



Reference Design

Implementing DirectFB Libraries on the *Helio View*

Version 2.0

March 20, 2014

Corporate HQ & Design Center
380 Stevens Ave. Suite 206
Solana Beach, CA 92075
<http://www.macnica-na.com>

About Macnica Americas

Macnica Americas is a franchised semiconductor distributor for multiple, high-tech suppliers within North America. Our business model emphasizes unsurpassed technical support and knowledge versus other distribution options at no cost premium. Macnica Americas is the North American based division of Macnica Inc., a \$2.4B global leader in semiconductor distribution. We maintain a field support staff as well as centralized design & applications teams.



Optional design services are headquartered in San Diego, CA., USA and offer partial or full turnkey design of FPGAs, power distribution networks, and full PCB design. Our expertise includes all aspects of high speed communications protocols and networking, video broadcast, signal processing, and storage applications. Macnica's specialty is high density, high speed complex FPGA designs utilizing multiple IP cores with fast time to market requirements.

Macnica can help you deliver a winning project with the unique combination of technical support, custom IP, and design services. Setup a meeting today!

<http://www.macnica-na.com>

License and Terms of Use

This lab with its associated source code and support files, are being provided on an "as-is" basis and as an accommodation. Therefore all warranties, representations or guarantees of any kind (whether express, implied or statutory) including, without limitation, warranties of merchantability, non-infringement, or fitness for a particular purpose, are specifically disclaimed.

This source code may only be used in an Altera programmable logic device and may not be distributed without permission from Macnica Americas, Inc. It is provided free of royalties or fees of any kind.

Table of Contents

About Macnica Americas	2
License and Terms of Use	2
1 Reference Design Overview	4
1.1 Introduction and Goals	4
1.2 Hardware and Software Requirements	4
1.2.1 Development environment.....	4
1.1.1 Evaluation board	4
1.1.2 SoC FPGA design	4
1.1.3 MAI Linux BSP	4
1.3 Assistance.....	4
1.4 Reference Design Framework.....	5
1.4.1 Hardware.....	5
1.4.2 Software	6
1.4.3 MAI Linux BSP Package	6
2 Reference Design Generation Instructions	8
2.1 MAI LCD Controller FPGA Implementation.....	8
2.1.1 License MAI LCD Controller IP.....	8
2.1.2 Generate Qsys system	8
2.1.3 Compile hardware reference design.....	8
2.2 MAI Linux BSP Implementation	9
2.2.1 Install and build Altera Linux.....	9
2.2.2 Build MAI LCD Controller drivers for Linux	13
2.2.3 Build DirectFB library and examples	17
2.3 Creating microSD Card from the components.....	22
2.3.1 Create partitions on the microSD card	22
2.4 Run DirectFB Demonstration	25
2.4.1 Install microSD card	25
2.4.2 Program FPGA with reference design.....	25
2.4.3 Reset system	25
3 Notes.....	27

1 Reference Design Overview

1.1 Introduction and Goals

This reference guide is designed as a self-paced-learning tool for understanding the fundamentals of graphics and video on the Macnica Helio View LCD module utilizing the Altera SoC. This guide simply instructs the user on the steps necessary to implement both the hardware (FPGA) and software (Linux and DirectFB libraries) components to achieve a working reference design. It is highly recommended persons attend additional training, such as that offered by Altera directly, for more detailed education on this rather complex flow and device family.

1.2 Hardware and Software Requirements

1.2.1 Development environment

- ✧ Linux host OS (An Oracle VirtualBox running CentOS 6.4 was used in the creation of this reference design)
- ✧ Altera Quartus II v13.1
- ✧ Altera SoC EDS v13.1

1.1.1 Evaluation board

- ✧ Macnica Helio SoC Evaluation kit with 2 micro-USB cables
- ✧ Macnica Helio View LCD Module

1.1.2 SoC FPGA design

- ✧ In order to use the MAI Linux BSP mentioned below, the SoC device on the Helio board must be configured so that the correct hardware features are enabled to support the Helio View LCD as well as other necessary features.

1.1.3 MAI Linux BSP

All necessary software components are included in the MAI Linux BSP package referenced by this manual.

- ✧ Target Embedded OS - Altera Linux on SoC FPGA
 - Poky 8.0 (Yocto Project 1.3 Reference Distro)
 - Linux source & tool chain: linux-socfpga-13.02-src.bsx
 - Helio preloader: helio_1gb_cl7_spl.bin
- ✧ MAI LCD Controller drivers
- ✧ DirectFB libraries and examples
- ✧ Refer to section 0 below.

1.3 Assistance

A dedicated e-mail account has been setup to receive support requests for the vWorkshop series. Please identify the course (in this case DirectFB Reference Design) in addition to details on the question.
workshophelp@macnica.com

1.4 Reference Design Framework

- ☐ *Implement FPGA hardware design with MAI LCD controller*
- ☐ *Build Yocto Altera distribution of Linux*
- ☐ *Install Linux driver for MAI LCD controller*
- ☐ *Build Yocto Helio Linux (based on Altera Linux)*
- ☐ *Build & Install DirectFB libraries and examples*
- ☐ *Create bootable microSD flash image*
- ☐ *Execute DirectFB examples on Helio View*

1.4.1 Hardware

The figures below represent the basic hardware setup required for the MAI Linux BSP. The Helio View communicates with the Altera SoC over the High Speed Mezzanine Card (HSMC) interface and the Linux kernel utilizes the RS232 connection (57600, 8, n, 1, n) for local communications with the user and the USB Blaster interface is used to configure the SoC device.

