StarFive 赛昉科技

JH7110 HDMI Developing Guide

VisionFive 2

Version: 1.0

Date: 2022/11/10

Doc ID: JH7110-DGEN-011

Legal Statements

Important legal notice before reading this documentation.

PROPRIETARY NOTICE

Information in this document is provided "as is," with all faults. Contents may be periodically updated or revised due to product development. Shanghai StarFive Technology Co., Ltd. (hereinafter "StarFive") reserves the right to make changes without further notice to any products herein.

StarFive expressly disclaims all warranties, representations, and conditions of any kind, whether express or implied, including, but not limited to, the implied warranties or conditions of merchantability, fitness for a particular purpose, and non-infringement.

StarFive does not assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages.

Contact Us

Address: Room 502, Building 2, No. 61 Shengxia Rd., China (Shanghai) Pilot Free Trade Zone, Shanghai, 201203, China

Website: http://www.starfivetech.com

Email:

- Sales: sales@starfivetech.com
- Support: support@starfivetech.com

Contents

List of Tables

List of Figures

Preface

About this guide and technical support information.

About this document

This document mainly provides the SDK developers with the programing basics and debugging know-how for the HDMI of the StarFive next generation SoC platform - JH7110.

Audience

This document mainly serves the HDMI relevant driver developers. If you are developing other modules, place a request to your sales or support consultant for our complete documentation set on JH7110.

Revision History

Table 0-1 Revision History

Version	Released	Revision
1.0		First official release.

Notes and notices

The following notes and notices might appear in this guide:

- **Tip:**
  Suggests how to apply the information in a topic or step.
- **Note:**
  Explains a special case or expands on an important point.
- **Important:**
  Points out critical information concerning a topic or step.
- **CAUTION:**
  Indicates that an action or step can cause loss of data, security problems, or performance issues.
- **Warning:**
  Indicates that an action or step can result in physical harm or cause damage to hardware.

1. Introduction

The display subsystem, named as dom_vout_top in the JH7110 system, includes front-end video data capture, display controller and display interface, such as RGB IF, HDMI and MIPI.

In the display subsystem, *High Definition Multimedia Interface (HDMI)* is used to transmit high quality video and audio data, with the transmission speed up to 5 Gbps. Before HDMI data transmission, no analog/digital data transformation is required, its simplicity ensures the highest quality signal transmission especially for high definition movies and musics.

See Block Diagram *(on page 7)* for more information.

1.1. Function Introduction

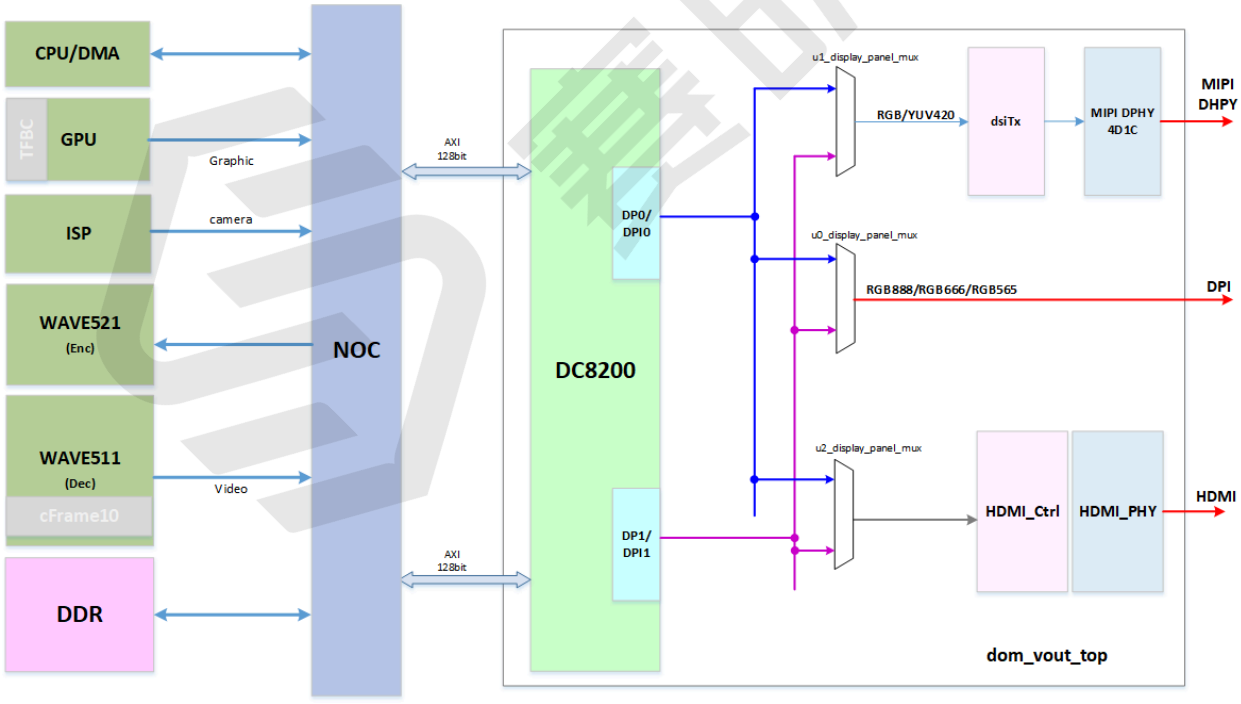
The JH7110 SoC Platform supports the following features and specifications on the HDMI2.0 interface.

- Support 4K, 1080p, 720p, 480p and other common resolutions.
- Support YUV444, YUV420, RGB444 and other common formats.
- Support 8-bit color width.
- Support BIST mode.
- Standard I2C communication.

1.2. Block Diagram

The block diagram of the display subsystem is displayed in following diagram.

Figure 1-1 Display Subsystem Block Diagram



Data Mapping

The DSI transmitter's pixel data could be from panel 0 or panel 1 interface of DC8200, and could be selected from DP or DPI interface. The RGB PAD and HDMI have similar mechanism.

