**Sleep or delay function in pmic(bootloader)**

/\* Add 1sec Delay to check user press a power key\*/

pm\_busy\_wait(1000\*1000);

**link for raise a ticket for Qcom:**

https://qualcomm-cdmatech-support.my.salesforce.com/5003A00000tKzF7

**1000 nit display:**

In lighting, the **nit** is a unit of visible-light intensity, commonly used to specify the **brightness** of a cathode ray tube or liquid crystal **display** computer **display**. One **nit** is equivalent to one candela per square meter.

**Duty cycle:**

the cycle of operation of a machine or other device which operates intermittently(occurring at irregular intervals; not continuous or steady.) rather than continuously.

D = (pulse width / T) \*100 ;

[uint32](http://157.235.208.175:8080/source/s?defs=uint32&project=SDM660O-PR) **[uWLEDValue](http://157.235.208.175:8080/source/s?refs=uWLEDValue&project=SDM660O-PR)** = (0xFFF \* [**pBacklightConfig**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/MDPPlatformLib/MDPPlatformLib.c#pBacklightConfig)->[uLevel](http://157.235.208.175:8080/source/s?defs=uLevel&project=SDM660O-PR)) / 100; // Calculate duty cycle based on 12 bit mode

**log message in bootloader:**

[boot\_log\_message](http://157.235.208.175:8080/source/s?defs=boot_log_message&project=SDM660O-PR)("pm\_sbl\_chg\_init, Start");

**pm\_log\_message("detecting sku... %d", Zebra\_sku);**

**PmicLib Directory:**

**PMIC**is **fully** programmable via the SPI/I2C interface

**PmicLib/target contains PMIC device initialization function where initial PMIC, SBL settings are configured and entry point of the pmic elf image**

**PmicLib/drivers will fetch the data from the registers and maintains some top-level API wrappers later used by PmicLib/app.**

**UEFI PMIC FLOW:**

**Qcom chr app**

**|**

**Qcom chr Dex**

**|**

**Chr Lib**

**|**

**Pmic Dex**

**|**

**Pmic Lib**

**|**

**SPMI Dex**

**|**

**PMIC hardware**

**Example(flow of Pmic and UEFI):**

**pm\_rgb\_led\_config :** ( it’s an API to configure the leds)

[**pm\_sbl\_schg\_config\_red\_led**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/app/chg/src/pm_app_chgr.c#pm_sbl_schg_config_red_led)**(**/[PmicLib](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/)/[app](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/app/)/chr**)** (this api used to produce some delay if not similar to next level function)

**|**

[**pm\_rgb\_led\_config**](http://157.235.208.175:8080/source/s?refs=pm_rgb_led_config&project=SDM660O-PR)**(**[PmicLib](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/)/[app](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/app/)/[rgb](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/app/rgb/)/**)**

**|**

[**pm\_rgb\_enable**](http://157.235.208.175:8080/source/s?refs=pm_rgb_enable&project=SDM660O-PR)**(**/[PmicLib](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/)/[drivers](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/drivers/)/[rgb](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/drivers/rgb/)/**)**

**|**

[**pm\_comm\_write\_byte\_mask**](http://157.235.208.175:8080/source/s?refs=pm_comm_write_byte_mask&project=SDM660O-PR)**(**[PmicLib](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/)/[comm](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/comm/)/[spmi\_lite](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Library/PmicLib/comm/spmi_lite/)**)**

**|**

[**SpmiBus\_ReadLong**](http://157.235.208.175:8080/source/s?refs=SpmiBus_ReadLong&project=SDM660O-PR) **(low level spmi bus)**

**|**

[**debugExecute**](http://157.235.208.175:8080/source/s?refs=debugExecute&project=SDM660O-PR) **(low level spmi bus)**

**|**

**(very low level Spmi bus)**

err\_flag = pm\_rgb\_led\_config(device\_index,

PM\_RGB\_1,

PM\_RGB\_SEGMENT\_R,

PM\_RGB\_VOLTAGE\_SOURCE\_VPH,

PM\_RGB\_DIM\_LEVEL\_HIGH,

toggle\_led);

link for TRM for PMIC:

