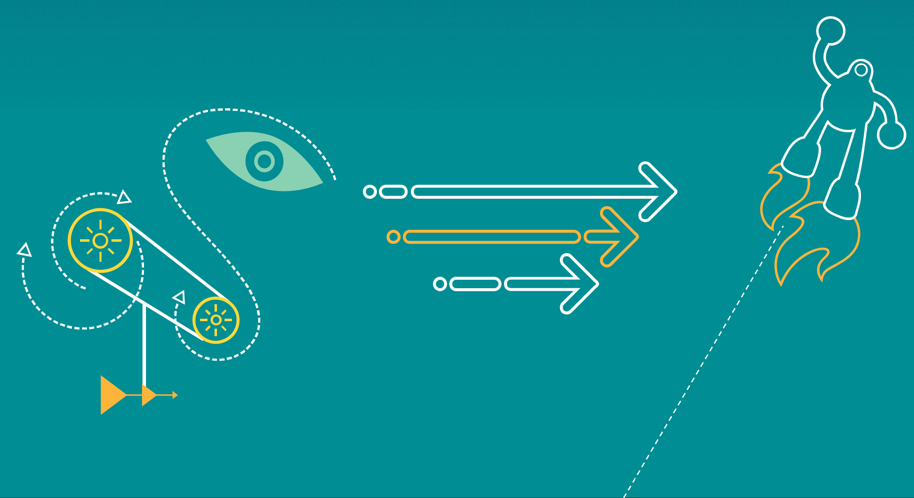
Modem and Core Power Debugging Overview

Qualcomm Technologies, Inc.



80-NR964-28 A

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

Revision History

Revision	Date	Description
A	Dec 2014	Initial release

Contents

- Documents
- Power Debugging Preparation
- Power Dashboard
- Setup for Power Measurements
- Logs Available for Power Debugging
- Components Impacting Sleep and Modem Power Use Cases
- Power Debugging – Sleep
- Power Debugging – Standby
- Power Debugging – Talk and Data
- Summary
- References
- Questions

Objectives

- This presentation covers the following topics:
  - Available documents
  - Power debug preparation
  - Power dashboard
  - Setup for power measurements
  - Logs available for power debugging
  - Components impacting sleep and modem power use cases
  - Power debugging
    - Sleep
    - Standby
    - Talk and data

Documents

Documents

- Download Qualcomm Technologies, Inc. (QTI) power document sets from the Documents and Downloads support website at https://support.cdmatech.com/.
  - Library/<Chipset>/SW/<Chipset> Linux/<Chipset> Linux SW User Guide/ Power_Thermal
  - Library/<Chipset>/SW/AMSS <Chipset>/AMSS Chipset SW User Guide/ Power_Thermal

Documents (cont.)

Title	MSM8909	MSM8916	MSM8994	MSM8939
Linux Android Current Consumption Data	80-NP408-7	80-NK807-7	80-NJ051-7	80-NM683-7
System Power Overview	80-NR964-5	80-NL239-6	80-NM328-15	80-NM846-2
Power Consumption Measurement Procedure for MSM™ (Android-Based)/MDM Devices	80-N6837-1	80-N6837-1	80-N6837-1	80-N6837-1
*Power Consumption Optimization and Debugging**	—	80-NL239-48*	80-NM328-98*	—
*Release Notes***	—	80-NM886-x 80-NM872-x	80-NN696-x	80-NP191-x 80-NP192-x
Modem/Multimedia Use Case Data Flows	—	80-NL239-60	80-NM328-704	—

***** Refer to this document for more details on power debugging related to sleep, standby, talk, and data use cases.

** The *x* in the release notes DCN changes with every software release; search on the numbers before the *x*.

Documents (cont.)

- QTI documents
  - Linux Android Current Consumption Data
    - Power tree
    - Projections – CS power goals
    - Measurements – Detailed breakdown of dashboard use cases
  - System Power Overview
    - Power system and features for the chipset
  - Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices
    - Setup and measurement procedure for power dashboard use cases
  - Power Consumption Optimization and Debugging
    - Log collection and debugging techniques
  - Release Notes
    - Software release notes that include measured dashboard battery level numbers
  - Modem/Multimedia Use Case Data Flow
    - Data flow and clock plan for dashboard use cases

Power Debugging Preparation

Power Debugging Preparation

- Before engaging in power debugging, QTI recommends using the following tools:
  - JTAG setup
    - PMIC, clocks, and GPIO dumps to debug modem dashboard power use cases requires JTAG setup
  - Power monitor with good sampling rate
    - Power monitoring tool to accurately measure use case power consumption and for waveform analysis
    - Power monitor with 5 kHz or higher sampling rates
  - Breakdown board
    - Build a special board to measure the current consumption at rail level for major rails, if not for all rails
    - SMPS and LDO input/output measurement capability for current and voltage at rail level breakdown is required for power debugging
    - Power grid for a particular chipset explains the rails available in the system

Power Dashboard

Power Dashboard

- Dashboard is a set of standard use cases published by QTI.
- Dashboard battery numbers are published in *Release Notes* for major software releases.
- Dashboard breakdowns are published in *Linux Android Current Consumption Data*; see References for the applicable DCN.
- QTI recommends customers measure dashboard use cases on devices under development and optimize by comparing it to QTI published data.
- QTI strongly recommends customers follow *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1) to ensure test conditions and measurements are correct.

Power Dashboard (cont.)

Test case	Code
Airplane mode	AIR1
WCDMA standby, 2.56 sec	WS1
CDMA QPCH 5.12 sec	CS2
GSM standby 1.18 sec	GS1
LTE standby 2.56 sec	LS1
LTE TDD standby 2.56 sec	LS3
LTE FDD CDRX standby (10 MHz, 320 ms)	CDRXS1
LTE TDD CDRX standby (20 MHz, 320 ms)	CDRXS4
TD-SCDMA standby 1.28 sec	TCS1
DSDS/DSDA G+G, 0.47 sec + 0.47 sec	GGS2
DSDS/DSDA W+G: 0.64 sec + 0.47 sec	WGS3
CDMA talk + 0 dBm	CT1
WCDMA talk + 0 dBm, IMT	WT1
GSM talk 5 dBm, no DTX, PGSM	GT1
VoLTE Talk 0 dBm 50% DTx, 40 ms CDRx, SPS	VoLTE1
TD-SCDMA talk 0 dBm, B34	TCT1
EV-DO DL 3.1 Mbps + 0 dBm, CEL, RxD OFF	DD2E

Power Dashboard (cont.)

Test case	Code
HSDPA DL 7.2 Mbps + 0 dBm, IMT (RxD/no RxD)	HS22E/HS21E
HSDPA DC 42 Mbps + 0 dBm, IMT RxD	HS62E
LTE Cat 3 (68/23 Mbps, +0 dBm, B13)	LTE1E
LTE Cat 3 (100/50 Mbps, 0 dBm, B7)	LTE6E
LTE Cat 3 2xCA 10+10 (100/25 Mbps, 0 dBm, B4+B17, Tx B17)	LTE8E
LTE Cat 4 (150/50 Mbps, 0 dBm, B7)	LTE7E
LTE FDD Cat 6 2xCA max DL+UL in 20+20 MHz, 0 dBm Tx (300/50 Mbps), B3+B7, TX B3	LTE10E
LTE cat6, 3xCA, 20MHz+20MHz+20MHz (300 Mbps DL + 50 Mbps UL, B1+B3+B7, TX B3), 0 dBm	LTE12E
LTE TDD Cat 3 20 MHz (60/18 Mbps, + 0 dbm, B38)	LTE5E
GPS 1 Hz Trk (DPO) with WCDMA standby	GPS2
GNSS 1 Hz Trk high sensitivity with WCDMA standby	GNSS1
MP3 playback 128 kbps TM	AU4A
Listen – 100% silence	AU34A
Listen – 100% speech	AU35A
Static Image	LCD04A

Power Dashboard (cont.)

