Dump GPIO status to RPM log

- **Applicable platform: General**

RPM Code Modification for GPIO Dump

- **rpm_proc/core/api/debugtrace/tracer_event_ids.h**

```
typedef enum {
……
        BUS_EVENT_ICB_LAST=78,
        TRACER_EVENT_ID_MAX,
        +SLEEP_GPIO_DUMP = 0x3FF,
        TRACER_EVENT_RESERVE_LAST=0x400,
        TRACER_EVENT_ALL=0x7FFFFFFF,
} tracer_event_id_t;
```

RPM Code Modification for GPIO Dump

- **Add below codes in rpm_proc/core/power/sleep/src/lpr_definition_uber.c**

```
#define TLMM_BASE                                   0x61000000
#define TLMM_CSR_REG_BASE1                          (TLMM_BASE      + 0x00010000)
#define HWIO_TLMM_GPIO_CFGn_ADDR(n)                 (TLMM_CSR_REG_BASE      + 0x00000000 + 0x1000 * (n))
#define HWIO_TLMM_GPIO_IN_OUTn_ADDR(n)              (TLMM_CSR_REG_BASE      + 0x00000004 + 0x1000 * (n))
```

//This is special for MSM8996, please modify it for other platform, you can get the macro definition from rpm_proc\core\power\gpio\target\<target name>\tlmm_hwio.h.

```
void gpio_dump()
{
        int num,i;
        volatile uint32 cfg, inout, val;

        num = 149; //8996 gpio[0:149] please modify it for your platform.
        for (i = 0; i <= num; i++)
        {
                cfg = *(volatile uint32*)HWIO_TLMM_GPIO_CFGn_ADDR(i);
                inout = *(volatile uint32*)HWIO_TLMM_GPIO_IN_OUTn_ADDR(i);
                val = ((cfg << 16)&0xffff0000) | (inout&0xffff);
                SWEVENT(SLEEP_GPIO_DUMP, i, val);
        }
}
```

RPM Code Modification for GPIO Dump

- **rpm_proc/core/power/sleep/src/lpr_definition_uber.c**

```
void deep_sleep_enter(bool mock_vdd_min)
{
  uint64 sleep_duration;
  pm_err_flag_type pmic_err;

  sleep_result = SLEEP_SUCCESS;

  rob_set_recording_speed (cpu_current_speed());
  rob_mark_event (ROB_EVENT_VDD_MIN_ENTER);

  +gpio_dump();   //If you want to dump GPIO dump before real VDD Min enter, add dump function in there.

#ifdef DDR_LPR_TRACING
  if(sleep_ddr_active())
```

Then rebuild the RPM image, which is ready for RPM code's modification.

RPM Code Modification for GPIO Dump

- **Rpm_proc\core\power\rpm\debug\scripts\rpm_parser.py**

```
class SLEEP_MSG_x149:
    __metaclass__ = Parser
    id = 0x149
    def parse(self, data):
        return 'mpm_actual_wakeup_time: (timetick: 0x%0.8x%0.8x)' % (data[1], data[0])

+class SLEEP_MSG_0x3FF :
    +__metaclass__ = Parser
    +id = 0x3ff
    +def parse(self, data):
        +return 'SLEEP_GPIO_DUMP: 0x%0.8x 0x%0.8x' % (data[0], data[1])
```

Ramdump files collection

- **Make sure device going into sleep, and trigger crash mode by ps_hold/volume- key, and save the ramdump files by QPST**

  - **ps_hold:** pull down ps_hold pin to GND within 200s, then device will enter crash mode

  - **Press Volume- key**
    - **Enable spmi debugfs write permission.**

    1) Disable “CONFIG_MSM_SPMI_DEBUGFS_RO“ in perf defconfig and rebuild&flash boot.img。

    # CONFIG_MSM_SPMI_DEBUGFS_RO is not set

    2) For sdm660, enable “CONFIG_REGMAP_ALLOW_WRITE_DEBUGFS” in perf defconfig and rebuild&flash boot.img.

