User's Guide
SLAU622–March 2015

MSP432P401R Bootstrap Loader (BSL) User's Guide

The MSP432™ BSL enables users to communicate with embedded memory in the MSP432 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.

Contents

List of Figures

List of Tables

1 Introduction

To use the bootstrap loader, a user-selectable 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 MSP432 memory through the BSL is protected against misuse by a user-defined password.

To avoid accidental overwriting of the BSL code, the code is protected in the flash by default. To prevent unwanted source readout, any BSL command that directly or indirectly allows data reading is password protected. For more information about password-protected commands, refer to **Section 5.3**.

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

2 Other Useful Documentation

MSP432P4xx Technical Reference Manual (SLAU356)

MSP430 BSL Wiki (http://processors.wiki.ti.com/index.php/BSL_(MSP430))

3 BSL Architecture

3.1 Physical Interfaces

The MSP432 BSL is implemented on UART, I²C, and SPI serial interfaces. In MSP432 devices, the BSL can automatically select the interface used to communicate with the device. The specific instance of the peripheral interfaces that is used depends on the selected device and can be found in the device-specific data sheet. This information is also part of the Device Descriptor Table (TLV) table, which is used by the BSL to select the correct instance of the interface. Table 1 shows the BSL interface information in the Device Descriptor Table (TLV). The BSL parses the TLV information and dynamically assesses the ports and interfaces to use. All interface setup calls from the BSL use the driver library in the ROM of the device.

Table 1. BSL Interface Selection Location in TLV

Information	TLV Address
Module ID_BSL tag	Base address of BSL information in TLV (BA)
BSL peripheral interface instance (BSL_PER_IF_SEL)	BA + 0x4
BSL port interface configuration UART (BSL_PORTCNF_UART)	BA + 0x8
BSL port interface configuration SPI (BSL_PORTCNF_SPI)	BA + 0xC
BSL port interface configuration I²C (BSL_PORTCNF_I2C)	BA + 0x10

3.1.1 BSL_PER_IF_SEL TLV Entry

BSL_PER_IF_SEL TLV entry helps the BSL to estimate the following:

1. Interface multiplexing status: If the corresponding interface is directly available on device pins or if it is multiplexed with GPIO.
2. Interface module: Module used to implement the interface (for example, eUSCIA or eUSCIB).
3. Instance of the module used by the BSL.

The BSL_PER_IF_SEL field is split as shown in Figure 1

Figure 1. BSL_PER_IF_SEL

31	30	29	28	27	26	25	24
Reserved							
r	r	r	r	r	r	r	r
23	22	21	20	19	18	17	16
I2C_MUX	Reserved	I2C_MOD		I2C_INST			
r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8
SPI_MUX	Reserved	SPI_MOD		SPI_INST			
r	r	r	r	r	r	r	r
7	6	5	4	3	2	1	0
UART_MUX	Reserved	UART_MOD		UART_INST			
r	r	r	r	r	r	r	r

Table 2. BSL_PER_IF_SEL Field Descriptions

Bit	Field	Value	Description
31-24	Reserved	FFh	Reserved for future use.
23	I2C_MUX		I^2C interface mux status
		1h	I^2C is muxed with GPIO
		0h	I^2C pins are dedicated on device pin out
22	Reserved	1h	Reserved for future use.
21-20	I2C_MOD		Module used to implement I^2C.
		0h	eUSCIB
		1h-3h	Reserved for future IPs implementing I^2C.
19-16	I2C_INST		Instance number of the IP specifically used to implement I^2C.
		0h	Use instance 0 of the IP used for implementing I^2C (for example, eUSCIB0 when I2C_MOD = 00)
		1h	Use instance 1 of the IP used for implementing I^2C (for example, eUSCIB1 when I2C_MOD = 00)
		...	...
		Fh	Use instance 15 of the IP used for implementing I^2C (for example, eUSCIB15, when I2C_MOD = 00)
15	SPI_MUX		SPI interface mux status
		1h	SPI is muxed with GPIO
		0h	SPI pins are dedicated on device pin out
14	Reserved	1h	Reserved for future use.
13-12	SPI_MOD		Module used to implement SPI.
		0h	eUSCIA
		1h	eUSCIB
		2h-3h	Reserved for future IPs implementing SPI.