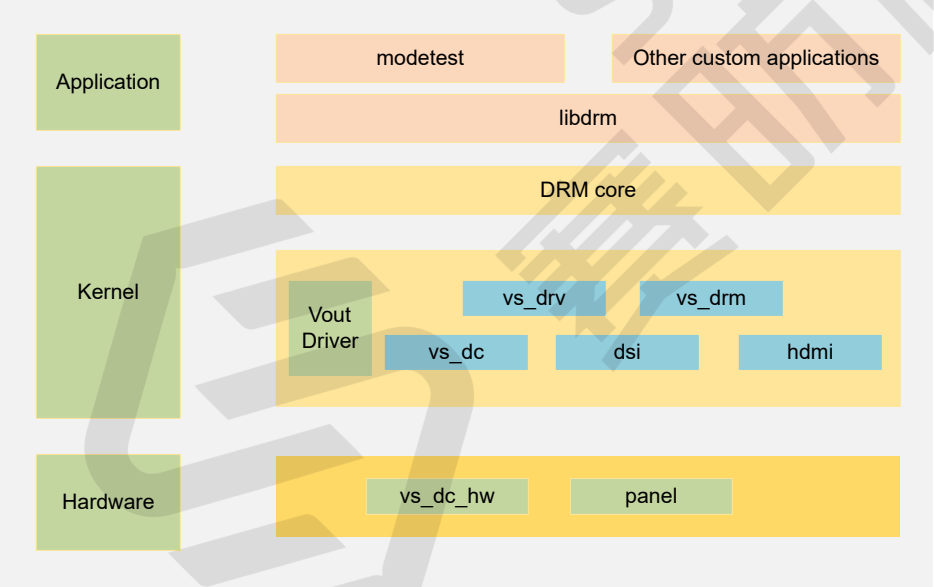
Table 1-1 Display Subsystem Data Mapping

Destination	Supported Data Mapping	Comment
DPI to PAD	• DP0/DP1 or DPI0/1 is used, default DPI is used. • RGB24, RGB666 (CFG1), RGB565 (CFG1) when DPI is used.	For flexibility
DSI Tx Data from DC8200	• Both DPI and DP are supported. • YUV420 8-bit only (CFG3). • YUV422 8-bit only (CFG1).	Default DPI
HDMI Data from DC8200	• Both DP0 and DP1 are used for RGB and YUV. • YUV444 and YUV422 8-bit/10-bit (CFG1). • YUV420 8-bit/10-bit (CFG3).	DP by default, and DPI for back-up

1.3. Video Output Driver Framework

The following figure shows the framework of the video output driver and the display controller.

Figure 1-2 Driver Framework



The video output driver framework has the following 3 layers.

- **Application** layer consists of application code and test code and communicate with kernel layer through **libdrm**.
- **Kernel** layer consists of **DRM core** and **Vout driver**. **DRM core** receives commands from **libdrm** and transfer to **Vout driver**.
- **Hardware** layer is connected with **Vout driver**, and it operates the hardware directly.

1.4. Source Code Structure

The following code block shows the source code structure of the HDMI driver.

```
linux-5.15.0
└-- drivers
└-- | --gpu
| -- | -- | -- drm
| -- | -- | -- |-- verisilicon
| -- | -- | -- | -- | -- inno_hdmi.c
| -- | -- | -- | -- | -- inno_hdmi.h
```

1.5. Device Tree Overview

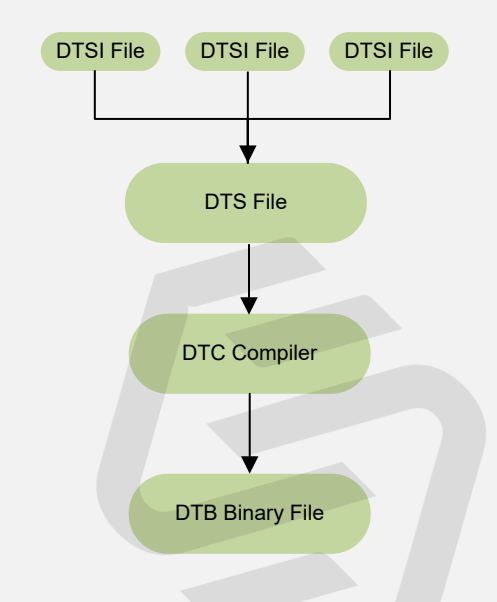
Since Linux 3.x, device tree is introduced as a data structure and language to describe hardware configuration. It is a system-readable description of hardware settings so that the operating system doesn't have to hard code details of the machine.

A device tree is primarily represented in the following forms.

- *Device Tree Compiler (DTC)*: The tool used to compile device tree into system-readable binaries.
- *Device Tree Source (DTS)*: The human-readable device tree description file. You can locate the target parameters and modify hardware configuration in this file.
- *Device Tree Source Information (DTSI)*: The human-readable header file which you can include in device tree description. You can locate the target parameters and modify hardware configuration in this file.
- *Device Tree Blob (DTB)*: The system-readable device tree binary blob files which is burned in system for execution.

The following diagram shows the relationship (workflow) of the above forms.

