Freescale Semiconductor
Application Note

Document Number: AN3814
Rev. 0, 2/2009

Static Serial Bootloader for MC56F800x/801x/802x/803x

by: William Jiang
Applications, Asia Pacific Microcontroller Solutions Group
Petr Stancik
Technical Information Center, Global Sales & Marketing

1 Introduction

Freescale offers serial bootloaders for digital signal controllers. Because MC56F800x/801x/802x/803x devices have critical flash resources, these serial bootloaders are designed to load application code only once, so as to allow applications to use the full flash memory space. However, some customers require the serial bootloader to stay in flash forever. This serial bootloader is a so-called static bootloader. This document describes the static serial bootloader and its requirements when used in applications.

This document is based on the Freescale serial bootloader designed under CW8.x with Processor Expert. About 2 KB in size, it is located in program flash from address 0x7B00 to 0x7FFF on the MC56F802x/3x and 0x1B00 to 0x1FFF on the MC56F800x/1x.

It is a reusable static bootloader. That is, after a reset occurs, the bootloader starts to run. If a boot delays in seconds without receiving the application s-record, it

Contents

will jump to the programmed application code. If it receives the application s-record, it will program the application code to the on-chip flash.

The page erasing method is used by the static bootloader. The bootloader section cannot be overwritten by the application code.

2 Static Serial Bootloader Process

Figure 1 shows the main process of the static serial bootloader.

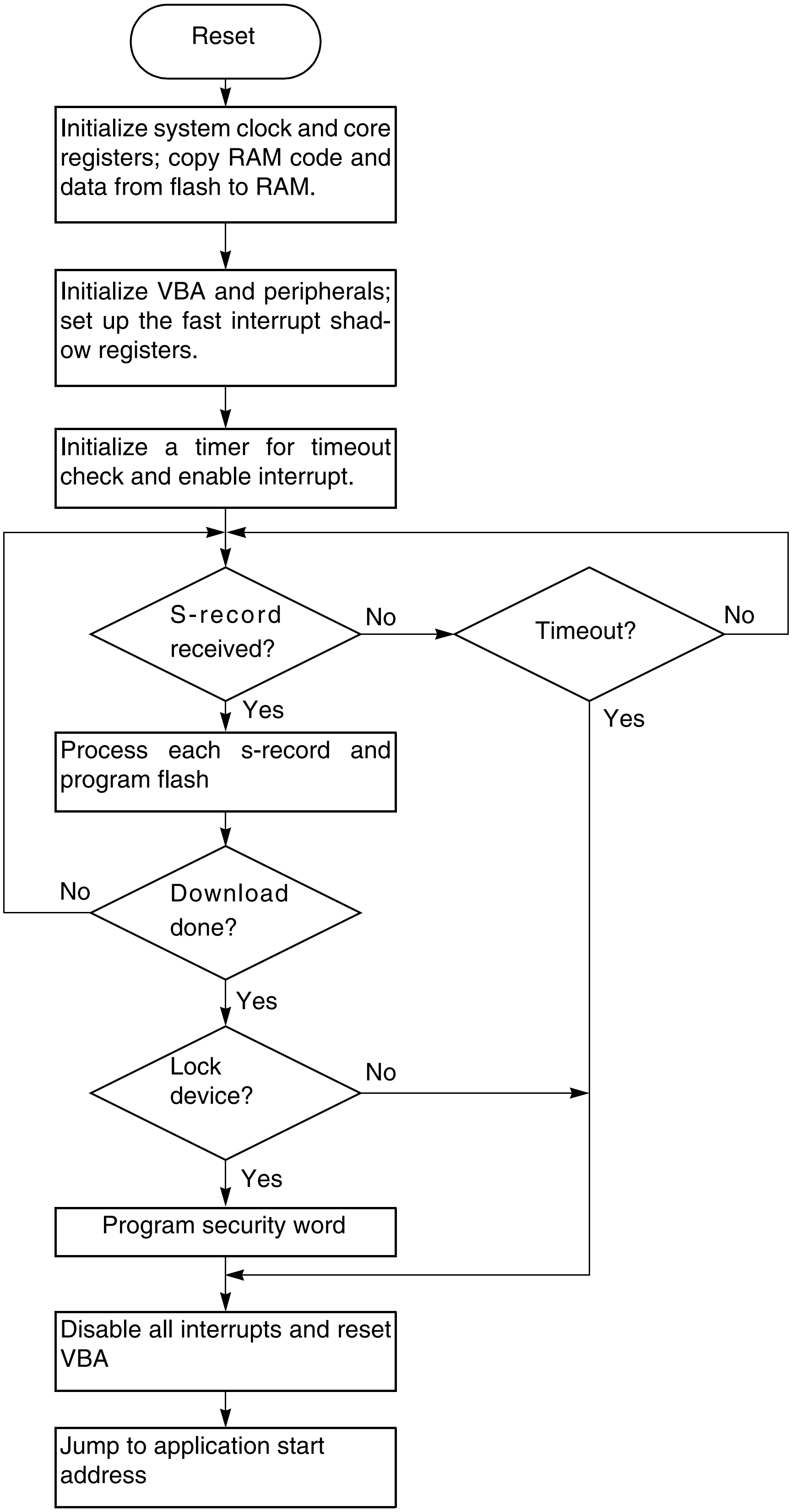

Reset
Initialize system clock and core registers; copy RAM code and data from flash to RAM.
Initialize VBA and peripherals; set up the fast interrupt shadow registers.
Initialize a timer for timeout check and enable interrupt.
S-record received?
No
Timeout?
No
Yes
Yes
Process each s-record and program flash
No
Download done?
Yes
Lock device?
No
Yes
Program security word
Disable all interrupts and reset VBA
Jump to application start address

Figure 1. Bootloader Process

3 Program Flash Usage

The flash usage is shown in Figure 2 and Figure 3. Please pay attention NOT to overwrite the memory range from 0x7B00 to 0x7FFF for MC56F802x/3x, or from 0x1B00 to 0x1FFF for MC56F800x/1x, because it is used by the bootloader.

