Application Report
SPRABE8–August 2010

Using Triple Read and Boost Algorithm Code to Enhance Adequate Programming Margin on TMS320F206

Tai Nguyen

ABSTRACT

This addendum to the *TMS320F20x/F24x DSP Embedded Flash Memory Technical Reference* (SPRU282) describes the triple read and boost operation of the embedded Flash EEPROM module on the TMS320F20x digital signal processor (DSP) devices to enhance the programming margin on all bits and ensure that programmed bits are not affected by programming other bits. It also provides sample code that you can use in developing your own software. The performance specifications of the embedded Flash memory of the TMS320F206 have been evaluated using the algorithms and techniques described in this document. TI does not recommend deviation from these algorithms and techniques, since doing so could affect device performance. The addendum does not describe the use of any specific Flash programming tool nor does it describe the external interface to the DSP.

Note that this addendum is not applicable to the TMS320F24x™ DSP devices.

Project collateral and source code discussed in this application report can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

Contents

1 How to Use This Manual

There are several stand-alone Flash programming tools for the TMS320F20x/F24x generation of DSPs. Using one of these stand-alone tools with the TMS320F20x/F24x requires only a basic understanding of the Flash operations.

NOTE: It is assumed that the reader of this addendum is familiar with F206 Flash programming and the *TMS320F20x/F24x DSP Embedded Flash Memory Technical Reference* (SPRU282).

2 Introduction

As you modify the contents of the F206 Flash array using your proprietary or TI-provided programming algorithm, it is possible for programmed bits to marginally affected by the programming operation on adjacent bits; this addendum addresses how to detect potential program disturbances and enhance the programming process to ensure that each bit has adequate margin when fully programmed.

Section 3 and Section 4 introduces the operations performed by the two algorithms, *Triple Read* and *Boost*, and points out where to access the code examples.

Section 5 briefly discusses how the erase pulse can be shortened as the manufacturing process of the F206 has matured.

3 Triple Read

Background: As you program bits up/down the array, the programming operation can potentially cause drift on bits already programmed on the same bit line. For example, assume that you are currently programming a given bit in row 0, which has previously passed a *Verify that it is zero* (Ver0 read) test. If you were to re-run this same Ver0 test after programming additional bits in that same column, it may no longer pass the Ver0 read test. Because the embedded Flash EEPROM module on the TMS320F20x consists of high-density bit cells, there is an inherent variability in how programming stress may affect other bits in the array. Additionally, the number of bits programmed on a given line in the array may affect whether, and the extent to which, previously programmed bits may be stressed. Given that there are 512 bits per bit line, there exists the possibility of between 0 and 511 bits in the line being affected. Even though the probability of such an occurrence is low, it is not zero.

The Ver0 read mode is a margined read. When a bit passes Ver0 read it ensures that there is adequate margin on the programmed bit. The use of 6.5 V for Ver0 was chosen with an expectation that stressed programmed bits would remain above VCC, and thereby continue to operate correctly within the array. There is the potential for programmed bits to be stressed by more than 0.5 V, which would result in those bits not passing Ver0 read back, but still operating correctly. Accordingly, Ver0 cannot be used to validate whether the array is still operating correctly.

The Triple Read algorithm checks three things:

- Are the erased bits ("1" bits) still erased adequately to operate across different speeds? This can be addressed with the Fast Read (single TBLR *). An error in the Fast Read indicates the bits are not erased adequately.
- Are the programmed bits ("0" bits) adequately programmed? This can be addressed with the Slow Read (three TBLR *). An error in the Slow Read indicates that one or more bits in the array do not have expected margin.
- Is there enough margin in both the erasure and programmation to operate at full speed with the CPU? This can be addressed with the XOR Read (Memory Contents XOR Memory Address). An error in the XOR Read indicates that the interaction between adjacent bit cells being different polarity may be causing speed issues when the CPU exercises worst case instruction sequencing.

Each read can be performed on any memory block and an associated checksum is calculated to determine PASS or FAIL. The Triple Read algorithm can be used to do a loop test while varying the F206 core voltage, if necessary.

The Triple Read program can also be used to check the erased, cleared, or checker board programmed devices.

