Kernel make file

arch/arm64/configs/**helios\_defconfig**

how to disable a driver in dtsi

qbt1000: qcom,qbt1000 {

// some code

};

&qbt1000 {

status = "disabled"; //to disable

};

Dtsi:

arch/arm/boot/dts/symbol/features/**helios-leds.dtsi contains the led functionalities (**1. Give default duty cycle parameter for RGB leds.

2. Enable blink functionality for each led.

3. This patch could be applied in BB and EV.**)**

arch/arm/boot/dts/symbol/**helios-bigboard.dtsi includes above dtsi file**

# [What is the difference between module\_init and subsys\_initcall while initializing the driver?](https://stackoverflow.com/questions/15541290/what-is-the-difference-between-module-init-and-subsys-initcall-while-initializin)

In case device is not booting up there may be chance of issue in below path also

drivers/power/supply/qcom/**qpnp-smb2.c**

below function is used to fetch the data from dts to kernel

const struct of\_device\_id \*of\_match\_node(const struct of\_device\_id \*matches,

const struct device\_node \*node)

\*pinctrl driver communicate with the i2c driver

\*device/symbol/valueadds  contains .so files for a added feature

Below function will get the irq number

int gpiod\_to\_irq(const struct gpio\_desc \*desc)

\*[gpio-keys.kl](https://gerrit.zebra.com/" \l "/c/141020/2/gpio-keys.kl) and .idc files contains keycodes

To create a enum or array dynamically in a driver :

1. In a header file we have to choose a specific structure to declare a pointer
2. In probe or some specific func, allocate dynamically memory for that pointer
3. If any error we need to kfree the memory
4. Later we can use the enum or array in specific use.

Below content taken from:

message:"keypad" branch:sdm660-helios-platform project:ZEUS/Common/kernel/msm-4.4 status:merged

* For every keycode there is a appropriate scan code will be available

Example:

hw\_trigger\_l {

label = "SCAN";

scancode = <0xF1>;

//flags = <0>;

flags = <(MPAKEYPAD\_KEYMAP\_ENTRY\_FLAG\_WAKEUP)>;

wakeup = <0>; // if wakeup source is 0 enabling device wkup while pressing the butto

key\_entry{

keycode = <310>;//KEY\_SYMBOL\_SCAN\_L1

flags = <0>;

};

};

Kernel would get a panic one of the reasons is that it would get a conflict with HW Beeper DTSI configuration.

* Instead of module\_init() we can use late\_initcall() it will change the probe time

Description: When DUT is resumed, shift lock key icon should not be seen

Solution: Release the shift key status while DUT is resumed

Example: info->modifier\_state[MPAKEYPAD\_MODIFIERS\_SHIFT] &=

~MPAKEYPAD\_MODIFIERS\_FLAGS\_TOGGLE\_MASK;

} in resume function

\*when we press the key in the keyboard we can get a sound to add the feature we need to add the symbol,keypadsound = <1>; in dtsi file.

[if (property\_get\_int32("](http://157.235.208.175:8080/source/xref/SDM660O-PL/LA.UM.6.2/LINUX/android/frameworks/native/services/inputflinger/InputReader.cpp" \l "2494)**ro.config.device.keypadsound**",0) != 0) {  in .cpp file

Below content taken by

message:"galactus" branch:sdm660-helios-platform project:ZEUS/Common/kernel/msm-4.4 status:merged

1) Description: Device date and time will offset to about ~03:08PM (over 3 minutes faster)

Solution: On PT4 board, using externel RTC to sync time to fix it.

Means in dtsi file we needto include the rtc-qpnp.dtsi file

2) Description: Although HW PWM can be enabled/disabled, PWM duty cycle cannot be configured accurately except 0% and 100%.

Solution: use i2c\_smbus\_write\_word\_data to write data into register

ret = i2c\_smbus\_write\_word\_data(lm->client, reg, buf);

3) Description: Add GPIO6 of PM660 support as 32k clock input to counter circute for hw beeper driver

Commit Reason: HW Beep counter circuit shall have 32K input via PM660 GPIO6

4) Description: DC IN should have higher priority than USB IN

Solution: Modify the INPUT\_PRIORITY

We can add changes in dtsi file

Qcom,input\_priority = <0x00>;

1. Dedicated **Charging** Port (**DCP**)

zebra,dcp\_current = <2400000>;

1. Description: OTG power should be provided individually when system connecting to DC/AC in based on PT4/EV HW

Solution: Under OTG, when VBUS regulator supplied, enable OTG\_OUT\_EN to enable OCP. When VBUS regulator gone,disable OTG\_OUT\_EN to disable OCP.