Figure 1-3 Device Tree Workflow



1.6. Device Tree Source Code

Overview Structure

The device tree source code of JH7110 is listed as follows:

```
linux
├── arch
|   ├── riscv
|   |   ├── boot
|   |   |   ├── dts
|   |   |   |   └── starfive
|   |   |   |       ├── codecs
```

```
|   |   |   |       |   ├── sf_pdm.dtsi
|   |   |   |       |   ├── sf_pwmdac.dtsi
|   |   |   |       |   ├── sf_spdif.dtsi
|   |   |   |       |   ├── sf_tdm.dtsi
|   |   |   |       |   └── sf_wm8960.dtsi
|   |   |   |       ├── evb-overlay
|   |   |   |       |   ├── jh7110-evb-overlay-can.dts
|   |   |   |       |   ├── jh7110-evb-overlay-rgb2hdmi.dts
|   |   |   |       |   ├── jh7110-evb-overlay-sdio.dts
|   |   |   |       |   ├── jh7110-evb-overlay-spi.dts
|   |   |   |       |   ├── jh7110-evb-overlay-uart4-emmc.dts
|   |   |   |       |   ├── jh7110-evb-overlay-uart5-pwm.dts
|   |   |   |       |   └── Makefile
|   |   |   |       ├── jh7110-clk.dtsi
|   |   |   |       ├── jh7110-common.dtsi
|   |   |   |       ├── jh7110.dtsi
|   |   |   |       ├── jh7110-evb-can-pdm-pwmdac.dts
|   |   |   |       ├── jh7110-evb.dts
|   |   |   |       ├── jh7110-evb.dtsi
|   |   |   |       ├── jh7110-evb-dvp-rgb2hdmi.dts
|   |   |   |       ├── jh7110-evb-pcie-i2s-sd.dts
|   |   |   |       ├── jh7110-evb-pinctrl.dtsi
|   |   |   |       ├── jh7110-evb-spi-uart2.dts
|   |   |   |       ├── jh7110-evb-uart1-rgb2hdmi.dts
|   |   |   |       ├── jh7110-evb-uart4-emmc-spdif.dts
|   |   |   |       ├── jh7110-evb-uart5-pwm-i2c-tdm.dts
|   |   |   |       ├── jh7110-fpga.dts
|   |   |   |       ├── jh7110-visionfive-v2.dts
|   |   |   |       ├── Makefile
|   |   |   |       └── vf2-overlay
|   |   |   |           ├── Makefile
|   |   |   |           └── vf2-overlay-uart3-i2c.dts
```

SoC Platform

The device tree source code of the JH7110 SoC platform is in the following path:

```
freelight-u-sdk/linux/arch/riscv/boot/dts/starfive/jh7110.dtsi
```

VisionFive 2

The device tree source code of the VisionFive 2 *Single Board Computer (SBC)* is in the following path:

```
freelight-u-sdk/linux/arch/riscv/boot/dts/starfive/jh7110-visionfive-v2.dts
-- freelight-u-sdk/linux/arch/riscv/boot/dts/starfive/jh7110-common.dtsi
-- freelight-u-sdk/linux/arch/riscv/boot/dts/starfive/jh7110.dtsi
```

2. Configuration

2.1. Device Tree Configuration

A DTS/DTSI file is used to store all the device tree configuration.

The device tree of HDMI is stored in the following path:

```
linux-5.10/arch/riscv/boot/dts/starfive/
```

The following code block shows the DTS file structure for HDMI.

```
linux-5.15.0
└-- arch
└-- | -- riscv
| -- | -- | -- boot
| -- | -- | -- | -- dts
| -- | -- | -- | -- | -- starfive
| -- | -- | -- | -- | -- | -- jh7110-common.dtsi
| -- | -- | -- | -- | -- | -- jh7110.dtsi
```

The following is an example of the HDMI configuration in the file `jh7110.dts`.

```
hdmi: hdmi@29590000 {
  compatible = "starfive,jh7100-hdmi","inno,hdmi";
   reg = <0x0 0x29590000 0x0 0x4000>;
   interrupts = <99>;
   status = "disabled";
   clocks = <&clkvout JH7110_U0_HDMI_TX_CLK_SYS>,
    <&clkvout JH7110_U0_HDMI_TX_CLK_MCLK>,
    <&clkvout JH7110_U0_HDMI_TX_CLK_BCLK>,
    <&hdmitx0_pixelclk>;
   clock-names = "sysclk", "mclk","bclk","pclk";
   resets = <&rstgen RSTN_U0_HDMI_TX_HDMI>;
   reset-names = "hdmi_tx";
  };
```

The following list provides explanations for the parameters included in the above code block.

- **compatible**: Compatibility information, used to associate the driver and its target device.
- **reg**: Register base address "`0x29590000`" and range "`0x4000`".
- **interrupts**: Hardware interrupt ID.
- **status**: The work status of the HDMI module. To enable the module, set this bit as "`okay`" or to disable the module, set this bit as "`disabled`".
- **clocks**: The clocks used by the HDMI module.
- **clock-names**: The names of the above clocks.
- **resets**: The reset signals used by the HDMI module.
- **reset-names**: The names of the above reset signals.

The following is an example of the HDMI configuration in the file `jh7110-common.dtsi`.

```
&hdmi {
 status = "okay";
 pinctrl-names = "default";
 pinctrl-0 = <&inno_hdmi_pins>;

 hdmi_in: port {
  #address-cells = <1>;
  #size-cells = <0>;
  hdmi_in_lcdc: endpoint@0 {
   reg = <0>;
```

```
    remote-endpoint = <&dc_out_dpi1>;
  };
 };
};
```

In the above code block, the parameters of **pinctrl-names** and **pinctrl-0** are used to configure the HDMI IOMUX pin configuration settings.

2.2. Driver Configuration

The following code block shows the driver configuration.

```
CONFIG_DRM_VERISILICON=y
CONFIG_STARFIVE_INNO_HDMI=y
```

