Qualcomm Technologies, Inc.

SA6155 Linux Android Software Porting Manual

SP80-PK753-6 B

December 21, 2018

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	November 2018	Initial release
B	December 2018	Added Chapter 12 and updated the rest of the document with the SPI Slave configuration details

Contents

1 Introduction 6

1.1 Purpose 6
1.2 Conventions 6
1.3 Technical assistance 6

2 Device tree 7

2.1 Device tree compiler 7
2.2 Boot loader 7
2.3 Kernel 9

3 GPIO – Hardware overview 10

3.1 GPIO configuration 10
3.2 Interrupt configuration 11

4 GPIO – Software configuration 13

4.1 Configure GPIO in the boot loader 13
4.2 Configure GPIO in the kernel 13

5 QUPv3 – Hardware overview 16

5.1 Qualcomm universal peripheral 16
5.2 QUPv3 mode of operation 16

6 I^2C – Hardware overview 17

6.1 I^2C core 17
6.2 Base addresses 17

7 I^2C – Software configuation 18

7.1 Configure QUPv3 core as I^2C in the kernel 18
7.2 Code modification to configure the SE as I^2C 18
7.3 Verify I^2C bus 33
7.4 Debug tips 33
7.5 Configure I^2C slave device 33

8 UART – Hardware overview 37

8.1 UART core 37
8.2 Base addresses 37

9 UART – Software configuration 38

9.1 Configure UART in the boot loader 38
9.1.1 Set the base address 38
9.1.2 Configure the GPIO 43
9.1.3 Debug tips 43
9.2 Configure console UART in the kernel 44
9.2.1 Code modification 44
9.2.2 Debug tips 47

10 SPI – Hardware overview 49

10.1 SPI core 49
10.2 Base addresses 49

11 SPI Master – Software configuration 50

11.1 Configure the QUPv3_SE core as SPI Master in the kernel 50
11.1.1 Code modification to enable the QUPv3_SE as SPI Master....50
11.2 Verify the SPI bus 58
11.2.1 Register a SPI slave device using the device tree....58

12 SPI Slave – Software configuration 63

12.1 Configure the QUPv3_SE core as SPI Slave in the kernel 63
12.1.1 Code modification to enable the QUPv3_SE as SPI Slave....63
12.2 Verify the SPI bus 67

A FAQs 68

B References....70

B.1 Related documents....70
B.2 Acronyms and terms....70

Figures

Figure 2-1 Device tree in boot image....8

Tables

Table 2-1 Register table in boot loader....8
Table 2-2 Device tree key API....9
Table 3-1 GPIO base address....10
Table 3-2 GPIO physical address....10
Table 3-3 GPIO configuration table....11
Table 3-4 GPIO interrupt configuration table....11
Table 3-5 Interrupt configuration....12
Table 3-6 GPIO wake-up interrupt....12
Table 6-1 QUP physical address....17
Table 8-1 UART physical address....37
Table 10-1 Base address....49

1 Introduction

1.1 Purpose

This document describes the device tree, and its usage, general-purpose input/output (GPIO), interintegrated circuit (I^2C) bus, serial peripheral interface (SPI) bus, camera architecture, display architecture, and their sample code.

The document is intended for engineers who develop and debug on the SA6155 platforms.

1.2 Conventions

Function declarations, function names, type declarations, attributes, and code samples appear in a different font, for example, `#include`.

Code variables appear in angle brackets, for example, `<number>`.

Commands to be entered appear in a different font, for example, **`copy a:*.* b:`**.

Button and key names appear in bold font, for example, click **Save** or press **Enter.**

Shading indicates content that has been added or changed in this revision of the document.

1.3 Technical assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies, Inc. (QTI) at https://createpoint.qti.qualcomm.com/.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

2 Device tree

To simplify the process for the embedded board vendors, a device tree is used to represent devices in the system by declaring nodes and properties in a .dts file. The device tree is also recommended for on-chip devices and other buses that do not specifically fit in an existing open firmware (OF) specification.

The device tree helps the kernel to probe and match drivers to the device without having to hard code all the tables. Also, it helps board vendors perform minor hardware upgrades without significantly impacting the kernel code or cluttering it with special cases.

Use the device tree to:

- Reduce the effort required to port to a new board
- Reuse code by making drivers generic enough to support a set of compatible devices with the differences represented in the device tree

To enable device tree, the file CONFIG_USE_OF is set to “y”.

2.1 Device tree compiler

Device tree compiler (DTC) parses the device tree script into the binary format. Use the following command to generate dtb zImage if the kernel does not change.

```
dtc -p 1024 -O dtb -o msmfalcon-mtp.dtb msmfalcon-mtp.dts
cat zImage msmfalcon-mtp.dtb > dtb-zimage
```

2.2 Boot loader

Boot loader is responsible for:

- Finding the SoC-specific (main DT blob) and board-specific (overlay DT blob) device tree blobs from the respective partition (DTBO and boot)
- Patching the main DT blob with the overlay blob using the DTC
- Creating one combined DT that will be passed to kernel

Boot loader parses the boot image and extracts the device tree information. The following is an example of how boot loader extracts the DT address:

DT address = Image address + page_size + kernel_actual + ramdisk_actual + second_actual

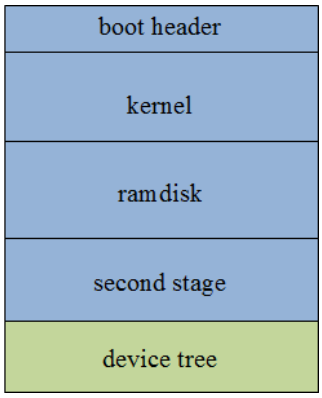

Figure 2-1 Device tree in boot image

The kernel uses R2 to save the DT table pointer.

Table 2-1 Register table in boot loader

Register	JTAG	DT
R0	NULL	NULL
R1	Machine type number	Unique machine ID is no longer required when boot loader passes a device tree; the passed-in ID must be 0×FFFFFFFF to the kernel in the r1 register
R2	Physical address of the tagged list in the system RAM	Physical pointer to the device-tree blob in the RAM; the device tree is located anywhere in the system RAM but is aligned on a 64‑bit boundary

NOTE: The code source path for 64‑bit is kernel/arch/arm64/boot/ and for 32‑bit is kernel/arch/arm/boot/.

2.3 Kernel

The relative codes are `kernel/drivers/of/device.c` and `include/linux/of_gpio.h`. Table 2-2 lists the key functions.

Table 2-2 Device tree key API

Function name	Description
of_platform_populate()	System init function; checks the DT table passed by fastboot

of_match_device()	Key function in the driver probe function is declared in the driver.
	For example:
	`static const struct of_device_id s5k3l1yx_dt_match[] = {`
	`{.compatible = "qcom,s5k3l1yx", .data = &s5k3l1yx_s_ctrl},`
	`{}`
	`};`
	
	`static struct platform_driver s5k3l1yx_platform_driver = {`
	`.driver = {`
	`.name = "qcom,s5k3l1yx",`
	`.owner = THIS_MODULE,`
	`.of_match_table = s5k3l1yx_dt_match,`
	`},`
	`};`
	
	`static int32_t s5k3l1yx_platform_probe(struct platform_device *pdev)`
	`{`
	`int32_t rc = 0;`
	`const struct of_device_id *match;`
	`match = of_match_device(s5k3l1yx_dt_match, &pdev->dev);`
	`rc = msm_sensor_platform_probe(pdev, match->data);`
	`return rc;`
	`}`

NOTE: For more information on the functions, see kernel/Documentation/devicetree/.

3 GPIO – Hardware overview

3.1 GPIO configuration

There are 123 chip level GPIOs and 32 LPI GPIOs in the SA6155 chipset. The chip level GPIOs are divided into three groups called the West, East, and South tiles.

Table 3-1 provides an overview of the hardware aspect of the GPIOs.

Table 3-1 GPIO base address

Module	Base address	Size	End address
LPASS_TLMM_GPIO	0 × 62B40000	0 × 20000	0 × 62B60000
TLMM_WEST	0 × 3500000	0 × 300000	0 × 3800000
TLMM_EAST	0 × 3100000	0 × 300000	0 × 3400000
TLMM_SOUTH	0 × 3D00000	0 × 300000	0 × 4000000

The following tables describe the GPIO configuration registers in the SA6155 chipset:

Table 3-2 GPIO physical address

Hardware register	Physical address	Reset state
LPASS_TLMM_GPIO_ISLAND_CFGn	0x62B40000+ 0 × 00000000+0 × 1000*(n)	0 × 00000001
LPASS_TLMM_GPIO_ISLAND_IN_OUTn	0x62B40000+ 0 × 00000004+0 × 1000*(n)	0 × 00000000
TLMM_WEST	0 × 03500000 + 0 × 00500000	0 × 00000001
TLMM_EAST	0 × 03100000 + 0 × 00900000	0 × 00000001
TLMM_SOUTH	0 × 03D00000 + 0 × 00100000	0 × 00000001

Table 3-3 GPIO configuration table

Bit	Field name	Description
10	GPIO_HIHYS_EN	Controls hihys_en for GPIO[n]
9	GPIO_OE	Controls OE for GPIO[n] when in the GPIO mode
8:6	DRV_STRENGTH	Controls the GPIO pad drive strength; applies regardless of the FUNC_SEL field selection ▪ 0 – DRV_2_MA ▪ 1 – DRV_4_MA ▪ 2 – DRV_6_MA ▪ 3 – DRV_8_MA ▪ 4 – DRV_10_MA ▪ 5 – DRV_12_MA ▪ 6 – DRV_14_MA ▪ 7 – DRV_16_MA
5:2	FUNC_SEL	▪ Many of the GPIO pads have one or more functional hardware interfaces behind them. This field controls the usage of the pad. ▪ Set this field to the appropriate value for the function intended.
1:0	GPIO_PULL	The pad is configured to employ an internal weak pull-up, pull-down, or keeper function. This applies regardless of the FUNC_SEL field selection. ▪ 0 × 0 – NO_PULL (disables all pull) ▪ 0 × 1 – PULL_DOWN (weak pull-down) ▪ 0 × 2 – KEEPER (weak keeper) ▪ 0 × 3 – PULL_UP (weak pull-up)

NOTE: In the CFG register:

- GPIO_OE, you can enable the GPIO configuration. Do not write TLMM_GPIO_OE_[0…3].
- GPIO_HIHYS_EN, you can enable GPIO hysteresis in the Input mode, which is useful for noisy low-frequency signals such as sleep clock.

3.2 Interrupt configuration

The following tables list the interrupt configuration details:

Table 3-4 GPIO interrupt configuration table

Hardware register	Reset state
TLMM_GPIO_INTR_CFGn, n=[0…122]	0×000000E2
TLMM_GPIO_INTR_STATUSn, n=[0…122]	0 × 00000000

Table 3-5 Interrupt configuration

Bit	Field name	Description
8	DIR_CONN_EN	▪ 0 – Disable ▪ 1 – Enable
7:05	TARGET_PROC	▪ 0 – SENSORS ▪ 1 – LPA_DSP ▪ 2 – RPM ▪ 3 – HMSS ▪ 4 – MSS ▪ 5 – TrustZone ▪ 6 – Turing ▪ 7 – None
4	INTR_RAW_STATUS_EN	▪ 0 – Disable ▪ 1 – Enable
3:02	INTR_DECT_CTL	▪ 0 – Level ▪ 1 – POS_EDGE ▪ 2 – NEG_EDGE ▪ 3 – DUAL_EDGE
1	INTR_POL_CTL	▪ 0 – POLARITY_0 ▪ 1 – POLARITY_1
0	INTR_ENABLE	▪ 0 – Disable ▪ 1 – Enable

NOTE: POS_EDGE (rising edge trigger) and NEG_EDGE (falling edge trigger) are supported.

