Implementing Second-Stage Loader on ADSP-BF707 Blackfin® Processors

Contributed by David H *Rev 1 – June 15, 2015*

Introduction

While the ADSP-BF70x Blackfin® processors natively support numerous boot modes, there may be applications that require additional boot-time functionality. To support extensions to the boot process, a second-stage loader (SSL) can be introduced, where a small application is initially loaded onto the processor using a natively supported boot mode, and then the application is customized to perform automated tasks as part of the processor boot process. This EE-Note describes how to write a SSL for the ADSP-BF707 Blackfin processor to selectively boot one of multiple executable files (DXEs) from SPI2 flash memory using basic GPIO push-buttons during system start-up.

This document describes what an SSL is, how to create loader (LDR) files from DXEs, and how to write LDR files to SPI flash memory. The example code associated with this document was tested with CrossCore® Embedded Studio (CCES) version 1.1.1 using a revision 1.0 ADSP-BF707-EZKIT evaluation system with processor silicon revision 1.0.

Second-Stage Loader (SSL)

An SSL can be used to allow the processor to boot from a selection of mapped applications.

In this example, the ADSP-BF707 Blackfin processor is operating in Memory-Mapped Mode for booting from SPI2 memory. The processor boots from SPI memory and loads the SSL application, which configures the processor to accept the PB1 and PB2 push-buttons as GPIO inputs before idling until either is pressed. The push-buttons determine which of two unique applications to boot from SPI memory by calling the `adi_rom_Boot()` function with an offset into SPI flash memory associated with applications that will blink either LED3 or LED4 on the board. The method described by this EE-Note uses three separate LDR images, each of which is created separately and programmed into specific locations within the SPI flash memory, as shown in Table 1.

LDR	SPI Memory Location
LED_4	0x40020000
LED_3	0x40010000
SSL	0x00000000

Table 1. SPI2 Flash Memory Composition

Booting DXE Applications

The ADSP-BF707 Blackfin processor ROM space includes the `adi_rom_Boot()` function for selecting a memory address to boot from. Detailed information on the ROM functions can be found in the *ADSP-BF707 Blackfin® Processor Hardware Reference Manual*[1] (HRM). To use the ROM API, the file `cdefBF70x_rom.h` needs to be included in the project:

```
#include <cdefBF70x_rom.h>
```

In this SSL application, `adi_rom_Boot()` is called in the interrupt routines for the two push-button switches, as follows:

```
adi_rom_Boot(LED_3,0,0,0,0x00000207);  // PB1
adi_rom_Boot(LED_4,0,0,0,0x00000207);  // PB2
```

`LED_3` and `LED_4` are macros in the `BF707_SSL.c` source file defining the physical addresses in SPI flash memory where the corresponding LDR files reside. `0x00000207` is the `dBootCommand` parameter, which is configuring the boot mode to SPI Master Memory-Mapped Mode, as described in Table 2.

Please refer to the *Boot Modes→SPI Master Boot Mode→Run-time API* section of the *Boot ROM and Booting the Processor* chapter of the HRM for details.

Bit No.	Bit Name	Description
31:28	SPEED	0x0 = Maximizes SPI Clock
27	CMDSKIPEN	0 = Disable Command Skip
26:25	IOPROT	b#00 = No SPI I/O Protocol
24:22	DUMMY	b#000 = Do Not Issue Dummy Bytes After Address
21:20	ADDR	b#00 = Issue 1 Address Byte for Read Command
19:16	BCODE	0x0 = Boot Mode Specific Code
14:12	SSEL	b#000 = SPI Slave Signal-Select Is SEL1
11:8	DEVENUM	0x2 = SPI2
6	NOAUTO	0 = Device Detection Enabled
5	NOCFG	0 = Device Configuration Mode Enabled
4	HOST	0 = Master Boot Mode
3:0	DEVICE	0x7 = SPI Memory-Mapped Mode

Table 2. SPI Master Boot dBootCommand

The ADSP-BF70x boot ROM provides the flexibility to choose which SPI chip-select output is used for SPI booting (SEL1 through SEL7), which allows for multiple SPI flash modules to be connected via SPI and booted from using the `adi_rom_Boot()` API.

ADSP-BF70x Blackfin Processor Instruction Cache

By default, revision 1.0 silicon of the ADSP-BF70x Blackfin processor has instruction cache memory enabled at start-up. In this example, instruction cache must be enabled for the SSL in order for it to boot properly. To enable instruction cache, open the `system.svc` file in the **Project Explore**r and click the **Startup Code/LDF** tab at the bottom. In the **Startup Code** tab, locate the **Instruction Cache** pull-down under the **Cache and Memory Protection** section and select **"Enable instruction cache"**.

