1. How to get dump with CPU context ?

- Non-secure device
  - On 8937/8953/8996 platform, need consider the possibility of calling sdi disable in kernel side (scm_disable_sdi), it will take effect before power cycle.
- Secure device
  - Legacy TZ platform: solution 00029197 works
  - Chipsets with QSEE 4.0:
    - Step1: kernel side enable CONFIG_CORESIGHT_DBGUI and CONFIG_MSM_DEBUG_LAR_UNLOCK
    - Step2: refer solution 00031166
    - Step3: secDbg_oem.c (both sbl1 and modem side), add #define SKIP_SERIAL_NUMBER_CHECK 1
    - Step4: need disable RPMB and reflash devcfg.mbn:

```
trustzone_images/core/securemsm/trustzone/qsee/mink/oem/config/
msm8937/oem_config.xml
@@ -4,10 +4,10 @@
</global_def>
<device id="/tz/oem">
<props name="OEM_keystore_enable_rpmb" type=
DALPROP_ATTR_TYPE_UINT32>
- 1
+ 0
</props>
<props name="OEM_counter_enable_rpmb" type=
DALPROP_ATTR_TYPE_UINT32>
- 1
+ 0
</props>
<props name="OEM_allow_rpmb_key_provision" type=
DALPROP_ATTR_TYPE_UINT32>
0
```

    - Step5: (preferred)

      1) In " xxxx_secimage.xml" , change OU to 0:

```
-<debug>0x0000000000000002</debug>

+<debug>0x0000000000000000</debug>
```

2) In sbl_mc.c

void sbl1_main_ctl(boot_pbl_shared_data_type *pbl_shared)

{

<snip>

/* Calculate the SBL start time for use during boot logger initialization. */

sbl_start_time = CALCULATE_TIMESTAMP(HWIO_IN(TIMETICK_CLK));

+ HWIO_OUT(OVERRIDE_2, HWIO_OVERRIDE_2_RMSK);

+ HWIO_OUT(OVERRIDE_3, HWIO_OVERRIDE_3_RMSK);

3) Re-sign image and flash

- Step5 (altative)

1) change OU to 2:

In " xxxx_secimage.xml"

-<debug>0x0000000000000002</debug>

+<debug>0x0000000000000003</debug>

2) Read the serial number of the device and add it in xxxx_secimage.xml (not in xxx_debugpolicy.xml) .

3) And re-sign sbl1.mbn and mba.mbn

- Other tips:
  - 3G, 4G memory SDI patches, if miss CR-1003318, possible cause SDI hung

void sys_debug_bric_settings(void)

{

// 1.5GB + 1.5GB configuration

if ((HAL_SDRAM_Get_Base_Addr(SDRAM_INTERFACE_0, SDRAM_CS0) == 0x0) && (device_density == 14) && (device_io_width == 1))

{…….

//SYSNOC slave address range configuration to 512MB

- HWIO_OUTXI(0x00402000, BRIC_GLOBAL2_
BRIC_SEGMENTN_ADDR_MASK_A_LOWER,2,0xE0000000);

+ HWIO_OUTXI(0x00402000, BRIC_GLOBAL2_
BRIC_SEGMENTN_ADDR_MASK_A_LOWER,1,0xE0000000);

}

// 2GB + 2GB configuration

else if (HAL_SDRAM_Get_Base_Addr(SDRAM_INTERFACE_0, SDRAM_CS0) ==
0x0) {

…….

//SYSNOC slave address range configuration to 256MB

-HWIO_OUTXI(0x00402000, BRIC_GLOBAL2_
BRIC_SEGMENTN_ADDR_MASK_A_LOWER,2,0xF0000000);

+ HWIO_OUTXI(0x00402000, BRIC_GLOBAL2_
BRIC_SEGMENTN_ADDR_MASK_A_LOWER,1,0xF0000000);

}

}

- If customer want to use disable hardware self refresh test patch, they need just only comment out the code in ABHN_SHKE_Enable_HW_Self_Refresh() and keep the code as it is in ABHN_SHKE_Disable_HW_Self_Refresh(), otherwise the phone cannot enter dload mode correctly.

