MSP430 Programming Via the Bootstrap Loader

User's Guide

TEXAS INSTRUMENTS

Literature Number: SLAU319A
July 2010–Revised August 2010

Contents

List of Figures

List of Tables

Chapter 1

SLAU319A–July 2010–Revised August 2010

Programming Via the Bootstrap Loader

The MSP430 BSL enables users to communicate with embedded memory in the MSP430 microcontroller during the prototyping phase, final production, and in service. Both the programmable memory (flash memory) and the data memory (RAM) can be modified as required. Do not confuse the bootstrap loader with programs found in some digital signal processors (DSPs) that automatically load program code (and data) from external memory to the internal memory of the DSP. These programs are often referred to as bootstrap loaders as well.

To use the bootstrap loader, a specific BSL entry sequence must be applied. An added sequence of commands initiates the desired function. A boot-loading session can be exited by continuing operation at a defined user program address or by the reset condition.

If the device is secured by disabling JTAG, it is still possible to use the BSL. Access to the MSP430 memory via the BSL is protected against misuse by a user-defined password.

1.1 Supplementary Online Information

As a compliment to this document, a BSL wiki page is available. This wiki contains links to additional BSL projects, information, and an errata for this document. The wiki can be found at http://processors.wiki.ti.com/index.php/BSL_(MSP430).

A zip file with additional information, executables, and code samples can be found at http://www.ti.com/lit/zip/slau319.

1.2 Introduction

This bootstrap loader provides a method to program the flash memory during MSP430 project development and updates. It can be activated by a utility that sends commands via the UART protocol. The BSL enables the user to control the activity of the MSP430 and to exchange data using a personal computer or other device.

To avoid accidental overwriting of the BSL code, this code is stored in a secure memory location, either ROM or specially protected flash. To prevent unwanted source readout, any BSL command that directly or indirectly allows data reading is password protected.

To invoke the bootstrap loader, a BSL entry sequence must be applied to dedicated pins. After that, a synchronization character, followed by the data frame of a specific command, initiates the desired function.

1.3 Standard RESET and BSL Entry Sequence

1.3.1 MSP430 Devices With Shared JTAG Pins

Applying an appropriate entry sequence on the $\overline{\text{RST}}$/NMI and TEST pins forces the MSP430 to start program execution at the BSL RESET vector instead of at the RESET vector located at address FFFEh.

If the application interfaces with a computer UART, these two pins may be driven by the $\overline{\text{DTR}}$ and $\overline{\text{RTS}}$ signals of the serial communication port (RS232) after passing level shifters. Detailed descriptions of the hardware and related considerations can be found in Chapter 4. The normal user reset vector at FFFEh is used, if TEST is kept low while $\overline{\text{RST}}$/NMI rises from low to high (standard method, see Figure 1-1).

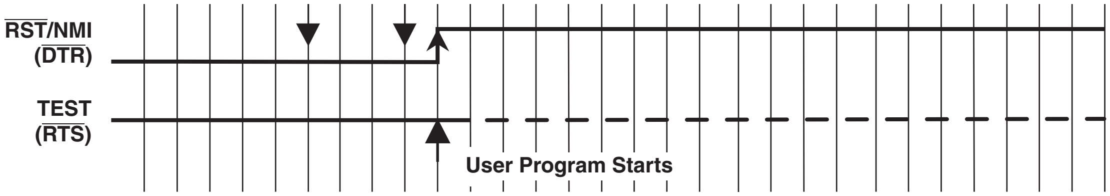
$\overline{\text{RST}}$/NMI
($\overline{\text{DTR}}$)
TEST
($\overline{\text{RTS}}$)
User Program Starts

Figure 1-1. Standard RESET Sequence

The BSL program execution starts when the TEST pin has received a minimum of two positive transitions and if TEST is high while $\overline{\text{RST}}$/NMI rises from low to high (BSL entry method, see Figure 1-2). This level/transition triggering improves BSL startup reliability.

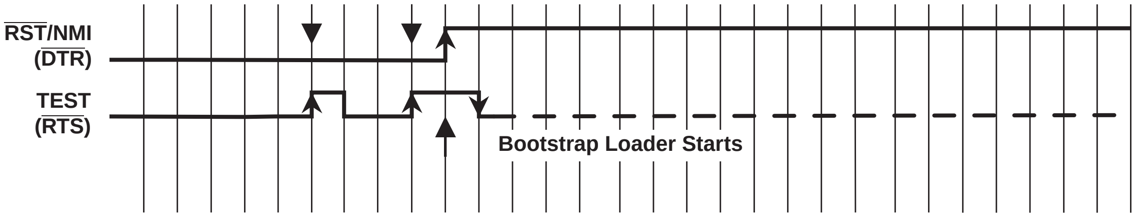
$\overline{\text{RST}}$/NMI
($\overline{\text{DTR}}$)
TEST
($\overline{\text{RTS}}$)
Bootstrap Loader Starts

Figure 1-2. BSL Entry Sequence at Shared JTAG Pins

The TEST signal is normally used to switch the port pins P1.7 to P1.4 between their application function and the JTAG function. If the second rising edge at the TEST pin is applied while $\overline{\text{RST}}$/NMI is still low, the TEST signal is kept low internally (application mode).

The BSL is not started via the BSL RESET vector if:

- There are fewer than two positive edges at the TEST pin while $\overline{\text{RST}}$/NMI is low.
- TEST is low when $\overline{\text{RST}}$/NMI rises from low to high.
- JTAG has control over the MSP430 resources.
- Supply voltage, V_{CC}, drops below its threshold, and a power-on reset (POR) is executed.
- $\overline{\text{RST}}$/NMI pin is configured for NMI functionality (NMI bit is set).

NOTE: The minimum timing for this sequence must be within the limits specified for the corresponding pin in the data sheet.

1.3.2 MSP430 Flash Devices With Dedicated JTAG Pins

Devices with dedicated JTAG pins use the TCK pin instead of the TEST pin.

The BSL program execution starts whenever the TCK pin has received a minimum of two negative transitions and TCK is low while $\overline{\text{RST}}$/NMI rises from low to high (BSL entry method, see Figure 1-3). This level/transition triggering improves BSL start-up reliability.

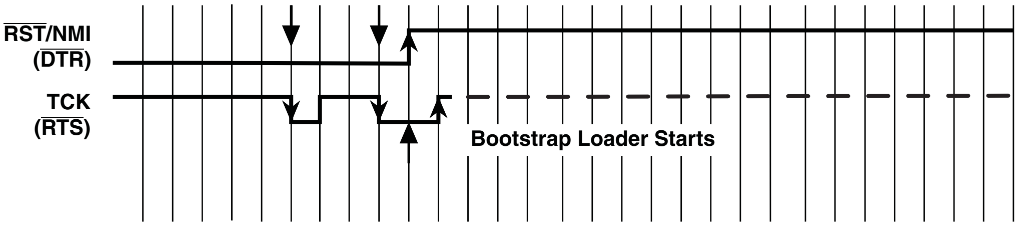

Figure 1-3. BSL Entry Sequence at Dedicated JTAG Pins

NOTE: The minimum timing for this sequence must be within the limits specified for the corresponding pin in the data sheet.

The BSL is not started via the BSL RESET vector if

- There are fewer than two negative edges at TCK pin while $\overline{\text{RST}}$/NMI is low.
- TCK is high if $\overline{\text{RST}}$/NMI rises from low to high.
- JTAG has control over the MSP430 resources.
- Supply voltage, V_{CC}, drops below its threshold, and a power-on reset (POR) is executed.
- $\overline{\text{RST}}$/NMI pin is configured for NMI functionality (NMI bit is set).

1.3.3 Devices With USB

Devices with USB are invoked when either of the following two conditions are met while the device is powered by VBUS:

- The device is powered up by USB and the reset vector is blank.
- The device powers up with the PUR pin tied to V_{USB}.

1.4 UART Protocol

The UART protocol applied here is defined as:

- Baud rate is fixed to 9600 baud in half-duplex mode (one sender at a time).
- Start bit, 8 data bits (LSB first), an even parity bit, 1 stop bit.
- Handshake is performed by an acknowledge character.
- Minimum time delay before sending new characters after characters have been received from the MSP430 BSL: 1.2 ms

NOTE: Applying baud rates other than 9600 baud at initialization results in communication problems or violates the flash memory write timing specification. The flash memory may be extensively stressed or may react with unreliable program/erase operations.

1.5 USB Protocol

The USB protocol applied here is defined as:

- HID protocol with one input endpoint and one output endpoint. Each endpoint has a length of 64 bytes.
- VID: 0x2047
- PID: 0x0200

ROM-Based Bootstrap Loader Protocol

2.1 Synchronization Sequence

Before sending any command to the BSL, a synchronization character (SYNC) with its value of 80h must be sent to the BSL. This character is necessary to calculate all the essential internal parameters, which maintain UART and flash memory program/erase timings. It provides the BSL system time reference. Once this is received, an acknowledge DATA_ACK = 90h is sent back by the BSL to confirm successful reception.

This sequence must be done before every command that is sent to the BSL.

NOTE: The synchronization character is not part of the Data Frame described in Section 2.4.

2.2 Commands

Two categories of commands are available: commands that require a password and commands that do not require a password. The password protection safeguards every command that potentially allows direct or indirect data access.

2.2.1 Unprotected Commands

- Receive password
- Mass erase (erase entire flash memory, main as well as information memory)
- Transmit BSL version (V1.50 or higher or in loadable BL_150S_14x.txt but not 2.x BSLs)
- Change baud rate (V1.60 or V1.61 or V2.0x or in loadable BL_150S_14x.txt)

2.2.2 Password Protected Commands

- Receive data block to program flash memory, RAM, or peripherals
- Transmit data block
- Erase segment
- Erase check (Present in V1.60 or higher or in loadable BL_150S_14x.txt)
- Set Memory Offset (Present in V2.12 or higher)
- Load program counter and start user program
- Change baud rate (BSL versions lower than V1.60 and higher than 2.10)

2.3 Programming Flow

The write access (RX data block command) to the flash memory/RAM or peripheral modules area is executed online. That means a data byte/word is processed immediately after receipt and the write cycle is finished before a following byte/word has completely arrived. Therefore, the entire write time is determined by the baud rate, and no buffering mechanism is necessary.