1. Add System Property for ro.config.device.touch\_term

touch\_term indicates which devices support All-Touch Terminal Emulation applications, such as Wavelink Velocity.

For a new system property we need to add code in the drivers/platform/symbol/**sw\_ver\_entries.c file**

->case HW\_CONFIG\_TOUCH\_TERM:

return hwconfig->touch\_term;

->static ssize\_t touch\_term\_show(struct kobject \*kobj, struct kobj\_attribute \*attr, char \*buf)

{

return sprintf(buf,"%d",GetHardwareConfiguration(HW\_CONFIG\_TOUCH\_TERM));

}

->static struct kobj\_attribute touch\_term\_attribute = \_\_ATTR(touch\_term, 0444, touch\_term\_show, NULL);

->&touch\_term\_attribute.attr,

->pdata->HWconfig.touch\_term = process\_sw\_ver\_entry(np, "symbol,touch\_term");

And in header file int touch\_term;,

HW\_CONFIG\_TOUCH\_TERM,

1. Description: Some SDXC high speed card have probability can't read Acceptance requirement: Read all SDXC well

Solution: Increase driving current of sdc2\_data and sdc2\_cmd from 10mA to 16mA.

NOTE:

It only improves to read more cards.The signal trace is over qcom suggestion range. It's HW limitation.

1. device is waking up when in suspend mode when scan key is pressed during HOT SWAP

Description:device is waking up when in suspend mode when scan key is pressed during HOT SWAP

Solution: Block key event when battery not present

If we want to block any keyevent, there is a function called mpa\_gpio\_keys\_report\_event in **mpa-gpio-keys.c** driver file we should return with a condition like

if (battery\_present() == 0{

return ;

}

1. Descptn:Device should resume when battery plugged in

Solution: Define LED to indicate user the DUT is in BSM

Like

IN drivers/power/supply/qcom/**qpnp-smb2.c** driver probe function

if (GetHardwareConfiguration(HW\_CONFIG\_BSWAP) == HOTSWAP\_SUPPORT) {

return rc = enable\_irq\_wake();

}

1. To create a sysfs for IR sensor

The IR sensor now reports the object temperature rather than ths ambient temperature,

as its primary temperature. A second sysfs entry was added to report the ambient temperature.

For that should create a structure

1) sysfs\_create\_group, sysfs\_remove\_group

2)static struct mlx90632\_attribute temp2\_attr = \_\_ATTR(temp2, 0444, mlx90632\_temperature2\_show, 0);

3)should add &temp2\_attr.attr, in static struct attribute \*MLX90632\_attrs[] = {

} structure.

4)should have the definition of the function mlx90632\_temperature2\_show();

12) [Galactus] [HS] Charging LED display abnormal when removing main battery for 5 minutes then inserting battery again to charge via AC or cradle

Description:

Try to read data when battery is unstable during hot-swap

Solution:

Use battery re-detection to recover i2c communication failure

To solve this issue we need to add code in drivers/power/supply/qcom/qpnp-smb2.c

13) Description: Tuning display brightness by ambient light sensor

Solution: Modify Galactus panel for display brightness max level

This can be done in dtsi file

14)if we want to change the any sensor names we can do in dtsi files because names are generic in nature

15)jiffies – a short time used in timer concept

16)

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

this file contains all the commands related to scanner.

Example : getvar,scanner

C:\Users\MK4379\Downloads\device\_recovery\helios\_user\_GMS\_Rel\_Key\_release\_2018-08-10-1901\_product\_SE\Images\prog\_emmc\_ufs\_firehose\_Sdm660\_ddr.elf

This elf file contains the firmware info to download from QFIL (programmed path)

Rawprogram – rawprogram0.xml

Patch -patch0.xml

What is an **MBN file**?

