Chapter 18
Supporting the i.MX53 Reference Board DISP0 LCD

This chapter explains how to support a new LCD on an i.MX53-based board, using display port 0. There are two options for adding support for a new LCD panel without modifying the BSP: letting the BSP calculate the timings using VESA defaults or reducing the blanking time. VESA and reduced blanking work for many LCDs but fail for some devices because of timing configuration constraints. For those devices, we need to modify the BSP and set the proper timing values. Modifying the boot arguments also allows us to include support for the new driver from LTIB device driver menu, call initialization routines, and load the driver by using the boot arguments.

This chapter focuses on the synchronous Parallel0 RGB interface. Common display cards can be attached to this interface. It provides connectivity for the Chunghwa CLAA057VA01CT VGA LCD and the Chunghwa CLAA070VC01 WVGA LCD panel.

Be aware that the DI RGB interface is multiplexed with all other asynchronous parallel interfaces. Therefore, users cannot send data to a synchronous display and another asynchronous parallel display device at the same time in the same DI. Instead, the i.MX53 sends data to the asynchronous panel (smart display) while the synchronous interface is inactive (during horizontal and vertical back porch and front porches). For this reason, the smart display’s frame rate can be affected when multiple displays are attached to the i.MX53.

18.1 Supported Display Interfaces

The i.MX53 processor supports the display interfaces shown in Figure 18-1.

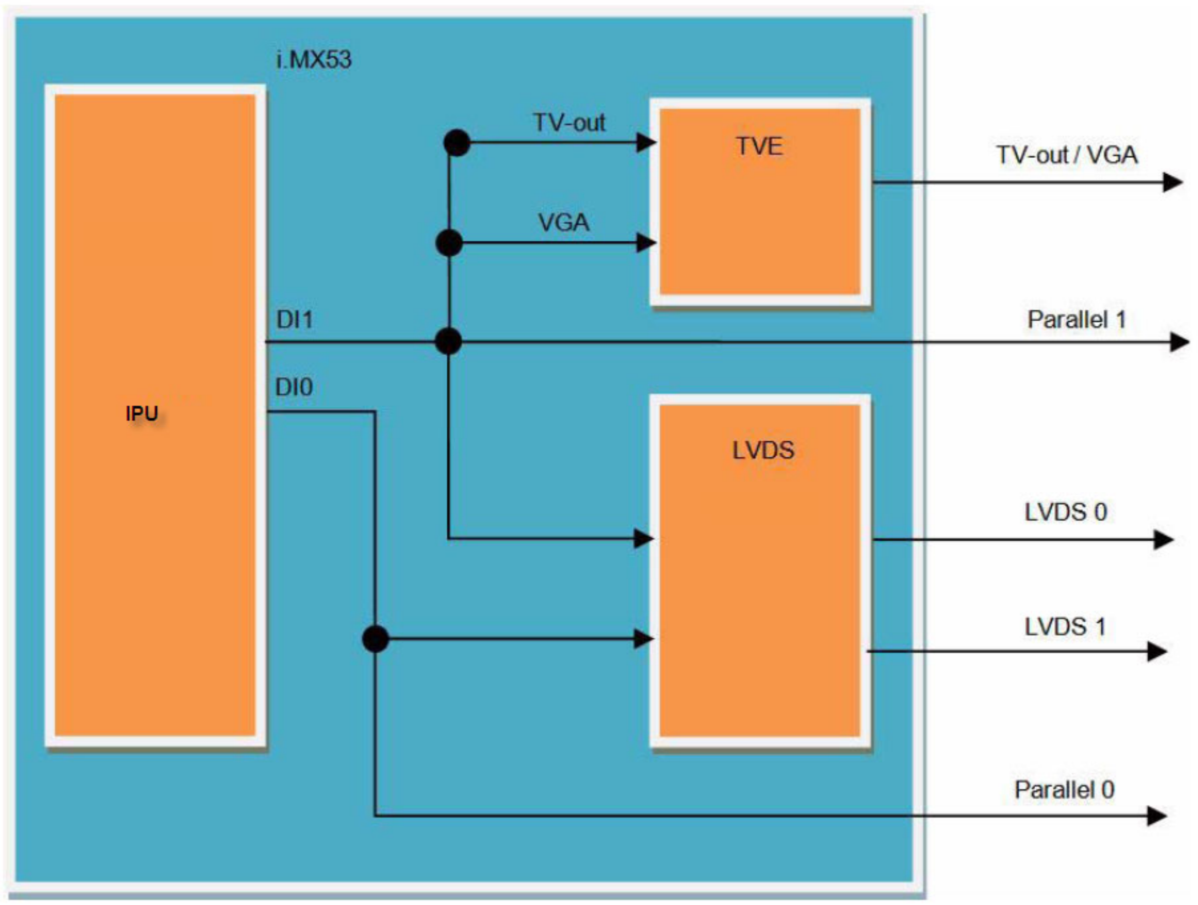


Figure 18-1. Available Display Interfaces

Table 18-1 describes the available interfaces.