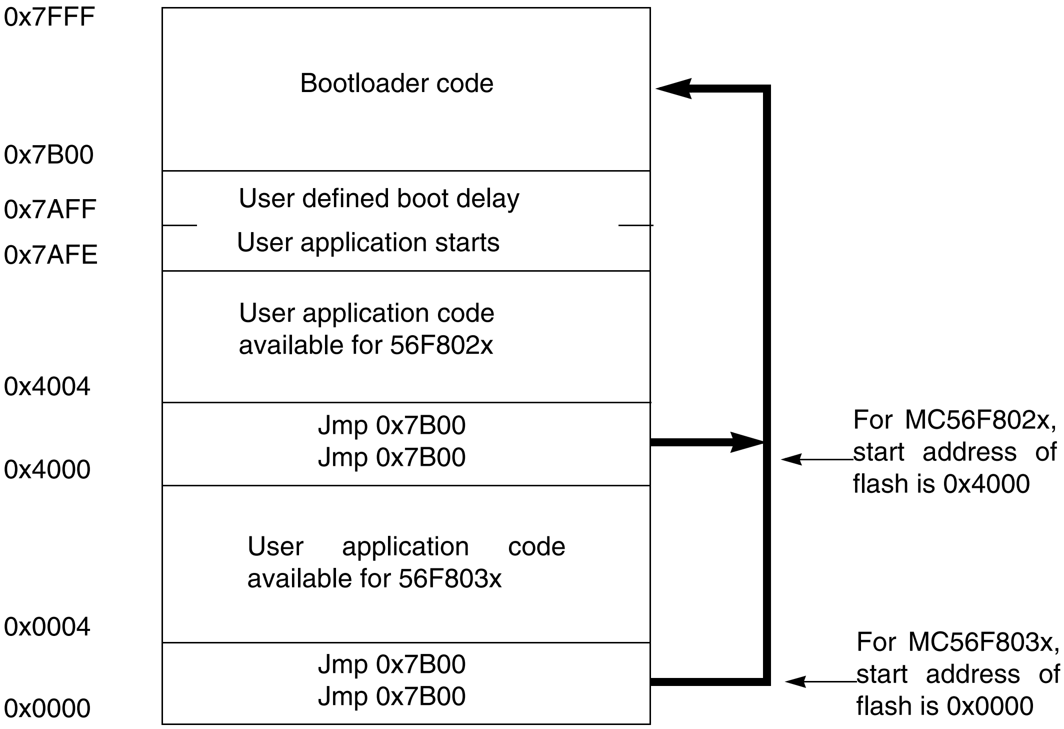

0x7FFF
Bootloader code
0x7B00
User defined boot delay
0x7AFF
User application starts
0x7AFE
User application code available for 56F802x
0x4004
Jmp 0x7B00
Jmp 0x7B00
0x4000
For MC56F802x, start address of flash is 0x4000
User application code available for 56F803x
0x0004
Jmp 0x7B00
Jmp 0x7B00
0x0000
For MC56F803x, start address of flash is 0x0000

Figure 2. Program Flash Usage for MC56F802x/3x Static Serial Bootloader

NOTE

For the MC56F802x, the lower memory area from 0x0000 to 0x3FFF is reserved. Both hardware reset and COP reset vectors are located at 0x4000 to 0x4003. Therefore, two “jump to bootloader” instructions must be placed at these locations. However, for the MC56F803x these locations can be used for user application code, because both reset vectors are located at 0x0000 to 0x0003.

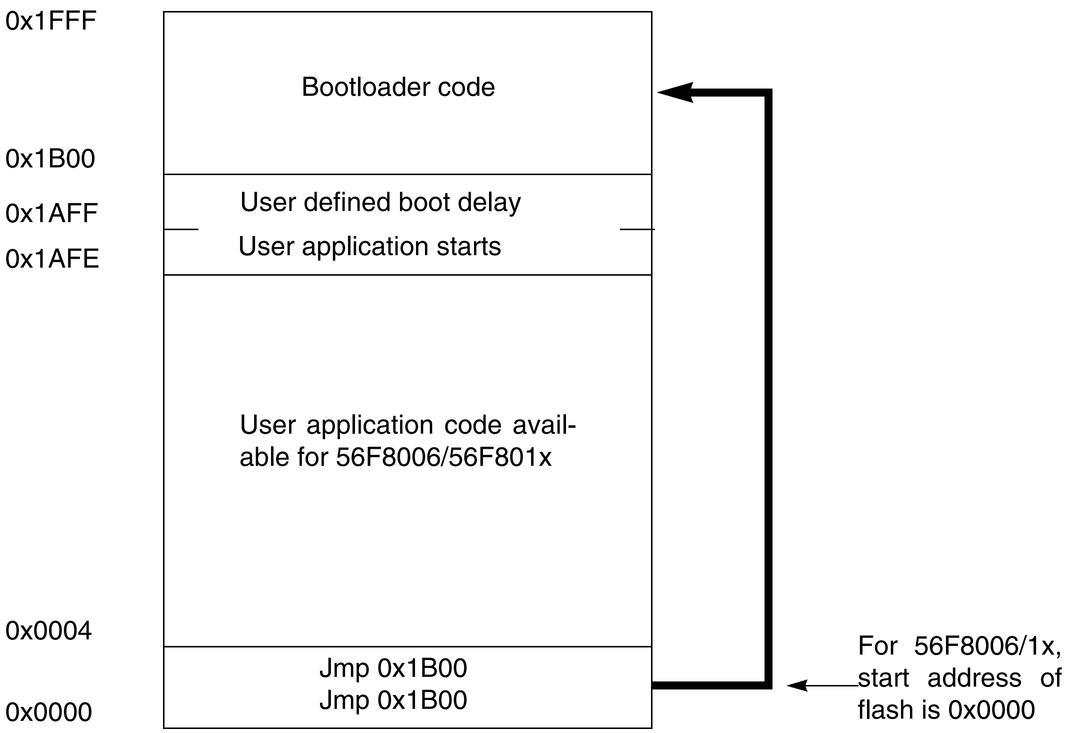


Figure 3. Program Flash Usage for 56F8006/1x Static Serial Bootloader

4 How to Make a Processor Expert Application Downloadable with Serial Bootloader

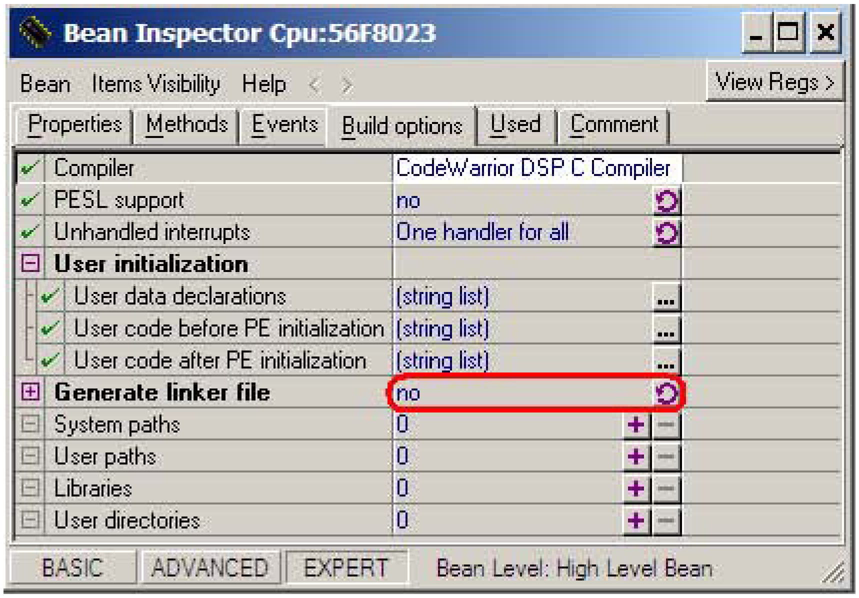
The loaded application should meet these requirements:

1. If it is necessary for the bootloader to start immediately after reset, a jump to address 0x7B00 for MC56F802x/3x or to 0x1B00 for MC56F800x/1x must be written into the reset and COP vector. Otherwise, there should be a jump to bootloader upon user event by calling this instruction:

```
asm{
    jmp  BOOTLOADER_ADDR
        }
```

The BOOTLOADER_ADDR is defined as 0x1B00 for MC56F800x/1x or 0x7B00 for MC56F802x/3x. Please follow the steps below to do the modification.