Table 3-6 GPIO wake-up interrupt

Hardware register	Physical address	Reset state
TLMM_WEST_MPM_WAKEUP_INT_EN_0	0 x 03596000	0 × 00000000
TLMM_EAST_MPM_WAKEUP_INT_EN_0	0 x 03196000	0 × 00000000
TLMM_SOUTH_MPM_WAKEUP_INT_EN_1	0 x 03D96000	0 × 00000000

4 GPIO – Software configuration

4.1 Configure GPIO in the boot loader

To configure a GPIO in XBL, use the following APIs:

- DalTlmm_ConfigGpio()
- DalTlmm_GpioIn()
- DalTlmm_GpioOut()
- DalTlmm_SetPort()

4.2 Configure GPIO in the kernel

To configure any of the GPIOs in the SA6155 chipset is described in the following steps:

For details, see *PINCTRL (PIN CONTROL) Subsystem* and *Documentation/devicetree/bindings/pinctrl/msm-pinctrl.txt.*

1. Add a pin grouping node to the msmfalcon-pinctrl.dtsi file.

```
&soc {
tlmm: pinctrl@03000000 {
     compatible = "qcom,msmfalcon-pinctrl";
     reg = <0x03000000 0xc00000>;
     interrupts = <0 208 0>;
     gpio-controller;
     #gpio-cells = <2>;
     interrupt-controller;
     #interrupt-cells = <2>;

     uart_console_active: uart_console_active {
          mux {
               pins = "gpio4", "gpio5";
               function = "blsp_uart2";
          };

          config {
               pins = "gpio4", "gpio5";
               drive-strength = <2>;
               bias-disable;
          };
     };
```

2. Modify the client node in the device tree (msmfalcon.dtsi or sdm670-<board>.dtsi).

```
     Soc {
          Client1 {
/*Please note default state is programmed by the kernel at the time of
kernel bootup. No driver changes necessary, since at probe time the
default state would already be programmed in TLMM */
          pinctrl-states = “default”;
          /* Use phandle referrence to default configuration node */
          pinctrl-0 = <&Client1_default>;
     };
}
```

Configure GPIOs for the I²C driver

1. Check `drivers/i2c/muxes/i2c-mux-pinctrl.c`.
2. Get GPIO information using the following command:
   `of_property_read_string_index(np, "pinctrl-names", i, out)`
3. Parse the pin control
   `devm_pinctrl_get()`
4. Install a state using pinctrl_lookup_state() and pinctrl_select_state().

```
for (i = 0; i < mux->pdata->bus_count; i++) {
          mux->states[i] = pinctrl_lookup_state(mux->pinctrl,
                              mux->pdata->pinctrl_states[i]);
                …
     }
…
for (i = 0; i < mux->pdata->bus_count; i++) {
     u32 bus = mux->pdata->base_bus_num?
                    (mux->pdata->base_bus_num + i): 0;

          mux->busses[i] = i2c_add_mux_adapter(mux->parent, &pdev->dev,
                                   mux, bus, i, 0,
                              i2c_mux_pinctrl_select,
                                   deselect);
…
}
```

5. Add DTS.

```
     /* I2C CONFIGURATION */
i2c_1 {
i2c_1_active: i2c_1_active {
Mux {
pins = "gpio2", "gpio3";
function = "blsp_i2c1";
```

```
};
config {
pins = "gpio2", "gpio3";
drive-strength = <2>;
bias-disable;
};
};
i2c_1_sleep: i2c_1_sleep {
mux {
pins = "gpio2", "gpio3";
function = "blsp_i2c1";
};
config {
pins = "gpio2", "gpio3";
drive-strength = <2>;
bias-pull-up;
};
};
};
};
```

5 QUPv3 – Hardware overview

5.1 Qualcomm universal peripheral

The Qualcomm universal peripheral (QUP)v3 is new core in the SA6155 for I^2C, SPI, and UART.

QUPv3 has one GSI core per engine and up to eight serial engines (SE). GSI and SE are firmware based cores. SE is a GENI based core.

Firmware size:

- SE – 1276 bytes
- GSI – 6500 bytes

Top level – There are two QUPv3 cores, each with up to four SEs. One SSC_QUP with Six SEs

NOTE: The SA6155 chipset contain two SE cores.

5.2 QUPv3 mode of operation

- GSI/GPI mode, preferred mode for DMA transfers:
  - Supports Multi-EE use cases
  - Requires GSI as well as SE cores to be initialized (as well as firmware loaded)
- FIFO mode:
  - PIO mode with support for large transfers, interrupt moderations using watermark
  - Requires SE to be initialized (firmware loaded)
- CPU DMA mode:
  - DMA mode internal to the GENI core
  - Requires SE to be initialized (firmware loaded)

6 I²C – Hardware overview

6.1 I²C core

The I²C core supports I²C standard speed (100 kHz), full speed (400 kHz), and fast speed (1 MHz) output clock frequencies.

6.2 Base addresses

Table 6-1 QUP physical address

QUPv3 hardware ID	QUP core	Physical address	IRQ #	Kernel I²C clock name
QUPV3_0	QUPV3_0_SE0	0x00880000	601	GCC_QUPV3_WRAP0_S0_CLK
QUPV3_0	QUPV3_0_SE1	0x00884000	602	GCC_QUPV3_WRAP0_S1_CLK
QUPV3_0	QUPV3_0_SE2	0x00888000	603	GCC_QUPV3_WRAP0_S2_CLK
QUPV3_0	QUPV3_0_SE3	0x0088C000	604	GCC_QUPV3_WRAP0_S3_CLK
QUPV3_1	QUPV3_1_SE0	0x00A80000	353	GCC_QUPV3_WRAP1_S0_CLK
QUPV3_1	QUPV3_1_SE1	0x00A84000	354	GCC_QUPV3_WRAP1_S1_CLK
QUPV3_1	QUPV3_1_SE2	0x00A88000	355	GCC_QUPV3_WRAP1_S2_CLK
QUPV3_1	QUPV3_1_SE3	0x00A8C000	356	GCC_QUPV3_WRAP1_S3_CLK
SSC_QUPV3	SSC_QUPV3_SE0	0 x 62680000	442	SCC_QUPV3_SE0_CLK
SSC_QUPV3	SSC_QUPV3_SE1	0 x 62684000	443	SCC_QUPV3_SE1_CLK
SSC_QUPV3	SSC_QUPV3_SE2	0 x 62688000	444	SCC_QUPV3_SE2_CLK
SSC_QUPV3	SSC_QUPV3_SE3	0 x 6268C000	445	SCC_QUPV3_SE3_CLK
SSC_QUPV3	SSC_QUPV3_SE4	0 x 62690000	446	SCC_QUPV3_SE4_CLK
SSC_QUPV3	SSC_QUPV3_SE5	0 x 62694000	447	SCC_QUPV3_SE5_CLK

7 I²C – Software configuation

7.1 Configure QUPv3 core as I²C in the kernel

The steps required to configure and use any of the SE cores that are available in the SA6155 chipset as an I²C device are described in this section.

For more information, see the following:

- Source code – /kernel/drivers/i2c/busses/i2c-qcom-geni.c
- Config file – /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi
- Pin control file – /kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi
- SSC Pin control file - /kernel/arch/arm64/boot/dts/qcom/sm6150-slpi-pinctrl.dtsi
- TrustZone changes - trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

7.2 Code modification to configure the SE as I²C

To configure QUPV3_X_SE as I²C for the SA6155 chipset

Modify: /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi

1. Add a device tree node. The following are few nodes that are already present. If not present for any QUP, add the nodes and change status to “ok”.

```
/* QUPv3 South Instances */
/* I2C */
qupv3_se1_i2c: i2c@884000 {
     compatible = "qcom,i2c-geni";
     reg = <0x884000 0x4000>;
     interrupts = <GIC_SPI 602 0>;
     #address-cells = <1>;
     #size-cells = <0>;
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP0_S1_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
     dmas = <&gpi_dma0 0 1 3 64 0>,
          <&gpi_dma0 1 1 3 64 0>;
     dma-names = "tx", "rx";
     pinctrl-names = "default", "sleep";
```

```
        pinctrl-0 = <&qupv3_se1_i2c_active>;
        pinctrl-1 = <&qupv3_se1_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_0>;
        status = "disabled";
    };

    qupv3_se2_i2c: i2c@888000 {
        compatible = "qcom,i2c-geni";
        reg = <0x888000 0x4000>;
        interrupts = <GIC_SPI 603 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP0_S2_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
        dmas = <&gpi_dma0 0 2 3 64 0>,
            <&gpi_dma0 1 2 3 64 0>;
        dma-names = "tx", "rx";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se2_i2c_active>;
        pinctrl-1 = <&qupv3_se2_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_0>;
        status = "disabled";
    };

    qupv3_se3_i2c: i2c@88c000 {
        compatible = "qcom,i2c-geni";
        reg = <0x88c000 0x4000>;
        interrupts = <GIC_SPI 604 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP0_S3_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
        dmas = <&gpi_dma0 0 3 3 64 0>,
            <&gpi_dma0 1 3 3 64 0>;
        dma-names = "tx", "rx";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se3_i2c_active>;
        pinctrl-1 = <&qupv3_se3_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_0>;
        status = "disabled";
        };

        /* QUPv3 North instances */
        /* I2C */
```

```
qupv3_se4_i2c: i2c@a80000 {
     compatible = "qcom,i2c-geni";
     reg = <0xa80000 0x4000>;
     interrupts = <GIC_SPI 353 0>;
     #address-cells = <1>;
     #size-cells = <0>;
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP1_S0_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
     dmas = <&gpi_dma1 0 0 3 64 0>,
          <&gpi_dma1 1 0 3 64 0>;
     dma-names = "tx", "rx";
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se4_i2c_active>;
     pinctrl-1 = <&qupv3_se4_i2c_sleep>;
     qcom,wrapper-core = <&qupv3_1>;
     status = "disabled";
};

qupv3_se5_i2c: i2c@a84000 {
     compatible = "qcom,i2c-geni";
     reg = <0xa84000 0x4000>;
     interrupts = <GIC_SPI 354 0>;
     #address-cells = <1>;
     #size-cells = <0>;
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP1_S1_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
     dmas = <&gpi_dma1 0 1 3 64 0>,
          <&gpi_dma1 1 1 3 64 0>;
     dma-names = "tx", "rx";
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se5_i2c_active>;
     pinctrl-1 = <&qupv3_se5_i2c_sleep>;
     qcom,wrapper-core = <&qupv3_1>;
     status = "disabled";
};

qupv3_se6_i2c: i2c@a88000 {
     compatible = "qcom,i2c-geni";
     reg = <0xa88000 0x4000>;
     interrupts = <GIC_SPI 355 0>;
     #address-cells = <1>;
     #size-cells = <0>;
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP1_S2_CLK>,
```

```
            <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        dmas = <&gpi_dma1 0 2 3 64 0>,
            <&gpi_dma1 1 2 3 64 0>;
        dma-names = "tx", "rx";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se6_i2c_active>;
        pinctrl-1 = <&qupv3_se6_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_1>;
        status = "disabled";
};

qupv3_se7_i2c: i2c@a8c000 {
        compatible = "qcom,i2c-geni";
        reg = <0xa8c000 0x4000>;
        interrupts = <GIC_SPI 356 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP1_S3_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
            <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        dmas = <&gpi_dma1 0 3 3 64 0>,
            <&gpi_dma1 1 3 3 64 0>;
        dma-names = "tx", "rx";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se7_i2c_active>;
        pinctrl-1 = <&qupv3_se7_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_1>;
        status = "disabled";
};
```

Steps to configure SSC_QUPs

```
/* SSC_QUPs I2C */
qupv3_se8_i2c: i2c@62680000 {
        compatible = "qcom,i2c-geni";
        reg = <0x62680000 0x4000>;
        interrupts = <GIC_SPI 442 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_scc SCC_QUPV3_SE0_CLK>,
            <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
            <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se8_i2c_active>;
        pinctrl-1 = <&qupv3_se8_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_2>;
```

```
        status = "disabled";
    };

    qupv3_se9_i2c: i2c@62684000 {
        compatible = "qcom,i2c-geni";
        reg = <0x62684000 0x4000>;
        interrupts = <GIC_SPI 443 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_scc SCC_QUPV3_SE1_CLK>,
            <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
            <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se9_i2c_active>;
        pinctrl-1 = <&qupv3_se9_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_2>;
        status = "disabled";
    };

    qupv3_se10_i2c: i2c@62688000 {
        compatible = "qcom,i2c-geni";
        reg = <0x62688000 0x4000>;
        interrupts = <GIC_SPI 444 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_scc SCC_QUPV3_SE2_CLK>,
            <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
            <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se10_i2c_active>;
        pinctrl-1 = <&qupv3_se10_i2c_sleep>;
        qcom,wrapper-core = <&qupv3_2>;
        status = "disabled";
    };

    qupv3_se11_i2c: i2c@6268C000 {
        compatible = "qcom,i2c-geni";
        reg = <0x6268C000 0x4000>;
        interrupts = <GIC_SPI 445 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_scc SCC_QUPV3_SE3_CLK>,
            <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
            <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
        pinctrl-names = "default", "sleep";
```

```
            pinctrl-0 = <&qupv3_se11_i2c_active>;
            pinctrl-1 = <&qupv3_se11_i2c_sleep>;
            qcom,wrapper-core = <&qupv3_2>;
            status = "disabled";
        };

        qupv3_se12_i2c: i2c@62690000 {
            compatible = "qcom,i2c-geni";
            reg = <0x62690000 0x4000>;
            interrupts = <GIC_SPI 446 0>;
            #address-cells = <1>;
            #size-cells = <0>;
            clock-names = "se-clk", "m-ahb", "s-ahb";
            clocks = <&clock_scc SCC_QUPV3_SE4_CLK>,
                <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
                <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
            pinctrl-names = "default", "sleep";
            pinctrl-0 = <&qupv3_se12_i2c_active>;
            pinctrl-1 = <&qupv3_se12_i2c_sleep>;
            qcom,wrapper-core = <&qupv3_2>;
            status = "disabled";
        };

        qupv3_se13_i2c: i2c@62694000 {
            compatible = "qcom,i2c-geni";
            reg = <0x62694000 0x4000>;
            interrupts = <GIC_SPI 447 0>;
            #address-cells = <1>;
            #size-cells = <0>;
            clock-names = "se-clk", "m-ahb", "s-ahb";
            clocks = <&clock_scc SCC_QUPV3_SE5_CLK>,
                <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
                <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
            pinctrl-names = "default", "sleep";
            pinctrl-0 = <&qupv3_se13_i2c_active>;
            pinctrl-1 = <&qupv3_se13_i2c_sleep>;
            qcom,wrapper-core = <&qupv3_2>;
            status = "disabled";
        };
```