Table 2. BSL_PER_IF_SEL Field Descriptions (continued)

Bit	Field	Value	Description
11-8	SPI_INST		Instance number of the IP specifically used to implement SPI.
		0h	Use instance 0 of the IP used for implementing SPI (for example, eUSCIB0 when SPI_MOD = 01)
		1h	Use instance 1 of the IP used for implementing SPI (for example, eUSCIB1 when SPI_MOD = 01)
		...	...
		Fh	Use instance 15 of the IP used for implementing SPI (for example, eUSCIB15, when I2C_MOD = 01)
7	UART_MUX		UART interface mux status
		1h	UART is muxed with GPIO
		0h	UART pins are dedicated on device pin out
6	Reserved	1h	Reserved for future use.
5-4	UART_MOD		Module used to implement UART.
		0h	eUSCIA
		1h-3h	Reserved for future IPs implementing UART.
3-0	UART_INST		Instance number of the IP specifically used to implement UART.
		0h	Use instance 0 of the IP used for implementing UART (for example, eUSCIA0, when UART_MOD = 00)
		1h	Use instance 1 of the IP used for implementing UART (for example, eUSCIA1, when UART_MOD = 00)
		...	...
		Fh	Use instance 15 of the IP used for implementing UART (for example, eUSCIA15, when UART_MOD = 00)

3.1.2 BSL_PORTCNF_UART, BSL_PORTCNF_SPI, BSL_PORTCNF_I2C TLV Entries

The BSL needs the GPIO configuration details for the module type, module instance, and mux status of the interface being used (obtained from BSL_PER_IF_SEL). The BSL_PORTCNF_UART, BSL_PORTCNF_SPI, and BSL_PORTCNF_I2C TLV entries provide the BSL with these details.

If eUSCIA is the module used to implement UART, BSL_PORTCNF_UART field is split as shown in Figure 2.

Figure 2. BSL_PORTCNF_UART

31	30	29	28	27	26	25	24
Reserved				Reserved	Reserved	PSEL1_TXD	PSEL1_RXD
r	r	r	r	r	r	r	r
23	22	21	20	19	18	17	16
Reserved				Reserved	Reserved	PSEL0_TXD	PSEL0_RXD
r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8
Reserved			Reserved			BITPOS_TXD	
r	r	r	r	r	r	r	r
7	6	5	4	3	2	1	0
BITPOS_TXD	BITPOS_RXD			UART_PORT			
r	r	r	r	r	r	r	r

Table 3. BSL_PORTCNF_UART Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	3Fh	Reserved for future use.
25	PSEL1_TXD		PSEL1 value for GPIO mux of UCAxTXD line
		0h	PSEL1 bit for UCAxTXD should be programed with a value of 0.
		1h	PSEL1 bit for UCAxTXD should be programed with a value of 1.
24	PSEL1_RXD		PSEL1 value for GPIO mux of UCAxRXD line
		0h	PSEL1 bit for UCAxRXD should be programed with a value of 0.
		1h	PSEL1 bit for UCAxRXD should be programed with a value of 1.
23-18	Reserved	3Fh	Reserved for future use.
17	PSEL0_TXD		PSEL0 value for GPIO mux of UCAxTXD line
		0h	PSEL0 bit for UCAxTXD should be programed with a value of 0.
		1h	PSEL0 bit for UCAxTXD should be programed with a value of 1.
16	PSEL0_RXD		PSEL0 value for GPIO mux of UCAxRXD line
		0h	PSEL0 bit for UCAxRXD should be programed with a value of 0.
		1h	PSEL0 bit for UCAxRXD should be programed with a value of 1.
15-10	Reserved	3Fh	Reserved for future use.
9-7	BITPOS_TXD		Bit position of UCAxTXD in the 8 bit port. For example, in MSP432P401x devices, UCA0TXD is muxed on P1.3. This field therefore reads as 0x3 to represent BIT3.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
6-4	BITPOS_RXD		Bit position of UCAxRXD in the 8 bit port. For example, in MSP432P401x devices, UCA0RXD is muxed on P1.2. This field therefore reads as 0x2 to represent BIT2.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
3-0	UART_PORT		Port number on which the UART IP is muxed.
		0h	Port 1
		1h	Port 2
		...	...
		Fh	Port 16

If eUSCIA or eUSCIB is the module used to implement SPI, BSL_PORTCNF_SPI field is split as shown in Figure 3.

