TMS320C2834x
SPI Serial EEPROM/Flash Programming Utilities

SDFlash EEPROM/Flash Programming Algo Version 1.0

This download includes algorithm files that can be used with SDFlash to program a SPI Serial EEPROM/Flash connected to SPI-A of a C2834x device.

SDFlash is a product of Spectrum Digital Inc. (www.spectrumdigital.com)

Document updated:
June 8, 2009

Contents:

1. Revision History

Version 1.0

- This version is the first release of the TMS320C2834x SDFlash SPI Programming Utilities.

2. Release Notes

- This release of the SDFlash 2834x Serial EEPROM/Flash Programming Algo is based on utility functions written to write and read from a Serial EEPROM/FLash connected to SPI-A of a 2834x device. The source code for the utility functions is provided as-is.
- SDFlash does not support the XDS560 scan controller.
- You should not run SDFlash and Code Composer at the same time. This results in two different applications trying to control the DSP. During programming, SDFlash will have complete control of the device. No user application code can be running in parallel.

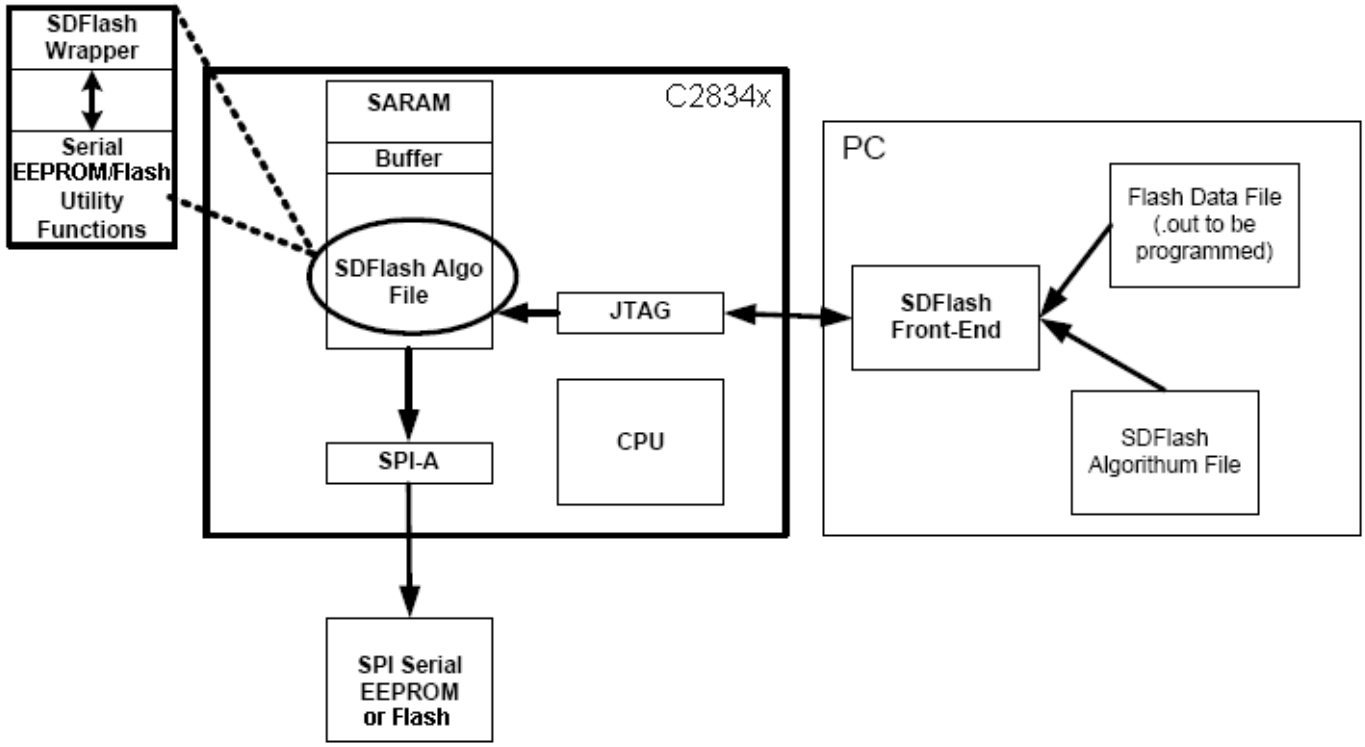
3. Known Limitations

- Support is currently provided for the Atmel SPI Serial EEPROMs (AT25HP256/512, AT25256A) and SPI Serial Flash (AT25F1024A). The utility functions can be modified by the customer to support other EEPROM's and Flash's if required.

4. SDFlash Overview

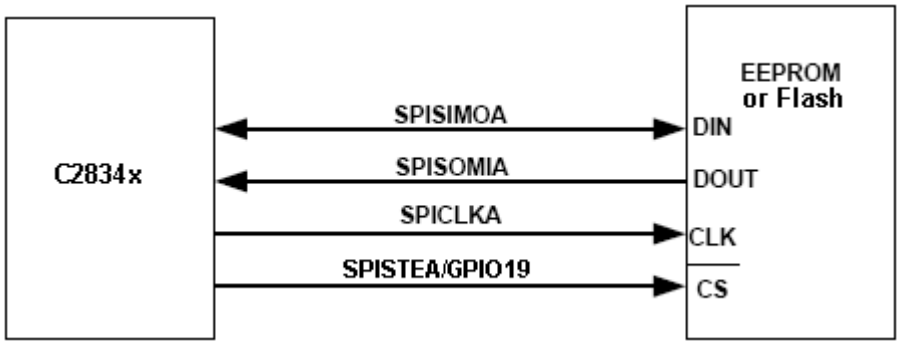
