
Idle XO Shutdown/VDD Minimization Overview



Qualcomm Technologies, Inc.

80-P0897-1 A

Confidential and Proprietary – Qualcomm Technologies, Inc.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.



Confidential and Proprietary – Qualcomm Technologies, Inc.

QUALCOMM
2017-08-07 18:22:11 PDT
lilubao@gionee.com

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to: DocCtrlAgent@qualcomm.com.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

Qualcomm is a trademark of Qualcomm Incorporated, registered in the United States and other countries. All Qualcomm Incorporated trademarks are used with permission. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

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

© 2015 Qualcomm Technologies, Inc. and/or its affiliated companies. All rights reserved.

Revision History

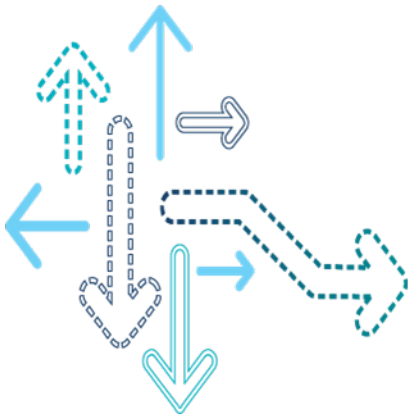
Revision	Date	Description
A	May 2015	Initial release

QUALCOMM
2017-08-07 18:22:11 PDT
lilubao@gionee.com

Contents

- Idle and Suspend
- GIC Interrupts vs. MPM Interrupts
- Linux IRQ Number vs. Hardware IRQ Number
- Idle XO Shutdown/VDD Minimization Debug
- Acronyms
- Questions?