2. Set up GPIO

   Modify: /kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi

```
/* QUPv3_0 South SE mappings */
        /* SE 1 pin mappings */
        qupv3_se1_i2c_pins: qupv3_se1_i2c_pins {
            qupv3_se1_i2c_active: qupv3_se1_i2c_active {
```

```
                mux {
                        pins = "gpio4", "gpio5";
                        function = "qup01";
                };

                config {
                        pins = "gpio4", "gpio5";
                        drive-strength = <2>;
                        bias-disable;
                };
        };

        qupv3_se1_i2c_sleep: qupv3_se1_i2c_sleep {
                mux {
                        pins = "gpio4", "gpio5";
                        function = "gpio";
                };

                config {
                        pins = "gpio4", "gpio5";
                        drive-strength = <2>;
                        bias-no-pull;
                };
        };
};

/* SE 2 pin mappings */
qupv3_se2_i2c_pins: qupv3_se2_i2c_pins {
        qupv3_se2_i2c_active: qupv3_se2_i2c_active {
                mux {
                        pins = "gpio0", "gpio1";
                        function = "qup02";
                };

                config {
                        pins = "gpio0", "gpio1";
                        drive-strength = <2>;
                        bias-disable;
                };
        };

        qupv3_se2_i2c_sleep: qupv3_se2_i2c_sleep {
                mux {
                        pins = "gpio0", "gpio1";
                        function = "gpio";
                };

                config {
```

```
                pins = "gpio0", "gpio1";
                drive-strength = <2>;
                bias-pull-up;
            };
        };
    };
    /* SE 3 pin mappings */
    qupv3_se3_i2c_pins: qupv3_se3_i2c_pins {
        qupv3_se3_i2c_active: qupv3_se3_i2c_active {
            mux {
                pins = "gpio18", "gpio19";
                function = "qup03";
            };

            config {
                pins = "gpio18", "gpio19";
                drive-strength = <2>;
                bias-disable;
            };
        };

        qupv3_se3_i2c_sleep: qupv3_se3_i2c_sleep {
            mux {
                pins = "gpio18", "gpio19";
                function = "gpio";
            };

            config {
                pins = "gpio18", "gpio19";
                drive-strength = <2>;
                bias-pull-up;
            };
        };
    };

    /* QUPv3_1 North instances */
    /* SE 4 pin mappings */
    qupv3_se4_i2c_pins: qupv3_se4_i2c_pins {
        qupv3_se4_i2c_active: qupv3_se4_i2c_active {
            mux {
                pins = "gpio20", "gpio21";
                function = "qup10";
            };

            config {
                pins = "gpio20", "gpio21";
                drive-strength = <2>;
                bias-disable;
```

```
                };
            };

            qupv3_se4_i2c_sleep: qupv3_se4_i2c_sleep {
                mux {
                    pins = "gpio20", "gpio21";
                    function = "gpio";
                };

                config {
                    pins = "gpio20", "gpio21";
                    drive-strength = <2>;
                    bias-pull-up;
                };
            };
        };
        /* SE 5 pin mappings */
        qupv3_se5_i2c_pins: qupv3_se5_i2c_pins {
            qupv3_se5_i2c_active: qupv3_se5_i2c_active {
                mux {
                    pins = "gpio14", "gpio15";
                    function = "qup11";
                };

                config {
                    pins = "gpio14", "gpio15";
                    drive-strength = <2>;
                    bias-disable;
                };
            };

            qupv3_se5_i2c_sleep: qupv3_se5_i2c_sleep {
                mux {
                    pins = "gpio14", "gpio15";
                    function = "gpio";
                };

                config {
                    pins = "gpio14", "gpio15";
                    drive-strength = <2>;
                    bias-pull-up;
                };
            };
        };
        /* SE 6 pin mappings */
        qupv3_se6_i2c_pins: qupv3_se6_i2c_pins {
            qupv3_se6_i2c_active: qupv3_se6_i2c_active {
                mux {
```

```
                pins = "gpio6", "gpio7";
                function = "qup12";
            };

            config {
                pins = "gpio6", "gpio7";
                drive-strength = <2>;
                bias-disable;
            };
        };

        qupv3_se6_i2c_sleep: qupv3_se6_i2c_sleep {
            mux {
                pins = "gpio6", "gpio7";
                function = "gpio";
            };

            config {
                pins = "gpio6", "gpio7";
                drive-strength = <2>;
                bias-pull-up;
            };
        };
    };
    /* SE 7 pin mappings */
    qupv3_se7_i2c_pins: qupv3_se7_i2c_pins {
        qupv3_se7_i2c_active: qupv3_se7_i2c_active {
            mux {
                pins = "gpio10", "gpio11";
                function = "qup13";
            };

            config {
                pins = "gpio10", "gpio11";
                drive-strength = <2>;
                bias-disable;
            };
        };

        qupv3_se7_i2c_sleep: qupv3_se7_i2c_sleep {
            mux {
                pins = "gpio10", "gpio11";
                function = "gpio";
            };

            config {
                pins = "gpio10", "gpio11";
                drive-strength = <2>;
```

```
                    bias-pull-up;
                };
            };
        };
```

Modify: /kernel/arch/arm64/boot/dts/qcom/sm6150-slpi-pinctrl.dtsi

```
/*SSC_QUP_SE0 pin mapping */
qupv3_se8_i2c_pins: qupv3_se8_i2c_pins {
            qupv3_se8_i2c_active: qupv3_se8_i2c_active {
                mux {
                    pins = "gpio0", "gpio1";
                    function = "func1";
                };

                config {
                    pins = "gpio0", "gpio1";
                    drive-strength = <2>;
                    bias-disable;
                };
            };

            qupv3_se8_i2c_sleep: qupv3_se8_i2c_sleep {
                mux {
                    pins = "gpio0", "gpio1";
                    function = "gpio";
                };

                config {
                    pins = "gpio0", "gpio1";
                    drive-strength = <2>;
                    bias-pull-up;
                };
            };
        };

/*SSC_QUP_SE1 pin mapping */
qupv3_se9_i2c_pins: qupv3_se9_i2c_pins {
            qupv3_se9_i2c_active: qupv3_se9_i2c_active {
                mux {
                    pins = "gpio2", "gpio3";
                    function = "func1";
                };

                config {
                    pins = "gpio2", "gpio3";
```

```
                    drive-strength = <2>;
                    bias-disable;
                };
            };

            qupv3_se9_i2c_sleep: qupv3_se9_i2c_sleep {
                mux {
                    pins = "gpio2", "gpio3";
                    function = "gpio";
                };

                config {
                    pins = "gpio2", "gpio3";
                    drive-strength = <2>;
                    bias-pull-up;
                };
            };
        };

/*SSC_QUP_SE2 pin mapping */
qupv3_se10_i2c_pins: qupv3_se10_i2c_pins {
            qupv3_se10_i2c_active: qupv3_se10_i2c_active {
                mux {
                    pins = "gpio8", "gpio9";
                    function = "func1";
                };

                config {
                    pins = "gpio8", "gpio9";
                    drive-strength = <2>;
                    bias-disable;
                };
            };

            qupv3_se10_i2c_sleep: qupv3_se10_i2c_sleep {
                mux {
                    pins = "gpio8", "gpio9";
                    function = "gpio";
                };

                config {
                    pins = "gpio8", "gpio9";
                    drive-strength = <2>;
                    bias-pull-up;
                };
            };
        };
```

```
/*SSC_QUP_SE3 pin mapping */
qupv3_se11_i2c_pins: qupv3_se11_i2c_pins {
              qupv3_se11_i2c_active: qupv3_se11_i2c_active {
                   mux {
                        pins = "gpio16", "gpio17";
                        function = "func3";
                   };

                   config {
                        pins = "gpio16", "gpio17";
                        drive-strength = <2>;
                        bias-disable;
                   };
              };

              qupv3_se11_i2c_sleep: qupv3_se11_i2c_sleep {
                   mux {
                        pins = "gpio16", "gpio17";
                        function = "gpio";
                   };

                   config {
                        pins = "gpio16", "gpio17";
                        drive-strength = <2>;
                        bias-pull-up;
                   };
              };
         };

/*SSC_QUP_SE4 pin mapping */
qupv3_se12_i2c_pins: qupv3_se12_i2c_pins {
              qupv3_se12_i2c_active: qupv3_se12_i2c_active {
                   mux {
                        pins = "gpio16", "gpio17";
                        function = "func2";
                   };

                   config {
                        pins = "gpio16", "gpio17";
                        drive-strength = <2>;
                        bias-disable;
                   };
              };

              qupv3_se12_i2c_sleep: qupv3_se12_i2c_sleep {
                   mux {
                        pins = "gpio16", "gpio17";
                        function = "gpio";
```

```
                    };

                    config {
                         pins = "gpio16", "gpio17";
                         drive-strength = <2>;
                         bias-pull-up;
                    };
               };
          };

     /*SSC_QUP_SE5 pin mapping */
     qupv3_se13_i2c_pins: qupv3_se13_i2c_pins {
               qupv3_se13_i2c_active: qupv3_se13_i2c_active {
                    mux {
                         pins = "gpio14", "gpio15";
                         function = "func2";
                    };

                    config {
                         pins = "gpio14", "gpio15";
                         drive-strength = <2>;
                         bias-disable;
                    };
               };

               qupv3_se13_i2c_sleep: qupv3_se13_i2c_sleep {
                    mux {
                         pins = "gpio14", "gpio15";
                         function = "gpio";
                    };

                    config {
                         pins = "gpio14", "gpio15";
                         drive-strength = <2>;
                         bias-pull-up;
                    };
               };
          };
```

3. Register the GPIOs in the Project_Root/kernel/drivers/i2c/busses/i2c-qcom-geni.c and /kernel/drivers/pinctrl/core.c files.

```
struct pinctrl_state *pinctrl_lookup_state(struct pinctrl *p, const char
*name)
{
     struct pinctrl_state *state;

     state = find_state(p, name);
     if (!state) {
          if (pinctrl_dummy_state) {
               /* create dummy state */
               dev_dbg(p->dev, "using pinctrl dummy state (%s)\n",
                    name);
               state = create_state(p, name);
          } else
               state = ERR_PTR(-ENODEV);
     }

     return state;
}
```

4. Change TrustZone

Modify:
trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

NOTE: The folder contains all the structures defined in the QUPAC_Access.c file.

To update I²C mode, change the structure (`qupv3_perms_auto`) as per the following example code:

```
{QUPV3_1_SE1, QUPV3_PROTOCOL_I2C, QUPV3_MODE_FIFO, AC_HLOS, TRUE, TRUE,
FALSE}
```

In case of SSC_QUPs, Change the structure (`ssc_qupv3_perms_auto`) as per the following code:

```
{ QUPV3_SSC_SE0, QUPV3_PROTOCOL_I2C, QUPV3_MODE_FIFO, AC_HLOS, TRUE,
TRUE, FALSE }
```

In the preceding example code:

- QUPV3_MODE_FIFO is a FIFO mode that can be modified to GSI mode by replacing FIFO with GSI, for example, QUPV3_MODE_GSI
- AC_HLOS is the access permission that can be modified to any sub system as per requirement, for example, AC_TZ, AC_ADSP_Q6_ELF
- SE protocol can have one of the following:
  - SPI
  - I²C
  - UART

7.3 Verify I²C bus

Ensure that the bus is registered. The I²C bus is registered under /dev/i2c-, where the cell-index matches the bus number.

```
adb shell --> Get adb shell
cd /dev/
ls i2c* --> to List all the I2C buses
root@android:/dev # ls i2c*
ls i2c*
i2c-0
i2c-10
i2c-2
```

If the bus is correctly registered, test it using a test program. The test code is found at /vendor/qcom/proprietary/kernel-tests-internal/i2c-msm/i2c-msm-test.c

```
#adb shell
#cd /data/kernel-tests/
#chmod 777 *
#./i2c-msm-test -D /dev/i2c-2 -l -v --> To scan the I2C bus and get
response 1 for slave address present on I2C bus.
./i2c-msm-test -v -D /dev/i2c-0 -s 0x50 -b 5  -n 50 -o 0  -i 1 --> To run
the I2C test
./i2c-msm-test –help --> To get the help on I2C test app parameter.
```

7.4 Debug tips

If a read/write and the I²C fails, perform the following:

1. Check SDA/SCL idling.
2. Verify the scope of the BUS ID to ensure that SDA/SCL is idle at the high logic level. If it is not idle at high, there is either a hardware configuration problem or the GPIO settings are invalid.

