Qualcomm Technologies, Inc.

Qualcomm Linux Display Guide - Addendum

80-70018-18A AC

April 11, 2025

Contents

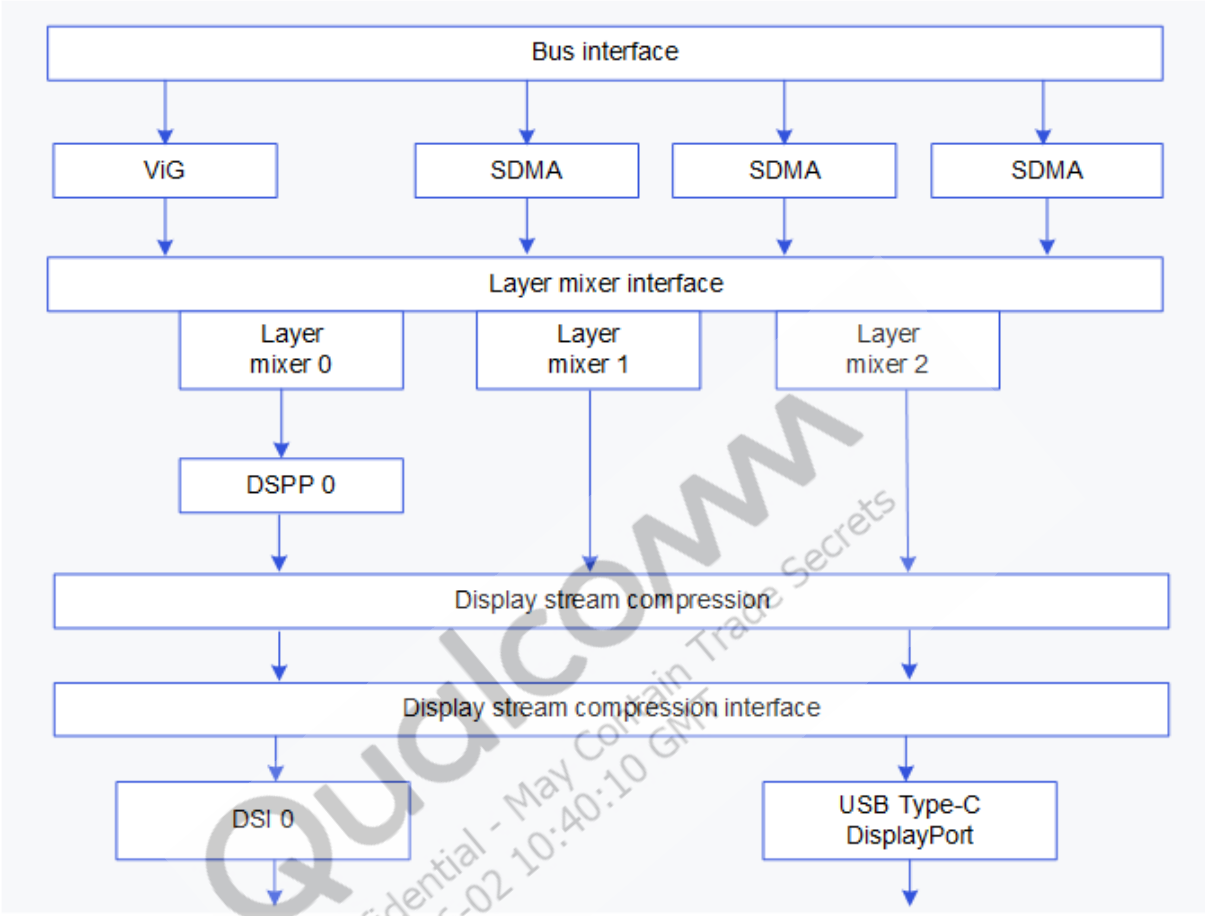
1 Advanced display architecture

This information explains the workflow of the primary components within the hardware and software architecture of the Qualcomm Linux display subsystem.

1.1 Display hardware architecture

See the appropriate chip-product for device-specific information.

QCS6490



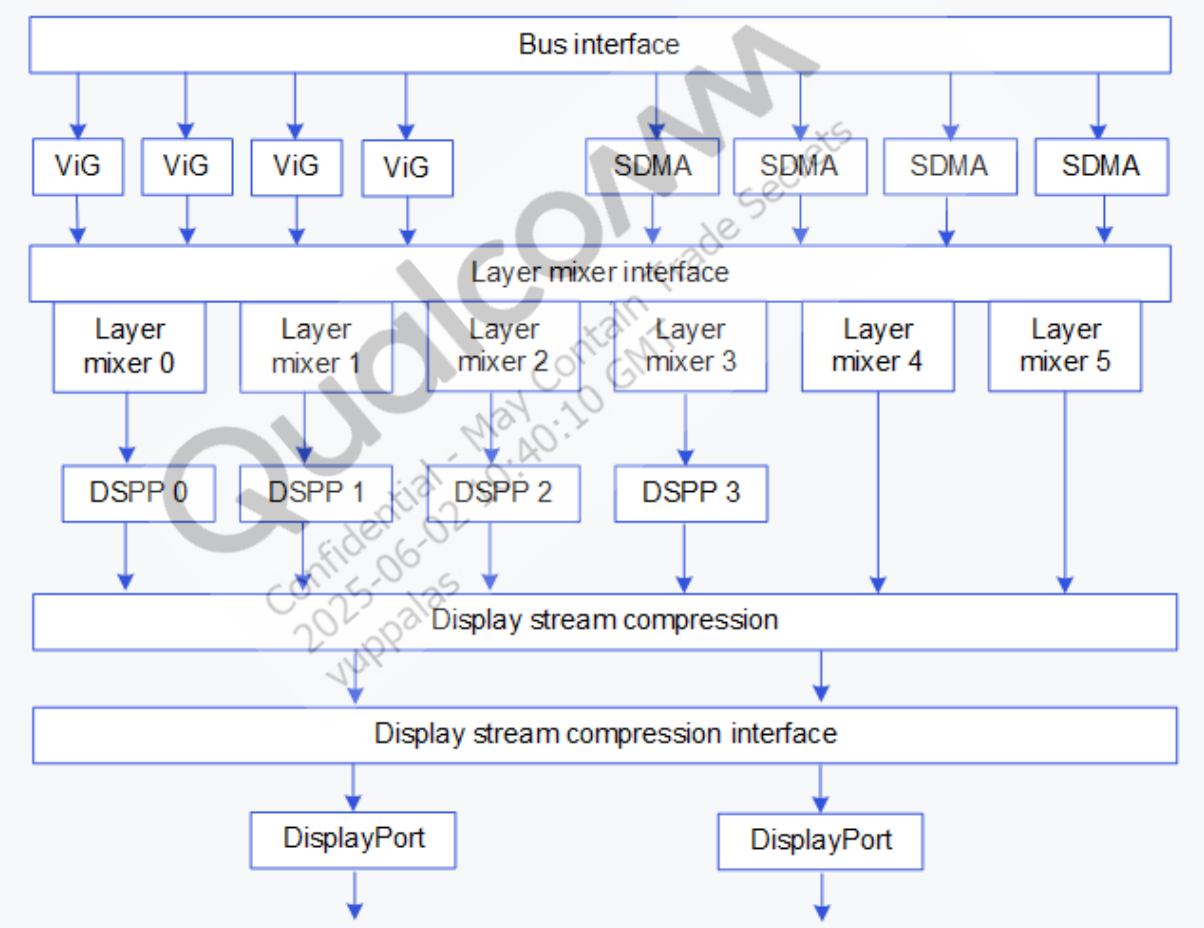
The components of the display hardware architecture are described as follows:

