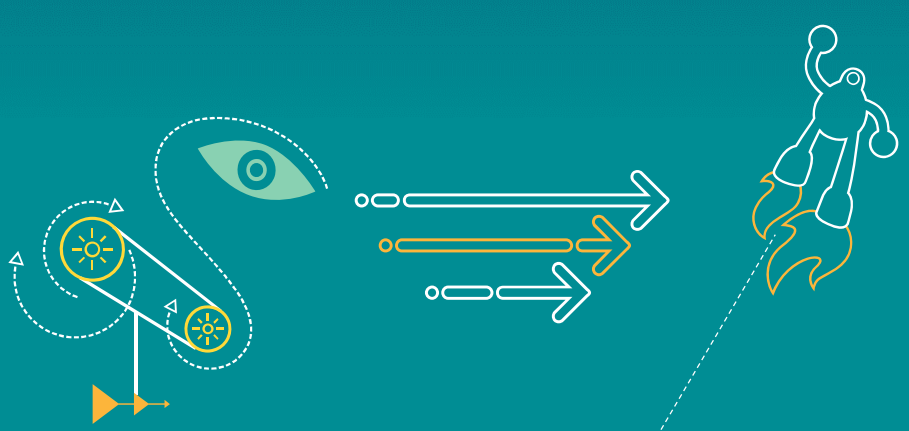
MSM8937/MSM8940/MSM8920 Linux USB Overview

Qualcomm Technologies, Inc.



80-P2485-17 E

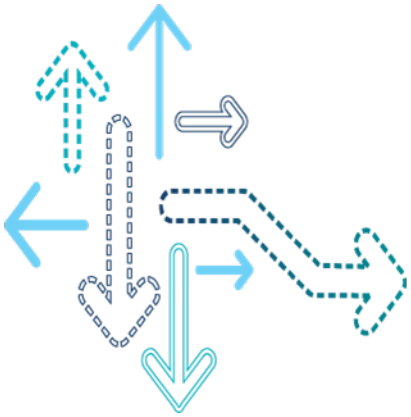
Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision History

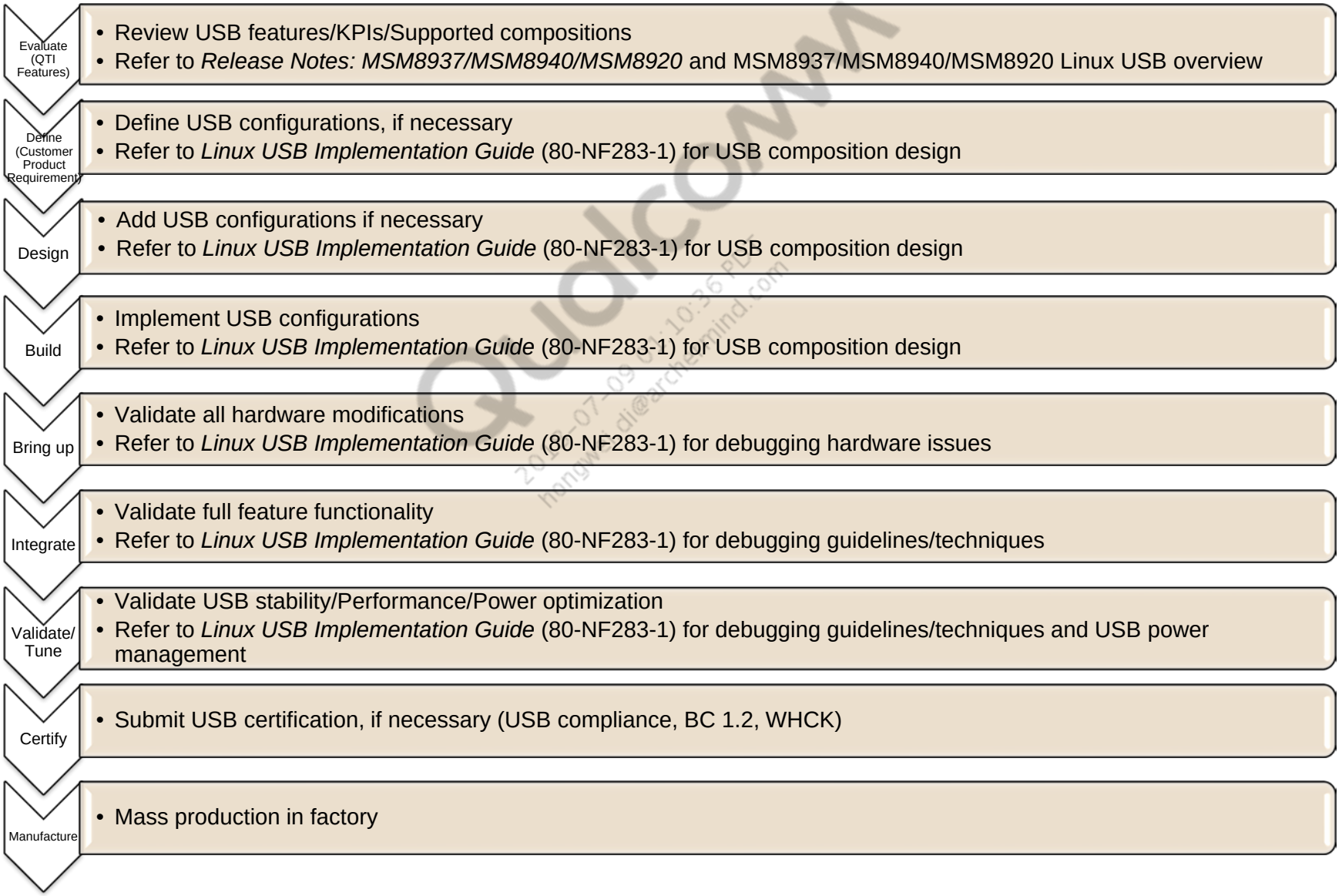
Revision	Date	Description
A	November 2015	Initial release
B	January 2016	Added slides 40 and 41
C	June 2016	Added slide 42; updates relating to MSM8940
D	July 2016	Updated Slides 12 and 24
E	November 2016	Updates for MSM8920