7.5 Configure I²C slave device

Register a slave device using the device tree

Once the I²C bus is verified, create a slave device driver and register it with the I²C bus. See the following files as examples:

- For an I²C slave device – /kernel/ arch/arm/boot/dts/qcom/ sm6150-qrd.dtsi
- For synaptic touchscreen driver registration – /drivers/input/touchscreen/hxchipset/himax_platform.c

Create a device tree node

Modify: /kernel/arch/arm64/boot/dts/qcom/sm6150-qrd.dtsi or sm6150-mtp.dtsi

Add a device tree node:

```
&qupv3_se1_i2c {
status = "okay";
himax_ts@48 {
        compatible = "himax,hxcommon";
        reg = <0x48>;
        interrupt-parent = <&tlmm>;
        interrupts = <125 0x2008>;
        vdd-supply = <&pm6150_l10>;
        avdd-supply = <&pm6150l_l7>;
        pinctrl-names = "pmx_ts_active","pmx_ts_suspend",
                                "pmx_ts_release";
        pinctrl-0 = <&ts_int_active &ts_reset_active>;
        pinctrl-1 = <&ts_int_suspend &ts_reset_suspend>;
        pinctrl-2 = <&ts_release>;
        himax,panel-coords = <0 1080 0 2160>;
        himax,display-coords = <0 1080 0 2160>;
        himax,irq-gpio = <&tlmm 89 0x00>;
        himax,rst-gpio = <&tlmm 88 0x00>;
        report_type = <1>;
        };
};
```

Modify code

```
#include <linux/module.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <linux/i2c.h>
#include <linux/interrupt.h>
#include <linux/slab.h>
#include <linux/gpio.h>
#include <linux/debugfs.h>
#include <linux/seq_file.h>
#include <linux/regulator/consumer.h>
#include <linux/string.h>
#include <linux/of_gpio.h>

#ifdef CONFIG_OF //Open firmware must be defined for dts usage
static struct of_device_id qcom_i2c_test_table[] = {
   { .compatible = "qcom,i2c-test",}, //Compatible node must match dts
   { },
};
#else
#define qcom_i2c_test_table NULL
#endif

//I2C slave id supported by driver
```

```
static const struct i2c_device_id qcom_id[] = {
   { "qcom_i2c_test", 0 },
   { }
};

static int i2c_test_test_transfer(struct i2c_client *client)
{
  struct i2c_msg xfer; //I2C transfer structure
  u8 buf = 0x55;  //data to transfer
  xfer.addr = client->addr;
  xfer.flags = 0;
  xfer.len = 1;
  xfer.buf = &buf;

  return i2c_transfer(client->adapter, &xfer, 1);
}

static int __devinit i2c_test_probe(struct i2c_client *client,
        const struct i2c_device_id *id)
{
  int irq_gpio = -1;
  int irq;
  int addr;
  //Parse data using dt.
  if(client->dev.of_node){
    irq_gpio = of_get_named_gpio_flags(client->dev.of_node,
"qcom_i2c_test,irq-gpio", 0, NULL);
  }
  irq = client->irq; //GPIO irq #. already converted to gpio_to_irq
  addr = client->addr; //Slave Address
  dev_err(&client->dev, "gpio [%d] irq [%d] gpio_irq [%d] Slaveaddr [%x]
\n", irq_gpio, irq,
    gpio_to_irq(irq_gpio), addr);

  //You can initiate an I2C transfer anytime using i2c_client *client
structure
  i2c_test_test_transfer(client);

  return 0;
}

//I2C Driver Information
static struct i2c_driver i2c_test_driver = {
   .driver = {
        .name     = "qcom_i2c_test",
        .owner    = THIS_MODULE,
        .of_match_table = qcom_i2c_test_table,
   },
```

```
    .probe        = i2c_test_probe,
    .id_table       = qcom_id,
};

//Easy wrapper to do driver initialization
module_i2c_driver(i2c_test_driver);

MODULE_DESCRIPTION("I2C TEST");
MODULE_LICENSE("GPL v2");
```

8 UART – Hardware overview

8.1 UART core

- Two wire without flow control and four wire with flow control, parity support, single EE operation.
- 75 bps to 4 Mbps, configurable character size and stop bit, half and full Duplex mode.

8.2 Base addresses

Table 8-1 UART physical address

QUPv3 hardware ID	QUP core	Physical address	IRQ #	Kernel UART clock name
QUPV3_0	QUPV3_0_SE0	0x00880000	601	GCC_QUPV3_WRAP0_S0_CLK
QUPV3_0	QUPV3_0_SE1	0x00884000	602	GCC_QUPV3_WRAP0_S1_CLK
QUPV3_0	QUPV3_0_SE2	0x00888000	603	GCC_QUPV3_WRAP0_S2_CLK
QUPV3_0	QUPV3_0_SE3	0x0088C000	604	GCC_QUPV3_WRAP0_S3_CLK
QUPV3_1	QUPV3_1_SE0	0x00A80000	353	GCC_QUPV3_WRAP1_S0_CLK
QUPV3_1	QUPV3_1_SE1	0x00A84000	354	GCC_QUPV3_WRAP1_S1_CLK
QUPV3_1	QUPV3_1_SE2	0x00A88000	355	GCC_QUPV3_WRAP1_S2_CLK
QUPV3_1	QUPV3_1_SE3	0x00A8C000	356	GCC_QUPV3_WRAP1_S3_CLK
SSC_QUPV3	SSC_QUPV3_SE0	0 x 62680000	442	SCC_QUPV3_SE0_CLK
SSC_QUPV3	SSC_QUPV3_SE1	0 x 62684000	443	SCC_QUPV3_SE1_CLK
SSC_QUPV3	SSC_QUPV3_SE2	0 x 62688000	444	SCC_QUPV3_SE2_CLK
SSC_QUPV3	SSC_QUPV3_SE3	0 x 6268C000	445	SCC_QUPV3_SE3_CLK
SSC_QUPV3	SSC_QUPV3_SE4	0 x 62690000	446	SCC_QUPV3_SE4_CLK
SSC_QUPV3	SSC_QUPV3_SE5	0 x 62694000	447	SCC_QUPV3_SE5_CLK

9 UART – Software configuration

9.1 Configure UART in the boot loader

By default, QTI has configured QUPv3 SE in the boot loader. This section describes the changes required to configure different UART for debugging purposes.