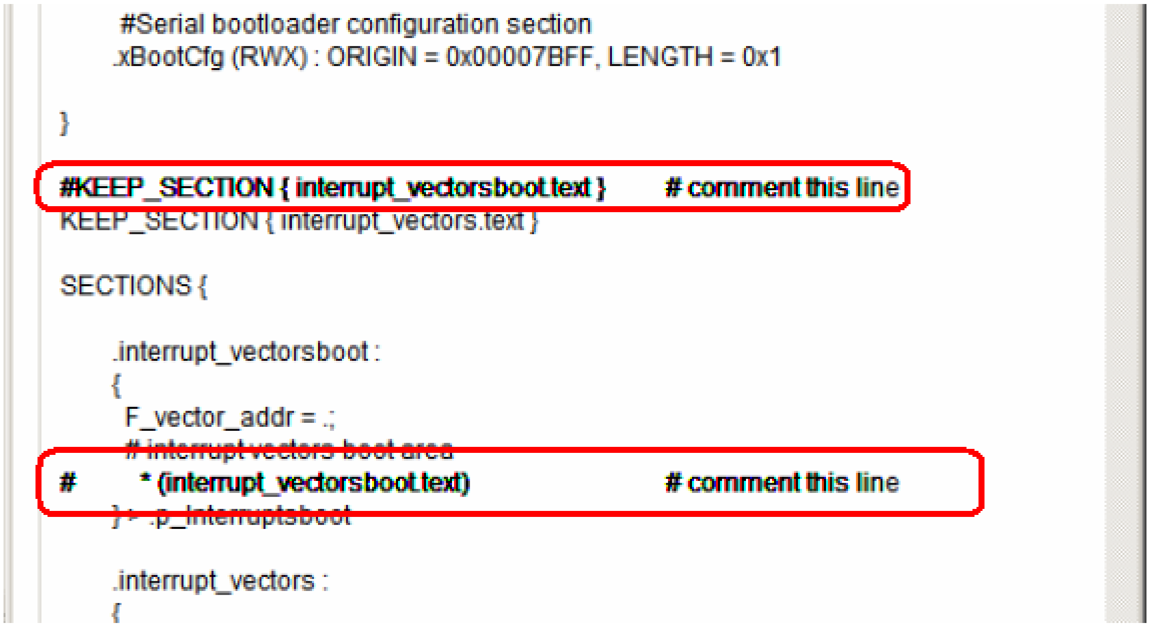
a) After the application code is generated, open the CPU bean and set the “Generate linker file” option to “no.” Thus the linker command file will not be regenerated and the changes can be saved in it.



Compiler	CodeWarrior DSP C Compiler
PESL support	no
Unhandled interrupts	One handler for all
User initialization	
User data declarations	(string list)
User code before PE initialization	(string list)
User code after PE initialization	(string list)
Generate linker file	no
System paths	0
User paths	0
Libraries	0
User directories	0

b) Open the linker command file and modify it:

Comment lines involve the *interrupt_vectorsboot* section, thus the code will not be generated there:



```
        #Serial bootloader configuration section
        .xBootCfg (RWX) : ORIGIN = 0x00007BFF, LENGTH = 0x1

}

#KEEP_SECTION { interrupt_vectorsboot.text }        # comment this line
KEEP_SECTION { interrupt_vectors.text }

SECTIONS {

        .interrupt_vectorsboot :
        {
          F_vector_addr = .;
          # interrupt vectors boot area
#           * (interrupt_vectorsboot.text)          # comment this line
        } > .p_interruptsboot

        .interrupt_vectors :
        {
```

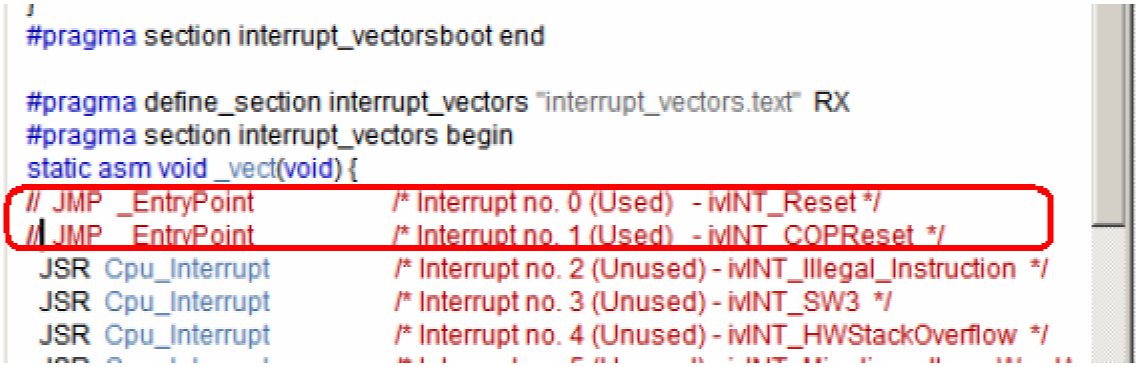
c) Write the jump-to-bootloader code into reset and COP vector by either of the following actions:

– Open the vector.c file and modify the first two lines in *interrupt_vector* section to have a jump to bootloader code:

```
#pragma define_section interrupt_vectors "interrupt_vectors.text"  RX
#pragma section interrupt_vectors begin
static asm void _vect(void) {
```

```
JMP  BOOTLOADER_ADDR //_EntryPoint                    /* Interrupt no. 0 (Used) - ivINT_Reset */
JMP  BOOTLOADER_ADDR //_EntryPoint                  /* Interrupt no. 1 (Used)-ivINT_COPReset  */
JSR  Cpu_Interrupt                     /* Interrupt no.2 (Unused) - ivINT_Illegal_Instruction  */
 JSR  Cpu_Interrupt                       /* Interrupt no.3 (Unused) - ivINT_SW3  */
 JSR  Cpu_Interrupt                       /* Interrupt no.4 (Unused) - ivINT_HWStackOverflow  */
JSR  Cpu_Interrupt                  /* Interrupt no.5 (Unused)-ivINT_MisalignedLongWordAccess  */
```