Multi boot binary firmware image used by a variety of Android devices, mainly Samsung mobile phones; contains binary data for a device's memory partitions, such as the resources and power manager, secondary bootloader, AP bootloader, and trust zone; used to update firmware on an Android device.

The first time boot ( after update image or Factory Reset) the UUT will pop up the error massage "Bluetooth keeps stopping"

After reboot ,will pop up "Bluetooth has stoped".

Turn on the BT,will pop up "Bluetooth keeps stopping".

Fail rate?: 100%

Solution: Remove "disable PM660 RTC" in commit

/delete-node/ &pm660\_rtc; remove this from dtsi file

Learn about pull up, pull down, no pull in gpio

How PWD Works?

As its name suggests, **pulse width modulation** speed control **works** by driving the motor with a series of “ON-OFF” **pulses** and varying the duty cycle, the fraction of time that the output voltage is “ON” compared to when it is “OFF”, of the **pulses**while keeping the frequency constant.

# [What is the difference between module\_init and subsys\_initcall while initializing the driver?](https://stackoverflow.com/questions/15541290/what-is-the-difference-between-module-init-and-subsys-initcall-while-initializin)

The difference relates to timing, or more precisely, order of execution. That is, the procedure declared as subsys\_initcall is guaranteed to be executed before the procedure declared as module\_init. This ordering ensures that subsystem and platform drivers are initialized before device drivers try to utilize the former's functionality (e.g. a device driver registers as a subsystem device).

Note that subsys\_initcall() can only be used by a built-in (statically linked) module.  
module\_init can be used by either built-in or loadable modules.

platform\_driver\_register

use a macro to avoid include chaining to get THIS\_MODULE,it is used to register a driver for platform-level devices

registering gpio pins into dtsi file

&soc {

beep\_pwm: beep\_pwm {

compatible = "beep\_pwm";

beep-pwm-gpios =

<&pm660\_gpios 2 0>, /\* HB Vol 1 \*/ /\*pm660 gpio\_2\*/

<&pm660\_gpios 11 0>, /\* HB Vol 2 \*/ /\*pm660 gpio\_11\*/

<&tlmm 13 0>; /\* Speaker Enable \*/ /\*SDM660 gpio\_13 Beep\_EN\*/

pwms = <&pm660l\_pwm\_4 0 370000>; /\* in nanosec \*/

pwm-names = "beep-pwm";

pinctrl-names = "beep\_en\_active","beep\_en\_suspend";

pinctrl-0 = <&beep\_en\_active>;

pinctrl-1 = <&beep\_en\_suspend>;

status = "okay";

};

};

Power key short to longer press change

Acceptance Criteria:

Device should boot only if we keep pressing power key for 3 seconds (long press).

Solution: Longer the power button check period to 3000m

In QcomPkg/Sdm660Pkg/Settings/PMIC/**pm\_config\_target.c file**

pm\_pwrkey\_dbnc\_chk\_type pm\_pon\_pwrkey\_dbnc\_chk[] = {{PM\_PON\_PWRKEY\_DBNC\_CHK\_AT\_CORE, 3000}} ; //check power key press at, time in milli sec

Device should not boot automatically

Description: Device boots up automatically upon removing and inserting battery

Acceptance Criteria:

Device should not boot up automatically when remove and insert battery

Solution: Disable SMPL function to solve it.

In QcomPkg/Library/PmicLib/target/sdm660\_pm660\_pm660l/psi/**pm\_config\_target\_sbl\_sequence.c file**

pm\_sbl\_seq [ ] =

{

{ 0, 0x00, 0x087F, 0xFF, PM\_SBL\_WRITE, 0, 0}, // Line 485 Source: Write(PM660.PON.SMPL\_CTL, 0x00)

};

Set PMIC PM660L GPIO\_6 output source selection in XBL during boot.

Description: 1) A 32 kHz of noise is getting injected in to Beep\_PWM during terminal boot up Process.

2) Beep\_PWM looks noisy until the home screen popping out after terminal boot.

Acceptance Criteria:

Beep\_PWM should be driven low during the entire terminal boot up process.

Solution: Set PMIC PM660L GPIO\_6 output source selection in XBL during boot.