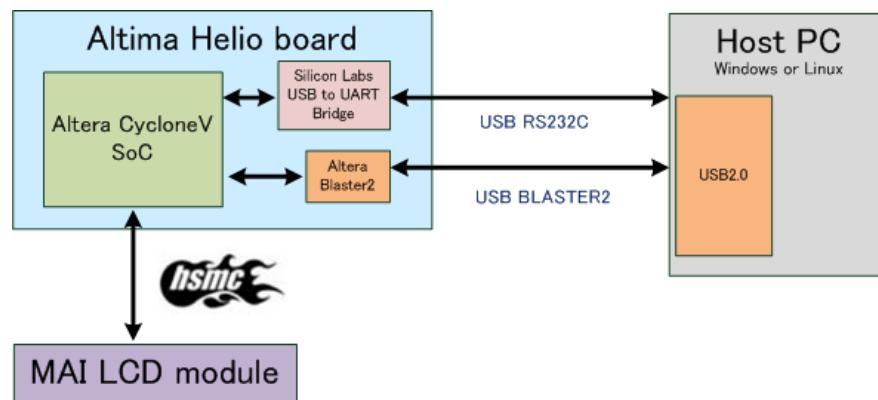


Figure 1-1 System Configuration

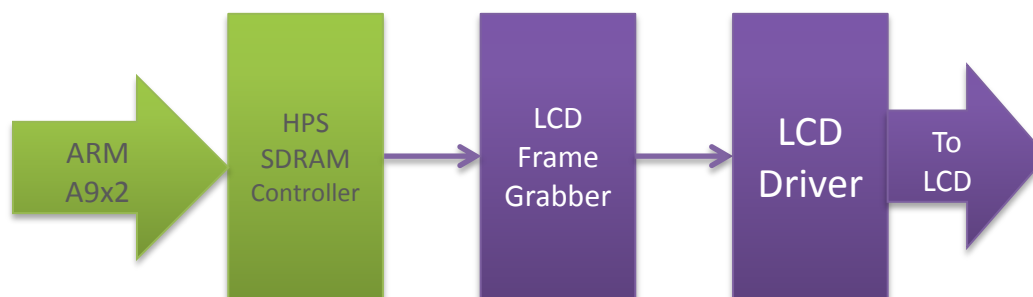


Figure 1-2 FPGA Configuration

1.4.2 Software

The figure below represents the basic software setup required for the MAI Linux BSP. The MAI LCD controller drivers are incorporated into the Linux kernel image and are loaded at kernel load time. The DirectFB examples are applications that utilize the DirectFB libraries that are included as part of the root file system.

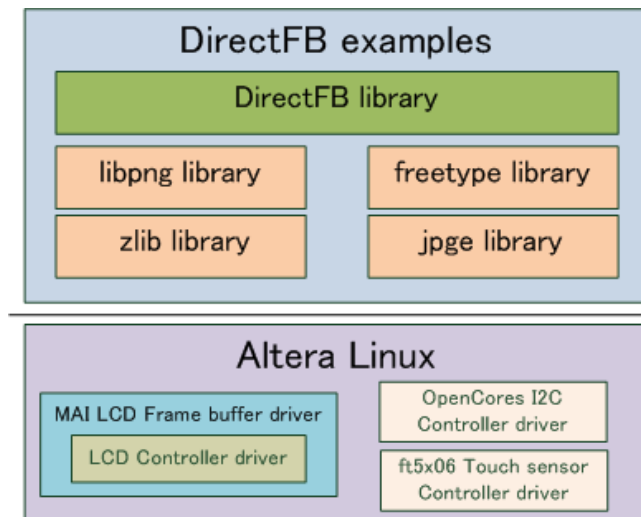


Figure 1-3 Software Configuration

1.4.3 MAI Linux BSP Package

The MAI_Linux_BSP_Package.tar.gz tar-ball referenced in this guide contains all necessary software components to configure and build the embedded Linux system that utilizes the Helio View touch LCD module.

- Altera Linux BSP – All source code & configurations for tool chain and Linux kernel build.
- Helio Drivers – All source code and scripts to build Helio View Linux drivers.
- DirectFB – All source code and scripts to build DirectFB thin libraries for Helio View applications.

The main installed directories as seen in the figure **Error! Reference source not found.** below are:

```
[MAI Linux BSP package]
|-[Update_SDcard.sh]           : Update micro SD card script
|-[Auto_login]                 : Auto login init file directory
|  +-[inittab]                  : Auto login init file
|-[Altera_linux]               : Altera BSP package directory
|  |-[linux-socfpga-13.02-src.bsx] : Altera BSP package
|  |-[helio_1gb_c17_spl.bin]      : Helio board SPL image
|  |-[Install_altera_yocto_linux.sh] : Altera BSP install script
|  +-[Build_altera_yocto_linux.sh] : Altera BSP build script
|-[Helio_drivers]              : Helio Board Linux driver directory
|  |-[Drivers]                  : Drivers directory
|  |  |-[MAI_lcd]                : MAI LCD board driver directory
|  |  |  +-[mai_fb.c]             : MAI LCD board frame buffer driver source file
|  |  |  |-[NEEK_lcd]            : NEEK LCD board driver directory
|  |  |  |  +-[neek_fb.c]         : NEEK LCD board frame buffer driver source file
|  |  |  |-[alt_vip_reg.h]        : Altera VIP suite register head file
|  |  |  |-[edt-ft5x06.patch]     : Patch file of touch screen driver
|  |  |  |-[i2c-ocores.patch]    : Patch file of OpenCores I2C driver
|  |  |  |-[socfpga_cyclone5.patch] : Patch file of Linux device tree source file
|  |  |  |-[Kconfig.patch]       : Patch file of video driver Kconfig file
|  |  |  +-[Makefile.patch]      : Patch file of video driver Makefile
|  |  +-[Helio_driver_build.sh]  : Drivers build script
|  +-[DirectFB]                 : DirectFB package directory
|  |-[zlib-1.2.3.tar.bz2]        : Zlib library package
|  |-[libpng-1.2.38.tar.bz2]     : Libpng library package
|  |-[jpegsrc.v7.tar.gz]        : jpeg library package
|  |-[freetype-2.3.5.tar.bz2]    : freetype library package
|  |-[DirectFB-1.4.11.tar.gz]    : DirectFB library package
|  |-[DirectFB-examples-1.2.0.tar.gz] : DirectFB examples package
|  |-[add_files]                : Add files directory
|  |  |-[data]                   : Data directory of demo application (Omit details)
|  |  |-[demo]                   : Script directory of demo application (Omit
details)
|  |  +-[src]                     : Source code directory of demo application (Omit
details)
|  |  |-[include.patch]          : Patch file of DirectFB head file
|  |  |-[sources.patch]          : Patch file of DirectFB source file
|  |  |-[Buid_dfb.sh]            : DirectFB build script
|  |  +-[Clean_dfb.sh]           : DirectFB build clean script
```