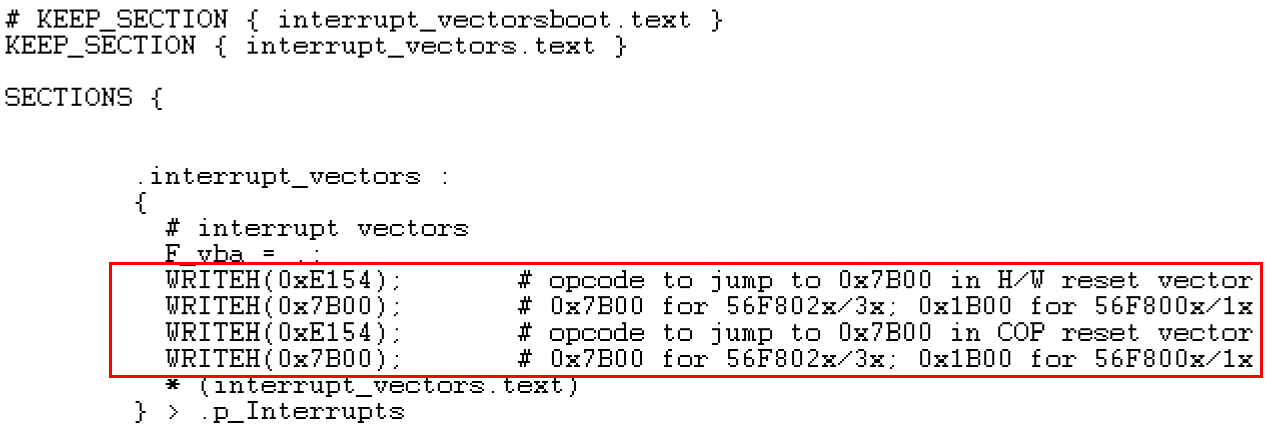
– Or, comment these two lines in vector.c:



```
#pragma section interrupt_vectorsboot end

#pragma define_section interrupt_vectors "interrupt_vectors.text"  RX
#pragma section interrupt_vectors begin
static asm void _vect(void) {
// JMP  _EntryPoint           /* Interrupt no. 0 (Used)   - ivINT_Reset */
// JMP   EntryPoint           /* Interrupt no. 1 (Used)   - ivINT_COPReset */
  JSR  Cpu_Interrupt          /* Interrupt no. 2 (Unused) - ivINT_Illegal_Instruction  */
  JSR  Cpu_Interrupt          /* Interrupt no. 3 (Unused) - ivINT_SW3  */
  JSR  Cpu_Interrupt          /* Interrupt no. 4 (Unused) - ivINT_HWStackOverflow  */
```

Also modify the linker command file to put two “JMP to 0x7B00” instructions (for MC56F802x/3x) or “JMP to 0x1B00” instructions (for MC56F800x/1x) in .p_Interrupts section as shown below:



```
# KEEP_SECTION { interrupt_vectorsboot.text }
KEEP_SECTION { interrupt_vectors.text }

SECTIONS {


        .interrupt_vectors :
        {
          # interrupt vectors
          F_vba = .;
          WRITEH(0xE154);         # opcode to jump to 0x7B00 in H/W reset vector
          WRITEH(0x7B00);         # 0x7B00 for 56F802x/3x; 0x1B00 for 56F800x/1x
          WRITEH(0xE154);         # opcode to jump to 0x7B00 in COP reset vector
          WRITEH(0x7B00);         # 0x7B00 for 56F802x/3x; 0x1B00 for 56F800x/1x
          * (interrupt_vectors.text)
        } > .p_Interrupts
```

NOTE

Be sure that the Processor Expert does not overwrite the vector.c file in the subsequent code generation. Otherwise, modify it as described earlier.

2. Write the application entry point at address 0x7AFE (MC56F802x/3x) or 0x1AFE (MC56F800x/1x) in program flash. Write the bootloader delay value at address 0x7AFF (MC56F802x/3x) or 0x1AFF (MC56F800x/1x) in the program flash.

   Define a new segment .xBootCfg and a new section .ApplicationConfiguration in the linker command file:

```
.xBootCfg (RWX) : ORIGIN = 0x00001AFE, LENGTH = 0x2  # for 56F800x/1x
```

```
        # .xBootCfg (RWX) : ORIGIN = 0x00007AFE, LENGTH = 0x2  # for 56F802x/3x

            .ApplicationConfiguration :
              {
                # Store the application entry point
                WRITEH(F_EntryPoint);

                    #Bootloader start delay in seconds
                    WRITEH(10);

              } > .xBootCfg
```

3. The VBA register must be set correctly before the application enables any interrupts.

   If interrupts are used in the application, the VBA register has to be restored to the proper value so that the chip can jump to the correct vector when an interrupt occurs. Check it in the application code. In the application's main routine, the following sequence must be added before calling PE_low_level_init() for MC56F802x/3x:

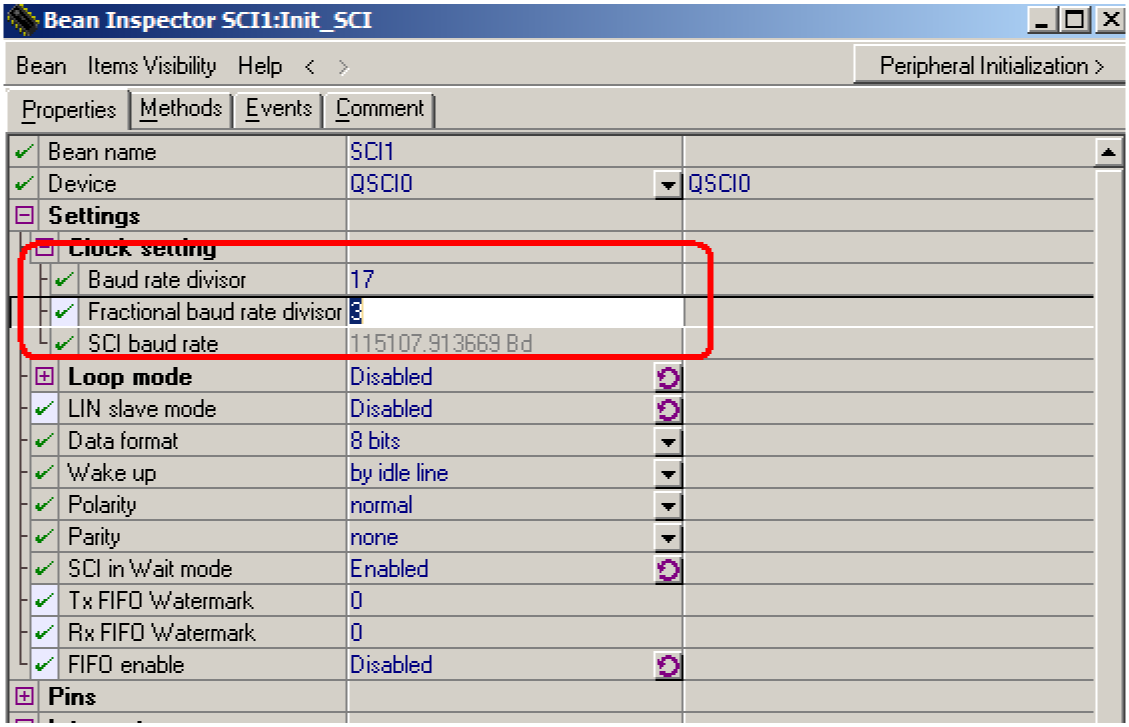
```
extern unsigned int _vba;
INTC_VBA = (int)&_vba >>7;  // restore VBA register to point to user defined vector table
```