Table : Hardware architecture components

Component	Description
ViG and DMA source surface processor pipes	• Reads RGB and YUV surfaces from gaming and video applications • Performs format conversion and quality improvements for source
Layer mixer (LM) interface	Connects the ViG and DMA pipes to the layer mixer
Layer mixer	Blends and mixes source surface together
Destination surface processor pipes (DSPP)	Converts, corrects, and adjusts the data based on panel characteristics

Component	Description
Display stream compression (DSC)	Reduces bandwidth and power consumption by sending compressed display buffer to the display
Display stream compression interface	Connects the DSC to the display interface
Display interface (DSI/DisplayPort/eDP)	Generates timings for the connected display peripherals

QCS9075

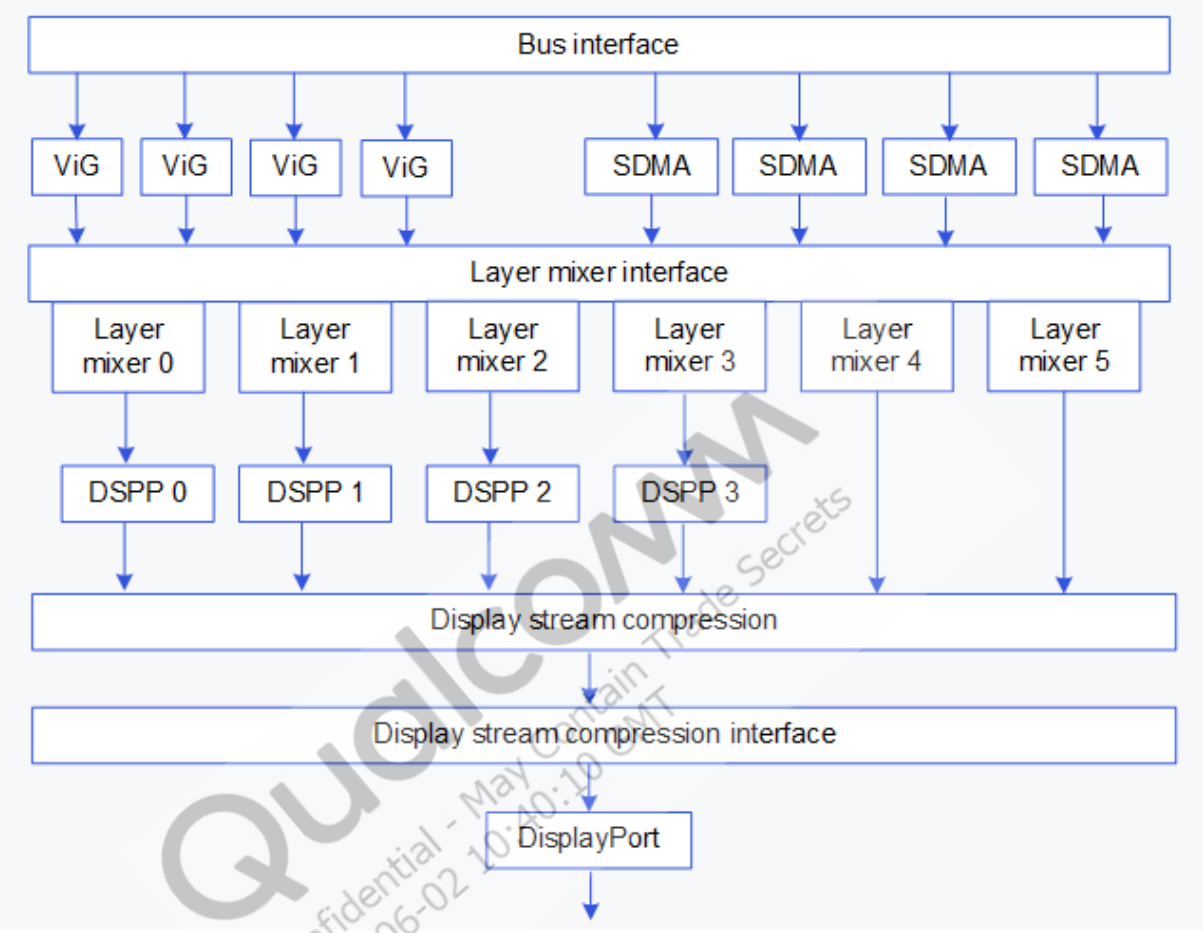


The components of the display hardware architecture are described as follows:

Table : Hardware architecture components

Component	Description
ViG and DMA source surface processor pipes	• Reads RGB and YUV surfaces from gaming and video applications • Performs format conversion and quality improvements for source
Layer mixer (LM) interface	Connects the ViG and DMA pipes to the layer mixer
Layer mixer	Blends and mixes source surface together
Destination surface processor pipes (DSPP)	Converts, corrects, and adjusts the data based on panel characteristics
Display stream compression (DSC)	Reduces bandwidth and power consumption by sending compressed display buffer to the display
Display stream compression interface	Connects the DSC to the display interface
Display interface (DP)	Generates timings for the connected display peripherals

QCS8275



The components of the display hardware architecture are described as follows:

Table : Hardware architecture components

Component	Description
ViG and DMA source surface processor pipes	• Reads RGB and YUV surfaces from gaming and video applications • Performs format conversion and quality improvements for source
Layer mixer (LM) interface	Connects the ViG and DMA pipes to the layer mixer
Layer mixer	Blends and mixes source surface together

Component	Description
Destination surface processor pipes (DSPP)	Converts, corrects, and adjusts the data based on panel characteristics
Display stream compression (DSC)	Reduces bandwidth and power consumption by sending compressed display buffer to the display
Display stream compression interface	Connects the DSC to the display interface
Display interface (DP)	Generates timings for the connected display peripherals

1.2 Display software architecture

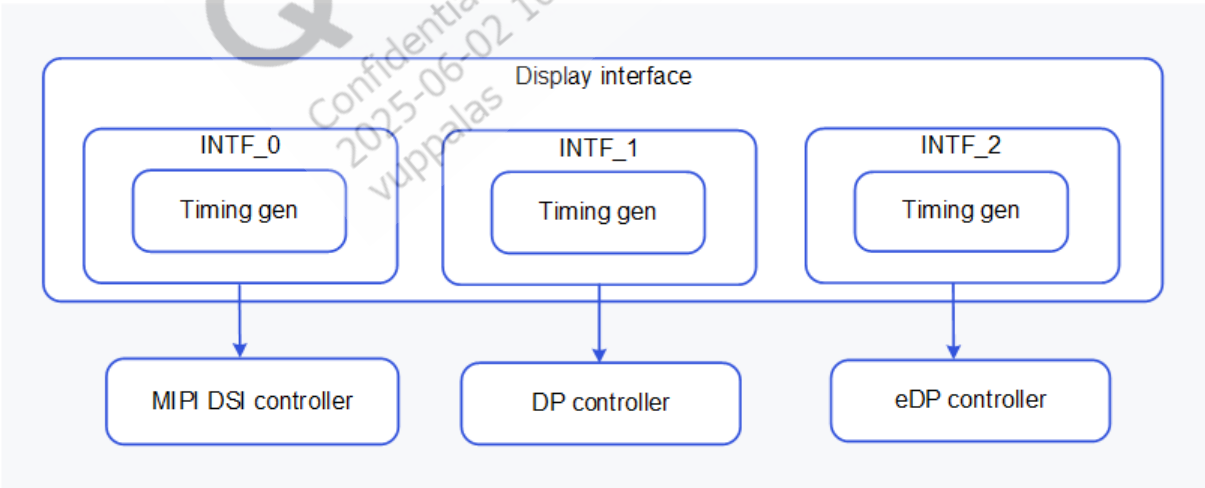
For detailed information, see Display software architecture.