Table 18-1. Available Interfaces

Feature	IPU (in i.MX53)
Number of ports	Two: Full dual-display support
Legacy I/F	Parallel and serial. Synchronous (for display refresh) and asynchronous (to memory) Very flexible—glue-less connection to RAM-less displays, display controllers, and TV encoders.
MIPI/DSI high-speed I/F	Full Support Up to 2 lanes, 800 Mbps per lane
Analog TV-out (composite, S-video, component)	Driven by TVE (Not supported on TO1) Up to 720p at 60 fps or 1080i at 30 fps (720p: 1280x720, 1080i: 1920x1080)

Table 18-1. Available Interfaces

Feature	IPU (in i.MX53)
VGA output	Driven by TVE (Not supported in TO1) Up to WSXGA+ @ 60 Hz, 24 bpp (WSXGA+: 1680x1050)
LVDS I/F	Up to UXGA or 2xWXGA @ 60 Hz, 24 bpp (UXGA: 1600x1200, WXGA: 1366x768
Note: VGA output is not supported on i.MX53 TO1 processors	

18.2 Adding Support for an LCD Panel

To provide an example for how to add support for an LCD panel, this section shows the code and commands used for adding the support for the CLAA057VA01CT LCD. CLAA057VA01CT is a 5.7" color TFT-LCD (Thin Film Transistor Liquid Crystal Display) module. It is composed of an LCD panel, driver ICs, control circuit, touch screen, and LED backlight. The 5.7" screen produces a high resolution image that is composed of 640 × 480 pixel elements in a stripe arrangement. It uses a 16 bit RGB signal input to display 262K colors.

Figure 18-2 shows the interface between an i.MX53-based board and Chunghwa CLAA057VA01CT 5.7” VGA LCD.