2.3. Kernel Menu Configuration

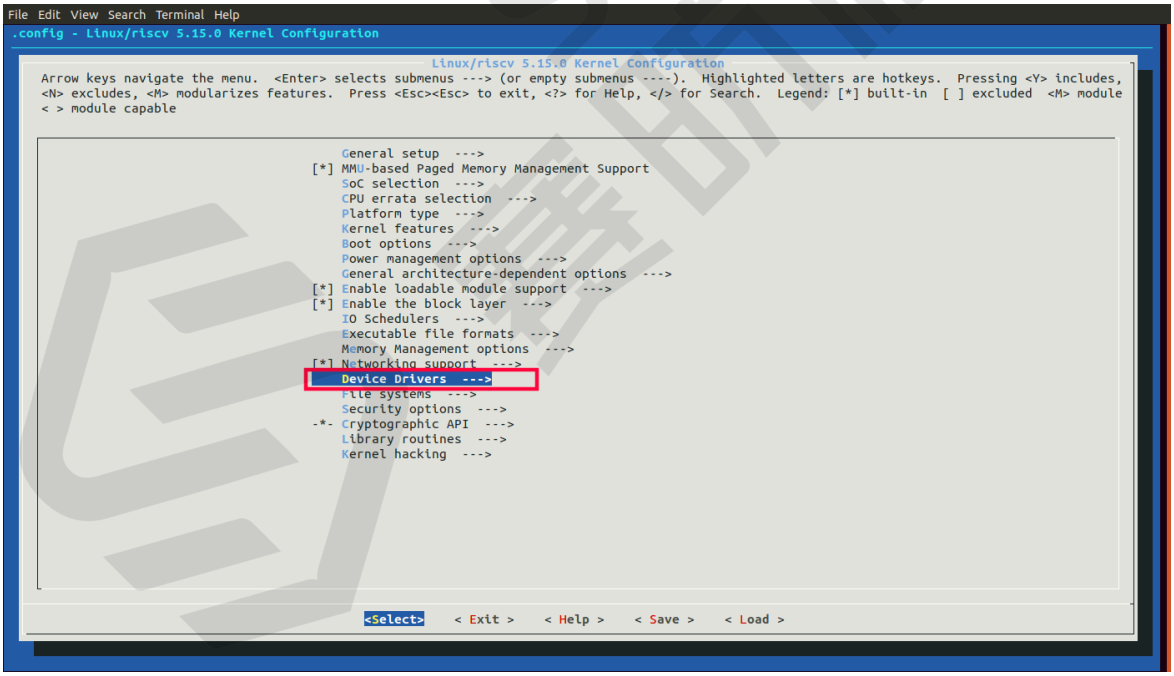
Follow the steps below to enable the kernel configuration for HDMI.

1. Under the root directory of `freelight-u-sdk`, type the following command to enter the kernel menu configuration GUI.

   ```
   make linux-menuconfig
   ```

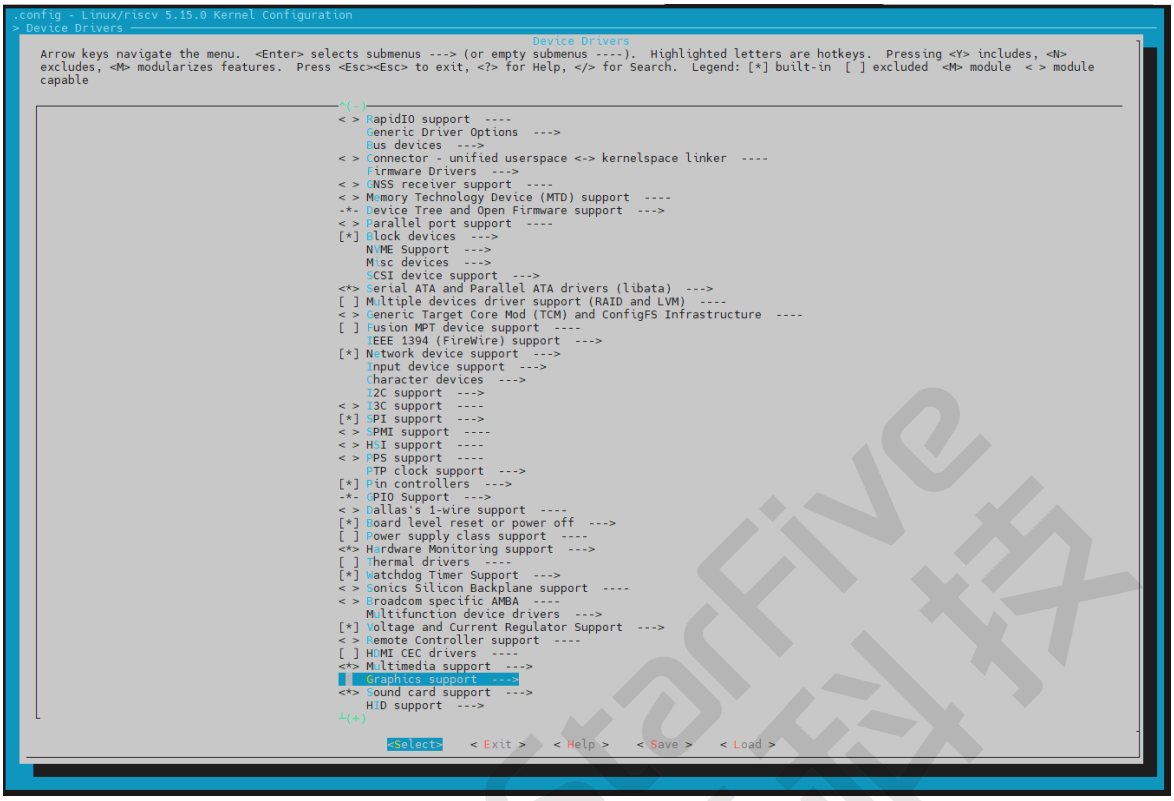
2. Enter the **Device Drivers** menu.