1.3 Display peripheral interfaces

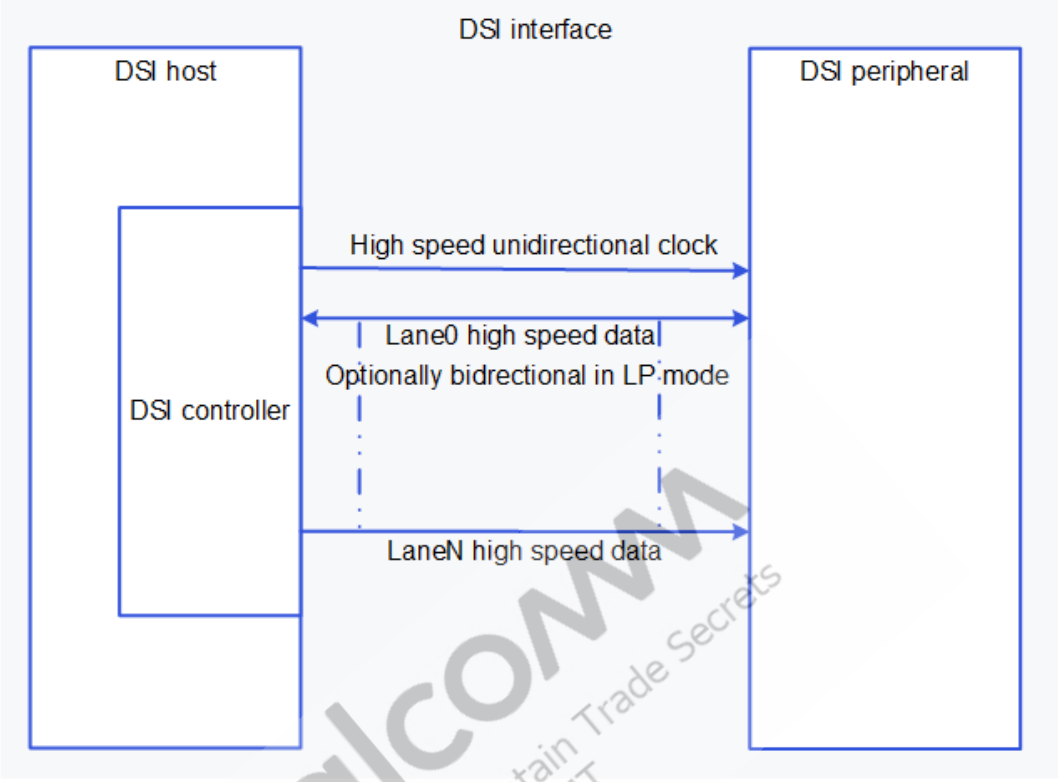
See the appropriate chip-product for device-specific information.

QCS6490

The following figure illustrates the display peripheral interfaces:



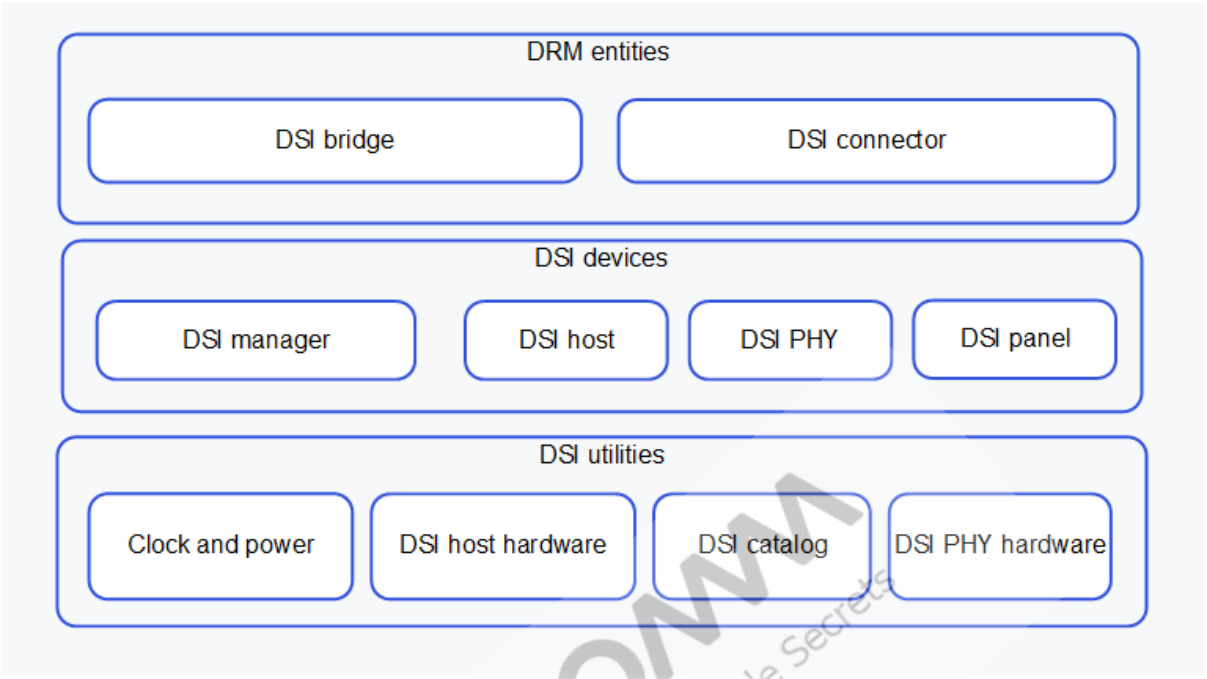
The display serial interface (DSI) defines protocols between a host processor and peripheral devices that adhere to the MIPI Alliance specifications for mobile device interfaces.



DSI is standardized using MIPI. DSI is a five-lane interface (four for data and one for clock) between the host and the peripheral. One of the data lanes operates bidirectionally, while the remaining lanes function unidirectionally. Both the control and the data (pixels) can be transmitted over the lanes.

The DSI interface adheres to the MIPI Alliance standard for DSI 1.2, supporting data rates of up to 2.5 Gbps per lane using D-PHY. In the D-PHY mode, each DSI interface accommodates one differential clock lane and up to four differential data lanes.