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

//Set PM660L GPIO06 GPIO06\_DIG\_OUT\_SOURCE\_CTL to 0x02 for Beep\_pwm initial low

err\_flag |= pm\_comm\_write\_byte\_mask(2, 0xC544, 0xFF, 0x02, 0);

Animation/Logos for 4.3" WVGA display

Description:Since we only got 1280x720 animation and logos during booting up, need a work to support WVGA(800x480) display.

Acceptance Criteria:

1.Zebra Logo shall be displayed on power-on stage

2.Fastboot screen shall be displayed on boot loader

3.Animation shall be displayed on Android booting-up

Solution: Due to capacitor C3803 makes LCM\_3P0 having slow slew rate,

populate delay 80ms workaround to make power sequence match spec.

Existed issue: dispay will fade away after logo show up

RTC time show abnormal

Description: Device date and time will offset to about ~03:08PM (over 3 minutes faster)

Solution: On PT4 board, using externel RTC to sync time to fix it.

Use the proper dtsi file to resolve the issue

OTG POWER SUPPLY:

OTG power should be provided individually when system connecting to DC/AC in based on PT4/EV HW

Description: OTG power should be provided individually when system connecting to DC/AC in based on PT4/EV HW

Solution: Under OTG, when VBUS regulator supplied, enable OTG\_OUT\_EN to enable OCP. When VBUS regulator gone

, disable OTG\_OUT\_EN to disable OCP.

Need to read the data from dtsi file in drivers/power/supply/qcom/qpnp-smb2.c

Like below

chg->otg\_vbus\_ocp = of\_property\_read\_bool(node, "qcom,otg-vbus-ocp");

and need to set the gpio\_set\_value(chg->otg\_en\_switch,1); in drivers/power/supply/qcom/smb-lib.c file

with out total build debugging

source build/envsetup.sh

lunch aosp\_arm-eng

make -j96

after this eng build we can directly adb push the apk and reboot for verification of apk or any script file or etc.

POWER ISSUES

Story 1:

fix device stability issue

Description: system shutdown due to battery low, correct eMMC power rails

Solution: change L8 to L13, this is HW changes not follow ref design.

vdd-io-supply = <&pm660\_l13>;

search in google for “vdd io supply chain”

[https://gerrit.zebra.com/#/c/82538/](https://gerrit.zebra.com/" \l "/c/82538/)

[https://gerrit.zebra.com/#/c/86245/5](https://gerrit.zebra.com/" \l "/c/86245/5)

story 2:

[https://gerrit.zebra.com/#/c/86263/](https://gerrit.zebra.com/" \l "/c/86263/)

implement interface to set/read charging and fuel gauge parameters

Description: implement interface to set/read charging and fuel gauge parameters

Solution: Add new parameters in power\_supply sysfs

POWER\_SUPPLY\_ATTR(temp\_cold), is the parameter to add for power supply sysfs

Path: /sys/class/power\_supply/battery

File: drivers/power/**power\_supply\_sysfs.c**

drivers/tty/serial/msm\_serial.c file contains the function static void msm\_shutdown(struct uart\_port \*port) to shut down the device

story 3:

Description: Wakeup on external power source

from BSM when AC/USB charger connected.

Solution: Remove usbin and dcin from ignore list.

[drivers/power/supply/qcom/qpnp-smb2.c](https://gerrit.zebra.com/" \l "/c/90315/2/drivers/power/supply/qcom/qpnp-smb2.c) file contains the function called

enable\_irq\_wake(smb2\_irqs[i].irq);

in above function if we add “*usbin-plugin, dcin-plugin”* those string(irq controls interrupt name)

system will wakeup.

 drivers/power/supply/qcom/**qpnp-smb2.c** contains the function called set\_property\_on\_fg which is used to update the fg related information

drivers/power/supply/qcom/**qpnp-smb2.c** contains **smb2\_get\_battery\_error\_status**

story 4:

Add new entry /sys/class/power\_supply/battery/error\_status

Using POWER\_SUPPLY\_ATTR(error\_status), in below file

drivers/power/**power\_supply\_sysfs.c**

story 5:

"Charging wirelessly" when the device is placed in single slot cradle with power supply.