Contents

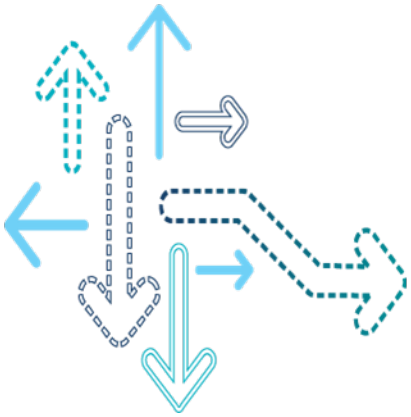
OEM Project Lifecycle



Documentation Map for Customer Project Phases



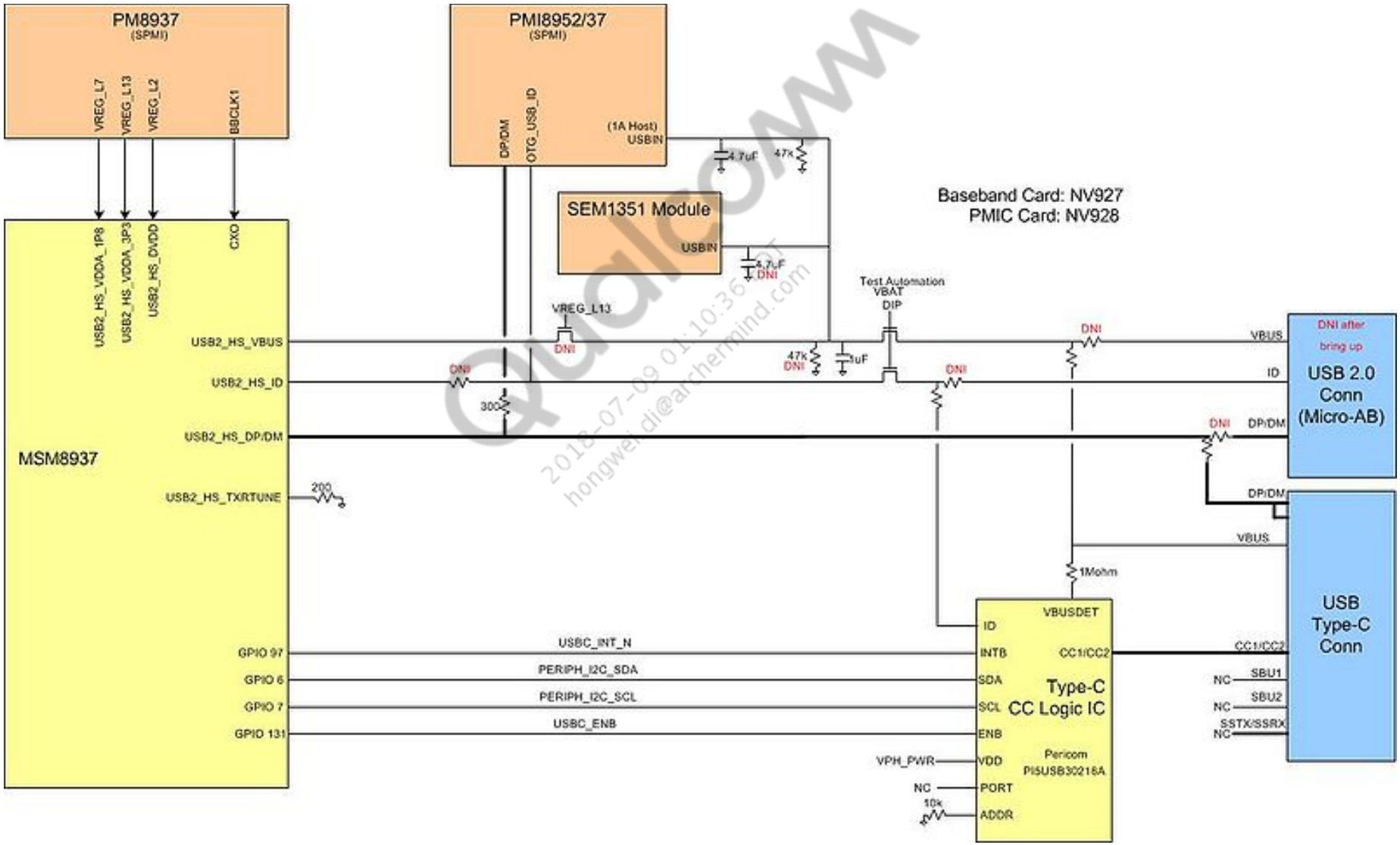
MSM8937/MSM8940/MSM8920 USB Hardware Overview



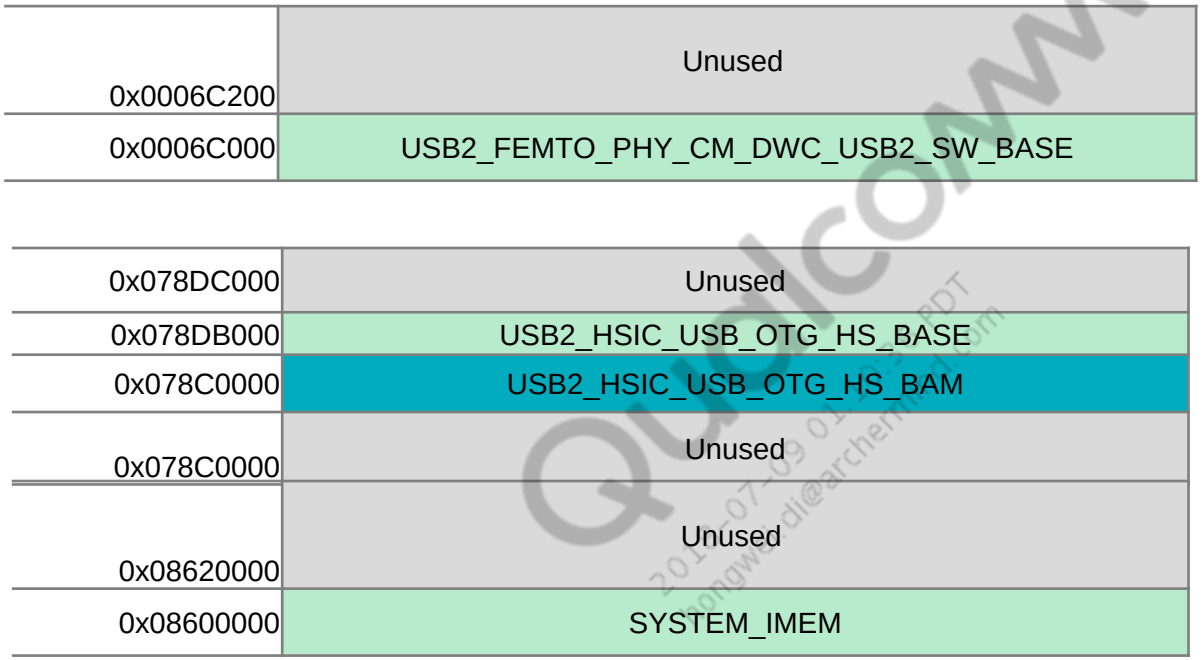
MSM8937/MSM8940/MSM8920 USB Hardware Overview

Hardware features	MSM8937/ MSM8940/ MSM8920	Comments
USB 2.0 core	1	Single USB port; new Synopsys 28 nm Femto PHY
Number of USB endpoints	32	16-in and 16-out including control endpoint
HSIC	No	Not supported
PMIC-based VBUS indication	Yes	VBUS is not routed to the USB PHY; charger driver notifies USB software of the VBUS event
PMIC-based ID indication	Yes	USB_ID is not routed to the USB PHY; PMIC notifies USB OTG software of the USB_ID event
Low Power Mode (LPM)	Yes	Supports LPM and allows VDD minimum on cable disconnect
USB type-C	Yes	Supports USB type-C connector