SDFlash is a generic front-end application owned by Spectrum Digital Inc (www.spectrumdigital.com). This application provides a generic interface to the JTAG communications channel that can be used to support programming of internal or external flash.

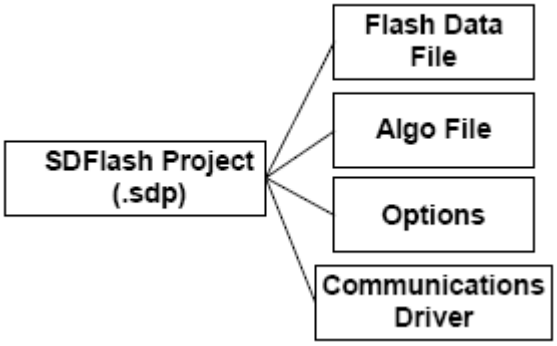
In order to perform the required operations, SDFlash downloads an algorithm file onto the DSP. This algorithm file is an executable (.out) file for the DSP being programmed and is executed on the DSP target. In this case, the SDFlash algorithm file consists of an SDFlash wrapper and 2834x serial EEPROM/Flash programming library functions (for reading from, writing to, and erasing external memory). The SDFlash wrapper has well defined standard functions and variables that are accessed by SDFlash over the JTAG channel. These functions in turn make calls to the EEPROM/Flash programming utility functions to perform operations on the serial EEPROM device. Note: because the SDFlash interface is separate from the algorithm file, the version of the SDFlash interface will differ from the version of the algorithm file.



The flash data file is first prepared to include the header information required by the on-chip SPI bootloader. With this information included, the "boot to SPI" option can then be used to copy the application into the 2834x device at boot time. Detailed information on the SPI boot header and the 2834x boot mode options can be found in *TMS320x2834x Delfino Boot ROM Reference Guide* (sprufn5) available on TI's website.

