Technical notes on using Analog Devices DSPs, processors and development tools
Visit our Web resources http://www.analog.com/ee-notes and http://www.analog.com/processors or e-mail processor.support@analog.com or processor.tools.support@analog.com for technical support.

In-Circuit Flash Programming on SHARC® Processors

Contributed by R. Murphy

Rev 2 – February 19, 2007

Introduction

ADSP-2116x, ADSP-2126x, ADSP-2136x and ADSP-2137x SHARC® processors, similar to most re-programmable processors, require that internal program code and data be boot-loaded at power-up. The code and data can be supplied by a host system (e.g., via an SPI or JTAG connection) or can be stored in an on-board non-volatile memory device such as a ROM or in a serial or parallel flash device. The SHARC EZ-KIT Lite® evaluation boards enable you to boot the processor in any of these modes. This EE-Note focuses on PROM booting. This is supported by an 8-Mbit flash memory device (AM29LV081B-120EC) from AMD.

Several options are available to system designers for programming boot code into the flash device.

This document discusses programming the flash using the Flash Programmer utility and bases the discussion on an example flash programmer project.

The discussed procedure in this EE-Note is for ADSP-21262 and ADSP-21161 processors, but it applies to all SHARC processors. ADSP-21161 processors use the external port for programming the flash, and ADSP-21262 processors use the parallel port to program the flash. The approach provided for ADSP-21161 processors applies to ADSP-21367, ADSP-21368, ADSP-21369, ADSP-21371, and ADSP-21375 processors. The approach provided for ADSP-21262 processors applies to ADSP-2126x, ADSP-21362, ADSP-21363, ADSP-21364, ADSP-21365, and ADSP-21366 processors.

Flash Programmer Utility

VisualDSP++® 4.5 development tools or higher includes a Flash Programmer utility that allows you to point the flash utility to a previously programmed loader (`.LDR`) file and program the flash based on a supplied driver file. The driver file is simply a DSP executable (`.DXE`) developed for a specific combination of processor and flash device. The VisualDSP++ tools include driver files for each of the SHARC EZ-KIT Lite evaluation boards, allowing you to program your flash device with your driver file. The driver files for the ADSP-21262 EZ-KIT Lite can be found in the following location:

```
VisualDSP 4.5\212xx\Examples\ADSP-21262
EZ-KIT Lite\Flash Programmer
```

Similar folders exist for all SHARC evaluation boards.

Using the Flash Programmer Code

Alternatively, the Flash Programmer utility can be circumvented; you can use the attached example code and code available in the listing to manually program the image into flash. This is useful for verifying the development of a custom driver for the Flash Programmer utility as well as for simply creating a custom executable for use in programming the end system.

Flash programmer code for the ADSP-21161 processor (using the external port) and the

ADSP-21262 processor (using the parallel port) are included in the associated ZIP file [1] for this EE-Note.

This example shows how the VisualDSP++ tools are used to program an application into flash. *In-circuit programming* (verses burning the flash before it is placed on the board) is frequently used to perform software or firmware updates to systems that are already deployed in the field.

The task requires two separate sets of code: a software routine that programs the data into the flash, and an end application that is ultimately boot-loaded into and run by the processor.

In this example, `flash_programmer.asm` contains a generic flash programming algorithm, a simple LED toggling routine in `blink.asm`. (This routine will serve as the data payload that `flash_programmer.asm` will program into the flash.) The ADSP-21161 blink application toggles flags 4-9 on the ADSP-21161N EZ-KIT Lite evaluation board. The ADSP-21262 blink application toggles LEDs 1-8 on the ADSP-21262 EZ-KIT Lite evaluation board.

In-Circuit Flash Programming

The attached code example demonstrates the four steps involved with in-circuit flash programming.

Step 1. Building the Application

Using the USB debug agent, a JTAG emulator, and/or the VisualDSP++ simulator, you can write and test the application. The application in **`blink.asm`** toggles the LEDs, which are connected to flags 4-9 sequentially for the ADSP-21161 processor. The application in **`ppflags.c`** toggles the LEDs, which are connected to flags 8-15 sequentially for the ADSP-21262 processor. Verify that this routine and your board are functional by activating an ADSP-21161N/ADSP-21262 EZ-KIT Lite debug target in the IDDE and then opening **`blink.dpj`**. Next, load **`blink.dxe`**. After the executable has been loaded, run the project (**`F5`**) and watch the LEDs.

Step 2. Creating the PROM Boot-Image

After verifying the application, use the VisualDSP++ loader to generate a boot-image from the source code. To do this, open the `Project Options` dialog box for `blink.dpj` and specify "Loader file" as the output file type (Figure 1).