Figure 3. BSL_PORTCNF_SPI

31	30	29	28	27	26	25	24
Reserved				PSEL1_STE	PSEL1_CLK	PSEL1_SIMO	PSEL1_SOMI
r	r	r	r	r	r	r	r
23	22	21	20	19	18	17	16
Reserved				PSEL0_STE	PSEL0_CLK	PSEL0_SIMO	PSEL0_SOMI
r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8
BITPOS_STE				BITPOS_CLK		BITPOS_SIMO	
r	r	r	r	r	r	r	r
7	6	5	4	3	2	1	0
BITPOS_SIMO	BITPOS_SOMI			SPI_PORT			
r	r	r	r	r	r	r	r

Table 4. BSL_PORTCNF_SPI Field Descriptions

Bit	Field	Value	Description
31-28	Reserved	Fh	Reserved for future use.
27	PSEL1_STE		PSEL1 value for GPIO mux of UCxxSTE line
		0h	PSEL1 bit for UCxxSTE should be programed with a value of 0.
		1h	PSEL1 bit for UCxxSTE should be programed with a value of 1.
26	PSEL1_CLK		PSEL1 value for GPIO mux of UCxxCLK line
		0h	PSEL1 bit for UCxxCLK should be programed with a value of 0.
		1h	PSEL1 bit for UCxxCLK should be programed with a value of 1.
25	PSEL1_SIMO		PSEL1 value for GPIO mux of UCxxSIMO line
		0h	PSEL1 bit for UCxxSIMO should be programed with a value of 0.
		1h	PSEL1 bit for UCxxSIMO should be programed with a value of 1.
24	PSEL1_SOMI		PSEL1 value for GPIO mux of UCxxSOMI line
		0h	PSEL1 bit for UCxxSOMI should be programed with a value of 0.
		1h	PSEL1 bit for UCxxSOMI should be programed with a value of 1.
23-20	Reserved	Fh	Reserved for future use.
19	PSEL0_STE		PSEL0 value for GPIO mux of UCxxSTE line
		0h	PSEL0 bit for UCxxSTE should be programed with a value of 0.
		1h	PSEL0 bit for UCxxSTE should be programed with a value of 1.
18	PSEL0_CLK		PSEL0 value for GPIO mux of UCxxCLK line
		0h	PSEL0 bit for UCxxCLK should be programed with a value of 0.
		1h	PSEL0 bit for UCxxCLK should be programed with a value of 1.
17	PSEL0_SIMO		PSEL0 value for GPIO mux of UCxxSIMO line
		0h	PSEL0 bit for UCxxSIMO should be programed with a value of 0.
		1h	PSEL0 bit for UCxxSIMO should be programed with a value of 1.
16	PSEL0_SOMI		PSEL0 value for GPIO mux of UCxxSOMI line
		0h	PSEL0 bit for UCxxSOMI should be programed with a value of 0.
		1h	PSEL0 bit for UCxxSOMI should be programed with a value of 1.

Table 4. BSL_PORTCNF_SPI Field Descriptions (continued)

Bit	Field	Value	Description
15-13	BITPOS_STE		Bit position of UCxxSTE in the 8 bit port. For example, in MSP432P401x devices, UCA0STE is muxed on P1.0. This field therefore reads as 0x0 to represent BIT0.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
12-10	BITPOS_CLK		Bit position of UCxxCLK in the 8 bit port. For example, in MSP432P401x devices, UCA0CLK is muxed on P1.1. This field therefore reads as 0x1 to represent BIT1.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
9-7	BITPOS_SIMO		Bit position of UCxxSIMO in the 8 bit port. For example, in MSP432P401x devices, UCA0SIMO is muxed on P1.3. This field therefore reads as 0x3 to represent BIT3.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
6-4	BITPOS_SOMI		Bit position of UCxxSOMI in the 8 bit port. For example, in MSP432P401x devices, UCA0SOMI is muxed on P1.2. This field therefore reads as 0x2 to represent BIT2.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
3-0	SPI_PORT		Port number on which the SPI IP is muxed.
		0h	Port 1
		1h	Port 2
		...	...
		Fh	Port 16

If eUSCIB is the module used to implement I²C, BSL_PORTCNF_I2C field is split as shown in Figure 4.

Figure 4. BSL_PORTCNF_I2C

31	30	29	28	27	26	25	24
Reserved				Reserved	Reserved	PSEL1_SDA	PSEL1_SCL
r	r	r	r	r	r	r	r
23	22	21	20	19	18	17	16
Reserved				Reserved	Reserved	PSEL0_SDA	PSEL0_SCL
r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8
Reserved			Reserved			BITPOS_SDA	
r	r	r	r	r	r	r	r
7	6	5	4	3	2	1	0
BITPOS_SDA	BITPOS_SCL			I2C_PORT			
r	r	r	r	r	r	r	r