   vba is defined in the linker command file as below:

```
.interrupt_vectors :
{
        # interrupt vectors
        F_vba = .;
        * (interrupt_vectors.text)
} > .p_Interrupts
```

5 Serial Communication Speed

The baud rate of the serial communication port is set to 9600 bps by default. Users can change it in the clock setting of SCI1 properties to 115200 bps, as shown below, and then regenerate the code:



Bean name	SCI1	
Device	QSCI0	QSCI0
Settings		
Clock setting		
Baud rate divisor	17	
Fractional baud rate divisor	3	
SCI baud rate	115107.913669 Bd	
Loop mode	Disabled	
LIN slave mode	Disabled	
Data format	8 bits	
Wake up	by idle line	
Polarity	normal	
Parity	none	
SCI in Wait mode	Enabled	
Tx FIFO Watermark	0	
Rx FIFO Watermark	0	
FIFO enable	Disabled	
Pins		

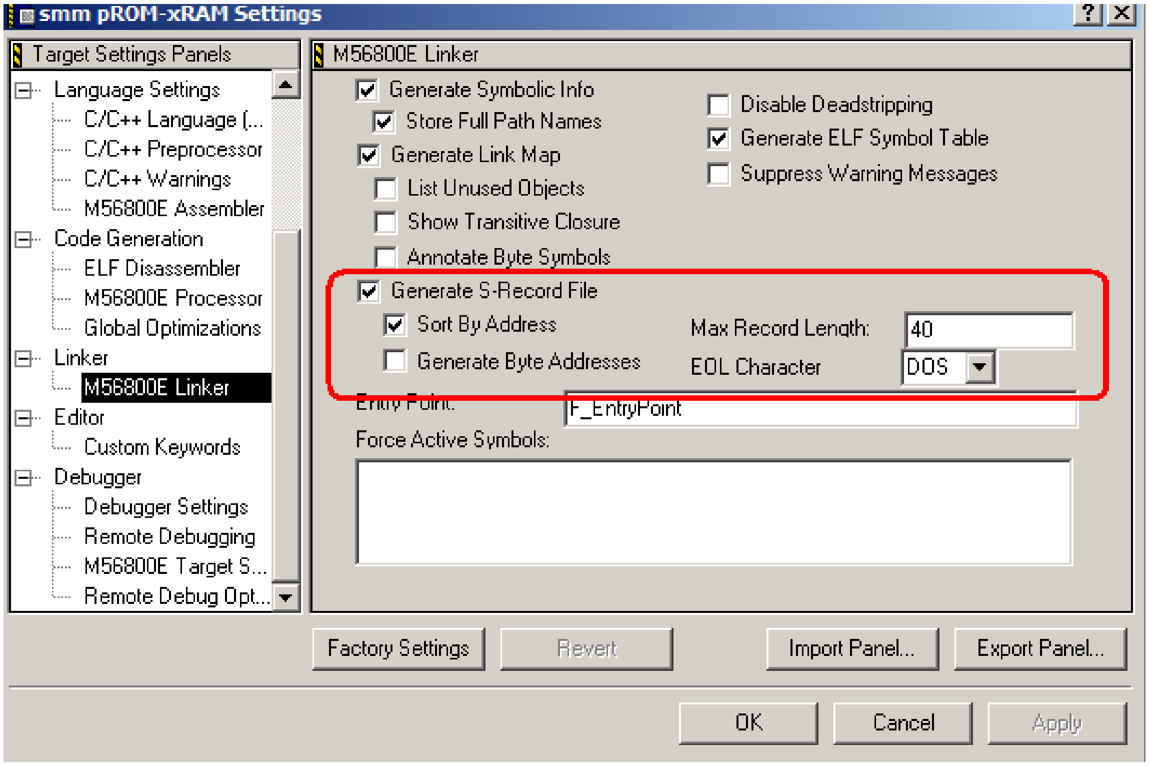
6 Serial Communication Protocol

Use XON/XOFF protocol.

7 How to Generate an S-Record File

1. Configure the M56800E linker as follows. Do not check “Generate Byte Addresses.” The recommended Max Record Length is 40. EOL character must be DOS.
2. After the configuration, click the OK button and then select “Project” menu. Then click “Make” to rebuild the project.

The generated s-record file to be used by the bootloader is the combined p and x s-record file without “.p” or “.x” in the extension name (.s).



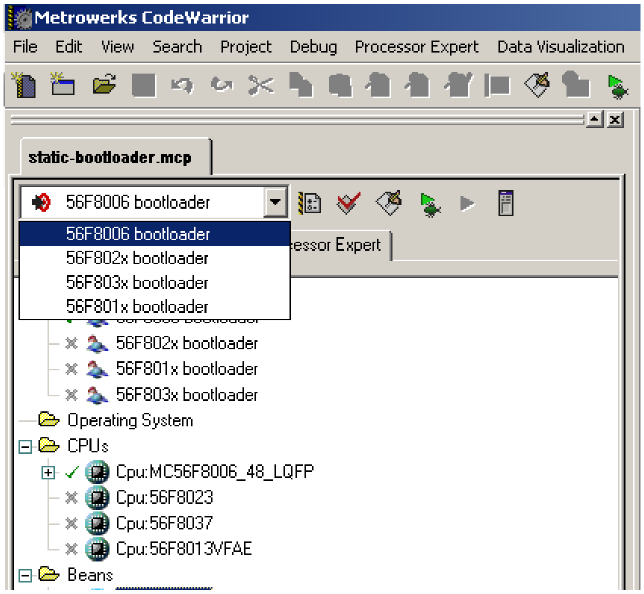
M56800E Linker

☑ Generate Symbolic Info
☑ Store Full Path Names
☑ Generate Link Map
☐ List Unused Objects
☐ Show Transitive Closure
☐ Annotate Byte Symbols
☑ Generate S-Record File
☑ Sort By Address
☐ Generate Byte Addresses
☐ Disable Deadstripping
☑ Generate ELF Symbol Table
☐ Suppress Warning Messages

Max Record Length: 40
EOL Character: DOS
Entry Point: F_EntryPoint
Force Active Symbols:

8 How to Choose the Target

There are four targets in the static serial bootloader project. The selection is done with the help of the Target pulldown menu of the Project window.