2 Reference Design Generation Instructions

Download compressed file that contains the Quartus design for the FPGA and the Linux BSP from the following link and unzip the reference design to a location of your choice: [SoC 8 Lab.zip](#)

2.1 MAI LCD Controller FPGA Implementation

A prebuilt FPGA image is provided if you do not wish to build it from source as directed below. It can be located in the solutions folder of the reference design downloaded from section 2 above.

- ☐ Use the Quartus Programmer to configure the FPGA with the MAI_LCD_CNTL.sof file.

2.1.1 License MAI LCD Controller IP

The MAI LCD controller is free but the source code is encrypted. Included in the reference design is the license file necessary to complete the FPGA build.

- ☐ Launch Quartus-II 13.0.1 SP1, select Tools -> License Setup. In the License File text box, point to the **free_license.dat** file that is included with the MAI LCD controller reference design in the **~/MAI_LCD_CNTL/ip/lcd_cntrl_qsys_comp** folder or append the details contained in the license file to your existing license file.
 - If you would like the unencrypted source for the MAI LCD controller please contact Macnica for details: [Macnica Americas](#)

2.1.2 Generate Qsys system

- ☐ Open **./MAI_LCD_CNTL_ref_design/MAI_LCD_CNTL.qpf**
- ☐ Select **Tools -> QSYS** and then **Open -> ./MAI_LCD_CNTL_ref_design/hps_top.qsys**
- ☐ Select the **Generation** tab then click **Generate**. Wait for completion message and then close QSYS

2.1.3 Compile hardware reference design

- ☐ In Quartus-II, select **Processing -> Start Compilation**. Wait for completion message.
- ☐ Use the Quartus Programmer to configure the FPGA with the MAI_LCD_CNTL.sof file.

2.2 MAI Linux BSP Implementation

A prebuilt microSD image with the Linux kernel and DirectFB Libraries and examples is provided if you do not wish to build it from source as directed below. It can be located in the solutions folder of the reference design downloaded from section 2 above.

- ❑ Use Win32DiskImager to image your microSD card from a Windows machine.

Software build outline

This section describes the steps required to build the complete MAI Linux BSP. There are four basic steps:

1. Build Altera Linux
2. Build Helio View LCD Drivers
3. Build DirectFB libraries and examples
4. Create microSD Card from the components

Each of the above has several detailed steps that are necessary to complete the build. The sections below instruct the user to launch several shell scripts that encompass these steps. It is recommended that the user explore the scripts to get an understanding of the exact flow that is required.

Download reference design

Download compressed file that contains MAI Linux BSP package from the following link: [MAI Linux BSP Package](#)

2.2.1 Install and build Altera Linux

- *Extract MAI Linux BSP package.*

Run the following command to extract the complete MAI Linux BSP software package.

```
$ cd ~/
$ tar xvf MAI_Linux_BSP_Package.tar.gz
```

- *Change current directory to Altera Linux BSP package directory.*

```
$ cd MAI_Linux_BSP_Package/Altera_linux
```

You can use the "ls" or "ll" command to confirm "Altera_linux" directory.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ cd Altera_linux/
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/Altera_linux$ ll
total 835920
drwxr-xr-x 2 macnica macnica    4096 Aug 22 14:19 ./
drwxr-xr-x 6 macnica macnica    4096 Aug 22 14:19 ../
-rwxr--r-- 1 macnica macnica    1893 Aug 22 14:19 Build_altera_yocto_linux.sh*
-rwxr--r-- 1 macnica macnica  262144 Aug 22 14:19 helio_1gb_cl7_spl.bin*
-rwxr--r-- 1 macnica macnica    1548 Aug 22 14:19 Install_altera_yocto_linux.sh*
-rwxr--r-- 1 macnica macnica 855697158 Aug 22 14:19 linux-socfpga-13.02-src.bsx*
```

Figure 2-1 Altera BSP Package directory

- *Install Altera BSP*

```
$ ./Install_altera_yocto_linux.sh
```

Confirm the install path. You can modify the above script as necessary.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/Altera_linux$ ./Install_altera_yocto_linux.sh
#####
### This script is setup Altera linux BSP and yocto project.          ###
#####
- Altera Linux BSP install path is default : /opt/altera-linux -
Do you wish to install Altera Linux BSP? (Y/y or N/n)
```

Figure 2-2 Altera BSP install script

- *Install Altera Linux BSP*

Sudo privileges are required to install the Altera Linux BSP.

```
- Directfb install path is default : /opt/altera-linux -
Do you wish to install Altera linux BSP? (Y/y or N/n)y
[sudo] password for macnica:
```

Figure 2-3 Sudo password

Accept the licensing terms.

```
Do you wish to install Altera Linux BSP? (Y/y or N/n)y
SoC-FPGA Source Code Release 13.02

Self-Extracting Installer

This release provides the open source software packages as listed in the file manifest.txt.
The licenses for each of the packages are provided within the respective package.
Press Enter to continue
```

Figure 2-4 License Confirmation

Verify successful BSP installation.

```
./sources/linux-altera-3.7.tgz
Installing Linaro toolchain to /opt/altera-linux/linaro

Now run this command (NOT as root) to install Yocto:
/opt/altera-linux/bin/install_altera_socfpga_src.sh ~/yocto

Install Altera linux BSP end.
=====
```