Table 5. BSL_PORTCNF_I2C Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	3Fh	Reserved for future use.
25	PSEL1_SDA		PSEL1 value for GPIO mux of UCBxSDA line
		0h	PSEL1 bit for UCBxSDA should be programed with a value of 0.
		1h	PSEL1 bit for UCBxSDA should be programed with a value of 1.
24	PSEL1_SCL		PSEL1 value for GPIO mux of UCBxSCL line
		0h	PSEL1 bit for UCBxSCL should be programed with a value of 0.
		1h	PSEL1 bit for UCBxSCL should be programed with a value of 1.
23-18	Reserved	3Fh	Reserved for future use.
17	PSEL0_SDA		PSEL0 value for GPIO mux of UCBxSDA line
		0h	PSEL0 bit for UCBxSDA should be programed with a value of 0.
		1h	PSEL0 bit for UCBxSDA should be programed with a value of 1.
16	PSEL0_SCL		PSEL0 value for GPIO mux of UCBxSCL line
		0h	PSEL0 bit for UCBxSCL should be programed with a value of 0.
		1h	PSEL0 bit for UCBxSCL should be programed with a value of 1.
15-10	Reserved	3Fh	Reserved for future use.
9-7	BITPOS_SDA		Bit position of UCBxSDA in the 8 bit port. For example, in MSP432P401x devices, UCB3SDA is muxed on P10.2. This field therefore reads as 0x2 to represent BIT2.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7

Table 5. BSL_PORTCNF_I2C Field Descriptions (continued)

Bit	Field	Value	Description
6-4	BITPOS_SCL		Bit position of UCBxSCL in the 8 bit port. For example, in MSP432P401x devices, UCB3SCL is muxed on P10.3. This field therefore reads as 0x3 to represent BIT3.
		0h	Bit 0
		1h	Bit 1
		2h	Bit 2
		3h	Bit 3
		4h	Bit 4
		5h	Bit 5
		6h	Bit 6
		7h	Bit 7
3-0	I2C_PORT		Port number on which the I^2C IP is muxed.
		0h	Port 1
		1h	Port 2
		...	...
		Fh	Port 16

3.2 UART Protocol Definition

The UART protocol used by the BSL is defined as:

1. Automatic baud-rate detection is run by the BSL slave to determine the initial baud rate of the host that is trying to connect into the BSL. The host can connect into the BSL with the initial baud rates of 9600, 19200, 38400, 57600, or 115200 baud. The sync character for this to be transmitted by the host is 0xFF. The BSL responds with an 0x00 in the BSL response cycle if the communication was successfully established with the BSL.
2. The protocol used for the communication is: Start bit, 8 data bits (LSB first), an even parity bit, 1 stop bit.
3. Handshake for commands is performed by an acknowledge character in the BSL core response format as specified in Table 17.
4. After characters have been received from the MSP432 BSL, there is no additional time delay required before sending new characters.

3.3 I^2C Protocol Definition

The I^2C protocol used by the BSL is defined as:

1. Master must request data from BSL slave.
2. 7-bit addressing mode is used, and by default, the slave listens to address 0x48. (Can be changed using a boot override).
3. In addition to the I^2C protocol based hardware ACK, handshake for commands is performed by an acknowledge character in the BSL core response format as specified in Table 19
4. After characters have been received from the MSP432 BSL, there is no additional time delay required before sending new characters.
5. Repeated starts are not required by the BSL but can be used.

3.4 SPI Protocol Definition

The SPI protocol used by the BSL is defined as:

1. 4-pin SPI slave mode is used. Data is changed on the first clock edge and captured on the following edge. (CKPH = 0). Clock is high when inactive (CKPL = 1). Slave transmit enable (STE) is active low.
2. Master must request data from the BSL slave by transmitting a dummy 0xFF byte. Dummy transmission is needed to drive the clock lines.
3. 8-bit serial data character format (MSB first transmit and receive) is used.
4. Handshake for commands is performed by an acknowledge character in the BSL core response format as specified in Table 21.

4 BSL Memories

4.1 BSL Memory Layout

Table 6 shows the BSL code space.

Table 6. BSL Code Space

Space	Flash Region	Length (Bytes)	Start Address
Code	Flash information memory: Bank 2 sectors 1 and 2	0x2000	0x0020:2000
Data	SRAM CTL Bank 0	0x800	0x2000:0000

4.2 BSL Memory Considerations

After initialization, the BSL uses RAM between the addresses 0x2000:0000 and 0x2000:07FF for data buffer and local variables. When invoking the BSL from a main application, the contents of RAM may be lost.

4.3 BSL Invocation

The MSP432 BSL is invoked by any of the three following approaches. Any one of the conditions is a sufficient condition for BSL entry.

1. The BSL is called by the bootcode when the application memory is erased. Bootcode reads out addresses 0x0 and 0x4 from the application flash memory and compares it with 0xFFFFFFFF to determine whether or not the application memory is erased.
2. The BSL is called by application software (refer to Section 4.3.1).
3. The BSL is called by the device bootcode by applying a hardware entry sequence (refer to Section 4.3.2).

