
LINUX BSP FILES FOR THE DEVICE SOLUTIONS TOPAZ I.MX25 DEVELOPMENT KIT

Release Notes, version 1.9

REVISION HISTORY

Revision #	Revision Date	Notes
1.0	June 13, 2010	Initial document
1.1	June 14, 2010	Reviewed and updated flashing instructions
1.2	June 15, 2010	Updated source files to match binaries in 1.1
1.3	July 14, 2010	Enabled USB in kernel.
1.4	July 28, 2010	Added additional directions for setting up a development environment.
1.5	August 4, 2010	Enable 7" LCD. Added simple install script.
1.6	August 9, 2010	Enable SD slot. Enable all expansion header IO signals. Enable 4.3" LCD. Panel type is read from LCD EEPROM. Enable SGTL5000 audio.
1.7	August 12, 2010	Add splash screens.
1.8	August 30, 2010	Fix boot from NAND. Add Topaz Flasher instructions. Add boot from NAND instructions.
1.9	February 15, 2011	Fix invalid MAC Address when booting from NAND. Support detection of newer LCD panels. Fix UART3 and UART4. Added 8kHz and 16kHz support to SGTL5000 codec.

INTRODUCTION

The file `Topaz_BSP_2_6_31_1_9.zip`, contains the binaries and patches to the LTIB tool to recreate the binaries. The file `Topaz_Binaries_2_6_31_1_9.zip` contains just the binaries.

Contents of the `Topaz_BSP_2_6_31_1_9` package are:

- Install - a simple installation script.
- Binaries directory - prebuilt binaries for bootloader, kernel and root file system.
- devicesolutions directory - Topaz LTIB configuration files
- Patches directory - patches to be applied to the kernel
- `mtdmap.txt` - NAND MTD partition layout.

The files contained in binaries are:

- u-boot - This image can be loaded into RAM using the JTAG debugger and executed. This is useful when the board does not yet have a bootloader in flash. The procedure for this is debugger dependent but once the image is in memory, the NAND flash and

TFTP capabilities of the bootloader can be used to download u-boot.bin and flash it to offset zero in the NAND flash.

- u-boot.bin - Flash this to the board to make the bootloader available on power up.
- rootfs.jffs2 - The file system is loaded to the board via tftp and flashed to the "nand.rootfs" offset listed in the mtdmap.txt file.
- ulmage - The Linux kernel. Flash it to the "nand.kernel" area of the mtdmap.txt file.

SYSTEM REQUIREMENTS

LINUX DISTRIBUTION

The tools for the binary demo should run on most Linux host distributions that have glibc-2.2.x or later. It was developed using Ubuntu 9.10.

<http://releases.ubuntu.com/karmic/>

BSP INSTALLATION

The following installation instructions assume the BSP zip file has been extracted to ~/Topaz/ by a user named "topaz" and will create an installation of the BSP in /home/topaz/TopazBSP/

DOWNLOAD LTIB

The Topaz BSP uses the LTIB (Linux Target Image Builder) tool as supported by Freescale to perform builds. Download the i.MX25 Linux 2.6.31 Source Code Files 2009.12 Release from Freescale at this URL:

http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=IMX25PDK&fosp=1&tab=Design_Tools_Tab

(Note: You will need to create a Freescale user account)

Extract LTIB from the tar file:

```
topaz@topaz:~$ tar xzf L2.6.31_09.12.00_SDK_source.tar.gz
```

This will create a folder named "L2.6.31_09.12.00_SDK_source"

INSTALL LTIB PREREQUISITES

LTIB requires several packages:

```
topaz@topaz:~$ sudo apt-get install gcc build-essential libncurses-dev m4  
bison rpm ccache flex tcl gettext libfreetype6-dev libglib2.0-dev libxt-dev  
libdbus-glib-1-2 libgtk2.0-dev liborbit2-dev libbonobo2-dev intltool libdbus-  
glib-1-dev
```

INSTALL LTIB

This will create a new LTIB installation that the Topaz BSP will be installed into. Change to the Freescale "L2.6.31_09.12.00_SDK_source" directory and run the "install" script:

```
topaz@topaz:~/L2.6.31_09.12.00_SDK_source$ ./install
```

Accept the LTIB EULA and then choose an installation directory of **/home/topaz/TopazBSP**. LTIB will now be extracted.

INSTALL TOPAZ BSP

Unzip the Topaz zip file and run the "install" script with an the LTIB install directory as an argument:

```
topaz@topaz:~$ unzip Topaz_BSP_2_6_31_1_5.zip
topaz@topaz:~$ cd Topaz
topaz@topaz:~/Topaz$ chmod +x install
topaz@topaz:~/Topaz$ ./install ~/TopazBSP
```

The Topaz configuration files and patches will now be added to the LTIB installation.