MSM8937/MSM8940/MSM8920 USB Configuration Example



USB Memory Address



USB device nodes

USB PHY base address

usb_otg:usb@0x78db000

usbbam@0x78C4000

android_usb@086000c8

Notes:

- The image shows the address range for the main USB controller
- reg = <0x086000c8 0xc8>; memory address for storing Download mode cookies
- qcom, pm-qos-latency = <2 1001 12701>; android_usb device node to pass pm_qos latency values
- For more details, see *Linux USB Implementation Guide* (80-NF283-1)

MSM8937/MSM8940/MSM8920 USB Power Rails

Voltage rail	Voltage level	Comments
VREG L7	1.8 V	1.8 V regulator is used for HSPHY
VREG L13	3.075 V	3.3 V regulator is used for HSPHY
VREG L2	1.2 V	HSPHY VDD core
smbcharger_charger_otg	5 V	USB OTG host regulator
Vdd-io-supply L5	1.8 V	1.8 V VDD supply for type-C

MSM8937/MSM8940/MSM8920 USB Clocks

Clock name	Operating frequency	Comments
clk_gcc_usb_hs_system_clk “core_clk”	133 MHz	Asynchronous to the bus clock
clk_gcc_usb_hs_ahb_clk “iface_clk”	100 MHz	Interface clock to the AHB BU
clk_gcc_usb2a_phy_sleep_clk “sleep_clk”	32 kHz	Sleep clock
clk_bimc_usb_a_clk “bimic_clk”	700 MHz	BIMIC clock vote when USB is in Peripheral mode
clk_snoc_usb_a_clk “snoc_clk”	200 MHz	SNoC clock vote when USB is in Peripheral mode
clk_pcnoc_usb_a_clk “pcnoc_clok”	100 MHz	PCNoC clock vote when USB is in Peripheral mode
clk_xo_otg_clk “xo”	19.2 MHz	External reference clock source to the HSPH
clk_gcc_usb_hs_phy_cfg_ahb_clk “phy_csr_clk”	100 MHz	Required to access PHY CSR registers via the AHB2PHY interface; this clock is turned off during LPM

MSM8937/MSM8940/MSM8920 USB Reset Control

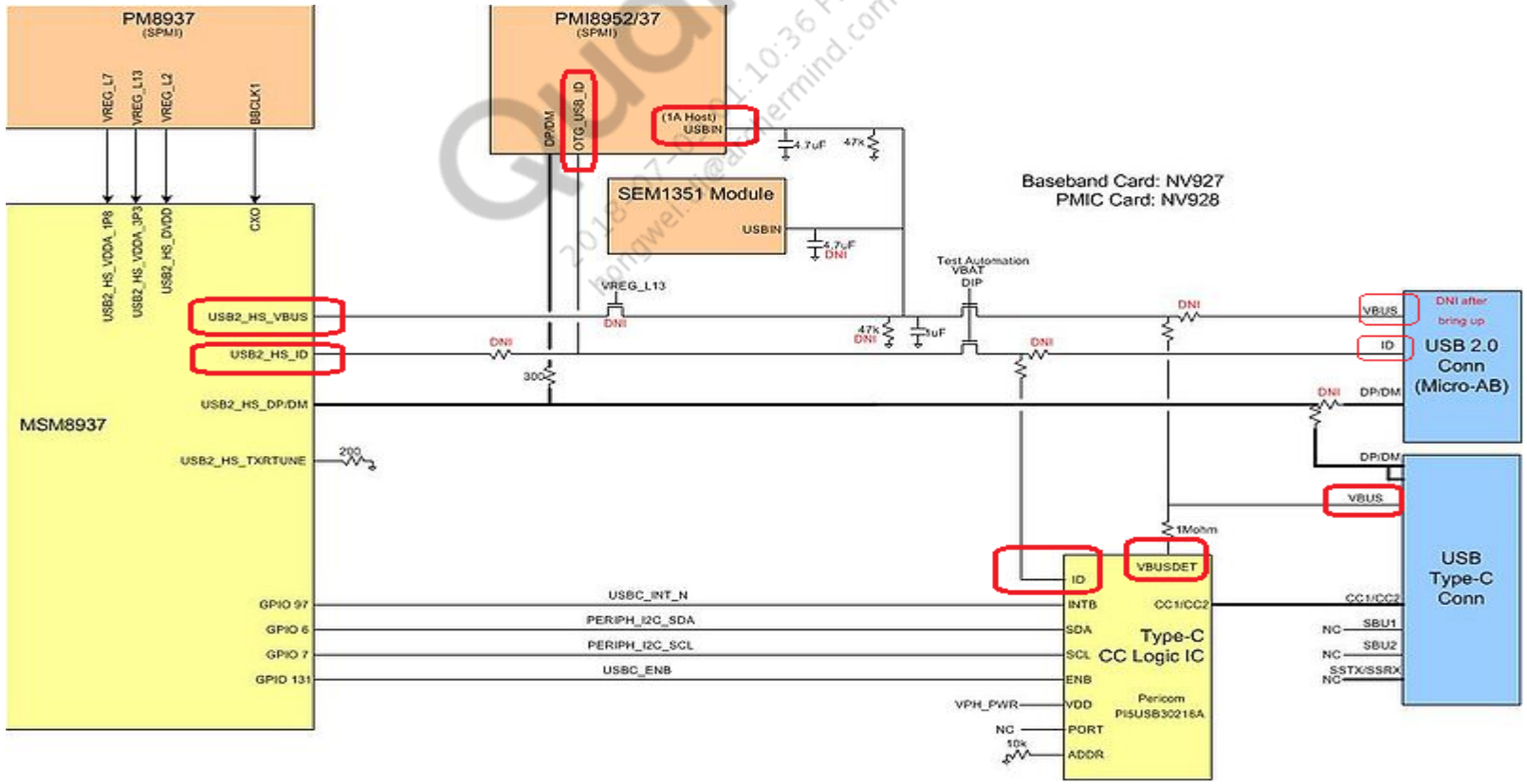
- Clock controller reset mechanism

Clock name	Comments
clk_gcc_qusb2_phy_clk “phy_reset_clk”	Reset signal resets all the PHY blocks. The reset is a one-time activity that is performed during USB initialization.
clk_gcc_usb2_hs_phy_only_clk “phy_por_clk”	The PHY POR pin is used to reset PHY POR after: ▪ Programming the ULPI override registers ▪ Coming out of LPM when PHY is taken out of retention

USB controller register	USB register bit field	Reset control	Comments
USBCMD	RST [Bit 1]	USB controller	▪ Resets the entire USB controller ▪ Required to redo controller initialization