Data sections located below the flash memory area address are assumed to be loaded into the RAM or peripheral module area and, thus, no specific flash control bits are affected.

NOTE: If control over the UART protocol is lost, either by line faults or by violating the data frame conventions, the only way to recover is to rerun the BSL entry sequence to initiate another BSL session.

2.4 Data Frame

To ensure high data security during the data transmission, a data frame protocol called serial standard protocol (SSP) is used. The BSL is considered the receiver in Table 2-1.

2.4.1 Data-Stream Structure

- The first eight bytes (HDR through LH) are mandatory (xx represents dummy data).
- Data bytes D1 to Dn are optional.
- Two bytes (CKL and CKH) for checksum are mandatory.
- Acknowledge done by the BSL is mandatory, except with the TX data block and TX BSL version commands.

Table 2-1. Data Frame of BSL Commands[1] [2] [3] [4] [5] [6]

Received BSL Command	HDR	CMD	L1	L2	AL	AH	LL	LH	D1	D2…Dn	CKL	CKH	ACK
RX data block	80	12	n	n	AL	AH	n–4	0	D1	D2 … Dn–4	CKL	CKH	ACK
RX password	80	10	24	24	xx	xx	xx	xx	D1	D2 … D20	CKL	CKH	ACK
Erase segment	80	16	04	04	AL	AH	02	A5	–	– – –	CKL	CKH	ACK
Erase main or info	80	16	04	04	AL	AH	04	A5	–	– – –	CKL	CKH	ACK
Mass erase	80	18	04	04	xx	xx	06	A5	–	– – –	CKL	CKH	ACK
Erase check	80	1C	04	04	AL	AH	LL	LH	–	– – –	CKL	CKH	ACK
Change baud rate	80	20	04	04	D1	D2	D3	xx	–	– – –	CKL	CKH	ACK
Set mem offset	80	21	04	04	xx	xx	AL	AH	–	– – –	CKL	CKH	ACK
Load PC	80	1A	04	04	AL	AH	xx	xx	–	– – –	CKL	CKH	ACK
TX data block	80	14	04	04	AL	AH	n	0	–	– – –	CKL	CKH	–
BSL responds	80	xx	n	n	D1	D2 ...	...	…	...	… Dn	CKL	CKH	–
TX BSL version	80	1E	04	04	xx	xx	xx	xx	–	– – –	CKL	CKH	–
BSL responds	80	xx	10	10	D1	D2 ...	...	…	…	… D10	CKL	CKH	–

[1] All numbers are bytes in hexadecimal notation.

[2] ACK is sent back by the BSL.

[3] The synchronization sequence is not part of the data frame.

[4] The erase check and TX BSL version commands are members of the standard command set in BSLs V1.50 or higher but excluding 2.x BSLs.

[5] The change baud rate command is not a member of the standard command set (V1.60 or higher or in loadable BL_150S_14x.txt).

[6] Abbreviations
HDR: Header. Any value between 80h and 8Fh (normally 80h).
CMD: Command identification
L1, **L2**: Number of bytes consisting of AL through Dn. Restrictions: L1 = L2, L1 < 255, L1 even
AL, **AH**: Block start address or erase (check) address or jump address LO/HI byte
LL, **LH**: Number of pure data bytes (250 max) or erase information LO/HI byte or block length of erase check (FFFFh max)
D1 … **Dn**: Data bytes
CKL, **CKH**: 16-bit checksum LO/HI byte
xx: Can be any data
–: No character (data byte) received/transmitted
ACK: The acknowledge character returned by the BSL. Can be either DATA_ACK = 90h: Frame was received correctly, command was executed successfully, or DATA_NAK = A0h: Frame not valid (e.g., wrong checksum, L1 ≠ L2), command is not defined, is not allowed, or was executed unsuccessfully.
n: Number of bytes consisting of AL through Dn

2.4.2 Checksum

The 16-bit (2 bytes) checksum is calculated over all received/transmitted bytes B1 ... Bn in the data frame, except the checksum bytes themselves, by XORing words (two successive bytes) and inverting the result.

This means that B1 is always the HDR byte and Bn is the last data byte just before the CKL byte.

Formula

CHECKSUM = INV [(B1 + 256 × B2) XOR (B3 + 256 × B4) XOR … XOR (Bn–1 + 256 × Bn)]
or
CKL = INV [B1 XOR B3 XOR … XOR Bn–1]
CKH = INV [B2 XOR B4 XOR … XOR Bn]

2.4.3 Example Sequence

The following example shows a request to read the memory of the MSP430 from location 0x0F00. All values shown below are represented in hexadecimal format.

TO BSL:	80 (Synchronization character sent to the BSL)
FROM BSL:	90 (Acknowledge from BSL)
TO BSL:	80 14 04 04 F0 0F 0E 00 85 E0 (Send Command to read memory from 0x0F00, length 0x000E)
FROM BSL:	80 00 0E 0E F2 13 40 40 00 00 00 00 00 00 02 01 01 01 C0 A2 (Returned values from BSL)

2.4.4 Commands – Detailed Description

See Table 2-1.

2.4.4.1 General

Following the header byte HDR (80h) and the command identification CMD, the frame length bytes L1 and L2 (which must be equal) hold the number of bytes following L2, excluding the checksum bytes CKL and CKH.

Bytes AL, AH, LL, LH, D1...Dn are command-specific. However, the checksum bytes CKL (low byte) and CKH (high byte) are mandatory.

If the data frame has been received correctly and the command execution was successful, an acknowledge character DATA_ACK = 90h is sent back by the BSL. Incorrectly received data frames, unsuccessful operations, and commands that are locked or not defined are confirmed with a DATA_NAK = A0h.

NOTE: BSL versions lower than V1.30 support only byte-access operations. Therefore, the peripheral module addresses at 0100h to 01FFh cannot be accessed correctly, because they are word-oriented. In version V1.30 and higher, addresses 0000h to 00FFh are accessed in byte mode; all others are accessed in word mode.

2.4.4.2 RX Data Block

The receive data block command is used for any write access to the flash memory/RAM or peripheral module control registers at 0000h to 01FFh. It is password protected.

The 16-bit even-numbered block start address is defined in AL (low byte) and AH (high byte). The 16-bit even-numbered block length is defined in LL (low byte) and LH (high byte). Because pure data bytes are limited to a maximum of 250, LH is always 0.

The following data bytes are succeeded by the checksum bytes CKL (low byte) and CKH (high byte). If the receipt and programming of the appropriate data block was successful, an acknowledge character DATA_ACK is sent back by the BSL. Otherwise, the BSL confirms with a DATA_NAK.

NOTE: BSL versions V1.40 and higher support online verification inside the MSP430 for addresses 0200h to FFFFh, which reduces programming/verification time by 50%. Online verification means that the data is immediately verified with the data that is written into the flash without transmitting it again. In case of an error, the loadable bootstrap loader BL_150S_14x.txt additionally stores the first incorrectly written location address+3 into the error address buffer in the RAM at address 0200h (021Eh for F14x devices).

2.4.4.3 RX Password

The receive password command is used to unlock the password-protected commands, which perform reading, writing, or segment-erasing memory access. It is not password protected.

Neither start address nor block length information is necessary, because the 32-byte password is always located at addresses FFE0h to FFFFh. Data bytes D1 to D20h hold the password information starting with D1 at address FFE0h.

If the receipt and verification of the password is correct, a positive acknowledge DATA_ACK is sent back by the BSL, and the password-protected commands become unlocked. Otherwise the BSL confirms with a DATA_NAK.

Once the protected commands become unlocked, they remain unlocked until another BSL entry is initiated.

2.4.4.4 Mass Erase

The mass erase command erases the entire flash memory area (main memory plus information memory, see corresponding data sheet). It is not password protected.

All parameters shown in Table 2-1 are mandatory. After erasing, an acknowledge character DATA_ACK is sent back by the BSL.

Mass erase initializes the password area to 32 times 0FFh.

NOTE: BSL versions V2.01 and higher support automatic clearing of the LOCKA bit protecting information flash memory. When the BSL is entered from a reset condition, LOCKA is cleared by the BSL to mass erase the flash, including information memory. When the BSL is entered in-application, user software should ensure that LOCKA is written as 1 prior to initiating the BSL. Otherwise, information flash is not erased during a BSL mass erase.

2.4.4.5 Erase Segment

The erase segment command erases specific flash memory segments. It is password protected.

The address bytes AL (low byte) and AH (high byte) select the appropriate segment. Any even-numbered address within the segment to be erased is valid. After segment erasing, an acknowledge character DATA_ACK is sent back by the BSL (V1.40 or lower).

BSL versions V1.60 or higher perform a subsequent erase check of the corresponding segment and respond with a DATA_NAK if the erasure was not successful. In this case, the first non-erased location address + 1 is stored in the error address buffer in the RAM at address 0200h (021Eh for F14x devices). In this version, a problem occurs if only one of the information memory segments is erased. In this case, an error is reported, because an automatic erase check over the whole information memory is performed. As a solution, either erase the whole information memory or do a separate erase check after the erase, even if the erase reported an error.

Erase segment 0 clears the password area and, therefore, the remaining password is 32 times 0FFh.

When applying LL = 0x04 and LH = 0xA5, a mass erasure of only the main memory is performed. Indeed, this command must be executed a minimum of 12 times to achieve a total erasure time of >200 ms. No subsequent erase check of the entire main memory is done. Use the erase check command additionally. Check the device data sheet for more information on the cumulative (mass) erase time that must be met and the number of erase cycles required.

2.4.4.6 Erase Main or Info

The erase main or info command erases specific flash memory section. It is password protected.

The address bytes AL (low byte) and AH (high byte) select the appropriate section of flash (Main or INFO). Any even-numbered address within the section to be erased is valid.

2.4.4.7 Erase Check

The erase check command verifies the erasure of flash memory within a certain address range. It is password protected.

The 16-bit block start address is defined in AL (low byte) and AH (high byte). The 16-bit block length is defined in LL (low byte) and LH (high byte). Both can be either even or odd numbered to allow odd boundary checking.

If the erase check of the appropriate data block was successful (all bytes contain 0FFh), an acknowledge character DATA_ACK is sent back by the BSL. Otherwise, the BSL confirms with a DATA_NAK and the first non-erased location address + 1 is stored in the error address buffer at address 0200h (021Eh for F14x devices).