56F8006 bootloader

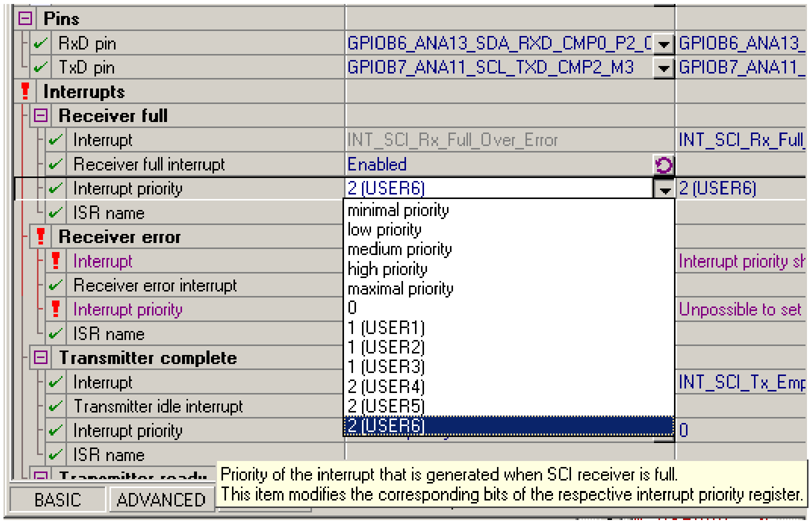
- 56F8006 bootloader
- 56F802x bootloader
- 56F803x bootloader
- 56F801x bootloader

- 56F802x bootloader
- 56F801x bootloader
- 56F803x bootloader
- ✓ Cpu:MC56F8006_48_LQFP
- Cpu:56F8023
- Cpu:56F8037
- Cpu:56F8013VFAE

When changing from the MC56F8006 target to the MC56F801x/2x/3x target, and vice versa, it is necessary to modify the SCI bean with respect to the interrupt priority. Otherwise the Processor Expert code generation fails. The modification is done like this:

8.1 MC56F8006 Bootloader

Select “2 (USER6)” value for the interrupt priority items of the “Receiver full” and “Receiver error” interrupts.

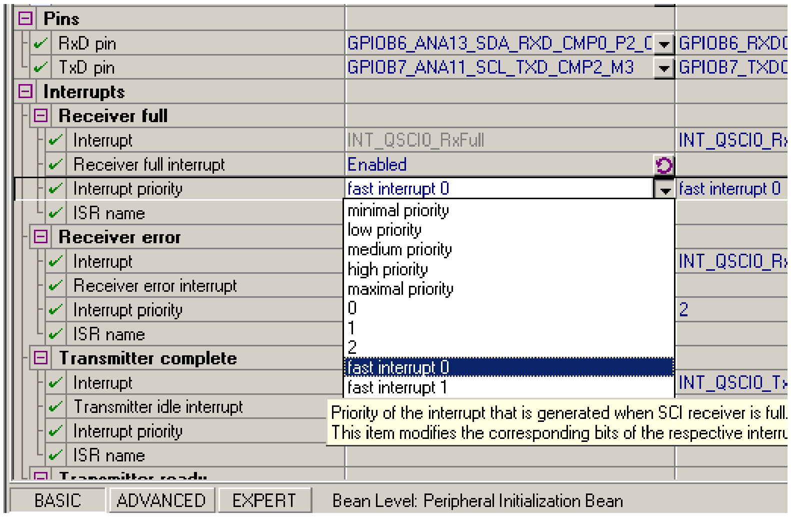


Pins		
RxD pin	GPIOB6_ANA13_SDA_RXD_CMP0_P2_C	GPIOB6_ANA13_
TxD pin	GPIOB7_ANA11_SCL_TXD_CMP2_M3	GPIOB7_ANA11_
Interrupts		
Receiver full		
Interrupt	INT_SCI_Rx_Full_Over_Error	INT_SCI_Rx_Full_
Receiver full interrupt	Enabled	
Interrupt priority	2 (USER6)	2 (USER6)
ISR name		
Receiver error		
Interrupt		Interrupt priority sh
Receiver error interrupt		
Interrupt priority		Unpossible to set
ISR name		
Transmitter complete		
Interrupt		INT_SCI_Tx_Emp
Transmitter idle interrupt		
Interrupt priority		0
ISR name		

Priority of the interrupt that is generated when SCI receiver is full.
This item modifies the corresponding bits of the respective interrupt priority register.

8.2 MC56F801x/2x/3x Bootloaders

Select “fast interrupt 0” value for the interrupt priority item of the “Receiver full” interrupt and value “2” for the “Receiver error” interrupt.



Pins		
RxD pin	GPIOB6_ANA13_SDA_RXD_CMP0_P2_C	GPIOB6_RXD(
TxD pin	GPIOB7_ANA11_SCL_TXD_CMP2_M3	GPIOB7_TXD(
Interrupts		
Receiver full		
Interrupt	INT_QSCI0_RxFull	INT_QSCI0_R›
Receiver full interrupt	Enabled	
Interrupt priority	fast interrupt 0	fast interrupt 0
ISR name		
Receiver error		
Interrupt		INT_QSCI0_R›
Receiver error interrupt		
Interrupt priority		2
ISR name		
Transmitter complete		
Interrupt		INT_QSCI0_T›
Transmitter idle interrupt		
Interrupt priority		
ISR name		

Priority of the interrupt that is generated when SCI receiver is full.
This item modifies the corresponding bits of the respective interru

