
block_size=1024                                   # SPI block Size
name=/dev/ttyUSB1                                 # File name
device_id=0                                       # Instance of Quartz device
it is associated with
num_service_q=6                                   # Number of associated
service queues
service_qid=0x00,0x01,0x02,0x03,0x04,0x05     # Service queue IDs
associated with this interface


[INTERFACE]
enable=0                                          # 0:Disable 1:Enable
type=1                                            # 0:UART, 1:SPI, 2:SDIO
speed=1000000                                     # Baud/frequency
flow_control=1                                    # UART Flow Control: 0-
disable, 1-enable
block_size=1024                                   # SPI block Size
intr_gpio=25                                      # interrupt GPIO pin number
name=/dev/spidev0.0                               # File name
device_id=0                                       # Instance of Quartz device
it is associated with
num_service_q=6                                   # Number of associated
service queues
service_qid=0x00,0x01,0x02,0x03,0x04,0x05     # Service queue  IDs
associated with this interface
```

3. Set 1 to "enable" the parameter of the desired interconnect

Use UART interconnect:

```
[INTERFACE]
enable=1                                    # 0:Disable 1:Enable
type=0                                      # 0:UART, 1:SPI, 2:SDIO
…
[INTERFACE]
enable=0                                    # 0:Disable 1:Enable
type=1                                      # 0:UART, 1:SPI, 2:SDIO
…
```

Use SPI interconnect:

```
[INTERFACE]
enable=0                                    # 0:Disable 1:Enable
type=0                                      # 0:UART, 1:SPI, 2:SDIO
…
[INTERFACE]
enable=1                                    # 0:Disable 1:Enable
type=1                                      # 0:UART, 1:SPI, 2:SDIO
…
```

6.2 Start IoT Daemon and NB_QCLI_demo

You will need two separate consoles (or terminals if you're using a desktop environment) to run the IoT Daemon and NB_QCLI_demo at the same time.

1. Start IoT Daemon

```
$ cd <SDK_source>/target/exthost/Linux/daemon
$ sudo ./output/iotd iotd_config.ini
IOTD: Opening config file iotd_config.ini
HTC: Initializing SPI interface for target ID 0
Iotd Manager: Recv MGMT_MSG_HELLO resp
Target QAPI Ver: 2.0.1    CRM  Num: 78
***** IOT Daemon started ****
```

2. Open another console by pressing Ctrl-Alt-F2 (or open a new terminal in the desktop environment) and login to the Host Processor Board. You can return to the original console using the Ctrl-Alt-F1 command.

3. Start the NB_QCLI_demo

```
$ cd <SDK_source>/target/exthost/Linux/app/NB_QCLI_demo/build

$ sudo ./nb_demo

Sending Hello Message

Received HELLO response from the server.
```

4. Hit return to see the CLI command prompt and type "help".

```
> help

Command List:

  Commands:

     0. Ver

     1. Help

     2. Exit

  Subgroups:

     3. BLE

     4. ZigBee

     5. Debug

     6. HMI

     7. Thread

     8. FwUp

     9. Coex

>
```

5. Now you can use the QCLI_demo command that is listed in Section 6 of **the QDN Document**.

Note: As of this writing, no WLAN commands were able to be implemented due to an unstable QAPI.