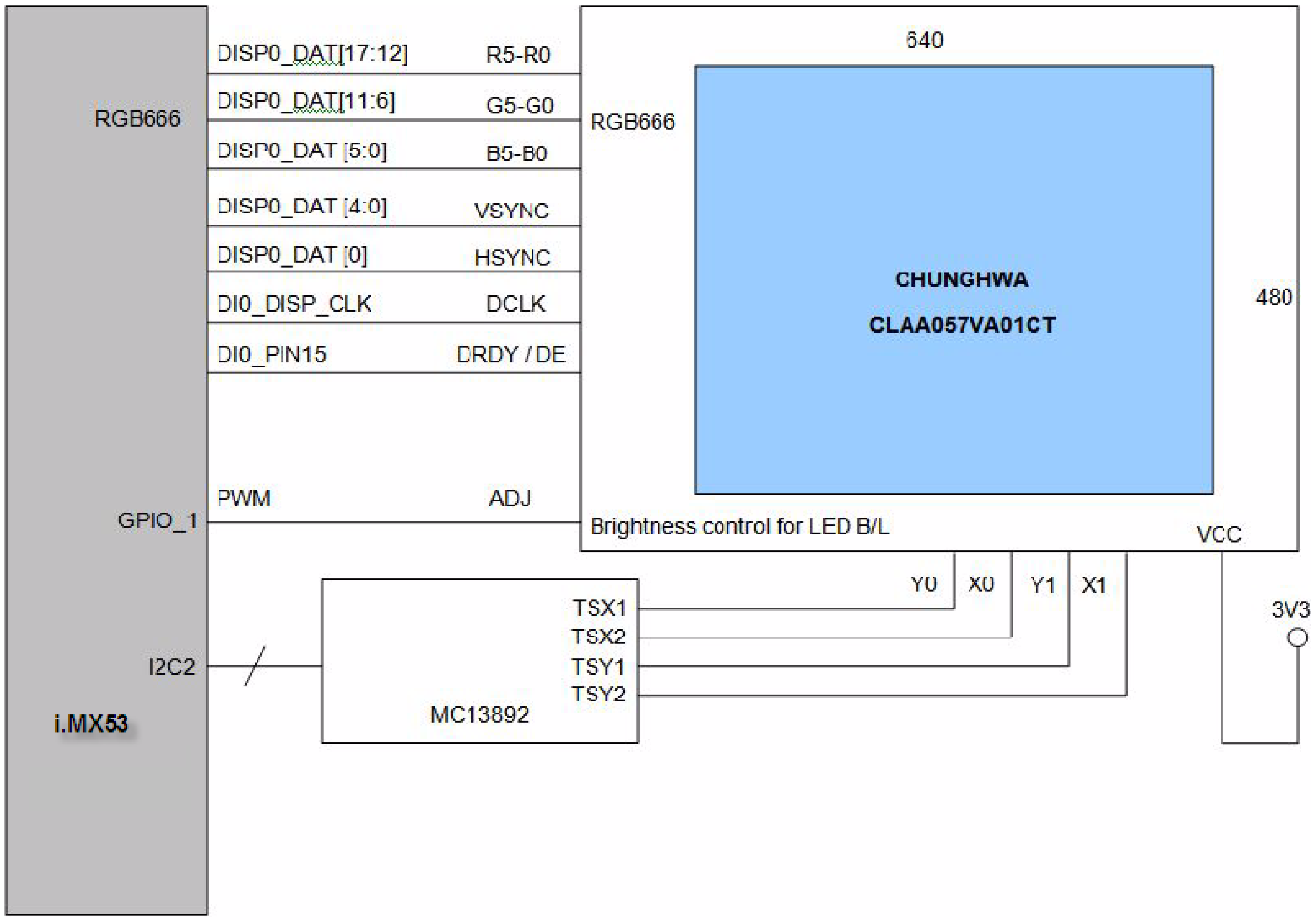


Figure 18-2. Interface

The LCD panel requires HSYNC, VSYNC, DE, PIXCLK, and part of the RGB data interface (DISPB_DATA[17:0]). No additional signals, such as a reset signal or serial interface initialization routine commands (SPI or I2C), are required. The backlight unit is controlled by a GPIO signal generated by the i.MX53 (PWM), and the PMIC controls the touch panel interface. The display card includes a connection for this panel.

Table 18-2 shows the timing parameters.

Table 18-2. Timing Parameters

Parameter	Symbol	Min	Typ	Max	Unit
Screen Height or vertical period	VP	515	525	560	Line
VSYNC pulse width	VSW	1	1	1	Line
Vertical back porch	VBP	34	34	34	Line
Vertical front porch	VFP	1	11	46	Line

Table 18-2. Timing Parameters (continued)

Parameter	Symbol	Min	Typ	Max	Unit
Active frame height	VDISP	—	480	—	Line
Vertical refresh rate	FV	55	60	65	Hz
Screen width or horizontal cycle	HP	750	800	900	PIXCLK
HSYNC pulse width	HSW	1	1	1	PIXCLK
Horizontal back porch	HBP	46	46	46	PIXCLK
Horizontal front porch	HFP	64	114	214	PIXCLK
Active frame width	HDISP	—	640	—	PIXCLK

18.3 Modifying Boot Kernel Parameters to Support a New LCD

Users can use the video and di1_primary kernel parameters to change all timing and interface aspect ratios without writing a single line of code by changing the settings through the default driver.

18.3.1 Setting the Video Kernel Parameter

The video kernel parameter is a multipurpose parameter used to configure display features in either display port 0 or display port 1. It controls the following features:

- Display resolution
- Pixel color depth
- Refresh rate
- IPU display output interface format

See the modedb.txt file located at `Documentation/fb/modedb.txt` for specific parameter information.

To set the parameter information for the video argument, use the following format. Variables between square brackets are optional.

```
<interface_id>:<ipu_out_fmt>,<xres>x<yres>[M][R][-<bpp>][@<refresh>][i][m]<name>[-<bpp>][@
<refresh>]
```

Table 18-3 defines the variables.

Table 18-3. Parameter Information

Argument Name	Definition	Units	Values
interface_id	Display interface id (DI0/DI1)	NA	mxcdi0fb, mxcdi1fb
interface_pix_fmt	Display Interface output format	NA	RGB565, RGB666, RGB24, YUV444
name	Video mode name	NA	String name
xres	Horizontal resolution	pixels	Decimal value
yres	Vertical resolution	lines	Decimal value

Table 18-3. Parameter Information

Argument Name	Definition	Units	Values
M	Timing calculated using VESA(TM)	NA	M
R	Timing using reduced blanking	NA	R
bpp	Bits per pixel on frame buffer	bits	Decimal value (16 or 24)
refresh	LCD refresh rate	Hz	Decimal value

When <name> is included in the mode_option argument parameters, the timing is not calculated. Instead, it is extracted from BSP code. Valid default modes can be found at `linux/drivers/video/modedb.c` and in files placed at `linux/drivers/video/mxc folder`.

Example 18-1. DVI Monitor Connected to Display Port 0

For a DVI monitor connected to display port 0, the kernel command is

```
video=mxcdi0fb:RGB24,1024x768M-16@60
```

Table 18-4 shows how the values in this example correspond to the argument names defined in Table 18-3.

Table 18-4. XGA DVI Monitor Example Variables

Argument Name	Value	Definition
interface_id	mxcdi0fb	Display interface 0 settings
interface_pix_fmt	RGB24	RGB bus is 24 bits width DISP0_DAT[23:0]- RGB888
name	Not used in this command	—
xres	1024	1024 pixels (Horizontal)
yres	768	768 lines (vertical)
M	M	Timings calculated using VESA(TM)
R	Not used in this command	—
bpp	16	Frame buffer is 16 bits per pixel
refresh	60	60 Hz

Example 18-2. VGA LCD Connected to Display Port 0

For a VGA LCD connected to display port 0, the kernel command is

```
video=mxcdi0fb:RGB565,800x480M-16@55
```

Table 18-5 shows how the values in this example correspond to the argument names defined in Table 18-3.