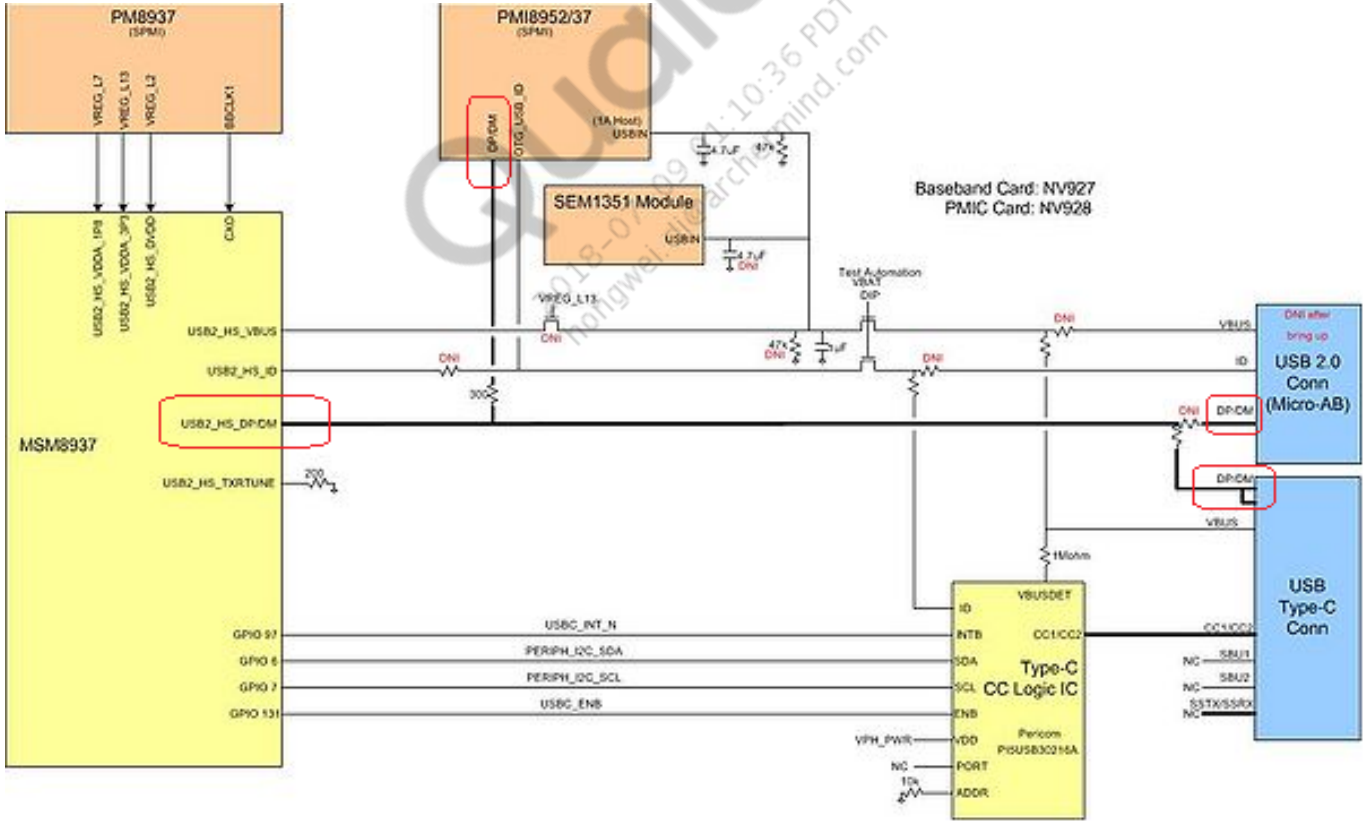
MSM8937/MSM8940/MSM8920 USB VBUS/ID

- Reference designs use the PMI8937 to handle VBUS/ID pin detection
- After the PMI8937 driver receives a USBIN valid interrupt and detects a charger, the driver notifies the USB driver of the connection event through power supply APIs



MSM8937/MSM8940/MSM8920 USB Charging

- Reference designs use the PMI8952 to handle USB charger detection.
- Once the PMI8952 completes detection, the charger type is passed to the USB driver using the power supply APIs.



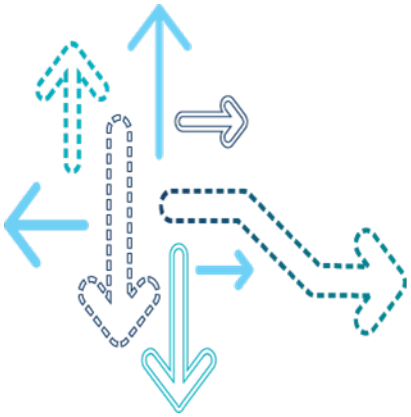
MSM8937/MSM8940/MSM8920 USB Interrupts

- Supports Type-C I2C interrupt (for type-C high current mode events) through GPIO
- Depends on I2C Int GPIO for Type-C charging modes
- I2C interrupt GPIO configuration through dtsi
- interrupts = <97 2>; /* MSM GPIO 97, TRIGGER_FALLING */
- Uses existing power supply framework to propagate type-C charging modes from type-C driver to USB driver
- File path – arch/arm/boot/dts/qcom/msm8937.dtsi

Interrupt name	Interrupt number	USB interrupts	Comments
async_irq	140	▪ USB Line state changes ▪ DP/DM changes ▪ Charger detection events	▪ When PHY is in LPM ▪ async_irq is used only when the USB controller is suspended
core_irq	134	▪ Bus events ▪ Suspend ▪ Resume ▪ Reset ▪ Controller event completion ▪ Transfer completion	Main USB interrupt that handles all USB controller events

Note: For more details, see *Linux USB Implementation Guide* (80-NF283-1).

MSM8937/MSM8940/MSM8920 USB Software Overview



MSM8937/MSM8940/MSM8920 USB Use Case Examples

- Software loading – Emergency download (PBL) (QFIL)
- Debugging tools
  - RAM dump collection (SBL)
  - ADB, Diag, and QDSS
- Device storage – MTP, PTP, and mass storage
- Device tethering – RNDIS and RmNet
- USB charging
- Host mode – Detection of hubs, HIDs, and mass storage devices

Linux USB Features

Feature	MSM8937	MSM8940/MSM8920	Comments
Peripheral ADB	Supported	Supported	–
Peripheral mass storage	Supported	Supported	–
Peripheral Diag	Supported	Supported	–
Peripheral DUN	Supported	Supported	–
Peripheral QMI RmNet	Supports Cat 4 speeds	Supports Cat6 Speeds	–
Peripheral RNDIS	Supports Cat 4 speeds	Supported Cat6 Speeds	–
Peripheral MBIM	Not supported	Not supported	–
Peripheral ECM	Not supported	Not supported	Not officially supported; however, OEMs can enable (throughput results differ)
Peripheral MTP	Supported	Supported	–
Android Open Accessory	Supported	Supported	Supported from Ice Cream Sandwich + 3.0 kernel
Peripheral Digital Audio	Not supported	Not supported	–
Peripheral HSIC	Not supported	Not supported	–
QDSS	Supported	Supported	–

Linux USB Features (cont.)