9.1.1 Set the base address

1. Modify the files at BOOT.XF.3.1/boot_images/QcomPkg/SDMPkg/6150/Settings/UART/UartSettings.c
2. Set the correct base address:

```
UART_PROPERTIES devices =
{
   // MAIN_PORT
   0x880000,     // uart_base
   0x8C0000,     // qup_common_base
   0x2001c101,  // gpio_tx_config gpio: 16 functionality : 1
   0x20008111,  // gpio_rx_config gpio: 17 functionality : 1
   0,            // gpio_cts_config
   0,            // gpio_rfr_config
   0,            // clock_id_index
   (void*)0,         // bus_clock_id
   (void*)CLK_QUPV3_WRAP0_S0,      // core_clock_id
   32,           // dummy irq number not really used
   0,   //TCSR base
   0,   // TCSR offset
   0    // TCSR value

};

FW download
uint8 __attribute__ ((aligned (4))) qup_v1_2_uart_fw[] =
{
    0x53, 0x45, 0x46, 0x57, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x20, 0x02, 0x00, 0x01, 0x00,
    0x07, 0x00, 0x6d, 0x01, 0x1c, 0x00, 0x42, 0x00, 0xd8, 0x06, 0xd0,
0x05, 0x32, 0x00, 0x00, 0x00,
    0x22, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x28, 0x0f, 0x00,
```

```
    0x20, 0x00, 0x78, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0x00, 0xc8, 0xc6, 0x7f,
0x00, 0x00, 0x00, 0x3c, 0x00,
    0xc8, 0xc6, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x48, 0x40, 0x00,
    0x10, 0x00, 0x7d, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x80, 0xb2, 0x00,
    0x08, 0x00, 0xb5, 0x00, 0x21, 0x38, 0x7f, 0x00, 0xa1, 0x0b, 0x7f,
0x00, 0x82, 0x38, 0x7f, 0x00,
    0x49, 0xd0, 0x9f, 0x00, 0x0d, 0x2e, 0x7f, 0x00, 0x63, 0x78, 0x9f,
0x00, 0x4f, 0xd0, 0x9f, 0x00,
    0x15, 0x2e, 0x7f, 0x00, 0x63, 0x78, 0x9f, 0x00, 0x57, 0xd0, 0x9f,
0x00, 0x45, 0x10, 0x99, 0x00,
    0x1d, 0x2e, 0x7f, 0x00, 0x63, 0x78, 0x9f, 0x00, 0x5f, 0x50, 0x9f,
0x00, 0x4b, 0x10, 0x99, 0x00,
    0x25, 0x2e, 0x7f, 0x00, 0x63, 0x78, 0x9f, 0x00, 0x53, 0x10, 0x99,
0x00, 0x2d, 0x2e, 0x7f, 0x00,
    0x6d, 0x10, 0x99, 0x00, 0x7f, 0x48, 0x9f, 0x00, 0xdb, 0x00, 0x9f,
0x00, 0x67, 0x18, 0x9f, 0x00,
    0x7f, 0x78, 0x9f, 0x00, 0xdb, 0x00, 0x9f, 0x00, 0x6d, 0x60, 0x9f,
0x00, 0x03, 0x3a, 0x7f, 0x00,
    0x79, 0x48, 0x9f, 0x00, 0x6c, 0x18, 0x9f, 0x00, 0x7f, 0x78, 0x9f,
0x00, 0xdb, 0x00, 0x9f, 0x00,
    0x7f, 0x18, 0x9f, 0x00, 0x79, 0x68, 0x9f, 0x00, 0xa1, 0x0b, 0x7f,
0x00, 0xdb, 0x00, 0x9f, 0x00,
    0x83, 0x60, 0x9f, 0x00, 0x28, 0x34, 0x7f, 0x00, 0x00, 0x08, 0x1f,
0x00, 0x04, 0x34, 0x7f, 0x00,
    0x07, 0x39, 0x7f, 0x00, 0x20, 0x38, 0x7f, 0x00, 0x8f, 0x60, 0x9f,
0x00, 0x00, 0x80, 0x04, 0x00,
    0x07, 0x39, 0x7f, 0x00, 0x8f, 0x50, 0x9f, 0x00, 0xe1, 0x08, 0x9f,
0x00, 0x99, 0x60, 0x9f, 0x00,
    0x08, 0x34, 0x7f, 0x00, 0xe9, 0x10, 0x99, 0x00, 0x08, 0x39, 0x7f,
0x00, 0x80, 0x1c, 0x7f, 0x00,
    0xab, 0x70, 0x9f, 0x00, 0x02, 0x1c, 0x7f, 0x00, 0xcd, 0x58, 0x9f,
0x00, 0xb9, 0x78, 0x9f, 0x00,
    0xcc, 0x30, 0x9f, 0x00, 0xb8, 0x10, 0x9f, 0x00, 0xcd, 0x78, 0x9f,
0x00, 0xdb, 0x00, 0x9f, 0x00,
    0xc5, 0x18, 0x9f, 0x00, 0xb1, 0x68, 0x9f, 0x00, 0xc5, 0x78, 0x9f,
0x00, 0xb9, 0x60, 0x9f, 0x00,
    0xc4, 0x10, 0x99, 0x00, 0x03, 0x3a, 0x7f, 0x00, 0xdb, 0x00, 0x9f,
0x00, 0xb1, 0x48, 0x9f, 0x00,
    0xbe, 0x18, 0x9f, 0x00, 0x20, 0x1c, 0x7f, 0x00, 0x28, 0x34, 0x7f,
0x00, 0x00, 0x00, 0x1f, 0x00,
    0x00, 0x40, 0x1f, 0x00, 0xdb, 0x00, 0x9f, 0x00, 0xd4, 0x60, 0x9f,
0x00, 0xcc, 0x10, 0x9f, 0x00,
```

```
    0x63, 0x78, 0x9f, 0x00, 0xd0, 0x18, 0x9f, 0x00, 0x28, 0x34, 0x7f,
0x00, 0xd1, 0x78, 0x9f, 0x00,
    0x01, 0x60, 0x7f, 0x00, 0xdd, 0x60, 0x9f, 0x00, 0xc7, 0x78, 0x9f,
0x00, 0xe1, 0x60, 0x9f, 0x00,
    0x00, 0x00, 0x05, 0x00, 0x07, 0x39, 0x7f, 0x00, 0x99, 0x78, 0x9f,
0x00, 0x07, 0x39, 0x7f, 0x00,
    0xeb, 0x60, 0x9f, 0x00, 0x18, 0x34, 0x7f, 0x00, 0x9f, 0x78, 0x9f,
0x00, 0x00, 0x00, 0x0f, 0x00,
    0xe1, 0x0b, 0x7f, 0x00, 0xf9, 0x00, 0x9f, 0x00, 0x00, 0x40, 0x1f,
0x00, 0x01, 0x60, 0x7f, 0x00,
    0xf7, 0x78, 0x9f, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x00, 0x40, 0x1f,
0x00, 0x01, 0x40, 0x7f, 0x00,
    0x75, 0x7a, 0x9d, 0x00, 0x5d, 0x7a, 0x9d, 0x00, 0x0b, 0x89, 0x98,
0x00, 0x00, 0x00, 0x0e, 0x00,
    0x01, 0x50, 0x7f, 0x00, 0x91, 0x79, 0x9d, 0x00, 0xc7, 0x8a, 0x98,
0x00, 0x43, 0x4a, 0x9f, 0x00,
    0x00, 0x22, 0x6a, 0x00, 0x01, 0x44, 0x6b, 0x00, 0x57, 0x32, 0x9f,
0x00, 0x1d, 0x39, 0x9f, 0x00,
    0x80, 0x1c, 0x7f, 0x00, 0x1d, 0x01, 0x9f, 0x00, 0x07, 0x39, 0x68,
0x00, 0x20, 0x38, 0x7f, 0x00,
    0x82, 0xb8, 0x7f, 0x00, 0x3b, 0x51, 0x9f, 0x00, 0xab, 0x41, 0x9f,
0x00, 0x05, 0x39, 0x7f, 0x00,
    0xd5, 0x11, 0x9f, 0x00, 0x2d, 0x61, 0x9f, 0x00, 0x01, 0x50, 0x7f,
0x00, 0x47, 0x49, 0x8b, 0x00,
    0x07, 0x39, 0x7f, 0x00, 0x08, 0x20, 0x7f, 0x00, 0x02, 0x24, 0x7f,
0x00, 0x05, 0x7a, 0x1f, 0x01,
    0x3f, 0x89, 0x98, 0x00, 0x4d, 0x7a, 0x9d, 0x00, 0x3f, 0x61, 0x9f,
0x00, 0x07, 0xb9, 0x68, 0x00,
    0x09, 0x20, 0x7f, 0x00, 0x25, 0x79, 0x9f, 0x00, 0x7f, 0x8a, 0x98,
0x00, 0x51, 0x49, 0x9f, 0x00,
    0x13, 0x91, 0x98, 0x00, 0x6b, 0x02, 0x9c, 0x00, 0x13, 0x79, 0x9f,
0x00, 0x01, 0x50, 0x7f, 0x00,
    0x57, 0x91, 0x98, 0x00, 0x6b, 0x02, 0x9c, 0x00, 0x01, 0x40, 0x7f,
0x00, 0x5f, 0x59, 0x9f, 0x00,
    0x12, 0x49, 0x9f, 0x00, 0x0d, 0x79, 0x9f, 0x00, 0x0a, 0x04, 0x7f,
0x00, 0x12, 0xc9, 0x9f, 0x00,
    0x82, 0x99, 0x9f, 0x00, 0x69, 0x89, 0x98, 0x00, 0xa1, 0xf9, 0x9f,
0x00, 0x12, 0xc9, 0x9f, 0x00,
    0x69, 0xd1, 0x9f, 0x00, 0x12, 0x49, 0x9f, 0x00, 0x02, 0x1c, 0x7f,
0x00, 0x5f, 0x59, 0x9f, 0x00,
    0x04, 0x20, 0x7f, 0x00, 0x7a, 0x19, 0x9f, 0x00, 0x12, 0x49, 0x9f,
0x00, 0x5f, 0x79, 0x9f, 0x00,
    0x01, 0x1c, 0x7f, 0x00, 0x07, 0x20, 0x7f, 0x00, 0x12, 0x49, 0x9f,
0x00, 0x0d, 0x79, 0x9f, 0x00,
    0x00, 0xa1, 0x7f, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0xa2, 0x7f,
0x00, 0x64, 0x99, 0x9f, 0x00,
    0x04, 0xa0, 0x7f, 0x00, 0x12, 0xc9, 0x9f, 0x00, 0x0d, 0xf9, 0x9f,
0x00, 0x00, 0x21, 0x7f, 0x00,
    0x00, 0x00, 0x1f, 0x00, 0x00, 0x22, 0x7f, 0x00, 0x9c, 0x19, 0x9f,
0x00, 0x04, 0x20, 0x7f, 0x00,
```

```
    0x01, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x1f, 0x00, 0x01, 0x00, 0x1f,
0x00, 0xa6, 0x89, 0x9f, 0x00,
    0x12, 0xc9, 0x8e, 0x00, 0x69, 0xf9, 0x9f, 0x00, 0x12, 0xc9, 0x8d,
0x00, 0x69, 0xf9, 0x9f, 0x00,
    0xbf, 0x11, 0x9f, 0x00, 0xad, 0x61, 0x9f, 0x00, 0xb4, 0x21, 0x8b,
0x00, 0x06, 0x39, 0x7f, 0x00,
    0x2d, 0x79, 0x9f, 0x00, 0x06, 0x39, 0x7f, 0x00, 0x01, 0x20, 0x7f,
0x00, 0x08, 0x20, 0x7f, 0x00,
    0x01, 0x24, 0x7f, 0x00, 0x2d, 0x79, 0x9f, 0x00, 0xbf, 0x61, 0x9f,
0x00, 0xc8, 0x21, 0x8b, 0x00,
    0x06, 0x39, 0x7f, 0x00, 0x01, 0x34, 0x7f, 0x00, 0xd7, 0x79, 0x9f,
0x00, 0x06, 0x39, 0x7f, 0x00,
    0x01, 0x24, 0x7f, 0x00, 0x08, 0x20, 0x7f, 0x00, 0x01, 0x20, 0x7f,
0x00, 0x02, 0x34, 0x7f, 0x00,
    0xd7, 0x79, 0x9f, 0x00, 0x01, 0x34, 0x7f, 0x00, 0xd7, 0x61, 0x9f,
0x00, 0xe5, 0x49, 0x8b, 0x00,
    0x07, 0x39, 0x7f, 0x00, 0x02, 0x34, 0x7f, 0x00, 0x08, 0x20, 0x7f,
0x00, 0x02, 0x24, 0x7f, 0x00,
    0x05, 0x7a, 0x1f, 0x01, 0xa1, 0x8a, 0x98, 0x00, 0x01, 0x34, 0x7f,
0x00, 0x01, 0x34, 0x7f, 0x00,
    0x01, 0x50, 0x7f, 0x00, 0x49, 0x49, 0x9f, 0x00, 0xf3, 0x91, 0x98,
0x00, 0x6b, 0x02, 0x9c, 0x00,
    0x13, 0x79, 0x9f, 0x00, 0xf9, 0x89, 0x98, 0x00, 0x00, 0x00, 0x0d,
0x00, 0x16, 0x09, 0x7f, 0x00,
    0x69, 0x11, 0x7f, 0x00, 0x00, 0x1a, 0x9f, 0x00, 0xfd, 0x79, 0x9f,
0x00, 0x01, 0x60, 0x7f, 0x00,
    0x00, 0x40, 0x1f, 0x00, 0x00, 0x30, 0x7f, 0x00, 0x00, 0x28, 0x7f,
0x00, 0xc0, 0x68, 0x7f, 0x00,
    0x01, 0x20, 0x7f, 0x00, 0x2d, 0x2a, 0x9f, 0x00, 0x29, 0x12, 0x9f,
0x00, 0x11, 0x62, 0x9f, 0x00,
    0x07, 0x39, 0x6b, 0x00, 0x1f, 0x4a, 0x9f, 0x00, 0x01, 0x50, 0x7f,
0x00, 0x25, 0x2a, 0x9f, 0x00,
    0x1f, 0x4a, 0x9f, 0x00, 0x11, 0x7a, 0x9f, 0x00, 0x01, 0x50, 0x7f,
0x00, 0x5d, 0x7a, 0x9d, 0x00,
    0x49, 0x79, 0x9f, 0x00, 0x04, 0x24, 0x7f, 0x00, 0x31, 0x7a, 0x9f,
0x00, 0x01, 0x34, 0x7f, 0x00,
    0x11, 0x7a, 0x9f, 0x00, 0x04, 0x24, 0x7f, 0x00, 0x3f, 0x12, 0x9f,
0x00, 0x01, 0x40, 0x7f, 0x00,
    0x37, 0x4a, 0x9f, 0x00, 0x33, 0x7a, 0x9f, 0x00, 0x08, 0x24, 0x7f,
0x00, 0x01, 0x50, 0x7f, 0x00,
    0x5d, 0x7a, 0x9d, 0x00, 0x49, 0x79, 0x9f, 0x00, 0x02, 0x34, 0x7f,
0x00, 0x31, 0x7a, 0x9f, 0x00,
    0x48, 0x0a, 0x9f, 0x00, 0x0d, 0x49, 0x8e, 0x00, 0x13, 0x79, 0x9f,
0x00, 0x0d, 0x49, 0x8d, 0x00,
    0x13, 0x79, 0x9f, 0x00, 0x52, 0x0a, 0x9f, 0x00, 0x00, 0x00, 0x0e,
0x00, 0x01, 0x00, 0x1f, 0x00,
    0x00, 0x00, 0x0d, 0x00, 0x01, 0x00, 0x1f, 0x00, 0x00, 0x1d, 0x7f,
0x00, 0x0f, 0x20, 0x7f, 0x00,
    0x1b, 0x79, 0x9f, 0x00, 0x8b, 0x68, 0x7f, 0x00, 0x40, 0x68, 0x7f,
0x00, 0x67, 0x42, 0x9f, 0x00,
```

```
    0xc1, 0x68, 0x7f, 0x00, 0x01, 0x00, 0x1f, 0x00, 0xc2, 0x68, 0x7f,
0x00, 0x01, 0x00, 0x1f, 0x00,
    0x6f, 0x2a, 0x9f, 0x00, 0x01, 0x00, 0x1f, 0x00, 0x01, 0x20, 0x7f,
0x00, 0x10, 0x24, 0x7f, 0x00,
    0x01, 0x00, 0x1f, 0x00, 0x01, 0x2c, 0x7f, 0x00, 0x7b, 0x42, 0x9f,
0x00, 0x01, 0x00, 0x1f, 0x00,
    0x05, 0x2c, 0x7f, 0x00, 0x01, 0x00, 0x1f, 0x00, 0x05, 0x39, 0x7f,
0x00, 0x81, 0x62, 0x9f, 0x00,
    0x99, 0x0a, 0x9f, 0x00, 0x40, 0x36, 0x7f, 0x00, 0x40, 0x35, 0x7f,
0x00, 0x01, 0x39, 0x7f, 0x00,
    0x8f, 0x4a, 0x99, 0x00, 0x49, 0x49, 0x9f, 0x00, 0x8b, 0x62, 0x9f,
0x00, 0x95, 0x4a, 0x99, 0x00,
    0x49, 0x49, 0x9f, 0x00, 0x9b, 0x0a, 0x9f, 0x00, 0x91, 0x7a, 0x9f,
0x00, 0x49, 0x49, 0x9f, 0x00,
    0x80, 0x36, 0x7f, 0x00, 0x80, 0x35, 0x7f, 0x00, 0x49, 0x79, 0x9f,
0x00, 0x05, 0x39, 0x7f, 0x00,
    0x01, 0x34, 0x7f, 0x00, 0x01, 0x34, 0x7f, 0x00, 0xa7, 0x62, 0x9f,
0x00, 0xbf, 0x0a, 0x9f, 0x00,
    0x40, 0x36, 0x7f, 0x00, 0x40, 0x35, 0x7f, 0x00, 0x01, 0x39, 0x7f,
0x00, 0xb5, 0x4a, 0x99, 0x00,
    0xeb, 0x49, 0x9f, 0x00, 0xb1, 0x62, 0x9f, 0x00, 0xbb, 0x4a, 0x99,
0x00, 0xeb, 0x49, 0x9f, 0x00,
    0xc1, 0x0a, 0x9f, 0x00, 0xb7, 0x7a, 0x9f, 0x00, 0xeb, 0x49, 0x9f,
0x00, 0x80, 0x36, 0x7f, 0x00,
    0x80, 0x35, 0x7f, 0x00, 0xeb, 0x79, 0x9f, 0x00, 0x0d, 0x49, 0x9f,
0x00, 0x13, 0x79, 0x9f, 0x00,
    0xcf, 0x8a, 0x98, 0x00, 0x00, 0x00, 0x0d, 0x00, 0x57, 0x08, 0x7f,
0x00, 0x29, 0x10, 0x7f, 0x00,
    0xd7, 0x02, 0x9f, 0x00, 0x00, 0x40, 0x1f, 0x00, 0x01, 0x60, 0x7f,
0x00, 0xb9, 0x78, 0x9f, 0x00,
    0x24, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x24, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x9a, 0x01, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00, 0xe0,
0x00, 0x20, 0x00, 0x01, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x00,
0x00, 0x00, 0x07, 0x00, 0x00,
    0x00, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xc0, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x24, 0x00, 0xc0, 0x00, 0x00, 0x0b, 0x00,
0x00, 0x31, 0x0a, 0x02, 0x00,
    0x05, 0xce, 0x00, 0x00, 0xe7, 0x60, 0x03, 0x00, 0xa8, 0xe6, 0x41,
0x09, 0x10, 0x05, 0x10, 0x00,
    0x51, 0x1e, 0xc0, 0x42, 0x01, 0x04, 0x00, 0x00, 0x14, 0x84, 0x2e,
0x00, 0x1a, 0x58, 0x94, 0x16,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00,
0x00, 0x1c, 0x03, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
```

```
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x06, 0x1c, 0x08, 0x00,
    0x10, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0d, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x06, 0x1c, 0x08, 0x00, 0x10, 0x40, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x15, 0x24, 0xc0,
0x00, 0x0e, 0x00, 0x00, 0x00,
    0x01, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x02,
0x03, 0x04, 0x05, 0x06, 0x07,
    0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f, 0x10, 0x11, 0x12,
0x40, 0x41, 0x42, 0x43, 0x44,
    0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e, 0x4f,
0x50, 0x51, 0x52, 0x53, 0x54,
    0x55, 0x56, 0x57, 0x58, 0x59, 0x5a, 0x5b, 0x5c, 0x5d, 0x5e, 0x5f,
0x60, 0x61, 0x62, 0x63, 0x64,
    0x65, 0x66, 0x67, 0x68, 0x69, 0x6a, 0x6b, 0x6c, 0x6d, 0x6e
};
```