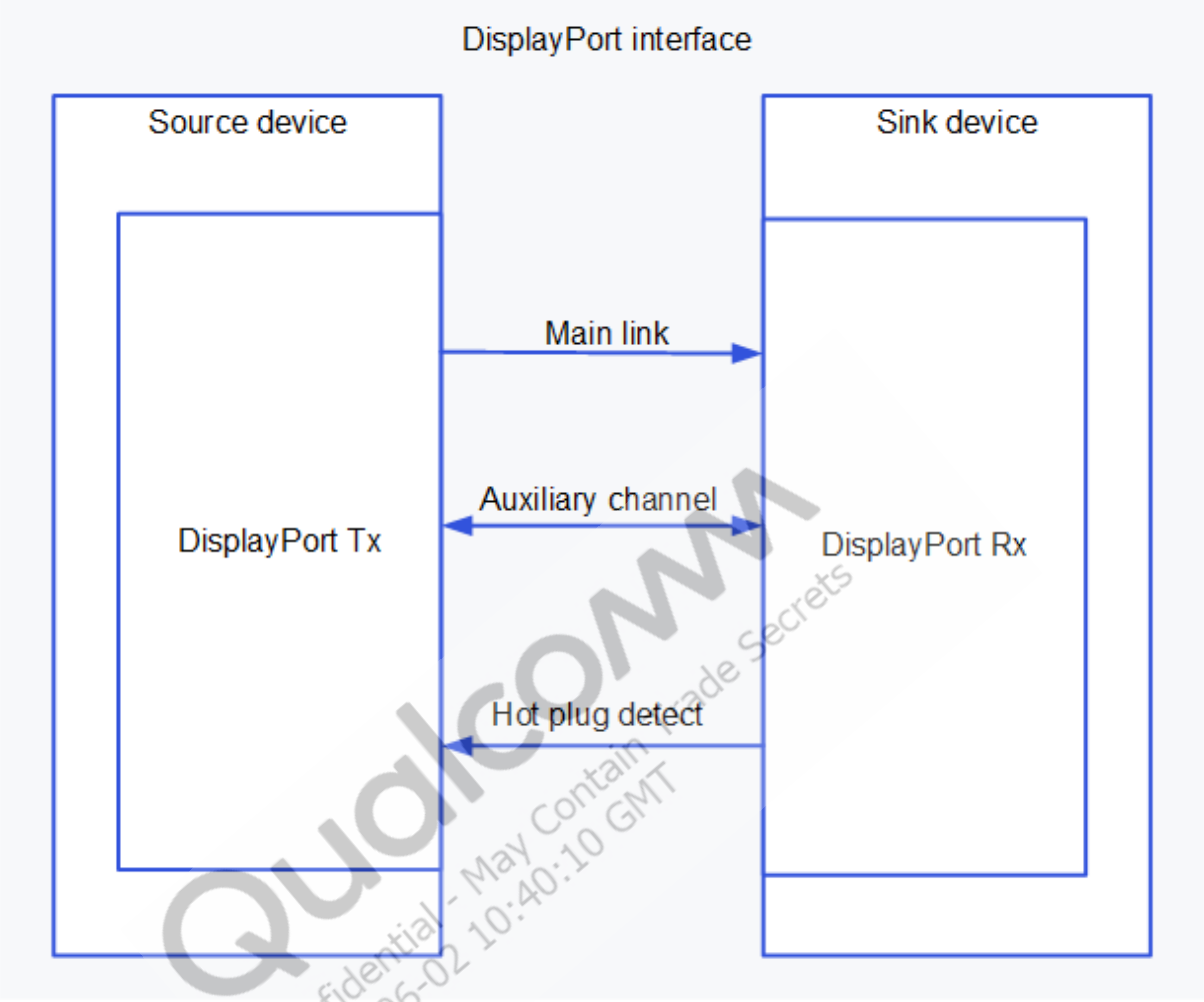
DSI display driver



The components from the DSI DRM driver architecture are explained as follows:

- **DSI manager**: Aggregates individual components, configures pipelines based on the device tree, establishes proper dependencies, and associates each DSI display with a DSI manager. For split DSI panels, one DSI manager handles enumeration while independent dual DSI panels employ two DSI managers, one for each panel.
- **DSI host driver**: Controls the DSI host controller, implementing top-level logic for DSI hardware control, independent of DSI hardware versions.
- **DSI host hardware driver**: Configures the DSI host controller by programming the hardware registers.
- **DSI PHY driver**: Controls the DSI PHY hardware, independent of the DSI PHY type present on the hardware.
- **DSI PHY hardware driver**: Performs the register writes for the DSI PHY. Based on the PHY version, the appropriate driver is selected.
- **DSI catalog**: Contains features supported for each version of the DSI controller.
- **DSI panel driver**: Provides an abstract interface, which the DSI driver uses to control the panel hardware and query panel-specific properties.

DisplayPort covers the DisplayPort modules and how these modules interact with the internal/external panel. The DisplayPort implements the VESA DisplayPort standard 1.4.



The DisplayPort link is the connection between the source and sink device. The DisplayPort link consists of a main link, an auxiliary channel, and a hot plug detect (HPD) signal explained as follows:

- **Main link**
  - It consists of four differential pairs of lanes, which are unidirectional
  - It supports the following link rates:
    - 1.62 reduced bit rate (RBR) Gbps per lane
    - 2.7 high bit rate (HBR) Gbps per lane
    - 5.4 (HBR2) Gbps per lane
    - 8.1 (HBR3) Gbps per lane
  - All the lanes carry data and the stream clock is recovered from the data stream

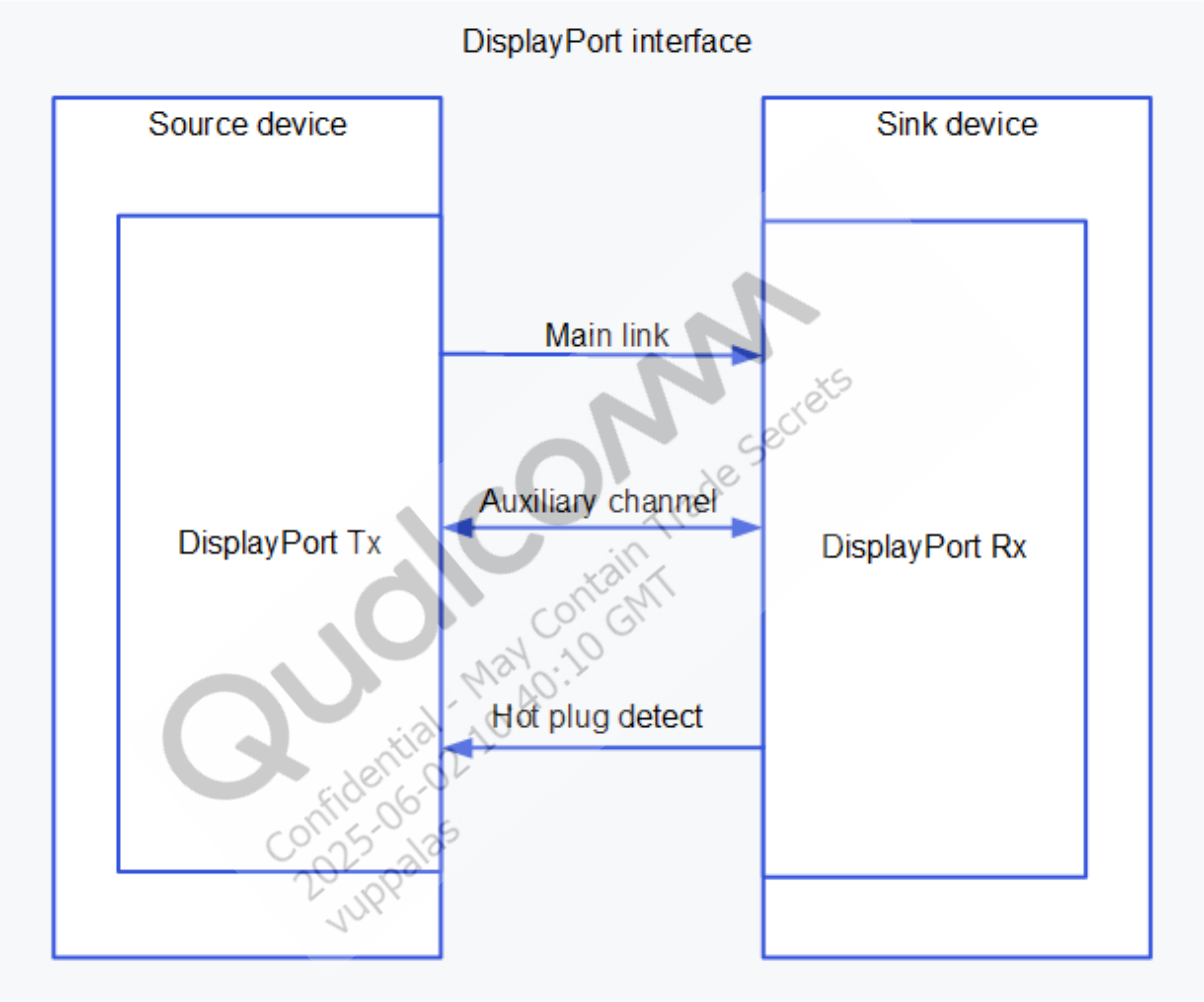
  - Data stream is transmitted as packets
- **Auxiliary channel**
  - It's half-duplex with one differential pair lane that is bidirectional
  - It's used to access the DisplayPort configuration data (DPCD) and extended display identification data (EDID) in the sink device
- **HPD**: It notifies connection and link status changes to the source