Feature	MSM8937/MSM8940/ MSM8920	Comments
Host HS USB	Supported	–
Host HSIC	Not supported	–
Host HID/MS/hub driver	Supported	–
Host digital audio driver	Supported	–
Charging Downstream Port (CDP)	Supported	–
Accessory Charger Adapter (ACA)	Not supported	Discontinued due to low market adaption
USB host PC charging (SDP)	Supported	BC1.2 charger detection
OTG	Not supported	Dual role device support possible (Host/Device mode support); no support for SRP/HNP
USB wall charging (DCP)	Supported	BC1.2 charger detection
Quick Charge 2.0/3.0	Supported	Depends on whether PMI8952 is used
USB Type C	Supported	Third-party USB type-C solution can be used to support the USB type-C; see *USB Type-C Integration Guide with Qualcomm Chipset* (80-VP447-26)

USB Kernel Configurations

- Required kernel configurations

Configuration	Enabled (Y/N)	Comments
CONFIG_USB	Y	Main USB kernel configuration
CONFIG_USB_PHY	Y	Enables the USB PHY framework (required if using separate PHY drivers)
CONFIG_USB_CI13XXX_MSM	Y	Enables the CI13XXX_MSM controller driver
CONFIG_USB_G_ANDROID	Y	Enables the Android gadget driver
CONFIG_USB_GADGET	Y	Enables USB gadget support (required for CONFIG_USB_G_ANDROID)
CONFIG_USB_EXT_TYPE_C_PERICOM	Y	Enables USB type-C support with external PERICOM solution

USB Kernel Configurations (cont.)

- Optional kernel configurations

Configuration	Enabled (Y/N)	Comments
CONFIG_USB_ANNOUNCE_NEW_DEVICES	Y	USB detects and announces the new devices in Host mode support
CONFIG_USB_GADGET_DEBUG_FILES	Y	Enables added debug files (normally in the UDC layer)
CONFIG_USB_GADGET_DEBUG_FS	Y	Enables debug files when debugfs is mounted
CONFIG_USB_EHCI_MSM	Y	Enables the EHCI host controller support
CONFIG_USB_EHCI_HCD	Y	Enables the EHCI host stack

USB Composition

- USB default composition – 0x9091 - DIAG + DUN + RmNet + ADB

Interface	Use case
DIAG	Diagnostics data for QXDM Professional™ (QXDM Pro)/QPST
DUN	Dial-up networking and AT commands
RmNet	RmNet tethering
ADB	Android Debug Bridge

Note: All available compositions are contained within init.qcom.usb.rc. See *Linux USB Implementation Guide* (80-NF283-1) for more information.

MSM8937/MSM8920 USB Power Management

- Main USB controller drive

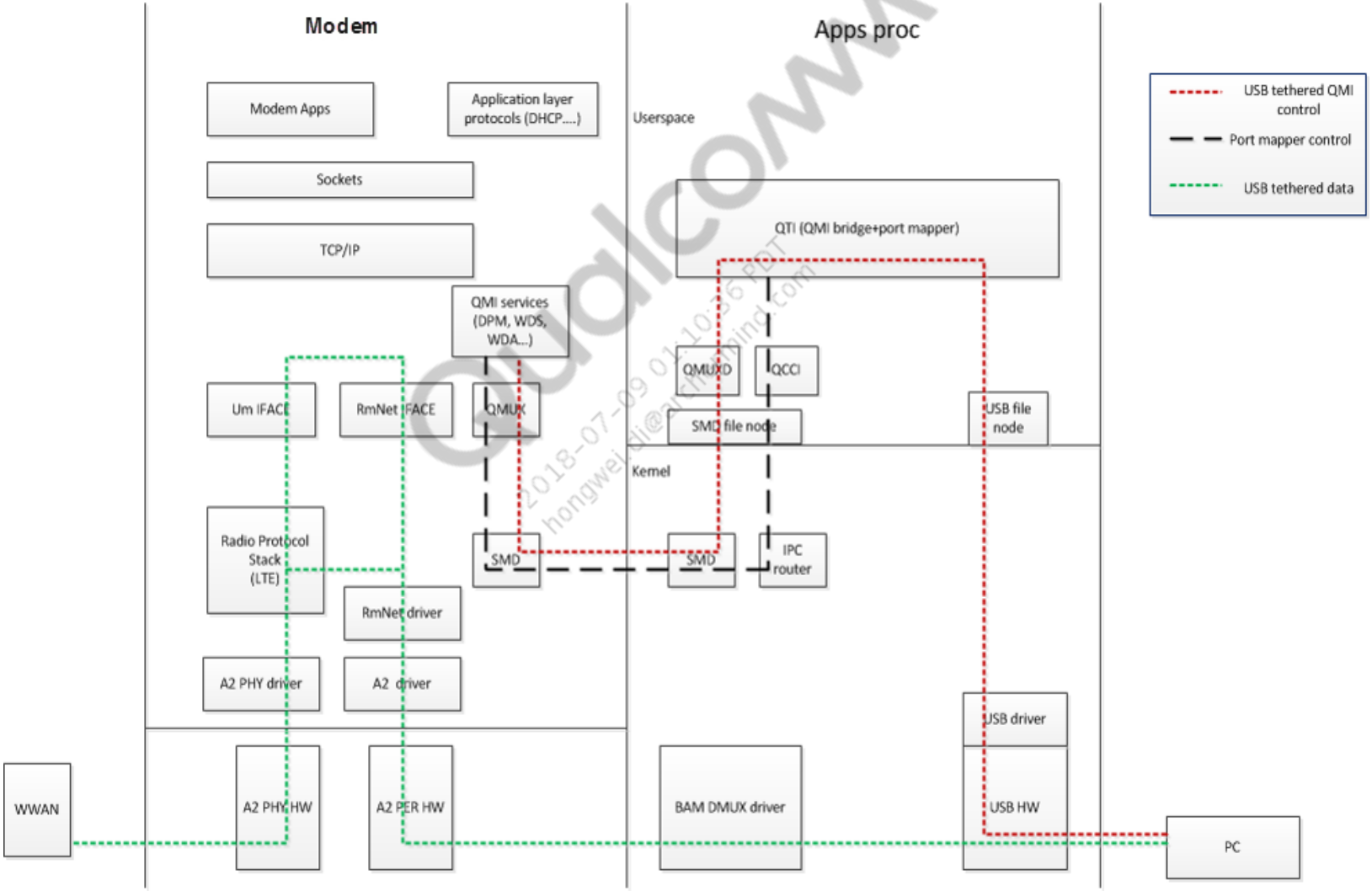
Operating mode	Conditions	Power collapse (HS PHY)	XO shutdown	XO shutdown + VDD minimization
Device	USB bus suspend	X	X	X
	USB disconnected	✓	✓	✓
	USB DCP/HVDCP charger connected	✓	✓	✓

Note: For detailed interrupt handling, see the section USB Power Management in *Linux USB Implementation Guide* (80-NF283-1).