<https://createpoint.qti.qualcomm.com/search/contentdocument/stream/463093?refererRoute=search%2FsearchArgs%2Fq%7C%7Cpm660%20hardware%20register%7C%7Crows%7C%7C10%7C%7CsortField%7C%7Cscore%7C%7CsortOrder%7C%7Cdesc%7C%7CproductkitLevelId%7C%7C341%7C%7Cfq_productkitNames%7C%7CSDM660%20Android%20Smartphone%20(Platform)&dcn=80-P7905-2X&currentPage=1&itemTotalIndex=1>

Serial logs flow:

Primary boot interface (eMMC)

|

PBL,start

elf\_loader\_entry, Start

PBL, End

|

SBL1, Start

PMIC Image Loaded, Delta

PM: [Helios] XBL Disabling charging (if fully charged)

PM: SBL Charging in progress.... (if not charged)

PM: Set max float voltage from battMem = 4200

PM: [Helios] XBL Re-Enable charging (for charging )

Pre\_DDR\_clock\_init, Start

Bootup frequency set to 1296000

RPM Image Loaded, Delta - (219340 Bytes)

SBL1, End

|

UEFI Start [ 2380] SEC

RAM Entry 0 : Base 0x0000000080000000 Size 0x0000000080000000

Load CPU 0 Slp: 0x82, Int: 0x10

Memory Base Address: 0x80000000

Decompressing kernel image start: 14269 ms

UEFI End

|

kernel

PBL:

[QcomPkg](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/)/[XBLLoader](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/XBLLoader/)/[boot\_pbl\_v2.c](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/XBLLoader/boot_pbl_v2.c)

This funcion will parse the PBL timestamp milestones passed to SBL

and insert them into the boot log. Currently PBL's unit of measure is

clock ticks. PBL does not pass the clock frequency yet so it is hard

wired to 19.2 Mhz. Also PBL does not pass the names of each field so for

now reference a structure of our own to get the names which will have

logic ready for the day PBL starts passing them

clock tick:

For personal computers, **clock ticks** generally refer to the main system **clock**, which runs at 66 MHz. This means that there are 66 million **clock ticks** (or cycles) per second. Since modern CPUs run much faster (up to 3 GHz), the CPU can execute several instructions in a single **clock tick**.

SBL:

This below file contains the main control for SBL1 execution.

QcomPkg/Sdm660Pkg/Library/XBLLoaderLib/sbl1\_mc.c

UEFI:

[QcomPkg](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/)/[XBLCore](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/XBLCore/)/[AARCH64](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/XBLCore/AARCH64/)/[ModuleEntryPoint.S](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/XBLCore/AARCH64/ModuleEntryPoint.S)

Above contains the UEFI entry address will be there

/[android](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/)/[bootable](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/)/[bootloader](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/)/[edk2](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/)/[QcomModulePkg](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/)/[Library](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/)/[BootLib](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/)/[BootLinux.c](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/BootLinux.c)

Above file contains the decompression of the kernel image

// These three regions should be reserved in memory map.

[KernelLoadAddr](http://157.235.208.175:8080/source/s?defs=KernelLoadAddr&project=SDM660O-PR) = ([EFI\_PHYSICAL\_ADDRESS](http://157.235.208.175:8080/source/s?defs=EFI_PHYSICAL_ADDRESS&project=SDM660O-PR))(**[BaseMemory](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/BootLinux.c" \l "BaseMemory)** | [PcdGet32](http://157.235.208.175:8080/source/s?defs=PcdGet32&project=SDM660O-PR)([KernelLoadAddress](http://157.235.208.175:8080/source/s?defs=KernelLoadAddress&project=SDM660O-PR)));