NOTE: This command is not a member of the standard command set. It is implemented in BSL version V1.60 and higher or in the loadable bootstrap loader BL_150S_14x.txt.

2.4.4.8 Change Baud Rate

The change baud rate command offers the capability of transmissions at higher baud rates than the default 9600 baud. With faster data transition, shorter programming cycles can be achieved, which is especially important with large flash memory devices. This command is not password protected.

Three control bytes (D1 to D3) determine the selected baud rate. D1 and D2 set the processor frequency ($f \geq f_{min}$), D3 indirectly sets the flash timing generator frequency ($f_{FTGmin} \leq f_{FTG} \leq f_{FTGmax}$). In detail:

D1:	F1xx: Basic clock module control register DCOCTL (DCO.2 to DCO.0)
	F2xx: Basic clock module control register DCOCTL (DCO.2 to DCO.0)
	F4xx: FLL+ system clock control register SCFI0 (D, FN_8 to FN_2)
D2:	F1xx: basic clock module control register BCSCTL1 (XT2Off, Rsel.2 to Rsel.0)
	F2xx: basic clock module control register BCSCTL1 (XT2Off, Rsel.2 to Rsel.0)
	F4xx: FLL+ system clock control register SCFI1 (N_{DCO})
D3	0: 9600 Baud
	1: 19200 Baud
	2: 38400 Baud

After receiving the data frame, an acknowledge character DATA_ACK is sent back, and the BSL becomes prepared for the selected baud rate. It is recommended for the BSL communication program to wait approximately 10 ms between baud rate alteration and succeeding data transmission to give the BSL clock system time for stabilization.

NOTE: The highest achievable baud rate depends on various system and environment parameters like supply voltage, temperature range, and minimum/maximum processor frequency. See the corresponding device specification/data sheet.

NOTE: This command is implemented on BSL versions V1.60 or higher or available in the loadable bootstrap loader BL_150S_14x.txt.

Table 2-2. Recommendations for MSP430F149 ['F449][1] (T_A = 25°C, V_{CC} = 3.0 V, f_{max} = 6.7 MHz)

BAUD RATE (Baud)	PROCESSOR FREQUENCY, f_{min} (MHz) [2]	D1 DCOCTL [SCFI0] [3]	D2 BCSCTL1 [SCFI1] [3]	D3 [3]	PROGRAM/VERIFY 60 Kbytes (sec) [4]
9600 (init)	1.05	0x80 [00]	0x85 [98]	00 [00]	78 + 3.7 [0.0]
19200	2.1	0xE0 [00]	0x86 [B0]	01 [01]	39 + 3.7 [2.4]
38400	4.2	0xE0 [00]	0x87 [C8]	02 [02]	20 + 3.7 [2.4]

[1] Values in brackets [ ] apply to MSP430F449.
[2] The minimum processor frequency is lower than in the standard ROM BSL (see Section 2.9.3, Initialization Status).
[3] D1 to D3 are bytes in hexadecimal notation.
[4] Additional 3.7 [2.4] seconds result from loading, verifying, and launching the loadable BSL.

Table 2-3. Recommendations for MSP430F2131[1] (T_A = 25°C, V_{CC} = 3.0 V, f_{max} = 6.7 MHz)

BAUD RATE (Baud)	PROCESSOR FREQUENCY, f_{min} (MHz) [2]	D1 DCOCTL [SCFI0] [3]	D2 BCSCTL1 [SCFI1] [3]	D3 [3]	PROGRAM/VERIFY 60 Kbytes (sec)
9600 (init)	1.05	0x80	0x85	00	78
19200	2.1	0x00	0x8B	01	39
38400	4.2	0x80	0x8C	02	20

[1] Values in brackets [ ] are related to MSP430F449.
[2] The minimum processor frequency is lower than in the standard ROM BSL (see Section 2.9.3, Initialization Status).
[3] D1 to D3 are bytes in hexadecimal notation.

2.4.4.9 Set Memory Offset

An offset for the memory pointer can be set for devices that have more than 64K of memory, specifically MSP430X architecture devices. The value for the memory offset is used as the memory pointer’s upper word.

MemoryAddress = OffsetValue << 16 + ActualAddress

NOTE: This command is implemented on BSL versions V2.12 and higher.

2.4.4.10 Load PC

The load program counter command directs the program counter (register R0) to any location within the entire address range. It is password protected.

After receiving the data frame, an acknowledge character (DATA_ACK) is sent back by the BSL. Then the selected address is moved into the program counter. The program flow continues operation there, and the BSL session is terminated.

Be aware that the password protection is not active at this time.

2.4.4.11 TX Data Block

The transmit data block command is used for any read access to the flash memory/RAM or peripheral module control registers at 0000h to 01FFh. It is password protected.

The 16-bit block start address is defined in AL (low byte) and AH (high byte). The 16-bit block length is defined in LL (low byte) and LH (high byte). Because pure data bytes are limited to a maximum of 250, LH is always 0. The checksum bytes CKL (low byte) and CKH (high byte) immediately follow this information.

Now the BSL responds with the requested data block. After transmitting HDR, dummy CMD, L1 and L2, The BSL sends data bytes D1 through Dn, followed by the checksum bytes CKL (low byte) and CKH (high byte). No acknowledge character is necessary.

2.4.4.12 TX BSL Version

The transmit BSL version command gives the user information about chip identification and bootstrap loader software version. It is not password protected.

The values for AL, AH, LL, and LH can be any data, but must be transmitted to meet the protocol requirements. The checksum bytes CKL (low byte) and CKH (high byte) follow this information.

After that, the BSL responds with a 16-byte data block. After transmitting HDR, dummy CMD, L1 and L2, the BSL sends data bytes D1 through D16 (decimal), followed by the checksum bytes CKL (low byte) and CKH (high byte). No acknowledge character is necessary.

D1, D2 and D11, D12 (decimal) hold the specific information:

- D1: Device family type (high byte)
- D2: Device family type (low byte)
- D11: BSL version (high byte)
- D12: BSL version (low byte) The remaining 12 bytes are for internal use only.

2.5 Loadable BSL

For upgrading the BSL functionality, sometimes it is suitable to load a higher version of BSL into the RAM of a device and apply the latest innovations. To do so, use the following BSL commands:

- RX password (unlock password protection for following commands)
- RX data block (code of loadable BSL, code section address ≥ 220h)
- TX data block (for verification)
- RX data block (get start address from first code section address)
- Load program counter PC (with start address of loadable BSL)
- Wait at least 5 ms until the new loaded BSL has executed the initialized routine
- RX password (unlock password protection for loaded BSL)
- Perform any command (with loaded BSL)

The following loadable BSLs are available:

- **BL_150S_14x**.txt is a complete BSL for the F14x/F13x family with BSL version 1.10. All features of BSL version V1.60 are supported. Because its code size is bigger than 1 Kbyte, it can be used only in 'F1x8 and 'F1x9 devices. The error address buffer address for "RX Block", "Erase Segment", and "Erase Check" commands is 021Eh. BL_150S_14x.txt could also be used as a replacement for PATCH.txt.
- **BS_150S_14x**.txt is a small BSL with reduced command set for the F14x/F13x family with BSL version 1.10. Because its code size is smaller than 512B, it can be used in 'F1x4 up to 'F1x9 devices. The following commands of BSL version V1.60 are supported: Change Baud Rate, RX Block (with online verification), Erase Check, and Load PC. If a TX Block command (redirected to ROM BSL) is needed (e.g., for transmitting error address or standalone Verify), the RAM BSL must be invoked again via the Load PC command. The error address buffer address for RX Block and Erase Check commands is 021Eh. BS_150S_14x.txt could also be used as a partial replacement for PATCH.txt. Note that no password is required, as the RX password command is stripped!

For more information on downloading a different bootstrap loader, see *Application of Bootstrap Loader in MSP430 With Flash Hardware and Software Proposal* (SLAU319).

Third-party software normally uses loadable BSLs to perform most functions, like online verification, and to improve speed for appropriate devices.

2.6 Exiting the BSL

To exit the BSL mode, two possibilities are provided:

- The microcontroller continues operation at a defined program address invoked by the load program counter command. Be aware of that the password protection is not active at this time. In this case, the user application should ensure that the flash is locked, as this is not done by the BSL. Leaving the BSL unlocked increases the risk of erroneously modifying the flash do to system or software errors. On ‘2xx devices, the correct setting of the LOCKA bit should also be checked.
- Applying the standard RESET sequence (see Figure 1-1) forces the MSP430 to start with the user reset vector at address 0FFFEh.

2.7 Password Protection

The password protection prohibits every command that potentially allows direct or indirect data access. Only the unprotected commands like mass erase and RX password (optionally, TX BSL version and change baud rate) can be performed without prior receipt of the correct password after BSL entry.

Applying the RX password command for receiving the correct password unlocks the remaining commands.

Once it is unlocked, it remains unlocked until initiating another BSL entry.

The password itself consists of the 16 interrupt vectors located at addresses FFE0h to FFFFh (256 bits), starting with the first byte at address FFE0h. After mass erase and with unprogrammed devices, all password bits are logical high (1).

BSL versions 2.00 and higher have enhanced security features. These features are controlled by the flash data word located beneath the interrupt vector table addresses (e.g., for the MSP430F2131, address 0xFFDE). If this word contains:

- 0x0000: The flash memory is not erased if an incorrect BSL password has been received by the target.
- 0xAA55: The BSL is disabled. This means that the BSL is not started with the default initialization sequence shown in Section 1.3.
- All other values: If an incorrect password is transmitted, the entire flash memory address space is erased automatically.

NOTE: The user must take care of password update after modifying the interrupt vectors and initiating another BSL session. It is also strongly recommended to initialize unused interrupt vectors to increase data security.

2.8 Code Protection Fuse

Once the JTAG fuse (code protection fuse) is blown, no further access to the JTAG/test feature is possible. The only way to get any memory read/write access is via the bootstrap loader by applying the correct password.

However, it is not possible for the BSL to blow the JTAG fuse. If fuse blowing is needed, use JTAG programming techniques.