Description:"Charging wirelessly" when the device is placed in single slot cradle with power supply.

Solution: Changing default charing when dc is pluggedin charging to Mains.

In drivers/power/supply/qcom/**qpnp-smb2.c** file add val->intval = POWER\_SUPPLY\_TYPE\_MAINS;

To enable the charging we need to add g\_chip->charging\_disable = false; in drivers/power/supply/qcom/**qpnp-smb2.c**

To read the legacy data(battery related data) from eeprom

We use the drivers/power/supply/qcom/**qpnp-smb2.c**

To enable and disable the wake up source we need to add functions like disable\_irq\_wake , enable\_irq\_wake in drivers/power/supply/qcom/**qpnp-smb2.c**

hwconfig\_kobj = kobject\_create\_and\_add("hardware\_config", NULL);

above function creates a entryyin /sys directory

ret = sysfs\_create\_group(hwid\_kobj, &hwid\_attr\_group); function will addthe attributes in the hardware\_config directory

[https://gerrit.zebra.com/#/c/103874/3](https://gerrit.zebra.com/" \l "/c/103874/3)

story 6:

Description: Current QC PMIC implementation is not working on Thunder design to support host mode

and charging using external power supply. When external regulator is enabled and device removed from cradle.

PMIC is not generating any interrupt as it consider that device is still charging due to VBUS is not powered by External 5V.

This issue is present with QC PMIC if MicroUSB mode is enabled. To overcome this PM660 GPIO 4 will be used for cradle detection and trigger Host mode accordingly.

Solution: 1. Enabled External 5V supply

2. Configured GPIO 04 to detect cradle

3. Moved host mode logic into cradle detection interrupt.

4. Other register changes as required.

[https://gerrit.zebra.com/#/c/104318/16](https://gerrit.zebra.com/" \l "/c/104318/16)

Learn about vbus

Learn about spikes in power supply

Story 7:

Description: Fix Bluetooth can't work.

Solution: The Vdd of touch is from VPH\_PWR not pm660\_l19.

It will influence Bluetooth power.

vdd-supply = <&pm660\_l19>; in dtsi file

if we use more printk functions in driver file device will goes to the download mode

**KERNEL GPIO**

Sleep in kernel:

 include <linux/delay.h> to use msleep in kernel space.

* Backed by busy-wait loop:  
  udelay(unsigned long usecs)
* Backed by hrtimers:  
  usleep\_range(unsigned long min, unsigned long max)
* Backed by jiffies / legacy\_timers  
  msleep(unsigned long msecs)  
  msleep\_interruptible(unsigned long msecs)

[https://gerrit.zebra.com/#/c/85935/3](https://gerrit.zebra.com/" \l "/c/85935/3)

Pinctrl for super cap and buck boost gpio

What is pinctrl ;

The **pinctrl** is nothing but a way to gather pins (not only GPIO), and pass them to the **driver**. The pin controller **driver** is responsible for parsing pin descriptions in the DT and applying their configuration in the chip. The **driver** usually needs a set of two nested nodes to describe group of pins configurations

To config pinctrl we need below functions

devm\_pinctrl\_get

pinctrl\_lookup\_state

pinctrl\_select\_state

after above three calls gpio request, gpio\_direction\_output functions will come

[https://gerrit.zebra.com/#/c/86147/6](https://gerrit.zebra.com/" \l "/c/86147/6)

pwd;

**Pulse width modulation** (**PWM**) is a powerful technique for controlling analog circuits with a microprocessor's digital outputs. **PWM** is employed in a wide variety of applications, ranging from measurement and communications to power control and conversion.

As its name suggests, **pulse width modulation** speed control **works** by driving the motor with a series of “ON-OFF” **pulses** and varying the duty cycle, the fraction of time that the output voltage is “ON” compared to when it is “OFF”, of the **pulses**while keeping the frequency constant.

Duty cycle

The term [*duty cycle*](https://en.wikipedia.org/wiki/Duty_cycle) describes the proportion of 'on' time to the regular interval or 'period' of time

* To map any key to gpio .kl and .idc file required

Conflicts with two services by gpio:

3.3V Buck boost is controlled by PM660 GPIO 11. Qcom is

using this GPIO in qbt1000 driver. Due to this driver buck

boost is not getting initialized.