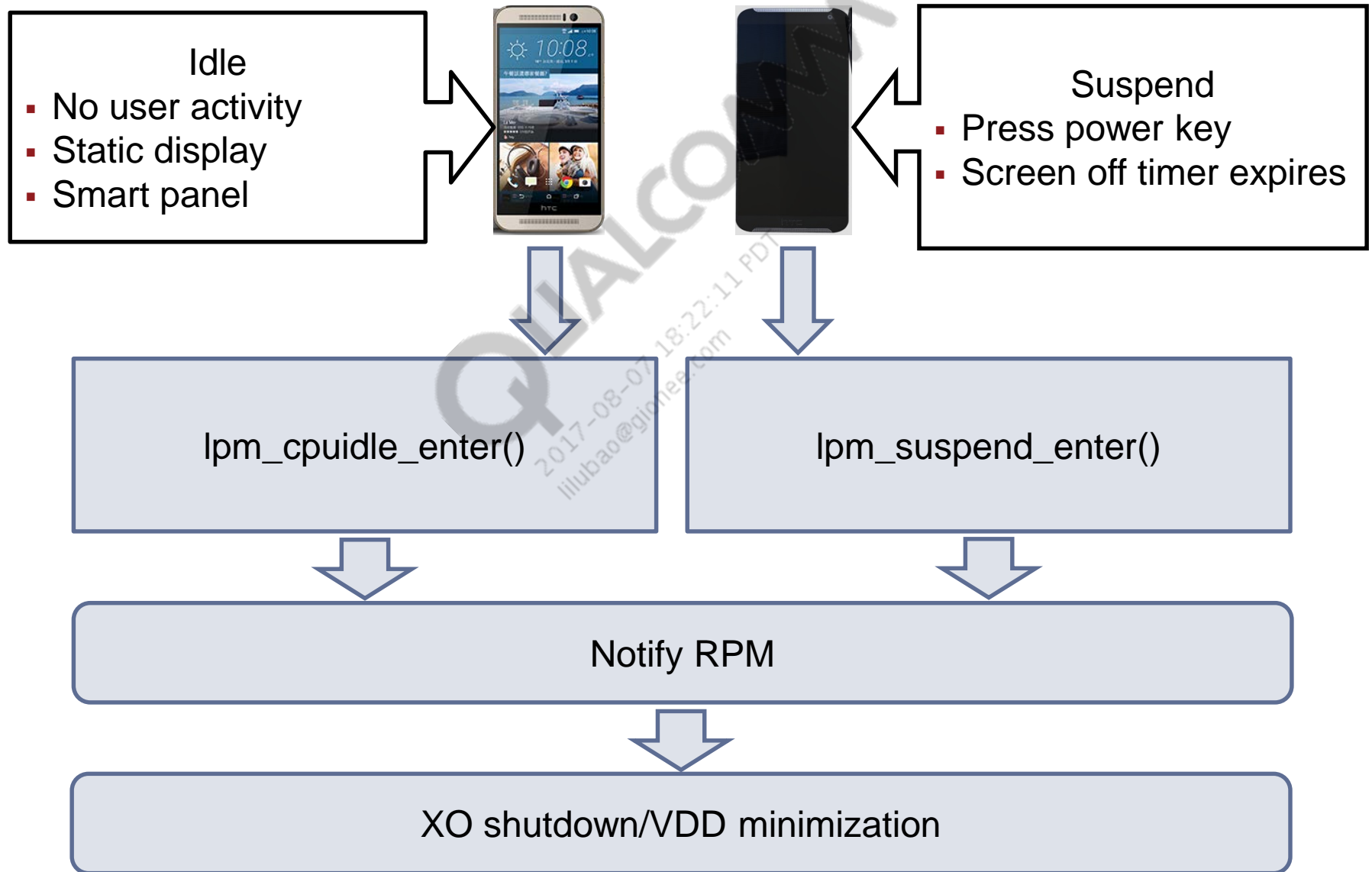
Idle and Suspend



Idle and Suspend Differences

- Idle
 - No user activity
 - Display is on and static
 - Device is capable of entering XO shutdown/VDD minimization if the device is equipped with a smart panel
- Suspend
 - Triggered by specific operations, e.g., pressing the power key and the screen off timer expired
 - Display is off
 - Device is capable of entering XO shutdown/VDD minimization

Idle and Suspend Process Flow



Idle XO Shutdown Path – Check GIC/GPIO IRQs Prevent Sleep

- No task runs
- CPU idle duration longer than sleep+resume latency



lpm_cpuidle_enter()



msm_mpm_interrupts_detectable()

- Device cannot enter XO shutdown/VDD minimization if:
 - Any GIC/GPIO IRQ is enabled
 - The enabled IRQ is not in the bypass list (defined in DTSI)



