Danville Signal Processing, Inc.

dspFlash™

SHARC/Blackfin Flash Programmer

User Manual

Version 1.10

Danville Signal Processing, Inc.
dspFlash™
User Manual

Danville Signal Processing, Inc. strives to deliver the best product to our customers. As part of this goal, we are constantly trying to improve our products. Danville Signal Processing, Inc., therefore, reserves the right to make changes to product specification or documentation without prior notice. Updated operating manuals and product specification sheets are available at our website for downloading.

This manual may contain errors, omissions or “typo’s”. Please send your comments, suggestions and corrections to:

Danville Signal Processing, Inc.
38570 100th Avenue
Cannon Falls, MN 55009-5534

Trademark Notice

Contact Information

Danville Signal Processing, Inc.
38570 100th Avenue
Cannon Falls, MN 55009

E-mail:	dsp@danvillesignal.com
Web Site:	http://www.danvillesignal.com
Voice:	507-263-5854
Fax:	877-230-5629

Table of Contents

Overview of dspFlash™

Danville Signal's dspFlash™ is an onboard programming tool for both serial and parallel flash memory. These flash devices are programmed through a JTAG interface, using proprietary JTAG commands similar to an emulator. This programming method uses the target processor to program its own flash rather than relying on public, but very slow, boundary scan commands.

The dspFlash has 64Mb of SDRAM to hold the file to be loaded into flash. The SDRAM needs to be uploaded to only once per session. The JTAG interface of the dspFlash programming module is hot swappable which allows for quick transition from board to board. These two features can greatly reduce programming time during production runs.

The dspFlash uses a supporting stand-alone Windows application. This program is regularly updated and may be downloaded from the Danville Signal website. The Windows application provides an easy-to-use graphical environment aimed at quick and simple production programming. The program also embeds all the device drivers, which allows for new devices to be added quickly. Should the need arise; the dspFlash programmer can even reprogram itself.

The dspFlash programming process requires very little knowledge of the target processor and basically reduces the flash programming process to the push of a button. The most demanding tasks to program a flash during production are reduced to: opening a previously saved configuration file, plugging a JTAG cable in properly, and possibly resetting the target processor in the event of an error.

To actually configure the dspFlash you need to know the SPI or parallel flash configuration on the target board. This includes the Chip Select configuration, target processor, and target flash models. The dspFlash Windows application cannot generate a LDR or binary file directly, but uses LDR or binary files that are created using VisualDSP++ or other development environments.

If you do not have a background with these skills, you may want to check out the Danville Signal website, as well as the Analog Devices website for links to useful references. Danville engineers are also available to discuss your application.

Getting Started

The dspFlash includes the following items:

Hardware:

- dspFlash Programmer
- USB Cable
- 2 dspFlash JTAG cables

Documents (CD):

- This Manual
- Windows Application Installer
- USB Drivers
- Microsoft .NET Framework Version 2.0 redistributable (required for dspFlash application)

We recommend that you have the following tools:

- Analog Devices VisualDSP++ 5.0 for SHARC or Blackfin
- Analog Device HP USB or USB ICE
- Any necessary JTAG adapters

You can purchase any of these tools from Danville Signal by visiting our website, where you can also find helpful links to other tools and documents.

The dspFlash application program requires Windows 2000, XP or Vista. You should have the latest service packs installed.

First Steps - Installation

Once you have received the dspFlash, the first thing to do is to plug it into the PC via the included USB cable. This will begin the driver installation process. Follow the Windows prompts and browse for the driver files on this CD located in the usbdrivers folder.

Once the drivers are installed, you can start the dspFlash programming application, which is also included on this CD. You can copy the executable file onto the hard drive of your pc and launch it. The programming application should automatically detect which COM port the dspFlash programmer is connected to and connect to it.