The DisplayPort supports the following:

- For DisplayPort 1.4, see the DisplayPort 1.4 specifications
- RGB color depths of 30 bits per pixel (bpp), 24 bpp, and 18 bpp
- Supports 1, 2, or 4 lanes, with data rates of 8.1 Gbps, 5.4 Gbps, 2.7 Gbps, or 1.62 Gbps per lane
- The system supports DisplayPort (2 lanes) alongside USB 3.0 (2 lanes)

QCS9075

DisplayPort covers the DisplayPort modules and how these modules interact with the internal/external panel. The DisplayPort implements the VESA DisplayPort standard 1.4.



The DisplayPort link is the connection between the source and sink device. The DisplayPort link consists of a main link, an auxiliary channel, and a hot plug detect (HPD) signal explained as follows:

- **Main link**
  - It consists of four differential pairs of lanes, which are unidirectional
  - It supports the following link rates:
    - 1.62 reduced bit rate (RBR) Gbps per lane

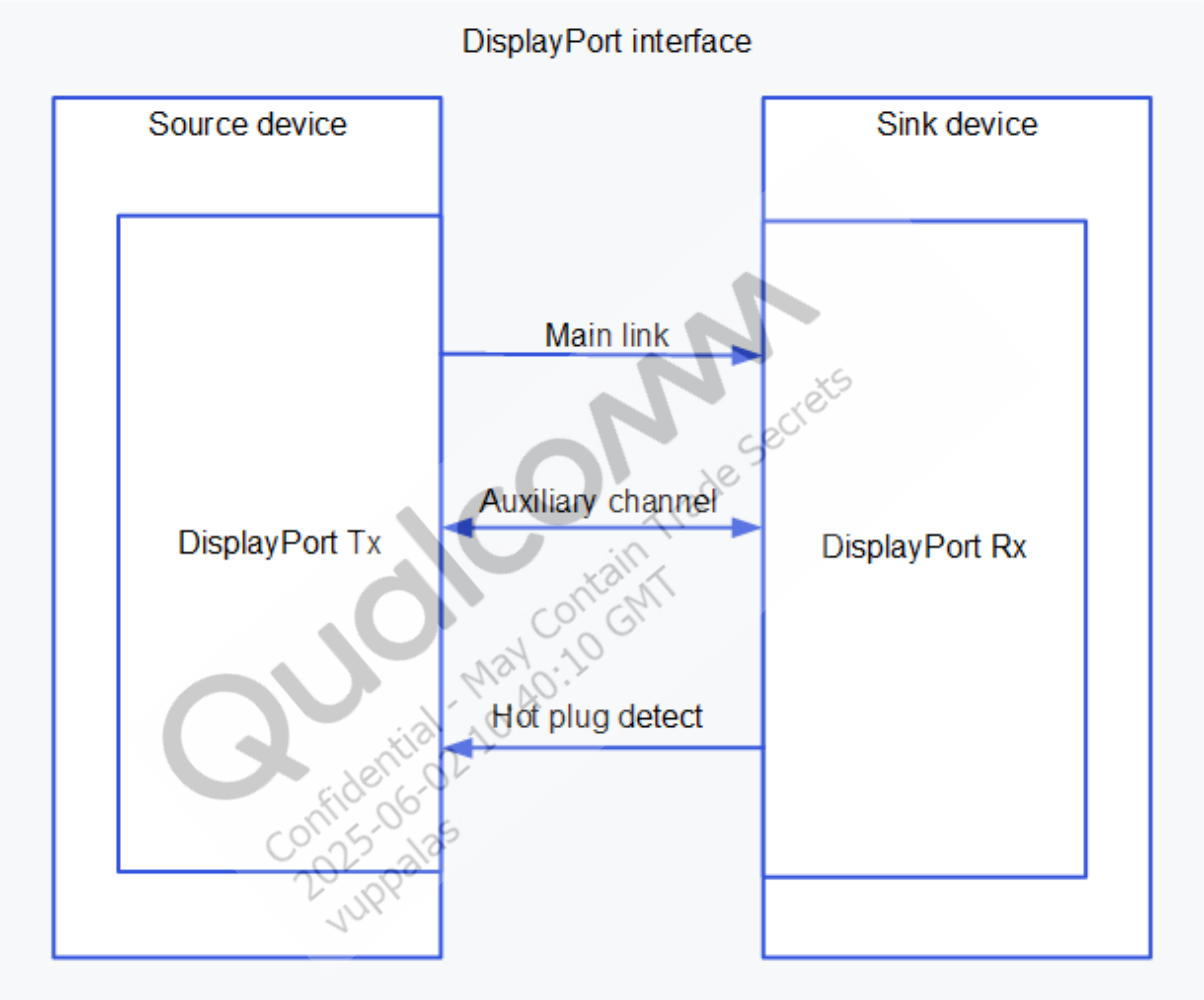
    - 2.7 high bit rate (HBR) Gbps per lane
    - 5.4 (HBR2) Gbps per lane
    - 8.1 (HBR3) Gbps per lane
  - All the lanes carry data and the stream clock is recovered from the data stream
  - Data stream is transmitted as packets
- **Auxiliary channel**
  - It's half-duplex with one differential pair lane that is bidirectional
  - It's used to access the DisplayPort configuration data (DPCD) and extended display identification data (EDID) in the sink device
- **HPD**: It notifies connection and link status changes to the source

The DisplayPort supports the following:

- For DisplayPort 1.4, see the DisplayPort 1.4 specifications
- RGB color depths of 30 bits per pixel (bpp), 24 bpp, and 18 bpp
- Supports 1, 2, or 4 lanes, with data rates of 8.1 Gbps, 5.4 Gbps, 2.7 Gbps, or 1.62 Gbps per lane
- System supports DisplayPort (2 lanes) alongside USB 3.0 (2 lanes)

QCS8275

DisplayPort covers the DisplayPort modules and how these modules interact with the internal/external panel. The DisplayPort implements the VESA DisplayPort standard 1.4.



The DisplayPort link is the connection between the source and sink device. The DisplayPort link consists of a main link, an auxiliary channel, and a hot plug detect (HPD) signal explained as follows:

- **Main link**
  - It consists of four differential pairs of lanes, which are unidirectional
  - It supports the following link rates:
    - 1.62 reduced bit rate (RBR) Gbps per lane

    - 2.7 high bit rate (HBR) Gbps per lane
    - 5.4 (HBR2) Gbps per lane
    - 8.1 (HBR3) Gbps per lane
  - All the lanes carry data and the stream clock is recovered from the data stream
  - Data stream is transmitted as packets
- **Auxiliary channel**
  - It's half-duplex with one differential pair lane that is bidirectional
  - It's used to access the DisplayPort configuration data (DPCD) and extended display identification data (EDID) in the sink device
- **HPD**: It notifies connection and link status changes to the source

The DisplayPort supports the following:

- For DisplayPort 1.4, see the DisplayPort 1.4 specifications
- RGB color depths of 30 bits per pixel (bpp), 24 bpp, and 18 bpp
- Supports 1, 2, or 4 lanes, with data rates of 8.1 Gbps, 5.4 Gbps, 2.7 Gbps, or 1.62 Gbps per lane
- System supports DisplayPort (2 lanes) alongside USB 3.0 (2 lanes)

Note: See Hardware SoCs that are supported on Qualcomm® Linux®.