Ensure that your development board has the required hardware connections and circuitry for your SPI-EEPROM or SPI-Flash memory chip The EEPROM/Flash should be connected to SPI-A and the boot mode pins set to “boot to SPI” mode. The connection between the DSP and the EEPROM/Flash chip are shown below:





For SDFlash to program the EEPROM or Flash, it must have the following information:

- Location of the EEPROM/Flash algorithm file
- Location of the flash data file – that is the prepared data (.out file) to program into EEPROM/Flash device.
- Which JTAG driver to use
- Information about the JTAG scan chain

All of this information is stored in an SDFlash project file (.sdp) that can be edited though the SDFlash GUI interface. A sample SDFlash project has been provided in this download.

Section 5 of this document will guide you through the SDFlash setup and show you how to use the provided sample SDFlash project to create your own project to program the Serial EEPROM/Flash.

5. Quick Start Guide

The following is a step-by-step guide for using the SDFlash utility to program the EEPROM or Flash chip via the SPI-A interface on your 2834x device.

This quick start guide will refer to the following default directory locations:

<CCS base>	default Code Composer Studio install directory: "c:\CCStudio_v3.3"
<SDFlash base>	default SDFlash directory <CCS base>\specdig\SDFlash

These directories may be different for your particular installation.

5.1. Prepare the .out (COFF) file of the application to be stored in the SPI Serial EEPROM/Flash.

The .out file of the application to be programmed must be re-formatted before it can be programmed into the Serial EEPROM/Flash device.

If you are using the example image provided you can skip this step and use the prepared image that is provided. When you are ready to prepare your own application, follow the detailed steps in section 6.

5.2. Make sure the target and emulator are setup properly:

If you are using an XDS510USB/XDS510/XDS510pp+/SPI515 emulation controller, use the SDConfig utility. SDConfig is part of the default emulation installation and will be installed in your *<CCS base>\specdig\sdconfig* directory

5.3. Install the SDFlash flash support utility.

A version of SDFlash is included with your Code Composer Studio install. SDFlash will typically be installed in your <CCS base>\specdig\SDFlash directory.

SDFlash is a generic utility supplied by Spectrum Digital Inc. to interface to user written algorithms. In this case, Texas Instruments Inc has supplied the algorithm file. Users should check the SD website for updates to this utility.

5.4. Download the latest 2834x SDFlash SPI EEPROM/Flash Programming algorithm files.

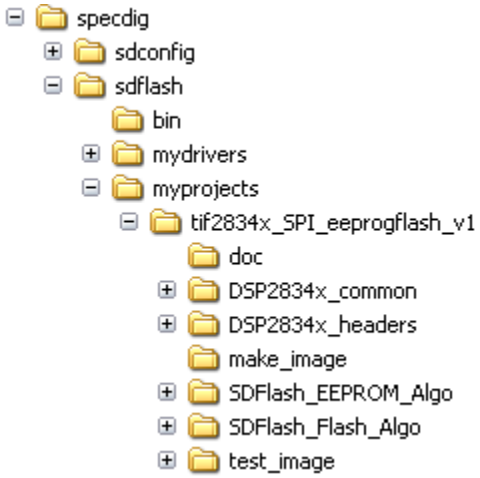
Future releases and upgrades will be available for download from the Spectrum Digital website in the *Support Utilities->SdFlashAlgo's* section.

5.5. Unzip the Serial EEPROM programming algorithm files into the myprojects subdirectory of SDFlash.

For a typical install this will be the *<CCS base>\specdig\SDFlash\myprojects* directory.

This will automatically create a directory indicating the processor and version of the utilities.

Note: If you had installed an earlier release of SDFlash on your system you may have additional sub-directories to those shown, such as an algo directory. With the release of SDFlash V1.3 the algo directory was replaced with the *myprojects* directory. Presence of this directory will not effect the operation of SDFlash.