2.9 BSL Internal Settings and Resources

The following paragraphs describe BSL internal settings and resources. Because the same device may have implemented different BSL versions, it is very important for the BSL communication program to know the settings and resources. Resources could be either device dependent (e.g., RX/TX pins) or BSL-version dependent (e.g., byte/word access). The following paragraphs describe the possible variations.

2.9.1 Chip Identification and BSL Version

The upper 16 bytes of the boot-ROM (0FF0h to 0FFFh) hold information about the device and BSL version number in BCD representation. This is common for all devices and BSL versions:

- 0FF0h to 0FF1h: Chip identification (e.g., F413h for an 'F41x device).
- 0FFAh to 0FFBh: BSL version number (e.g., 0130h for BSL version V1.30).

See the MSP430 device/BSL version assignment in Chapter 5.

2.9.2 Vectors to Call the BSL Externally

The entry part of the boot ROM holds the calling vectors for BSL access by program:

- 0C00h: Vector for cold-start (mnemonic: BR &0C00h) (recommended)
- 0C02h: Vector for warm-start (mnemonic: BR &0C02h). V1.30 or higher.
- 0C04h: Vector(s) for future use. This table is expandable.

NOTE: A warm-start does not modify the stack pointer. Additionally, the status register for the BSL is not cleared, which could cause a warm started BSL to come up in an unlocked state. Warm start possibility exists only for highly specialized instances where it is absolutely mandatory that a running application be returned to after a BSL session without resetting the device. In almost all cases it is better to start the BSL from user code by calling the cold start vector.

2.9.3 Initialization Status

When activating the BSL, the following settings take effect:

- Stop Watchdog Timer
- Disable all interrupts (GIE = 0)
- *V1.10*
  The stack pointer is not modified, except when it points to an excluded memory area. If so, it is initialized to 021Ah.
  V1.30 or higher
  The stack pointer is not modified if the BSL is called by program via the warm-start vector. It is initialized to 0220h if the BSL starts via the BSL RESET sequence or is called by program via the cold-start vector.
- *F1xx*
  Determine basic clock module so that minimum frequency is 1.5 MHz:
  BCSCTL1 = 85h (RSEL = 5, XT2Off = 1)
  DCOCTL = 80h (DCO = 4, MOD = 0)
  BCSCTL2 = 00h only at cold-start
  SR: SCG1 = 00h (SMCLK on) only at cold-start
  F2xx
  Determine basic clock module so that minimum frequency is 1.5 MHz:
  BCSCTL1 = 88h (RSEL = 8, XT2Off = 1)
  DCOCTL = 80h (DCO = 4, MOD = 0)
  BCSCTL2 = 00h only at cold-start
  SR: SCG1 = 00h (SMCLK on) only at cold-start
  F4xx
  Determine FLL oscillator/system clock so that minimum frequency is 1.5 MHz:
  SCFI0 = 00h (D = 0, FN_x = 0)
  SCFI1 = 98h (N_DCO)
  SCFQCTL: (M = 0)
  SR: SCG0 = 1 (FLL loop control off)
  FLL_CTL1 = 00h only at cold-start
- SW-UART: Timer_A operates in continuous mode with MCLK source (Div = 1)
  CCR0 used for compare
  CCTL0 used for polling of CCIFG0
- TX pin is set to output HI for RS232 idle state
- RX pin is set to input
- Password-protected commands are locked (only at cold-start)

After system initialization, the BSL is ready for operation and waits for the first synchronization sequence (SS) followed by a data frame containing the first BSL command.

2.9.4 Memory Allocation and Resources

- The BSL program code is located in the boot-ROM area 0C00h to 0FEFh.
- Addresses 0FF0h to 0FFFh hold the device identification.
- The BSL variables occupy the RAM area
  - 0200h to 0213h (V1.10)
  - 0200h to 0219h (V1.30 or higher)
- The BSL stack occupies the RAM area
  - 0214h to 0219h (V1.10)
  - 021Ah to 021Fh (V1.30 or higher, only at cold-start)
- The working registers used are:
  - R5 to R9 (V1.30 or lower) or
  - R5 to R10 (V1.40) or
  - R5 to R11 (V1.60) or
  - R5 to R14 (V2.00 or higher)

  Their contents are not buffered.
  - F1xx/F2xx:
    - The basic clock module registers used are:
      - DCOCTL at address 056h
      - BCSCTL1 at address 057h
  - F4xx:
    - The FLL oscillator/system clock registers used are:
      - SCFI0 at address 050h
      - SCFI1 at address 051h
      - SCFQCTL at address 052h
    - The Timer_A control registers used are:
      - TACTL at address 0160h
      - CCTL0 at address 0162h
      - TAR at address 0170h
      - CCR0 at address 0172h
    - The flash control registers used are:
      - FCTL1 at address 0128h
      - FCTL2 at address 012Ah
      - FCTL3 at address 012Ch
- No interrupt service is affected.

Flash-Based Bootstrap Loader Protocol

3.1 BSL Data Packet

The BSL data packet has a layered structure. The BSL core command contains the actual command data to be processed by the BSL. In addition the standard BSL commands, there can be wrapper data before and after each core command known as the peripheral interface code (PI Code). This wrapper data is information that is specific to the peripheral and protocol being used, and it contains information that allows for correct transmission of the BSL core command. Taken together, the wrapper and core command constitute a BSL data packet.

PI Code	BSL Core Command	PI Code

3.2 UART Peripheral Interface (PI)

3.2.1 Wrapper

The default BSL430 programmed in each non-USB MSP430F5xx device communicates using a UART peripheral interface (PI). The UART protocol interface has the format shown in Table 3-1. All numbers are in hexadecimal format.

Table 3-1. UART Protocol Interface

Header	Length	Length	BSL Core Command	CKL	CKH	ACK
0x80	NL	NH	See Table 3-4	CKL	CKH	(ACK) from BSL

3.2.2 Abbreviations

CKL, CKH

CRC checksum high and low bytes. The checksum is computed on bytes in BSL core command section only. The CRC is computed using the MSP430F5xx CRC module specification (see the CRC chapter of the *MSP430x5xx User's Guide* (SLAU208) for implementation details).

NL, NH

Number of bytes in BSL core data packet, broken into high and low bytes.

ACK

Sent by the BSL after the packet is received to acknowledge receiving the data correctly. This does not imply the BSL core data is a correct command or that it was executed correctly. ACK signifies only that the packet was formatted as expected and had a correct checksum.

NOTE: If the PI encounters an error at any stage of receiving the packet, it immediately responds with the appropriate error message.

3.2.3 Messages

The peripheral interface section of the BSL430 software parses the wrapper section of the BSL data packet. If there are errors with the data transmission, an error message is sent immediately. An ACK is sent after all data has been successfully received and does not mean that the command has been correctly executed (or even that the command was valid) but, rather, that the data packet was formatted correctly and passed on to the BSL core software for interpretation.

Table 3-2. UART Error Messages

Data	Meaning
0x00	ACK
0x51	Header incorrect. The packet did not begin with the required 0x80 value.
0x52	Checksum incorrect. The packet did not have the correct checksum value.
0x53	Packet size zero. The size for the BSL core command was given as 0.
0x54	Packet size exceeds buffer. The packet size given is too big for the RX buffer.
0x55	Unknown error
0x56	Unknown baud rate. The supplied data for baud rate change is not a known value.

3.2.4 Interface Specific Commands

The Timer UART protocol interface also accepts the following command when it is transmitted as BSL core data.

BSL Command	CMD	AL	AM	AH	Data
Change baud rate	0x52	–	–	–	D1

3.2.4.1 Change Baud Rate

This command changes the baud rate for all subsequently received data packets. The command is acknowledged with either a single ACK or error byte. No subsequent message packets can be expected.

D1: Valid values

- 0x02: 9600
- 0x03: 19200
- 0x04: 38400
- 0x05: 57600
- 0x06: 115200

3.3 USB Peripheral Interface

3.3.1 Wrapper

The Peripheral Interface for the USB Bootstrap loader has the following wrapper format. There are no interface specific commands or replies for the USB BSL. The only variable byte, NL, should describe the number of bytes contained in the BSL Core Command packet.

Table 3-3. USB Peripheral Interface

Header	Length	BSL Core Command
0x3F	NL	See Table 3-4

3.3.2 *Hardware Requirements*

The USB Peripheral Interface requires the use of a high frequency oscillator on XT2. For the BSL to function properly, the oscillator can be 24 MHz, 12 MHz, 8 MHz, or 4 MHz.

3.4 BSL Core Command Structure

The BSL core command is transmitted in the format shown in Table 3-4. All numbers are in hexadecimal format.

NOTE: See Section 5.1 for using the following commands with the BSL in the MSP430F5438 (non-A version).

Table 3-4. BSL Core Commands

BSL Command	CMD	AL	AM	AH	Data
RX Data Block	0x10	(AL)	(AM)	(AH)	D1 ... Dn
RX Data Block Fast	0x1B	(AL)	(AM)	(AH)	D1 ... Dn
RX Password	0x11	–	–	–	D1 ... D33
Erase Segment	0x12	(AL)	(AM)	(AH)	–
Unlock/Lock Info	0x13	–	–	–	–
Reserved	0x14	–	–	–	–
Mass Erase	0x15	–	–	–	–
CRC Check	0x16	(AL)	(AM)	(AH)	Length (low byte), Length (high byte)
Load PC	0x17	(AL)	(AM)	(AH)	–
TX Data Block	0x18	(AL)	(AM)	(AH)	Length (low byte), Length (high byte)
TX BSL Version	0x19	–	–	–	–
TX Buffer Size	0x1A	–	–	–	–

NOTE: BSLs programmed in Flash which communicate via USB contain only a subset of the above commands. These commands can be used to load in a full BSL into RAM for Flash programming. The commands in this subset are RX DATA BLOCK FAST, RX PASSWORD, and LOAD PC

3.4.1 *Abbreviations*

–

No data required. No delay should be given, and any subsequently required data should be sent as the immediate next byte.

AL, AM, AH

Address bytes. The low, middle, and upper bytes, respectively, of an address.

D1 ... Dn

Data bytes 1 through n (Note: n must be 4 less than the BSL buffer size.)

Length

A byte containing a value from 1 to 255 describing the number of bytes to be transmitted or used in a CRC. In the case of multiple length bytes, they are combined together as described to form a larger value describing the number of required bytes.