Notify RPM

Handover MPM Configuration to RPM

`lpm_cpuidle_enter()` or `lpm_suspend_enter`



Write the intended MPM interrupt to the virtual MPM register in SMED



Notify RPM

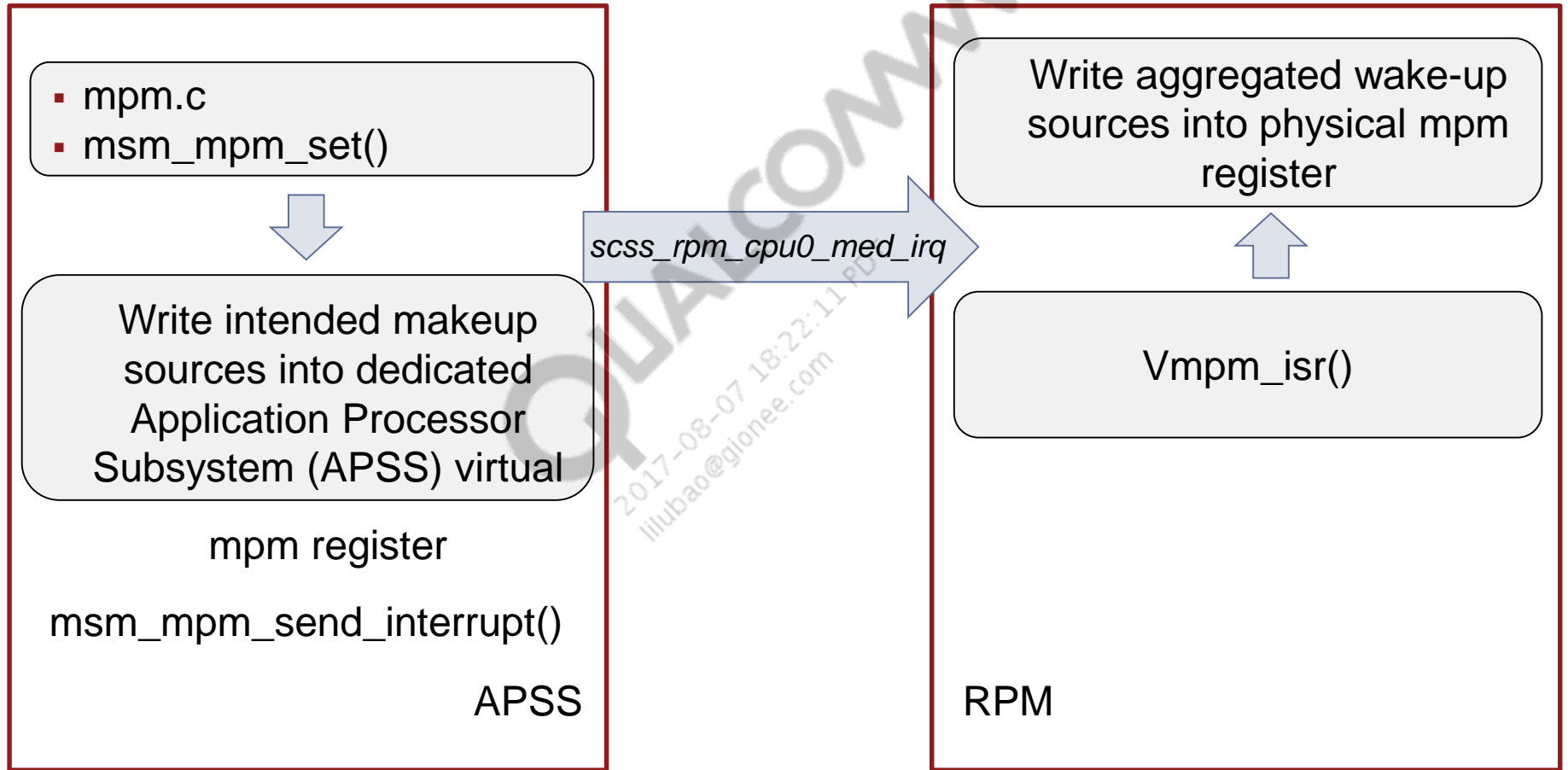


RPM aggregates all subsystems' Virtual MSM Power Manager (VMPPM) to write into the hardware MPM register

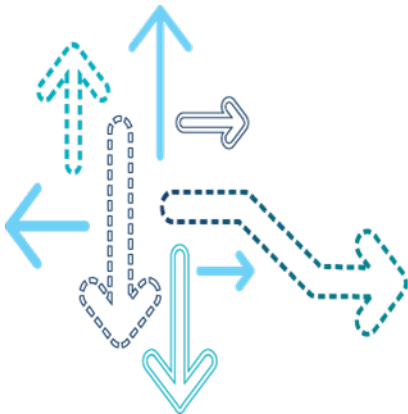


XO shutdown/VDD minimization

Handover MPM Configuration to RPM (cont.)