5.6. Make any required changes to the EEPROM/Flash Programming Algorithms.

The EEPROM/Flash programming algorithms configure the 2834x device with the following configuration parameters:

PLLCR = 19 for SYSCLKOUT = 20*(19+1)/2 = 200 MHz (works on all C2834x devices)
LOSPCP = 0x0002 for LSPCLK = 200 Mhz/4 = 50Mhz

SPIBRR = 0x0015 for AT25256A serial EEPROM and AT25F1024A serial flash
SPIBRR = 0x0027 for AT25HP256 and AT25HP512 serial EEPROMs

These parameters result in an SPI clock rate of 50MHz/(0x0015+1) = 2.27MHz or 50MHz/(0x0027+1) = 1.25MHz.
This configuration and algorithms have been tested on the C28346 with an Atmel AT25HP256, AT25HP512, and AT25256A serial EEPROMs, and AT25F1024A serial flash. If your system has other requirements you must modify the algorithms for your system.

To change the algorithms, load the SDFlash algo project into Code Composer Studio. This project is located at:

EEPROM: *<SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\SDFlash2834x_Eeprog.pjt*
Flash: *<SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\SDFlash2834x_Flashprog.pjt*

To modify the LOSPCP and SPIBRR values, edit the file: *SPIEEPROM28_SPI_Eeprog_Library.h* for the EEPROM algorithm and *SPIFLASH28_SPI_Flashprog_Library.h*
To modify the PLLCR value, edit the file: *SDFlash28x_SPI_Eeprog_Wrapper.h* for the EEPROM algorithm or *SDFlash28x_SPI_Flashprog_Wrapper.h* for the flash algorithm.

After your modifications, re-build the algorithm file.

5.7. Make sure Code Composer Studio is not running and Run SDFlash.

5.8. Load the supplied SDFlash sample project.

SDFlash uses .sdp project files to store information required to erase a device and program an .out file into a device. Sample 2834x projects have been included in *tif2834x_eeprogflash_v1.zip* for use as project templates.

Using *File->Open Project* in SDFlash, browse to and load the appropriate sample SDFlash project. For a typical installation, these files will be found in the following location:

2834x XDS510USB:
EEPROM: *<SDFlash base>\myprojects\tif2834x_eeprogflash_v1\Sample_2834x_Eeprog.sdp*
Flash: *<SDFlash base>\myprojects\tif2834x_eeprogflash_v1\Sample_2834x_Flash.sdp*

5.9. Modify the SDFlash project (if required) to locate the various elements such as device driver, algorithm file and flash data file.

If you installed CCS and SDFlash in the *<CCS base>* and *<SDFlash base>* directories shown below, then usually only the *Flash Data File* on the **Program Tab**, and possibly the *Emulator Address/ID* on the **Target Tab** needs to be changed.

By default all flash projects are setup relative to the default TI CCS base directory "c:\ CCStudio_v3.3" for Code Composer Studio v3.3 For example:

<CCS base>	default Code Composer install directory: "*c:\CCStudio_v3.3*"
<SDFlash base>	default is *<CCS base>\specdig\SDFlash*
SDFlash binary	default is *<CCS base>\specdig\SDFlash*
Flash projects	default is *<CCS base>\specdig\SDFlash\myprojects\<projectname>*

To change any of the directory paths or project settings from their default values, open the project settings dialog box: *Project->Settings*

Target Tab:

- *Driver:* This is the Code Composer Studio™ emulation driver (*.dvr) file that is used to communicate with the target. The driver files can be found in the *<CCS base>\drivers* directory.
  default for XDS510USB: <CCS base>\drivers\sdgo28xusb.dvr
- *Emulator Address/ID:*
  default for USB: 510
- *Board file:* File that provides SDFlash information on how many devices are on the JTAG scan chain. For a single 28x device on the scan chain, the default board file can be used. For systems with more devices on the scan chain, use the board file generated by Code Composer Studio to access your device. This file is found in the *<CCS base>\cc\bin\BrdDat* directory. The default board file is *<SDFlash base>\myprojects\ tif2834x_eeprogflash_v1\ccBrd028x.dat*
- *Processor name:* default is cpu_0

Erase Tab:

- *Algorithm File:*
  For EEPROM:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_eeprog.out
  For Flash:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_flash.out.out
- *Timeout:* leave as 200 or higher
- *For all other boxes the default is blank.*