BUILD TOPAZ BSP

Change to the LTIB installation directory and run "ltib":

```
topaz@topaz:~$ cd TopazBSP/ltib
topaz@topaz:~$ ./ltib
```

The first time through it will run through three configuration menus, you should accept all the default settings. The Topaz BSP will now be built. These are some of the important outputs of the LTIB build process:

```
~/TopazBSP/ltib/roofs.jffs2  The Linux root file-system in a format that can be
                             programmed into NAND flash.
~/TopazBSP/ltib/rootfs/      The Linux root file-system in a format suitable for an NFS
                             root mount.
~/TopazBSP/ltib/rootfs/boot/ Contains the bootloader (u-boot) and the Linux kernel
                             image (ulmage).
```

For further instructions on using LTIB and to learn how to modify source code and rebuild individual packages please visit the main LTIB page at <http://ltib.org/>

FLASHING INSTRUCTIONS

To get the initial boot loader on the device, you may use one of the following procedures:

- Flash using the Topaz Flasher
- Flashing using JTAG

FLASHING USING JTAG AND U-BOOT

The following is the flashing procedure using the OpenOCD debugger and the Olimex ARM-USB-OCD JTAG dongle .

Flashing U-Boot is a two-stage process. The first stage places u-boot.bin in memory. U-boot is then used to download and flash u-boot.bin from a TFTP server on the host machine. Once u-boot is programmed into NAND you can skip the JTAG portion of these instructions for subsequent updates of the bootloader and other NAND resident files.

You will need:

- TFTP server with u-boot.bin binary.
- Properly setup and configured JTAG debugger. (OpenOCD)

1. Connect a serial port and network cable to the board. Open a serial port terminal program (minicom is recommended for Linux). Settings are 115200 8N1 no flow control.

2. In the openocd scripts directory, run the following command to start the JTAG server:

```
topaz@topaz:~/Topaz/openocd/scripts$ openocd -f interface/olimex-arm-usb-ocd.cfg -f board/topaz.cfg
```

3. Connect to the JTAG server in a separate terminal by running the following:

```
topaz@topaz:~$ telnet localhost 4444
```

This will start an interface to the JTAG server - this is where you will enter commands to reset the target and transfer images etc..

4. Inform the JTAG where to get the file for a subsequent load. In the telnet session to the JTAG, enter the command:

```
> fast_load_image <path_to_file>/u-boot
```

5. Reset the Topaz target to a known state and copy the image data to it. In the telnet session to the JTAG, enter the commands:

```
> reset init
> fast_load
```

Note the load address (0x83f00000).

6. Execute the bootloader by resuming at the load address. In the telnet session to the JTAG, enter the commands:

```
> resume 0x83f00000
```

The output from the bootloader will be displayed in the serial terminal. Press any key to stop the auto boot countdown. You should be at 'Topaz U-Boot >' prompt. Note: It is normal to see some CRC errors reported or some bad blocks with NAND flash.

7. Clear the NAND flash to a known state:

```
Topaz U-Boot > nand erase clean
```

8. Setup u-boot IP addresses for the board (ipaddr) and for the TFTP server that has the u-boot.bin file (serverip):

```
Topaz U-Boot > setenv ipaddr <board_ip>
Topaz U-Boot > setenv serverip <server_ip>
```

10. Download u-boot.bin:

```
Topaz U-Boot > tftp u-boot.bin
```

The TFTP command always loads files into RAM at the address, 0x80800000. Note the size of the file that has been loaded (typically less than 0x30000 for the bootloader).

11. Flash u-boot.bin:

```
Topaz U-Boot > nand write 0x80800000 0 0x30000
```

12. U-Boot will now run when you reset the Topaz board.

This same method can be used to program the kernel (ulmage) and root file system (rootfs.jffs2) to NAND. If the bootloader is already present in NAND then use that, there is no need to use JTAG to load a new one. Use TFTP to download the file and then select the correct offset by referencing the mtdmap.txt. The new offset is then used as a the second parameter to the “nand write” command in step 11.

FLASHING USING THE TOPAZ FLASHER

The Topaz Flasher may be downloaded from:

<http://guruce.com/topaz-flasher>

Note that this tool runs on 32-bit or 64-bit Windows. **If you are running Linux, you will need to use the JTAG flashing instructions.**

1. Make sure the USB connection between the desktop and Topaz board is disconnected and the Topaz is not powered
2. Start the Topaz Flasher Utility
3. Put the board into USB boot mode by setting SW1-1 and SW1-2 to ON.
4. Power the Topaz by connecting the USB cable between the desktop and the board
5. If the driver correctly installed, the Topaz flash utility will report “Topaz initialized”
6. Select the menu item “Settings->Flash...” and choose u-boot.bin as the Custom File entry. Make sure the Address field is set to zero. Select “Save” to save these new settings.
7. Make sure the “Flash Custom” select box is ticked and then click “Flash!”. The u-boot image will quickly be programmed into FLASH.
8. Power down the Topaz board and revert SW1-1 and SW1-2 to OFF.