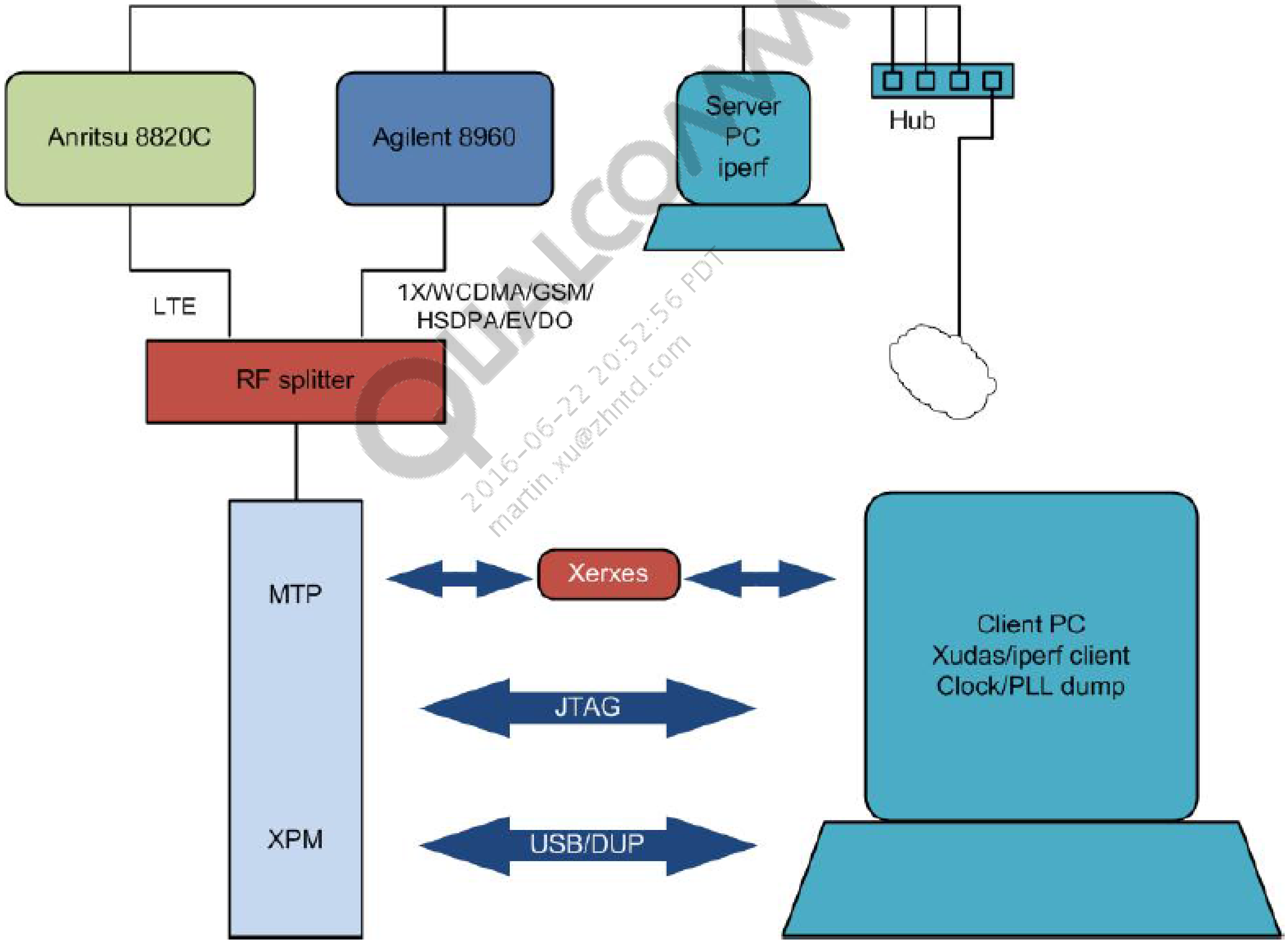
Test case	Code
H.264 720p decode, 30 fps	QTC77A
H.264 30 fps 1080p, 20 Mbps, decode	QTC88A
H.264 30 fps 1080p, 20 Mbps, encode AAC + 128 kbps 44 kHz stereo	QMC31A
3D UI 30 fps (Graphic, PowerLift)	QGC23A
3D Gaming (Egypt 2.5HD) 60 fps at 75°C	QGC26A
BT sniff/scan on WCDMA standby	BT2
WLAN DTIM1 on WCDMA standby	WLS1
WLAN DTIM5 on WCDMA standby	WLS5
Background 20 Hz buffering of accelerometer data with 1 Hz ADSP wake-up	SNS6
Accel active use case at 15 Hz	SNS4A
Browser over Wi-Fi	WB1A
Video streaming over Wi-Fi	VS6A

Setup for Power Measurements

Setup for Power Measurements

- QTI recommends customers refer to *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1) for setup and test procedure details.
- Basic Device Under Test (DUT) settings when measuring modem use case current consumption include the following but are not limited to:
  - Disable Wi-Fi and Bluetooth.
  - Disable GPS, unless measuring GPS power.
  - Disable data when measuring non-data use cases, such as talk.
  - Always disable data monitoring.
  - Use Android user/secondary boot image and disable all debug flags.
  - Disable QDSS.
  - Select the intended air interface from QXDM Professional™ (QXDM Pro) NV to ensure no IRAT activity occurs during measurements.
  - Check for correct NV setup for each use case (80-N6837-1).
  - Disable any activity due to sensors.
  - Always ensure that a device can enter VDD minimization state before starting measurements.
  - Always ensure the correct setting for Tx and Rx power when doing measurements of modem use cases.
  - Ensure Power Amplifier (PA) is operating at the right gain state for talk and data use cases.

Setup for Power Measurements (cont.)



Logs Available for Power Debugging

Logs Available for Power Debugging

- Clock dumps
  - Provides state of each clock in the system
- GPIO dumps
  - Provides state of the GPIOs
  - Useful in debugging the pad current leakages during system sleep
- PMIC dumps
  - Provides LDO/SMPS states
  - Useful in debugging all use cases and scenarios
- QXDM logs
  - Useful for air interface protocol analysis and debugging
- Node Power Architecture (NPA) logs (RPM/modem)
  - Helps understand the state of system resources, e.g., shared clocks, LDO/ SMPS, NOC, etc., and clients requesting these resources
  - Useful in subsystem debugging, e.g., MPSS, RPM, LPASS, etc.
  - Can be obtained from RAM dumps or JTAG debugging
  - Available on modem and RPM processors

Logs Available for Power Debugging (cont.)