[**RamdiskLoadAddr**](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/BootLinux.c#RamdiskLoadAddr) = ([EFI\_PHYSICAL\_ADDRESS](http://157.235.208.175:8080/source/s?defs=EFI_PHYSICAL_ADDRESS&project=SDM660O-PR))(**[BaseMemory](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/BootLinux.c" \l "BaseMemory)** | [PcdGet32](http://157.235.208.175:8080/source/s?defs=PcdGet32&project=SDM660O-PR)([RamdiskLoadAddress](http://157.235.208.175:8080/source/s?defs=RamdiskLoadAddress&project=SDM660O-PR)));

s[DeviceTreeLoadAddr](http://157.235.208.175:8080/source/s?defs=DeviceTreeLoadAddr&project=SDM660O-PR) = ([EFI\_PHYSICAL\_ADDRESS](http://157.235.208.175:8080/source/s?defs=EFI_PHYSICAL_ADDRESS&project=SDM660O-PR))(**[BaseMemory](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/bootable/bootloader/edk2/QcomModulePkg/Library/BootLib/BootLinux.c" \l "BaseMemory)** | [PcdGet32](http://157.235.208.175:8080/source/s?defs=PcdGet32&project=SDM660O-PR)([TagsAddress](http://157.235.208.175:8080/source/s?defs=TagsAddress&project=SDM660O-PR)));

Read real time interrupt status on PMIC:

Status = PmicPONProtocol->GetPonRtStatus(0, IRQType, pGpioButtonPressed);

**How to enable gpio pins:**

[Status](http://157.235.208.175:8080/source/s?defs=Status&project=SDM660O-PR) = [**TLMMProtocol**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c#TLMMProtocol)->[ConfigGpio](http://157.235.208.175:8080/source/s?defs=ConfigGpio&project=SDM660O-PR)(([UINT32](http://157.235.208.175:8080/source/s?defs=UINT32&project=SDM660O-PR))[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?defs=EFI_GPIO_CFG&project=SDM660O-PR)([GpioNumber](http://157.235.208.175:8080/source/s?defs=GpioNumber&project=SDM660O-PR), 0, [GPIO\_INPUT](http://157.235.208.175:8080/source/s?defs=GPIO_INPUT&project=SDM660O-PR), [GPIO\_PULL\_UP](http://157.235.208.175:8080/source/s?defs=GPIO_PULL_UP&project=SDM660O-PR), [GPIO\_2MA](http://157.235.208.175:8080/source/s?defs=GPIO_2MA&project=SDM660O-PR)),

[TLMM\_GPIO\_ENABLE](http://157.235.208.175:8080/source/s?defs=TLMM_GPIO_ENABLE&project=SDM660O-PR));

#**define** **[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?refs=EFI_GPIO_CFG&project=SDM660O-PR)**(**[gpio](http://157.235.208.175:8080/source/s?refs=gpio&project=SDM660O-PR)**, **[func](http://157.235.208.175:8080/source/s?refs=func&project=SDM660O-PR)**, **[dir](http://157.235.208.175:8080/source/s?refs=dir&project=SDM660O-PR)**, **[pull](http://157.235.208.175:8080/source/s?refs=pull&project=SDM660O-PR)**, **[drive](http://157.235.208.175:8080/source/s?refs=drive&project=SDM660O-PR)**) \

  (((**[gpio](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Include/Protocol/EFITlmm.h" \l "gpio)**) & 0x3FF)<< 4 | \

  ((**[func](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Include/Protocol/EFITlmm.h" \l "func)**) & 0xF)| \

  ((**[dir](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Include/Protocol/EFITlmm.h" \l "dir)**) & 0x1) << 14| \

  (([**pull**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Include/Protocol/EFITlmm.h#pull)) & 0x3) << 15| \

  (([**drive**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Include/Protocol/EFITlmm.h#drive))& 0xF) << 17| 0x20000000)

**Learn how to convert int to char:**

/\* Convert part number from uint8 to char first\*/

memcpy(extPartNumStr, extendedPartNumber.extendedPartNumber, sizeof(extPartNumStr));

read data from eeprom to set float voltage

i2cResult = battMemGetAll();

/\* check if data is corrupt or not \*/

is\_data\_valid = check\_battMem\_checksum((uint8 \*)&(SD660BattMem.parameters1725.recordType),

(sizeof(SD660BattMem.parameters1725) - 1),

SD660BattMem.parameters1725.checksum);

threshold voltage:

The **threshold voltage**, commonly abbreviated as Vth, of a field-effect transistor (FET) is the minimum gate-to-source **voltage** VGS (th) that is needed to create a conducting path between the source and drain terminals.