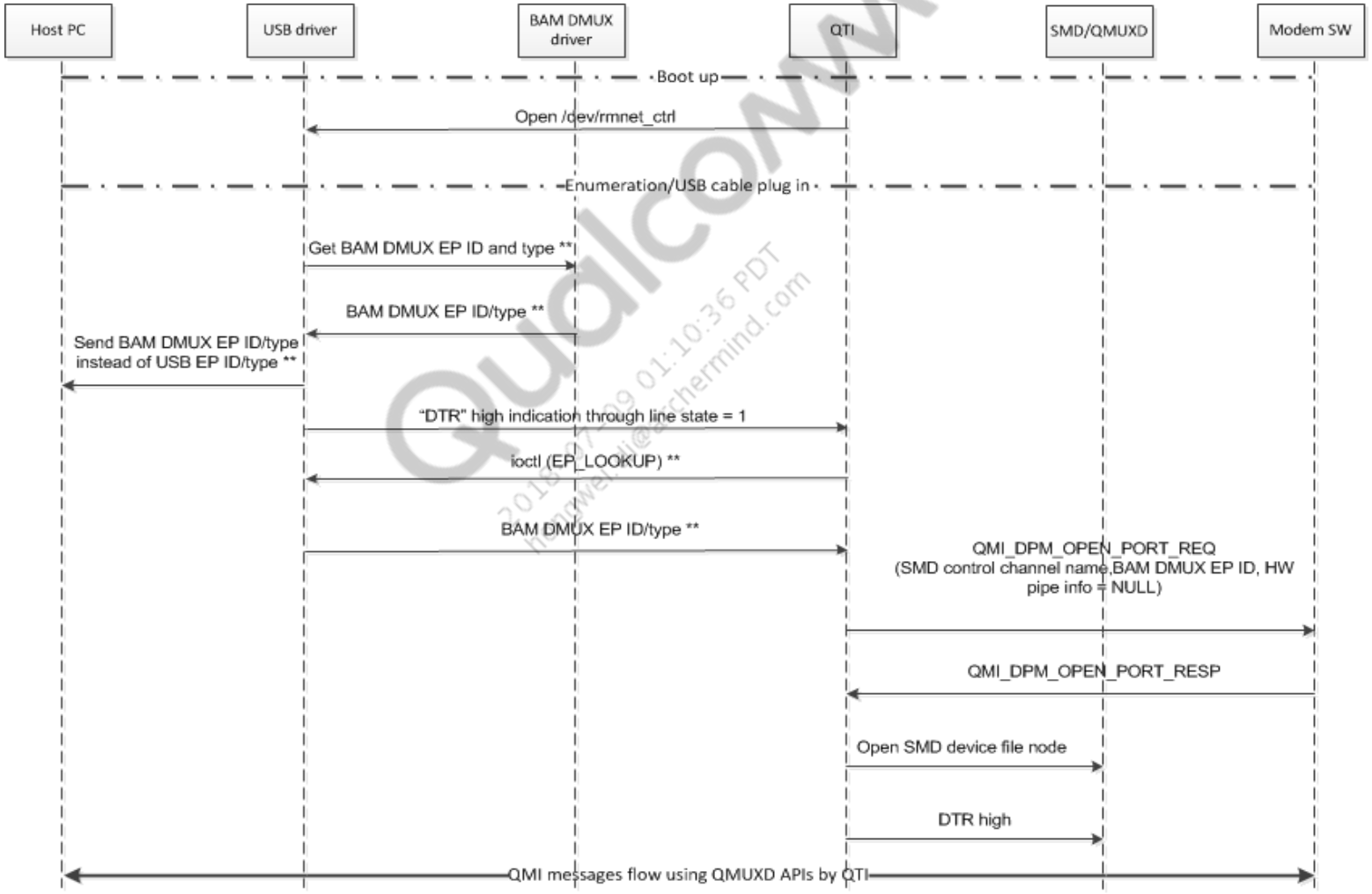
Software Change Items

- A new PHY type is introduced to make Femto PHY-specific changes in the USB driver.
- For MSM8937 RNDIS and RmNet functions use the software path.
- For MSM8937 The USB RmNet function control path is changed to QTI (user space) from SMD transport.
- MSM8940/MSM8920 use IPA/BAM support for data tethering interfaces
  - RNDIS and RmNet only

MSM8937 USB RmNet Tethering Architecture



MSM8937 USB RmNet Tethering Call Flow



MSM8937/MSM8940/MSM8920 USB Function Driver Changes for 64-bit

- Minimal changes are required to support 64-bit architecture:
  - Introduction of compat_ioctl() callbacks for any function driver interacting with user space
  - Changing variable types to avoid compilation errors
- 32-bit user space calling into a 64-bit kernel requires the compat_ioctl callback.

MSM8937/MSM8940/MSM8920 USB Compat_Ioctl()

- Drivers affected
  - F_mtp
  - F_accessory
    - Both drivers expose a device node where the user space can make IOCTL calls to the driver.
- F_mtp
  - Requires to convert compat values passed from user space into pre-existing MTP event/data structures
    - The MTP driver uses an event structure that defines various kinds of events that occurred on the application side.
    - The MTP driver uses a data structure that defines any transfer-related parameters, such as length, read/write command, etc.
  - Introduces a new compat_ioctl call, which handles this translation