- Universal Logs (Ulogs) (RPM/modem)
  - Ulogs on RPM provide information of states/requests to RPM by individual subsystem
  - Ulogs on the modem have information on sleep requests, sleep states entered, NPA request history, wake-up interrupt reasons, etc.
- Dmesg and Logcat logs
  - Android kernel and userspace logs; provides information on kernel and applications activity
- Ftrace logs
  - Log trace events such as work queues, interrupts, scheduler activity, low power mode activity, etc., on Android kernel side
  - Determines which processes in the system are active during a timeframe and how they impact system/power behavior
- Wakelocks/wake-up sources
  - Provides information on which resource holds the apps processor power collapse in Android
- Msm_pm_stats/lpm_stats
  - Provides time spent in various low power states by the apps processor

Logs Available for Power Debugging (cont.)

- Hansei.py for RPM RAM dump parsing
  - Hansei – RPM RAM dump parser tool found in the rpm_proc\core\bsp\rpm\scripts\hansei folder (requires Python Ver 2.7.2)
    - Usage – hansei.py [-h] --elf rpm.elf [--output path] dumpfile [dumpfile ...]
    - For example, hansei.py –elf rpm.elf -o . rpm_code_ram.bin rpm_data_ram.bin rpm_msg_ram.bin
  - Summary of the output file
    - rpm-summary.txt – General information about the health of the RPM, including the core dump state and various fault information
    - rpm-log.txt – Postprocessed RPM external log
    - rpm-rawts.txt – Same log as rpm-log.txt but with the raw timestamp
    - npa-dump.txt – Standard NPA dump format
    - ee-status.txt – Information about which subsystems (and their cores) are active or sleeping
    - reqs_by_master/* – Folder containing a file for each execution environment, detailing all current requests that EE has in place with the RPM
    - reqs_by_resource/* – Folder structure containing a folder for each of the resource types registered with the RPM server and a subfolder with a file containing all requests to each resource of that type

Note: See log collection procedures in the Power Consumption Optimization and Debugging document published for each target during the FC software release timeframe, e.g., *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1) for MSM8916.

Components Impacting Sleep and Modem Power Use Cases

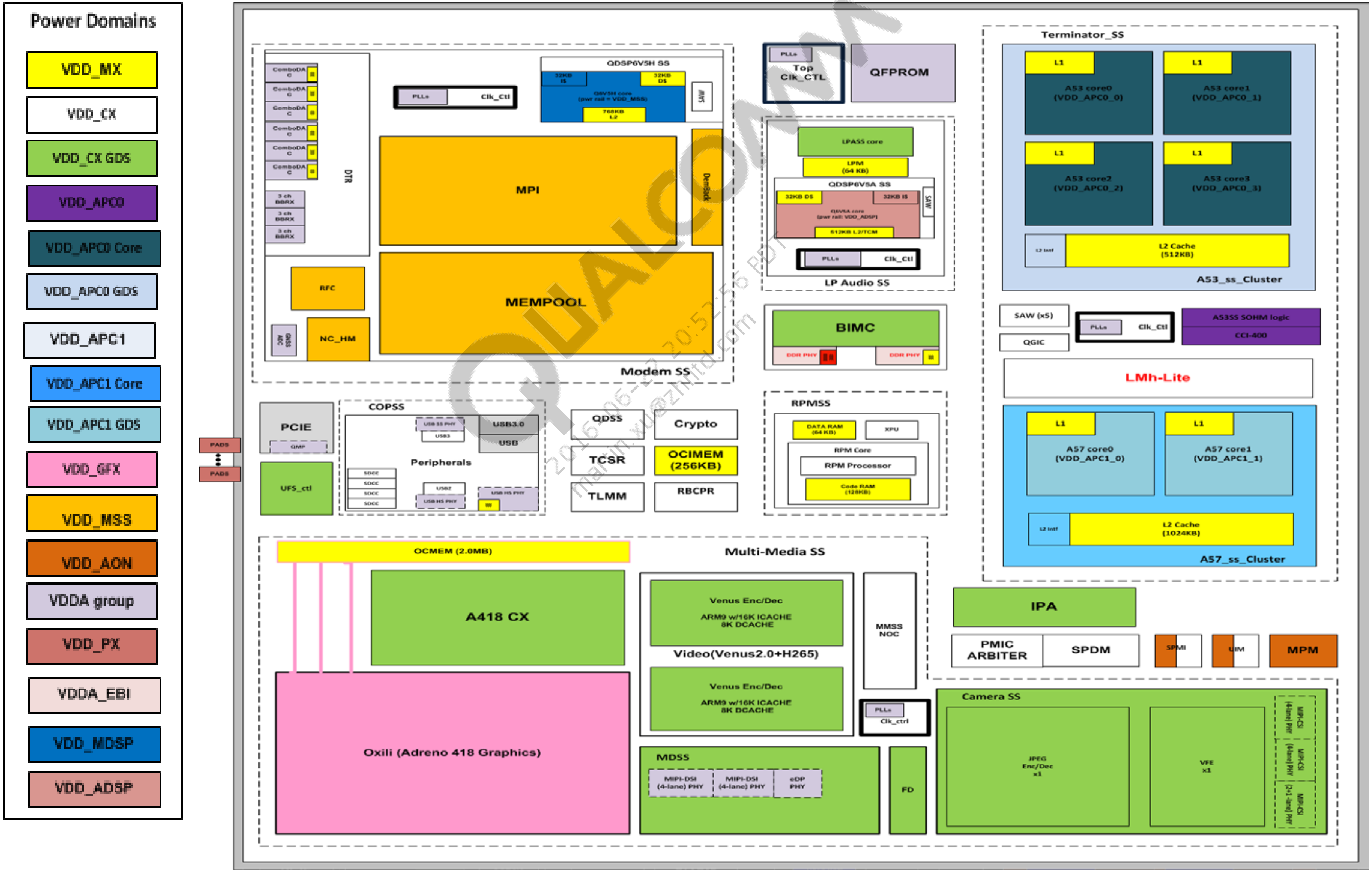
Components Impacting Sleep Power

- Most subsystems, e.g., MPSS, APPS, GPU, LPASS, and MMSS, are power collapsed during system sleep
- Domains expected to consume current during system sleep
  - VDD_CX – Core/Digital (logic portion)
  - VDD_MX – Memories (DDR, cache, etc.)
  - GPIO pads
  - PMIC internal consumption
  - DDR self-refresh consumption
- All external peripherals and components are turned off in software/ hardware or put in their lowest power state during system sleep
- Digital and memory rails (VDD_Cx and VDD_Mx) are put in their retention voltage levels