Figure 2-5 Successful Altera Linux BSP installation

- *Install Yocto recipe of Altera Linux*

Confirm the install path. You can modify the above script as necessary.

```
Do you wish to install Altera's BSP with Yocto recipe? (Y/y or N/n)y
Install Altera's BSP with Yocto recipe.
Installing Yocto to /home/macnica/yocto
Complete

To build Linux using Yocto, type the following (assuming you want to build in /home/macnica/
yocto/build):
    source /home/macnica/yocto/altera-init /home/macnica/yocto/build
    bitbake virtual/kernel virtual/bootloader altera-image

Install Altera's BSP with Yocto recipe end.
=====
```

Figure 2-6 Successful Yocto recipe installation

- **Build Altera Linux BSP**

```
$ ./Build_altera_yocto_linux.sh
```

Note: You may receive a prompt to reconfigure dash. You can safely ignore this.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/Altera_linux$ ./Build_altera_yoc
to_linux.sh
=====
### This script is build yocto project of Altera linux. ###
=====
In the reconfigure dash menu please select [No]!
Please press [Enter] go to reconfigure dash menu.
```

Figure 2-7 Yocto project build script

- **Build u-boot**

The next step in the script will build u-boot from the reference Yocto distribution.

```
A new build directory has been created with a conf subdirectory. This
contains the default configuration for building for altera hardware.
You can modify the conf/local.conf file to adjust various settings, such
as building with an initramfs.

### Shell environment set up for builds. ###

You can now run 'bitbake <target>'

Common targets are:
    bitbake virtual/kernel
    bitbake virtual/bootloader
    bitbake altera-image
/home/macnica/yocto/build
Build Altera's BSP package with Yocto.
Do you wish to build u-boot? (Y/y or N/n)
```

Figure 2-8. Build u-boot

```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 529 tasks of which 529 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
Build u-boot end.
=====
```

Figure 2-9. Successful u-boot build completion

- **Build Altera Linux kernel**

The next step in the script will build the Altera Linux kernel from the reference Yocto distribution. This kernel is referenced as “Altera Linux kernel.”

```
Do you wish to build Altera Linux kernel? (Y/y or N/n)y
Build Altera Linux kernel.
WARNING: Host distribution "Ubuntu 12.04.2 LTS" has not been validated with this version of the build system; you may possibly experience unexpected failures. It is recommended that you use a tested distribution.
Loading cache: 100% |#####| ETA: 00:00:00
Loaded 1162 entries from dependency cache.
Parsing recipes: 0% |#####| ETA: --:--:--
```

Figure 2-10. Build Altera Linux kernel

```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 588 tasks of which 585 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
Build Altera Linux kernel end.
=====
```

Figure 2-11. Successful Altera Linux kernel build

- **Build root file system**

The next step in the script will build a root file system from the reference Yocto distribution.

```
Do you wish to build Altera Linux root filesystem? (Y/y or N/n)y
Build root filesystem.
WARNING: Host distribution "Ubuntu 12.04.2 LTS" has not been validated with this version of the build system; you may possibly experience unexpected failures. It is recommended that you use a tested distribution.
Loading cache: 100% |#####| ETA: 00:00:00
Loaded 1162 entries from dependency cache.
```

Figure 2-12. Build Altera Linux root file system

```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 1956 tasks of which 1948 didn't need to be rerun and
all succeeded.

Summary: There was 1 WARNING message shown.
Build Altera Linux root filesysteml end.
=====
```

Figure 2-13. Successful Altera Linux root file system build

At this point, a complete Altera Linux bootloader, kernel and root file system has been built for a default Altera Linux system.

2.2.2 Build MAI LCD Controller drivers for Linux

- *Change current directory to Helio driver directory.*

```
$ cd ~/MAI_Linux_BSP_Package/Helio_drivers
```

You can use the "ls" or "ll" command confirm "Helio_drivers" directory.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ cd Helio_drivers/
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/Helio_drivers$ ll
total 16
drwxr-xr-x 3 macnica macnica 4096 Aug 22 14:08 ./
drwxr-xr-x 6 macnica macnica 4096 Aug 22 13:53 ../
drwxr-xr-x 4 macnica macnica 4096 Aug 22 10:05 Drivers/
-rwxr--r-- 1 macnica macnica 3815 Aug 22 10:05 Helio_driver_build.sh*
```

Figure 2-14. Helio View drivers package directory

- *Run the custom drivers build script.*

```
$ ./Helio_driver_build.sh
```

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ ./Helio_driver_build.sh
=====
### This script is build the Helio linux kernel to custom. ###
### (Add some LCD panel drivers to Helio linux kernel) ###
=====
- Altera Linux BSP install path is default : /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi/linux-altera-3.7-r1/linux-altera-3.7 -
Do you wish to copy and patched the drivers in Linux kernel? (Y/y or N/n)
```

Figure 2-15. Helio View LCD custom kernel build script

Patch the MAI LCD Controller drivers into Altera Linux kernel

The first step of this script patches the Helio View drivers into the Altera Linux kernel built in the previous section.

```
Do you wish to copy and patched the drivers in Linux kernel? (Y/y or N/n)y
Copy MAI LCD drivers mai_fb.c to Linux kernel.
Copy NEEK LCD drivers to neek_fb.c to Linux kernel.
Copy Altera Vip suite register head file alt_vip_reg.h to Linux kernel.
Patched the open cores i2c driver.
patching file /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi
/linux-altera-3.7-r1/linux-altera-3.7/drivers/i2c/busses/i2c-ocores.c
Patched the edt-ft5x06 touch screen driver.
patching file /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi
/linux-altera-3.7-r1/linux-altera-3.7/drivers/input/touchscreen/edt-ft5x06.c
Patched the video driver Kconfig file.
patching file /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi
/linux-altera-3.7-r1/linux-altera-3.7/drivers/video/Kconfig
Patched the video driver Makefile file.
patching file /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi
/linux-altera-3.7-r1/linux-altera-3.7/drivers/video/Makefile
Patched the Helio kernel device tree source file.
patching file /home/macnica/yocto/build/tmp/work/socfpga_cyclone5-poky-linux-gnueabi
/linux-altera-3.7-r1/linux-altera-3.7/arch/arm/boot/dts/socfpga_cyclone5.dts
Copy and patched the drivers to Linux kernel is end.
=====
```

Figure 2-16. Patch the drivers to Linux kernel

- **Kernel menuconfig**

The built-in kernel configuration tool, menuconfig, must be run to incorporate the MAI LCD Controller drivers into a new Helio Linux kernel. There are issues with the default display format of the menuconfig screen for CentOS. You need to modify the type of terminal that the Yocto recipe calls.

Comment (#) out the line “OE_TERMINAL_EXPORTS += "HOST_EXTRACFLAGS HOSTLDFLAGS HOST_LOADLIBES" in the ~/yocto/meta/classes/cml1.bbclass file.