4.3.1 Software BSL Invocation

The BSL has an API table at address 0x0020:2000 that the application can call. The table contains the address of the function that starts the BSL execution. The application needs to pass a certain set of parameters that determines the following:

1. BSL interface (automatic, UART, I^2C, SPI).
2. BSL I^2C slave address (default = 0x48).

The API table contains the address of the BSL entry function. This function takes a 32-bit argument that is formatted the same as the BSL hardware invoke parameters in the Boot Override Flash Mailbox (see Table 7).

Table 7. BSL Entry Function Parameters

Bit	Value
31	N/A for BSL as entry function parameter.
30:26	Reserved. Default should be 0x1F, but N/A for BSL as entry function parameter.
25:16	I^2C slave address. Default = 0x48.
15:13	Interface selection. 7h = Automatic 6h = UART 5h = SPI 4h = I^2C 3h-0h = Reserved for future expansion; defaults to automatic mode.
12	N/A for BSL as entry function parameter.
11:7	Reserved. Default should be 0x1F, but N/A for BSL as entry function parameter.
6:4	N/A for BSL as entry function parameter.
3:0	N/A for BSL as entry function parameter.

The following C code example demonstrates how the address from the API table is used to start the BSL and pass the BSL entry function parameters by the application:

```
#define BSL_PARAM           0xFC48FFFF  // I2C slave address = 0x48, Interface selection = Auto
#define BSL_API_TABLE_ADDR  0x00202000  // Address of BSL API table
#define BSL_ENTRY_FUNCTION  (*((uint32_t *)BSL_API_TABLE_ADDR))

((void (*)())BSL_ENTRY_FUNCTION)((uint32_t)BSL_PARAM);  // Call the BSL with given BSL parameters
```

4.3.2 Hardware BSL Invocation

Hardware invocation on the MSP432 BSL differs from the hardware invocation on the MSP430 devices. On MSP432 devices, the BSL invocation takes advantage of the boot-override mechanism. (See the SYSCTL chapter in the *MSP432P4xx Technical Reference Manual* (SLAU356) for details on boot overrides). The boot-override command used for setting up BSL for hardware invocation is BSL_PARAMS. Table 8 provides the flash mailbox structure for the boot-overrides possible in MSP432.

Table 8. Boot Override Flash Mailbox

Mailbox Offset	Group	Description	Value
0x0		MB_START	Mail box start 0x0115ACF6
0x4	GEN_PARAMS	CMD	Command for Boot override operations.
0x8	FACTORY_RESET	ACK	Acknowledgment for this command
0xC		Reserved	0xFFFFFFFF
0x10	JTAG_SWD_LOCK_PARAMS	JTAG_SWD_LOCK_SECEN	JTAG and SWD Lock Enable Disable = 0xFFFFFFFF Enable = 0x00000000
0x14-0x20		JTAG_SWD_LOCK_AES_INIT_VECT[0-3]	JTAG and SWD lock AES initialization vector for AES-CBC to be used for enrypted updates
0x24-0x40		JTAG_SWD_LOCK_AES_SECKEYS[0-7]	JTAG and SWD lock AES CBC security Keys 0-7. This should be the key that is used to generate the ENCPAYLOAD when the user intends to do an upgrade. 0xFFFFFFFF when disabled.
0x44-0x50		JTAG_SWD_LOCK_UNENC_PWD[0-3]	JTAG and SWD lock unencrypted password 0xFFFFFFFF when security is disabled.
0x54		ACK	Acknowledgment for this command
0x58-0x5C		Reserved	Reserved
0x60	SEC_ZONE0_PARAMS	SEC_ZONE0_SECEN	Disable = 0xFFFFFFFF Enable = 0x00000000
0x64		SEC_ZONE0_START_ADDR	Start address of IP protected secure zone 0. Should be aligned to 4KB boundary.
0x68		SEC_ZONE0_LENGTH	Length of IP protected secure zone 0 in number of bytes. Should be multiples of 4KB flash sector size.
0x6C-0x78		SEC_ZONE0_AESINIT_VECT[0-3]	IP protected secure zone 0 AES initialization vector for AES-CBC to be used for Encrypted Updates.
0x7C-0x98		SEC_ZONE0_SECKEYS[0-7]	AES-CBC security keys. This should be the key that is used to generate the ENCPAYLOAD when the user intends to do an upgrade. 0xFFFFFFFF when IP protected secure zone 0 security disabled.
0x9C-0xA8		SEC_ZONE0_UNENC_PWD[0-3]	Unencrypted password for authentication. 0xFFFFFFFF when IP protected secure zone 0 security disabled.
0xAC		SEC_ZONE0_ENCUPDATE_EN	Disable = 0xFFFFFFFF Enable = 0x0000000
0xB0		SEC_ZONE0_DATA_EN	Disable = 0xFFFFFFFF Enable = 0x00000000
0xB4		ACK	Acknowledgment for this command
0xB8-BC		RESERVED	0xFFFFFFFF