Connecting to the target is as simple as plugging in the JTAG cable. The target device can be either powered or unpowered when the cable is plugged in. Once your target is powered, the LED indicating the logic voltage level should light. If this does not light, you should refer to Analog Devices application note EE-68 which describes the proper JTAG connections, specifically pulling pin 5 high via a pull-up resistor. If this resistor is not present the voltage level of the target is assumed to be 3.3V. Configurations using lower logic level voltages without this pull-up resistor are not supported by the dspFlash.

Target Hardware Requirements

JTAG Requirements

The JTAG connections to your target processor are assumed to comply with Analog Devices Application Note EE-68. The pin connections are required to be as follows.

Ground	1	2	EMU
Key, This pin should be removed	3	4	Ground
Pulled to Vddio through a 4.7KO resistor[1]	5	6	TMS
Ground[1]	7	8	TCK
Ground[1]	9	10	TRST
Ground[1]	11	12	TDI
Ground	13	14	TDO

Note [1]: If you have wired your JTAG for a "Local Boundary Scan Controller" then you may have different connections for these pins. If you have wired for a local boundary scan controller then you will need an adapter on the target side to compensate for this.

It is also important to note that the 4.7KO pull up resistor on Pin 5 is required for applications using logic level voltages other than 3.3 Volts. If you do not have this pull-up resistor and are using a logic level other than 3.3 Volts your configuration will not be supported. Any other exception can be worked around, this one cannot.

Please note that the "Autostart on Device Detection" utilizes the pull-up resistor on pin 5 as well. This is essentially how the dspFlash can detect when a new device has been connected. You cannot use the Autostart feature without this configuration.

Flash Connection Requirements

The dspFlash also has certain requirements for the target flash connections. When you select your target processor you will notice that the rest of the blanks automatically fill in with values. These default settings are the boot configuration of your processor.

You can select your chip select and for most processors the list of available chip selects is comprehensive, however the rest of the SPI lines are required to be in a bootable configuration. What this really means is that the MOSI, MISO, and SCK lines are all assumed to be in a bootable configuration.

The parallel ports are assumed to use one of the applicable memory selects. In the case of the SHARC ADSP-21262 or the ADSP-21364 families, the base address of the flash must be specified. The base address is specified for other processors but is calculated based upon the memory select you are using.

Operation

Connecting To The dspFlash

For version 1.1 and later
Your dspFlash should connect automatically during the startup of the Windows application. If this fails, you can try it again by clicking "Connect to dspFlash" on the tools menu. If you have multiple devices connected, you can connect to one at a time, a good way to accomplish this is to connect to one and then plug in the second and connect to it, then connect to a third etc.

For version 1.0
Your dspFlash should connect automatically during the startup of the Windows application. If this fails, you can try it again by clicking "Connect to dspFlash" on the tools menu. If you have multiple devices connected, you can manually connect by unchecking the Auto-Detect COM port and then choosing connect. This will bring up a box that will allow you to choose which COM port you are connected to.

To determine which COM port the dspFlash is connected to, you can open the control panel, and then the System properties. In system properties, the Hardware tab has a device manager button. If you find the ports category and expand it, you will see a Danville Signal Flash Programmer followed by a (COMXX). This COM number will correspond to your port.

Setting Up Your Target

Setting up a connection on the dspFlash is simple.

Step 1: Select the target processor. To do this, select the processor on the target board from the pull down box.
Step 2: Select your flash configuration, at the current time both SPI and parallel flashes are supported. If you are using a SPI flash the next step is to specify your chip select line by choosing it from the CS pull-down box.

If you are using a parallel flash you have to specify either the memory select line that the flash is connected to or the base address. The base address is only user supplied for the ADSP-21262, and ADSP-21364 families and is used to signal the location of the flash to both the dspFlash programming module and to the target.

Step 3: Select the manufacturer of the target flash to be programmed, then the model of the flash. Then select the type of your file, whether it is Intel Hex, ADI ASCII, or binary. Many file types can be written to flash by selecting the binary file type and they will be written byte for byte into flash. If you are programming an audio file, a video file, or a binary loader file you can do it with this file type.