    - **Config PMIC setting by adb command.**

cd /sys/kernel/debug/spmi/spmi-0;

echo 0x844 > address && echo 1 > count && echo 0x00 > data && echo 0x845 > address && echo 0x00 > data && echo 0x846 > address && echo 0x01 > data && echo 0x847 > address && echo 0x80 > data

sdm660:

cd /sys/kernel/debug/regmap/spmi0-00

echo 0x844 > address && echo 1 > count && echo 0x00 > data && echo 0x845 > address && echo 0x00 > data && echo 0x846 > address && echo 0x01 > data && echo 0x847 > address && echo 0x80 > data

  - **Ensure device enter sleep mode, press Volume- key**

Parse RPM log with hansei scripts.

- **Please refer rpm overview document for the details. For msm8976, page 42 & 43 of DCN#80-NU154-10**
  - Hansei RAM dump parser is a tool that parses debug information out of the RAM dump. It generates RPM logs, NPA logs, master status, resource states, etc.
  - Install Hansei RAM dump parser:
    1. Install Python 2.7.x (not 2.6.x).
       Check version by running `python -V`.
    2. Install pyelftools library to support the ARM compiler; the mainline version does not work. Instead, use:
       https://bitbucket.org/pplesnar/pyelftools-pp
       Install command – `python setup.py install`
  - Hansei script release
    - Released since RPM 100
    - Location – rpm_proc\core\bsp\rpm\scripts\hansei\
  - Usage

    ```
    hansei.py [-h] --elf rpm.elf --output <OutPut Path> dumpfile <dump path>
    ```

    Dump path should contain `rpm_code_ram.bin rpm_data_ram.bin rpm_msg_ram.bin`

Parse RPM log with hansei scripts.

- **RPM log will prints GPIO status as below**

627.988211: SLEEP_GPIO_DUMP: 0x00000000 0x00840000
627.988231: SLEEP_GPIO_DUMP: 0x00000001 0x00840000
627.988252: SLEEP_GPIO_DUMP: 0x00000002 0x00840001
627.988272: SLEEP_GPIO_DUMP: 0x00000003 0x00840000
627.988293: SLEEP_GPIO_DUMP: 0x00000004 0x02080001
627.988314: SLEEP_GPIO_DUMP: 0x00000005 0x02080000
627.988334: SLEEP_GPIO_DUMP: 0x00000006 0x004c0001
627.988355: SLEEP_GPIO_DUMP: 0x00000007 0x004c0001
627.988376: SLEEP_GPIO_DUMP: 0x00000008 0x02010000
627.988396: SLEEP_GPIO_DUMP: 0x00000009 0x02010003
627.988417: SLEEP_GPIO_DUMP: 0x0000000a 0x00050000
627.988437: SLEEP_GPIO_DUMP: 0x0000000b 0x00010000

Parse GPIO status from RPM log

- **RPM log will prints GPIO status as below**

627.988211: SLEEP_GPIO_DUMP: 0x00000000 0x00840000
627.988231: SLEEP_GPIO_DUMP: 0x00000001 0x00840000
627.988252: SLEEP_GPIO_DUMP: 0x00000002 0x00840001
627.988272: SLEEP_GPIO_DUMP: 0x00000003 0x00840000
627.988293: SLEEP_GPIO_DUMP: 0x00000004 0x02080001

- **Use rpmlog_to_gpiodump.py to parse GPIO status**
  - Find the attachment for script file rpmlog_to_gpiodump.py (or file a case for this script)
  - Run command “rpmlog_to_gpiodump.py -l rpm-log.txt”, then gpiodump.txt will be generated.
  - The contents in gpiodump.txt

```
using log file rpm-log1.txt
gpio [0]    is func 1   INPUT              NO PULL
gpio [1]    is func 1   INPUT              NO PULL
gpio [2]    is func 1   INPUT              NO PULL
gpio [3]    is func 1   INPUT              NO PULL
gpio [4]    is func 2   OUTPUT   LOW       NO PULL
gpio [5]    is func 2   OUTPUT   LOW       NO PULL
```