Topaz Flasher can also be used to program the kernel (ulmage) and root file system (rootfs.jffs2) to NAND. Simply select the correct offset by referencing the mtdmap.txt file.

U-BOOT CONFIGURATION

BOOTING FROM NAND

The kernel and root file system can be programmed into NAND to enable a standalone boot mode. You can use this mode to evaluate the prebuilt binaries that ship with the BSP.

Follow the directions in the “Flashing Instructions” section of this document and program the ulmage file into NAND at offset 0x300000 and the rootfs.jffs2 file at offset 0x800000. The offsets are listed in the mtdmap.txt file.

Run the “bootcmd_nand” macro within the bootloader to boot from NAND:

```
Topaz U-Boot > run bootcmd_nand
```

By default the board will attempt to boot from the network. You can change this behavior by setting the “bootcmd” environment variable:

```
Topaz U-Boot > setenv bootcmd 'run bootcmd_nand'  
Topaz U-Boot > saveenv
```

BOOTING FROM THE NETWORK

The bootloader defaults to boot from the network. You will need to supply correct IP addresses for “ipaddr” and “serverip” environment variables. You will also need to change the “nfsrootfs” environment variable to point to the NFS exported LTIB root filesystem.

OTHER TOOLS

GDB

The GNU Project Debugger (GDB) can be used for application debugging by connecting to the target server over Ethernet. It may be used for kernel debugging via a JTAG or the kernel debugging enhancements. JTAG was used in this case. GDB can be found here:

<http://www.gnu.org/software/gdb/>

OPENOCD AND OLIMEX

The Open On-Chip Debugger (OpenOCD) along with the Olimex ARM-USB-OCD JTAG dongle provides JTAG access. The OpenOCD board configuration file for the Topaz board is provided. The Olimex ARM-USB-OCD JTAG dongle can be found here:

<http://www.olimex.com/dev/index.html>

OpenOCD can be found at: <http://sourceforge.net/projects/openocd/>

CONNECTING THE GDB DEBUGGER TO OPENOCD

Start the OpenOCD debugger’s GDB server:

```
topaz@topaz:~/Topaz/openocd/scripts$ openocd -f interface/olimex-arm-usb-ocd.cfg -f board/topaz.cfg
```

Start an interface to the JTAG in a separate terminal window by running the following:

```
topaz@topaz:~$ telnet localhost 4444
```

In a third terminal window start GDB (this assumes the ARM cross-compiler tool chain is on the path):

```
topaz@topaz:~$ arm-none-linux-gnueabi-gdb
```

In the GDB console, connect to the target via OpenOCD.

```
(gdb) target remote localhost:3333
```

Alternatively use a pipe to connect to OpenOCD allowing GDB to start and stop the debugger. See (<http://openocd.berlios.de/doc/html/GDB-and-OpenOCD.html>) for an example.

Add symbol files for the kernel and/or boot loader (assumes these are built with debug symbols). VMLinux for the kernel, u-boot for the bootloader.

TFTP

TFTP is a simple file transfer protocol used by the bootloader to download files from a server. For Ubuntu, the package “tftpd” provides the needed server (use the Synaptic Package Manager available with Ubuntu and type “tftpd” in the Quick Search window). After installation, the file

```
/etc/inetd.conf
```

will contain the required initialization parameters. The last item is the location of the directory used by the server for the files and is usually modified to be /tftpboot.

After modifying this file, restart the TFTP server:

```
topaz@topaz:~$ sudo /etc/init.d/openbsd_inetd restart
```

On subsequent boots of the host the server will automatically be started.

NFS

NFS is used to host the root file system on the host during development. For Ubuntu, the package “nfs-kernel-server” provides the NFS server component.

After installation, add a line similar to the following to the file /etc/exports to export the root file system:

```
/home/topaz/TopazBSP/ltib/rootfs *(rw,no_root_squash,no_subtree_check, sync)
```

Then, restart the NFS service:

```
topaz@topaz:~$ sudo /etc/init.d/nfs-kernel-server restart
```

KNOWN ISSUES

Random segment faults can occur when the JTAG debugger is connected and the screen saver is activated. Segment faults do not occur if the JTAG is disconnected.

RELATED LINKS AND DOCUMENTATION

Ubuntu	Ubuntu 9.10 (Karmic Koala)	http://releases.ubuntu.com/karmic/
LTIB	Linux Target Image Builder	http://ltib.org/
GDB	GNU Project Debugger	http://www.gnu.org/software/gdb/

OpenOCD	Open On-Chip Debugger	http://sourceforge.net/projects/openocd/
---------	-----------------------	---

CONTACT INFORMATION

For any questions, please contact info@trygtech.com