```
Do you wish to set Helio Linux kernel menuconfig? (Y/y or N/n)y

### Shell environment set up for builds. ###

You can now run 'bitbake <target>'

Common targets are:
    bitbake virtual/kernel
    bitbake virtual/bootloader
    bitbake altera-image
/home/macnica/yocto/build
WARNING: Host distribution "Ubuntu 12.04.2 LTS" has not been validated with this ver
sion of the build system; you may possibly experience unexpected failures. It is rec
ommended that you use a tested distribution.
```

Figure 2-17. Launching “menuconfig”

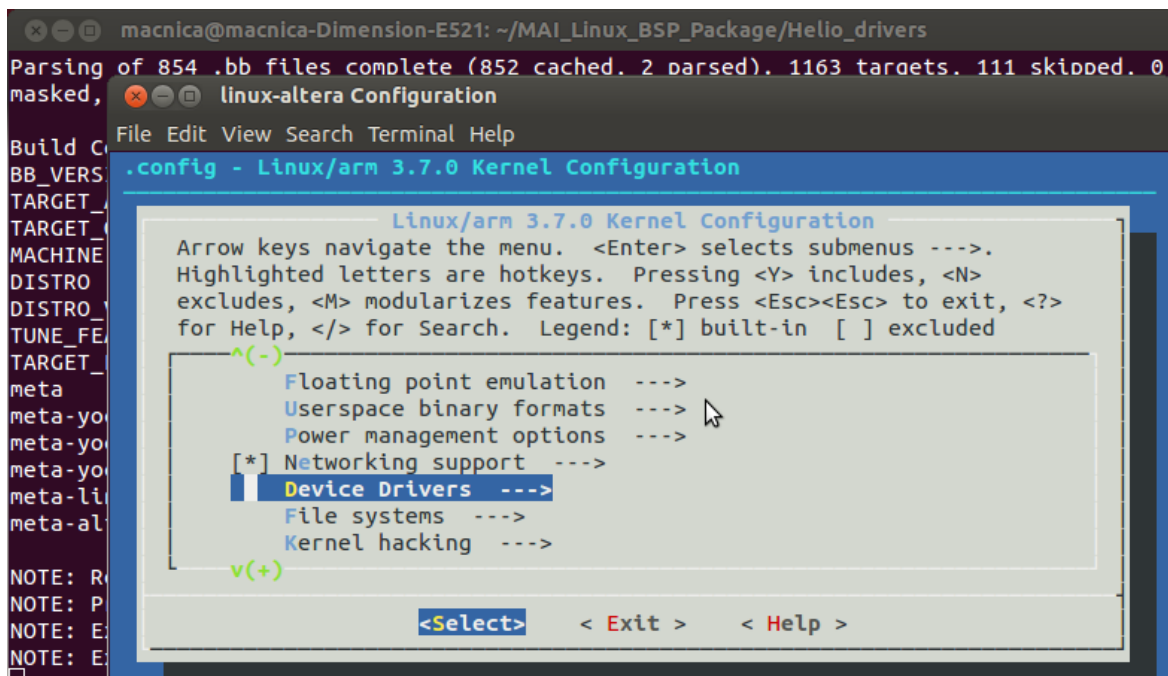


Figure 2-18. “menuconfig” screen

Use the arrow keys to navigate and space bar to select the following options in menuconfig.

```
Device Driver --->
  Input device support --->
    (800) Horizontal screen resolution
    (480) Vertical screen resolution
  [*] Touchscreen --->
    <*> EDT FocalTech FT5x06 I2C Touchscreen support
  <*> I2C support --->
    <*> I2C bus multiplexing support
    Multiplexer I2C Chip support --->
      <*> GPIO-based I2C multiplexer
      <*> NXP PCA9541 I2C Master Selector
      <*> philips PCA954x I2C Mux/switches
    I2C Hardware Bus support --->
      <*> Synopsys DesignWare Platform
      <*> OpenCores I2C Controller
  Graphics support --->
    <*> Support for frame buffer devices --->
    <*> MAI LCD panel frame buffer support
```

- *After setting the kernel menuconfig, exit menuconfig.*


```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 111 tasks of which 110 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
Set Helio Linux kernel menuconfig end.
=====
```

Figure 2-19. “menuconfig” exit

- ***Build new Linux kernel with MAI LCD Controller drivers***

The next step in the script will build a new Linux kernel from the modified reference Yocto distribution. This new kernel is referenced as “Helio Linux kernel.”

```
Do you wish to build Helio Linux kernel? (Y/y or N/n)y
WARNING: Host distribution "Ubuntu 12.04.2 LTS" has not been validated with this version of the build system; you may possibly experience unexpected failures. It is recommended that you use a tested distribution.
Loading cache: 100% |#####| ETA: 00:00:00
Loaded 1162 entries from dependency cache.
Parsing recipes: 100% |#####| Time: 00:00:00
Parsing of 854 .bb files complete (852 cached, 2 parsed). 1163 targets, 111 skipped, 0 masked, 0 errors.
```

Figure 2-20. Build Helio Linux kernel

```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 588 tasks of which 579 didn't need to be rerun and all succeeded.

Summary: There was 1 WARNING message shown.
Build Helio Linux kernel end.
=====
```

Figure 2-21. Successful Helio Linux kernel build

- ***Build root file system with MAI LCD Controller drivers***

The next step in the script will build a root file system from the Helio modified reference Yocto distribution.