Table 18-5. VGA LCD Example Variables

Argument Name	Value	Definition
interface_id	mxcdi0fb	Display interface 0 settings
interface_pix_fmt	RGB565	RGB bus is 16 bits width DISP0_DAT[16:0]- RGB565
name	Not used in this command	—
xres	800	800 pixels (Horizontal)
yres	480	480 lines (vertical)
M	M	Timing calculated using VESA (TM)
R	Not used in this command	—
bpp	16	Frame buffer is 16 bits per pixel
refresh	55	55 Hz

Example 18-3. 720P TV Connected to Display Port 1

For a 720P TV connected to display port 1, the kernel command is `video=mxcdi1fb:YUV444,720P60`

Table 18-6 shows how the values in this example correspond to the argument names defined in Table 18-3.

Table 18-6. 720P TV Example Variables

Argument Name	Value	Definition
interface_id	mxcdi1fb	display interface 1 settings
interface_pix_fmt	YUV444	YUV444 output
name	720P60	Defined in linux/drivers/video/mxc/tve.c
xres	Not used in this command	From tve.c: 1280
yres	Not used in this command	From tve.c: 720
M	Not used in this command	—
R	Not used in this command	—
bpp	Not used in this command	—
refresh	Not used in this command	—

18.3.2 Setting the di1_primary Kernel Parameter

The di1_primary kernel parameter informs the kernel/driver that DI1 is the primary display interface instead of DI0, which is the default setting. Set this kernel parameter by adding the label di1_primary to the boot kernel arguments.

For example, the kernel command for an XGA LVDS connected to LVDS0 using DI1 as the primary display interface is as follows:

```
video=mxcdi0fb:RGB24,LDB-XGA di1_primary
```

18.3.3 Modifying the Bits per Pixel Setting

The default bits per pixel setting is 16 bits. To change the default value to another depth, modify the relevant lines in the mxc_ipuv3_fb.c file located at

```
<ltib dir>/rpm/BUILD/linux/drivers/video/mxc/mxc_ipuv3_fb.c
```

Example 18-4. Changing the Frame Buffer to 32 bpp

The following example code shows how to change the frame buffer from 16 bpp to 32 bpp.

```
static int mxcfb_probe(struct platform_device *pdev)
{
   ...

   if (!mxcfbi->default_bpp)
      mxcfbi->default_bpp = 16;
   // Change Default BPP to 32 bpp
   printk(KERN_INFO "mxcfb_probe() - default_bpp = 32\r\n");
   mxcfbi->default_bpp = 32;
   ...
   return ret;
}
```

Check the frame buffer bpp and other settings in the /sys/class folder. The output should look like the following:

```
root@freescale ~$ cd /sys/class/graphics/fb0/
root@freescale /sys/devices/platform/mxc_sdc_fb.1/graphics/fb0$ ls
bits_per_pixel     device             pan                subsystem
blank              fsl_disp_property  power              uevent
console            mode               rotate             virtual_size
cursor             modes              state
dev                name               stride
root@freescale /sys/devices/platform/mxc_sdc_fb.1/graphics/fb0$ cat bits_per_pixel
32
```

Note that the final line shows the bits per pixel to be 32, reflecting our change from the default of 16 bpp.

18.3.4 Modifying Display Timing for CLAA057VA01CT Using Kernel Parameters

By using the video and di1_primary kernel parameters, the frame buffer driver is able to change all timing and interface aspect ratios. Timing is calculated using the VESA standard.

The video parameter format is `<interface_id>:<ipu_out_fmt>,<xres>x<yres>[M][R][-<bpp>][@<refresh>][i][m]<name>[-<bpp>][@<refresh>]` with variables between square brackets optional.

Table 18-7 provides sample values for an interface.

Table 18-7. Sample Values

Argument Name	Value	Definition
interface_id	mxcdi0fb	display interface 0 settings
interface_pix_fmt	RGB666	RGB bus is 18 bits width DISP0_DAT[17:0]- RGB565
name	Not used in this command	—
xres	640	640 pixels (Horizontal)
yres	480	480 lines (vertical)
bpp	16	Frame buffer is 16 bits per pixel
refresh	55	55 Hz
M	M	Timing calculated using VESA(TM)
R	Not used in this command	—

For these sample values, the video parameter is `video=mxcdi0fb:RGB666,640x480M-16@55`

Example 18-5. Kernel Image Stored in the SD; file system from NFS

The following commands are for a kernel image stored in the SD with a file system loaded from NFS. All commands are executed on U-Boot:

To configure the network, use the following:

```
EVK U-Boot > setenv serverip 10.112.98.65
EVK U-Boot > setenv fec_addr 00:04:9f:00:ea:d3
EVK U-Boot > setenv ethaddr 00:04:9f:00:ea:d3
EVK U-Boot > setenv bootfile uImage
EVK U-Boot > saveenv
EVK U-Boot > reset
```

To load uImage from server, use the following:

```
EVK U-Boot > bootp $ {loadaddr} ernesto/53/uImage
```

To write uImage to SD, use the following:

```
EVK U-Boot > mmc write 0 ${loadaddr} 0x800 0x1800
Set NFS file system path
EVK U-Boot > setenv nfsroot /tftpboot/ernesto/ltib
```

To configure boot arguments to use NFS and CLAA057VA01CT LCD panel, use the following:

```
EVK U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/nfs ip=dhcp
video=mxcdi0fb:RGB666,640x480M-16@55 nfsroot=${serverip}:${nfsroot},v3,tcp'
Launch OS image
EVK U-Boot > run bootcmd_mmc
```

18.4 Adding Support for a New LCD

If neither VESA nor reducing the blanking calculation works for your LCD or if you need a special function, add the support for the new LCD in the BSP.