2. Patches which can be tried if hardware related.

- 1. Enable RTB

  +CONFIG_MSM_RTB=y
  +CONFIG_MSM_RTB_SEPARATE_CPUS=y

- 2. Disable L2 GDHS

```
diff --git a/rootdir/etc/init.qcom.post_boot.sh b/rootdir/etc/init.qcom.post_boot
index 14e3346..640afbf 100644
--- a/rootdir/etc/init.qcom.post_boot.sh
+++ b/rootdir/etc/init.qcom.post_boot.sh
@@ -1330,6 +1330,10 @@ case "$target" in
                # re-enable thermal core_control now
                echo 1 > /sys/module/msm_thermal/core_control/enabled

+               # Disable L2-GDHS low power modes
+               echo N > /sys/module/lpm_levels/perf/perf-l2-gdhs/idle_enabled
+               echo N > /sys/module/lpm_levels/perf/perf-l2-gdhs/suspend_enable
+
                # Bring up all cores online
                echo 1 > /sys/devices/system/cpu/cpu1/online
                echo 1 > /sys/devices/system/cpu/cpu2/online
@@ -1455,6 +1459,12 @@ case "$target" in
                # re-enable thermal core_control now
                echo 1 > /sys/module/msm_thermal/core_control/enabled

+               # Disable L2-GDHS low power modes
+               echo N > /sys/module/lpm_levels/system/pwr/pwr-l2-gdhs/idle_enab
+               echo N > /sys/module/lpm_levels/system/pwr/pwr-l2-gdhs/suspend_e
+               echo N > /sys/module/lpm_levels/system/perf/perf-l2-gdhs/idle_en
+               echo N > /sys/module/lpm_levels/system/perf/perf-l2-gdhs/suspend
+
                # Bring up all cores online
                echo 1 > /sys/devices/system/cpu/cpu1/online
                echo 1 > /sys/devices/system/cpu/cpu2/online
```

- 3. Limiting Cx Automode to BIMC @384MHz from 556MHz

  In rpm_proc\core\boot\ddr\hw\msm8937\ddr_target.h
  - #define DDR_PMIC_AUTOMODE_THRESHOLD_2GB 556800
  - #define DDR_PMIC_AUTOMODE_THRESHOLD_3GB 556800

  + #define DDR_PMIC_AUTOMODE_THRESHOLD_2GB 384000
  + #define DDR_PMIC_AUTOMODE_THRESHOLD_3GB 384000

- 4. CX auto mode totally disable

  1. boot_images/core/systemdrivers/pmic/config/msm8937/pmi8950/
  pm_config_target_sbl_sequence.c

  // MODE - S2_HFS_CONFIG: 14

  //sid data base_addr offset reg_op rev_id_op rev_id

  -- { 1, 0x40, 0x1700, 0x045, PM_SBL_WRITE, EQUAL, REV_ID_COMMON}, // 18

  ++ { 1, 0x80, 0x1700, 0x045, PM_SBL_WRITE, EQUAL, REV_ID_COMMON}, // 18

  2. rpm_proc/core/systemdrivers/pmic/config/msm8937/pm_config_target.c

--- {300, 0, PM_ACCESS_ALLOWED, PM_ALWAYS_ON, PM_NPA_SW_MODE_SMPS__AUTO, PM_CLK_3p2_MHz, PM_CLK_3p2_MHz, PM_DROOP_DETECT_DIS, 500, 1375, 0, PM_SETTLING_ERR_EN, PM_SETTLING_EN, 0}, // HFS 2

+++{300, 0, PM_ACCESS_ALLOWED, PM_ALWAYS_ON, PM_NPA_SW_MODE_SMPS__NPM, PM_CLK_3p2_MHz, PM_CLK_3p2_MHz, PM_DROOP_DETECT_DIS, 500, 1375, 0, PM_SETTLING_ERR_EN, PM_SETTLING_EN, 0}, // HFS 2

- 5. Change VDD APC to max

---
arch/arm/boot/dts/qcom/msm8937-regulator.dtsi | 6 ++++--
1 file changed, 4 insertions(+), 2 deletions(-)

diff --git a/arch/arm/boot/dts/qcom/msm8937-regulator.dtsi b/arch/arm/boot/dts/qcom/msm8937-regulator.dtsi
index 9f0b79d..4124fda 100644
--- a/arch/arm/boot/dts/qcom/msm8937-regulator.dtsi
+++ b/arch/arm/boot/dts/qcom/msm8937-regulator.dtsi
@@ -359,8 +359,10 @@
regulator-max-microvolt = <6>;

qcom,cpr-fuse-corners = <3>;
- qcom,cpr-voltage-ceiling = <1155000 1225000 1350000>;
- qcom,cpr-voltage-floor = <1050000 1050000 1090000>;
+ //qcom,cpr-voltage-ceiling = <1155000 1225000 1350000>;
+ //qcom,cpr-voltage-floor = <1155000 1225000 1350000>;
+ qcom,cpr-voltage-ceiling = <1350000 1350000 1350000>;
+ qcom,cpr-voltage-floor = <1350000 1350000 1350000>;
vdd-apc-supply = <&pm8937_s5>;

mem-acc-supply = <&mem_acc_vreg_corner>;