Table 8. Boot Override Flash Mailbox (continued)

Mailbox Offset	Group	Description	Value
0xC0	SEC_ZONE1_PARAMS	SEC_ZONE1_SECEN	Disable = 0xFFFFFFFF Enable = 0x00000000
0xC4		SEC_ZONE1_START_ADDR	Start address of IP protected secure zone 1. Should be aligned to 4KB boundary.
0xC8		SEC_ZONE1_LENGTH	Length of IP protected secure zone 1 in number of bytes. Should be multiples of 4KB flash sector size.
0xCC-0xD8		SEC_ZONE1_AESINIT_VECT[0-3]	IP protected secure zone 1 AES initialization vector for AES-CBC to be used for Encrypted Updates.
0xDC-0xF8		SEC_ZONE1_SECKEYS[0-7]	AES-CBC security keys. This should be the key that is used to generate the ENCPAYLOAD when the user intends to do an upgrade. 0xFFFFFFFF when IP protected secure zone 1 security disabled.
0xFC-0x108		SEC_ZONE1_UNENC_PWD[0-3]	Unencrypted password for authentication. 0xFFFFFFFF when IP protected secure zone 1 security disabled.
0x10C		SEC_ZONE1_ENCUPDATE_EN	Disable = 0xFFFFFFFF Enable = 0x0000000
0x110		SEC_ZONE1_DATA_EN	Disable = 0xFFFFFFFF Enable = 0x00000000
0x114		ACK	Acknowledgment for this command
0x118-0x11C		RESERVED	0xFFFFFFFF
0x120	SEC_ZONE2_PARAMS	SEC_ZONE2_SECEN	Disable = 0xFFFFFFFF Enable = 0x00000000
0x124		SEC_ZONE2_START_ADDR	Start address of IP protected secure zone 2. Should be aligned to 4KB boundary.
0x128		SEC_ZONE2_LENGTH	Length of IP protected secure zone 2 in number of bytes. Should be multiples of 4KB flash sector size.
0x12C-0x138		SEC_ZONE2_AESINIT_VECT[0-3]	IP protected secure zone 2 AES initialization vector for AES-CBC to be used for Encrypted Updates.
0x13C-0x158		SEC_ZONE2_SECKEYS[0-7]	AES-CBC security keys. This should be the key that is used to generate the ENCPAYLOAD when the user intends to do an upgrade. 0xFFFFFFFF when IP protected secure zone 2 security disabled.
0x15C-0x168		SEC_ZONE2_UNENC_PWD[0-3]	Unencrypted password for authentication. 0xFFFFFFFF when IP protected secure zone 2 security disabled.
0x16C		SEC_ZONE2_ENCUPDATE_EN	Disable = 0xFFFFFFFF Enable = 0x0000000
0x170		SEC_ZONE2_DATA_EN	Disable = 0xFFFFFFFF Enable = 0x00000000
0x174		ACK	Acknowledgment for this command
0x178-0x17C		RESERVED	0xFFFFFFFF

Table 8. Boot Override Flash Mailbox (continued)