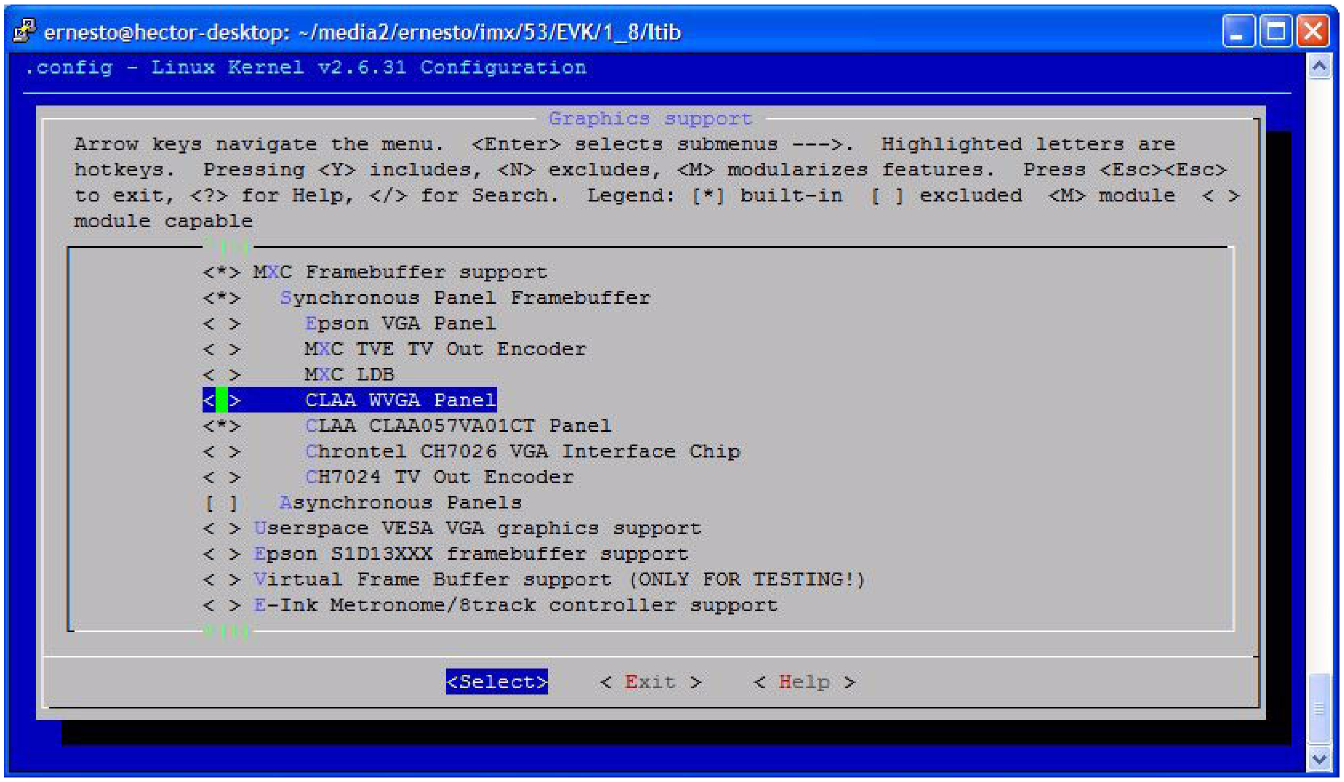
Perform the following steps to modify the i.MX53 BSP to add the support for synchronous panels:

1. Add a display entry in the ltib catalog.
2. Create the madglobal LCD panel file.
3. Add compilation flag for the new display.
4. Configure LCD timings and display interface.
5. Use boot command to select the new LCD.

The following subsections describe these steps in detail.