   Figure 2-1 Device Drivers



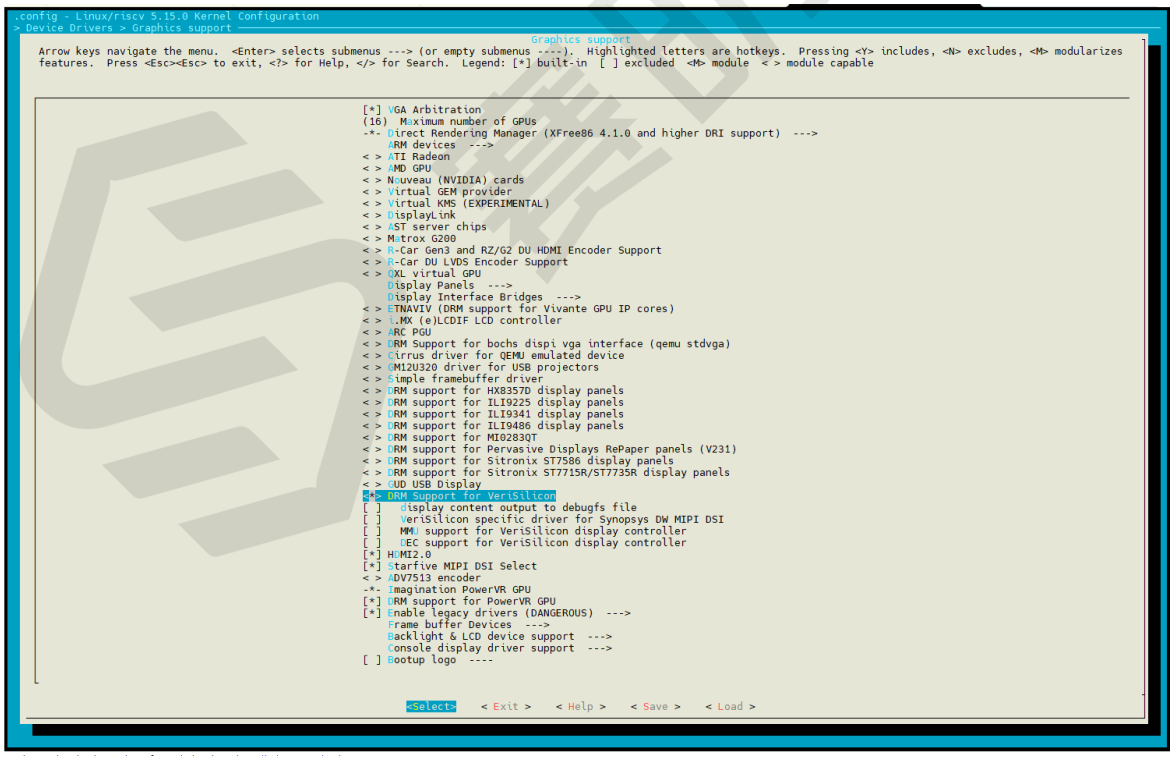
3. Enter the **Graphics support** menu.

Figure 2-2 Graphics Support



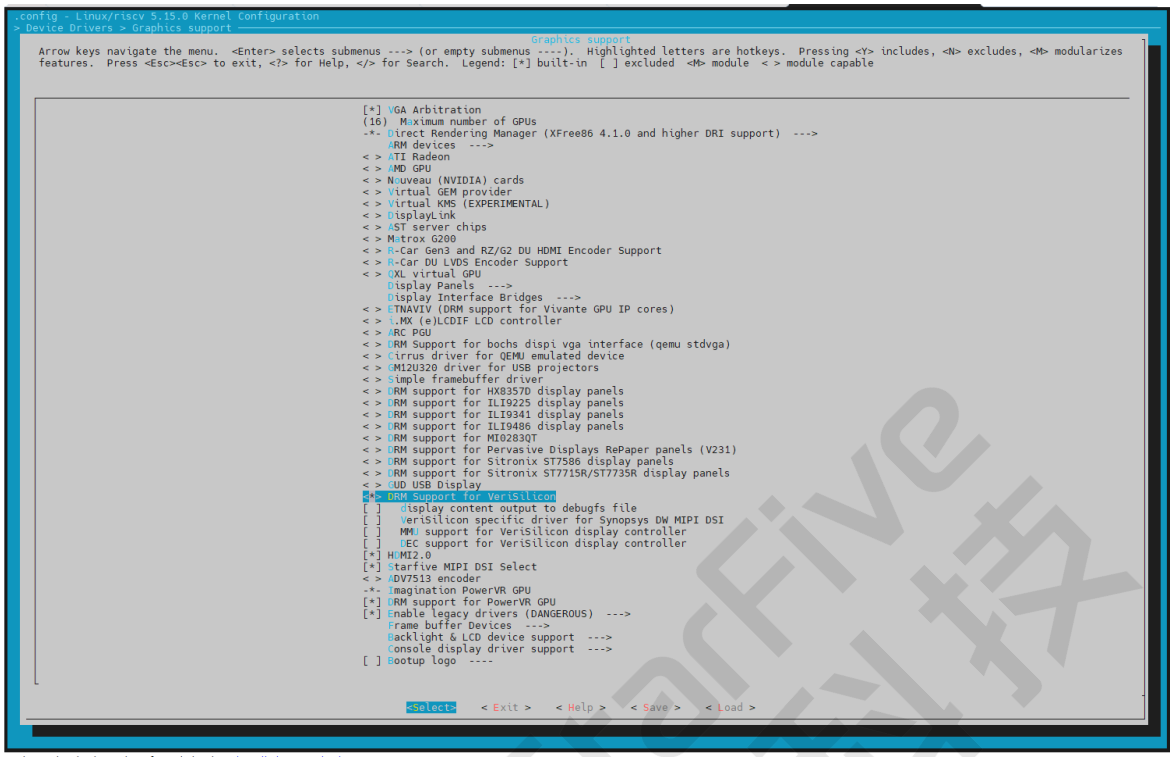
4. In the **Graphics support** menu, select the **DRM Support** option to enable video output.

Figure 2-3 DRM Support



5. In the **Graphics support** menu, select the **HDMI2.0** option.

Figure 2-4 HDMI2.0



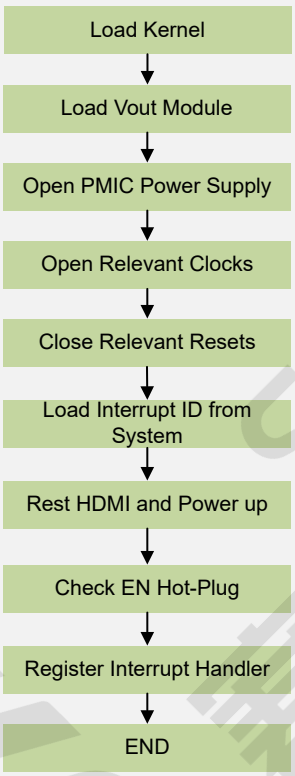
6. Save your change before you exit the kernel configuration dialog.