```
Do you wish to build Helio Linux root filesystem? (Y/y or N/n)y
Build Helio Linux root filesystem.
WARNING: Host distribution "Ubuntu 12.04.2 LTS" has not been validated with this version of the build system; you may possibly experience unexpected failures. It is recommended that you use a tested distribution.
```

Figure 2-22. Build Helio Linux root file system


```
NOTE: Resolving any missing task queue dependencies
NOTE: Preparing runqueue
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 1956 tasks of which 1948 didn't need to be rerun and
all succeeded.

Summary: There was 1 WARNING message shown.
Build Helio Linux root filesystem end.
=====
```

Figure 2-23. Successful Helio Linux root file system build

At this point, a complete Helio Linux kernel and root file system has been built for the MAI Linux BSP system.

2.2.3 Build DirectFB library and examples

- *Change current directory to DirectFB package directory.*

Run the following command to change to the DirectFB working directory.

```
cd ~/MAI_Linux_BSP_Package/DirectFB
```

You can use the "ls" or "ll" command confirm "DirectFB" directory.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ cd DirectFB/
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/DirectFB$ ll
total 7988
drwxr-xr-x 3 macnica macnica  4096 Aug 22 10:05 ./
drwxr-xr-x 6 macnica macnica  4096 Aug 22 10:05 ../
drwxr-xr-x 5 macnica macnica  4096 Aug 22 10:05 add_files/
-rwxr--r-- 1 macnica macnica 10832 Aug 22 10:05 Build_dfb.sh*
-rwxr--r-- 1 macnica macnica   447 Aug 22 10:05 Clean_dfb.sh*
-rwxr--r-- 1 macnica macnica 3110010 Aug 22 10:05 DirectFB-1.4.11.tar.gz*
-rwxr--r-- 1 macnica macnica 1693501 Aug 22 10:05 DirectFB-examples-1.2.0.tar.gz*
-rwxr--r-- 1 macnica macnica 1279861 Aug 22 10:05 freetype-2.3.5.tar.bz2*
-rwxr--r-- 1 macnica macnica   2774 Aug 22 10:05 include.patch*
-rwxr--r-- 1 macnica macnica  960379 Aug 22 10:05 jpegsrc.v7.tar.gz*
-rwxr--r-- 1 macnica macnica  662908 Aug 22 10:05 libpng-1.2.38.tar.bz2*
-rwxr--r-- 1 macnica macnica    961 Aug 22 10:05 sources.patch*
-rwxr--r-- 1 macnica macnica  425209 Aug 22 10:05 zlib-1.2.3.tar.bz2*
```

Figure 2-24. DirectFB Package directory

- *Run the DirectFB auto build script.*

```
$ ./Build_dfb.sh
```

Confirm the install path. You can modify the above script as necessary.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package/DirectFB$ ./Bulid_dfb.sh
#####
### This script is build the DirectFB library and examples.          ###
### DirectFB                                                         Ver.1.4.11                             ###
### DirectFB examples                                               Ver.1.2.0                               ###
#####
- Directfb install path is default : /usr -
```

Figure 2-25. Execution of DirectFB automated build script

- *Create DirectFB library install directory.*

```
- Directfb install path is default : /usr -
Do you wish to Create DirectFB library install directory? (Y/y or N/n)y
Create DirectFB library install directory /usr.
sudo mkdir /usr/dfb
sudo chmod -R 777 /usr/dfb
Create DirectFB library install directory /usr end.
=====
```

Figure 2-26. Create DirectFB library install directory

- *Set environment variables*

```
Do you wish to set environment variables? (Y/y or N/n)y
Set environment variables.
export PATH=/usr/lib/lightdm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/b
in:/sbin:/bin:/usr/games:/opt/altera-linux/linaro/gcc-linaro-arm-linux-gnueabi-4.7
-2012.11-20121123_linux/bin
Set environment variables end.
=====
```

Figure 2-27. Set environment variables for DirectFB build

- *Extract source files*

```
Do you wish to extract library and examples packages? (Y/y or N/n)y
.
Extract library and examples packages end.
=====
Add image files to DirectFB examples directory.
Add source files to DirectFB examples directory.
Add image files and source files to DirectFB examples directory end.
=====
```

Figure 2-28. Extract DirectFB library and examples package

- *Build and install zlib library.*

```
Build and install zlib library.
Do you wish to build and install zlib library? (Y/y or N/n)y
Configure zlib library.
CC=arm-linux-gnueabi-gcc ./configure --shared --prefix=/usr/dfb
Checking for shared library support...
Building shared library libz.so.1.2.3 with arm-linux-gnueabi-gcc.
Checking for unistd.h... Yes.
Checking whether to use vs[n]printf() or s[n]printf()... using vs[n]printf()
Checking for vsnprintf() in stdio.h... Yes.
```

Figure 2-29. Build and install zlib library

```
cp zlib.3 /usr/dfb/share/man/man3
chmod 644 /usr/dfb/share/man/man3/zlib.3
Build and install zlib library end.
=====
```

Figure 2-30. Successful build and install of zlib library

Build and install jpeg library.