18.4.1 Adding a Display Entry in the ltib Catalog

Select specific display drivers in `Device Drivers > Graphics support`.



```
ernesto@hector-desktop: ~/media2/ernesto/imx/53/EVK/1_8/ltib
.config - Linux Kernel v2.6.31 Configuration
                         Graphics support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are
  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc>
  to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >
  module capable

          <*> MXC Framebuffer support
          <*>   Synchronous Panel Framebuffer
          < >     Epson VGA Panel
          < >     MXC TVE TV Out Encoder
          < >     MXC LDB
          < >     CLAA WVGA Panel
          <*>     CLAA CLAA057VA01CT Panel
          < >     Chrontel CH7026 VGA Interface Chip
          < >     CH7024 TV Out Encoder
          [ ]   Asynchronous Panels
          < > Userspace VESA VGA graphics support
          < > Epson S1D13XXX framebuffer support
          < > Virtual Frame Buffer support (ONLY FOR TESTING!)
          < > E-Ink Metronome/8track controller support

                    <Select>    < Exit >    < Help >
```

Figure 18-3. Graphics Support Options Menu

To add an entry for a new LCD, perform the following steps:

1. Enter the i.MX53 display specific folder as follows.
   ```
   $ cd <ltib dir>/rpm/BUILD/linux/drivers/video/mxc
   ```
2. Open the Kconfig file with the command `gedit Kconfig &`
3. Use the following code to add the entry where you want it to appear.

```
config FB_MXC_CLAA057VA01CT_SYNC_PANEL
       depends on FB_MXC_SYNC_PANEL
       tristate "CLAA CLAA057VA01CT Panel"
```

18.4.2 Creating the LCD Panel File (initialization, reset, power settings, backlight)

Because power settings are handled by the ATLAS APL PMIC and other voltage regulators, the display driver must configure the APL PMIC during initialization to set up the power voltage configuration if this has not already been done. Also, the reset waveform and initialization routine must be included. To do these tasks, create an LCD file with panel-specific functions at the following location:

```
<ltib dir>/rpm/BUILD/linux/drivers/video/mxc/mxcfb_CLAA057VA01CT.c
```

WARNING

Before connecting an LCD panel to the i.MX53 board, check whether the LCD is powered with the proper supply voltages and whether the display data interface has the correct VIO value. Incorrect voltages and values may harm the device.

The LCD file must include the definition of four basic functions described in Table 18-8 and can include other functions and macros as needed.

Table 18-8. Required Functions

Function Name	Function Declaration	Description
lcd_probe	static int __devinit lcd_probe(struct platform_device *pdev)	Called when the LCD module is loaded. It should contain, pmic configuration, reset, power on sequence and the initialization routine.
lcd_remove	static int __devexit lcd_remove(struct platform_device *pdev)	Called when the LCD module is removed. It should contain power off pmic configuration, power off sequence and the de-initialization routine.
lcd_suspend	static int lcd_suspend(struct platform_device *pdev, pm_message_t state)	Not always implemented, but used for enhance low power modes on the device. Usually called when system goes to suspend mode.
lcd_resume	static int lcd_resume(struct platform_device *pdev)	Not always implemented, but used for enhance low power modes on the device. Usually called when system returns from suspend mode.

Next, create a platform device that can be loaded and unloaded. This example declares the new platform device using the devices.h and devices.c files located at:

```
<ltib dir>//rpm/BUILD/linux/arch/arm/mach-mx5/
```

1. Declare the plaform device on madglobal:devices.h using the following:

   ```
   extern struct platform_device mxc_disp_devices[];
   ```

2. Add a new entry on madglobal:devices.c using the following:

   ```
   struct platform_device mxc_disp_devices[] = {
      {
         .name = "lcd_CHUNGHWA_CLAA057VA01CT",
         .id = 0,
   ```

```
    },
};
```

Be careful to use the same name for the new platform device entry as the name included in madglobal:mxcfb_CLAA057VA01CT.c for the driver.

```
static struct platform_driver lcd_driver = {
 .driver = {
     .name = "lcd_CHUNGHWA_CLAA057VA01CT"},
 .probe = lcd_probe,
 .remove = __devexit_p(lcd_remove),
 .suspend = lcd_suspend,
 .resume = lcd_resume,
};
```

3. Register the device at `<ltib dir>//rpm/BUILD/linux/arch/arm/mach-mx5/madglobal:mxc53_<reference board name>.c` by using the following code:

```
static int __init mxc_init_fb(void)
{
 ....
 mxc_register_device(&mxc_disp_devices[0], NULL);
 ....
 return 0;
}
```