- 6. Boost LDO2 , set LDO2 to 1.25V always

1) boot_images/core/systemdrivers/pmic/target/msm8937_pm8937_pmi8937/system/src/pm_sbl_boot_oem.c

pm_err_flag_type
pm_driver_post_init(void)
{

```
pm_err_flag_type err_flag = PM_ERR_FLAG__SUCCESS;
+++ err_flag = pm_ldo_volt_level(0, PM_LDO_2, 1250000);
return err_flag;
}
```

2) rpm_proc/core/systemdrivers/pmic/config/msm8937/pm_config_target.c

```
pm_rpm_ldo_rail_info_type ldo_rail_a[] =
{
{5, 62.5, 0, PM_ACCESS_ALLOWED, PM_NONE, PM_NPA_SW_MODE_LDO__NPM,
PM_NPA_BYPASS_DISALLOWED, PM_DROOP_DETECT_DIS, 1000, 1050, 0,
PM_SETTLING_ERR_EN, PM_SETTLING_EN, 0}, // LDO1 ULT N600_Stepper
- {5, 62.5, 0, PM_ACCESS_ALLOWED, PM_ALWAYS_ON, PM_NPA_SW_MODE_LDO
__IPEAK, PM_NPA_BYPASS_DISALLOWED, PM_DROOP_DETECT_DIS, 1200, 1300,
0, PM_SETTLING_ERR_EN, PM_SETTLING_EN, 0}, // LDO2 ULT N600_Stepper
+ {5, 62.5, 0, PM_ACCESS_ALLOWED, PM_ALWAYS_ON, PM_NPA_SW_MODE_LDO
__IPEAK, PM_NPA_BYPASS_DISALLOWED, PM_DROOP_DETECT_DIS, 1250, 1300,
0, PM_SETTLING_ERR_EN, PM_SETTLING_EN, 0}, // LDO2 ULT N600_Stepper
```

3) core/systemdrivers/pmic/config/msm8937/pm_config_rpm_npa_pam.c

```
/* LPDDR Client */
static pm_npa_ldo_int_rep
pm_rpm_pam_lpddr_a_ldo2[] =
{
/**< Mode: PMIC_NPA_MODE_ID_DDR_SLEEP_SPEED*/
{
PM_NPA_GENERIC_DISABLE, /**< [Disable (default), Enable] -> max aggregation (left
to right). */
PM_NPA_SW_MODE_LDO__IPEAK, /**< [BYPASS, IPEAK (default), NPM] -> max
aggregation (left to right). */
PM_NPA_PIN_CONTROL_ENABLE__NONE, /**< [NONE, EN1, EN2, EN3, EN4] ->
ORed value of list. */
PM_NPA_PIN_CONTROL_POWER_MODE__NONE, /**< [NONE, EN1, EN2, EN3, EN4
, SLEEPB] -> ORed value of list. */
2, /**< Perpherial Index */
0, /**< Primary (0) or Secondary (1) PMIC */
0, /**< If((old sw_en == disable) && (new sw_en == enable) || new pc_en == enable then
ldo_en_trans = true else ldo_en_trans = false */
PM_NPA_BYPASS_DISALLOWED, /**< [Allowed (default), Disallowed]*/
0, /**< reserve 1 - for 32 bit boundary */
```