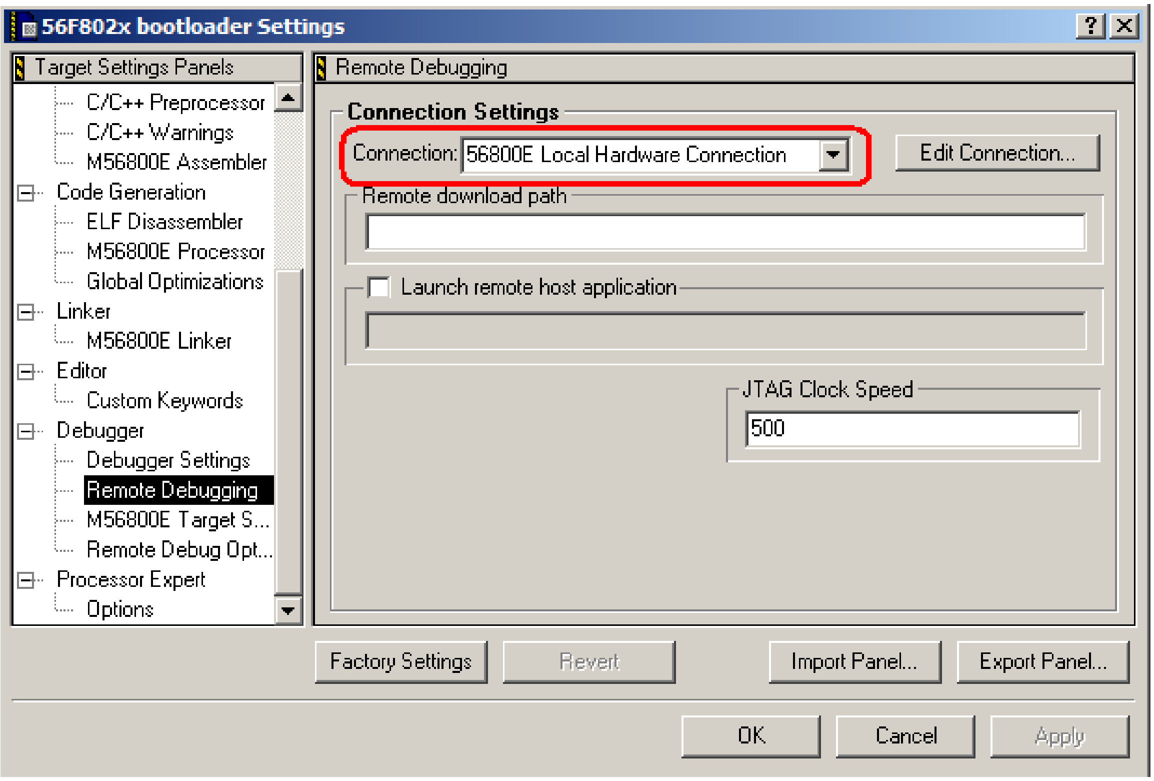
Bean Level: Peripheral Initialization Bean

9 How to Download the Bootloader Code

There are two connection options:

- 56800E Local Hardware Connection: Requires a parallel interface to the PC.
- 56800E Local USBTAP Connection: Requires a USB interface to the PC.

The default setting for the bootloader is “56800E Local USBTAP Connection.” Please follow the picture below to select the one suitable to your development environment and then click the OK button. After that, select “Project” menu, followed by clicking “Debug,” to download the code.



56F802x bootloader Settings

Remote Debugging

Connection Settings

Connection: 56800E Local Hardware Connection

Remote download path

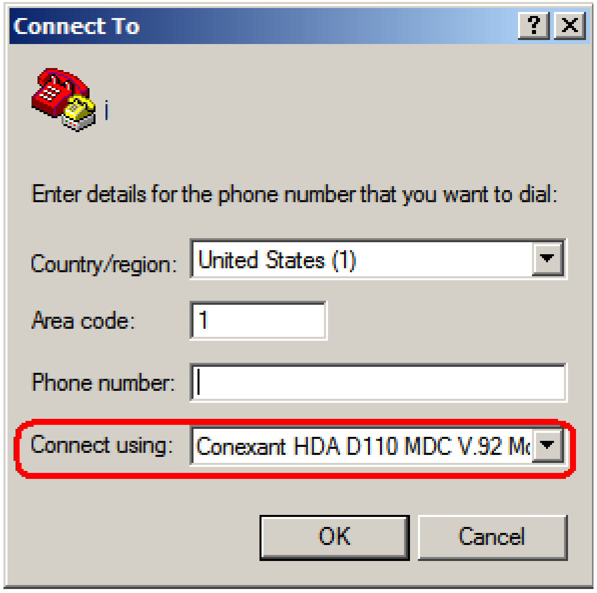
☐ Launch remote host application

JTAG Clock Speed: 500

10 How to Download the Application Code

Most serial terminal programs can be used to download an s-record file from a host to an MC56F800x/1x/2x/3x device via the serial bootloader. For example, HyperTerminal can be used in a Windows platform.

1. Click Windows Start menu. Select “Programs” followed by “Accessories,” and then select “HyperTerminal” under “Communication” menu item to open the HyperTerminal.
2. Enter a name for the connection. Click the “OK” button. The following dialog box pops up. A COM must be selected in the “Connect using” dropdown list.



Connect To

i

Enter details for the phone number that you want to dial:

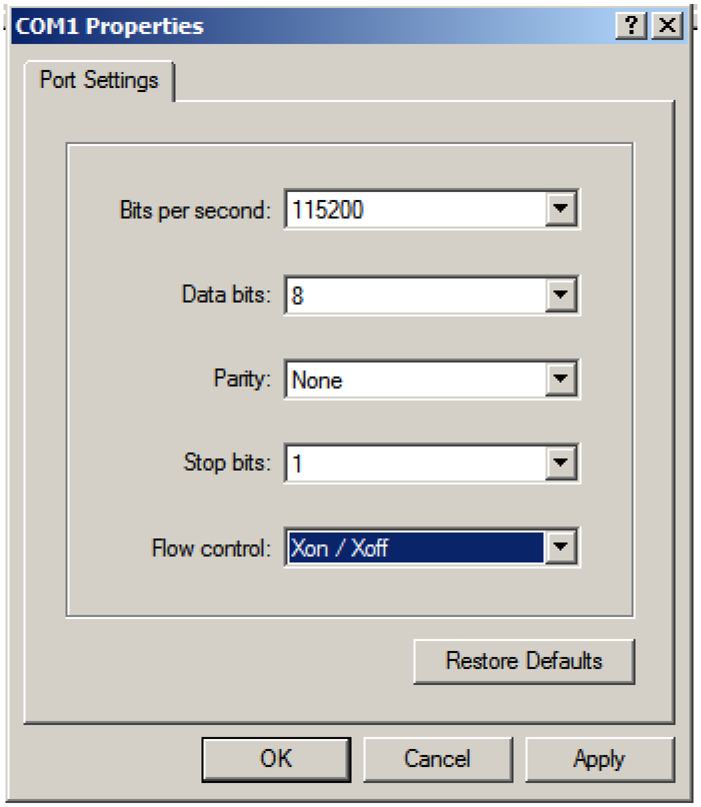
Country/region: United States (1)

Area code: 1

Phone number:

Connect using: Conexant HDA D110 MDC V.92 Mc

3. Configure the COM properties as follows:
   — Baud rate: 9600 bps
   — 8N1: 8 data bits, no parity, 1 stop bit character format
   — Flow control protocol: Xon/Xoff