```
Build and install jpeg library.
Do you wish to Build and install jpeg library? (Y/y or N/n)y
Configure jpeg library.
CC=arm-linux-gnueabi-gcc ./configure --host=arm-linux-gnueabi-gcc --prefix=/usr/dfb
--enable-shared
configure: WARNING: If you wanted to set the --build type, don't use --host.
If a cross compiler is detected then cross compile mode will be used.
checking build system type... x86_64-unknown-linux-gnu
```

Figure 2-31. Build and install jpeg library

```
/usr/bin/install -c -m 644 jconfig.h /usr/dfb/include/jconfig.h
test -z "/usr/dfb/include" || /bin/mkdir -p "/usr/dfb/include"
/usr/bin/install -c -m 644 jerror.h jmorecfg.h jpeglib.h '/usr/dfb/include'
test -z "/usr/dfb/share/man/man1" || /bin/mkdir -p "/usr/dfb/share/man/man1"
/usr/bin/install -c -m 644 cjpeg.1 djpeg.1 jpegtran.1 rdjpgcom.1 wrjpgcom.1 '/usr/dfb/share/man/man1'
make[1]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/jpeg-7'
Build and install jpeg library end.
=====
```

Figure 2-32. Successful build and install of jpeg library

- *Build and install libpng library.*

```
Build and install libpng library.
Do you wish to build and install libpng library? (Y/y or N/n)y
Configure libpng library.
CC=arm-linux-gnueabi-gcc ./configure --host=arm-linux-gnueabi-gcc --prefix=/usr/dfb
--enable-shared
configure: WARNING: If you wanted to set the --build type, don't use --host.
If a cross compiler is detected then cross compile mode will be used.
checking for a BSD-compatible install... /usr/bin/install -c
```

Figure 2-33. Build and install libpng library

```
cd /usr/dfb/lib/pkgconfig; ln -s libpng12.pc libpng.pc
make[2]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/libpng-1.2
.38'
make[1]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/libpng-1.2
.38'
Build and install libpng library end.
=====
```

Figure 2-34. Successful build and install of libpng library

- *Build and install freetype library.*

```
Build and install freetype library.
Do you wish to build and install freetype library? (Y/y or N/n)y
Configure freetype library.
CC=arm-linux-gnueabi-gcc ./configure --host=arm-linux-gnueabi-gcc --prefix=/usr/dfb
--enable-shared

FreeType build system -- automatic system detection
```

Figure 2-35. Build and install freetype library

```
/usr/bin/install -c -m 644 ./builds/unix/freetype2.m4 \
    /usr/dfb/share/aclocal/freetype2.m4
/usr/bin/install -c -m 644 ./builds/unix/freetype2.pc \
    /usr/dfb/lib/pkgconfig/freetype2.pc
Build and install freetype library end.
=====
```

Figure 2-36. Successful build and install of freetype library

- *Build and install DirectFB library.*

```
Build and install DirectFB library.
Do you wish to build and install DirectFB library? (Y/y or N/n)y
Pating some source file
patching file interfaces/IDirectFBFont/ldirectfbfont_ft2.c
patching file tools/mkdgiff.c
Pating some include hedar file
```

Figure 2-37. Build and install DirectFB library

```
test -z "/usr/dfb/lib/pkgconfig" || /bin/mkdir -p "/usr/dfb/lib/pkgconfig"
/usr/bin/install -c -m 644 directfb.pc directfb-internal.pc '/usr/dfb/lib/pkgconfig'
make[2]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/DirectFB-1
.4.11'
make[1]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/DirectFB-1
.4.11'
Build and install DirectFB library end.
=====
```

Figure 2-38. Successful build and install of DirectFB library

- *Build and install DirectFB examples.*

```
Build and install DirectFB examples.
Do you wish to build and install DirectFB examples? (Y/y or N/n)y
Configure DirectFB examples.
CC=arm-linux-gnueabi-gcc ./configure --host=arm-linux-gnueabi --prefix=/usr/dfb
configure: WARNING: If you wanted to set the --build type, don't use --host.
If a cross compiler is detected then cross compile mode will be used.
checking for a BSD-compatible install... /usr/bin/install -c
```

Figure 2-39. Build and install DirectFB examples

```
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/DirectFB-examples-1.2.0'
make[1]: Leaving directory `/home/macnica/MAI_LCD_Driver_Package/DirectFB/DirectFB-examples-1.2.0'
Build and install DirectFB examples end.
=====
```

Figure 2-40. Successful build and install of DirectFB examples

- *Build the Helio board push button detect program.*

```
Do you wish to build check event program for demo application? (Y/y or N/n)y
Build check event program for demo application.
```

Figure 2-41. Build check event program for demo application

```
Do you wish to build check event program for demo application? (Y/y or N/n)y
Build check event program for demo application.
Build check event program for demo application end.
=====
DirectFB and some library build and install end.
```

Figure 2-42. Successful build of demo application

2.3 Creating microSD Card from the components

This section includes how to create and partition the Helio board bootable microSD card and how to update SW on the microSD card.

2.3.1 Create partitions on the microSD card

- *Invoke fdisk*

Insert microSD card on host PC and invoke fdisk with the correct device selected.

```
$ sudo fdisk /dev/sdx
```

The sd"x" is the microSD card mounted path on host PC. Use 'sudo blkid' and locate the device that has the "vfat" and "ext3" partitions. This is your microSD card. Typically the microSD is mounted as /dev/sdb. (/dev/sda and /dev/mapper are part of the Linux VM)

Within fdisk, use the "p" command to display all existing partitions on the microSD card, and then use "d" command to delete all partitions. Use "p" command again to make sure there are no partitions left.

- *Create the Linux root file system area*

Within fdisk, create the 1GB Linux partition using the following options, entered one by one, with pressing "ENTER" key between each of them:

```
n p 2 14336 +1048576K t 83
```

- *Create the Linux kernel image and device tree binary area*

Within fdisk, create the 20MB FAT32 partition that will hold the kernel image and device tree file using the following options, entered one by one, with pressing "ENTER" key between each of them:

```
n p 1 2121728 +20480K t 1 b
```

- *Create the preloader and U-Boot area*

Within fdisk, create the 1MB custom partition that will hold the SPL preloader and U-Boot using the following options, entered one by one, with pressing "ENTER" key between each of them:

```
n p 3 2048 +1024K t 3 a2
```

- *Write changes to disk and exit fdisk by typing "w" followed by pressing "ENTER".*

```
w
```

- *Format Linux kernel image and device tree binary area*

```
$ sudo mkdosfs /dev/sdx1
```

- *Format Linux root file system area*

```
$ sudo mkfs.ext3 /dev/sdx2
```

- *Update the Linux kernel and root file system*
- *Copy the Helio board SPL image to the micro microSD card.*