**//DC charger present, if DCIN\_UV\_RT\_STS and DCIN\_OV\_RT\_STS status is 0 (INT\_RT\_STS : 0x1410[3] and [2] == 0)**

**err\_flag |= pm\_schg\_dc\_irq\_status(device\_index, PM\_SCHG\_DC\_IRQ\_DCIN\_UV, PM\_IRQ\_STATUS\_RT, &dcbin\_uv\_status);**

This below function powers off the PMIC

pm\_log\_message("Auto power on, shudown!");

boot\_hw\_powerdown();

**VSP/VSN voltage:**

**Gas gauge and fuel gauge:**

**JEITA: (only for batteries)**

If battery charge current is not consistent we can check in jeita

Float voltage and [Full charge count of the battery in milliamp-hours](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/Documentation/devicetree/bindings/batterydata/batterydata.txt#22) (FV and FCC) can be set in UEFI path: / boot\_images/QcomPkg/Drivers/PmicDxe/PmicSchgProtocol.c

Disable the charging in XBL

**Below can be done in path:** boot\_images/QcomPkg/Library/PmicLib/app/chg/src/**pm\_app\_chgr.c**

//Work-around for OVP issue: charging is by default enabled by hardware and this will lead to safety circuit OVP to trigger.

//Solution is to disable charging in early stage of XBL and re-enabling after reading the charging parameters from memory.

err\_flag |= pm\_schg\_chgr\_set\_chgr\_cmd\_type(device\_index, PM\_SCHG\_CHGR\_CMD\_CHARGING\_ENABLE, TRUE);

**below can be done in path:**  boot\_images/QcomPkg/Library/PmicLib/target/sdm660\_pm660\_pm660l/system/src/**pm\_sbl\_boot\_oem.c**

//Work-around for OVP issue: charging is by default enabled by hardware and this will lead to safety circuit OVP to trigger.

//Solution is to disable charging in early stage of XBL and re-enabling after reading the charging parameters from memory.

pm\_comm\_write\_byte\_mask(0, 0x1051, 0xC0, 0x00, 0);//Configure CHG\_EN active high

pm\_comm\_write\_byte\_mask(0, 0x1042, 0x01, 0x00, 0);//Disable Charging

detect the DC charger status to avoid the shout down condition

path:  [boot\_images/QcomPkg/Library/PmicLib/app/chg/src/pm\_app\_chgr.c](https://gerrit.zebra.com/#/c/120058/4/boot_images/QcomPkg/Library/PmicLib/app/chg/src/pm_app_chgr.c)

//DC charger present, if DCIN\_UV\_RT\_STS and DCIN\_OV\_RT\_STS status is 0 (INT\_RT\_STS : 0x1410[3] and [2] == 0)

err\_flag |= pm\_schg\_dc\_irq\_status(device\_index, PM\_SCHG\_DC\_IRQ\_DCIN\_UV, PM\_IRQ\_STATUS\_RT, &dcbin\_uv\_status);

Bootup the device on charging if user press a power key.

Path:

boot\_images/QcomPkg/Library/PmicLib/app/pon/src/**pm\_app\_key\_press.c**

write one function, call the fun in pm\_app\_chgr.c with below things

if (err\_flag == PM\_ERR\_FLAG\_\_SUCCESS && key\_press\_status == TRUE) {

/\* Add 1sec Delay to check user press a power key\*/

pm\_busy\_wait(1000\*1000);

err\_flag |= pm\_pon\_irq\_status(PM\_DEVICE\_INDEX, PM\_PON\_IRQ\_KPDPWR\_ON, PM\_IRQ\_STATUS\_RT, &key\_press\_status);

Input current limit needs to be set to 2500mA in

Bootloader

boot\_images/QcomPkg/Sdm660Pkg/Settings/PMIC/pm\_config\_target.c

This file contains customizable target specific

\* driver settings & PMIC registers. This file is generated from database functional

\* configuration information that is maintained for each of the targets.