3. Work Process

3.1. Initialization Process

The following diagram shows the HDMI initialization process for JH7110.

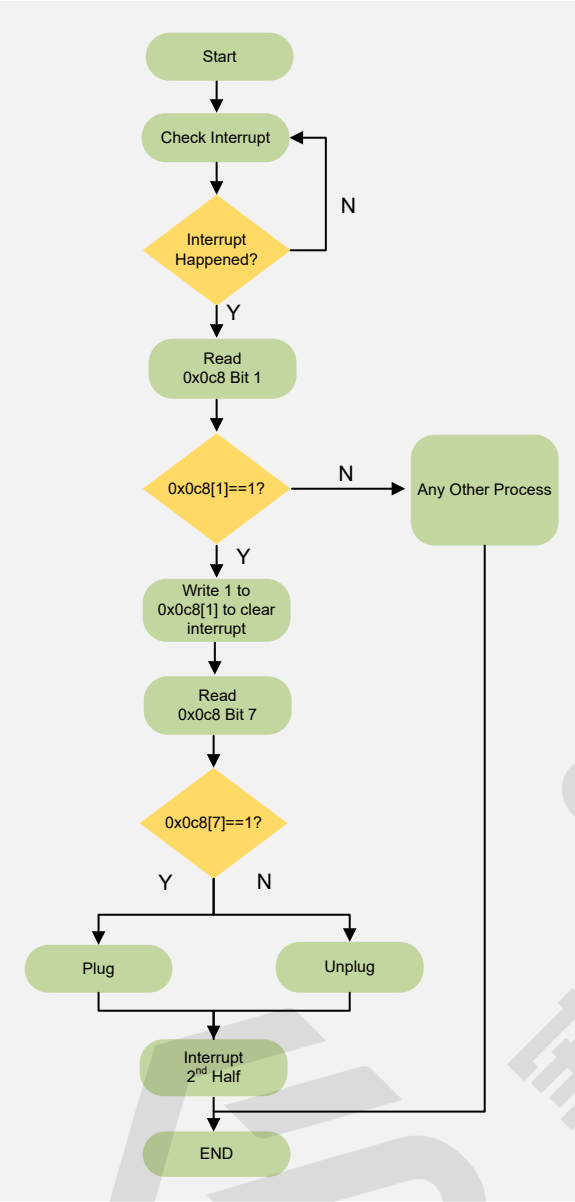
Figure 3-1 Initialization Process



3.2. Plug and Unplug Process

The following diagram shows the HDMI plug and unplug procedure.

Figure 3-2 Plug and Unplug Process



4. Debug HDMI

4.1. Test Case Configuration

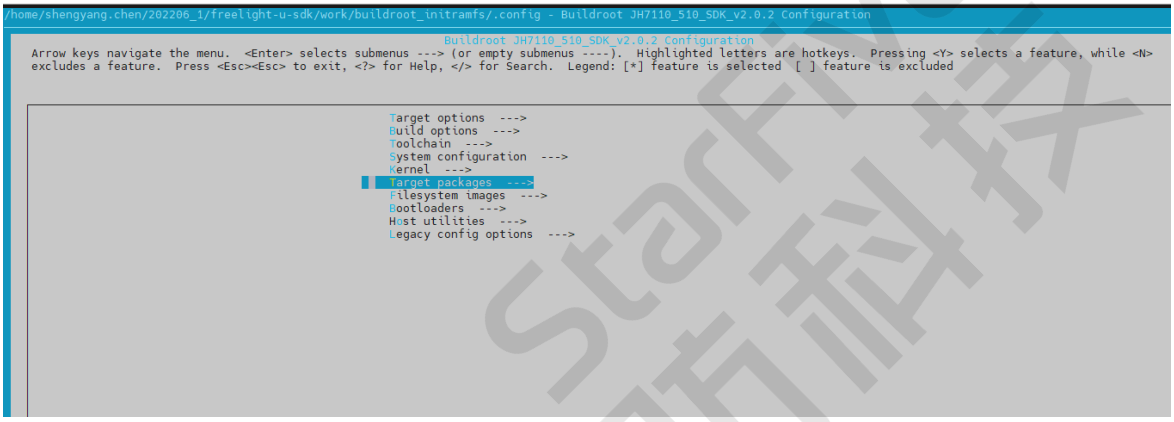
Follow the steps below to enable the kernel configuration for HDMI.

1. Under the root directory of `freelight-u-sdk`, type the following command to enter the kernel menu configuration GUI.

```
make linux-menuconfig
```

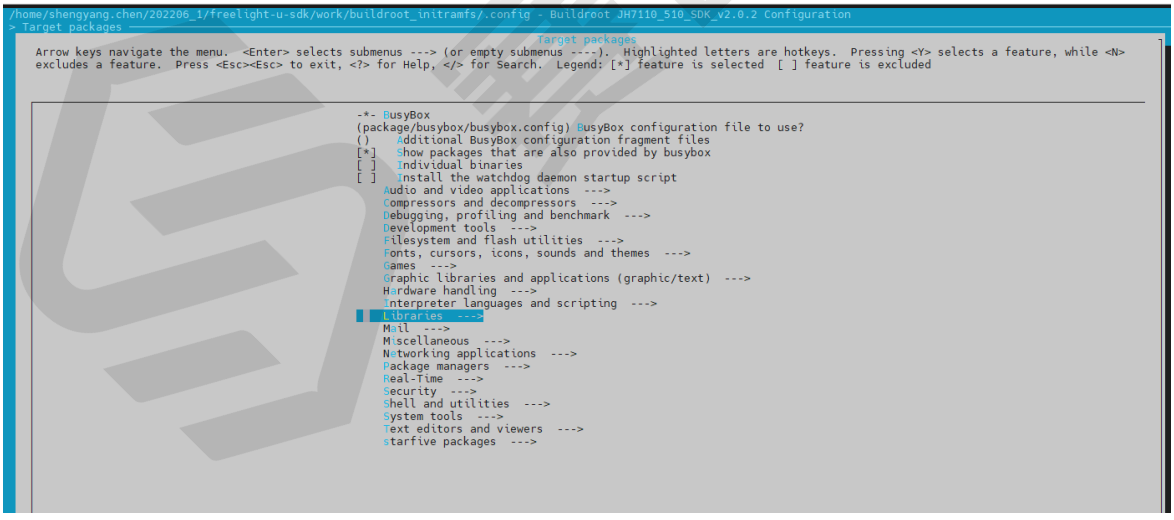
2. Enter the **Target packages** menu.