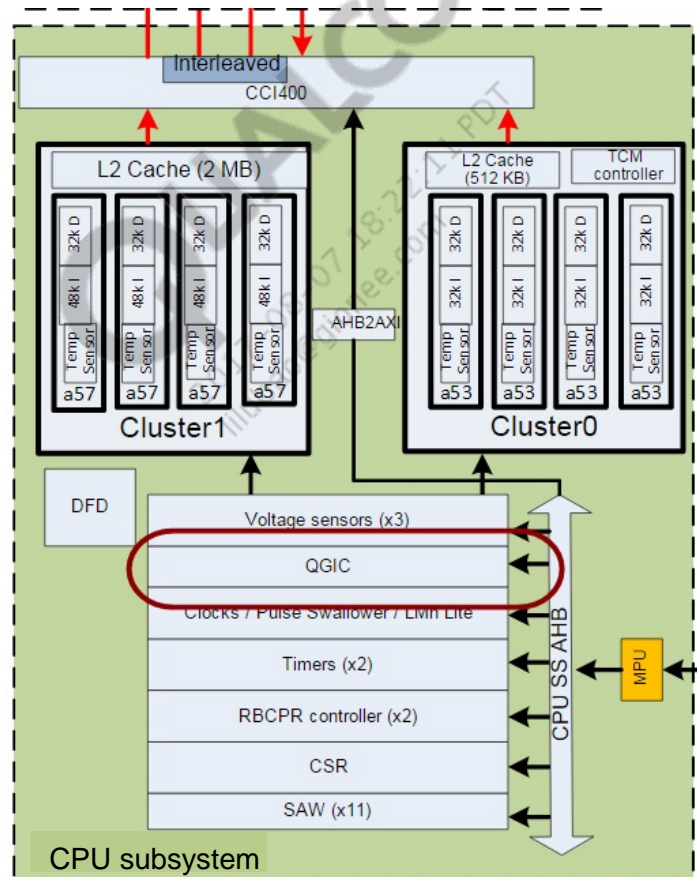


GIC Interrupts vs. MPM Interrupts



GIC Interrupts vs. MPM Interrupts

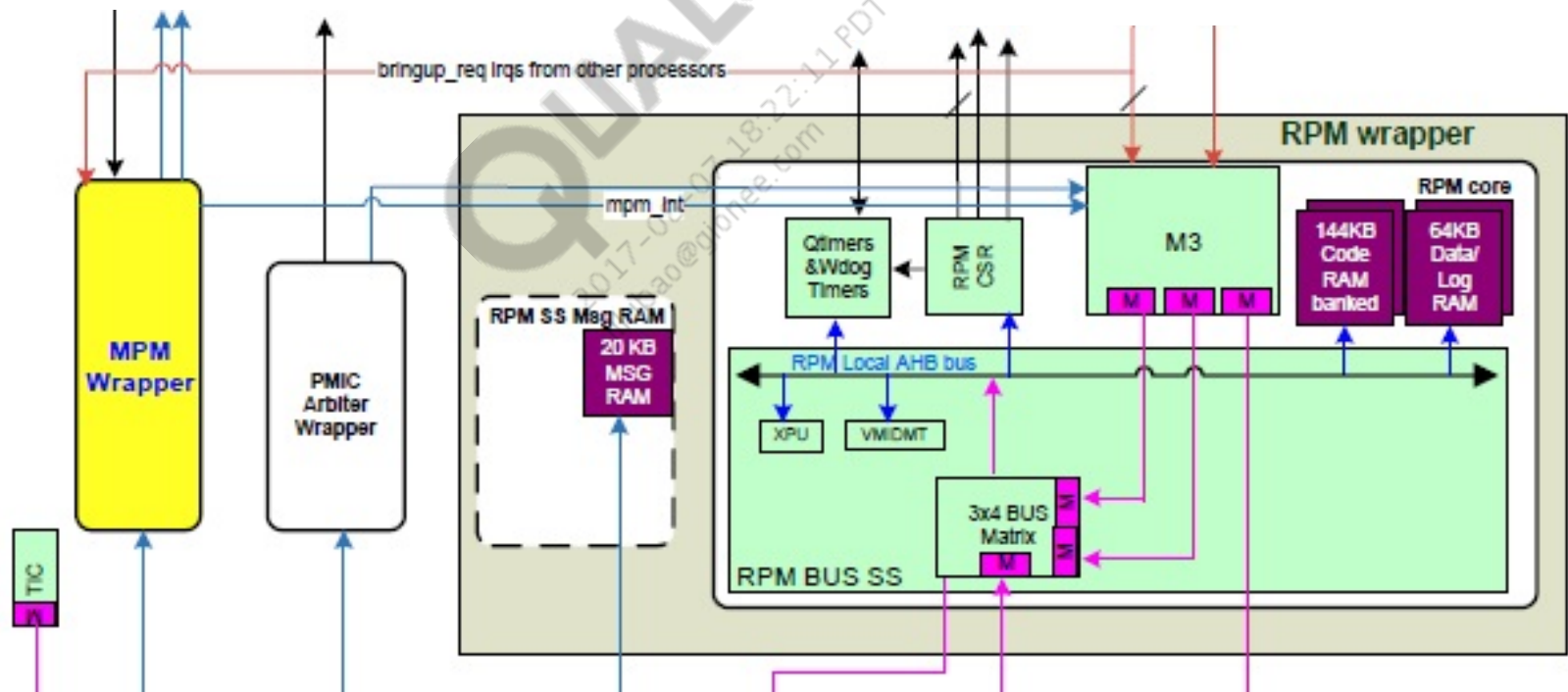
- GIC interrupts
 - Controlled by the APSS subsystem
 - Not capable of waking up the device from XO shutdown/VDD minimization