Program Tab:

- *Algorithm File:*
  For EEPROM:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_eeprog.out
  For Flash:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_flash.out
- *Flash Data File:* This is the prepared .out file that you want to program into the SPI serial EEPROM/Flash.The steps for preparing the source application file to the correct format are described in section 6.
  For the example image, the prepared file can be found at:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\test_image\Debug\spi_rom_2834x.out
  The example application toggles the GPIO pin 0 while checking an initialized memory pattern. The memory is initialized during boot time. If anything is found incorrect in the initialized memory the GPIO toggling will stop.
- *Timeout:* leave as 200 or higher
- *For all other boxes the default is blank.*

Verify Tab:

- *Algorithm File:*
  For EEPROM:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_eeprog.out
  For Flash:
  <SDFlash base>\myprojects\tif2834x_eeprogflash_v1\SDFlash_Algo\Debug\SDFlash2834x_spi_flash.out.out

- *Timeout:* leave as 200 or higher
- *For all other boxes the default is blank.*

5.10. Save the SDFlash project file: *File->Save Project As.*

Once you have made the required changes select ok and save the project using the name of your choice: *File->Save Project As.*

If you changed the *Emulator Address/ID* setting on the **Target Tab**, you should get a message that the current driver was unloaded and a new driver has been loaded. This operation is required to synchronize the SDFlash project settings with SDConfig. If you did not change the *Emulator Address/ID*, then you will not get this message.

Should you want to program a different .out file into the EEPROM/Flash, use this project as a template and change the *Flash Data File* on the **Program Tab.**

5.11. Reset the device: *Device->Reset*.

You will get a pass/fail message in the output window.

5.12. Erase/Program/Verify your device: *Device->-Flash*.

Check or un-check the operation(s) you want to perform then select start. Each checked operation is executed from left to right, with continue on success and abort on fail.

Note that the erase function has been included only for completeness. For an EEPROM it is not required to erase the device before programming and thus this step is not required.

5.13. Optional: Repeat erase/programming for each device to be programmed.

If additional devices are to be programmed, the target can be powered down and a new target connected without closing the SDFlash utility. Once the new target is connected, reset the part (*Device->Reset*) and erase, program, verify (*Device->Flash*) the device as described in 5.11 and 5.12.

5.14. Boot the code from SPI-A

Power cycle or hardware reset the device. The boot mode options should be set to “boot to SPI-A”. The code from the EEPROM/Flash will be loaded into the device. If you are using the example image, you should see GPIO0 toggle. Note, if you reset the device from the debugger, the PLLCR register is not reset. This can cause a baud rate issue due to SYSCLKOUT being higher then expected. For power-on or a hardware reset PLLCR is cleared and this is not an issue.

6. Preparing the Application .out File for Programming

There are a few basic steps that need to be completed before programming the application .out file into the Serial EEPROM/Flash device. These two steps are described in detail in this section.

6.1. Locate and modify the included fmt_2834x.bat batch (.bat) file.

This batch file will be used to help you automate the file preparation process. This .bat file can be found in the following location:

<SDFlash base>myprojects\tif2834x_eeprogflash_v1\make_image\fmt_2834x.bat

and can be used as a template and modified to suite your preferences. This file was tested on a Windows XP system.

6.1.1.Specify the location of files that will be used.

Create a copy of **fmt_2834x.bat** and edit the top of this of the file to specify the directory path to the required utilities, name, and directory location of the .out file that you want to prepare. As supplied the file is setup for a default installation and will prepare the sample image.

```
REM Code Composer Studio install location. This is used to access the hex2000 utility.
SET CCS_DIR=C:\CCStudio_v3.3

REM Specify the directory where the image will be made.
SET
MAKE_IMAGE_DIR=C:\CCStudio_v3.3\specdig\sdflash\myprojects\tif2834x_eeprogflash_v1\test_image\debug

REM Specify the name of the file to prepare
SET SPI_COFF=image.out

REM Specify location of Asciihexlib.exe
SET ASCIIHEX_DIR= C:\CCStudio_v3.3\specdig\sdflash\myprojects\tif2834x_SPI_eeprogflash_v1
```

6.1.2.Specify the SPICLK rate that will be used during the boot load process.