Solution: Disable qbt1000 driver in dtsi.

Dtsi file for scan key

Scan\_key\_L {

label = "Scan\_key\_L";

gpios = <&tlmm 45 0x1>;

linux,input-type = <1>;

linux,code = <310>;

debounce-interval = <15>;

gpio-key,wakeup;

};

Disable buttons:

We can disable buttons using disable\_irq(bdata->irq); function

Disables button pointed by @bdata. This is done by masking

IRQ line. After this function is called, button won't generate

input events anymore. Note that one can only disable buttons

that don't share IRQs.

To enable enable\_irq(bdata->irq);

sysfs entry for gpio;

/\* Physical GPIO number <N> may be accessed through /[sys](http://157.235.208.175:8080/source/s?path=/sys/&project=SDM660O-PR)/[class](http://157.235.208.175:8080/source/s?path=/sys/class/&project=SDM660O-PR)/[gpio](http://157.235.208.175:8080/source/s?path=/sys/class/gpio/&project=SDM660O-PR)/[gpio](http://157.235.208.175:8080/source/s?path=/sys/class/gpio/gpio&project=SDM660O-PR)<M>/,

\* but <N> and <M> may differ by some offset <O>. To determine that constant,

\* we look for a directory named /[sys](http://157.235.208.175:8080/source/s?path=/sys/&project=SDM660O-PR)/[class](http://157.235.208.175:8080/source/s?path=/sys/class/&project=SDM660O-PR)/[gpio](http://157.235.208.175:8080/source/s?path=/sys/class/gpio/&project=SDM660O-PR)/[gpiochip](http://157.235.208.175:8080/source/s?path=/sys/class/gpio/gpiochip&project=SDM660O-PR)<O>/. If there's not

\* exactly one match for that, we're SOL. \*/

[https://gerrit.zebra.com/#/c/90110/7](https://gerrit.zebra.com/" \l "/c/90110/7) (sysfs entry created to read the gpio key information)

input events

[input\_event](http://157.235.208.175:8080/source/s?defs=input_event&project=SDM660O-PL)([input](http://157.235.208.175:8080/source/s?defs=input&project=SDM660O-PL), [EV\_MSC](http://157.235.208.175:8080/source/s?defs=EV_MSC&project=SDM660O-PL), [MSC\_SCAN](http://157.235.208.175:8080/source/s?defs=MSC_SCAN&project=SDM660O-PL), [scancode](http://157.235.208.175:8080/source/s?defs=scancode&project=SDM660O-PL));

above function plays a major role in getting events from keys

probe function:

it creates a driver instance for the specific drivers and register it into a system of input device.

[https://gerrit.zebra.com/#/c/95295/5](https://gerrit.zebra.com/" \l "/c/95295/5) (holster hall sensor driver)

A **Hall effect sensor** is a device that is used to measure the magnitude of a magnetic field. Its output voltage is directly proportional to the magnetic field strength through it. **Hall effect sensors** are used for proximity**sensing**, positioning, speed detection, and current**sensing** applications.

A **Hall effect sensor** is a transducer that varies its output voltage in response to a magnetic field.**Hall effect sensors** are used for proximity switching, positioning, speed detection, and current **sensing** applications.

Probe: it will perform below 8 main activities

1 hall\_s = devm\_kzalloc(&pdev->dev,sizeof(struct hallsensor\_device), GFP\_KERNEL);

2 holster\_hall\_pinctrl = pinctrl\_get\_select(&pdev->dev,

pinctrl\_state\_name);

3 ret = sysfs\_create\_group(&hall\_s->input\_dev->dev.kobj, &hallsensor\_attr\_group);

4 ret = gpio\_request(hall\_s->gpio, "HALL\_SENSOR\_GPIO");

5 ret = gpio\_direction\_input(hall\_s->gpio);

6 irq = gpio\_to\_irq(hall\_s->gpio);

7 ret = request\_threaded\_irq(hall\_s->irq, NULL, hall\_sensor\_irq,

IRQF\_TRIGGER\_RISING|IRQF\_TRIGGER\_FALLING|IRQF\_ONESHOT,

HALL\_SENSOR\_NAME, hall\_s);

8 pm\_runtime\_enable(&pdev->dev);

debounce\_delay

input\_event:

\* input\_event() - report new input event

\* @dev: device that generated the event

\* @type: type of the event

\* @code: event code

\* @value: value of the event

\* This function should be used by drivers implementing various input

\* devices to report input events. See also input\_inject\_event().