9.1.2 Configure the GPIO

1. Modify the files at

   /BOOT.XF.3.1/boot_images/QcomPkg/Library/UartQupv3LibUartXBLLoaderOs.c

2. Configure the correct GPIO:

```
Tlmm_ConfigGpioGroup(DAL_TLMM_GPIO_ENABLE,(DALGpioSignalType*)&h-
>properties->gpio_tx_config, 1);
Tlmm_ConfigGpioGroup(DAL_TLMM_GPIO_ENABLE,(DALGpioSignalType*)&h-
>properties->gpio_rx_config, 1);
```

9.1.3 Debug tips

If the UART is configured, the following message is displayed on the serial console:

uart_open

If you do not see a message, use the following debugging tips:

1. Set a breakpoint in the boot loader in the uart_initialize function line at register_init(); at /BOOT.XF.3.1/boot_images/QcomPkg/Library/UartQupv3Lib/UartXBL.c.
2. Check if the GPIOs are configured correctly.
3. Check the GPIO configuration register, GPIO_CFGn, to ensure that the GPIO settings are valid.

9.2 Configure console UART in the kernel

Low-speed UART is a FIFO-based UART driver designed for small data transfer at a slow rate, such as console debugging. If a large amount of data is transferred, or it is a high-speed transfer, QTI recommends using the high-speed UART driver.

9.2.1 Code modification

The following examples enable you to configure QUPv3 SE to use the low-speed UART driver on the SD chipsets.

- Device tree source – /kernel/arch/arm/boot/dts/qcom/sm6150-qupv3.dtsi
- GPIO table – kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi

NOTE: Currently, there is no explicit ID field to assign to the UART. Therefore, the first UART registered is ttyHSL0, the second one is ttyHSL1, and so on.

1. Modify the files at /kernel/arch/arm/boot/dts/qcom/sm6150-qupv3.dtsi
2. Add a new device tree node:

```
qupv3_se0_2uart: qcom,qup_uart@0x880000 {
     compatible = "qcom,msm-geni-console";
     reg = <0x880000 0x4000>;
     reg-names = "se_phys";
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP0_S0_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se0_2uart_active>;
     pinctrl-1 = <&qupv3_se0_2uart_sleep>;
     interrupts = <GIC_SPI 601 0>;
     qcom,wrapper-core = <&qupv3_0>;
     status = "disabled";
};
```

3. Set up the GPIO.

   Modify the files at kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi

```
/* QUPv3_0 South SE mappings */
      /* SE 0 pin mappings */
      qupv3_se0_2uart_pins: qupv3_se0_2uart_pins {
            qupv3_se0_2uart_active: qupv3_se0_2uart_active {
                  mux {
                        pins = "gpio16", "gpio17";
                        function = "qup00";
```

```
                        };

                        config {
                                pins = "gpio16", "gpio17";
                                drive-strength = <2>;
                                bias-disable;
                        };
                };
        qupv3_se0_2uart_sleep: qupv3_se0_2uart_sleep {
                        mux {
                                pins = "gpio16", "gpio17";
                                function = "gpio";
                        };

                        config {
                                pins = "gpio16", "gpio17";
                                drive-strength = <2>;
                                bias-disable;
                        };
                };
        };
```

Code modification

The preceding examples enable you to configure QUPv3 as a high-speed UART on the SA6155 chipset. To enable the driver, change the status to “ok”.

- Device tree source – /kernel/arch/arm/boot/dts/qcom/sm6150-qupv3.dtsi

```
qupv3_se7_4uart: qcom,qup_uart@0xa8c000 {
        compatible = "qcom,msm-geni-serial-hs";
        reg = <0xa8c000 0x4000>;
        reg-names = "se_phys";
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP1_S3_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se7_ctsrx>, <&qupv3_se7_rts>,
                        <&qupv3_se7_tx>;
        pinctrl-1 = <&qupv3_se7_ctsrx>, <&qupv3_se7_rts>,
                        <&qupv3_se7_tx>;
        interrupts-extended = <&pdc GIC_SPI 356 0>,
                 <&tlmm 13 0>;
        status = "disabled";
        qcom,wakeup-byte = <0xFD>;
```

```
            qcom,wrapper-core = <&qupv3_1>;
    };
```

- GPIO table – kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi

```
        qupv3_se7_4uart_pins: qupv3_se7_4uart_pins {
                qupv3_se7_ctsrx: qupv3_se7_ctsrx {
                        mux {
                                pins = "gpio10", "gpio13";
                                function = "qup13";
                        };

                        config {
                                pins = "gpio10", "gpio13";
                                drive-strength = <2>;
                                bias-no-pull;
                        };
                };

                qupv3_se7_rts: qupv3_se7_rts {
                        mux {
                                pins = "gpio11";
                                function = "qup13";
                        };

                        config {
                                pins = "gpio11";
                                drive-strength = <2>;
                                bias-pull-down;
                        };
                };

                qupv3_se7_tx: qupv3_se7_tx {
                        mux {
                                pins = "gpio12";
                                function = "qup13";
                        };

                        config {
                                pins = "gpio12";
```

```
                        drive-strength = <2>;
                        bias-pull-up;
                    };
                };
        };
```

- GPIO table for SSC_QUPs - kernel/arch/arm64/boot/dts/qcom/sm6150-slpi-pinctrl.dtsi

  Create active and sleep configuration as above for SSC_QUPs with respective GPIOs.

9.2.2 Debug tips

This section describes how to verify the high-speed UART.

1. Check registration and ensure that the UART is correctly registered with the TTY stack. Run the following commands:

   ```
   adb shell -> start a new shell
   ls /dev/ttyHS* -> Make sure UART is properly registered
   ```

   If the device is not visible, check the code modification to ensure that all the information is defined and accurate.

2. Check the internal loopback

   a. Run the following commands to enable loopback:

   ```
   adb shell
   mount -t debugfs none /sys/kernel/debug  -> mount debug fs
   # echo 3 > /sys/bus/platform/devices/880000.qcom,qup_uart/loopback
   ```

   b. Run the following command to perform UART kernel test.

   ```
   ./msm_uart_test -l -d /dev/ttyHS0 -b 115200 -s 400
   ```

3. Modify the files in TZ at:

   trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

NOTE: The folder contains all the structures defined in the QUPAC_Access.c file.

To update UART mode, change the structure (`qupv3_perms_auto`) as per the following example code:

```
{ QUPV3_0_SE2, QUPV3_PROTOCOL_UART_2W, QUPV3_MODE_FIFO,AC_HLOS,TRUE,
   TRUE,FALSE } → Two wire UART
{ QUPV3_0_SE6, QUPV3_PROTOCOL_UART_4W, QUPV3_MODE_FIFO,AC_HLOS,TRUE,
   TRUE,FALSE } → Four wire UART
```

In case of SSC_QUPs, Change the structure (`ssc_qupv3_perms_auto`) as per the following code:

```
{ QUPV3_SSC_SE3, QUPV3_PROTOCOL_UART_2W, QUPV3_MODE_FIFO, AC_HLOS, TRUE,
TRUE, FALSE } → Two wire UART
```

```
{ QUPV3_SSC_SE4, QUPV3_PROTOCOL_UART_4W, QUPV3_MODE_FIFO, AC_HLOS, TRUE,
TRUE, FALSE } → Four wire UART
```

In the preceding example code:

- QUPV3_MODE_FIFO is a FIFO mode that can be modified to GSI mode by replacing FIFO with GSI, for example, QUPV3_MODE_GSI
- AC_HLOS is the access permission that can be modified to any sub system as per requirement, for example, AC_TZ, AC_ADSP_Q6_ELF
- SE protocol can have one of the following:
  - SPI
  - I^2C
  - UART

10 SPI – Hardware overview

10.1 SPI core

The SPI allows full half-duplex, synchronous, and serial communication between a master and slave. There is no explicit communication framing, error checking, or defined data word length. Key features are:

- Supports a maximum of four Chip Selects (CS) per bus
- Four wire protocol, MOSI, MISO, CLK and chip select/multi chip select
- 1 MHz to 50 MHz support with full duplex support
- Four modes for clock and polarity settings, configurable bits per word

10.2 Base addresses

Table 10-1 Base address

QUPv3 hardware ID	QUP core	Physical address	IRQ #	Kernel SPI clock name
QUPV3_0	QUPV3_0_SE2	0x00888000	603	GCC_QUPV3_WRAP0_S2_CLK
QUPV3_1	QUPV3_1_SE0	0x00A80000	353	GCC_QUPV3_WRAP1_S0_CLK
QUPV3_1	QUPV3_1_SE2	0x00A88000	355	GCC_QUPV3_WRAP1_S2_CLK
QUPV3_1	QUPV3_1_SE3	0x00A8C000	356	GCC_QUPV3_WRAP1_S3_CLK
SSC_QUPV3	SSC_QUPV3_SE0	0 x 62680000	442	SCC_QUPV3_SE0_CLK
SSC_QUPV3	SSC_QUPV3_SE1	0 x 62684000	443	SCC_QUPV3_SE1_CLK
SSC_QUPV3	SSC_QUPV3_SE2	0 x 62688000	444	SCC_QUPV3_SE2_CLK
SSC_QUPV3	SSC_QUPV3_SE3	0 x 6268C000	445	SCC_QUPV3_SE3_CLK
SSC_QUPV3	SSC_QUPV3_SE4	0 x 62690000	446	SCC_QUPV3_SE4_CLK
SSC_QUPV3	SSC_QUPV3_SE5	0 x 62694000	447	SCC_QUPV3_SE5_CLK

11 SPI Master – Software configuration

11.1 Configure the QUPv3_SE core as SPI Master in the kernel

This section describes the steps required to configure, and use any of the QUPv3 SE cores available in the SA6155 chipset as an SPI device.

By default, QTI has preconfigured QUPv3 SE as SPI. For detailed information, see /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi

11.1.1 Code modification to enable the QUPv3_SE as SPI Master

The following examples enable you to configure QUPv3 SE as SPI for the SA6155 chipset.

- Device tree source – /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi
- GPIO table – /kernel/arch/arm/boot/dts/qcom/sm6150-pinctrl.dtsi
- GPIO table for SC_QUPs – /kernel/arch/arm/boot/dts/qcom/sm6150-slpi-pinctrl.dtsi
- TrustZone changes - trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

1. Add a device tree node. The following nodes already present. If the nodes are not present for a QUP supporting SPI, add the nodes and change status to “ok”.