NOTE: The device should be checked using this algorithm each time if it is programmed in the field.

3.1 How to Use the Triple Read Program

This program can be compiled/assembled and used as a standalone test or used by the *C2xx_brx.asm* file.

- For the standalone case, use the read_stdalone.cmd to compile and link. Note that when using this mode, the compile variable,PRG2xx, must be set to 0; additionally, you should set the CNF to 1 by using Code Composer Studio™ software before code is downloaded onto the target board, since the code is downloaded on the on-chip B0 block.
- When using the control module C2xx_brx.asm and the utility code Sutils20.asm, use the C2xx_brx(_CCS).cmd to compile and link. Note that when using this mode, the compile variable,PRG2xx, must be set to 1. The PRG_2XX(W).exe automatically sets the CNF bit to 1 before downloading the C2xx_bbx.out onto the F206.

3.2 Triple Read Program

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

3.3 Linker Command File for Triple Read Program in stand_alone Mode (read_stdalone.cmd)

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

3.4 Linker Command File for Triple Read Program in Control Mode

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

```
/*  Linker command file to specify c2xxprog sections          */
/*  and define PRG2xxw declarations                           */
/*  Locates program section in B0 ram and uses B1 data buffer */
/*  for Flash 0 array programming                             */
/*  Rev : 1.10                                                */
/*  Modified by : Sam Saba, TI Houston             12/24/96   */
-e PRG_init
-o c2xx_brX.out
-m c2xx_brX.map

MEMORY
{
  PAGE 0 : PROG:   origin = 0xfe00, length  = 0x100
  PAGE 0 : SPROG:  origin = 0x8000, length  = 0x400 /* 1kx16 for algorithm*/

  PAGE 1 : VARS:   origin = 0x0300, length  = 0x10
  PAGE 1 : DATA:   origin = 0x0320, length  = 0xd0
  PAGE 1 : SDATA:  origin = 0xc00,  length  = 0xc00 /* 3Kx16 up to 0x1800 */
  PAGE 1 : DSPAD1: origin = 0x310,  length  = 0x10

}


SECTIONS
{
  PRG_parm  : {} > DSPAD1 PAGE 1
  PRG_data  : {} > DATA PAGE 1
  ary_var   : {} > 0x300 PAGE 1
  PRG_text  : {} > PROG PAGE 0
  boost     : {} > PROG PAGE 0
  DLY       : {} > PROG PAGE 0
  REG       : {} > PROG PAGE 0
  ARY       : {} > PROG PAGE 0

}
```

3.5 Project File Example for Triple Read "test_triple_rd.pjt"

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

4 Boost

Background: As discussed in Section 3, some bits might be affected as one programs bits up/down the array. This phenomenon can be detected by using the Triple Read algorithm; if it is detected, programming margins can be enhanced by using the Boost algorithm.

The Boost algorithm finds the bits that have drifted in value and applies another programming pulse to those bits to boost their margin. Since the Boost algorithm is only applying the boost to a few bits across the array, the likelihood of stressing other bits is minimal.

It is possible to execute a second pass of the algorithm that programmed the pattern into the array instead of the Boost Algorithm. However, this is a much slower process and requires a lot more code. Additionally, it is more difficult to execute remotely.

An error in the Slow Read indicates that one or more bits in the array do not have enough margin.

An error in the Fast Read indicates the bits are not erased adequately.

An error in the XOR Read indicates that the interaction between adjacent bit cells being different polarity is causing speed issues when the CPU exercises worst case instruction sequencing.

Most of the time this can be resolved executing a Clear, Erase, Program and Boost sequence. There should be no issues with Fast or XOR Reads if your application is running at ~20 MHz. If it were running at a frequency higher than ~20 MHz, then the Fast and XOR Read errors may start to show up. If you see a part that fails the XOR Read, but passes the Fast and Slow Reads, it might indicate a slow path between the CPU and the Flash and that the part should be replaced.

4.1 How to Use the Boost Program

This function can be compiled/assembled and used as a standalone test or used by the *C2xx_bbx.asm* file.