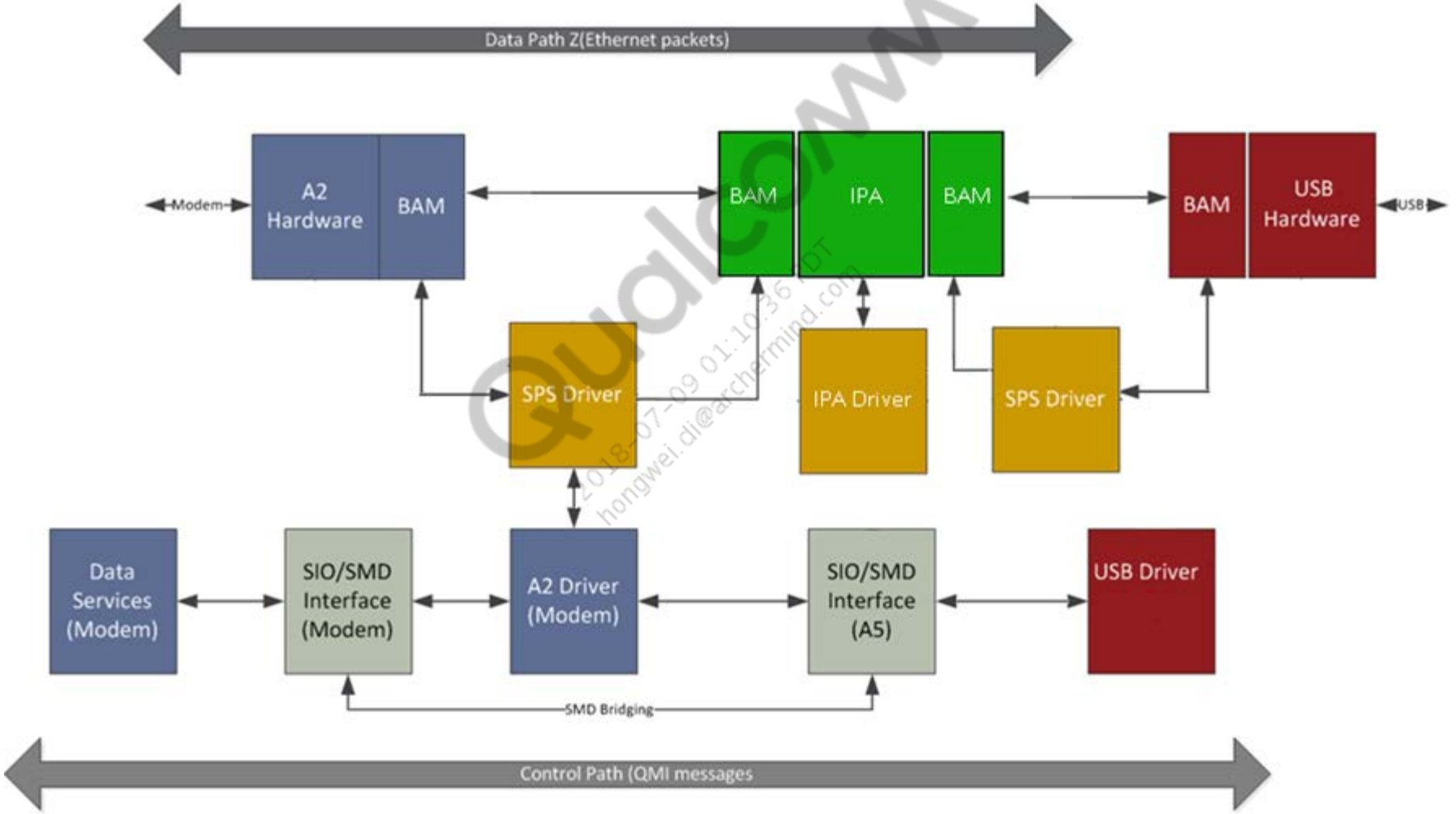
MSM8937/MSM8940/MSM8920 Variable Types

- Data types

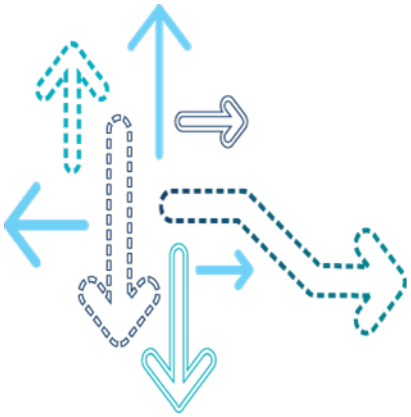
Type	ILP32 size and alignment (bytes)	LP64 size and alignment	Printf length modifier
char	1	1	hh
short	2	2	h
int	4	4	–
long	4	8	l
size_t	4	8	z
pointer	4	8	p
long long	8 (alignment varies)	8	ll

- In kernel code, uses size-specific types where appropriate
  - In driver implementation – u32, u64, etc.
  - In User mode interface – _u32, _u64, etc.
- Different alignment requirements could change struct sizes in nonobvious ways
  - LP64 requires more padding before long, size_t, or pointer members
  - Use naturally aligned structs if possible
  - Alignment of long is different on 32-bit ARM (8 bytes) and 32-bit x86 (4 bytes)

MSM8940/MSM8920 USB Tethering with IPA



MSM8937/MSM8940/MSM8920 USB Debugging



MSM8937/MSM8940/MSM8920 USB Debug Details

- Enable USB debug logging through debugfs
  - Device mode
    - Dynamic UDC debug
    - `adb shell mount -t debugfs none /sys/kernel/debug`

      ```
      Echo 'file ci13xxx_msm.c +p' > /sys/kernel/debug/dynamic_debug/control
      Echo 'file phy-msm-usb.c +p' > /sys/kernel/debug/dynamic_debug/control
      Echo 'file ci13xxx_udc.c +p' > /sys/kernel/debug/dynamic_debug/control
      ```

    - Kernel UDC debug configurations
      - CONFIG_USB_GADGET_DEBUG
    - Function driver dynamic debug (depending on the interface being debugged)

      ```
      Echo 'file f_rmnet.c +p' > /sys/kernel/debug/dynamic_debug/control
      Echo 'file u_bam.c +p' > /sys/kernel/debug/dynamic_debug/control
      ```

  - Host mode
    - Dynamic HCD debug

      ```
      Echo 'file hub.c +p' > /sys/kernel/debug/dynamic_debug/control
      ```

Note: For more details, see *Linux USB Implementation Guide* (80-NF283-1).

MSM8937/MSM8940/MSM8920 USB Debug Details (cont.)

- HSUSB debugfs entries
  - Sysfs directory
    - /sys/bus/platform/devices/78d9000.usb
    - /sys/devices/platform/msm_hsusb/gadget

Filename	Read/Write	Comments
events	R	Outputs the HSUSB static debug buffer, capturing certain USB events (default log mask is set to log EP0 and bus events)
inters	R	Outputs interrupt statistics for each endpoint
device	R	Read – Outputs current USB operating mode (Host or Device mode)
qheads	RW	▪ Read – Outputs current pending USB requests for an endpoint ▪ Write ▪ Sets the required USB endpoint ▪ echo num dir > sys/devices/platform/msm_hsusb/gadget/qheads (num = 0..7, dir = 1 (IN), 0 (OUT))
registers	R	Dumps all HSUSB controller registers
requests	RW	▪ Read – Outputs all allocated USB requests for an endpoint ▪ Write ▪ Sets the required USB endpoint ▪ echo num dir > sys/devices/platform/msm_hsusb/gadget/requests (num = 0..7, dir = 1 (IN), 0 (OUT))

MSM8937/MSM8940/MSM8920 USB Debug Details (cont.)

- HSUSB debugfs entries
  - Debugfs entry – /sys/kernel/debug/msm_otg

Filename	Read/Write	Comments
Bus_voting	RW	▪ Read – Outputs the current USB bus voting status ▪ Write – Enables/Disables bus voting. Example: echo enable/disable > /sys/kernel/debug/msm_otg/bus_voting
chg_type	R	Read – Outputs the current charger type connected to device
otg_state	R	Read – Outputs the current state of the OTG state-machine; OTG state is specified in Peripheral/Host mode

- By default debug buffer logging is enabled
  - To disable debug buffer logging

    ```
    echo 0 > /sys/module/phy_msm_usb/parameters/enable_dbg_log
    ```

  - To enable debug buffer logging

    ```
    echo 1 > /sys/module/phy_msm_usb/parameters/enable_dbg_log
    ```

MSM8937/MSM8940/MSM8920 USB Debug Details (cont.)