The `.LDR` file to be generated will ultimately be included in the flash programming routine as an array. To this end, the `.LDR` file should be in "Include" format (essentially comma-delimited ASCII) with each word being 32-bits, as shown in Figure 2.

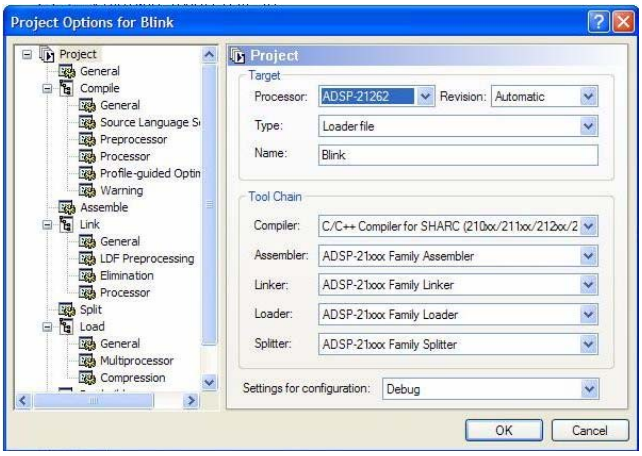

Figure 1. Specifying a loader file as output file type

Click `OK` and then click `Rebuild All`. This rebuilds the project according to the specified project options and generates `blink.ldr`.

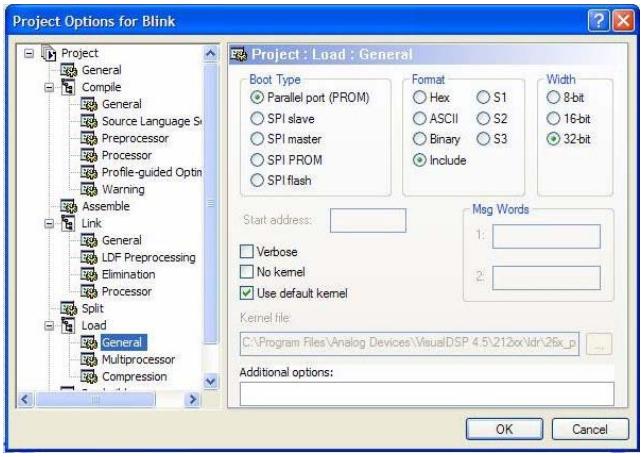

Figure 2. Specifying Parallel port (PROM) as boot type

The flash programmer utility requires that the loader file be in Intel Hex format for programming the flash.

Step 3. Programming the Flash Device

Next, use either the Flash Programmer utility (Step 3A) or the included flash-programmer project (Step 3B) to program the `.LDR` image into the flash device.

Step 3A. Using the Flash Programmer Utility

Perform the following steps:

1. From the VisualDSP++ `Tools` menu, choose `Flash Programmer`.

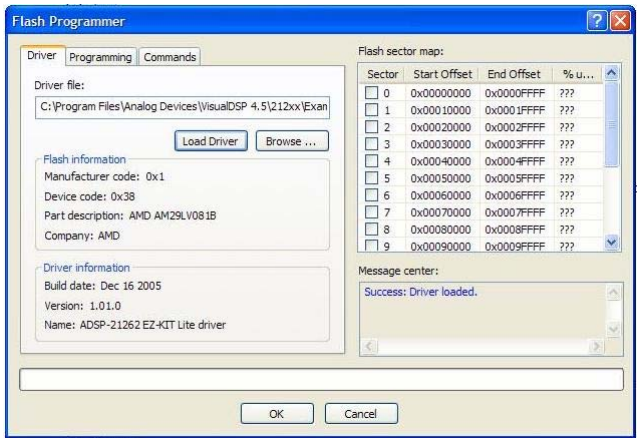

Figure 3. Loading the driver

2. Click `Load Driver`. The utility will load a supplied driver for the AMD flash onto the EZ-KIT Lite board (or you can browse to locate a custom flash driver file).
3. Click the `Programming` tab. Browse to the folder containing `blink.ldr` and click `Program`.

