* Using keymapping service we can add all keys to the sysfs location.

If we do any changes in sysfs directories we need to modify in framework also there will be a string named as SYSFS\_PATH, here we need to add the correct modified path.

* Learn about dtsi file nodes
* pinctrl\_lookup\_state() is called in process context to obtain a handle to a specific state for a client device. This operation may be slow, too.
* Scanning Framework check both RTS and CTS values before starting the communication.

The msm-serial driver has been changed to return read uart registers and return correct RTS and CTS values.

So, in msm\_serial.c file update RTS, CTS state.there are some functions called msm\_read, msm\_write there we need to collect the state

(read about CTS, RTS)

* Read about BLSP2 and BLSP4

Sometimes rtc mismatch will happen so,

Some issues like below

there is TZ issue in XBL and no choise in SW side. QC deny releasing new TZ to allow access in XBL.

HW workaround for TZ issue.

* We can configure the gpio keys in dtsi file.

Like specific\_node {

Gpio = <tlmm gpio\_number flags>

};

* VBUS: **USB** requires a shielded cable containing 4 wires. ... The signals on these two wires **are** referenced to the (third) GND wire. The fourth wire **is** called **VBUS**, and carries a nominal 5V supply, which may be used by a device for power.

If we want to enable a feature like vbus

1. We should configure a gpio pin in dtsi file
2. In qpnp-smb2.c file we need to read the dtsi
3. In smb-lib.c file we need to enable and disable using a function gpio\_set\_value().

* In notification bar need to be notify the modifier like blue or alt or shift for that in dtsi file we need to be enable the flags if everything is proper from framework side.

Ex: flags = <”from NONE to ONESHOOT”>

* If we wanna change the version of audio or video we need to modify in makefile also
* Key layout file:

**Key layout files** (.kl **files**) map Linux **key** codes and axis codes to Android **key** codes and axis codes and specify associated policy flags. Device-specific **key layout files** are: Required for internal (built-in) input devices with **keys**, including special **keys** such as volume, power, and headset media **keys**.

**We will keep this .kl file in device/symbol/ directory**

* **Scan code:** means this is the code which is present physically in the keyboard

Ex: scancode = <MPA\_SCANCODE(5, 4)>; (with coordinates)

**Keycode:** it is a code which is mapped to the scancode it can be any code while pressing will reflect on the screen,scan code is a physical key location

* If we add /sys/symbol-keypad in file\_context and file.te,then it create sepolicy label. With current change shell has access to all sysfs files

This is one of the uses of sepolicies

* platform/hardware/libhardware directory contains HAL and HIDL info
* **mutex**

Description: drivers: soc: qcom: Added check to avoid opening multiple instance

Opening of multiple instance of voice\_svc user space from app will

lead to pointer deference of private data within apr callback. As

multi-instance not supported added check to deny open() from user

space if previous instance hasn't been closed.

struct mutex mutex\_lock;

mutex\_init(&mutex\_lock);

mutex\_lock(&mutex\_lock);

inside code of mutex lock

mutex\_unlock(&mutex\_lock);

mutex\_destroy(&session\_lock);

* **Spinlock**

First add header

spinlock\_t lock\_q[NUM\_QUEUES];

spin\_lock\_irqsave(&cci\_dev->cci\_master\_info[master].

lock\_q[queue], flags);

//inside code if spin lock

spin\_unlock\_irqrestore(&cci\_dev->cci\_master\_info[master].

lock\_q[queue], flags);

* we can use spin lock insie of mutex
* irq

in drivers/power/supply/qcom/**qpnp-smb2.c driver file**

static int smb2\_suspend(struct device \*dev) function we can disable irq with function call of disable\_irq(gpio\_to\_irq(g\_chip->dt.gpio\_irq));

similarly

static int smb2\_resume(struct device \*dev) we can enable irq with function call of enable\_irq(gpio\_to\_irq(g\_chip->dt.gpio\_irq));

* In dtsi file we can Config gpio 74 as irq pin and default pull-up.