\* NOTE: input\_event() may be safely used right after input device was

\* allocated with input\_allocate\_device(), even before it is registered

\* with input\_register\_device(), but the event will not reach any of the

\* input handlers. Such early invocation of input\_event() may be used

\* to 'seed' initial state of a switch or initial position of absolute

\* axis, etc.

\*/

void input\_event(struct input\_dev \*dev,

unsigned int type, unsigned int code, int value)

**static** [irqreturn\_t](http://157.235.208.175:8080/source/s?defs=irqreturn_t&project=SDM660O-PL) [**qpnp\_kpdpwr\_irq**](http://157.235.208.175:8080/source/s?refs=qpnp_kpdpwr_irq&project=SDM660O-PL)(**int** [**irq**](http://157.235.208.175:8080/source/s?refs=irq&project=SDM660O-PL), **void** \*[**\_pon**](http://157.235.208.175:8080/source/s?refs=_pon&project=SDM660O-PL))

above is the function for power button pressed events

regmap\_read function will read the registers

hbtp\_input:

hbtp\_input is a kernel driver that provides functionality needed by

Host Based Touch Processing (HBTP) from the kernel. One of the

functionality is to manage the power source for touch Analog Front

end.

Spin lock:

spin\_lock\_irqsave(&info->lock, lock\_flag);

critical\_section\_fun(); //critical section will be in between spin lock

spin\_unlock\_irqrestore(&info->lock, lock\_flag);

dtsi:

chip->disable\_esr\_pulse = of\_property\_read\_bool(node,

"zebra,disable\_esr\_pulse");

Above func will read the dtsi file

[https://gerrit.zebra.com/#/c/104318/](https://gerrit.zebra.com/" \l "/c/104318/) (Ethernet cradle)

[https://gerrit.zebra.com/#/c/108052/](https://gerrit.zebra.com/" \l "/c/108052/) (battery swap gpio led)

[https://gerrit.zebra.com/#/c/110488/](https://gerrit.zebra.com/" \l "/c/110488/) (Enable gpio vbus and display port setting)

[https://gerrit.zebra.com/#/c/116038/](https://gerrit.zebra.com/" \l "/c/116038/) (cold and warm boot)

[https://gerrit.zebra.com/#/c/118365/](https://gerrit.zebra.com/" \l "/c/118365/) (diamond key) new key implementation

Continuation :

Probe function is called by **static** **int** [**really\_probe**](http://157.235.208.175:8080/source/s?refs=really_probe&project=SDM660O-PR)(**struct** [device](http://157.235.208.175:8080/source/s?defs=device&project=SDM660O-PR) \*[**dev**](http://157.235.208.175:8080/source/s?refs=dev&project=SDM660O-PR), **struct** [device\_driver](http://157.235.208.175:8080/source/s?defs=device_driver&project=SDM660O-PR) \*[**drv**](http://157.235.208.175:8080/source/s?refs=drv&project=SDM660O-PR))

Located in [kernel](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/)/[msm-4.4](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/)/[drivers](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/drivers/)/[base](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/drivers/base/)/[dd.c](http://157.235.208.175:8080/source/xref/SDM660O-PR/LA.UM.6.2/LINUX/android/kernel/msm-4.4/drivers/base/dd.c)

pdata->have\_shutdown = of\_property\_read\_bool(np, "have-custom-shutdown");

"have-custom-shutdown" is a property of node in dtsi file

of\_property\_read\_bool can read the dtsi file.