18.4.3 Adding the Compilation Flag for the New Display

After the LCD file has been created and the entry has been added to the Kconfig file, modify the makefile to include the LCD file in the compilation by using the code shown below. The makefile is in the same folder as the new LCD file: `<ltib dir>/rpm/BUILD/linux/drivers/video/mxc/makefile`

```
ifeq ($(CONFIG_ARCH_MX21)$(CONFIG_ARCH_MX27)$(CONFIG_ARCH_MX25),y)
        obj-$(CONFIG_FB_MXC_TVOUT)              += fs453.o
        obj-$(CONFIG_FB_MXC_SYNC_PANEL)         += mx2fb.o mxcfb_modedb.o
        obj-$(CONFIG_FB_MXC_EPSON_PANEL)        += mx2fb_epson.o
else
ifeq ($(CONFIG_MXC_IPU_V1),y)
        obj-$(CONFIG_FB_MXC_SYNC_PANEL)         += mxcfb.o mxcfb_modedb.o
else
        obj-$(CONFIG_FB_MXC_SYNC_PANEL)         += mxc_ipuv3_fb.o
endif
        obj-$(CONFIG_FB_MXC_EPSON_PANEL)        += mxcfb_epson.o
        obj-$(CONFIG_FB_MXC_EPSON_QVGA_PANEL)   += mxcfb_epson_qvga.o
        obj-$(CONFIG_FB_MXC_TOSHIBA_QVGA_PANEL) += mxcfb_toshiba_qvga.o
        obj-$(CONFIG_FB_MXC_SHARP_128_PANEL)    += mxcfb_sharp_128x128.o
endif
obj-$(CONFIG_FB_MXC_EPSON_VGA_SYNC_PANEL)   += mxcfb_epson_vga.o
obj-$(CONFIG_FB_MXC_CLAA_WVGA_SYNC_PANEL)   += mxcfb_claa_wvga.o
obj-$(CONFIG_FB_MXC_CLAA057VA01CT_SYNC_PANEL)    += mxcfb_CLAA057VA01CT.o
obj-$(CONFIG_FB_MXC_TVOUT_CH7024)           += ch7024.o
obj-$(CONFIG_FB_MXC_TVOUT_TVE)              += tve.o
obj-$(CONFIG_FB_MXC_LDB)                    += ldb.o
obj-$(CONFIG_FB_MXC_CH7026)        += mxcfb_ch7026.o
#obj-$(CONFIG_FB_MODE_HELPERS)    += mxc_edid.o
```

Note that a new object, mxcfb_CLAA057VA01CT.o, is created when CONFIG_FB_MXC_CLAA057VA01CT_SYNC_PANEL flag is set. The LCD module with the initialization and de-initialization routines is only available to the kernel after this object has been created. If the LCD does not need a particular configuration, you may omit the usage of the LCD file and discard any changes on Kconfig and Makefile.

18.4.4 Configuring LCD Timings and the Display Interface

To support the new LCD, include the specification for the following LCD characteristics in the madglobal:mxc53_<reference board name>.c file (located at `<ltib dir>//rpm/BUILD/linux/arch/arm/mach-mx5/madglobal:mxc53_<board name>.c`):

- Display resolution
- Pixel color depth
- Refresh rate
- RGB display waveform description.
- IPU display output interface format

For the display, resolution, refresh rate, and RGB display waveform descriptions, add a new `fb_videomode struct` into the video_modes[] array based on the LCD timing and waveforms. See the CLAA-VGA entry on the following example code.

```
static struct fb_videomode video_modes[] = {
      {
       /* NTSC TV output */
       "TV-NTSC", 60, 720, 480, 74074,
       122, 15,
       18, 26,
       1, 1,
       FB_SYNC_HOR_HIGH_ACT | FB_SYNC_VERT_HIGH_ACT | FB_SYNC_EXT,
       FB_VMODE_INTERLACED,
       0,},

        ......

      {
       /* 640x480 @ 60 Hz */
       "CLAA-VGA", 60, 640, 480, 39683, 45, 114, 33, 11, 1, 1,
       FB_SYNC_CLK_LAT_FALL,
       FB_VMODE_NONINTERLACED,
       0,},
};
```

After including the new entry for the CLAA057VA01CT into the video_modes[] array, point the DISP0 configuration to this new `fb_videomode`. The link can be done by using an `mxc_fb_platform_data struct` when the frame buffer device is registered, as follows.

```
static struct mxc_fb_platform_data CLAA057VA01CT_fb_data =
{
   .interface_pix_fmt = IPU_PIX_FMT_RGB666,
   .mode_str = "CLAA-VGA",
   .mode = video_modes,
   .num_modes = ARRAY_SIZE(video_modes),
};
```

Use this new `mxc_fb_platform_data struct` to register DISP0 in the mxc_init_fb() function, as follows.

```
static int __init mxc_init_fb(void)
{
   ...
   memcpy(&fb_data[0], &CLAA057VA01CT_fb_data, sizeof(struct mxc_fb_platform_data));
   mxc_register_device(&mxc_disp_devices[0], NULL);
   ...
   return 0;
}
```

This code replaces the default WVGA panel settings with the new LCD entry.

18.4.5 Adding BSP Support for a New Boot Command to Select CLAA057VA01CT LCD

To take advantage of the kernel boot process, use the kernel boot arguments to select the new LCD. Do this by using the following steps to add extra code to the mxc53_<board name>.c file located at
`<ltib dir>//rpm/BUILD/linux/arch/arm/mach-mx5/mxc53_<board name>.c`

1. Select a new flag; this example code uses CLAA057VA01CT.
2. Create a variable to save if the CLAA057VA01CT flag has been included in the command line. In this case 0 is the initial value and it means that flag is not present.

   ```
   static int __initdata enable_CLAA057VA01CT = { 0 };
   ```
3. Use the following code to set the enable_CLAA057VA01CT variable if “CLAA057VA01CT” is included on the command line.