Like below

alps\_int,irq-gpio = <&tlmm 74 0x0>;

Sometimes we can see the issues like below

Description: RTP will sometims hang up.

After using oscilloscope for i2c and irq pin,

we observed that when irq is rapidly debouncing,

IC might fail to respond to a low irq trigger and thus

irq will not return to high anymore until touch is reset.

This is also not an i2c issue since we can still use i2c

to communicate after touch hangs up.

Increasing the time interval between irq high/low can solve

this issue and this change has been tested on several devices

to prove its workability.

Solution: Expand the time between irq disabling/ enabling

msleep(IRQ\_DEBOUNCE\_INTERVAL); we cann use delay like this

* Display (DSI)

Sometimes we can face a problem like

Display MIPI-DSI bus test Fail

To that we can resolve using display timing issue in fbdev drivers

Like below

MIPI\_OUTP(base + DSIPHY\_LANE\_TEST\_STR, 0x88);

And in bootloader we need to increase the data/clock strength

Like below

sPhySettings.sLaneConfig.uDataStrengthHS = 0x88;

sPhySettings.sLaneConfig.uClockStrengthHS = 0x88;

* **MIPI-DSI issue**

**Root Cause Details:**

For the 2-lane panels, there is an DSI fifo overflow issue in UEFI.  
Besides, the panel tm043ydhg43 needs the configuration to make the DSI clock work correctly.

" <DSILowPowerModeInBLLPEOF units='Bool'>True</DSILowPowerModeInBLLPEOF>\n" " <DSILowPowerModeInBLLP units='Bool'>True</DSILowPowerModeInBLLP>\n"

For the 2-lane panels, there is an DSI fifo overflow issue in UEFI. Besides, the panel tm043ydhg43 needs the configuration to make the DSI clock work correctly. " <DSILowPowerModeInBLLPEOF units='Bool'>True</DSILowPowerModeInBLLPEOF>\n" " <DSILowPowerModeInBLLP units='Bool'>True</DSILowPowerModeInBLLP>\n"

* **Corrective Action:**

1.For the 2-lane panels, needs patch to fix the UEFI overflow issue.

2. Adding the two settings for the panel tm043ydhg43 to make the DSI clock work correctly.

" <DSILowPowerModeInBLLPEOF units='Bool'>True</DSILowPowerModeInBLLPEOF>\n" " <DSILowPowerModeInBLLP units='Bool'>True</DSILowPowerModeInBLLP>\n"

Display would fade away after showing the Zebra LOGO and keep blank until

getting into Android(Need power button to trigger reloading the display

configuration from kernel)

Acceptance Criteria:Display shall not dim

Solution: 1.For the KBA-170731012154 "How to config 2 lane for MIPI DSI on MSM8998,

SDM630 and SDM660" to apply the patch to UEFI.

2.Enable the two features DSILowPowerModeInBLLPEOF and DSILowPowerModeInBLLP

for DSI on UEFI.

So, in bootloader we can add below code

out\_dword(uOffset + HWIO\_MMSS\_DSI\_0\_LOGICAL\_LANE\_SWAP\_CTRL\_ADDR, uLaneSwapCtrlReg);

* We can add register settings for MIPI

Description: SE2100 MIPI test failures. Need to implement necessary timing parameters.

Solution: Added few more register settings for SE2100 to

u8 reg\_settings\_bar\_code\_reading[][2] = {} array

* Setting framerate to 60hz in MIPI

Description : For ET5x8, set framerate to 60Hz

Solution : Change framerate from 50 to 60Hz, and change MIPI driving strength to resolve flicker problem

<DSIRefreshRate units='integer Q16.16'>0x3C0000</DSIRefreshRate>\n"

In xml array we need to add this.

* MIPI-DSI timing

[Description]: Adjust rear camera DATA TX THS-PREPARE timing to meet MIPI spec.

[Solution]: Adjust rear camera DATA TX THS-PREPARE timing to meet MIPI spec, modify from 40.52ns to 49.56ns