Mailbox Offset	Group	Description	Value	
0x180	SEC_ZONE3_PARAMS	SEC_ZONE3_SECEN	Disable = 0xFFFFFFFF Enable = 0x00000000	
0x184		SEC_ZONE3_START_ADDR	Start address of IP protected secure zone 3. Should be aligned to 4KB boundary.	
0x188		SEC_ZONE3_LENGTH	Length of IP protected secure zone 3 in number of bytes. Should be multiples of 4KB flash sector size.	
0x18C-0x198		SEC_ZONE3_AESINIT_VECT[0-3]	IP protected secure zone 3 AES initialization vector for AES-CBC to be used for Encrypted Updates.	
0x19C-0x1B8		SEC_ZONE3_SECKEYS[0-7]	AES-CBC security keys. This should be the key that is used to generate the ENCPAYLOAD when the user intends to do an upgrade. 0xFFFFFFFF when IP protected secure zone 3 security disabled.	
0x1BC-0x1C8		SEC_ZONE3_UNENC_PWD[0-3]	Unencrypted password for authentication. 0xFFFFFFFF when IP protected secure zone 3 security disabled.	
0x1CC		SEC_ZONE3_ENCUPDATE_EN	Disable = 0xFFFFFFFF Enable = 0x0000000	
0x1D0		SEC_ZONE3_DATA_EN	Disable = 0xFFFFFFFF Enable = 0x00000000	
0x1D4		ACK	Acknowledgment for this command	
0x1D8-0x1DC		RESERVED	0xFFFFFFFF	
0x1E0	BSL_PARAMS	BSL Enable	Disable = 0xFFFFFFFF Enable = 0x00000000	
0x1E4		BSL Start Address	Contains the pointer to the BSL function. Bootcode reads this location and uses this as a function pointer. Default = TI BSL API table address.	
0x1E8		BSL hardware invoke parameters	Hardware invoke field. Bootcode will jump to the BSL after matching the values in the field with the corresponding port pins of the device.	
			Bit 31	1h = Disable 0h = Enable
			Bits 30:26	Reserved. Default should be 0x1F.
			Bits 25:16	I^2C slave address. Default = 0x48.
			Bits 15:13	Interface selection. 7h = Automatic 6h = UART 5h = SPI 4h = I^2C 3h-0h = Reserved for future expansion; Defaults to automatic mode.
			Bit 12	Polarity of port pin. 0h = Low 1h = High
			Bits 11:7	Reserved for future expansion. Default should be 0x1F.
			Bits 6:4	Pin number of the port to be used for BSL entry. 0h = BIT0 1h = BIT1 2h = BIT2 3h = BIT3 4h = BIT4 5h = BIT5 6h = BIT6 7h = BIT7
			Bits 3:0	Port to be used for HW BSL entry sequence. 0h = P1 1h = P2 2h = P3 3h-Fh = Reserved
0x1EC-0x1F0		Reserved	0xFFFFFFFF	
0x1F4		ACK	Acknowledgment for this command	

Table 8. Boot Override Flash Mailbox (continued)

Mailbox Offset	Group	Description	Value
0x1F8	JTAG_SWD_LOCK_ ENC_UPDATE	JTAG_SWD_LOCK_ENCPAYLOADADDR	Start address where the payload is loaded in the device.
0x1FC		JTAG_SWD_LOCK_ENCPAYLOADLEN	Length of the encrypted payload in bytes. This value should be a multiple of 4KB flash sector size.
0x200		JTAG_SWD_LOCK_DST_ADDR	Destination address where the final data needs to be stored into the device. Should be aligned to 4KB boundary.
0x204		ACK	Acknowledgment for this command
0x208		RESERVED	0xFFFFFFFF
0x20C	SEC_ZONE0_UPDATE	SEC_ZONE0_PAYLOADADDR	Start address where the payload is loaded in the device.
0x210		SEC_ZONE0_PAYLOADLEN	Length of the payload in bytes. This value should match the SEC_ZONE0_LENGTH parameter + 128 bits. There is a limitation that the IP protected secure zone update restricts the user to update a full secure zone. User cannot initiate a partial update to the IP protected secure zone.
0x214		ACK	Acknowledgment for this command
0x218		Reserved	0xFFFFFFFF
0x21C	SEC_ZONE1_UPDATE	SEC_ZONE1_ENCPAYLOADADDR	Start address where the payload is loaded in the device.
0x220		SEC_ZONE1_ENCPAYLOADLEN	Length of the payload in bytes. This value should match the SEC_ZONE1_LENGTH parameter + 128 bits. There is a limitation that the IP protected secure zone update restricts the user to update a full secure zone. User cannot initiate a partial update to the IP protected secure zone.
0x224		ACK	Acknowledgment for this command
0x228		Reserved	0xFFFFFFFF
0x22C	SEC_ZONE2_UPDATE	SEC_ZONE2_ENCPAYLOADADDR	Start address where the payload is loaded in the device.
0x230		SEC_ZONE2_ENCPAYLOADLEN	Length of the payload in bytes. This value should match the SEC_ZONE2_LENGTH parameter + 128 bits. There is a limitation that the IP protected secure zone update restricts the user to update a full secure zone. User cannot initiate a partial update to the IP protected secure zone.
0x234		ACK	Acknowledgment for this command
0x238		Reserved	0xFFFFFFFF
0x23C	SEC_ZONE3_UPDATE	SEC_ZONE3_ENCPAYLOADADDR	Start address where the payload is loaded in the device.
0x240		SEC_ZONE3_ENCPAYLOADLEN	Length of the payload in bytes. This value should match the SEC_ZONE3_LENGTH parameter + 128 bits. There is a limitation that the IP protected secure zone update restricts the user to update a full secure zone. User cannot initiate a partial update to the IP protected secure zone.
0x244		ACK	Acknowledgment for this command
0x248		Reserved	0xFFFFFFFF
0x24C		MB_END	Mailbox end 0x011E11D

