Helio View LCD

Graphics Rendering on Linux

Reference Design Guide

Version 1.0

October 2013
1. 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.

2. 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/web/americas/home
3. Document Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>October 31, 2013</td>
<td>Initial Draft</td>
</tr>
</tbody>
</table>
4. Contents

1. LICENSE AND TERMS OF USE ....................................................................................................................... 2
2. ABOUT MACNICA AMERICAS ......................................................................................................................... 2
3. DOCUMENT REVISION HISTORY .................................................................................................................. 3
5. REFERENCE DESIGN OVERVIEW .................................................................................................................. 6
   5.1. Introduction and Goals ................................................................................................................................. 6
   5.2. Hardware and Software Requirements ....................................................................................................... 6
       5.2.1. Development environment .................................................................................................................. 6
       5.2.2. Evaluation board .................................................................................................................................. 6
       5.2.3. SoC FPGA design ............................................................................................................................... 6
       5.2.4. MAI Linux BSP .................................................................................................................................... 6
   5.3. Assistance .................................................................................................................................................... 6
   5.4. Reference Design Framework .................................................................................................................... 7
       5.4.1. Implement FPGA hardware design with MAI LCD controller ............................................................. 7
       5.4.2. Build Yocto Altera distribution of Linux .............................................................................................. 7
       5.4.3. Install Linux driver for MAI LCD controller .......................................................................................... 7
       5.4.4. Build Yocto Helio Linux (based on Altera Linux) .................................................................................. 7
       5.4.5. Build & Install DirectFB libraries and examples .................................................................................. 7
       5.4.6. Create bootable SD flash image ........................................................................................................... 7
       5.4.7. Execute DirectFB examples on Helio View .......................................................................................... 7
6. BASIC CONFIGURATION ................................................................................................................................... 8
   6.1. Hardware .................................................................................................................................................... 8
   6.2. Software ..................................................................................................................................................... 8
       6.2.1. MAI Linux BSP Package .................................................................................................................... 9
7. MAI LCD CONTROLLER FPGA IMPLEMENTATION ...................................................................................... 12
   7.1. Create FPGA reference design .................................................................................................................. 12
       7.1.1. Download reference design ................................................................................................................ 12
       7.1.2. License MAI LCD Controller IP .......................................................................................................... 12
       7.1.3. Open Project ....................................................................................................................................... 12
       7.1.4. Generate QSYS system .................................................................................................................... 12
       7.1.5. Compile reference design .................................................................................................................. 12
8. MAI LINUX BSP IMPLEMENTATION ............................................................................................................ 13
   8.1. Software build ............................................................................................................................................ 13
8.1.1. Download reference design ............................................................................................................. 13  
8.1.2. Install and build Altera Linux ........................................................................................................ 13  
8.1.3. Build MAI LCD Controller drivers for Linux .............................................................................. 18  
8.1.4. Build DirectFB library and examples .......................................................................................... 22  
8.2. Creating micro SD Card from the components ............................................................................... 27  
8.2.1. Create partitions on the SD card ................................................................................................ 27  
8.2.2. Update the Linux kernel and root file system .......................................................................... 28  
8.2.3. Update the Linux root file system and kernel image file to micro SD card. .............................. 28  
9. RUN DIRECTFB DEMONSTRATION .............................................................................................. 30  
9.1.1. Install micro SD card .................................................................................................................. 30  
9.1.2. Program FPGA with reference design ..................................................................................... 30  
9.1.3. Reset system ............................................................................................................................ 30  
9.1.4. Start DirectFB example applications ...................................................................................... 30  
9.1.5. Stop DirectFB example application ......................................................................................... 30
5. Reference Design Overview

5.1. 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.

5.2. Hardware and Software Requirements

5.2.1. Development environment
- Linux host OS (An Oracle VirtualBox running CentOS 6.4 was used in the creation of this reference design)
- Altera Quartus-II v13.0.1 SP1
- Altera SoC EDS v13.0.1

5.2.2. Evaluation board
- Macnica Helio SoC Evaluation kit with 2 micro-USB cables
- Macnica Helio View LCD Module

5.2.3. 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.
- Refer to section 7 below.

5.2.4. 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 8 below.

5.3. Assistance
A dedicated mail account has been setup to receive support requests for this vWorkshop series. Please identify the course (in this case SoC SW4) in addition to details on the question. WorkshopHelp@macnica.com
5.4. Reference Design Framework

Below is a simplified outline and flow of the steps taken to create this reference design.

5.4.1. Implement FPGA hardware design with MAI LCD controller

5.4.2. Build Yocto Altera distribution of Linux

5.4.3. Install Linux driver for MAI LCD controller

5.4.4. Build Yocto Helio Linux (based on Altera Linux)

5.4.5. Build & Install DirectFB libraries and examples

5.4.6. Create bootable SD flash image

5.4.7. Execute DirectFB examples on Helio View
6. Basic Configuration

6.1. Hardware

Figure 6-1 and Figure 6-2 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 6-1. System configuration

6.2. Software

Figure 6-3 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.