Path:

boot\_images/QcomPkg/Library/PmicLib/target/sdm660\_pm660\_pm660l/system/src/**pm\_sbl\_boot\_oem.c**

//set current limit to 2500mA for DC\_IN mode

err\_flag |= pm\_comm\_write\_byte\_mask(0, 0x1470, 0xFF, 0x64, 0);

//set current limit to 2500mA for USB\_IN mode

err\_flag |= pm\_comm\_write\_byte\_mask(0, 0x1370, 0xFF, 0x64, 0);

config PON\_RESIN\_N\_RESET\_S1\_TIMER(address:0x0844) to 4480ms

path:  [boot\_images/QcomPkg/Library/PmicLib/target/sdm660\_pm660\_pm660l/system/src/pm\_sbl\_boot\_oem.c](https://gerrit.zebra.com/#/c/103682/2/boot_images/QcomPkg/Library/PmicLib/target/sdm660_pm660_pm660l/system/src/pm_sbl_boot_oem.c)

err\_flag |= pm\_app\_pon\_reset\_cfg( PM\_APP\_PON\_RESET\_SOURCE\_KPDPWR, PM\_APP\_PON\_CFG\_DVDD\_HARD\_RESET, 4408, 1000); //PON KPDPWR PON Reset configuration

err\_flag |= pm\_pon\_reset\_source\_ctl(0, PM\_PON\_RESET\_SOURCE\_KPDPWR, PM\_ON);

Recovery mode using PTT btton instead of Vol UP

Step 1 :

Define a macro in path: boot\_images/MdePkg/Include/Protocol/**SimpleTextInEx.h**

#define SCAN\_PTT 0x0138

Step 2:

Give an entry in key type path: / boot\_images/QcomPkg/Include/Library/ButtonsLib.h

// key types

typedef enum {

PWR,

VOL\_UP,

VOL\_DOWN,

PTT,

}KEY\_TYPE;

Step 3 :

Path:  boot\_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c

Enable gpio and read gpio status using

Status = TlmmEnableInput(PTT\_BUTTON\_GPIO);

Status = TlmmReadGpioStatus(PTT\_BUTTON\_GPIO, &ButtonPressed );

Fuel Gauge register setting:

Description: battery charging calibration for frozone battery

Solution:

a. Disable TYPE-C mode and enable micro USB mode (Reg addr:0x1358)

b. DCIN has priority / Forces QC3.0 / HVDCP Auto Increment Mode Enabled by Command Bit (Reg addr:0x1362)

c. Disable WIPWR for DC\_IN mode (Reg addr:0x1495)

d. Set AICL threshold to 4.7V for DC\_IN mode (Reg addr:0x1481)

e. Set Minimum system voltage to 3.6V (Reg addr:0x1683)

**err\_flag |= pm\_comm\_write\_byte\_mask(0, 0x1358, 0xFF, 0x49, 0);**

**Thermistor**

**Thermistor**. ... **Thermistors** are widely used as inrush current limiters, temperature sensors (negative temperature **coefficient**or NTC type typically), self-resetting overcurrent protectors, and self-regulating heating elements (positive temperature**coefficient** or PTC type typically).

GPIO CONFIG:

Configure the gpio of the scan key, Implement key detection to enter fastboot and recovery mode can be done using below functions

We should implement in path:  boot\_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/**ButtonsLib.c**

To detect key there is a array

\*(pButtonArray + 4) = ButtonPressed;

If single device

// due to single key, here ignore to check other keys.

\*(pButtonArray + 0) = FALSE;

\*(pButtonArray + 2) = FALSE;

\*(pButtonArray + 3) = FALSE;

Enable the GPIO for Input

[Status](http://157.235.208.175:8080/source/s?defs=Status&project=SDM660O-PR) = [**TLMMProtocol**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c#TLMMProtocol)->[ConfigGpio](http://157.235.208.175:8080/source/s?defs=ConfigGpio&project=SDM660O-PR)(

  ([UINT32](http://157.235.208.175:8080/source/s?defs=UINT32&project=SDM660O-PR))[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?defs=EFI_GPIO_CFG&project=SDM660O-PR)([GpioNumber](http://157.235.208.175:8080/source/s?defs=GpioNumber&project=SDM660O-PR), 0, [GPIO\_INPUT](http://157.235.208.175:8080/source/s?defs=GPIO_INPUT&project=SDM660O-PR), [GPIO\_PULL\_UP](http://157.235.208.175:8080/source/s?defs=GPIO_PULL_UP&project=SDM660O-PR), [GPIO\_2MA](http://157.235.208.175:8080/source/s?defs=GPIO_2MA&project=SDM660O-PR)),