Path: /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi

```
/* QUPv3 South Instances */
/* SPI */
qupv3_se2_spi: spi@888000 {
     compatible = "qcom,spi-geni";
     #address-cells = <1>;
     #size-cells = <0>;
     reg = <0x888000 0x4000>;
     reg-names = "se_phys";
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP0_S2_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se2_spi_active>;
     pinctrl-1 = <&qupv3_se2_spi_sleep>;
     interrupts = <GIC_SPI 603 0>;
     spi-max-frequency = <50000000>;
```

```
        qcom,wrapper-core = <&qupv3_0>;
        dmas = <&gpi_dma0 0 2 1 64 0>,
             <&gpi_dma0 1 2 1 64 0>;
        dma-names = "tx", "rx";
        status = "disabled";
    };


    /* QUPv3 North instances */
    /* SPI */
    qupv3_se4_spi: spi@a80000 {
        compatible = "qcom,spi-geni";
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0xa80000 0x4000>;
        reg-names = "se_phys";
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP1_S0_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se4_spi_active>;
        pinctrl-1 = <&qupv3_se4_spi_sleep>;
        interrupts = <GIC_SPI 353 0>;
        spi-max-frequency = <50000000>;
        qcom,wrapper-core = <&qupv3_1>;
        dmas = <&gpi_dma1 0 0 1 64 0>,
             <&gpi_dma1 1 0 1 64 0>;
        dma-names = "tx", "rx";
        status = "disabled";
    };

    qupv3_se6_spi: spi@a88000 {
        compatible = "qcom,spi-geni";
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0xa88000 0x4000>;
        reg-names = "se_phys";
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP1_S2_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se6_spi_active>;
        pinctrl-1 = <&qupv3_se6_spi_sleep>;
        interrupts = <GIC_SPI 355 0>;
        spi-max-frequency = <50000000>;
        qcom,wrapper-core = <&qupv3_1>;
```

```
        dmas = <&gpi_dma1 0 2 1 64 0>,
             <&gpi_dma1 1 2 1 64 0>;
        dma-names = "tx", "rx";
        status = "disabled";
    };

    qupv3_se7_spi: spi@a8c000 {
        compatible = "qcom,spi-geni";
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0xa8c000 0x4000>;
        reg-names = "se_phys";
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_gcc GCC_QUPV3_WRAP1_S3_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
             <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se7_spi_active>;
        pinctrl-1 = <&qupv3_se7_spi_sleep>;
        interrupts = <GIC_SPI 356 0>;
        spi-max-frequency = <50000000>;
        qcom,wrapper-core = <&qupv3_1>;
        dmas = <&gpi_dma1 0 3 1 64 0>,
             <&gpi_dma1 1 3 1 64 0>;
        dma-names = "tx", "rx";
        status = "disabled";
    };
```

Steps to configure SSC_QUPs

```
/* SPI */
qupv3_se9_spi: spi@62684000 {
        compatible = "qcom,spi-geni";
        reg = <0x62684000 0x4000>;
        reg-names = "se_phys";
        interrupts = <GIC_SPI 443 0>;
        #address-cells = <1>;
        #size-cells = <0>;
        clock-names = "se-clk", "m-ahb", "s-ahb";
        clocks = <&clock_scc SCC_QUPV3_SE1_CLK>,
             <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
             <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&qupv3_se9_spi_active>;
        pinctrl-1 = <&qupv3_se9_spi_sleep>;
        spi-max-frequency = <50000000>;
        qcom,wrapper-core = <&qupv3_2>;
        status = "disabled";
};
```

```
qupv3_se10_spi: spi@62688000 {
     compatible = "qcom,spi-geni";
     reg = <0x62688000 0x4000>;
     reg-names = "se_phys";
     interrupts = <GIC_SPI 444 0>;
     #address-cells = <1>;
     #size-cells = <0>;
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_scc SCC_QUPV3_SE2_CLK>,
          <&clock_scc SCC_QUPV3_M_HCLK_CLK>,
          <&clock_scc SCC_QUPV3_S_HCLK_CLK>;
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se10_spi_active>;
     pinctrl-1 = <&qupv3_se10_spi_sleep>;
     spi-max-frequency = <50000000>;
     qcom,wrapper-core = <&qupv3_2>;
     status = "disabled";
};
```

2. Set the GPIO. Modify the files at /kernel/arch/arm64/boot/dts/qcom/sm6150-pinctrl.dtsi.

```
/* QUPv3_0 South SE mappings */
/* SE 2 pin mappings */
qupv3_se2_spi_pins: qupv3_se2_spi_pins {
               qupv3_se2_spi_active: qupv3_se2_spi_active {
                    mux {
                         pins = "gpio0", "gpio1", "gpio2",
                                        "gpio3";
                         function = "qup02";
                    };

                    config {
                         pins = "gpio0", "gpio1", "gpio2",
                                        "gpio3";
                         drive-strength = <6>;
                         bias-disable;
                    };
               };

               qupv3_se2_spi_sleep: qupv3_se2_spi_sleep {
                    mux {
                         pins = "gpio0", "gpio1", "gpio2",
                                        "gpio3";
                         function = "gpio";
                    };
```

```
                    config {
                        pins = "gpio0", "gpio1", "gpio2",
                                "gpio3";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };
            };
        qupv3_se4_spi_pins: qupv3_se4_spi_pins {
                qupv3_se4_spi_active: qupv3_se4_spi_active {
                    mux {
                        pins = "gpio20", "gpio21", "gpio22",
                                "gpio23";
                        function = "qup10";
                    };

                    config {
                        pins = "gpio20", "gpio21", "gpio22",
                                "gpio23";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };

                qupv3_se4_spi_sleep: qupv3_se4_spi_sleep {
                    mux {
                        pins = "gpio20", "gpio21", "gpio22",
                                "gpio23";
                        function = "gpio";
                    };

                    config {
                        pins = "gpio20", "gpio21", "gpio22",
                                "gpio23";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };
            };
    qupv3_se6_spi_pins: qupv3_se6_spi_pins {
                qupv3_se6_spi_active: qupv3_se6_spi_active {
                    mux {
                        pins = "gpio6", "gpio7", "gpio8",
                                "gpio9";
                        function = "qup12";
                    };

                    config {
```

```
                    pins = "gpio6", "gpio7", "gpio8",
                                "gpio9";
                    drive-strength = <6>;
                    bias-disable;
               };
          };

          qupv3_se6_spi_sleep: qupv3_se6_spi_sleep {
               mux {
                    pins = "gpio6", "gpio7", "gpio8",
                                "gpio9";
                    function = "gpio";
               };

               config {
                    pins = "gpio6", "gpio7", "gpio8",
                                "gpio9";
                    drive-strength = <6>;
                    bias-disable;
               };
          };
     };

qupv3_se7_spi_pins: qupv3_se7_spi_pins {
          qupv3_se7_spi_active: qupv3_se7_spi_active {
               mux {
                    pins = "gpio10", "gpio11", "gpio12",
                                "gpio13";
                    function = "qup13";
               };

               config {
                    pins = "gpio10", "gpio11", "gpio12",
                                "gpio13";
                    drive-strength = <6>;
                    bias-disable;
               };
          };

          qupv3_se7_spi_sleep: qupv3_se7_spi_sleep {
               mux {
                    pins = "gpio10", "gpio11", "gpio12",
                                "gpio13";
                    function = "gpio";
               };

               config {
                    pins = "gpio10", "gpio11", "gpio12",
```

```
                              "gpio13";
                    drive-strength = <6>;
                    bias-disable;
               };
          };
     };
```

Modify the files at /kernel/arch/arm64/boot/dts/qcom/sm6150-slpi-pinctrl.dtsi. for SSC_QUPs

```
qupv3_se9_spi_pins: qupv3_se9_spi_pins {
               qupv3_se9_spi_active: qupv3_se9_spi_active {
                    mux {
                         pins = "gpio2", "gpio3", "gpio4",
                                   "gpio5";
                         function = "func1";
                    };

                    config {
                         pins = "gpio2", "gpio3", "gpio4",
                                   "gpio5";
                         drive-strength = <6>;
                         bias-disable;
                    };
               };

               qupv3_se9_spi_sleep: qupv3_se9_spi_sleep {
                    mux {
                         pins = "gpio2", "gpio3", "gpio4",
                                   "gpio5";
                         function = "gpio";
                    };

                    config {
                         pins = "gpio2", "gpio3", "gpio4",
                                   "gpio5";
                         drive-strength = <6>;
                         bias-disable;
                    };
               };
          };

          qupv3_se10_spi_pins: qupv3_se10_spi_pins {
               qupv3_se10_spi_active: qupv3_se10_spi_active {
                    mux {
                         pins = "gpio8", "gpio9", "gpio10",
                                   "gpio11";
                         function = "func1";
                    };
```

```
                config {
                    pins = "gpio8", "gpio9", "gpio10",
                                "gpio11";
                    drive-strength = <6>;
                    bias-disable;
                };
            };

            qupv3_se10_spi_sleep: qupv3_se10_spi_sleep {
                mux {
                    pins = "gpio8", "gpio9", "gpio10",
                                "gpio11";
                    function = "gpio";
                };

                config {
                    pins = "gpio8", "gpio9", "gpio10",
                                "gpio11";
                    drive-strength = <6>;
                    bias-disable;
                };
            };
        };
```

3. Change in TrustZone.

   Modify:
   trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

NOTE: The folder contains all the structures defined in the QUPAC_Access.c file.

To update the SPI Master mode, change the structure (`qupv3_perms_auto`) as per the following example code:

```
{ QUPV3_1_SE2, QUPV3_PROTOCOL_SPI,     QUPV3_MODE_GSI,  AC_HLOS,
TRUE, TRUE, FALSE }
```

In case of SSC_QUPs, Change the structure (`ssc_qupv3_perms_auto`) as per the following code:

```
{ QUPV3_SSC_SE0, QUPV3_PROTOCOL_SPI, QUPV3_MODE_FIFO, AC_HLOS, TRUE,
TRUE, FALSE }
```

In the preceding example code:

- QUPV3_MODE_FIFO is a FIFO mode that can be modify to GSI mode by replacing FIFO with GSI, for example, QUPV3_MODE_GSI
- AC_HLOS is the access permission that can be modify to any sub system as per requirement, for example, AC_TZ, AC_ADSP_Q6_ELF

- SE protocol can have one of the following:
  - SPI
  - I^2C
  - UART

11.2 Verify the SPI bus

Find the SPI bus registered under /sys/class/spi_master/spi, where cell-index matches the bus number.

```
adb shell --> Get adb shell
cd /sys/class/spi_master to list all the spi master
root@android:/sys/class/spi_master # ls
ls
spi0
spi6
spi7

Enable the SPI loop back test.
&spi_10 {
        status = "ok";
        spidev@0 {
              compatible = "spidev";
              spi-max-frequency = <19200000>;
              reg = <0>;
        };
};
#cd /data/kernel-tests
#chmod 777 *
# ./spidevtest spidev3.1 -v -l -t 1 -c -z 15060000 → To run the SPI
loopback test.
```

11.2.1 Register a SPI slave device using the device tree

After the SPI bus is registered, create a slave device driver and register it with the SPI master. For examples of SPI slave devices, see the following files:

- /kernel/Documentation/devicetree/bindings/spi/qcom,spi-geni-qcom.txt
- //kernel/arch/arm/boot/dts/qcom/sm6150-qupv3.dtsi

The following example shows the minimum requirements to register a slave device:

1. Create a device tree node, per the above mentioned nodes.

```
qupv3_spi10: spi@a84000 {
              compatible = "qcom,spi-geni";
```

```
                #address-cells = <1>;
                #size-cells = <0>;
                reg = <0xa84000 0x4000>;
                reg-names = "se_phys";
                clock-names = "se-clk", "m-ahb", "s-ahb";
                clocks = <&clock_gcc GCC_QUPV3_WRAP0_S0_CLK>,
                        <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
                        <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
                pinctrl-names = "default", "sleep";
                pinctrl-0 = <&qup_1_spi_2_active>;
                pinctrl-1 = <&qup_1_spi_2_sleep>;
                interrupts = <GIC_SPI 354 0>;
                spi-max-frequency = <19200000>;
                qcom,wrapper-core = <&qupv3_0>;

                dev@0 {
                        compatible = "dummy,slave";
                        reg = <0>;
                        spi-max-frequency = <9600000>;
                };
        };
```

2. Sample slave device driver.

```
#include <linux/module.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <linux/spi/spi.h>
#include <linux/interrupt.h>
#include <linux/slab.h>
#include <linux/gpio.h>
#include <linux/debugfs.h>
#include <linux/seq_file.h>
#include <linux/regulator/consumer.h>
#include <linux/string.h>
#include <linux/of_gpio.h>

#ifdef CONFIG_OF //Open firmware must be defined for dts usage
static struct of_device_id qcom_spi_test_table[] = {
     { .compatible = "qcom,spi-test",}, //Compatible node must match dts
     { },
};
#else
#define qcom_spi_test_table NULL
#endif
#define BUFFER_SIZE 4<<10
struct spi_message spi_msg;
struct spi_transfer spi_xfer;
u8 *tx_buf; //This needs to be DMA friendly buffer
static int spi_test_transfer(struct spi_device *spi)
{
     spi_message_init(&spi_msg);
```

```
     spi_xfer.tx_buf = tx_buf;
     spi_xfer.len = BUFFER_SIZE;
     spi_xfer.bits_per_word = 8;
     spi_xfer.speed_hz = spi->max_speed_hz;

     spi_message_add_tail(&spi_xfer, &spi_msg);

     return spi_sync(spi, &spi_msg);
}


static int __devinit spi_test_probe(struct spi_device *spi)
{

     int irq_gpio = -1;
     int irq;
     int cs;
     int cpha,cpol,cs_high;
     u32 max_speed;

     dev_err(&spi->dev, "s\n", __func__);

        //allocate memory for transfer
     tx_buf = kmalloc(BUFFER_SIZE, GFP_ATOMIC);
     if(tx_buf == NULL){
          dev_err(&spi->dev, "s: mem alloc failed\n", __func__);
          return -ENOMEM;
     }
     //Parse data using dt.
     if(spi->dev.of_node){
       irq_gpio = of_get_named_gpio_flags(spi->dev.of_node,
"qcom_spi_test,irq-gpio", 0, NULL);
     }
     irq = spi->irq;
     cs = spi->chip_select;
     cpha = ( spi->mode & SPI_CPHA ) ? 1:0;
     cpol = ( spi->mode & SPI_CPOL ) ? 1:0;
     cs_high = ( spi->mode & SPI_CS_HIGH ) ? 1:0;
     max_speed = spi->max_speed_hz;
     dev_err(&spi->dev, "gpio [d] irq [d] gpio_irq [d] cs [x] CPHA [x]
CPOL [x] CS_HIGH [x]\n",
          irq_gpio, irq, gpio_to_irq(irq_gpio), cs, cpha, cpol,
cs_high);

     dev_err(&spi->dev, "Max_speed [d]\n", max_speed );

     //Once you have a spi_device structure you can do a transfer
anytime
```

```
     spi->bits_per_word = 8;
     dev_err(&spi->dev, "SPI sync returned [d]\n",
spi_test_transfer(spi));
     return 0;
}


//SPI Driver Info
static struct spi_driver spi_test_driver = {
     .driver = {
          .name     = "qcom_spi_test",
          .owner    = THIS_MODULE,
          .of_match_table = qcom_spi_test_table,
     },
     .probe         = spi_test_probe,
};


static int __init spi_test_init(void)
{
     return spi_register_driver(&spi_test_driver);
}

static void __exit spi_test_exit(void)
{
     spi_unregister_driver(&spi_test_driver);
}
module_init(spi_test_init);
module_exit(spi_test_exit);
MODULE_DESCRIPTION("SPI TEST");
MODULE_LICENSE("GPL v2");
```

In the kernel log, the following message indicates that the device tree is successfully configured.

```
<3>[    2.503571] qcom_spi_test spi6.0: spi_test_probe
<3>[    2.507305] qcom_spi_test spi6.0: gpio [61] irq [306] gpio_irq
[306]
                      cs [0] CPHA [1] CPOL [1] CS_HIGH [1]
<3>[    2.516825] qcom_spi_test spi6.0: Max_speed [4800000]
<3>[    2.521932] qcom_spi_test spi6.0: SPI sync returned [0]
```

12 SPI Slave – Software configuration

NOTE: This chapter was added to this document revision.