Components Impacting Modem Power

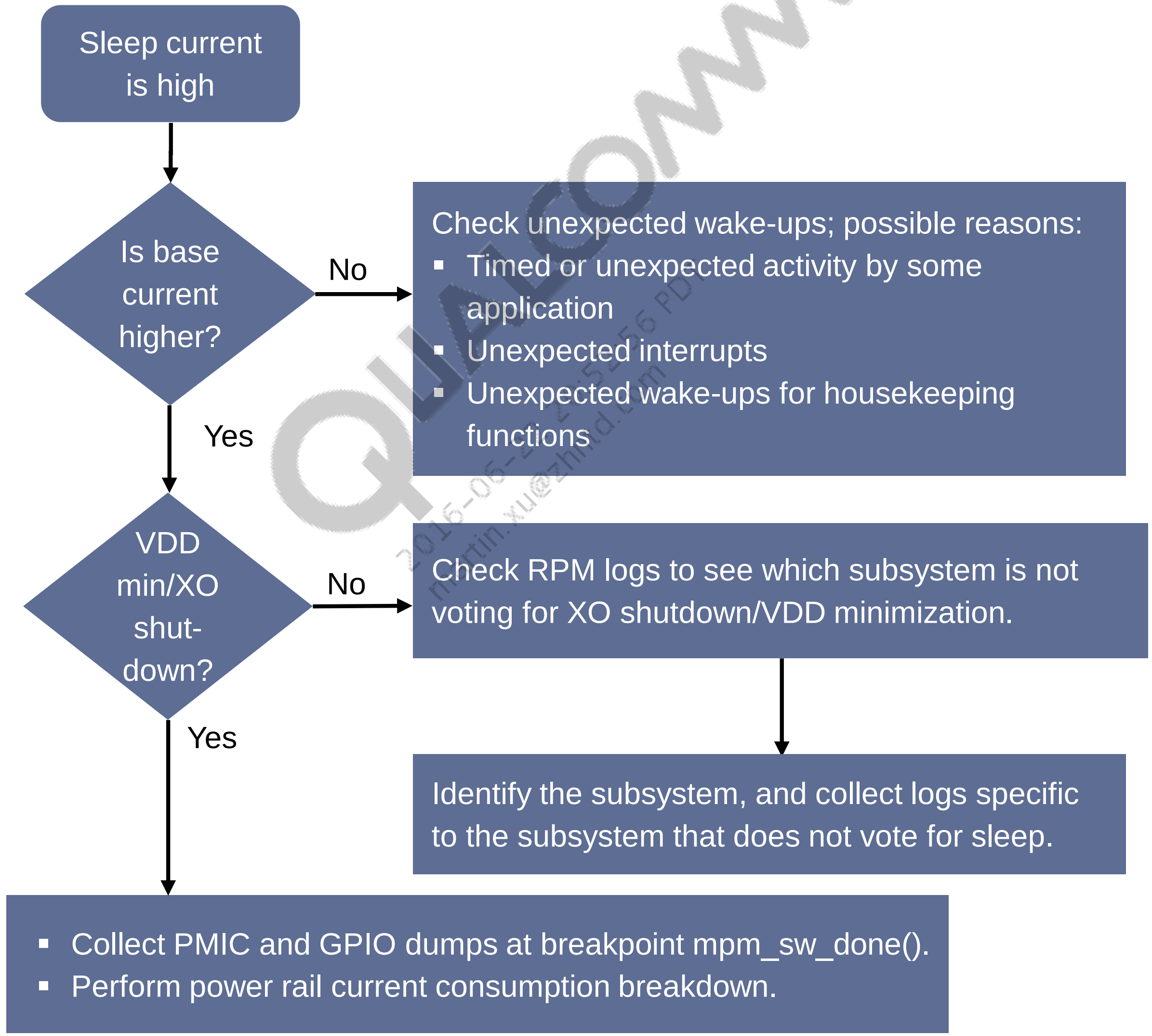
- In awake, talk, and data use cases, the following components are the major contributors of current consumption:
  - RF front end – WTR, PA
  - Hexagon-modem/audio
  - Modem hardware
  - Audio codec for talk use cases
  - DDR
  - Apps processor for data use cases
- Modem power is specifically sensitive to network conditions and configuration.
  - High UE PA Tx power can result in higher current consumption
  - Lab setup measurements are recommended before live network measurements

Power Domain for MSM8992



Power Debugging – Sleep

Power Debugging Process – Sleep



Check Subsystem Status in RPM

- Confirm the power collapse status of subsystems by checking the RPM data structure.
- After the display is off and given enough time to go to sleep:
  - Break the Trace32 (T32) execution in RPM randomly to monitor the below variables.
  - If necessary, get a crash dump using system reset by PS_Hold and analyze RPM dumps using a T32 simulator.
- Check rpm.ees[0].subsystem_status, 0→APSS information
- Check rpm.ees[1].subsystem_status, 1→MPSS information
- Check rpm.ees[2].subsystem_status, 2→LPASS information
  - If the status is SPM_SLEEPING, the subsystem can be considered in the lowest power state.
  - If the status is SPM_AWAKE, the subsystem is awake and not in the lowest power state.

Check XO Shutdown and VDD Minimization Count from the Apps Processor

- Use ADB shell commands for XO shutdown and VDD minimization count; commands to get RPM statistics:

```
mount -t debugfs none /sys/kernel/debug
cat /sys/kernel/debug/rpm_stats
```

- Output of the above commands display as follows; the count represents how many times the XO shutdown and VDD_min occur:

RPM Mode:xosd
count:0
time in last mode(msec):0
time since last mode(sec):791
actual last sleep(msec):0
client votes: 0x00020001
RPM Mode:vmin
count:28
time in last mode(msec):8000
time since last mode(sec):475
actual last sleep(msec):233000
client votes: 0x00000000

Check Why the Apps Processor Is Not In Power Collapse

- Check wake locks.

```
sleep 60 && cat /d/wakeup_sources > /data/wakelocks.txt &
```

  - If wake locks are active for a long time in the active since field of this log, check them because they could be holding xosd/vmin.
- Check frequent interrupts activity using the following ADB command:

```
sleep 20 && cat /proc/interrupts > /data/interrupt1.txt && sleep 30
&& cat /proc/interrupts > /data/interrupt2.txt &
```

- Check clocks preventing XO shutdown and VDD minimization.
  - Enable the clock debug suspend using `echo 1 > /d/clk/debug_suspend`.
  - After enabling this flag, the enabled clocks are displayed when the system goes into Suspend mode in the Dmesg logs.
  - Some of the clocks are always expected to be shown as enabled in this log. However, if any clock other than the usual major system clocks are seen to be enabled, it can be the reason for preventing power collapse.
  - Examples of clocks that must not be visible in this log
    - Peripheral clocks
    - Display-related clocks (MDSS)
    - Multimedia subsystem related clock

Check Why the Apps Processor Is Not In Power Collapse (cont.)