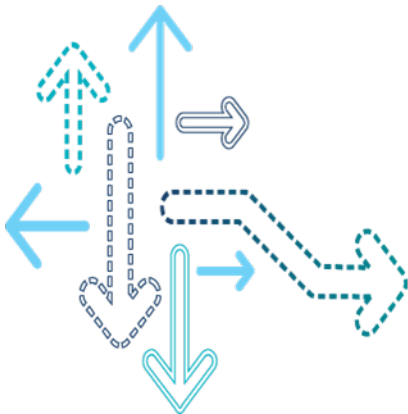
MSM8994 V2 Chip Block Diagram

GIC Interrupts vs. MPM Interrupts (cont.)

- MPM interrupts
 - Controlled by the RPM
 - Capable of waking up the device from XO shutdown/VDD minimization



Linux IRQ Number vs. Hardware IRQ Number



Linux IRQ Number vs. Hardware IRQ Number Device Tree

```
#interrupt-cells = <1>;
interrupt-map-mask = <0xffffffff>;
interrupt-map = <0 &intc 0 271 0
```

```
1 &intc 0 272 0
2 &intc 0 273 0
3 &intc 0 274 0
4 &intc 0 275 0
5 &intc 0 276 0
6 &intc 0 277 0
7 &intc 0 278 0
8 &intc 0 279 0
9 &intc 0 280 0
10 &intc 0 281 0
11 &intc 0 282 0>
```

- 1 – Interrupt type
 - 0 – SPI
 - 1 – PPI
- 2 – Interrupt number of interrupt type
- 3 – Trigger type
 - 1 – Low-to-high edge triggered
 - 2 – High-to-low edge triggered
 - 4 – Active high-level sensitive
 - 8 – Active low-level sensitive

```
interrupt-names = "int_msi", "int_a"
```