The LED examples used in this note do not require instruction cache; therefore, code must be placed in the SSL source to disable it prior to calling `adi_rom_Boot()`, which requires resetting the L1 Instruction Memory Control register (L1IM_ICTL) to 0:

```
*pREG_L1IM_ICTL = 0;
```

Creating LDR Files

With the project selected in the CCES Project Explorer view, select **File→Properties** from the pull-down menu:

1> In the tree to the left, select the **C/C++ Build→Settings** page.

2> On the **Build Artifact** tab, select **Loader File** from the **Artifact Type** pull-down.

3> Return to the **Tool Settings** tab and select the **CrossCore Blackfin Loader→General** page.

4> Set the following in the **Settings** page that is displayed:

Boot Mode: SPIO master

Boot Format: Binary

Output Width: 16 bits

Boot code: 1

Use default start address checked

Next to the **Initialization file** dialog box, **Browse…** to the initialization file for the ADSP-BF707 EZ-KIT (the default location is `<CCES root directory>\Blackfin\ldr\init_code\BF707_init`) and click OK.

If the above directory doesn't contain a valid DXE file, one must be built for inclusion with this project. For this exercise, no modifications to the default initialization code are necessary. Simply open the **BF707_init_vxx** project in the above directory and build it. Once built, the needed **BF707_init_vxx.dxe** file will be generated in the **\Debug** directory associated with the initialization code project.

Build the project in Debug mode, and the LDR file will be generated in the **\Debug** folder.

Programming Multiple LDR Files to SPI Flash

Once the LDR images for the SSL and applications have been generated, each LDR needs to be programmed into defined offsets within the SPI flash memory. This is accomplished using the CCES Command Line Device Programmer utility (`cldp.exe`), which runs a kernel program on the processor to read the LDR image over the debug interface and initiate the proper commands to erase and program the SPI flash memory on the ADSP-BF707 EZ-KIT evaluation system. More information regarding the switches and arguments for the CLDP utility can be found in the **Help Contents** in CCES.

The following is the means of utilizing the CLDP to write the three LDRs of interest to the SPI flash using the ICE-1000 that ships with the LITE evaluation board:

1. **Load BF707_SSL.ldr into Flash:**

```
cldp -emu ICE-1000 -proc ADSP-BF707 -cmd prog -driver "<path name>\bf707_w25q32bv_dpia.dxe" -format bin -erase all -offset 0 -file "<path name>\BF707_SSL.ldr"
```

2. **Load LED_3.ldr into Flash:**

```
cldp -emu ICE-1000 -proc ADSP-BF707 -cmd prog -driver "<path name>\bf707_w25q32bv_dpia.dxe" -format bin -erase affected -offset 0x10000 -file "<path name>\LED3_blink.ldr"
```

3. **Load LED_4.ldr into Flash:**

```
cldp -emu ICE-1000 -proc ADSP-BF707 -cmd prog -driver "<path name>\bf707_w25q32bv_dpia.dxe" -format bin -erase affected -offset 0x20000 -file "<path name>\LED4_blink.ldr"
```

For all three invocations of the CLDP utility, the <*path name*> is the full path to the serial flash programmer driver (that comes with the Board Support Package for the evaluation board) and the LDR files for the SSL and example LED applications (e.g., "<ADSP-BF707-EZ-Board BSP Root Directory>\BF707_EZ-Board\Blackfin\Examples\Device_programmer").

Once all three are programmed to the SPI memory, the example application can be run.

Using the SSL Example Program

After all three LDR files are programmed into Flash memory, reset or power-cycle the ADSP-BF707 EZ-Kit to load and run the SSL. Pushing PB1 will boot from memory location 0x40010000 (LED_3), and PB2 will boot from memory location 0x40020000 (LED_4).

Once one application has been booted, the SSL is exited and cannot be used again until the next power-cycle or reset.

References

[1] *ADSP-BF70x Blackfin Processor Hardware Reference.* Rev 0.2, May 2014. Analog Devices, Inc.

[2] *ADSP- BF70x Blackfin Processor Data Sheet.* Rev PrD, 2014. Analog Devices, Inc.

[3] *Embedded Processing and DSP (http://www.analog.com/processors).* May 2005. Analog Devices, Inc.

Document History

Revision	Description
Rev 1 – June 15, 2015 *by David H*	Initial Revision