[TLMM\_GPIO\_ENABLE](http://157.235.208.175:8080/source/s?defs=TLMM_GPIO_ENABLE&project=SDM660O-PR));

Enable the GPIO for Output

[Status](http://157.235.208.175:8080/source/s?defs=Status&project=SDM660O-PR) = [**TLMMProtocol**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c#TLMMProtocol)->[ConfigGpio](http://157.235.208.175:8080/source/s?defs=ConfigGpio&project=SDM660O-PR)(

  ([UINT32](http://157.235.208.175:8080/source/s?defs=UINT32&project=SDM660O-PR))[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?defs=EFI_GPIO_CFG&project=SDM660O-PR)([GpioNumber](http://157.235.208.175:8080/source/s?defs=GpioNumber&project=SDM660O-PR), 0, [GPIO\_OUTPUT](http://157.235.208.175:8080/source/s?defs=GPIO_OUTPUT&project=SDM660O-PR), [GPIO\_NO\_PULL](http://157.235.208.175:8080/source/s?defs=GPIO_NO_PULL&project=SDM660O-PR), [GPIO\_2MA](http://157.235.208.175:8080/source/s?defs=GPIO_2MA&project=SDM660O-PR)),

[TLMM\_GPIO\_ENABLE](http://157.235.208.175:8080/source/s?defs=TLMM_GPIO_ENABLE&project=SDM660O-PR));

Read gpio status on TLMM

[Status](http://157.235.208.175:8080/source/s?defs=Status&project=SDM660O-PR) = [**TLMMProtocol**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c#TLMMProtocol)->[GpioIn](http://157.235.208.175:8080/source/s?defs=GpioIn&project=SDM660O-PR)(

  ([UINT32](http://157.235.208.175:8080/source/s?defs=UINT32&project=SDM660O-PR))[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?defs=EFI_GPIO_CFG&project=SDM660O-PR)([GpioNumber](http://157.235.208.175:8080/source/s?defs=GpioNumber&project=SDM660O-PR), 0, [GPIO\_INPUT](http://157.235.208.175:8080/source/s?defs=GPIO_INPUT&project=SDM660O-PR), [GPIO\_PULL\_UP](http://157.235.208.175:8080/source/s?defs=GPIO_PULL_UP&project=SDM660O-PR), [GPIO\_2MA](http://157.235.208.175:8080/source/s?defs=GPIO_2MA&project=SDM660O-PR)),

  &[GpioStatus](http://157.235.208.175:8080/source/s?defs=GpioStatus&project=SDM660O-PR));

Write gpio status on TLMM

[Status](http://157.235.208.175:8080/source/s?defs=Status&project=SDM660O-PR) = [**TLMMProtocol**](http://157.235.208.175:8080/source/xref/SDM660O-PR/BOOT.XF.1.4/boot_images/QcomPkg/Sdm660Pkg/Library/ButtonsLib/ButtonsLib.c#TLMMProtocol)->[GpioOut](http://157.235.208.175:8080/source/s?defs=GpioOut&project=SDM660O-PR)(

  ([UINT32](http://157.235.208.175:8080/source/s?defs=UINT32&project=SDM660O-PR))[EFI\_GPIO\_CFG](http://157.235.208.175:8080/source/s?defs=EFI_GPIO_CFG&project=SDM660O-PR)([GpioNumber](http://157.235.208.175:8080/source/s?defs=GpioNumber&project=SDM660O-PR), 0, [GPIO\_OUTPUT](http://157.235.208.175:8080/source/s?defs=GPIO_OUTPUT&project=SDM660O-PR), [GPIO\_NO\_PULL](http://157.235.208.175:8080/source/s?defs=GPIO_NO_PULL&project=SDM660O-PR), [GPIO\_2MA](http://157.235.208.175:8080/source/s?defs=GPIO_2MA&project=SDM660O-PR)),

[GpioStatus](http://157.235.208.175:8080/source/s?defs=GpioStatus&project=SDM660O-PR));

<https://gerrit.zebra.com/#/c/99937/>

Enabling 5V Buck–boost converter

Set float voltage

[**JEITA**](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/Documentation/devicetree/bindings/power/supply/qcom/qpnp-fg-gen3.txt#142)