2 Display software modules

The display software explores the user space and kernel space modules, which are available in the Linux-enabled display software directory.

2.1 Linux-enabled display software directory

The display software directory has user-space modules and Kernel mode modules. To fetch the source code paths, see Use of devtool in the Qualcomm Linux Yocto Guide.

The following tables provide various user space and Kernel mode modules and the steps to retrieve their associated source tree paths using the devtool.

Note: Go to the workspace `<workspace>/build-qcom-wayland$` to access the source code trees using the devtool.

Table : User-space modules

Module	Source path
Weston and `libgbm`	To extract the Weston and the `libgbm` source trees to the `<workspace>/build-qcom-wayland/workspace/sources/msm` directory, run the following command: `devtool modify msm`
`libdrm`	To extract the `libdrm` source tree to the `<workspace>/build-qcom-wayland/workspace/sources/libdrm` directory, run the following command: `devtool modify libdrm`

Table : Kernel mode modules

Module	Source path
Linux Kernel source	To extract the display device tree and drivers source tree to the `<workspace>/build-qcom-wayland/workspace/sources/kernel_platform` directory, run the following command: `devtool modify linux-qcom-base`

2.2 Build instructions

For more information about building the latest software, see Qualcomm Linux Build Guide.

3 Enable advanced features

This information explains the following configurations:

- Set up custom splash logo in UEFI
- Set up the DisplayPort with multistream transport (MST)
- Enable eDP and DisplayPort with SDM backend

3.1 Set up custom splash logo

See the appropriate chip-product for device-specific information.

QCS6490

The following are the supported image formats:

- 8-bit
- 24-bit
- 32-bit
- 8-bit indexed BMP

Note: Use only uncompressed bitmaps.

Create FAT file

1. To create a FAT file, run the following commands on a Linux host computer:

```
dd if=/dev/zero of=logo.bin bs=512 count=4096
```

```
mkfs.vfat logo.bin
```

```
mkdir samp
```

```
sudo mount logo.bin samp
```

```
sudo cp logo_custom.bmp samp
```

```
sudo umount samp
```

Note: Ensure that the logo is available in the directory when you run the `cp` command and rename the logo file as `logo_custom.bmp`.

2. To move the device into Fastboot mode, run the following command:

Note: You must enable SSH to access your host device. For instructions, see Sign in using SSH.

```
reboot bootloader
```

3. To flash the `logo.bin` into the `imagefv` partition, run the following command:

```
fastboot flash imagefv_a logo.bin
```

4. Reboot the device using the following command:

```
fastboot reboot
```

To see the custom splash logo on the HDMI screen, repeat step 2.

QCS9075

Not applicable.

QCS8275

Not applicable.

3.2 Set up DisplayPort with MST

See the appropriate chip-product for device-specific information.

QCS6490

Not applicable.

QCS9075

Connect the DisplayPort cables to the eDP0 and eDP1 connector as shown in the following figure.

Figure : 2xDisplayPort setup

To verify the 2xDisplayPort setup, connect the eDP0 and eDP1 connectors and power on the device. Weston starts automatically when the device boots up. You should see the following Weston flower pattern on the two DisplayPorts.

Figure : Weston flower pattern

QCS8275

Connect the DisplayPort cables to the eDP1 connector as shown in the following figure.

Figure : 1xDisplayPort setup

To verify the 1xDisplayPort setup, connect the DisplayPort cables with the DisplayPort dongle and power on the device. Weston starts automatically when the device boots up.

You should see the following Weston flower pattern on the two DisplayPorts.

Figure : Weston flower pattern

3.3 Enable display downstream drivers

See the appropriate chip-product for device-specific information.

QCS6490

The default build supports the `libdrm` backend and upstream Weston. However, the build doesn't facilitate the native eDP and DisplayPort over Type-C interfaces. To facilitate them, enable SDM backend and DRM/KMS display downstream drivers.

Enable SDM and downstream display drivers

1. Ensure that you have the latest QLI software. If you don't, see Release Note and Qualcomm Linux Build Guide.
2. To enable SDM backend and display drivers, go to the `<workspace>/layers/meta-qcom-hwe` path and apply the patch.
3. To rebuild the image with the changes, see Rebuild.
4. To flash the modified software images, see Flash images.

Verify display

To verify the HDMI display setup, power on the device. Weston starts automatically when the device boots up. You will see the following Weston flower pattern after running the commands:

Figure : Weston flower pattern

Note: If you experience a no-display screen issue, see Relaunch Weston.

Run DRM mode test

If you want to verify the display driver topology, run the DRM modetest. This test ensures that the DSI and the DPU paths are working effectively.

Note: To use the DRM mode test application, you must Generate the DRM mode test binary.

1. To kill the Weston client application, run the following commands in the device shell:

   Note: You must enable SSH to access your host device. For instructions, see Sign in using SSH.

   ```
   mount -o remount,rw /
   ```

```
ps -A | grep weston
```

Sample output:

```
824 ?        00:00:12 weston
1090 ?        00:00:00 weston-keyboard
1092 ?        00:00:01 weston-desktop-
```

Use the Weston process ID from the sample Weston process list to kill the Weston processes.

2. To kill all Weston processes, run the following command:

```
killall weston
```

3. To change permissions of the modetest application, run the following command:

```
chmod 0777 /usr/bin/modetest
```

4. To view the connector IDs and modes, run the following command:

```
/usr/bin/modetest -M msm > /opt/drm_mode.txt
```

5. To verify DRM modes and connector information, pull the `/opt/drm_mode.txt` file locally on your host computer. Use the Secure Copy Protocol (SCP) to pull the `drm_mode.txt` file from the device to your host computer. For example,

```
scp -r root@<IP of the device>:/opt/drm_mode.txt
<directory path of the host machine>
```

Note: When prompted for a password, enter `oelinux123`.

The output as shown in the following figure is from the `/opt/drm_mode.txt` file.

```
sh-5.1# /usr/bin/modetest -M msm_drm
Encoders:
id      crtc    type    possible crtcs  possible clones
31      0       DSI     0x0000000f      0x00000001
55      0       TMDS    0x0000000f      0x00000002

Connectors:
id      encoder status          name            size (mm)       modes   encoders
32      0       connected       DSI-1           620x340         10      31
  modes:
        index name refresh (Hz) hdisp hss hse htot vdisp vss vse vtot
  #0 1920x1080 60.00 1920 2008 2052 2200 1080 1084 1089 1125 148500 flags: nhsync, nvsync; type: preferred, driver
```

Figure : Command prompt output

6. To fetch the connector ID and mode name, use the relevant values from the `/opt/drm_mode.txt` file:

```
modetest -M msm -s <Connector id>:<mode name>
```

7. To launch the modetest application, run the following sample command using the connector ID and mode name retrieved from the `/opt/drm_mode.txt` file. For example:

```
/usr/bin/modetest -M msm -s 32:1920x1080-60
```

Note: On launching the DRM test application, ignore the warning log: `Failed to set gamma: Function not implemented`.

Figure : DRM modetest sample application output screen

8. To kill the modetest application, select **CTRL + C**.

Note: Ensure that you Relaunch Weston.

Verify DisplayPort over Type-C

By default, the Type-C over DisplayPort interface is enabled. To verify the DisplayPort, connect the DisplayPort cable to the Type-C port.