3.4.2 Command Descriptions

RX Data Block

The BSL core writes bytes D1 through Dn starting from the location specified in the address fields.

RX Data Block Fast

This command is identical to RX Data Block, except there is no reply indicating the data was correctly programmed. It is used primarily to speed up USB programming.

RX Password

The BSL core receives the password contained in the packet and unlocks the BSL protected commands if the password matches the top 16 words in the BSL interrupt vector table (located between addresses 0xFFE0 and 0xFFFF). When an incorrect password is given, a mass erase is initiated. This means all code flash is erased, **but not Information Memory**.

Erase Segment

The flash segment containing the given address is subjected to an erase.

Unlock/Lock Info

This command causes the INFO_A lock to toggle to either protect or lock the INFO_A segment. See the *MSP430x5xx User's Guide* (SLAU208) for more detail on this lock. This command must be sent before an erase segment command for INFO_A but is not required before a mass erase.

Erase Block

The flash block containing the given address is subjected to an erase.

Mass Erase

All code Flash in the MSP430 is erased. This function **does not** erase Info Memory.

CRC Check

The MSP430 performs a 16-bit CRC check using the CCITT standard. The address given is the first byte of the CRC check. Two bytes are used for the length.

Load PC

Causes the BSL to begin execution at the given address using a CALLA instruction. As BSL code is immediately exited with this instruction, no core response can be expected.

TX BSL Version

BSL transmits its version information (see Section 3.6.3 for more details).

TX Buffer Size

The BSL transmits a value that represents the number of bytes available in its data buffer for sending or receiving BSL core data packets.

3.5 BSL Security

3.5.1 Protected Commands

To protect data within the device, most core commands are protected. A protected command is successfully complete only after the device has been unlocked by sending the RX Password command with the correct password. In addition, commands specific to the peripheral interface are not protected.

Unprotected Commands

RX Password
Mass Erase

Password Protected Commands

RX Data Block to Address (flash or RAM)
TX BSL Version
TX Data Block
Erase Segment
Erase Bank
Set Program Counter
Toggle INFO_A Lock
Erase Main
CRC Check

3.5.2 RAM Erase

At startup, the BSL performs a RAM erase, writing a constant word to certain RAM locations in a device. Usually these is the smallest shared RAM addresses within a family. See Chapter 5 for device specific information.

3.6 BSL Core Responses

The BSL core responses are always wrapped in a peripheral interface wrapper with the identical format to that of received commands. The BSL core can respond in the format shown in Table 3-5. All numbers are in hexadecimal format.

Table 3-5. BSL Core Responses

BSL Response	CMD	Data
Data Block	0x3A	D1 ... Dn
BSL Version	0x3A	D1 ... D4
CRC Value	0x3A	DL, DH
Buffer Size	0x3A	NL, NH
Message	0x3B	MSG

3.6.1 Abbreviations

CMD

A required field used to distinguish between a message from the BSL and a data transmission from the BSL.

MSG

A byte containing a response from the BSL core describing the result of the requested action. This can either be an error code or an acknowledgment of a successful operation. It should be noted, in cases where the BSL is required to respond with data (for example, memory, version, CRC, or buffer size), no successful operation reply occurs, and the BSL core immediately sends the data.

D1, Dx

Data bytes containing the requested data.

DL, DH

Data low and high bytes, respectively, of a requested 16-bit CRC value.

NL, NH

Data bytes describing the length of the buffer size in bytes. To manage sizes above 255, the size is broken up into a low byte and a high byte.

3.6.2 BSL Core Messages

Table 3-6 describes the BSL core messages.

Table 3-6. BSL Core Messages

MSG	Meaning
0x00	Operation Successful
0x01	Flash Write Check Failed. After programming, a CRC is run on the programmed data. If the CRC does not match the expected result, this error is returned.
0x02	Flash Fail Bit Set. An operation set the FAIL bit in the flash controller (see the *MSP430x5xx User's Guide* (SLAU208) for more details on the flash fail bit).
0x03	Voltage Change During Program. The VPE was set during the requested write operation (see the *MSP430x5xx User's Guide* (SLAU208) for more details on the VPE bit).
0x04	BSL Locked. The correct password has not yet been supplied to unlock the BSL.
0x05	BSL Password Error. An incorrect password was supplied to the BSL when attempting an unlock.
0x06	Byte Write Forbidden. This error is returned when a byte write is attempted in a flash area.
0x07	Unknown Command. The command given to the BSL was not recognized.
0x08	Packet Length Exceeds Buffer Size. The supplied packet length value is too large to be held in the BSL receive buffer.

3.6.3 BSL Version Number

The BSL version number is a 4-byte array.

Byte1: BSL Vendor information

TI BSL is always 0x00. Non-TI BSLs may use this area in another manner.

Byte 2: Command Interpreter Version

The version number for the section of code that interprets BSL core commands.

Byte 3: API Version

The version number for the section of code that reads and writes to MSP430 memory.

Reserved bits:

0x80 Indicates this BSL can only execute the following commands:

RX Data Block Fast (and can only write to RAM)

RX Password

Set PC

0x30 Indicates this BSL API interfaces with FRAM:

Byte 4: Peripheral Interface Version

The version number for the section of code that manages communication.

Reserved numbers:

0x00-0x2F: Indicates a Timer_A based UART

0x20-0x4F: Indicates USB

0x50-0x6F: Indicates USCI based UART

0x70-0x8F: Indicates eUSCI based UART

3.6.4 Example Sequences for UART BSL

NOTE: All values in the example sequences are hexadecimal.

Changing baud rate to 9600
Host: 80 02 00 52 02 90 55
BSL: 00

Get buffer size
Host: 80 01 00 1A 8B 52
BSL: 00 80 03 00 3A 04 01 1D 12

Get BSL version
Host: 80 01 00 19 E8 62
BSL: 00 80 05 00 3A 00 01 01 01 6C 4F

RX password to unlock BSL
Host: 80 11 00 11 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 5C 38 4F
BSL: 00 80 02 00 3B 00 60 C4

3.7 BSL Public Functions and Z-Area

The BSL Z-Area is a small section of memory that can be read and invoked from Application code. It is located at memory addresses 0x1000-0x100F.

Memory location 0x1000 contains a jump instruction pointing to the BSL start, it can be used to invoke the BSL from a running application.

Memory location 0x1002 contains a jump to the "BSL Action" function. To invoke the action function 3 parameters are needed. The first parameter is a number describing which function, the second two are simply known values indicating the function was called on purpose.
R12: The function number
R13: 0xDEAD
R14: 0xBEEF

3.7.1 Starting the BSL From an External Application

Setting the program counter to the memory location 0x1000 starts the BSL. The stack is always reset, and RAM is cleared. It should be noted that the GIE bit is not disabled, so this should be done by the calling application if interrupts are not desired and appropriately returned from via "Return to BSL" if they are used.

Because the stack is reset, the location 0x1000 may also be called as a C function, as in the following example code:

```
((void (*)())0x1000)();
```

3.7.2 Function Description

Function number: 2

Function Name: Return to BSL

Description: Any supplied function number calls the return to BSL function. This function can be used is the BSL has written a program into Flash/RAM, started that program via "Set PC", and then the program needs to return to the BSL. This function executes the following code:

```
RETURN_TO_BSL
         POP.W    RET_low         ; remove first word from return addr
         POP.W    RET_high        ; remove second word from return addr

         RETA                     ; should now return to the BSL location
```

Bootstrap Loader Hardware

This chapter describes simple and low-cost hardware and software solutions to access the bootstrap loader functions of the MSP430 flash devices via the serial port (RS-232) of a PC.

4.1 Hardware Description

The low-cost hardware presented in this document (see Figure 1-3) consists mainly of a low-dropout voltage regulator, some inverters, and operational amplifiers. There are also some resistors, capacitors, and diodes. A complete parts list is provided in Section 4.1.4.

The functional blocks are described in more detail in the following subsections.

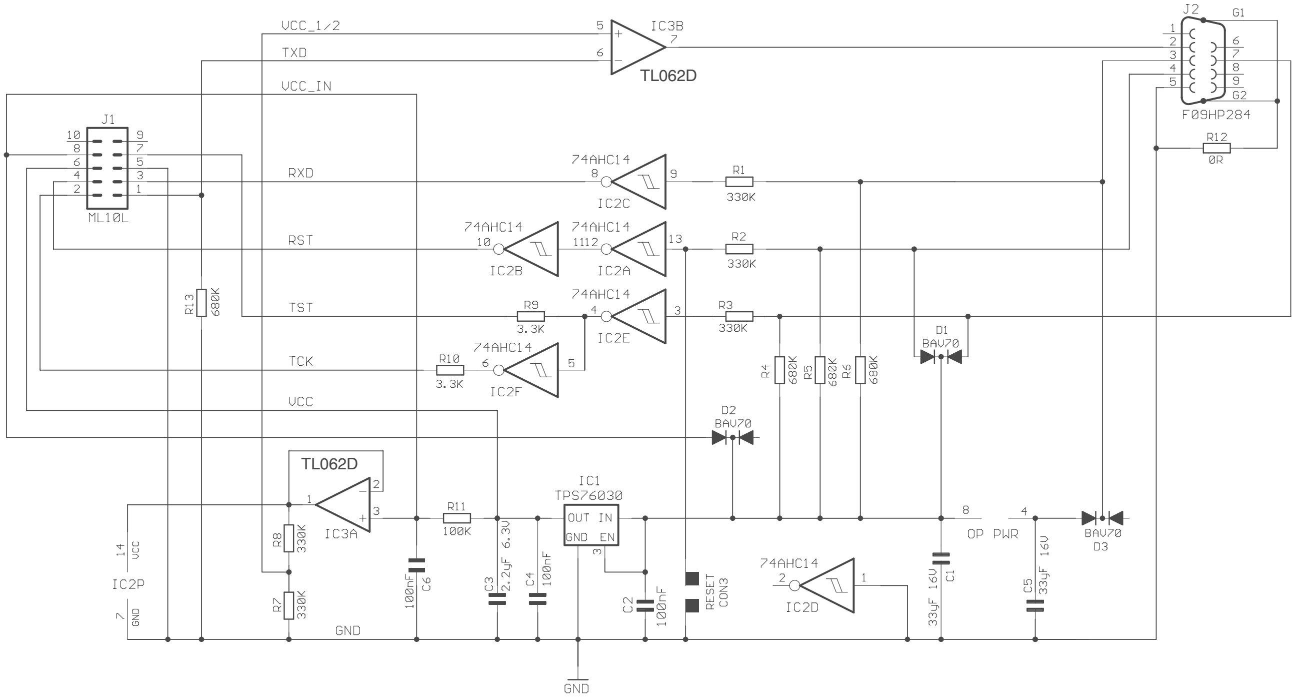

Figure 4-1. Bootstrap Loader Interface Schematic

4.1.1 Power Supply

Power for the bootstrap loader hardware can be supplied via the RS-232 interface. RS-232 signals DTR (pin 7 of the serial connector) and RTS (pin 4 of the serial connector) normally deliver a positive voltage to load capacitor C1 and power to the low-dropout voltage regulator IC1 (TI TPS76030 or LP2980-3.0, or equivalent 3-V low-dropout regulator).