![Diagram showing DirectFB examples]

**Figure 6-3. Software configuration**

### 6.2.1. 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. The three main installed directories as seen in Figure 6-4 below are:

- **Altera Linux**
  - MAI LCD Frame buffer driver
  - LCD Controller driver
- **DirectFB**
  - All source code and scripts to build DirectFB thin libraries for Helio View applications.
- **Helio Drivers**
  - All source code and scripts to build Helio View Linux drivers.
[MAI Linux BSP package]
|   | -[Update_SDcard.sh] : Update micro SD card script
|   | -[Auto_login] : Auto login init file directory
|   |   | +-[initab] : Auto login init file
|   | -[Altera_linux] : Altera BSP package directory
|   |   | -[linux-socfpga-13.02-src.bsx] : Altera BSP package
|   |   | -[helio_1gb_cl7_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
Macnica Americas: Helio View Reference Design Guide

Figure 6-4 MAI Linux BSP Package configuration

<table>
<thead>
<tr>
<th>DirectFB</th>
<th>DirectFB package directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>zlib-1.2.3.tar.bz2</td>
<td>Zlib library package</td>
</tr>
<tr>
<td>libpng-1.2.38.tar.bz2</td>
<td>Libpng library package</td>
</tr>
<tr>
<td>jpegsrc.v7.tar.gz</td>
<td>Jpeg library package</td>
</tr>
<tr>
<td>freetype-2.3.5.tar.bz2</td>
<td>Freetype library package</td>
</tr>
<tr>
<td>DirectFB-1.4.11.tar.gz</td>
<td>DirectFB library package</td>
</tr>
<tr>
<td>DirectFB-examples-1.2.0.tar.gz</td>
<td>DirectFB examples package</td>
</tr>
<tr>
<td>add_files</td>
<td>Add files directory</td>
</tr>
<tr>
<td>data</td>
<td>Data directory of demo application (Omit details)</td>
</tr>
<tr>
<td>demo</td>
<td>Script directory of demo application (Omit details)</td>
</tr>
<tr>
<td>src</td>
<td>Source code directory of demo application (Omit details)</td>
</tr>
<tr>
<td>include.patch</td>
<td>Patch file of DirectFB head file</td>
</tr>
<tr>
<td>sources.patch</td>
<td>Patch file of DirectFB source file</td>
</tr>
<tr>
<td>Build_dfb.sh</td>
<td>DirectFB build script</td>
</tr>
<tr>
<td>Clean_dfb.sh</td>
<td>DirectFB build clean script</td>
</tr>
</tbody>
</table>
7. MAI LCD Controller FPGA Implementation

7.1. Create FPGA reference design

7.1.1. Download reference design

1. Download compressed file that contains Quartus design from the following link and unzip the reference design to a location of your choice.

https://macnica.box.com/shared/static/9sc7wsi9382al6v9gtle.zip

7.1.2. 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.

2. 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_ref_design/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:

http://www.macnica-na.com/web/americas/contact-us

7.1.3. Open Project

3. Open ./MAI_LCD_ref_design/MAI_LCD_ref_design.qpf

7.1.4. Generate QSYS system

4. Select Tools -> QSYS and then Open -> ./MAI_LCD_ref_design/hps_top.qsys

5. Select the Generation tab then click Generate. Wait for completion message and then close QSYS

7.1.5. Compile reference design

8. MAI Linux BSP Implementation

A prebuilt SD image with the Linux kernel and DirectFB Libraries and examples is available if you do not wish to build it from source as directed below.

1. Download the compressed SD image from the following link and uncompress the image to a location of your choice.
   https://macnica.box.com/shared/static/5f2f0hnr8zohp35tjk4w.rar
2. You can use Win32DiskImager (http://sourceforge.net/projects/win32diskimager/) to image your SD card from a Windows machine.

8.1. Software build

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 micro SD Card from the components

Each of the above has several additional detailed steps that are necessary to complete the build. The instructions below have the user 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.

8.1.1. Download reference design

Download compressed file that contains MAI Linux BSP package from the following link:
https://macnica.box.com/shared/static/q1bpxp55c96kve71ihtc.zip

8.1.2. Install and build Altera Linux

8.1.2.1. 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
```

8.1.2.2. 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.
8.1.2.3. **Install Altera BSP**

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

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

- **Install Altera Linux BSP**

  Sudo privileges are required to install the Altera Linux BSP.

```bash
$ sudo
Password:
```

Accept the licensing terms.

```bash
Do you wish to install Altera Linux BSP? (Y/y or N/n)
```

Verify successful BSP installation.