COM1 Properties

Port Settings

Bits per second: 115200

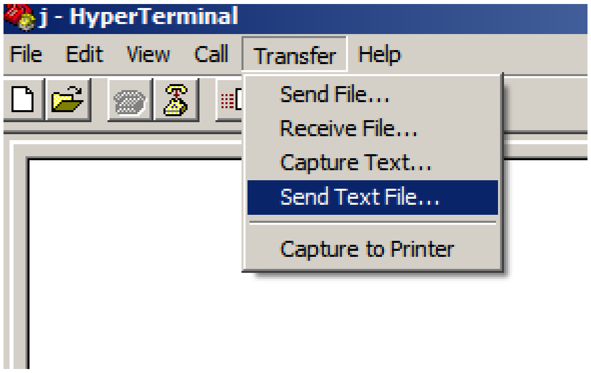
Data bits: 8

Parity: None

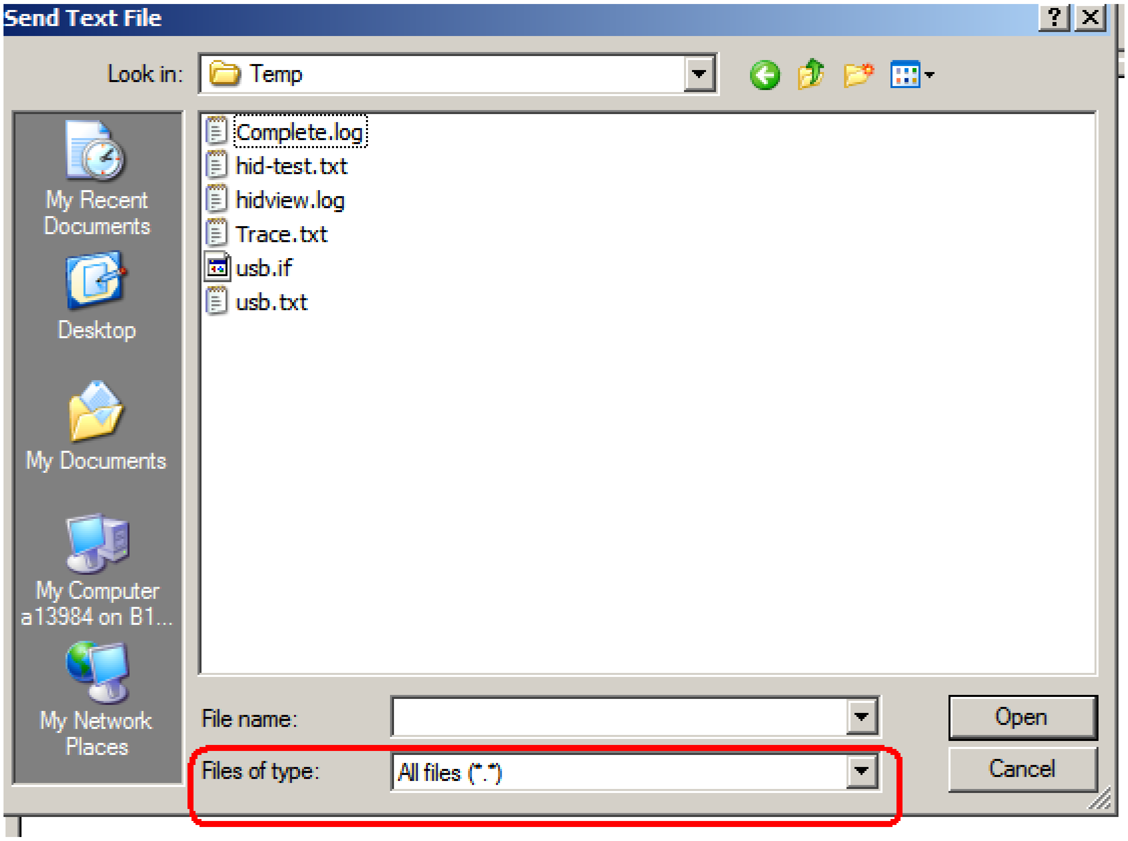
Stop bits: 1

Flow control: Xon / Xoff

4. Follow the following picture to select “Send Text File” when the bootloader startup banner appears:



Now select the application s-record file in the Send Text File dialog. Please note: change the file type to “All files(*.*)” as shown below:



Look in: Temp

Files of type: All files (*.*)

11 Error Code

Error Codes	Meaning	Possible Reasons	What to do
xx01	S-record checksum error	S-record file format is incorrect or serial connection does not support XON/XOFF protocol.	Refer to Section 7, “How to Generate an S-Record File,” and use the serial connection supporting XON/XOFF protocol.
xx02	S-record type error	S-record file format is incorrect or serial connection does not support XON/XOFF protocol.	Refer to Section 7, “How to Generate an S-Record File,” and use the serial connection supporting XON/XOFF protocol.
xx10	Flash program error	May be caused by low voltage of the power supply to the chip.	Check the power supply to see if the voltage is below 2.7 V.
xx20	Wrong code/data start address	S-record file format is incorrect.	Refer to Section 7, “How to Generate an S-Record File.”
xx40	SCI receiving error	The host transmits characters much faster than can be managed by the bootloader, which is normally caused by a breakpoint in the bootloader during debugging.	Remove all breakpoints during the bootloader debugging; lower the baud rate of the bootloader and host terminal.

Note: “x” means don’t care.

12 Technical Support

For any question/problem, you can

- Raise your request at http://www.freescale.com/support
- Send the request to support@freescale.com
- Call hotlines as below:

Region	Country	Language	Phone Number	Office Hours	Time Zone	Time Zone Offset W (S)
North and South America						
	USA	English	1 800 521 6274	8AM-6PM	CT	GMT-6 (-5)
	Mexico	Spanish	00 1800 514 3392	8AM-6PM	CT	GMT-6 (-5)
	Other	Spanish	+52 33 3283-0770	8AM-6PM	CT	GMT-6 (-5)
Europe						
	France	French	+33 1 69 35 48 48	9AM-5PM	CET	GMT+1 (+2)
	Germany	German	+49 89 92103 559	9AM-5PM	CET	GMT+1 (+2)
	UK	English	+44 1296 380 456	9AM-5PM	CET	GMT+1 (+2)
Asia						
	China	Chinese	800 990 8188	9AM-5PM	HKG	GMT+8 (+9)
	Hong Kong	English, Chinese	2666 8080	9AM-5PM	HKG	GMT+8 (+9)
	India	English	000 800 852 1155	9AM-5PM	HKG	GMT+8 (+9)
	Asia Pacific (all others)	English, Chinese	+800 2666 8080	9AM-5PM	HKG	GMT+8 (+9)
	Japan	Japanese	120 191 014	8AM-5PM	TKY	GMT+9 (+10)

Abbreviations:

- GMT: Greenwich Mean Time
- CT: Central Time
- CET: Central European Time
- HKG: Hong Kong Time
- TKY: Tokyo Time
- W: Winter
- S: Summer (Daylight Saving Time)

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale’s Environmental Products program, go to http://www.freescale.com/epp.

freescale™
semiconductor