- Debug buffer logs

```
v.v motg = 0xEDB20000 -> (
phy = (dev = 0xEE311A10, label = 0x0 -> NULL, flags = 1, type = USB_PHY_TYPE
pdata = 0xEDF1B810,
irq = 166,
async_irq = 172,
phy_irq = 168,
.......
phy_irq_pending = FALSE,
dbg_idx = 189,
dbg_lock = (raw_lock = (lock = 0), magic = 0, owner_cpu = 4294967295, owner
buf = (
[0] = "[    3.044464]: BUS VOTE :2:0",
[1] = "[    3.047344]: OTG BUS FREQ SET :0:400000000",
[2] = "[    3.047370]: OTG BUS FREQ SET :1:200000000",
[3] = "[    3.047395]: OTG BUS FREQ SET :2:100000000",
[4] = "[    3.051583]: CONFIG VDDCX :1200000:1200000",
[5] = "[    3.052390]: USB REG MODE :1:0",
[6] = "[    3.053541]: OTG PROBE :90:0",
[7] = "[    3.053571]: RUNTIME IDLE :0:0",
```

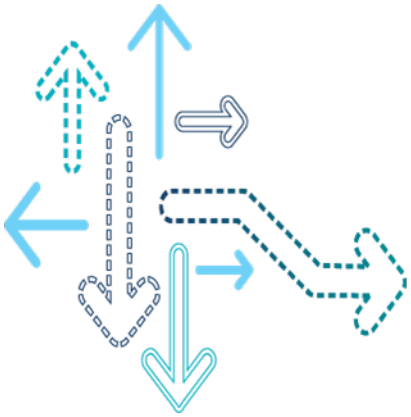

MSM8937/MSM8940/MSM8920 USB Debug Details (cont.)

- Android_usb main entries
  - Sysfs directory – /sys/class/android_usb/android0

Filename	Read/Write	Comments
functions	RW	▪ Read – Outputs the current USB composition ▪ Write – Stores the new USB composition
iManufacturer	RW	▪ Read – Outputs the current manufacturer string ▪ Write – Stores a new manufacturer string
iProduct	RW	▪ Read – Outputs the current product string value ▪ Write – Stores a new product string value
iSerial	RW	▪ Read – Outputs the current serial number value ▪ Write – Stores a new serial number value
idProduct	RW	▪ Read – Outputs the current PID value ▪ Write – Stores a new PID value
idVendor	RW	▪ Read – Outputs the current VID value ▪ Write – Stores a new VID value

Note: For detailed USB composition design, see the section USB Composition Design of *Linux USB Implementation Guide* (80-NF283-1).

USB KPIs



USB KPIs

- Mass storage performance numbers
- Test setup
  - Secondary boot image with performance enabled
  - Performance setting – Scaling governor = Performance, all cores set to online
  - SD card used – SanDisk 4 GB Class 6
  - Host PC – Lab4119, Model – HPZ220, OS – Windows 7 64-bit, USB controller – Intel 8 Series/C220 series USB EHCI
  - Miscellaneous settings – Dirty bit ratio is 20

Test case	Performance numbers
1 GB file transfer test – On to SD card (using stopwatch)	▪ Read – 20.48 MBps ▪ Write – 14.62 MBps
Crystal Disk Bench Mark tool	▪ Read – 21.89 MBps ▪ Write – 16.50 MBps

Note: The USB throughput numbers are provided on the reference hardware with the HSUSB controller. The exact throughput numbers on MSM8937/MSM8940 were not available when this document was being published.

USB KPIs (cont.)

- USB Media Transfer Protocol (MTP) performance numbers
- Test setup
  - Secondary boot image with performance enabled
  - Performance setting – Scaling Governor = Performance, all cores set to online
  - SD card used – SanDisk 4 GB Class 6
  - Host PC – Lab4119, Model – HPZ220, OS – Windows 7 64-bit, USB controller – Intel 8 Series/C220 Series USB EHCI
  - Miscellaneous settings – Dirty bit ratio is 20

Test case	Performance numbers	Comments
1 GB file transfer test – On to internal storage eMMC card (using stopwatch)	▪ Read – 26.94 MBps ▪ Write – 20.48 MBps	▪ Pass ▪ CPU usage – 4%
5 MB small files of 1 GB to eMMC	▪ Read – 22.02 MBps ▪ Write – 13.76 MBps	▪ Pass ▪ CPU usage – 4%
1 GB file transfer test – On to internal storage SD card (using stopwatch)	▪ Read – 21.33 MBps ▪ Write – 14.62 MBps	▪ Pass ▪ CPU usage – 4%

Note: The USB throughput numbers are provided on the reference hardware with the HSUSB controller. The exact throughput numbers on MSM8937/MSM8940 were not available when this document was being published.

Type-C Cable Recommendations

- To connect to legacy host machines (that do not support USB type-C), USB type C to standard A cables are required. The type C hosts provide 500/900 mA, 1500 mA, and 3000 mA by applying appropriate pull-up resistance on the CC line. Legacy host machines supply a maximum of 500 or 900 mA depending on the speed of communication. The USB type-C specification to standard A cables should have only 56 kΩ pull-up resistance, so that the devices are connected to the legacy host PC and can draw 500 mA current after the USB is configured. There are some faulty cables available in the market, which have 10 kΩ pull-up resistance on the CC line. These cables violate the type C specification.

Note: Following are the recommended values from type-C specification.

DFP advertisement	Current source to 1.7–5.5 V	Resistor pull-up to 4.75–55 V	Resistor pull-up to 3.3 V ± 5%
Default USB power	80 µA ± 20%	56 KΩ ± 20%	36 KΩ ± 20%
1.5 A at 5 V	180 µA ± 8%	22 KΩ ± 5%	12 KΩ ± 5%
3.0 A at 5 V	330 µA ± 8%	10 KΩ ± 5%	4.7 KΩ ± 5%

Type-C Cable Recommendations (cont.)

- Following are the issues observed with non-complaint type-C cables:
  - Device can incorrectly detect the power as 3 A and lead to a power surge issue.
  - A faulty type-C could cause USB ID pin toggling, in case it is connected to the wall charger that is not connected to power socket.
- To avoid these issues, it is recommended that OEMs use good cables that are compliant to USB type-C specification.
- The following links provide information on the cables that are validated, tested, and confirmed as compliant:
  - http://www.amazon.com/dp/B00VIWE1ZY
  - http://www.amazon.com/dp/B0119EIHTG

Add Floating Charger Support

- The code change attempts to detect the charger twice. After PMIC is complete and notifies the USB driver. The USB driver notifies the B Session Valid (BSV) event, and the OTG State Machine triggers the USB PHY charger detection as floated charger detection depends on the dtsi parameters enabled.
- To enable floated charger support, add the following changes. https://www.codeaurora.org/patches/quic/la/PATCH_146909_Addfloatingchargersupportfornewpla_20160405.tar.gz

```
--- a/arch/arm/boot/dts/qcom/msm8937.dtsi
+++ b/arch/arm/boot/dts/qcom/msm8937.dtsi
@@ -1032,6 +1032,7 @@
qcom,phy-dvdd-always-on;
qcom,boost-sysclk-with-streaming;
qcom,axi-prefetch-enable;
+ qcom,floated-charger-enable = <1>;
qcom,msm-bus,name = "usb2";
qcom,msm-bus,num-cases = <3>;
```


References

Title	Number
Qualcomm Technologies, Inc.	
Linux USB Implementation Guide	80-NF283-1
USB Type-C Integration Guide with Qualcomm Chipset	80-VP447-26

Acronym or term	Definition
Term	**Definition**
ACA	Accessory Charger Adapter
CDP	Charging Downstream Port
DCP	Dedicated Charging Port
LPM	Low Power Mode

Questions?

https://createpoint.qti.qualcomm.com