- For the standalone case, use the boost_stdalone.cmd to compile and link. Note that when using this mode, the compile variable,PRG2xx, must be set to 0; additionally, you should set the CNF to 1 (by using the Code Composer Studio software) before code is downloaded onto the target board, since the code is downloaded on the on-chip B0 block.
- When using with the control module C2xx_bbx.asm and the utility code Sutils20.asm, use the C2xx_bbx(_CCS).cmd to compile and link. Note that when using this mode, the compile variable,PRG2xx, must be set to 1. The PRG_2XX(W).EXE automatically sets the CNF bit to 1 before downloading the C2xx_bbx.out onto the F206.

Within the Boost Program, a compile time variable, DBUG, is available that provides the following capabilities:

- DBUG = 0 to turn debug off, no map
- DBUG = 1 to map the weak programming bits and no boost
- DBUG = 2 to map the weak programming bits and boost them

4.2 Boost Code

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

4.3 Linker Command Example File Used the Boost Code in stand-alone Mode (boost_stdalone.cmd)

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

```
/* F206 BOOST */
/*
-c
-e GBST
-m BOOST.map
-o BOOST.out
*/
/* BOOST206.obj */

-e START

MEMORY
{
     PAGE 0 : TB0:     origin = 0h,     length = 0F0h
          .PSA:        origin = 0F0h,   length = 0Fh
           B0:         origin = 0FE00h, length = 0100h
           FLASH0:     origin = 0100h,  length = 04F00h
           FLASH1:     origin = 05000h, length = 05000h

     PAGE 1 : RAMB2:   origin = 0060h,  length = 0020h
           RAMB0:      origin = 0200h,  length = 0100h
           RAMB1:      origin = 0300h,  length = 0100h
           FLASH2:     origin = 01000h, length = 01000h

}

SECTIONS
```

```
{
     .text > B0 PAGE0
     .bss  > RAMB2 PAGE1
}
```

4.4 Linker Command Example File Used the Boost Code in Control Mode

This source code can be downloaded from the following URL: http://www-s.ti.com/sc/techlit/sprabe8.zip.

```
/*  Linker command file to specify c2xxprog sections          */
/*  and define PRG2xxw declarations                           */
/*  Locates program section in B0 ram and uses B1 data buffer */
/*  for Flash 0 array programming                             */
/*  Rev : 1.10                                                */
-e  PRG_init
-o  c2xx_bbX.out
-m  c2xx_bbX.map
/*
c2xx_bbX.obj
algos\sclr20.obj
algos\sutils20.obj
*/
MEMORY
{
  PAGE 0 : PROG:   origin = 0xfe00, length = 0x100
  PAGE 0 : SPROG:  origin = 0x8000, length = 0x400/* 1kx16 for algorithm*/

  PAGE 1 : VARS:   origin = 0x0300, length = 0x10
  PAGE 1 : DATA:   origin = 0x0320, length = 0xd0
  PAGE 1 : SDATA:  origin = 0xc00,  length = 0xc00 /* 3Kx16 up to 0x1800 */
  PAGE 1 : DSPAD1: origin = 0x310,  length = 0x10

}


SECTIONS
{
  PRG_parm  : {} > DSPAD1 PAGE 1
  PRG_data  : {} > DATA PAGE 1
  ary_var   : {} > 0x300 PAGE 1
  PRG_text  : {} > PROG PAGE 0
  boost     : {} > PROG PAGE 0
  DLY       : {} > PROG PAGE 0
  REG       : {} > PROG PAGE 0
  ARY       : {} > PROG PAGE 0

}
```

4.5 Project File Code

The following source code can be downloaded from the following URL:
http://www-s.ti.com/sc/techlit/sprabe8.zip.

- Project File Examples *test_boost.pjt*
- Utility Code *sutils20.asm*
- Variables Header File *fl_vars.h*
- Variables Header File *svar20.h*

5 Erase Pulse

It is suggested to replace the 7 ms erase pulse with one of 5 ms length, due to maturing of the manufacturing process. The current 7 ms erase pulse specification will not harm the device, but a 5 ms pulse is optimal given today's silicon.

6 References

- *TMS320F20x/F24x DSP Embedded Flash Memory Technical Reference* (SPRU282)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265