To do this, specify the initial value for the LOSPCP register and the SPIBRR register in your copy of the batch file. These values will be included in the SPI boot header. These values will be used to initialize the LOSPCP register and SPIBRR register by the SPI boot loader.

```
REM Specify the LOSPCP register value for the SPI bootloader
SET LOSPCP_REG=0x00

REM Specify the SPIBRR register value for the SPI bootloader
SET SPIBRR_REG=0x07
```

Caution:
The user should select the SPICLK rate based on their design specific EEPROM requirements. The C2834x SPI boot loader has an option to change the SPICLK rate based on information included in the SPI boot header.

After reset, prior to executing the SPI boot loader, the C2834x clock rate is 20 MHz/2 = 10MHz. The boot ROM sets LSPCLK is set to divide by 4 (lospcp = 0x02), and the SPI baud rate register is set to 0x7F (spibrr = 0x7F). Hence, the SPI clock rate is initially (10 MHz/4)/(127+1) = 19.5 kHz.

In this example, the SPI boot loader changes the SPICLK rate based on the boot loader header information received. The LOSPCP is set to 0x00 (LSPCLK is divide by 1), and SPIBRR is set to 0x7. Hence, the SPI clock rate is changed to: (10MHz/1)/(3+1) = 2.50 MHz for an ATMEL 25256A EEPROM and 25F1024A Flash by the SPI bootload process.

For the ATMEL AT25HP256/512, the LOSPCP register value in fmt_2834x.bat file should be modified as follows:

REM Specify the LOSPCP register value for the SPI bootloader
SET LOSPCP_REG=0x01

In this case, SPI boot loader changes the SPICLK rate based on the boot loader header information received. The LOSPCP is set to 0x01 (LSPCLK is divide by 2), and SPIBRR is set to 0x7. Hence, the SPI clock rate is changed to: (10MHz/2)/(3+1) = 1.25 MHz for an ATMEL AT25HP256/512 EEPROM by the SPI bootload process.

6.2. Invoke the batch file.

When invoked, the batch file will process the specified out file in two steps. During the 2nd step you will be prompted to make a simple edit to one of the files. This section describes the steps and the modification you will be asked to make.

Batch File Step 1: Run hex2000 to add the boot loader header information to each section.

The boot header information is required for the 2834x SPI boot loader to properly download the application into the device for execution. This header information is added to the file by converting the .out file to ASCII-HEX using the C2000 hex utility that is included with the codegen tools in Code Composer Studio. The example below shows how the file *sourcefile.out* would be converted to the ASCII-HEX file *spi_rom_out.asc*:

```
hex2000 –a –boot –spi8 –lospcp %LOSPCP_REG –spibrr SPIBRR_REG –o
spi_rom_2834x.asc spi_source.out
```

where:

-a	Create an ASCII-Hex file.
-boot	Use boot header format.
-spi8	Use the SPI 8-bit boot header format
-lospcp	Specify the initial value for the LOSPCP register. This value should be selected based on the system's specific requirements the EEPROM/Flash device being used.
-spibrr	Specify the initial value for the SPIBRR register. This value should be selected based on the system's specific requirements the EEPROM/Flash device being used.
-e	Specify the code entry point to begin execution after boot loading.

Refer to *TMS320C2834x Boot ROM Reference Guide* (sprufn5) for more information on the SPI-boot header requirements and hex2000 options.