Choosing and Uploading a file

With the dspFlash you can write many types of files into flash including LDR files which can be used to boot the processor. There are specific configurations you must adhere to write bootable code to the flash. For example, if you are using a Blackfin BF-532 and a SPI flash, you have to have a CS of PF2, and write with an offset of zero.

Bootable code must be written with an offset address of zero. If you are using a SPI flash and a processor, you also must bit reverse your data file. This is done because of differences in the booting SPI configuration and normal SPI configurations. This bit reversing can be done by checking the *Bit Reverse Data File* box under file type. If you have already uploaded your code and must change this setting, it is necessary to upload the file again.
If your file is not intended to boot your processor your options are much more flexible. There are many more chip select lines to choose from, and you can make use of loading with offsets. These files should be written in binary format and will be imaged bit for bit into flash.

Once you have chosen your data file and bit reversed it if need be, the next step is to upload. Uploading your data file to the dspFlash programmer will send the configuration, processor, and flash to the dspFlash. This is important because if any of these things are changed, uploading will need to be repeated. Once your file has been uploaded then you are ready to perform flash operations.

Writing a File

There are two options to consider when writing a file to flash. The first of these is the offset address. This will dictate to the dspFlash where to start the file writing process. This address will need to be zero for booting cases, and can be nonzero for non-booting applications. The average write speed of the dspFlash is 33kB/s; this may vary depending on your processor and flash.

The other option to consider while writing files to flash is the preprogram erase operation. This operation has three possible settings: erase affected, erase all, and no erase. If you choose to erase affected it will only erase the sectors that will be affected by the writing operation. If you choose erase all, the entire chip will be erased prior to writing, and if you choose no erase, no erase will take place prior to the writing.

Due to the huge erase times of large flashes, which can be greater than 3 minutes, erase affected can often be more efficient. If your file is large enough to occupy 85-90% of the flash you are using, then erase all may be a better option. If the flash on your target board is blank then no erase is definitely the best option, because it adds no extra time to your operation.

During a production run, the write file button can be the only button to click to trigger the programming of another board, assuming the target board is the same as the previous board. This can save valuable time during a long production run. If you have the pull up resistor on pin 5 of JTAG, then you can save even more time by clicking the *Auto start on Device Detection* checkbox and then when a new device is detected the write file programming sequence will be triggered. This can save a significant amount of time and can keep an operator from making unnecessary repetitive motions to click a button over and over again.

Auxiliary Erase Operations

If you do not need to write an entire file you can still perform individual erase operations. When erasing there are two options, erase chip and erase sector. Erasing the chip will take a noticeable amount time, but will erase an entire chip. The average erase time is 7.5 seconds per Mb. If you have a 64Mb flash, then your erase time could top 3 minutes. This is the kind of operation that is limited by the flash and cannot be sped up by any programmer.

A more efficient way to erase is to erase single sectors. Like the erase affected option, the erase sector operation will only erase a predefined portion of the flash. These sectors are defined in the datasheet of your flash device and will vary greatly from model to model. To select which sector you want to erase, type an address into the starting address box in the Auxiliary operations box. If your flash has multiple sector erase commands, the command for the smallest block will be used to give maximum erase flexibility.

Auxiliary Write Operations

The auxiliary write operation can be used for many things. You can write individual values or you may write ASCII strings. For example, you could use it to write copyright notices into flash using the write string operation. You could also use it to write a serial number or calibration values.

In order to write values, you have to specify three things:

1) The starting address in flash where the writing begins.
2) Select the size of the word (8, 16 or 32 bits).
3) Specify the data to be written. This data is specified in hexadecimal characters.

Strings are written in a similar fashion. You specify the starting address and string radio button. The string can be up to 64 characters long. It is automatically terminated with a null character (0x00).