- Snippet of clock debug from Dmesg logs

Normal Log where System Is able to enter XO Shutdown and VDD

```
Enabled clocks:
    pnoc_a_clk:1:1 [19200000]
    bimc_a_clk:1:1 [199884800]
    bb_clk2_pin:1:1 [1000]
    bimc_msmbus_a_clk:1:1 [199884800] -> bimc_a_clk:1:1 [199884800]
    cxo_clk_src_ao:1:1 [19200000]
    mmssnoc_ahb_a_clk:1:1 [40000000]
    pnoc_keepalive_a_clk:1:1 [19200000] -> pnoc_a_clk:1:1 [19200000]

    gpll0_out_msscc:1:1 [0]
    usb_ss_phy_ldo:1:1 [0]
    gcc_boot_rom_ahb_clk:1:1 [0]
    gcc_lpass_q6_axi_clk:1:1 [0]
    gcc_mss_cfg_ahb_clk:1:1 [0]
    gcc_mss_q6_bimc_axi_clk:1:1 [0]
    gcc_usb2_hs_phy_sleep_clk:1:1 [0]
    gcc_usb30_sleep_clk:1:1 [0]

    xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_pll0:1:1 [1536000000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_pll1:1:1 [1593600000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a57_pll1:1:1 [1593600000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_lf_mux:1:1 [384000000] -> a53_lf_mux_pll0_div:1:1 [384000000] -> a53_pll0_main:1:1 [768000000]
    a53_hf_mux:1:1 [384000000] -> a53_lf_mux:1:1 [384000000] -> a53_lf_mux_pll0_div:1:1 [384000000] ->
    a53_pll0_main:1:1 [768000000] -> a53_pll0:1:1 [1536000000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_s
    a53_lf_mux_pll0_div:1:1 [384000000] -> a53_pll0_main:1:1 [768000000] -> a53_pll0:1:1 [1536000000,
    a53_clk:1:1 [384000000, 3] -> a53_hf_mux:1:1 [384000000] -> a53_lf_mux:1:1 [384000000] -> a53_lf_m
    cci_hf_mux:1:1 [499200000] -> cci_pll_main:1:1 [499200000] -> cci_pll:1:1 [998400000, 2] -> xo_ao:
    cci_clk:1:1 [249600000, 3] -> cci_hf_mux:1:1 [499200000] -> cci_pll_main:1:1 [499200000] -> cci_pl
    cci_pll:1:1 [998400000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
Enabled clock count: 27
```

Abnormal Log which shows more than 27 clocks are enabled. Major culprit here is mdp_clk_src as rest are supporting PLL's and CXO Clock. This clearly suggests the problem is happening due to MDP clock.

```
Enabled clocks:
    cxo_clk_src:1:1 [19200000]
    pnoc_a_clk:1:1 [19200000]
    bimc_a_clk:1:1 [299892736]
    bb_clk2_pin:1:1 [1000]
    bimc_msmbus_a_clk:1:1 [299892736] -> bimc_a_clk:1:1 [299892736]
    cxo_clk_src_ao:1:1 [19200000]
    mmssnoc_ahb_a_clk:1:1 [40000000]
    pnoc_keepalive_a_clk:1:1 [19200000] -> pnoc_a_clk:1:1 [19200000]
    gcc_xo:1:1 [19200000] -> cxo_clk_src:1:1 [19200000]
    gpll0:1:1 [600000000] -> gcc_xo:1:1 [19200000] -> cxo_clk_src:1:1 [19200000]
    gpll0_out_main:1:1 [600000000] -> gpll0:1:1 [600000000] -> gcc_xo:1:1 [19200000] -> cxo_clk_src:1
    gpll0_out_mmsscc:1:1 [0] -> gpll0_out_main:1:1 [600000000] -> gpll0:1:1 [600000000] -> gcc_xo:1:1
    gpll0_out_msscc:1:1 [0]
    usb_ss_phy_ldo:1:1 [0]
    gcc_boot_rom_ahb_clk:1:1 [0]
    gcc_lpass_q6_axi_clk:1:1 [0]
    gcc_mss_cfg_ahb_clk:1:1 [0]
    gcc_mss_q6_bimc_axi_clk:1:1 [0]
    gcc_usb2_hs_phy_sleep_clk:1:1 [0]
    gcc_usb30_sleep_clk:1:1 [0]
    mmsscc_gpll0:1:1 [600000000] -> gpll0_out_mmsscc:1:1 [0] -> gpll0_out_main:1:1 [600000000] -> gpl
    mdp_clk_src:1:1 [85710000, 1] -> mmsscc_gpll0:1:1 [600000000] -> gpll0_out_mmsscc:1:1 [0] -> gpll
    mdss_mdp_clk:1:1 [240000000] -> mdp_clk_src:1:1 [85710000, 1] -> mmsscc_gpll0:1:1 [600000000] ->
    xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_pll0:1:1 [1536000000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_pll1:1:1 [1593600000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a57_pll1:1:1 [1593600000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
    a53_lf_mux:1:1 [384000000] -> a53_lf_mux_pll0_div:1:1 [384000000] -> a53_pll0_main:1:1 [768000000]
    a53_hf_mux:1:1 [384000000] -> a53_lf_mux:1:1 [384000000] -> a53_lf_mux_pll0_div:1:1 [384000000] -
    a53_pll0_main:1:1 [768000000] -> a53_pll0:1:1 [1536000000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_
    a53_lf_mux_pll0_div:1:1 [384000000] -> a53_pll0_main:1:1 [768000000] -> a53_pll0:1:1 [1536000000,
    a53_clk:1:1 [384000000, 3] -> a53_hf_mux:1:1 [384000000] -> a53_lf_mux:1:1 [384000000] -> a53_lf_
    cci_hf_mux:1:1 [499200000] -> cci_pll_main:1:1 [499200000] -> cci_pll:1:1 [998400000, 2] -> xo_ao
    cci_clk:1:1 [249600000, 3] -> cci_hf_mux:1:1 [499200000] -> cci_pll_main:1:1 [499200000] -> cci_p
    cci_pll:1:1 [998400000, 2] -> xo_ao:4:4 [19200000] -> cxo_clk_src_ao:1:1 [19200000]
Enabled clock count: 35
```


Check Why the Modem Is Not In Power Collapse

- Check the NPA log that contains information about various NPA requests made by the MPSS.
- Check for subsystems that block sleep; search for the starting point of the logging of system resource states in NPA logs using the term npa_dump.
- Check the active_state of resources that can provide information about the state of the resource; check the main resources CXO, VDD_CX, VDD_MX, and CPU_VDD (modem subsystem rail), e.g.:

```
npa_resource (name: "/core/cpu/vdd") (handle: 0xD38ECA84) (units: active/off) (resource
max: 1) (active max: 0) (active state: 1 ) (active headroom: 1) (request state: 0)
npa_client (name: gps_pe) (handle: 0xD3B6E450) (resource: 0xD38ECA84) (type:
NPA_CLIENT_REQUIRED) (request: 0)
npa_client (name: gps_rx) (handle: 0xD3B6E660) (resource: 0xD38ECA84) (type:
NPA_CLIENT_REQUIRED) (request: 0)
npa_client (name: GPS_MC_CPU_VDD_CLIENT) (handle: 0xD3B12850) (resource: 0xD38ECA84)
(type: NPA_CLIENT_REQUIRED) (request: 1 )
npa_client (name: RFCA_NPA_CLIENT) (handle: 0xD3B12A60) (resource: 0xD38ECA84) (type:
NPA_CLIENT_REQUIRED) (request: 0)
npa_client (name: a2_latency_client) (handle: 0xD394F778) (resource: 0xD38ECA84) (type:
NPA_CLIENT_REQUIRED) (request: 0)
npa_client (name: mcpm_wakeup_priority_cpu_vdd_client) (handle: 0xD394D068) (resource:
0xD38ECA84) (type: NPA_CLIENT_REQUIRED) (request: 0)
npa_client (name: mcpm_cpu_vdd_client) (handle: 0xD394D0C0) (resource: 0xD38ECA84) (type:
NPA_CLIENT_REQUIRED) (request: 0)
end npa_resource (handle: 0xD38ECA84)
```


Check Why the Modem Is Not In Power Collapse (cont.)

- In the code example, the resource /core/cpu/vdd is requested by the GPS client, which means that GPS holds CPU_VDD (MPSS rail) from going into power collapse. Further analysis needs to be done from the GPS side to examine why GPS does not go into power collapse.