0, /**< [X uV] -> max aggregation. */
0, /**< [X mA] -> summed aggregation. */
0, /**< [X uV] -> voltage headroom needed. */
0, /**< [X uV] -> max aggregation. */
//PM_NPA_CORNER_MODE__NONE, /**< [None, Level1 (Retention), Level2, Level3, Level4, Level5, Level6 (SuperTurbo), Not Used] */
},
/**< Mode: PMIC_NPA_MODE_ID_DDR_LOW_SPEED*/
{
PM_NPA_GENERIC_ENABLE, /**< [Disable (default), Enable] -> max aggregation (left to right). */
PM_NPA_SW_MODE_LDO__NPM, /**< [BYPASS, IPEAK (default), NPM] -> max aggregation (left to right). */
PM_NPA_PIN_CONTROL_ENABLE__NONE, /**< [NONE, EN1, EN2, EN3, EN4] -> ORed value of list. */
PM_NPA_PIN_CONTROL_POWER_MODE__NONE, /**< [NONE, EN1, EN2, EN3, EN4, SLEEPB] -> ORed value of list. */
2, /**< Perpherial Index */
0, /**< Primary (0) or Secondary (1) PMIC */
0, /**< If((old sw_en == disable) && (new sw_en == enable) || new pc_en == enable then ldo_en_trans = true else ldo_en_trans = false */
PM_NPA_BYPASS_DISALLOWED, /**< [Allowed (default), Disallowed]*/
0, /**< reserve 1 - for 32 bit boundary */
0, /**< [X uV] -> max aggregation. */
300, /**< [X mA] -> summed aggregation. */
100, /**< [X uV] -> voltage headroom needed. */
1250000, /**< [X uV] -> max aggregation. */
//PM_NPA_CORNER_MODE__NONE, /**< [None, Level1 (Retention), Level2, Level3, Level4, Level5, Level6 (SuperTurbo), Not Used] */
},
/**< Mode: PMIC_NPA_MODE_ID_DDR_MID_SPEED*/
{
PM_NPA_GENERIC_ENABLE, /**< [Disable (default), Enable] -> max aggregation (left to right). */
PM_NPA_SW_MODE_LDO__NPM, /**< [BYPASS, IPEAK (default), NPM] -> max aggregation (left to right). */
PM_NPA_PIN_CONTROL_ENABLE__NONE, /**< [NONE, EN1, EN2, EN3, EN4] -> ORed value of list. */
PM_NPA_PIN_CONTROL_POWER_MODE__NONE, /**< [NONE, EN1, EN2, EN3, EN4, SLEEPB] -> ORed value of list. */
2, /**< Perpherial Index */
0, /**< Primary (0) or Secondary (1) PMIC */

0, /**< If((old sw_en == disable) && (new sw_en == enable) || new pc_en == enable then ldo_en_trans = true else ldo_en_trans = false */
PM_NPA_BYPASS_DISALLOWED, /**< [Allowed (default), Disallowed]*/
0, /**< reserve 1 - for 32 bit boundary */
0, /**< [X uV] -> max aggregation. */
400, /**< [X mA] -> summed aggregation. */
100, /**< [X uV] -> voltage headroom needed. */
1250000, /**< [X uV] -> max aggregation. */
//PM_NPA_CORNER_MODE__NONE, /**< [None, Level1 (Retention), Level2, Level3, Level4, Level5, Level6 (SuperTurbo), Not Used] */
},
/**< Mode: PMIC_NPA_MODE_ID_DDR_HIGH_SPEED*/
{
PM_NPA_GENERIC_ENABLE, /**< [Disable (default), Enable] -> max aggregation (left to right). */
PM_NPA_SW_MODE_LDO__NPM, /**< [BYPASS, IPEAK (default), NPM] -> max aggregation (left to right). */
PM_NPA_PIN_CONTROL_ENABLE__NONE, /**< [NONE, EN1, EN2, EN3, EN4] -> ORed value of list. */
PM_NPA_PIN_CONTROL_POWER_MODE__NONE, /**< [NONE, EN1, EN2, EN3, EN4, SLEEPB] -> ORed value of list. */
2, /**< Perpherial Index */
0, /**< Primary (0) or Secondary (1) PMIC */
0, /**< If((old sw_en == disable) && (new sw_en == enable) || new pc_en == enable then ldo_en_trans = true else ldo_en_trans = false */
PM_NPA_BYPASS_DISALLOWED, /**< [Allowed (default), Disallowed]*/
0, /**< reserve 1 - for 32 bit boundary */
0, /**< [X uV] -> max aggregation. */
700, /**< [X mA] -> summed aggregation. */
100, /**< [X uV] -> voltage headroom needed. */
1250000, /**< [X uV] -> max aggregation. */
//PM_NPA_CORNER_MODE__NONE, /**< [None, Level1 (Retention), Level2, Level3, Level4, Level5, Level6 (SuperTurbo), Not Used] */
},
};