Auxiliary Read Operations

There are four options for auxiliary read operations, which can be used to read the contents of your flash. You can read one, two, or four bytes, or you can read an ASCII string. The difference here is that if you read an ASCII string the values will be decoded and the read will stop at the terminating null character, or if this character is not found, a read of 64 characters will be performed.

The read operation is performed as follows; first specify the starting address in the start address box, second select the read you want, and then perform the read.

Tools

Setting up a Connection

dspFlash should automatically detect the COM port number and settings in most cases. If this does not work for you then you may have to manually configure which COM port you are connected to. To do this you have to turn off auto detect, and click connect again, it will then bring up a window where you will have the option to choose which COM port you are connected to.

Versions 1.1 and later make use of a different driver which does not utilize the virtual COM port so determining a COM port is no longer necessary. The difference is you will be unable to connect to a dspFlash with more than one unconnected dspFlash connected to your system.

See Connecting to the dspFlash on page 5.

Upgrading Firmware

With every application update there may be a firmware upgrade. The process of upgrading the firmware of the dspFlash is simple; just click the Upgrade Firmware button on the Tools Menu. You will be asked to confirm firmware upgrade, after it is completed, you will need to restart both the application and the dspFlash programming module. When you restart the application you should see your new firmware revision in the message center.

SDRAM Test

Testing the SDRAM of your dspFlash programming module may occasionally be necessary. It would also be recommended if you start to see problems across multiple boards. To execute the test click the SDRAM test under the tools menu. If there is a hardware problem, call or e-mail Danville Signal support to discuss options.

JTAG Chaining

Please note: Only JTAG devices with 5 bit instruction registers are currently supported; this includes all ADI Sharcs, Blackfins, and TigerSharcs. Look for more widespread JTAG chained device support in future releases.

JTAG chaining gives a hardware designer the ability to control multiple JTAG devices through a single JTAG connector on the board. The implementation of JTAG chaining is presumed to comply with both Analog Devices application note EE-68 and the IEEE 1149.1 JTAG boundary scan standard.

When you enable JTAG chaining two boxes will appear next to your select processor box. These boxes are JTAG chain length, and JTAG chain position. The JTAG chain length counts from one and represents the number of JTAG devices in the scan chain.

The other box is the JTAG chain position. The device, which is connected to the TDO of the JTAG connector, is device zero. The device, which is connected to TDI of device zero, is device one. If you follow this chain you should be able to determine which device is which. If you have four devices in you JTAG scan chain then the last device in your scan chain is device three.

Advanced Options

These advanced options are not intended for every user. Most users will not need to make use of these options.

Core Clock Modification

This feature will not be available upon initial release but will be implemented on future releases.

To increase core clock speed, you can adjust the core clock registers from the advanced menu**. Note: This procedure should be done with caution. Improper settings in the core registers can cause the target DSP to meltdown.**

If you know the values you would like to write to the registers, you can uncheck the use default values box, and then input the applicable values for the registers for your processor. Since these values are dependent on the oscillator you are using, predefined register values will be an unlikely upgrade in the future.

Connection Register Modification

Possibly the best way to improve performance is to adjust the values written to the connection registers, either the SPI or parallel controls. In order to change these register values, you have to click the use default values checkbox, which will allow editing of the registers. The values you choose will be used in place of the default register values we used for testing at Danville Signal. These values generally use a 10MHz SPI clock frequency, or a medium number of wait states on a parallel connection.   This was written to try to compromise widespread device support with the same values while still maintaining an acceptable level of performance.

Please refer to your target processors hardware reference manual available from Analog Devices for register definitions.