Check Why the Modem Is Not In Power Collapse (cont.)

- Modem ULOG – Sleep Info.Ulog is available by getting ULOGs on the modem side; this log provides some useful information
  - Selection of the Sleep state by the MPSS
  - Time the MPSS is expected to be in a Sleep state
  - Total time that MPSS was in a Sleep state
  - The interrupt that caused the MPSS to wake up
  - When the modem entered a Sleep state
  - If the Sleep state was RPM assisted

Check Why the Modem Is Not In Power Collapse (cont.)

- Snippet of Sleep Info.Ulog
  - 0x00000000AA34545E – Solver entry (cpu frequency: 576000) (hard duration: 0xb299c) (soft duration: 0x249fffff8fa) (latency budget: 0xffffffff)
    - This message specifies when the system enters the sleep solver to determine the best mode.
    - The value 0x00000000AA34545E is the timestamp of the log and is converted in seconds by using the formula Decimalof(Hex TimeStamp) / (Frequency of CXO=19200000)
      - Note: Convert hex timestamps in this log also using this formula.
  - 0x00000000AA34552E – Mode chosen: ("npa_scheduler.fork + rbcpr.disable + l2.noret + tcm.ret + cxo.shutdown + rpm.sync + crypto_nav.bcr_hm + cpu_vdd.pc_l2_noret")
    - This message typically specifies the mode chosen by the modem side.
    - This message specifies that the modem chooses to vote for XO shutdown, L2 Nonretention mode, Crypto or NAV engine low power state, and TCM to be in Retention mode.
    - rpm.sync specifies that the modem is performing an RPM-assisted power collapse.
  - 0x00000000AA34572D – Entering modes (hard deadline: 0xaa3f7dce) (backoff deadline: 0xaa3f417e) (backoff: 0x3c50) (sleep duration: 0xaea7c)
    - This message specifies that the modem is entering the mode chosen in the 0x00000000AA34552E message.
    - The hard deadline provides information on when the modem will be awakened from sleep and is a timed wake-up for the modem.
    - The backoff deadline provides information on when the entire MSM will be awakened from sleep and the modem wake-up deadline is met.
    - Sleep duration provides information on how long the modem subsystem can be in the power collapse state in case there are no extra interrupts.

Check Why the Modem Is Not In Power Collapse (cont.)

- Snippet of Sleep Info.Ulog (cont.)
  - 0x00000000AA3461AD – Sleep set sent (wakeup time requested: 0xaa419f27)
    - This message specifies that the sleep set for various shared resources is sent by the modem to the RPM.
  - 0x00000000AA41C8D6 – Exiting modes
    - This message specifies when the modem wakes up from power collapse.
  - 0x00000000AA41E1B6 – Master wakeup stats (reason: Timer) (int pending: 33) (Actual: 0xaa419db2) (Expected: 0xaa419f27)
    - This message specifies if the interrupt is pending on wake-up and the reason for wake-up is a timer interrupt (Interrupt Number 33).
    - If the actual value is much less than the expected value, then the wake-up is an unscheduled wake-up.

Check Why LPASS Is Not In Power Collapse

- Check the LPASS NPA log to determine why the LPASS does not go into power collapse.
- Check the active_state of resources that can provide information about the state of the resource, including CXO, VDD_CX and VDD_MX.
- Check client votes to determine the reason for resource not being in its low power state.

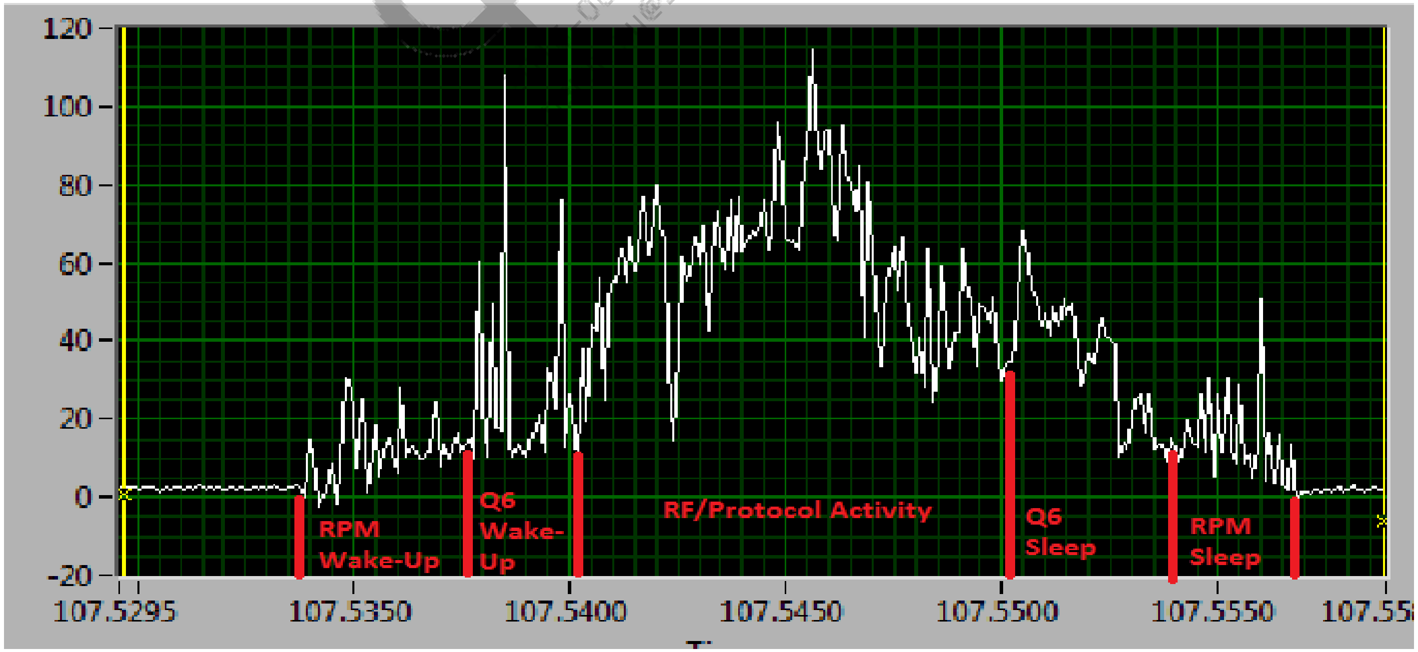
Analysis of Sleep Current Deltas and Leakages

- GPIO dumps – Conduct a GPIO dump hardware review to determine if the GPIOs are in the lowest power state possible.
- PMIC dumps – Conduct a PMIC dump review to check that there are no unknown regulators left ON during sleep.
- Power rail breakdown – Compare the DUT power rail breakdown against the QTI Reference Device (QRD). Based on the power rail differences, focus on respective power rail optimizations.
- Estimate design deltas from schematics
  - Analyze the extra components on the DUT compared to the QRD.
  - Determine the current consumption cost of these components
    - Differences between Wi-Fi or Bluetooth componentw
    - Differences such as DDR memory or NFC

Power Debugging – Standby

Standby

- The figure shows the awake penalty of paging with wake-up divided into time slices.
  - RPM Wake-up – Time taken by the RPM to wake up from sleep
  - Q6 Wake-up – Time taken by the Q6/hardware to wake-up from sleep
  - RF/Protocol Activity – Time taken by RF and protocol activities during paging
  - Q6 Sleep – Time taken by Q6/hardware to return to sleep
  - RPM Sleep – Time taken by the RPM to put the system to sleep



Standby (cont.)