Using a fairly big capacitor, it is possible to draw a short-duration current that is higher than the driving serial port can supply. This feature is required to program the flash memory, for example.

It is also possible to connect an external supply voltage to the hardware via pin 8 of the BSL target connector (J1). Diodes are used to prevent reverse-polarity flow.

4.1.2 Serial Interface

Table 4-1 shows the signals used to communicate with the bootstrap loader (via connector J2). The names reflect the pin function as seen from the PC. For example, the PC receives data via the RxD pin, whereas the bootstrap loader needs to drive this signal.

Table 4-1. Serial-Port Signals and Pin Assignments

PIN NAME	FULL NAME (PC)	9-PIN SUB-D	FUNCTION ON BSL INTERFACE
RxD	Receive data	2	Transmit data to PC
TxD	Transmit data	3	Receive data from PC (and negative supply)
DTR	Data terminal ready	4	Reset control (and positive supply)
RTS	Request to send	7	TEST or TCK control (and positive supply)
GND	Ground	5	Ground

4.1.2.1 Level Shifting

Simple CMOS inverters with Schmitt-trigger characteristics (IC2) are used to transform the RS-232 levels (see Table 4-2) to CMOS levels.

Table 4-2. RS-232 Levels

LOGIC LEVEL	RS-232 LEVEL	RS-232 VOLTAGE LEVEL
1	Mark	–3 V to –15 V
0	Space	3 V to 15 V

The inverters are powered via the operational amplifier IC3A. This amplifier permits adjusting the provided logic level to the requirements of the connected target application. A voltage applied to pin 8 of the BSL target connector (VCC_IN) overrides the default 3-V level provided via IC1 and the 100-kΩ series resistor R11. Thus, the output voltage of the operational amplifier is pulled to the applied voltage VCC_IN.

Depending on the overvoltage protection of the device family selected, the excess voltage is either conducted to V_{CC} (as in the TI '74HC14) or to GND (as in the TI '74AHC14). If the protection diode conducts to V_{CC}, the operational amplifier IC3A needs to compensate for the overvoltage. Therefore, the '74AHC14 device, which conducts to ground (GND), is recommended.

To avoid excessive power dissipation and damage of the protection diodes, series resistors (R1, R2, and R3) are used to limit the input current.

An operational amplifier (IC3B) is used to generate RS-232 levels out of CMOS levels. The level at the positive input is set to $V_{CC}/2$ (1.5 V nominal). If the level at the negative input rises above this level, the output is pulled to the negative supply of the operational amplifier (mark). If the level drops below $V_{CC}/2$, the output is pulled to the positive rail (space).

The positive supply of the operational amplifier is the same as the input to the voltage regulator. A separate capacitor (C5) is used to generate the negative-supply voltage. This capacitor is charged via the receiving signal of the bootstrap loader hardware (pin 3 on SUB-D connector J2).

During an asynchronous serial communication, the combination of stop bit and start bit is used to synchronize sender and receiver. After the transmission of a data byte, the stop bit forces the transmission line into a defined state, which is usually a logic 1 or, in RS-232 terms, a mark. This means that the transmission-line voltage is negative when there is no transmission and the capacitor can be charged. Diodes are used to prevent discharge of the capacitor during transmission.

In very rare circumstances, the data sent to the bootstrap loader interface might hold too many zeros, so that the capacitor C5 required for the negative supply is discharged, causing a malfunction of the interface. (A possible workaround is to send the data in smaller chunks.) But under normal operating conditions, even data containing all zeros does not cause problems.

4.1.2.2 Control of $\overline{\text{RST}}$/NMI and TEST or TCK Pins

The two pins used to invoke the bootstrap loader software of the MSP430—$\overline{\text{RST}}$/NMI and TEST or TCK (for devices without a dedicated TEST pin)—are controlled via the DTR and RTS signals, respectively. These signals also deliver a positive voltage to supply the bootstrap loader hardware.

For devices with a dedicated TEST pin, the levels at RST/NMI and TEST during normal operation are logic 1 and logic 0, respectively. To achieve these levels and to use the corresponding RS-232 signals as power-supply lines, it is necessary to use two inverters (IC1A, IC2B) for the RST/NMI pin and one inverter (IC2E) for the TEST pin.

Devices without the TEST pin require the inverted TEST pin sequence on their TCK pin to invoke the bootstrap loader. Thus, the corresponding signal is inverted (inverter IC2F).

Diodes prevent discharge of capacitor C1 to allow control of the RS-232 lines (RTS and DTR).

4.1.3 Target Connector

Table 4-3. Pin Assignment of Target Connector[1]

PIN	SIGNAL NAME	DEVICES WITH TEST PIN	PIN ON MSP430F13x OR MSP430F14x	PIN ON MSP430F4xx
1	TXD	P1.1	P1.1	P1.0
2	TCK	Do not connect[2]	TCK	TCK
3	RXD	P2.2	P2.2	P1.1
4	RST	$\overline{\text{RST}}$/NMI	$\overline{\text{RST}}$/NMI	$\overline{\text{RST}}$/NMI
5	GND	GND	GND	GND
6	V_{CC} (3.0 V)	V_{CC} [3]	V_{CC} [3]	V_{CC} [3]
7	TST	TEST	Do not connect	Do not connect
8	VCC_IN	V_{CC} [3]	V_{CC} [3]	V_{CC} [3]
9	Not connected	—	—	—
10	Not connected	—	—	—

[1] For device-specific BSL pin information, see the applicable device data sheet.

[2] Signal TCK must not be connected on devices with the TEST pin.

[3] Pin V_{CC} (3.0 V) is a voltage source that can provide a limited current, depending on the serial port driver capability. If an external power supply is used, V_{CC} (3.0 V) must not be connected to the target. In this case, the external supply voltage must be connected to pin VCC_IN. Otherwise, VCC_IN must be unconnected.

4.1.4 Parts List

Table 4-4. Universal BSL Interface Parts List

PART	VALUE/PART NUMBER	PACKAGE	COMMENT
C1	33 µF, 16 V	SMD 7243	
C2	100 nF	SMD 0805	
C3	2.2 µF, 6.3 V	SMD 1206	
C4	100 nF	SMD 0805	
C5	33 µF, 16 V	SMD 7243	
C6	100 nF	SMD 0805	
D1	BAV70	SOT23	High-speed double diode
D2	BAV70	SOT23	High-speed double diode
D3	BAV70	SOT23	
IC1	TPS76030	SOT23/5	TI
IC2	74AHC14	SO14	TI
IC3	TL062D	SO8	TI
R1	330 kΩ	SMD 0805	
R2	330 kΩ	SMD 0805	
R3	330 kΩ	SMD 0805	
R4	680 kΩ	SMD 0805	
R5	680 kΩ	SMD 0805	
R6	680 kΩ	SMD 0805	
R7	330 kΩ	SMD 0805	
R8	330 kΩ	SMD 0805	
R9	3.3 kΩ	SMD 0805	
R10	3.3 kΩ	SMD 0805	
R11	100 kΩ	SMD 0805	
R12	0 kΩ	SMD 0805	
R13	680 kΩ	SMD 0805	
J1	Header 2x5	2X05	Target connector (see Table 4-3)
J2	F09HP284	9-SUB-D female	RS-232 connector
CON3	RESET	SMD0805	Pads to connect an optional reset button

Differences Between Devices and Bootstrap Loader Versions

5.1 5xx/6xx BSL Versions

BSL Version	00.01.01.01
Devices	MSP430F543x family
RAM erased	none
Buffer size for Core Commands	260 bytes
Notable Information	1. The BSL does not contain a mechanism to unlock the BSL area for erasing and writing a custom BSL. This function is not supported in this version. The BSL may be safely erased however. 2. The only supported baud rates are 9600 and 57600. 3. The BSL transmits on TA0.0 and receives on TA0.1. 4. The BSL does not expect a parity bit.
Known Bugs	1. The password for the BSL is the bytes between addresses 0xFFF0 and 0xFFFF. This means that this BSL version expects only 16 bytes for a password in the RX Password command. Sending 32 bytes returns an error. 2. If the addresses 0x20396 or 0x20397 are included in the address range of the CRC command, the returned data is incorrect. 3. The Mass Erase command also erases Info_A. 4. On incorrect password, the device erases all RAM, including its stack. Thus, proper returning of an error code is not assured. 5. The total number of bytes for the CRC function is masked with 0x7FFF and is, therefore, limited to 32767.

BSL Version	00.05.04.03
Devices	MSP430F543xA family
RAM erased	0x1C00–0x5BFF
Buffer size for Core Commands	260 bytes
Notable Information	1. The BSL transmits on TA0.0 and receives on TA0.1.
Known Bugs	1. The baud rate 115k cannot be ensured across all clock, voltage, and temperature variations. 2. To invoke this BSL using the TEST/RESET method, the second low pulse on TEST must be shorter than 15 µs.

BSL Version	00.05.04.52
Devices	CC430F613x family
RAM erased	0x1C00–0x23FF
Buffer size for Core Commands	260 bytes
Notable Information	1. The BSL transmits on UCA0TXD and receives on UCA0RXD.
Known Bugs	1. The baud rate 115k can not be ensured across all clock, voltage, and temperature variations.