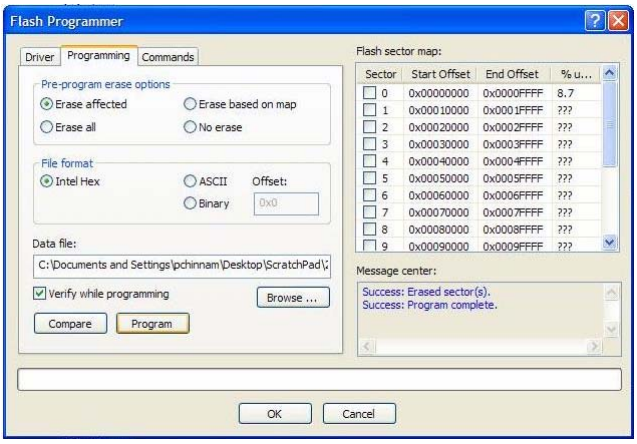

Figure 4. Programming the flash

4. When a "Program complete" message appears, reset the board to boot the new application code.

Step 3B. Using the Included flash_programmer Project

Perform the following steps:

5. Open `flash_programmer.dpj`.
6. Rebuild the project with your application code.

   The last step modifies the flash programming application to include the recently generated `blink.ldr`. To do this, in the file `flash_programmer.asm`, simply declare an array in a section mapped to external memory and initialize it with the payload `.LDR` file. To use the example application (blink), this is done as follows:

```
.section/dm seg_ext8;
.var my_file[]= "blink.ldr";
```

The Linker Description File (`.LDF`) is then used to pack all `seg_ext8` input sections to an external memory segment (also named `seg_ext8`).

With the correct payload file specified, simply rebuild the project to generate the flash programming executable. (`flash_programmer.dxe`).

7. Program the flash, as described next.

   Four LEDs provide feedback on the progress of the flash programming. As each of the following five steps completes, an LED illuminates to indicate that it was successful (or blinks rapidly to indicate failure).

   The five steps are as follows:

   1. Verify that the payload fits into the 1 M x 8-bit flash.
   2. Reset the flash.
   3. Erase the flash (completely or by-sector).
   4. Write the payload `.LDR` file into the flash.
   5. Verify the data.

   If all five steps are successful, all of the LEDs flash briefly, indicating that the in-circuit flash programming was successful. You may now reset the board, and the new application code will be booted.

Additional Notes

In this example, the payload buffer, `my_file[]`, is initiated at build time. In a real system, this buffer would be initialized during execution. The flash data may be imported from a serial port, link port, or an SPI port. The data may be placed into memory by an on-board host processor.

Viewing the flash and external memory space via memory windows in the IDDE is supported by the VisualDSP++ tools using an Analog Devices In-Circuit-Emulator (ICE). However, during debug of the application, this may cause the debugger window to hang momentarily, especially while single stepping. To avoid this bottleneck, close the external memory windows while the code is running, and open them as needed to verify the memory contents.

Summary

This EE-Note discusses two ways of programming the in-circuit flash on SHARC processors. The example code provided with the EE-Note demonstrates the in-circuit flash programming for ADSP-21161 and ADSP-21262 processors. The example code is tested on the flash in the ADSP-21161N and ADSP-21262 EZ-KIT Lite evaluation boards.

References

[1] *Associated .ZIP File*. Revision 2, February 2007. Analog Devices, Inc.

[2] *ADSP-2126x SHARC DSP Core Manual*, Revision 2.0, February 2004. Analog Devices, Inc.

[3] *ADSP-2126x SHARC Processor Peripherals Manual*, Revision 3.0, December 2005. Analog Devices, Inc.

[4] *ADSP-21262 EZ-KIT Lite Manual*, Revision 3.0, August 2006. Analog Devices, Inc.

[5] *ADSP-21262 SHARC Processor Data Sheet*, Revision B, August 2005. Analog Devices, Inc.

[6] *AM29LV081B 1M x 8-Bit Uniform Sector Flash Memory Data Sheet*, Revision D, September 2003. AMD

[7] *ADSP-21161 SHARC Processor Hardware Reference Manual*, Revision 4.0, February 2005. Analog Devices, Inc.

[8] *ADSP-21160 SHARC DSP Hardware Reference Manual*, Revision 3.0, November 2003. Analog Devices, Inc.

[9] *ADSP-2136x SHARC Processor Hardware Reference Manual for the ADSP-21362/3/4/5/6 Processors*, Revision 1.0, October 2005. Analog Devices, Inc.

[10] *ADSP-21368 SHARC Processor Hardware Reference Manual*, Revision 1.0, September 2006. Analog Devices, Inc.

Document History

Version	Description
Rev 2 – February 19, 2007 by C. Prabakaran and Jeyanthi Jegadeesan	Modified the document to make it common for all SHARC processors. Merged with the contents from *In-Circuit Flash Programming on the ADSP-21161 EZ-Kit Lite (EE-150)*.
Rev 1 – January 23, 2004 by R. Murphy	Initial Release