[https://gerrit.zebra.com/#/c/132144/7](https://gerrit.zebra.com/" \l "/c/132144/7)

[https://gerrit.zebra.com/#/c/146906/](https://gerrit.zebra.com/" \l "/c/146906/)

[https://gerrit.zebra.com/#/c/147993/](https://gerrit.zebra.com/" \l "/c/147993/)

[https://gerrit.zebra.com/#/c/150436/](https://gerrit.zebra.com/" \l "/c/150436/)

[https://gerrit.zebra.com/#/c/150925/](https://gerrit.zebra.com/" \l "/c/150925/)

[https://gerrit.zebra.com/#/c/152528/](https://gerrit.zebra.com/" \l "/c/152528/)

[https://gerrit.zebra.com/#/c/154251/](https://gerrit.zebra.com/" \l "/c/154251/)

[https://gerrit.zebra.com/#/c/127701/1](https://gerrit.zebra.com/" \l "/c/127701/1)

work queues

Workqueues are used to schedule actions to run in process context. The base unit with which they work is called work. There are two types of work:

* **structure work\_struct** - it schedules a task to run at a later time
* **struct delayed\_work** - it schedules a task to run after at least a given time interval

A delayed work uses a timer to run after the specified time interval. The calls with this type of work are similar to those forstructwork\_struct, but has **\_delayed** in the functions names.

INIT\_WORK(**struct** work\_struct \*work, void(\*function)(**struct** work\_struct \*));

INIT\_DELAYED\_WORK(**struct** delayed\_work \*work, void(\*function)(**struct** work\_struct \*));

Above two are macros for work queue actions

We can use these macros kernel modules

Example:

1)RKES is triggering the wakeup-sources based on resume call of gpio drivers.

Solution: RKES logic is moved to work\_queue rather than report\_events such that fake triggers won’t happen.

2)

INIT\_DELAYED\_WORK(&ec\_dev->event\_work, vc83\_event\_work\_handler);

|

Inside vc83\_event\_work\_handler

to\_delayed\_work(work);

container\_of(work, struct vc83\_ec\_device, event\_work.work);

queue\_delayed\_work();

functions plays major roles

3) Description: Cradle unlock is not responding for first boot /enterprise

reset with device out of cradle and then inserted.

Solution: instead of using hr timer replaced it with workqueue.

4) Description:There should not be much delay in Led blink and options enable in

cradle utility.

Solution: Replaced delayed workqueue with high resolution timer.

Irq:

#define IRQ\_DEBOUNCE\_INTERVAL 20

Above will Expand the time between irq disabling/ enabling using msleep function

We can collect the irq status with variable name of rc

rc = msm\_camera\_enable\_irq(ispif->irq, 1);

mpm driver:

MSM Sleep Power Manager (mpm-v2)

The MPM acts a sleep power manager to shutdown the clock source and put the

device into a retention mode to save power. The MPM is also responsible for

waking up and bringing up the resources from sleep. The MPM driver configures

interrupts monitored by the MPM hardware before entering sleep through a

RPM interface.

Example: Description: Disable MPM interrupts while in battery swap except for the power key

Solution: Modify mpm-of.c to mask all mpm interrupts except power key while in battery swap.

[https://gerrit.zebra.com/#/c/86245/5](https://gerrit.zebra.com/" \l "/c/86245/5)

SPMI protocol:

Kernel resuming on battery removal/insertion

Description: Kernel is resuming on battery removal/insertion

from battery swap mode due to

Limits Management Hardware (LMH) interrupts.

Also kernel resuming intermittently due to mpm

interrupt.

Solution: 1. Disable LMH interrupts while entering battery swap.

2. Configure APSS next wakeup time as infinite.

3. Add debug logs if device wakes up due to PMIC.

[https://gerrit.zebra.com/#/c/137037/5/drivers/spmi/spmi-pmic-arb.c](https://gerrit.zebra.com/" \l "/c/137037/5/drivers/spmi/spmi-pmic-arb.c)

low battery led indication while pressing power key

[https://gerrit.zebra.com/#/c/130698/](https://gerrit.zebra.com/" \l "/c/130698/)

[Description]:When cradle without power and PS20 on in cradle,there is a leakage issue to make cradle MCU activity abnormally

[Solution]: Loop toggle PS20 RX pin low 4s and high 10s when cradle without power or PS20 out of cradle to try to minus leakage time and

maintain reboot tool key feature,single scan key feature and cradle feature work as expected

[https://gerrit.zebra.com/#/c/133695/](https://gerrit.zebra.com/" \l "/c/133695/)