Danville Signal JTAG Adapters

You may notice that the Danville Signal JTAG adapter is available for purchase along with your dspFlash programmer. On our boards we use a male 16 pin, 2mm dual row header as our JTAG connection. This is smaller than the connectors used by Analog Devices. We offer an adapter to convert this 2mm header to the standard 0.100 headers used by the Analog Devices emulators as well as the dspFlash JTAG connection. If you want to use the Danville JTAG adapter, the connections are exactly the same as the larger ADI connector with the exception of Pins 15 & 16. These pins do not exist on the ADI JTAG connector and are tied to Vd+3.3 (IO Voltage) on Danville boards.

Troubleshooting and FAQs

When programming flash a number of problems can occur. A brief discussion of common problems can be found below. If you have a problem not listed here feel free to contact Danville Signal at support@danvillesignal.com.

Problem: I can't seem to connect to the dspFlash programming module

Solution: First try power cycling the dspFlash, simply unplug and plug it in again. Restart the application and see if that solves the problem. If not, is the status light illuminated on the dspFlash? If the status light is not illuminated, there is possibly a problem with the driver. Reinstall the driver provided on the CD. If it still doesn't work, contact Danville Signal.

Problem: I wrote a working LDR file to a target but it doesn't boot properly

Solution: The most likely cause of this is writing the file improperly. If you are using a processor with an SPI flash then bit reversing is necessary. The other possible programming problem would be if the file were written with an offset. Writing with offset is not a bootable configuration.

The other cause of this could be hardware. If your hardware is not configured properly booting will not take place. Most processors have specific requirements regarding which chip selects can be used, and how the flash is connected. In a parallel setup a certain memory select must be used.

Problem: I wrote something to the flash and now I can't connect to my target, even with an emulator

Solution: This probably happened because what was written to flash caused a processor core fault which results from bad code in the boot area of the flash. This can be remedied by changing the boot configuration pins of your processor. If these pins are tied directly to a supply and not through a pull-up or pull-down resistor, then this method probably will not be possible.

This can also be remedied with an emulator and exceptional timing. The trick to fixing the problem this way is to connect to your device between power-up and boot. This is possible on most processors, however can be quite tricky. Once the processor is booted and not faulted you should be able to erase the flash either with the emulator or with the dspFlash. If you are going to switch from the emulator to the dspFlash, you should be careful not to power cycle or reset the device or you will have to start the process over again.

If neither of the other options worked it may be necessary to remove the flash and erase it by some other method.

RoHS Compliance

The European Union approved a directive on the restriction of the use of certain hazardous substances in electrical and electronic equipment. This directive is commonly known as RoHS, EU Directive 2002/95/EC. This directive severely limits the amount of lead and 5 other substances that can be in contained in nonexempt products. The directive became European law in February 2003 and took effect July 1, 2006.

The dspFlash is RoHS compliant.

Product Warranty

Danville Signal Processing, Inc. products carry the following warranty:

Danville Signal Processing products are warranted against defects in materials and workmanship. If Danville Signal Processing receives notice of such defects during the warranty period, Danville Signal Processing shall, at its option, either repair or replace hardware products which prove to be defective.

Danville Signal Processing software and firmware products which are designated by Danville Signal Processing for use with our hardware products are warranted not to fail to execute their programming instructions due to defects in materials and workmanship. If Danville Signal Processing receives notice of such defects during the warranty period, Danville Signal Processing shall, at its option, either repair or replace software media or firmware which do not execute their programming instructions due to such defects. Danville Signal Processing does not warrant that operation of the software, firmware, or hardware shall be uninterrupted or error free.

The warranty period for each product is one year from date of installation.

Limitation of Warranty:

The forgoing warranty shall not apply to defects resulting from:

- Improper or inadequate maintenance by the Buyer;
- Buyer-supplied software or interfacing;
- Unauthorized modification or misuse;
- Operation outside the environmental specification of the product;
- Improper site preparation and maintenance.

Exclusive Remedies:

The remedies provided herein are the Buyer's sole and exclusive remedies. In no event shall Danville Signal Processing, Inc. be liable for direct, indirect, special, incidental or consequential damages (including loss of profits) whether based on contract, tort, or any other legal theory.