- 7. VDD_CX voltage to increase from NOMINAL to NOMINAL_HIGH when BIMC running at 748.8Mhz

  [SBL1]
  boot_images/core/systemdrivers/clock/hw/msm8937/src/ClockSBLConfig.c
  Clock_SBLConfigType Clock_SBLConfigData =

```
{
<snip>

/* BIMC configuration BIMC_Cfg */
.BIMC_Cfg_Feero =
{
<snip>
{ /* BIMC configuration BIMC_Cfg : at 748.8 Mhz */
- 748800000, RAIL_VOLTAGE_LEVEL_NOMINAL, &BIMCPLLConfig[6],
+ 748800000, RAIL_VOLTAGE_LEVEL_NOMINAL_HIGH, &BIMCPLLConfig[6],
{ /* Single clock root for BIMC and DDR. BIMC_GPU must be atleast half of BIMC */
{0},
{ HWIO_ADDR(GCC_DDR_CMD_RCGR), MUX_BIMC, SRC_RAW, 2, 0, 16, 0},/* GCC
DDR Mux, ddr_clk @ 748.8MHZ */
{ HWIO_ADDR(GCC_BIMC_GPU_CMD_RCGR), MUX_BIMCGPU, SRC_GPLL0, 5, 0, 0
, 0} /* GCC BIMC GPU Mux, BIMC_GPU_clk at 320MHZ */
}
},
<snip>
{0},
},


[RPM]
rpm_proc/core/systemdrivers/clock/config/msm8937/ClockBSP.c
ClockMuxConfigType BIMCClockConfig[] =
{
<snip>
- { 748800000, { HAL_CLK_SOURCE_RAW1, 2, 0, 16, 0 },
CLOCK_VREG_LEVEL_NOMINAL, 0, CHIPINFO_FAMILY_UNKNOWN, &
BIMCPLLConfig[6]},
+ { 748800000, { HAL_CLK_SOURCE_RAW1, 2, 0, 16, 0 },
CLOCK_VREG_LEVEL_NOMINAL_PLUS, 0, CHIPINFO_FAMILY_UNKNOWN, &
BIMCPLLConfig[6]},
<snip>
{ 0 }
};
```

- 8. Increase static margin on CX on some modes, for example, increase 25mV on SVS+ and NORMINAL mode:

  \boot_images\core\power\cpr\common\target\8937\ cpr_enablement_bsp.c:

// Cx config

static cpr_enablement_versioned_rail_config_t TSMC_8937_versioned_cpr_enablement =

{

.hw_versions =

{

.foundry_range = (const cpr_config_foundry_range[])

{

// Foundry Chip Min Rev Chip Max Rev CPR Min Rev CPR Max Rev

{CPR_FOUNDRY_TSMC, CPR_CHIPINFO_VERSION(0,0), CPR_CHIPINFO_VERSION(0xFF, 0xFF), 0, 0xFF},

},

.foundry_range_count = 1,

},

.enablement_init_params = &CPR_ENABLE_CLOSED_LOOP,

.supported_level = (cpr_enablement_supported_level_t[])

{

//Mode Static-Margin (mV) Custom static margin function aging_scaling_factor

{CPR_VOLTAGE_MODE_SVS, 25, NULL, 1},

{CPR_VOLTAGE_MODE_SVS_L1, 50+25, NULL, 1},

{CPR_VOLTAGE_MODE_NOMINAL, 50+25, NULL, 1},

{CPR_VOLTAGE_MODE_NOMINAL_L1, 25, NULL, 1},

{CPR_VOLTAGE_MODE_SUPER_TURBO, 63, NULL, 1},

},

- 9. Test by pegging Cx and Mx to Turbo

\rpm_proc\core\power\railway_v2\src\8937\railway_config.c
Code change:-

```
void railway_init_proxies_and_pins(void)
{
+ const int cx_rail_id = rail_id("vddcx");
+ assert(RAIL_NOT_SUPPORTED_BY_RAILWAY != cx_rail_id);
+ railway_voter_t cx_pin = railway_create_voter(cx_rail_id, true,
RAILWAY_RPM_INIT_VOTER);
+ railway_corner_vote(cx_pin, RAILWAY_SUPER_TURBO);
```

Please NOTE that if we peg CX at Super Turbo, MX will also will peg to the same corner voltage.