$hwirq = GIC_PPI_START / GIC_SPI_START + \text{interrupt number}$

QUALCOMM INNOVATION CENTER, INC

ref: /LA.BR.1/kernel/arch/arm/mach-msm/include/mach/irqs.h

Home | History | Annotate | Line# | Navigate | Raw | Download

☐ only in /LA.BR.1/kernel/arch/arm/mach-msm/include/mach/

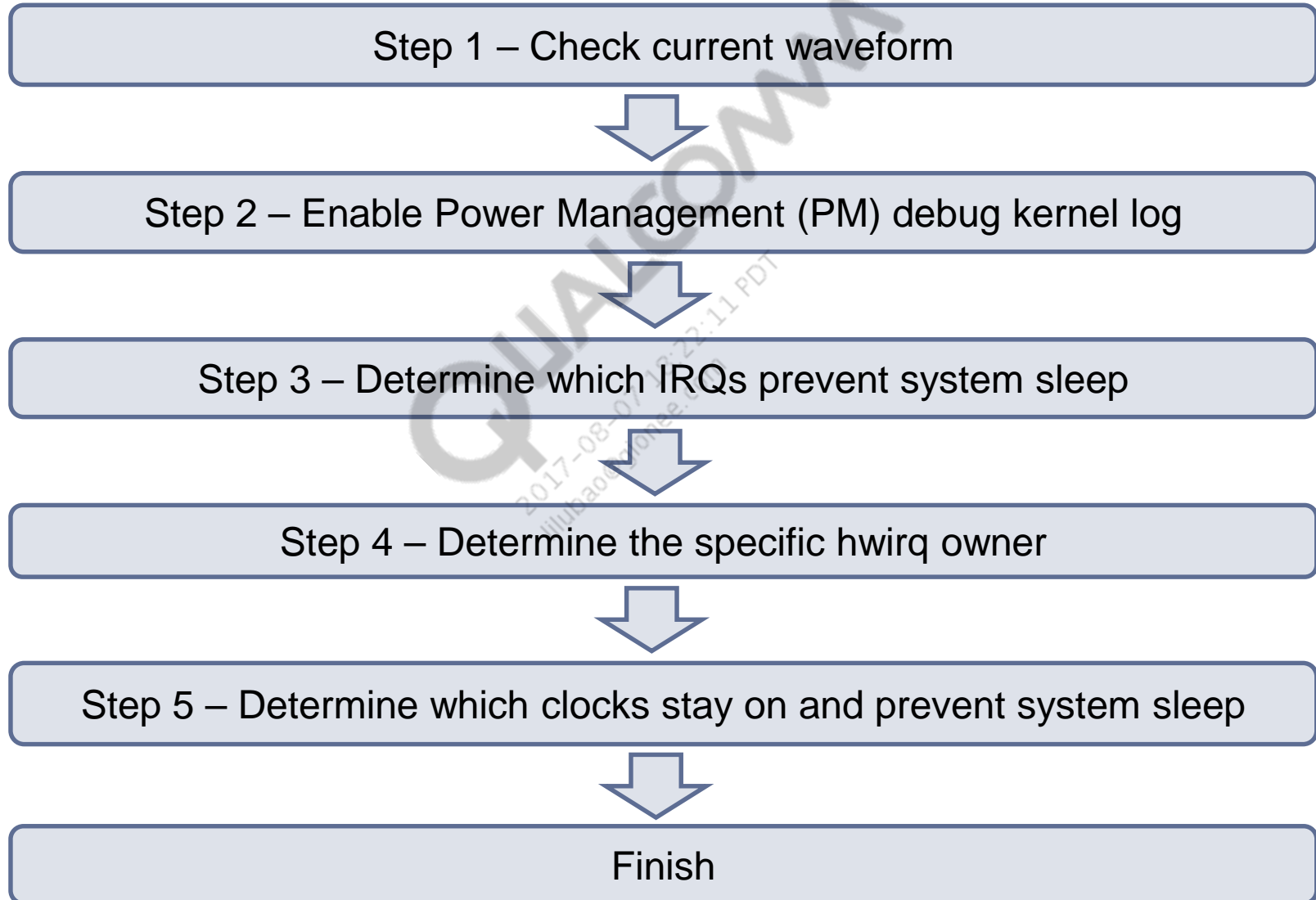
```
21 * 0-15: SSI/SGI (software triggered/generated interrupts)
22 * 16-31: PPI (private peripheral interrupts)
23 * 32+: SPI (shared peripheral interrupts)
24 */
25 #define GIC_PPI_START 16
26 #define GIC_SPI_START 32
27
```