Figure 4-1 Target Packages



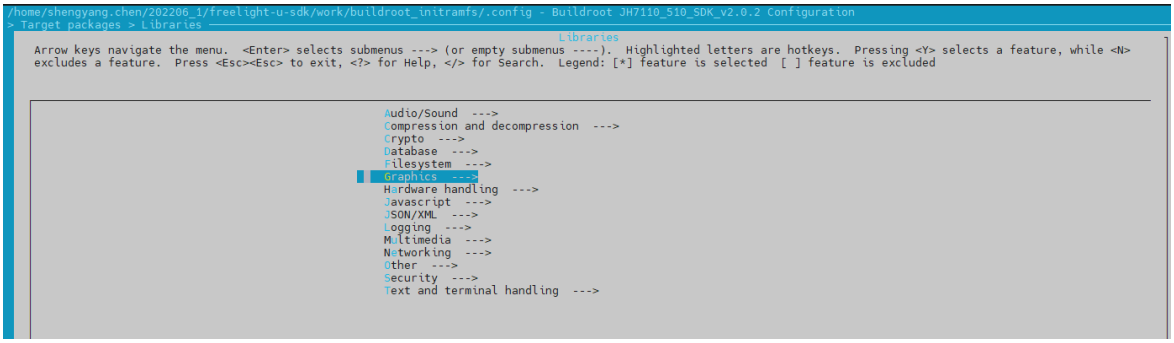
3. Enter the **Libraries** menu.

Figure 4-2 Libraries



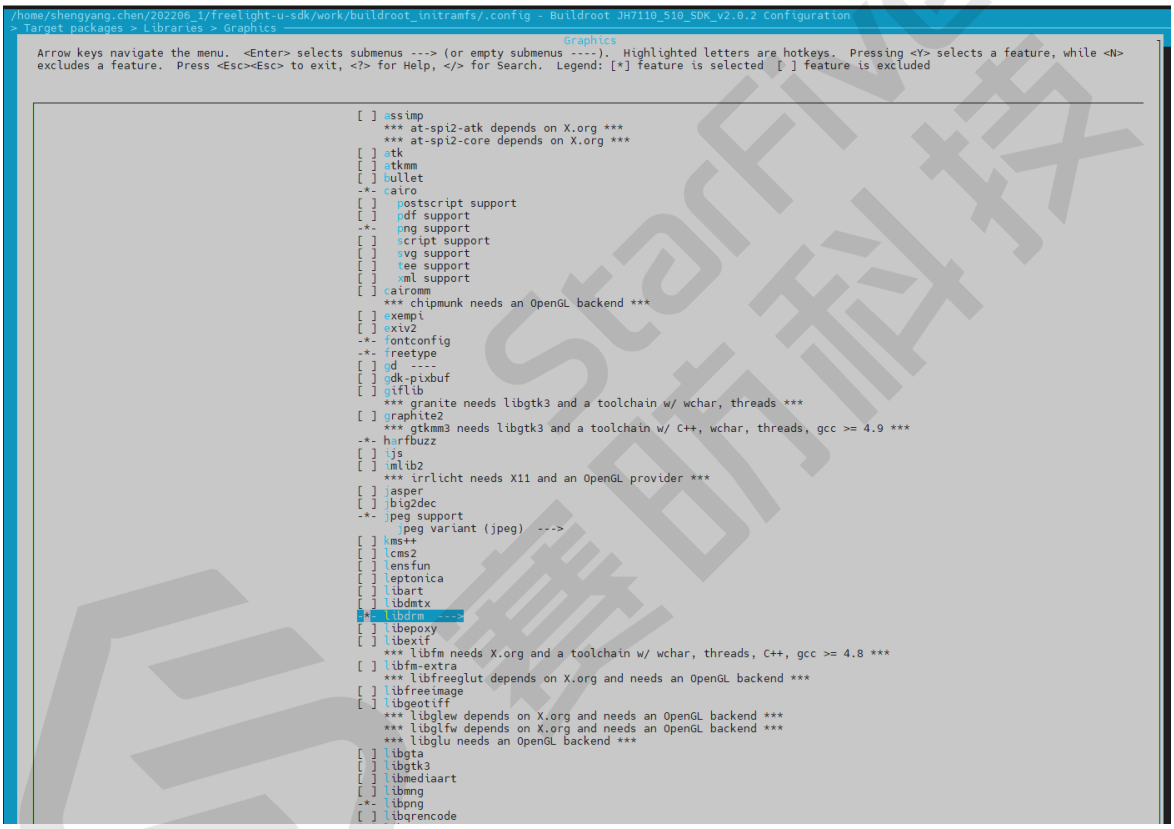
4. Enter the **Graphics** menu.