BSL Version	00.03.83.33
Devices	MSP430F552x family
RAM erased	0x2400–0x33FF
MAX_BUFFER_SIZE	62 bytes
Notable Information	1. BSL in device is RAM write only. Full BSL must first be loaded into device RAM and started to perform flash write. Only the commands RX PASSWORD, RX DATA BLOCK FAST, and SET PC are supported.
Known Bugs	

BSL Version	00.04.31.71
Devices	MSP430FR573x family
RAM erased	0x1C00–0x1FFF
Buffer size for Core Commands	260 bytes
Notable Information	1. The BSL transmits on UCA0TXD and receives on UCA0RXD. 2. If the BSL is started with MPULOCK bit set, the BSL is unable to clear it, and mass erases could be incomplete.
Known Bugs	

5.2 Special Consideration for ROM BSL Version 1.10

The first official version V1.10 of the ROM BSL requires a small loadable patch sequence, PATCH.TXT, to reliably execute the RX block command. The same procedure must be executed if a loadable BSL is downloaded to such a device. After the BSL has been started, proceed in the following manner:

- RX password (unlock password protection for the following command)
- Load program counter (PC) with 0C22h (initialize stack pointer to a safe address)
- RX password again (unlock password protection for subsequent commands)
- RX data block (code of loadable patch, code section address is 0220h)
- TX data block (code of loadable patch for verification)

From this time forward, the RX block and TX block commands can be used with one restriction: prior to their invocation, the program counter must be set to the start address of the patch.

- Load program counter (PC) with start address 0220h of loadable patch
- RX data block (code to be programmed at any location), or
- TX data block (from any location)

5.3 ROM BSL Known Issues

BSL Command	Erase Main or Info
Versions Affected	1.x
Description	Does not erase information memory when supplied with address in info memory
Workarounds	Use Erase Segment Command

BSL Command	Erase Main or Info
Versions Affected	1.x
Description	Reports failure when first segment of code memory is supplied as address. Erase is properly performed however
Workarounds	Use any other main memory address

BSL Command	Erase Segment
Versions Affected	1.x
Description	Reports failure when used in information memory. Erase is properly performed however
Workarounds	none

In summary, the tables in this chapter show the key information of MSP430 device/BSL version assignment related to their hard/software resources.

Table 5-1. BSL Version 1.10 on 'F13x, 'F14x(1) (excluding Rev AA), 'F11x, and 'F11x1

Device		**F13x** **F14x(1) up to Rev N**	**F11x (obsolete)** **F11x1 (obsolete)**
BSL Version		**1.10**	
BSL vector address	Cold start	0C00h	
	Warm start	—	
Chip ID address		0FF0h	
Chip ID data		F149h	F112h
BSL version address		0FFAh	
BSL version data		0110h	
Mass erase time, nominal (ms)		17.2[1]	
Read/write access at 0000h to FFFFh		Byte	
Verification during write (online)		No	
Stack pointer initialization	SP critical	021Ah	
	SP not critical	Unchanged	
Resources Used by BSL			
Transmit pin (TX)/Receive pin (RX)		P 1.1 / P2.2	
RAM/stack used		0200h to 0219h	
Working registers		R5 to R9	
System clock, affected controls		BCSCTL1, DCOCTL	
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0	

Preparation for software call		`mov    #00h, &CCTL0`
		`bic.b  #02h, &P1SEL`
		`bic.b  #04h, &P2SEL`
		`bic.b  #32h, &IE1`
		`mov.b  #00h, &BCSCTL2`
		`mov    #00h, SR`
		`br     &0C00h`	

Comment 1 Workaround mandatory		Load PATCH.TXT to eliminate ROM bug (see Section 5.2 and Section 2.5).	
Comment 2 Optional for 'F148, 'F149 only: Use loadable BSL (>1 KB RAM required)		Load BL_150S_14x.txt to get all features of V1.60 plus valid erase segment command (see Section 2.5).	
Comment 3 Optional for 'F1x4 to 'F1x9: Use small loadable BSL (<512B RAM required)		Load BS_150S_14x.txt to get some features of V1.60 (see Section 2.5).	

[1] To reach the required mass erase time as specified in the data sheet, the mass erase command must be executed several times.

Table 5-2. BSL Version 1.30 on 'F41x, 'F11x, and 'F11x1

Device		'F41x	'F11x (obsolete) 'F11x1A
BSL Version		**1.30**	
BSL vector address	Cold start	0C00h	
	Warm start	0C02h	
Chip ID address		0FF0h	
Chip ID data		F143h	F112h
BSL version address		0FFAh	
BSL version data		0130h	
Mass erase time, nominal (ms)		206.4	
Read/write access at	0000h to 00FFh	Byte	
	0100h to FFFEh	Word	
Verification during write (online)		No	
Stack pointer initialization	Cold start	0220h	
	Warm start	Unchanged	
Resources Used by BSL			
Transmit pin (TX)		P1.0	P1.1
Receive pin (RX)		P2.1	P2.2
RAM/stack used		0200h to 021Fh	
Working registers		R5 to R9	
System clock, affected controls		SCFI0, SCFI1, SCFQCTL	BCSCTL1, DCOCTL
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0	

Preparation for software call		`mov #00h, &CCTL0`
		`mov.b #00h, &FLLCTL1`
		`br &0C00h`
			`mov #00h, &CCTL0`
			`mov.b #00h, &BCSCTL2`
			`mov #00h, SR`
			`br &0C00h`

Table 5-3. BSL Version 1.40 on 'F12x

Device		**'F122, 'F123x**
BSL Version		**1.40**
BSL vector address	Cold start	0C00h
	Warm start	0C02h
Chip ID address		0FF0h
Chip ID data		F123h
BSL version address		0FFAh
BSL version data		0140h
Mass erase time, nominal (ms)		206.4
Read/write access at	0000h to 00FFh	Byte
	0100h to FFFEh	Word
Verification during write (online)		For addresses 0200h to FFFEh
Stack pointer initialization	Cold start	0220h
	Warm start	Unchanged
Resources Used by BSL		
Transmit pin (TX)		P1.1
Receive pin (RX)		P2.2
RAM/stack used		0200h to 021Fh
Working registers		R5 to R10
System clock, affected controls		BCSCTL1, DCOCTL
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0

Preparation for software call		`mov.b  #00h, &BCSCTL2`
		`mov    #00h, SR`
		`br     &0C00h`

Table 5-4. BSL Version 1.60 on 'F11x2, 'F12x2, 'F43x, 'F44x, 'FE42x, 'FW42x, 'F(G)43x, 'F415, 'F417

Device		**'F1122, 'F1132**	**'F1222, 'F1232**	**'F43x, 'F44x**	**'FE42x, 'FW42x, 'F415, 'F417**	**'F(G)43x**
BSL Version		**1.60**				
BSL vector address	Cold start	0C00h				
	Warm start	0C02h				
Chip ID address		0FF0h				
Chip ID data		1132h	1232h	F449h	F427h	F439h
BSL version address		0FFAh				
BSL version data		0160h				
Mass erase time, nominal (ms)		206.4				
Read/write access at	0000h to 00FFh	Byte				
	0100h to FFFEh	Word				
Verification during write (online)		For addresses 0200h to FFFEh				
Erase check command		Yes (error address 0200h)				
Erase segment command		With erasure verification (error address 0200h)				
TX identification command		Yes				
Change baud rate command		Yes				
Stack pointer initialization	Cold start	0220h				
	Warm start	Unchanged				
Resources Used by BSL						
Transmit pin (TX)		P1.1		P1.0		
Receive pin (RX)		P2.2		P1.1		
RAM/stack used		0200h to 021Fh				
Working registers		R5 to R12				
System clock, affected controls		BCSCTL1, DCOCTL		SCFI0, SCFI1, SCFQCTL		
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0				

Preparation for software call		`mov.b #00h, &BCSCTL2`
		`mov #00h, SR`
		`br &0C00h`
				`mov.b #00h, &FLLCTL1`
				`br    &0C00h`		

Comment	Erase segment command	Addresses 1000h to 11FFh are verified coherently (three segments). Also use erase check command.				

Table 5-5. BSL Version 1.61 on 'F16x, 'F161x, 'F42x0, 'F13x rev AA, 'F14x(1) rev AA

Device		'F16x	'F161x	'F149 Rev AA	'F42x0	'41x2	'47197
BSL Version		**1.61**					
BSL vector address	Cold start	0C00h					
	Warm start	0C02h					
Chip ID address		0FF0h					
Chip ID data		0F169h	0F16Ch	F149h	F427h	4152h	F47Fh
BSL version address		0FFAh					
BSL version data		0161h					
Mass erase time, nominal (ms)		206.4					
Read/write access at	0000h to 00FFh	Byte					
	0100h to FFFEh	Word					
Verification during write (online)		For addresses 0200h to FFFEh					
Erase check command		Yes (error address 0200h)					
Erase segment command		With erasure verification (error address 0200h)					
TX identification command		Yes					
Change baud rate command		Yes					
Stack pointer initialization	Cold start	0220h					
	Warm start	Unchanged					
Resources Used by BSL							
Transmit pin (TX)		P1.1			P1.0		
Receive pin (RX)		P2.2			P1.1		
RAM/stack used		0200h to 021Fh					
Working registers		R5 to R14					
System clock, affected controls		BCSCTL1, DCOCTL			SCFI0, SCFI1, SCFQCTL		
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0					

Preparation for software call		`mov.b #00h, &BCSCTL2`
		`mov #00h, SR`
		`br &0C00h`
					`mov.b #00h, &FLLCTL1`
					`br    &0C00h`		

Comment	Erase segment command	Addresses 1000h to 11FFh are verified coherently (three segments). Also use erase check command.					