start index of PPI and SPI

Idle XO Shutdown/VDD Minimization Debug

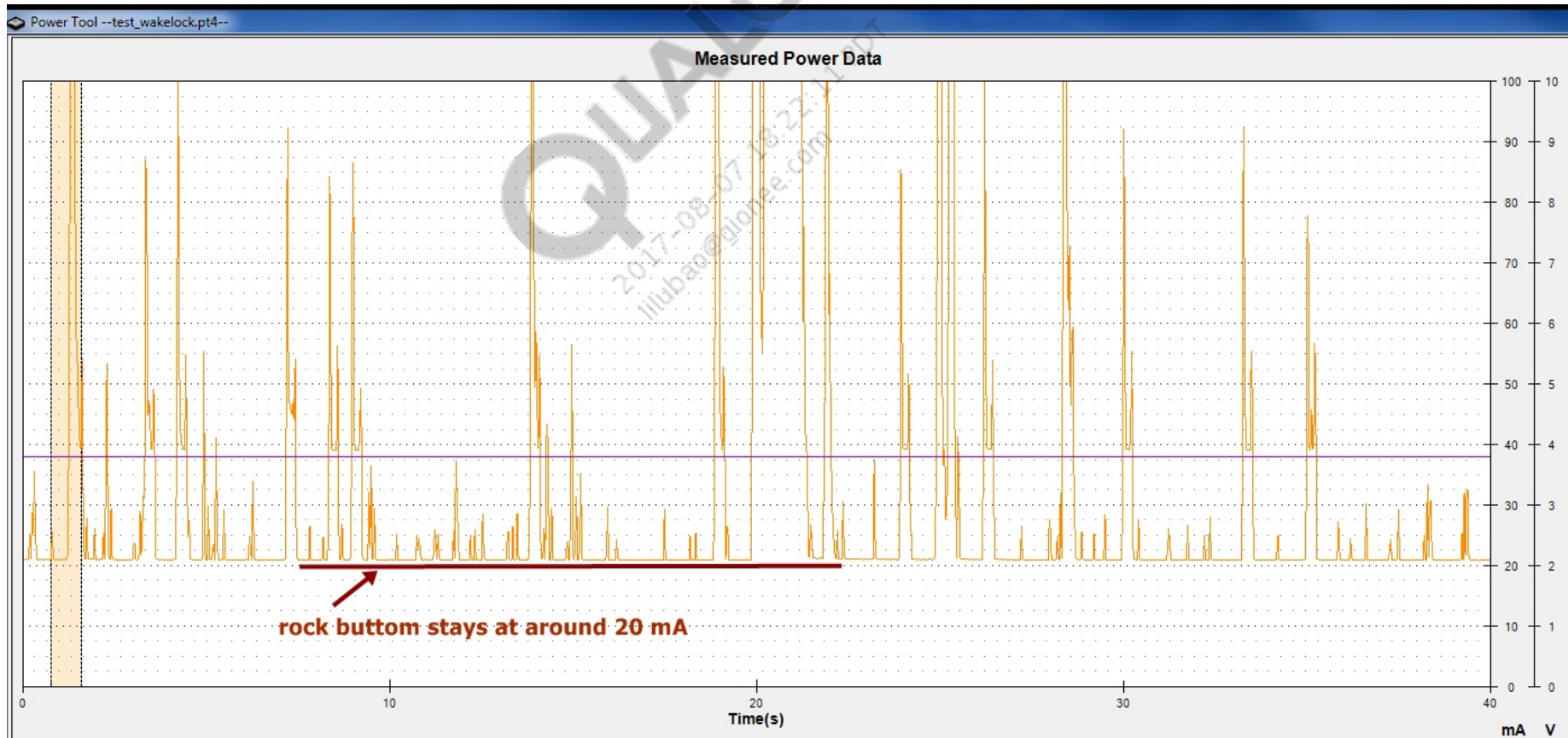


Handover MPM Configuration to RPM



Step 1 – Check Current Waveform

- Rock bottom of VDD minimization should be 2 mA to 3 mA.
- Rock bottom of static display fails if it stays at around 20 mA; this indicates that something negates the XO shutdown and VDD minimization.



Step 2 – Enable PM Debug Kernel Log

- `echo 32 > /sys/module/msm_pm/parameters/debug_mask`

```
enum {  
    MSM_PM_DEBUG_SUSPEND = BIT(0),  
    MSM_PM_DEBUG_POWER_COLLAPSE = BIT(1),  
    MSM_PM_DEBUG_SUSPEND_LIMITS = BIT(2),  
    MSM_PM_DEBUG_CLOCK = BIT(3),  
    MSM_PM_DEBUG_RESET_VECTOR = BIT(4),  
    MSM_PM_DEBUG_IDLE_CLK = BIT(5),  
    MSM_PM_DEBUG_IDLE = BIT(6),  
    MSM_PM_DEBUG_IDLE_LIMITS = BIT(7),  
    MSM_PM_DEBUG_HOTPLUG = BIT(8),  
};
```

- `echo 8 > /sys/module/mpm_of/parameters/debug_mask`

```
enum {  
    MSM_MPM_DEBUG_NON_DETECTABLE_IRQ = BIT(0),  
    MSM_MPM_DEBUG_PENDING_IRQ = BIT(1),  
    MSM_MPM_DEBUG_WRITE = BIT(2),  
    MSM_MPM_DEBUG_NON_DETECTABLE_IRQ_IDLE = BIT(3),  
};
```

Step 3 – Determine Which IRQs Prevent System Sleep

- Log example

```
<6>[ 1695.567203] [0: kworker/u:0: 6] msm_mpm_interrupts_detectable(): gic
preventing system sleep modes during idle
<6>[ 1695.567252] [0: kworker/u:0: 6] hwirq: 65 -> kgs1-3d0
<6>[ 1695.567277] [0: kworker/u:0: 6] hwirq: 157 -> mmc1
<6>[ 1695.567303] [0: kworker/u:0: 6] hwirq: 159 -> mmc2
<6>[ 1695.567328] [0: kworker/u:0: 6] hwirq: 253 -> msm_sdcc.2
<6>[ 1695.567354] [0: kworker/u:0: 6] hwirq: 256 -> msm_sdcc.3
```