- Standby current is a combination of sleep and awake current averaged over N number of cycles.
- Separate the issues by determining which part of standby causes extra current consumption.
  - High sleep current
  - Longer protocol paging awake period
  - Higher average power during paging awake
  - Extra wake-ups – Wake-ups other than paging awake

Test Setup-Related Issues

- For test setup issues, check the setup using the *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1)
- Ensure that testing is done with a callbox.
  - **Note**: QTI does not recommend testing in a live network due to higher awake current penalty and inconsistent run-to-run results.
- Disable neighbor cells in the callbox.
- Disable all data operations and browsing in the phone through the GUI.
- Ensure power levels, such as Rx cell power, Tx cell power, and AGC, used in the callbox are as recommended in *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1).
- Ensure all the debug logging is disabled through NV items as specified in *Power Consumption Measurement Procedure for MSM (Android-Based)/ MDM Devices* (80-N6837-1).

Higher Paging Awake Penalty

- Waveforms from current consumption capturing tools are very helpful in analyzing the awake penalty incurred by paging wake-ups.
- Compare waveforms of the paging awake cycle to determine which part of wake-up consumes more time or current consumption.
  - System wake-up time
  - RF wake-up
  - RF and protocol processing
  - RF sleep
  - System sleep
- Collect F3 logs by setting the appropriate configuration for log messages; F3 logs are helpful in debugging the awake penalty during paging.
- F3 logs are detailed for protocol, power, and RF with accurate timestamps to help determine what part of the system takes more time and narrows down the issue.
- In case of high-peak currents during the paging awake cycle, a rail level breakdown narrows down which rail causes the peak current consumption.

Extra Wake-Ups

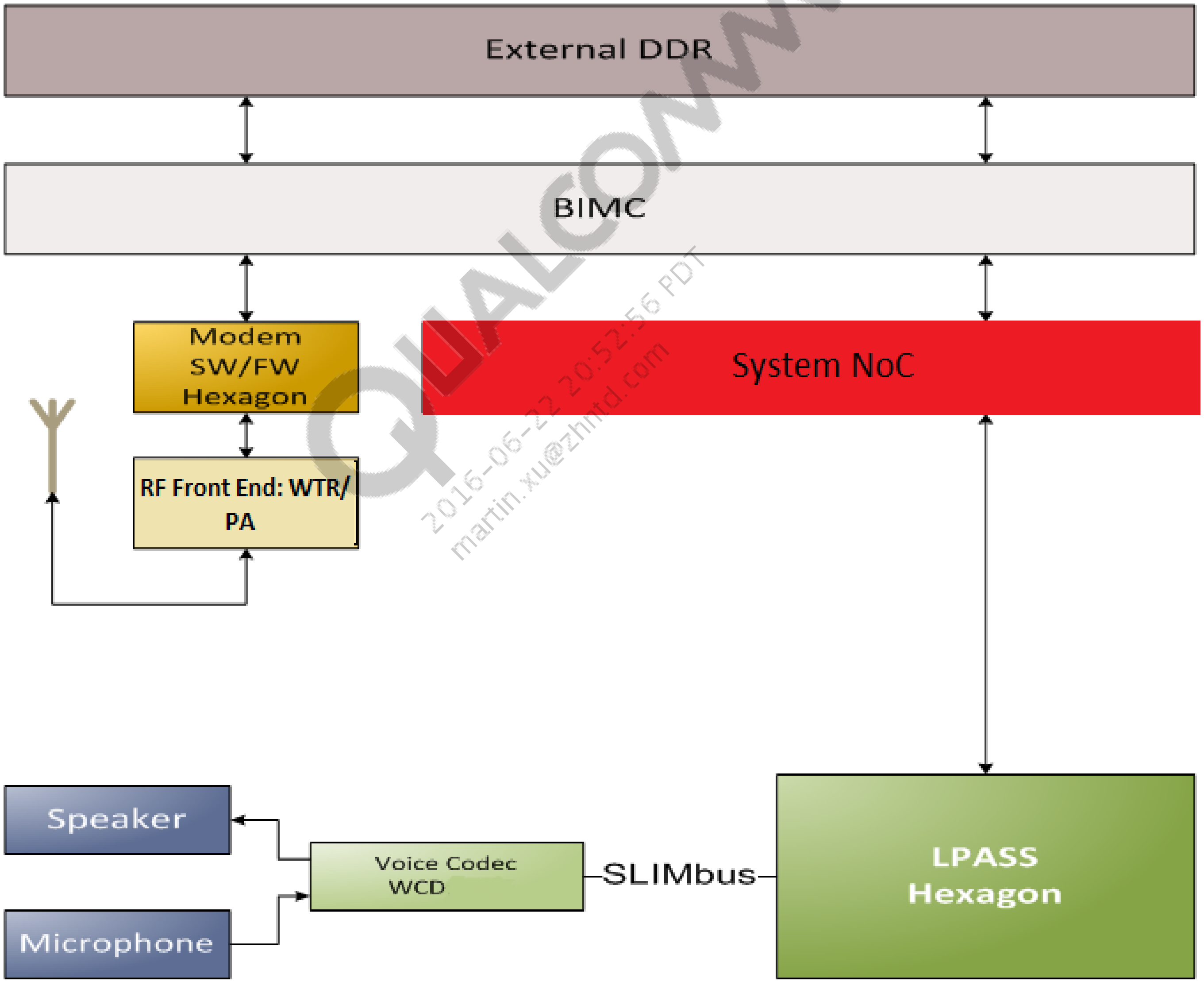
- Extra or unexpected wake-ups during paging standby are common and result in making the average current higher than expected.
- In current measurement waveforms, debug and justify the cause of extra wake-ups using the following methods:
  - MPSS
    - Collect and check modem QXDM Pro logs or Sleep Info.Ulog for wake-ups, except paging wake-ups that cause the modem to run more.
  - APSS
    - When the device is in Standby mode, the APSS must not wake up.
    - To determine if the APSS wakes up, log the apps processor through the serial interface. When the system enters a Sleep state, observe wake-ups from the APSS side on the console.
    - If serial logging is not possible and after observing wake-up in the current monitor, connect the USB and get the kernel logs (kmsg or Dmesg logs). The PM: Suspend string can be searched in the Dmesg logs to determine the wake-up before connecting the USB and justifying the cause.
    - If wake-ups are not always evident from the kernel logs, check logcat logs for user-space application information that results in system wake-up.

Extra Wake-Ups (cont.)

- In current measurement waveforms, debug and justify the cause of extra wake-ups using the following methods (cont.):
  - External components
    - If wake-up is not attributed to the APSS or MPSS, it is possible that some external modules cause wake-ups, e.g., sensors and Wi-Fi and Bluetooth connectivity.
    - Use power rail breakdown to check external modules generating extra wake-ups.