Enable eDP

By default, the DSI interface is enabled. To bring up the eDP panel instead of DSI, do the following:

1. Disable the existing DSI interface and enable the eDP interface.
2. Configure the following display .dtsi files to modify the eDP:
   - `display/qcm6490-sde.dtsi`
   - `display/qcm6490-sde-display-rb3gen2.dtsi`
3. To bring up the eDP interface as the primary interface, integrate the following patch:

```
diff --git a/display/qcm6490-sde-display-rb3gen2.dtsi b/
display/qcm6490-sde-display-rb3gen2.dtsi
index f898866..fd02c74 100644
--- a/display/qcm6490-sde-display-rb3gen2.dtsi
+++ b/display/qcm6490-sde-display-rb3gen2.dtsi
&sde_dsi {
+    status = "disabled";
     /delete-property/ lab-supply;
    /delete-property/ ibb-supply;
    vddio-supply = <&vreg_l8c_1p62>;
    qcom,dsi-default-panel = <&dsi_ext_bridge_4k30p>;
    boot-panel-param = "disabled";

&mdss_dsi_phy0 {
+    status = "disabled";
    /delete-property/ qcom,dsi-pll-ssc-en;
    /delete-property/ qcom,dsi-pll-ssc-mode;
    vdda-0p9-supply = <&vreg_l10c_0p88>;
};

&mdss_dsi0 {
+    status = "disabled";
    reg = <0 0xae94000 0 0x400>,
       <0 0xaf08000 0 0x4>,
       <0 0x0ae36000 0 0x300>,
};
```

```
&mdss_edp0 {
+   status = "ok";
   qcom,display-type = "primary";
   qcom,dp-low-power-hw-hpd;
   vdda-1p2-supply = <&vreg_l6b_1p2>;
};

&mdss_mdp0 {
-   connectors = <&smmu_sde_unsec &sde_dsi &mdss_dp0>;
+   connectors = <&smmu_sde_unsec &mdss_edp0 &mdss_dp0>;
};
```

Verify eDP

Note: You must enable SSH to access your host device. For instructions, see Sign in using SSH.

2. To mount the `debugfs` file system to access the `debugfs` details of the display subsystem, run the following commands:

```
mount -o remount,rw /
```

```
mount -t debugfs none /sys/kernel/debug
```

2. After the integration is complete, verify if the eDP Weston screen appears. Run the following commands to ensure that the eDP clocks are running:

```
cd /sys/kernel/debug/clk/
```

```
cat clk_summary  | grep edp
```

The following output shows all the eDP clocks enabled:

```
edp_phy_pll_vco_div_clk              1        1        0
1350000000          0     0  50000          Y
disp_cc_mdss_edp_pixel_clk_src       1        1        0
 148500000          0     0  50000          Y
disp_cc_mdss_edp_pixel_clk       1        1        0
148500000          0     0  50000          Y
edp_phy_pll_link_clk                 2        2        0
```

```
 270000000           0      0  50000          Y
disp_cc_mdss_edp_link_clk_src        3         3         0
270000000           0      0  50000          Y
disp_cc_mdss_edp_link_div_clk_src        1         1
0    67500000           0      0  50000          Y
disp_cc_mdss_edp_link_intf_clk        1         1         0
  67500000           0      0  50000          Y
disp_cc_mdss_edp_link_clk        1         1         0
270000000           0      0  50000          Y
gcc_edp_clkref_en                    1         1         0
         0           0      0  50000          Y
disp_cc_mdss_edp_aux_clk_src        1         1         0
19200000           0      0  50000          Y
disp_cc_mdss_edp_aux_clk        1         1         0
19200000           0      0  50000          Y
```

QCS9075

Not applicable.

QCS8275

Not applicable.

4 Debug DPU

This information provides the necessary details to troubleshoot and analyze various components, such as the DSI panel clock, DPU, and Weston.

4.1 DPU layer mixers and layer information

To see the DPU layer mixer and layer information, run the following commands:

Note:

- You must enable SSH to access your host device. For instructions, see Sign in using SSH.
- To mount the `debugfs` file system to access the `debugfs` details of the display subsystem, run the following commands:

```
mount -o remount,rw /
```

```
mount -t debugfs none /sys/kernel/debug
```

See the appropriate chip-product for device-specific information.

QCS6490

- `cat /sys/kernel/debug/dri/0/crtc*/state`

```
sh-5.2# cat /sys/kernel/debug/dri/0/crtc*/state
client type: 0
intf_mode: 0
core_clk_rate: 0
bw_ctl: 0
max_per_pipe_ib: 0
client type: 0
intf_mode: 2
core_clk_rate: 136080000
bw_ctl: 622080000
max_per_pipe_ib: 1600000
```

- `cat /sys/kernel/debug/dri/0/crtc*/status`

```
sh-5.2# cat /sys/kernel/debug/dri/0/crtc*/status
crtc:63 width:0 height:0


crtc:64 width:1920 height:1080

        mixer:0 ctl:0 width:1920 height:1080

        plane:39 stage:1
        fb:70 image format:XR24 wxh:1920x1080 cpp[0]:4 cpp[1]:0 cpp[2]:0 cpp[3]:0
        modifier:360287970189639681
        pitches[0]:    7680 pitches[1]:       0 pitches[2]:       0 pitches[3]:       0
        offsets[0]:       0 offsets[1]:       0 offsets[2]:       0 offsets[3]:       0
        src_x:   0 src_y:   0 src_w:125829120 src_h:70778880
        dst_x:   0 dst_y:   0 dst_w:1920 dst_h:1080
        sspp[0]:sspp_0
        multirect[0]: mode: 0 index: 0

vblank fps:0 count:104 total:533987ms total_framecount:62
```

QCS9075

- `cat /sys/kernel/debug/dri/0/crtc*/state`

```
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# cat /sys/kernel/debug/dri/0/crtc*/state
client type: 0
intf_mode: 0
core_clk_rate: 0
bw_ctl: 0
max_per_pipe_ib: 0
client type: 0
intf_mode: 2
core_clk_rate: 0
bw_ctl: 0
max_per_pipe_ib: 0
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# ^C
```

- `cat /sys/kernel/debug/dri/0/crtc*/status`

```
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# cat /sys/kernel/debug/dri/0/crtc*/status
crtc:87 width:0 height:0


crtc:88 width:3840 height:2160

        mixer:0 ctl:2 width:1920 height:2160
        mixer:1 ctl:2 width:1920 height:2160

        plane:45 stage:1
        fb:90 image format:XB24 wxh:3840x2160 cpp[0]:4 cpp[1]:0 cpp[2]:0 cpp[3]:0
        modifier:       0
        pitches[0]:   15360 pitches[1]:       0 pitches[2]:       0 pitches[3]:       0
        offsets[0]:       0 offsets[1]:       0 offsets[2]:       0 offsets[3]:       0
        src_x:    0 src_y:    0 src_w:251658240 src_h:141557760
        dst x:    0 dst_y:    0 dst_w:3840 dst_h:2160
        sspp[0]:sspp_1
        multirect[0]: mode: 0 index: 0

vblank fps:0 count:81 total:332387ms total_framecount:61
```

QCS8275

- `cat /sys/kernel/debug/dri/0/crtc*/state`

```
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# cat /sys/kernel/debug/dri/0/crtc*/state
client type: 0
intf_mode: 0
core_clk_rate: 0
bw_ctl: 0
max_per_pipe_ib: 0
client type: 0
intf_mode: 2
core_clk_rate: 0
bw_ctl: 0
max_per_pipe_ib: 0
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# ^C
```

- `cat /sys/kernel/debug/dri/0/crtc*/status`

```
root@qcs9100-ride-sx:/sys/kernel/debug/dri/0/debug/core_perf# cat /sys/kernel/debug/dri/0/crtc*/status
crtc:87 width:0 height:0


crtc:88 width:3840 height:2160

        mixer:0 ctl:2 width:1920 height:2160
        mixer:1 ctl:2 width:1920 height:2160

        plane:45 stage:1
        fb:90 image format:XB24 wxh:3840x2160 cpp[0]:4 cpp[1]:0 cpp[2]:0 cpp[3]:0
        modifier:       0
        pitches[0]:   15360 pitches[1]:       0 pitches[2]:       0 pitches[3]:       0
        offsets[0]:       0 offsets[1]:       0 offsets[2]:       0 offsets[3]:       0
        src_x:    0 src_y:    0 src_w:251658240 src_h:141557760
        dst x:    0 dst_y:    0 dst_w:3840 dst_h:2160
        sspp[0]:sspp_1
        multirect[0]: mode: 0 index: 0

vblank fps:0 count:81 total:332387ms total_framecount:61
```

4.2 Check DPU clock information

The common DPU debugging information is as follows:

DPU clock rate

Note: You must enable SSH to securely access your host device. For instructions, see Sign in using SSH.