```bash
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.
```
- **Install Yocto recipe of Altera Linux**

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

  ![Figure 8-6 Successful Yocto recipe installation](image)

  **8.1.2.4. Build Altera Linux BSP**

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

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

  ![Figure 8-7 Yocto project build script](image)

- **Build u-boot**

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

  ![Figure 8-8. Build u-boot](image)
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.”

Build root file system

The next step in the script will build a root file system from the reference Yocto distribution.
At this point, a complete Altera Linux bootloader, kernel and root file system has been built for a default Altera Linux system.
8.1.3. Build MAI LCD Controller drivers for Linux

8.1.3.1. Change current directory to Helio driver directory.

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

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

![Figure 8-14. Helio View drivers package directory](image)

8.1.3.2. Run the custom drivers build script.

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

![Figure 8-15. Helio View LCD custom kernel build script](image)

- 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.

![Figure 8-16. Patch the drivers to Linux kernel](image)
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.

Figure 8-17. Launching “menuconfig”

Figure 8-18. “menuconfig” screen
Use the arrow keys to navigate and space bar to select the following options in menuconfig.

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

After setting the kernel menuconfig, exit menuconfig.

**Figure 8-19. “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.”
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.

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

8.1.4.1. 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.

![Figure 8-24. DirectFB Package directory](image)

8.1.4.2. Run the DirectFB auto build script.

```
./Build_dfb.sh
```

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

![Figure 8-25. Execution of DirectFB automated build script](image)

- Create DirectFB library install directory.

```
sudo mkdir /usr/lib
sudo chmod -R 777 /usr/lib/
Create DirectFB library install directory /usr/lib.
```

![Figure 8-26. Create DirectFB library install directory](image)
Set environment variables

Do you wish to set environment variables? (Y/y or N/n)y
Set environment variables:
export PATH=/usr/lib/lightdm/lightdm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/usr/games:/opt/altera-linux/llnaro/gcc-llnaro-arm-linux-gnueabihf-4.7.2012.11-20121123_linux/bin
Set environment variables end.

Figure 8-27. 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 8-28. Extract DirectFB library and examples package

Build and install zlib library.

Building zlib library.
Do you wish to build and install zlib library? (Y/y or N/n)y
Configure zlib library.
CC=arm-linux-gnueabihf-gcc ./configure --shared --prefix=/usr/dfb
Checking for shared library support...
Building shared library libz.so.1.2.3 with arm-linux-gnueabihf-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 8-29. 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 8-30. 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-gnueabihf-gcc ./configure --host=arm-linux-gnueabihf --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 8-31. Build and install 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-gnueabihf-gcc ./configure --host=arm-linux-gnueabihf --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 8-33. Build and install 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-gnueabihf-gcc ./configure --host=arm-linux-gnueabihf --prefix=/usr/dfb --enable-shared
FreeType build system -- automatic system detection

Figure 8-35. Build and install freetype library
Build and install freetype library.

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

Figure 8-36. 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
Paring some source file
patching file interfaces/IDirectFBFont/idirectfbfont_ft2.c
patching file tools/mkdglff.c
Pating some include header file

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 8-37. Build and install 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-gnueabihf-gcc ./configure --host=arm-linux-gnueabihf --prefix=/usr/dfb
cfgure: 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
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 8-38. Successful build and install of DirectFB library

Build and install DirectFB examples.

```
build -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 examples end.
```

Figure 8-39. Build and install DirectFB examples

Build and install DirectFB examples.

```
build -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 examples end.
```

Figure 8-40. 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 8-41. 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 8-42. Successful build of demo application
8.2. Creating micro SD Card from the components

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

8.2.1. Create partitions on the SD card

8.2.1.1. Invoke fdisk

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

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

- The sd"x" is the micro SD card mounted path on host PC.
- Within fdisk, use the "p" command to display all existing partitions on the micro SD card, and then use "d" command to delete all partitions. Use "p" command again to make sure there are no partitions left.

1. Create a 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
```

2. Create a 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
```

3. Create a SPL 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
```

4. Write changes to disk and exit fdisk by typing "w" followed by pressing “ENTER”.

```
w
```

8.2.1.2. Format Linux kernel image and device tree binary area

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

8.2.1.3. Format Linux root file system area

```
$ sudo mkfs.ext3 /dev/sdx2
```
8.2.2. Update the Linux kernel and root file system

8.2.2.1. Copy the Helio board SPL image to the micro SD card.

```bash
$ cd ~/MAI_Linux_BSP_Package/Altera_linux
$ sudo dd if=helio_1gb_cl7_spl.bin of=/dev/sdx3 bs=64k seek=0
```

8.2.2.2. Copy the U-boot image to the micro SD card

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

8.2.3. Update the Linux root file system and kernel image file to micro SD card.

8.2.3.1. Mount the micro SD card partition to host PC.

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

8.2.3.2. Change to the software package directory.

```bash
$ cd ~/MAI_Linux_BSP_Package
```

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

---

Figure 8-43. Driver Package directory
8.2.3.3. Run the update micro SD card script.

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

Figure 8-44. Auto update SD card script

```
Extract root file system to SD card.

Copy image files to SD card.
Update the SD card end.
```

Figure 8-45. Successful completion of SD card update
9. Run DirectFB Demonstration

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

9.1.1. Install micro SD card

1. Insert the micro SD card into the Helio main board micro SD card slot.

9.1.2. Program FPGA with reference design

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

9.1.3. Reset system

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

```
Figure 9-1. DirectFB demo application main screen
```

9.1.4. Start DirectFB example applications

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

9.1.5. Stop DirectFB example application

9. 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.