   ```
   static int __init CLAA057VA01CT_setup(char *__unused)
   {
      printk(KERN_INFO "CLAA057VA01CT flag\r\n");
      enable_CLAA057VA01CT = 1;
      return 1;
   }
   __setup("CLAA057VA01CT", CLAA057VA01CT_setup);
   ```
   Once the enable_CLAA057VA01CT variable is set, it is easy to distinguish which LCD should be used on DISP0 interface. The code looks like the following:

   ```
   static int __init mxc_init_fb(void)
   {
      ...

      if(enable_CLAA057VA01CT)
      {
         memcpy(&fb_data[0], &CLAA057VA01CT_fb_data, sizeof(struct mxc_fb_platform_data));
         mxc_register_device(&mxc_disp_devices[0], NULL);
         printk(KERN_INFO "DI0 driving CLAA057VA01CT\n");
      }
      else
      {
         printk(KERN_INFO "DI0 driving CLAA-WVGA\n");
      }
      ...
   ```

```
        return 0;
    }
```

4. Modify the boot command with the following code.

```
EVK U-Boot > setenv bootargs_mmc 'setenv bootargs ${bootargs} root=/dev/nfs ip=dhcp
CLAA057VA01CT nfsroot=${serverip}:${nfsroot},v3,tcp'

EVK U-Boot > run bootcmd_mmc
```

18.5 i.MX53 Display Interface Helpful Information

TFT panels are handled by the i.MX53 IPU legacy interface, which enables the display port to use different types of display interfaces that conform to the following features and restrictions.

- The IPU supports four displays in total.
- The display port has two DI interfaces.
- Each interface can handle up to three displays.
- Each DI can handle up to two asynchronous interfaces (e.g. Smart LCD, Graphic accelerator).
- Only one asynchronous interface per DI can be serial.
- Each DI can handle one synchronous interface (e.g. TV, dumb LCD).
- Asynchronous displays that are accessed in synchronous mode are considered a synchronous interface (dual_mode).

Each DI in the i.MX53 can handle one synchronous (dumb display).

Figure 18-4 shows an example i.MX53 board display interface layout.

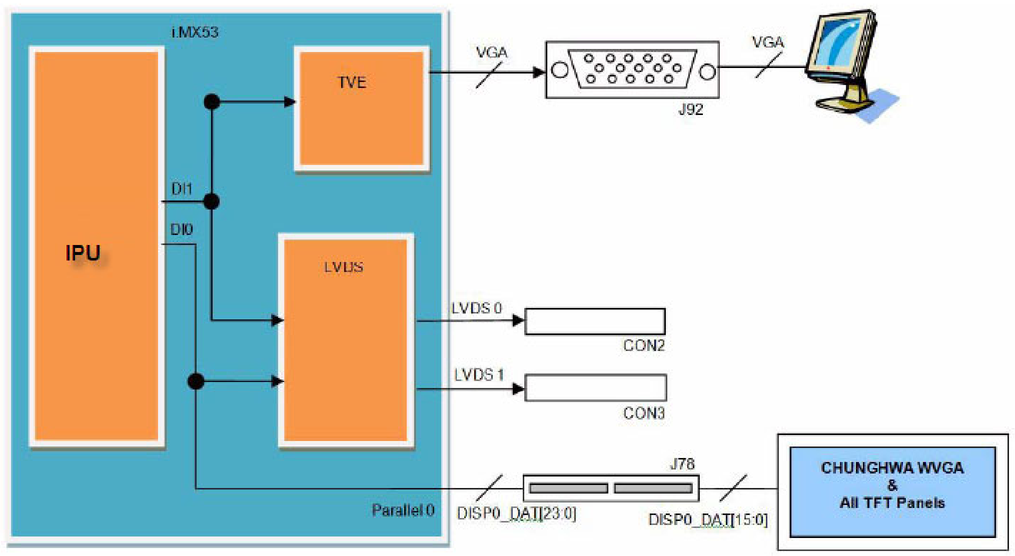


Figure 18-4. i.MX53 Board Display Interface Layout

The example board’s two display interface (DI) modules are each configured to handle one or more different kind of panels. The DI module is responsible for the timing waveforms for each signal in its display’s interface. It is composed of the following:

- 8 sets of waveform generators (PIN1–PIN8) that control signals associated with the DI’s clock, such as VSYNC and HSYNC.
- 12 sets of waveform generators (PIN11–PIN17 + 2 CS), controlling signals associated with data, such as DRDY/DE, CS, or RS

The DI also generates the display’s clock based on IPU HSP_CLK or from an external clock (PLL or pin).

The IPU provides the flexibility to select from a range of pins to use as an output for the synchronization signals. Therefore, there is no unique pin for VSYNC, HSYNC and DE. However, the i.MX51 reference boards have been assigned a specific pin for each signal, which is reflected in the schematics and BSP support.

To develop a system with a new LCD panel that does not have a driver already implemented, it is necessary to implement the new driver into the Linux's kernel and do it taking the advantage of all processor's hardware designed to the respective task, like the IPU from the i.MX53 in order to enhance the processor's performance.

For additional details about timing and TFT signals, see AN3977, AN3978, and AN4041 (available on the Freescale website).