To check the DPU clock rate, run the following commands:

```
cat /sys/kernel/debug/clk/clk_summary | grep disp_cc
```

4.3 Retrieve display interface information

To retrieve the debug dump output (display interface number, VSync count, underrun count, and interface modes), run the following command:

```
cat /sys/kernel/debug/dri/0/encoder*/status
```

Sample output

```
intf:1  wb:-1  vsync:      93     underrun:        0     mode:
INTF_MODE_VIDEO
intf:-1  wb:2  vsync:      0     underrun:        0     mode:
INTF_MODE_WB_LINE
```


4.4 Check regulator information

To check the regulator status and the voltage, run the following command:

```
cat /sys/kernel/debug/regulator/regulator_summary
```

5 References

5.1 Related documents

Title	
Open-source	
DRM internals	https://www.kernel.org/doc/html/v5.4/gpu/drm-internals.html
Kernel mode setting (KMS)	https://www.kernel.org/doc/html/v5.4/gpu/drm-kms.html
Standards	
HDMI Specification	www.hdmi.org/
MIPI Alliance Specification for D-PHY	mipi.org
MIPI Alliance Specification for DSI	mipi.org
MIPI Alliance Specification for DCS	mipi.org
VESA Display Stream Compression	www.vesa.org
VESA DisplayPort	www.vesa.org

5.2 Acronyms and terms

Acronym or Term	Definition
BTA	Bus turnaround
DM	Display manager
DMA	Direct memory access
DPU	Display processing unit
DSC	Display stream compression
DSI	Display serial interface
DPCD	Display port configuration data
DRM	Direct rendering manager
DSPP	Destination surface processor
DDIC	Display driver IC
EDID	Extended display identification data

Acronym or Term	Definition
EMI	Electromagnetic interference
EOF	End of frame
ESD	Electrostatic discharge
FHD	Full HD
HDCP	High-bandwidth digital content protection
HPD	Hot plug detect
KMS	Kernel mode setting
LM	Layer mixer
SSPP	Source surface processor pipes
SSC	Spread spectrum clocking
SST	Single stream transport
MIPI DSI	Mobile industry processor interface display serial interface
MST	Multi-stream transport
IOCTL	I/O control
CRTC	Cathode ray tube controller
PD	Power delivery
TE	Tearing effect
UEFI	Unified extensible firmware interface
WQHD	Wide Quad HD

LEGAL INFORMATION

Your access to and use of this material, along with any documents, software, specifications, reference board files, drawings, diagnostics and other information contained herein (collectively this "Material"), is subject to your (including the corporation or other legal entity you represent, collectively "You" or "Your") acceptance of the terms and conditions ("Terms of Use") set forth below. If You do not agree to these Terms of Use, you may not use this Material and shall immediately destroy any copy thereof.

1) Legal Notice.

This Material is being made available to You solely for Your internal use with those products and service offerings of Qualcomm Technologies, Inc. ("**Qualcomm Technologies**"), its affiliates and/or licensors described in this Material, and shall not be used for any other purposes. If this Material is marked as "**Qualcomm Internal Use Only**", no license is granted to You herein, and You must immediately (a) destroy or return this Material to Qualcomm Technologies, and (b) report Your receipt of this Material to qualcomm.support@qti.qualcomm.com. This Material may not be altered, edited, or modified in any way without Qualcomm Technologies' prior written approval, nor may it be used for any machine learning or artificial intelligence development purpose which results, whether directly or indirectly, in the creation or development of an automated device, program, tool, algorithm, process, methodology, product and/or other output. Unauthorized use or disclosure of this Material or the information contained herein is strictly prohibited, and You agree to indemnify Qualcomm Technologies, its affiliates and licensors for any damages or losses suffered by Qualcomm Technologies, its affiliates and/or licensors for any such unauthorized uses or disclosures of this Material, in whole or part.

Qualcomm Technologies, its affiliates and/or licensors retain all rights and ownership in and to this Material. No license to any trademark, patent, copyright, mask work protection right or any other intellectual property right is either granted or implied by this Material or any information disclosed herein, including, but not limited to, any license to make, use, import or sell any product, service or technology offering embodying any of the information in this Material.

THIS MATERIAL IS BEING PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESSED, IMPLIED, STATUTORY OR OTHERWISE. TO THE MAXIMUM EXTENT PERMITTED BY LAW, QUALCOMM TECHNOLOGIES, ITS AFFILIATES AND/OR LICENSORS SPECIFICALLY DISCLAIM ALL WARRANTIES OF TITLE, MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, COMPLETENESS OR ACCURACY, AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MOREOVER, NEITHER QUALCOMM TECHNOLOGIES, NOR ANY OF ITS AFFILIATES AND/OR LICENSORS, SHALL BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY EXPENSES, LOSSES, USE, OR ACTIONS HOWSOEVER INCURRED OR UNDERTAKEN BY YOU IN RELIANCE ON THIS MATERIAL.

Certain product kits, tools and other items referenced in this Material may require You to accept additional terms and conditions before accessing or using those items.

Technical data specified in this Material may be subject to U.S. and other applicable export control laws. Transmission contrary to U.S. and any other applicable law is strictly prohibited.

Nothing in this Material is an offer to sell any of the components or devices referenced herein.

This Material is subject to change without further notification.

In the event of a conflict between these Terms of Use and the *Website Terms of Use* on www.qualcomm.com, the *Qualcomm Privacy Policy* referenced on www.qualcomm.com, or other legal statements or notices found on prior pages of the Material, these Terms of Use will control. In the event of a conflict between these Terms of Use and any other agreement (written or click-through, including, without limitation any non-disclosure agreement) executed by You and Qualcomm Technologies or a Qualcomm Technologies affiliate and/or licensor with respect to Your access to and use of this Material, the other agreement will control.

These Terms of Use shall be governed by and construed and enforced in accordance with the laws of the State of California, excluding the U.N. Convention on International Sale of Goods, without regard to conflict of laws principles. Any dispute, claim or controversy arising out of or relating to these Terms of Use, or the breach or validity hereof, shall be adjudicated only by a court of competent jurisdiction in the county of San Diego, State of California, and You hereby consent to the personal jurisdiction of such courts for that purpose.

2) Trademark and Product Attribution Statements.

Qualcomm is a trademark or registered trademark of Qualcomm Incorporated. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the U.S. and/or elsewhere. The Bluetooth® word mark is a registered trademark owned by Bluetooth SIG, Inc. Other product and brand names referenced in this Material may be trademarks or registered trademarks of their respective owners.

Snapdragon and Qualcomm branded products referenced in this Material are products of Qualcomm Technologies, Inc. and/or its subsidiaries. Qualcomm patented technologies are licensed by Qualcomm Incorporated.