```
$ cd ~/MAI_Linux_BSP_Package/Altera_linux  
$ sudo dd if=helio 1gb c17 spl.bin of=/dev/sdx3 bs=64k seek=0
```

- *Copy the U-boot image to the microSD card*

```
$ cd ~/yocto/build/tmp/deploy/images  
$ sudo dd if=u-boot-socfpga_cyclone5.img of=/dev/sdx3 bs=64K seek=4
```

- *Update the Linux root file system and kernel image file to microSD card.*
- *Mount the microSD card partition to host PC.*

```
$ sudo mkdir /sdcard_1  
$ sudo mkdir /sdcard_2  
$ sudo mount /dev/sdx1/sdcard_1  
$ sudo mount /dev/sdx2/sdcard_2
```

- *Change to the software package directory.*

```
$ cd ~/MAI_Linux_BSP_Package
```

You can use the "ls" or "ll" command confirm "Altera_linux" directory.

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ ll  
total 28  
drwxr-xr-x  6 macnica macnica 4096 Aug 22 13:53 ./  
drwxr-xr-x 39 macnica macnica 4096 Aug 22 10:11 ../  
drwxr-xr-x  2 macnica macnica 4096 Aug 22 10:05 Altera_linux/  
drwxr-xr-x  2 macnica macnica 4096 Aug 22 13:53 Auto_login/  
drwxr-xr-x  3 macnica macnica 4096 Aug 22 10:05 DirectFB/  
drwxr-xr-x  3 macnica macnica 4096 Aug 22 10:05 Helio_drivers/  
-rwxr--r--  1 macnica macnica 1984 Aug 22 10:05 Update_SDcard.sh*
```

Figure 2-43. Driver Package directory

- *Run the update microSD card script.*

```
$ ./Update_SDcard.sh
```

```
macnica@macnica-Dimension-E521:~/MAI_Linux_BSP_Package$ ./Update_SDcard.sh
=====
### This script is auto update micro-SD card for Helio board boot.      ###
=====
- SD card image files path is default      : /sdcard_1 -
- SD card root file system path is default : /sdcard_2 -
Do you wish to update the SD card? (Y/y or N/n)
```

Figure 2-44. Auto update microSD card script

```
Extract root file system to SD card.
      :
      :
Copy image files to SD card.
Update the SD card end.
=====
```

Figure 2-45. Successful completion of microSD card update

2.4 Run DirectFB Demonstration

This section describes the DirectFB demo application operation on the Helio View LCD module and associated software.

2.4.1 Install microSD card

- *Insert the microSD card into the Helio main board microSD card slot.*

2.4.2 Program FPGA with reference design

- *Start the Quartus programmer `~/quartus/bin/quartus_pgmw`*
- *Power on Helio board*
- *Click on Auto Detect and select 5CSXFC6C6ES. Click on Yes to overwrite settings.*
- *Select the 5CSXFC6C6ES device in the graphical chain. Click on Change File and select `./MAI_LCD_CNTL_ref_design/output_files/MAI_LCD_CNTL.sof`*
- *Select Program/Configure check box for the 5CSXFC6C6ES device. Click Start. Wait for Progress bar to reach 100% complete.*

2.4.3 Reset system

- *On Helio main board, press SW6, Warm Resetn (top of board on back). Wait 30 seconds. This boots Linux and starts the DirectFB demonstration application.*

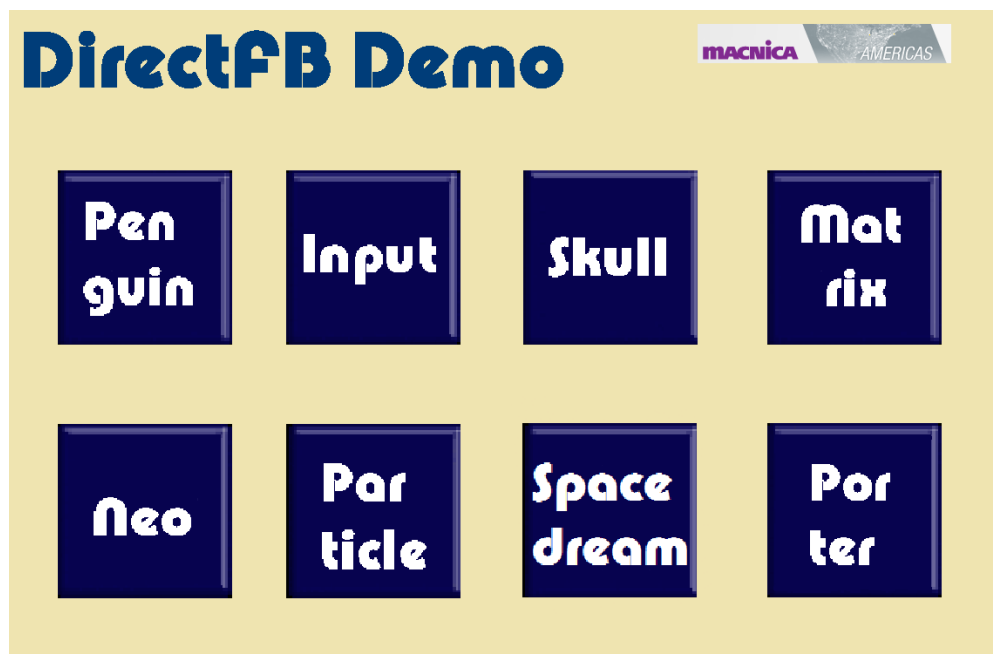


Figure 2-46. DirectFB demo application main screen

- *Start DirectFB example applications*

Tap one of the application buttons on the main screen to launch the corresponding application.

- *Stop DirectFB example application*

While running any DirectFB example application, press the SW11 push-button near the power connector on the back of the Helio main board to stop the application and return to the demo application main screen.

3 Notes

Document Revision History

Revision	Date	Comments
0.1		Initial Draft
0.2		Internal Review
1.0		Customer Release
2.0	March 20, 2014	Reformatted