- Analysis – Search hwirq to determine which IRQs prevent system sleep
- Solution 1 – Drivers should disable the IRQ if it is *not* required
- Solution 2 – Add the IRQ to the bypass list in DTSI

```
qcom,gic-parent = <&intc>;
qcom,gic-map = <2 216>, /* tsens_upper_lower_int */
               <47 165>, /* usb30_hs_phy_irq */
               <50 172>, /* usb1_hs_async_wakeup_irq */
               <53 104>, /* mdss_irq */
               <62 222>, /* ee0_krait_hlos_spmi_periph_irq */
               <0xff 57>, /* mss_to_apps_irq(0) */
               <0xff 58>, /* mss_to_apps_irq(1) */
               <0xff 59>, /* mss_to_apps_irq(2) */
```

Step 4 – Determine the Specific hwirq Owner

- Cat/proc/interrupt

270:	2	0	0	0	GIC	sps
273:	0	0	0	0	GIC	msm_iommu_nonsecure_irq
274:	0	0	0	0	GIC	msm_iommu_nonsecure_irq
280:	0	0	0	0	GIC	mobicore
288:	6	0	0	0	qnpn-int	qnpn_kdpwr_status

- Determine which function enables the specific IRQ that causes sleep fail

```
--- a/arch/arm/mach-msm/mpm-of.c
+++ b/arch/arm/mach-msm/mpm-of.c
@@ -252,6 +252,11 @@ static int msm_mpm_enable_irq_exclusive(
irq_apps = wakeset ? unlisted_irqs[i].wakeup_irqs :
unlisted_irqs[i].enabled_irqs;

+ if (d->hwirq == 280 && enable) {
+ pr_err ("!!!hwirq 280 registered, wakeset %d\n", wakeset);
+ dump_stack();
+ }
+
if (enable)
▪ __set_bit(d->hwirq, irq_apps);
▪ else
```

Step 4 – Determine the Specific hwirq Owner (cont.)

- Dump result sample

[<c017ccd8>] msm_mpm_enable_irq_exclusive+0x164

[<c017cdcc>] __msm_mpm_enable_irq+0x38

[<c0115600>] gic_unmask_irq+0x38

[<c01f6fb8>] irq_enable+0x28

[<c01f59f4>] __enable_irq+0x9c

[<c01f5a78>] enable_irq+0x60

[<c05a5258>] qup_i2c_xfer+0x28c

[<c05a0ea0>] i2c_transfer+0xb8

[<c05948c4>] gtp_i2c_read+0x60

Step 5 – Determine Which Clocks Stay On and Prevent System Sleep

- Log example

```
<6>[ 1353.995631] c0 0 Enabled clock count: 32
<6>[ 1353.996721] c0 0 Enabled clocks:
<6>[ 1353.996731] c0 0 cxo_clk_src:1:1 [19200000]
<6>[ 1353.996743] c0 0 pnoc_clk:1:1 [9600000]
<6>[ 1353.996756] c0 0 pnoc_a_clk:1:1 [19200000]
<6>[ 1353.996767] c0 0 bimc_clk:1:1 [9600000]
<6>[ 1353.996778] c0 0 bimc_a_clk:1:1 [459931648]
<6>[ 1353.996789] c0 0 snoc_clk:1:1 [9600000]
<6>[ 1353.996799] c0 0 snoc_a_clk:1:1 [100000000]
<6>[ 1353.996812] c0 0 bimc_msmbus_clk:1:1 [9600000] -> bimc_clk:1:1 [9600000]
```

- Search “Enabled clocks” to determine which clocks prevent XO shutdown and VDD minimization
- The serial console is enabled by default in the engineering build; if the engineering build causes an idle XO shutdown fail, disable the serial console to fix it

```
kernel/arch/arm/configs/msm8994_defconfig
#CONFIG_SERIAL_MSM_HSL=y
#CONFIG_SERIAL_MSM_HSL_CONSOLE=y
```

Acronyms

Acronyms	
Term	Definition
hwirq	Hardware Interrupt Request
PM	Power Management
VMPPM	Virtual MSM Power Manager