Table 5-6. BSL Version 2.02 on 'F21xx, 'F22xx, 'F24x, 'F23x

Device		**'F21x1**	**'F22xx**	**'F24x**
BSL Version		**2.02**		
BSL vector address	Cold start	0C00h		
	Warm start	0C02h[1]		
Chip ID address		0FF0h		
Chip ID data		F213h	F227h	F249h
BSL version address		0FFAh		
BSL version data		0140h		
Mass erase time, nominal (ms)		206.4		
Read/write access at	0000h to 00FFh	Byte		
	0100h to FFFEh	Word		
Verification during write (online)		For addresses 0200h to FFFEh		
Erase check command		Yes (error address 0200h)		
Erase segment command		With erasure verification (error address 0200h)		
TX identification command		Yes		
Change baud rate command		Yes		
Stack pointer initialization	Cold start	0220h		
	Warm start	Unchanged		
Resources Used by BSL				
Transmit pin (TX)		P1.1		
Receive pin (RX)		P2.2		
RAM/stack used		0200h to 021Fh		
Working registers		R5 to R14		
System clock, affected controls		BCSCTL1, DCOCTL		
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0		

Preparation for software call		`mov.b #00h, &BCSCTL2`
		`mov #00h, SR`
		`br &0C00h`		

Comment	Erase segment command	Addresses 1000h to 11FFh are verified coherently (five segments). Also use erase check command.		

[1] The LOCK and LOCKA bits must be cleared by the user application before entering the BSL mov.w #FWKEY+LOCKA, &FCTL3

Table 5-7. BSL Version 2.12/2.13 on 'FG46xx, 'F261x, 'F471xx

Device		**'FG46xx**	**'F471xx**	**'F261x**
BSL Version		**2.12**	**2.13**	**2.13**
BSL vector address	Cold start	0C00h		
	Warm start	0C02h[1]		
Chip ID address		0FF0h		
Chip ID data		F46Fh		F26Fh
BSL version address		0FFAh		
BSL version data		0212h	0213h	0213h
Mass erase time, nominal (ms)		206.4		
Read/write access at	0000h to 00FFh	Byte		
	0100h to FFFEh	Word		
Verification during write (online)		For addresses 0200h to FFFEh		
Erase check command		Yes (error address 0200h)		
Erase segment command		Erase segment command with erasure verification (error address 0200h)		
TX identification command		Yes		
Change baud rate command		Yes		
Stack pointer initialization	Cold start	0224h		
	Warm start	Unchanged		
Resources Used by BSL				
Transmit pin (TX)		P1.0		P1.1
Receive pin (RX)		P1.1		P2.2
RAM/stack used		0200h to 0223h		
Working registers		R4 to R15		
System clock, affected controls		SCFI0, SCFI1, SCFQCTL		
Timer_A, affected controls		TACTL, TAR, CCTL0, CCR0		

Preparation for software call		`mov.b #00h, &FLLCTL1`
		`br &0C00h`		

Comment	Erase segment command	Addresses 1000h to 11FFh are verified coherently (five segments). Also use erase check command.		

[1] The LOCK and LOCKA bits must be cleared by the user application before entering the BSL mov.w #FWKEY+LOCKA, &FCTL3.

5.4 Special Note on the MSP430F14x Device Family BSL

Revision AA of the MSP430F14x devices have a BSL that was updated to Version 1.61. The primary reasons for this change were to increase IP security, ensure correct flash programming and mass erase, and to negate the need for using any patches during programming. To ensure a smooth transition, a programming environment should be updated to take into account the following changes:

- The Mass Erase command needs to be issued only once, and execution of this command takes longer.
- Only word writes are allowed to flash memory.
- No patch or RAM loadable BSL is required.
- BSL has "online verification" to speed programming.
- Memory allocation has changed (see BSL version charts in Section 5.3).
- BSL can return a NAK if Segment Erase fails.
- Transmit BSL Version and Change Baud rate are now unprotected.

Bootstrap Loader PCB Layout Suggestion

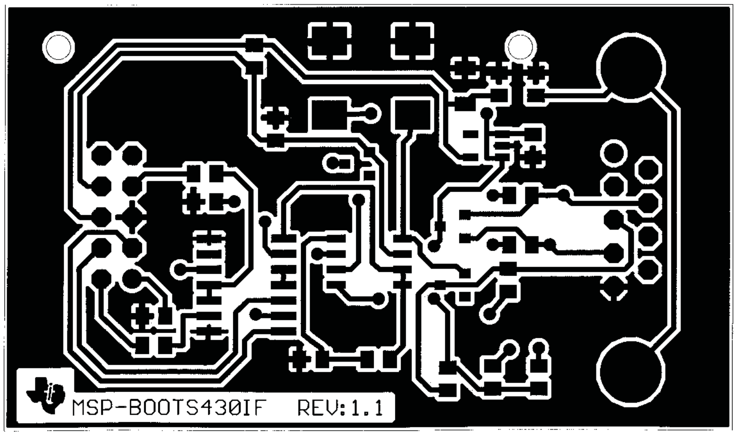

Figure 6-1. Universal BSL Interface PCB Layout, Top

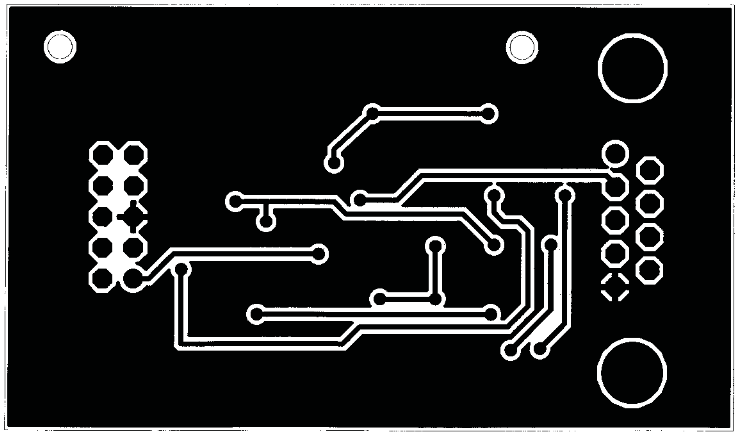

Figure 6-2. Universal BSL Interface PCB Layout, Bottom

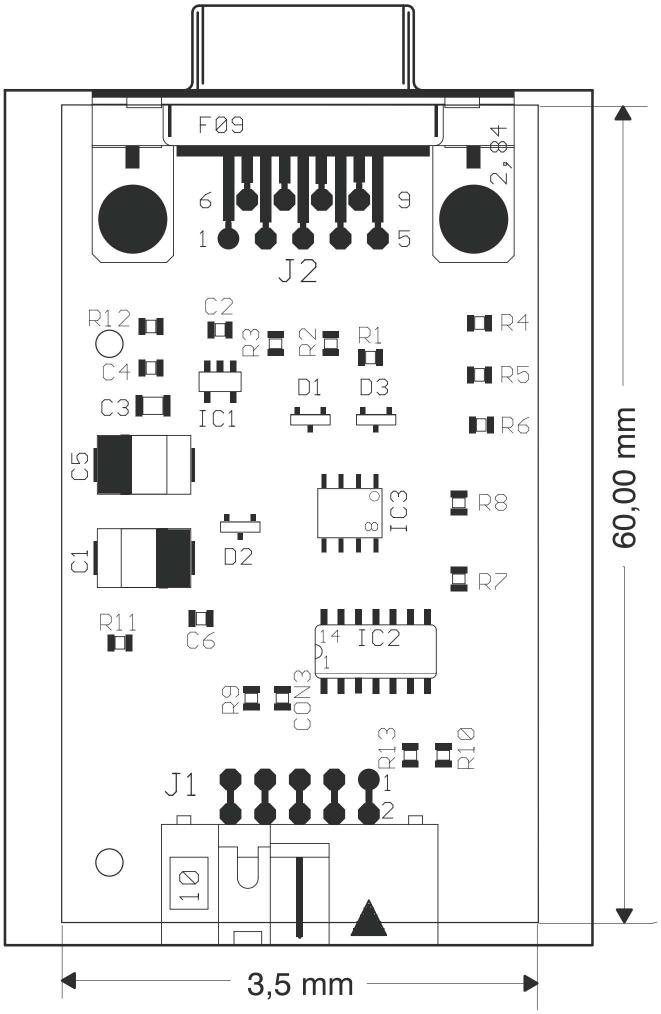

Figure 6-3. Universal BSL Interface Component Placement

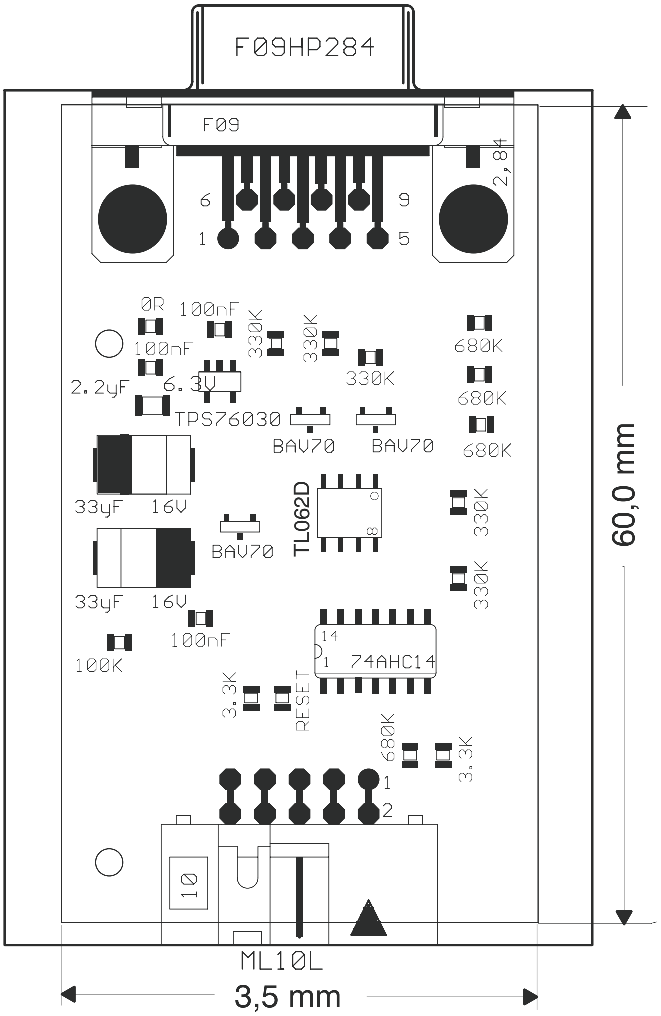

Figure 6-4. Universal BSL Interface Component Placement

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**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap		
Wireless Connctivity	www.ti.com/wirelessconnectivity		
		TI E2E Community Home Page	e2e.ti.com

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