Power Debugging – Talk and Data

Voice Flow – Components Impacting Voice and Talk Power



Power Debugging – Talk and Data

- Higher talk (call) or data current is caused due to one the following:
  - Test setup related issues
  - System resources, auch as PMIC, clocks and shared resource issues
  - Other subsystems impacting current consumption
  - PA/RF front end consuming higher current consumption

Test Setup-Related Issues

- Set NV 00010 to the modem technology under test, i.e., WCDMA only, GSM only, 1X only, and LTE only, to ensure no IRAT searches occur.
- Set the Tx and Rx power at the specified power level as in the test case definition for QTI standard power dashboard and verify that the PA gain state is set to zero.
- Disable QDSS when measuring power.
- Disable data when measuring talk current consumption because it could impact talk current.
- See *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1) for any test setup-related issues and correct NV configurations for each use case.

System Resources – PMIC, Clocks, Shared Resource Issues

- Check clock dumps.
  - Clocks running at a higher frequency than expected in the system can cause the system to consume more power. A clock dump can be taken and compared with the QRD clock dump to ensure that clocks are at the expected state.
  - Clock dumps can be taken by breaking the system at any point through RPM and running the clock dump script.
  - The following are important clocks to compare, however, a complete comparison is always more accurate and helpful:
    - BIMC (DDR) controller clocks
    - Modem clock
    - LPASS clock
    - Buses (System NOC, Peripheral NOC, Config NOC)
- Check PMIC dumps.
  - No extra SMPS/LDOs should be ON, which could impact higher current consumption.
  - Check state and voltage level of each SMPS/LDO against the QRD; this can provide information if any SMPS/LDO is either running at a higher voltage or has been misconfigured.
  - Check that no extra PMIC modules, e.g., WLED, are turned off correctly.

System Resources – PMIC, Clocks, Shared Resource Issues (cont.)

- Check NPA dumps.
  - Compare modem NPA dumps against the QRD.
  - NPA dumps can provide information on various shared system resources.
  - Compare the state of each shared resource, such as SNOC, PNOC, or CPU_VDD, between the DUT and QRD.

Protocol Issues

- Protocol-related issues can be debugged using QXDM Pro logging.
  - In QXDM Pro, select individual air interface messages and log packets.
  - Collect QXDM Pro logs during the time of issue, which can help QTI and OEM protocol teams analyze the issue.
  - For any talk and data related use cases, compare QXDM Pro logs for differences in any settings between the QRD and DUT.

Other Subsystems Impacting Current Consumption

- Talk use case
  - During talk (call) use cases, the APSS is expected to be in a Suspend state, while work is done by the MPSS and LPASS.
  - Check the following subsystems if the current consumption is higher:
    - Ensure that the APSS is in suspend power collapse.
    - Ensure that LPASS is running at similar frequency as QRD. Check it by taking clock dumps.
    - If there are OEM modifications in LPASS, e.g., OEM proprietary voice algorithms, account for these changes.
    - Ensure that the Wi-Fi and Bluetooth subsystems and the sensors subsystem are off when measuring current consumption and that all power rails related to these components are off to minimize leakage due to these components.
- Data use case
  - Ensure that LPASS is power collapsed.
  - Ensure that the Wi-Fi and Bluetooth subsystems and the sensors subsystem are off.
  - The apps processor can be one of the major reasons for higher current consumption during data use cases.

Other Subsystems Impacting Current Consumption (cont.)

- Use the following methods or logs to determine if APSS consumes higher current consumption.
  - Analyze powertop logs and compare them against the QRD.
    - In powertop logs, observe the time in each C-state (low power states).
    - In powertop logs, check the time in each frequency and interrupt.
    - These tests provide a rough estimate of why the apps processor consumes higher current.
    - These logs can be compared with QRD logs for a comparative analysis of which system components are supposed to be up.
  - Analyze top logs and compare them against the QRD.
    - Stop any unexpected processes that are displayed in top logs that consume CPU time.
    - Check for any extra running processes using top data.
  - Analyze FTrace logs.
    - Analyze any process or API that runs in the kernel resulting in extra CPU time.
    - Analyze the amount of time each core is in a particular frequency.
    - If kworker thread activity is higher, check which process monopolizes the CPU. Check FTrace logs to determine which processes initiate kworker.
    - Compare FTrace logs with QRD to determine which processes are more active on the APSS and whether they can be stopped.

RF/PA Front End Higher Power Consumption

- Set RF and PA at correct Tx and Rx levels as mentioned in *Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices* (80-N6837-1).
- Verify Tx and Rx power by measurements on a call box or by QXDM Pro logs.
- Check Tx and Rx power graphical representations at runtime in QXDM Pro, e.g., in WCDMA, go to **View > New > WCDMA > Power**.
- Measure PA power rails as the first part of breakdown effort to verify that its working as expected.

Rail Level Current Consumption Breakdown

- Break down current consumption for individual power rails to narrow in on which component of the system consumes a higher current consumption.
  1. Break down current consumption of components directly powered by VPH, i.e., components not powered by PMIC.
  2. Collect further breakdowns of components connected to a particular SMPS to pinpoint which component consumes higher current.
- Collect the current consumption of all PMIC rails on PMIC input, e.g., SMPS current and any LDOs that are not sourced by SMPS.

Summary

Summary

- Setup is a common problem in all modem power issues.
- Test in the lab first before testing in a live network.
- Sleep, Rock Bottom, and Airplane
  - Check the settings and disable applications.
  - Make sure all subsystems are in Sleep mode.
  - Identify the culprit subsystem and debug further.
  - Check power rails consuming higher current using rail level breakdown measurement.
- Standby
  - Switch off neighbor cells and test.
  - Set correct DRX cycle setting according to the dashboard.
    - GSM DRX cycle – 1.18 sec
    - WCDMA DRX cycle – 2.56 sec
    - CDMA DRX cycle – 5.12 sec
    - LTE DRX cycle – 2.56 sec

Summary (cont.)

- Talk
  - Verify that the apps processor is in a power collapsed state.
  - Check the Tx power setting according to the dashboard recommendation.
- Data
  - Disable the applications running in the background.
  - Disable the data monitor.
  - Most high current issues are from the apps processor.
  - Investigate what is keeping the apps processor occupied.

References

References

Documents	
Qualcomm Technologies, Inc.	
Power Consumption Measurement Procedure for MSM (Android-Based)/MDM Devices	80-N6837-1
Linux Android Current Consumption Data	▪ MSM8909 – 80-NP408-7 ▪ MSM8916 – 80-NK807-7 ▪ MSM8994 – 80-NJ051-7 ▪ MSM8939 – 80-NM683-7
System Power Overview	▪ MSM8909 – 80-NR964-5 ▪ MSM8916 – 80-NL239-6 ▪ MSM8994 – 80-NM328-15 ▪ MSM8939 – 80-NM846-2
Modem/Multimedia Use Case Details	▪ MSM8916 – 80-NL239-60 ▪ MSM8994 – 80-NM328-704
*Power Consumption Optimization and Debugging**	▪ MSM8916 – 80-NL239-48 ▪ MSM8994 – 80-NM328-98
Release Notes	▪ MSM8916 – 80-NM886-x, 80-NM872-x ▪ MSM8994 – 80-NN696-x ▪ MSM8939 – 80-NP191-x, 80-NP192-x

*Refer to this document for more details on power debugging related to sleep, standby, talk, and data use cases.

Questions?

https://support.cdmatech.com