12.1 Configure the QUPv3_SE core as SPI Slave in the kernel

This section describes the steps required to configure the QUPv3 SE (QUP0_SE2, QUP1_SE2) cores available in the SA6155 chipset as an SPI slave device.

By default, QTI has preconfigured QUPv3 SE as SPI. For detailed information, see /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi

12.1.1 Code modification to enable the QUPv3_SE as SPI Slave

The following examples enable you to configure QUPv3 SE as SPI for the SA6155 chipset.

- Add alias for the node - /kernel/arch/arm64/boot/dts/qcom/sm6150.dtsi
- Device tree source – /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi
- Device tree change for slave - /kernel/arch/arm64/boot/dts/qcom/sa6155-adp-star.dtsi
- GPIO table – /kernel/arch/arm/boot/dts/qcom/sm6150-pinctrl.dtsi
- TrustZone changes - trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

1. Add alias for the required node in /kernel/arch/arm64/boot/dts/qcom/sm6150.dtsi file.

```
aliases {
   ……
   spi2 = &qupv3_se2_spi;
   spi6 = &qupv3_se6_spi;
   ……
};
```

2. The following device tree nodes are already present and configured as SPI Master; no change is required.

   Path : /kernel/arch/arm64/boot/dts/qcom/sm6150-qupv3.dtsi

```
/* QUPv3 South Instances */
/* SPI */
```

```
qupv3_se2_spi: spi@888000 {
     compatible = "qcom,spi-geni";
     #address-cells = <1>;
     #size-cells = <0>;
     reg = <0x888000 0x4000>;
     reg-names = "se_phys";
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP0_S2_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_0_S_AHB_CLK>;
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se2_spi_active>;
     pinctrl-1 = <&qupv3_se2_spi_sleep>;
     interrupts = <GIC_SPI 603 0>;
     spi-max-frequency = <50000000>;
     qcom,wrapper-core = <&qupv3_0>;
     dmas = <&gpi_dma0 0 2 1 64 0>,
          <&gpi_dma0 1 2 1 64 0>;
     dma-names = "tx", "rx";
     status = "disabled";
};

/* QUPv3 North Instances */
/* SPI */
qupv3_se6_spi: spi@a88000 {
     compatible = "qcom,spi-geni";
     #address-cells = <1>;
     #size-cells = <0>;
     reg = <0xa88000 0x4000>;
     reg-names = "se_phys";
     clock-names = "se-clk", "m-ahb", "s-ahb";
     clocks = <&clock_gcc GCC_QUPV3_WRAP1_S2_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_M_AHB_CLK>,
          <&clock_gcc GCC_QUPV3_WRAP_1_S_AHB_CLK>;
     pinctrl-names = "default", "sleep";
     pinctrl-0 = <&qupv3_se6_spi_active>;
     pinctrl-1 = <&qupv3_se6_spi_sleep>;
     interrupts = <GIC_SPI 355 0>;
     spi-max-frequency = <50000000>;
     qcom,wrapper-core = <&qupv3_1>;
     dmas = <&gpi_dma1 0 2 1 64 0>,
          <&gpi_dma1 1 2 1 64 0>;
     dma-names = "tx", "rx";
     status = "disabled";
};
```

2. Add SPI slave-related changes in /kernel/arch/arm64/boot/dts/qcom/sa6155-adp-star.dtsi for the required SE.

```
&qupv3_se2_spi {
        status = "ok";
        spi-max-frequency = <19200000>;
        qcom,slv-ctrl;

        slave@0 {
                compatible = "spidev";
                reg = <0>;
                spi-max-frequency =  <19200000>;
        };
};
```

3. Set the GPIO configuration. Modify /kernel/arch/arm/boot/dts/qcom/sm6150-pinctrl.dtsi. These entries are already present and no change is required.

```
/* SE 2 pin mappings */
qupv3_se2_spi_pins: qupv3_se2_spi_pins {
            qupv3_se2_spi_active: qupv3_se2_spi_active {
                mux {
                    pins = "gpio0", "gpio1", "gpio2",
                                "gpio3";
                    function = "qup02";
                };

                config {
                    pins = "gpio0", "gpio1", "gpio2",
                                "gpio3";
                    drive-strength = <6>;
                    bias-disable;
                };
            };

            qupv3_se2_spi_sleep: qupv3_se2_spi_sleep {
                mux {
                    pins = "gpio0", "gpio1", "gpio2",
                                "gpio3";
                    function = "gpio";
                };

                config {
                    pins = "gpio0", "gpio1", "gpio2",
```

```
                                        "gpio3";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };
            };
    /*SE 6 pin mapping */
    qupv3_se6_spi_pins: qupv3_se6_spi_pins {
                qupv3_se6_spi_active: qupv3_se6_spi_active {
                    mux {
                        pins = "gpio6", "gpio7", "gpio8",
                                        "gpio9";
                        function = "qup12";
                    };

                    config {
                        pins = "gpio6", "gpio7", "gpio8",
                                        "gpio9";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };

                qupv3_se6_spi_sleep: qupv3_se6_spi_sleep {
                    mux {
                        pins = "gpio6", "gpio7", "gpio8",
                                        "gpio9";
                        function = "gpio";
                    };

                    config {
                        pins = "gpio6", "gpio7", "gpio8",
                                        "gpio9";
                        drive-strength = <6>;
                        bias-disable;
                    };
                };
            };
```

3. Change in TrustZone.

   Modify:
   trustzone_images\core\settings\buses\qup_accesscontrol\qupv3\config\6150\QUPAC_Access.c

NOTE: The folder contains all the structures defined in the QUPAC_Access.c file.

To update the SPI Slave mode, change the structure (`qupv3_perms_auto`) as per the following example code:

```
{ QUPV3_0_SE2, QUPV3_PROTOCOL_SPI_SLAVE, QUPV3_MODE_FIFO,  AC_HLOS,
TRUE, TRUE, FALSE }, // SPI Slave
```

In the preceding example code:

- QUPv3_MODE_FIFO is a FIFO mode that can be modify to GSI mode by replacing FIFO with GSI, for example, QUPV3_MODE_GSI
- AC_HLOS is the access permission that can be modify to any sub system as per requirement, for example, AC_TZ, AC_ADSP_Q6_ELF
- SE protocol can have one of the following:
  - SPI
  - I^2C
  - UART

12.2 Verify the SPI bus

Find the SPI bus registered under /sys/class/spi_slave/spi#, where cell-index matches the bus number.

```
adb shell --> Get adb shell
cd /sys/class/spi_slave to list all the spi master
root@android:/sys/class/spi_slave # ls
ls
spi2
spi6
```

A FAQs

QUESTION 1 What is the optimal configuration for minimum sleep current for an unconnected GPIO?

ANSWER For unconnected pins, input pull-down or output pull-down provides the least leakage.

QUESTION 2 How do I check GPIO owner configuration (Modem/APSS side)?

ANSWER The modem or application processor does not own the GPIO. All the GPIOs are accessible on both the modem and the apps side. Based on the design, GPIOs are configured on either the apps or the modem side.

QUESTION 3 How do I check if any given GPIO is a wake-up-capable pin?

ANSWER See Table 3-6 for the list of wake-up-capable GPIOs.

QUESTION 4 What are I^2C SDA and SCL output fall time specifications?

ANSWER These values are as per standard I^2C specifications.

Fall time of SCLH signal	Minimum	Maximum
Capacitive load from 10 pF to 100 pF with capacitive load of 400 pF	10	40
Capacitive load of 400 pF with capacitive load from 10 pF to 100 pF	20	80

Fall time of SDAH signal	Capacitive load of 400 pF	Maximum
Capacitive load from 10 pF to 100 pF	10	80
Capacitive load of 400 pF	20	160

QUESTION 5 Is the I^2C clock stretching supported?

ANSWER Yes.

QUESTION 6 Are the SDM and QTI chipsets I^2C timing parameters the same as I^2C standard specifications?

ANSWER Yes.

QUESTION 7 What is the maximum baud rate and how can I change it?

ANSWER There are two types of UART drivers supported:

- High-speed UART (msm_serial_hs.c) – supports up to 4 Mbps
- Low-speed UART (msm_serial_hs_lite.c) – supports up to 11.2 Kbps

The UART driver exposes the functionality of the baud rate. Hence, OEMs have the feasibility to change it as per their requirement.

QUESTION 8 Which UART modes are supported?

ANSWER There are two UART modes supported:

- FIFO (msm_serial_hs_lite.c)
- BAM (msm_serial_hs.c)

Depending on the requirement, OEMs can choose the corresponding driver.

QUESTION 9 How do I check that CDT is loaded from XBL image or from CDT partition table?

ANSWER See Boot up logs for the following message.

a. If the delta is 0 Bytes, then CDT is not updated and it is taking default CDT only.

```
B -    394151 - Using default CDT
D -     28761 - boot_config_data_table_init, Delta - (0 Bytes)
B -    396103 - CDT Version:3,Platform ID:8,Major ID:1,Minor
ID:0,Subtype:0
```

b. If the delta is 420 Bytes, then CDT is updated and it is taking from eMMC partition table.

```
D -     28761 - boot_config_data_table_init, Delta - (420 Bytes)
B -    396103 - CDT Version:3,Platform ID:8,Major ID:1,Minor
ID:0,Subtype:0
```

QUESTION 10 How do I check CDT is loaded from XBL image or from CDT partition table.

ANSWER See Boot up logs for the following message.

i) If the Delta - (0 Bytes) then CDT is not updated and it is taking default CDT only.

```
B -    394151 - Using default CDT
D -     28761 - boot_config_data_table_init, Delta - (0 Bytes)
B -    396103 - CDT Version:3,Platform ID:8,Major ID:1,Minor ID:0,Subtype:0
```

ii) If the Delta - (420 Bytes) then CDT is updated and it is taking from eMMC partition table.

```
D -     28761 - boot_config_data_table_init, Delta - (420 Bytes)
B -    396103 - CDT Version:3,Platform ID:8,Major ID:1,Minor ID:0,Subtype
```

B References

B.1 Related documents

Title	Number
Qualcomm Technologies, Inc.	
XML Tag Description For Display Module In GCDB	80-BA103-1
DSI Timing Parameters User Interactive Spreadsheet	80-NH713-1
Multimedia Driver Development and Bringup Guide - Camera	80-NU323-2
Linux Android Display Driver Porting Guide	80-NN766-1
Peripheral Bus Client Drivers Limitation for 6 GB RAM support	80-NU154-43
Third-Party Fingerprint Solution Integration on Qualcomm Platforms	80-NV103-1
SA6155 Linux Android Software User Manual	SP80-PK753-4
Security TrustZone QSEE Overview	80-NL239-5
Resources	
PINCTRL (PIN CONTROL) subsystem	

B.2 Acronyms and terms

Acronym or term	Definition
ADM	Application data mover
BAM	Bus access manager
DTC	Device tree compile
DMA	Direct memory access
EEPROM	Erasable programmable read-only memory
GPIO	General purpose input/output
I^2C	Interintegrated circuit
IRQ	Interrupt request
LK	Little kernel
OF	Open firmware
QUP	Qualcomm universal peripheral
QRD	Qualcomm reference design
SPI	Serial peripheral interface
TLMM	Top-level mode multiplexer
UART	Universal asynchronous receiver/transmitter