Note that while the code entry point for the application can be passed to the hex2000 utility by using the –e option, it is normally included as part of the .out file built by Code Composer Studio. For example, by default for a C program the entry point defined in the .out file is the C-initialization routine _c_int00. If required, during compile time the linker's –e option can be used to manually specify the application's entry point. The –e option allows you to manually specify a symbol that corresponds to the start of the section which should be the entry point. This information is then included in the SPI-boot header and used by the boot loader to transfer control to the application that has been downloaded into the device through the SPI module.

Batch File Step 2: Create a single-section COFF file to program into the EEPROM.

The ASCII-HEX file is then converted to a single section COFF file that will be sequentially programmed into the Serial EEPROM/Flash device. The batch file will open the ASCII-Hex file created by the hex2000 utility in the previous step in Notepad. You will be prompted to edit the ASCII-HEX file to include a base address

required by the `Asciihexlib.exe` utility from Spectrum Digital (duplicated in the *<SDFlash base>\myprojects\tif2834x_eeprogflash_v1* directory). Before you edit the file, the top of the .asc file will look similar to the following:

```
€
AA 08 00 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E0 00 04
00 00 00 00 00 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 00 09 00
. . .
```

To add the base address, after the first non-printable character, add the following:

```
<enter>$A0000,
```

Where <enter> is the enter key on your keyboard. The file will now look like:

```
€
$A0000,
AA 08 00 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E0 00 04
00 00 00 00 00 00 01 00 02 00 03 00 04 00 05 00 06 00 07 00 08 00 09 00
. . .
```

Be sure to include all 4 zeros and the comma. There is no return after the comma.

Save the file once you are finished editing it and close Notepad.

The batch file will then automatically run the Spectrum Digital Axiihexlib.exe utility to convert the ASCII-HEX file to a single-section COFF file.

This can also be done manually as follows:

```
Asciihexlib.exe -hex spi_rom_2834x.asc -out spi_rom_2834x.out
```

7. Modifying the Algorithm Files for Other Serial EEPROM/Flash Devices

As-is, the SDFlash2834x EEPROM/Flash Programmer Utility algorithm files support the ATMEL AT25256A, AT25H256, AT25H512 EEPROMs and AT25F1024A Flash.

To modify the algorithm files for other serial EEPROM/Flash devices, the following files must be changed:

1. *SDFlash28x_SPI_Flashprog_Wrapper.h* / *SDFlash28x_SPI_Eeprog_Wrapper.h*
   - ❑ Change the PLLCR_VALUE and DIVSEL_VALUE defines for the desired SYSCLKOUT frequency for your device.
2. *SPIFLASH28_SPI_Flashprog_Library.h / SPIFLASH28_SPI_Flashprog_Library.h*
   - ❑ Change the LSPCLK_DIV and SPI_BAUD defines for your desired low-speed clock divider and SPI baud rate, respectively.
   - ❑ Update the CPU_RATE define with the CPU clock speed represented by 1/SYSCLKOUT.
   - ❑ Notice that there is a #define specific to the exact EEPROM or Flash device. Then the associated #if and #endif combination for the EEPROM/Flash device surrounds the serial commands, chip select timings, and memory sector/block/page sizes specific to that particular chip. For a non-supported memory chip, these parameters will need to change (and new parameters may need to be added).
3. *SDFlash28x_SPI_Flash_Wrapper.c/ SDFlash28x_SPI_EEPROM_Wrapper.c*
   - ❑ Change this file only if new SPI EEPROM/Flash library functions have been added to SPIFLASH28_SPI_Flashprog_Library.c / SPIEEPROM28_SPI_Eeprog_Library.c and the new functions need to be called within the SDFlash programming functions.

4. *SPIEEPROM28_SPI_Eeprog_Library.c / SPIFLASH28_SPI_Flashprog_Library.c*
   - This file includes all the functions which read/write/program/erase the SPI EEPROM or flash. Make changes to this function if your particular chip sends commands, writes to, programs, or erases the EEPROM/Flash in a different order or manner from the provided functions, or add new functions here.
   - Note: You can use the functions available in this file to manually read from/write to the SPI EEPROM/Flash separate from SDFlash also.