Figure 4-3 Graphics



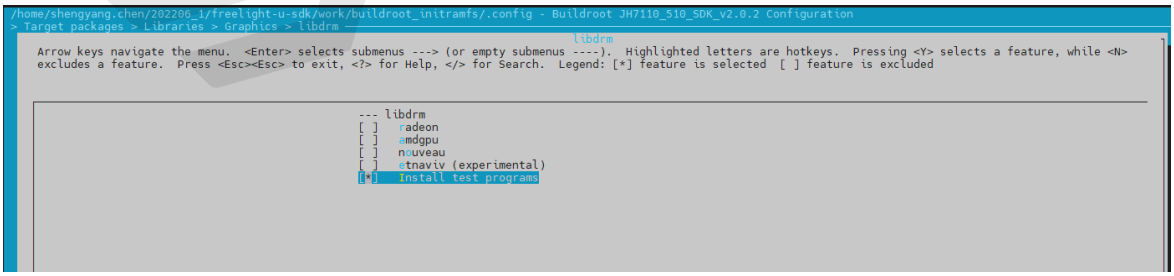
5. Enter the **libdrm** menu.

Figure 4-4 libdrm



6. Select the **Install test programs** option, or you may select ALL options under this menu.

Figure 4-5 Install Test Programs



Result: After you have completed all the above configuration, you can use the `modetest` tool in kernel for testing.

7. Save your change before you exit the kernel configuration dialog.

4.2. Debug Display

Follow the steps below to debug the display functions for your JH7110.

1. Follow the steps in Test Case Configuration *(on page 17)* to configure the test environment.

   Note:
   Make sure you have configured **libdrm** and **modetest** before compiling and burning an image.

2. After you have completed the kernel start-up, use the following command to verify the display functions and connection status.

   ```
   modetest -M starfive
   ```

   The following legends and tables display an example output and descriptions.

   - Debug output 1:

Figure 4-6 Debug Display 1

```
# modetest -M starfive
Encoders:
id      crtc    type    possible crtcs  possible clones
115     0       TMDS    0x00000001      0x00000001
117     0       DSI     0x00000002      0x00000002

Connectors:
id      encoder status          name            size (mm)       modes   encoders
116     0       connected       HDMI-A-1        0x0             10      115
  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: phsync, pvsync; type: driver
  #1 1920x1080 59.94 1920 2008 2052 2200 1080 1084 1089 1125 148352 flags: phsync, pvsync; type: driver
  #2 1920x1080 50.00 1920 2448 2492 2640 1080 1084 1089 1125 148500 flags: phsync, pvsync; type: driver
  #3 1280x720 60.00 1280 1390 1430 1650 720 725 730 750 74250 flags: phsync, pvsync; type: driver
  #4 1280x720 59.94 1280 1390 1430 1650 720 725 730 750 74176 flags: phsync, pvsync; type: driver
  #5 1280x720 50.00 1280 1720 1760 1980 720 725 730 750 74250 flags: phsync, pvsync; type: driver
  #6 1280x720 48.00 1280 2240 2280 2500 720 725 730 750 90000 flags: phsync, pvsync; type: driver
  #7 1280x720 47.95 1280 2240 2280 2500 720 725 730 750 89910 flags: phsync, pvsync; type: driver
  #8 640x480 60.00 640 656 752 800 480 490 492 525 25200 flags: nhsync, nvsync; type: driver
  #9 640x480 59.94 640 656 752 800 480 490 492 525 25175 flags: nhsync, nvsync; type: driver
  props:
        1 EDID:
                flags: immutable blob
                blobs:

                value:
                        00ffffffffffff004a8b201980102019
                        001e0103800000078ecee91a3544c9926
                        0f5054230800d1c0b300950081006140
                        4540814081c0023a801871382d40582c
                        250058c31000001e000000fc00000a20
                        2020202020202020202020000000ff0000
                        0a2020202020202020202020000000fd
                        00383f545413000a202020202020001a3
                        020332f24f04051013141f6c6c6c276c
                        6c6c4b4ce200d5e305c00023097f0783
                        01000067030c001000383ce606050169
                        694f023a801871382d40582c250058c3
                        1000001e011d8018711c1620582c2500
                        58c31000009e000000000000000000000
                        000000000000000000000000000000000
                        00000000000000000000000000000007a
        2 DPMS:
                flags: enum
                enums: On=0 Standby=1 Suspend=2 Off=3
                value: 0
        5 link-status:
                flags: enum
                enums: Good=0 Bad=1
                value: 0
        6 non-desktop:
                flags: immutable range
                values: 0 1
```

Table 4-1 Debug Display 1

Legend	Label	Description
①	**possible crtcs**	Available *Cathode Ray Tube Controller (CRTC)* devices
②	**status**	Whether the display connector is connected or not
③	**name**	The name (type) of the display connector

Legend	Label	Description
④	**encoders**	The connected encoders
⑤	**modes**	The supported display modes
⑥	**value**	The *Extended Display Identification Data (EDID)* of the screen

◦ Debug output 2:

Figure 4-7 Debug Display 2

Table 4-2 Debug Display 2

Legend	Label	Description
①	**id**	The CRTC 0x00000001 mentioned in row ① of table Table 4-1 : Debug Display 1 *(on page 19)*, which means the CRTC is available for use.

Legend	Label	Description
②	**id**	The CRTC 0x00000002 mentioned in row ① of table Table 4-1 : Debug Display 1 *(on page 19)*, which means the CRTC is available for use.

Note:
If the displayed CRTC is 0x00000003, both of the CRTCs are available for use.

◦ Debug output 3:

Figure 4-8 Debug Display 3

Table 4-3 Debug Display 3

Legend	Description
①	The CRTC and its connected plane

4.3. Test Example

For HDMI Output

The following command shows an example for testing the HDMI output.

```
modetest -M starfive -D 0 -a -s 116@31:1920x1080 -P 39@31:1920x1080@RG16 -Ftiles
```

The following list provides explanations for the parameters in the above example command.

- **116@31:1920x1080** - <Connector ID>@<CRTC ID>: <Resolution>
- **39@31:1920x1080@RG16** - <Plane ID>@<CRTC ID>: <Resolution>@<Format>

Output Result

The following photo shows the output generated from the above example command.

Figure 4-9 Test Example