The hardware based BSL invocation enable is different from the "BSL enable" in theboot-override fields. The hardware based BSL invocation enable will not be affected by using the"BSL Enable" field of the boot override mail box. The user needs to configure hardware based BSLinvoke using the boot-override mailbox with the "BSL hardware invoke parameters".

for example, Assume a user wishes to use the I^2C mode of BSL with a slave address of 0x42, and wishes to use Port 1-bit 0-level 1 as the BSL entry sequence. The user will need to populate the mailbox command of BSL_PARAMS with the following:

1. BSL Enable = 0x0
2. BSL start address = TI BSL API table address.
3. BSL hardware invocation parameters = 0x7C429F80. (Bit 31 = 0 (Enable), Bits 25:16 = 0x42 (I^2C address), Bits 15:13 = 0x4 (I^2C), Bit 12 = 1 (Level 1), Bits 6:4 = 0x0 (BIT0), Bits 3:0 = 0x0 (PORT1)

After this the user needs to issue a reboot-reset into the device. This will enable the Hardware based BSL invocation in the system. From then on the device would enter BSL whenever it finds the sequence as provided in Figure 6 (this figure is applicable for the previous example).

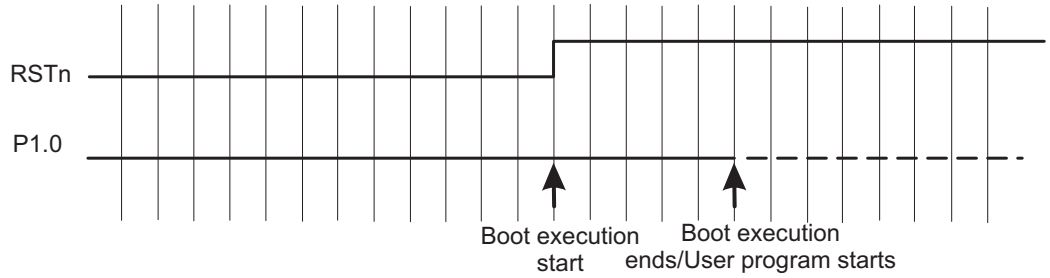

Figure 5. Standard RESET Sequence

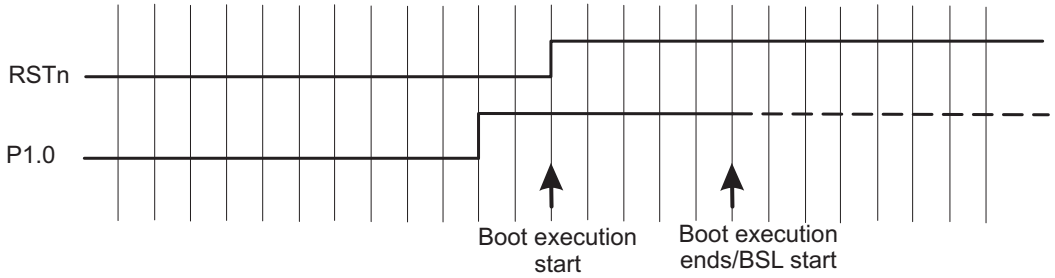

Figure 6. BSL Entry Sequence at Configured GPIO Pin

5 BSL Protocol

5.1 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.

Table 9. BSL Data Packet

PI Code	BSL Core Command	PI code

5.2 BSL Security

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.

5.3 BSL Core Commands

Table 10 shows the format of the BSL core commands.

Table 10. BSL Core Commands

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
RX Data Block	Yes	0x10	(A0)	(A1)	(A2)	-	D1..Dn	Yes
RX Data Block 32	Yes	0x20	(A0)	(A1)	(A2)	(A3)	D1..Dn	Yes
RX Password	No	0x21	-	-	-	-	D1..D256	Yes
Erase Sector	Yes	0x12	(A0)	(A1)	(A2)	-	-	Yes
Erase Sector 32	Yes	0x22	(A0)	(A1)	(A2)	(A3)	-	Yes
Mass Erase	No	0x15	-	-	-	-	-	Yes
Reboot reset	No	0x25	-	-	-	-	-	No
CRC Check	Yes	0x16	(A0)	(A1)	(A2)	-	Length (low byte), Length (high byte)	Yes
CRC Check 32	Yes	0x26	(A0)	(A1)	(A2)	(A3)	Length (low byte), Length (high byte)	Yes
Load PC	Yes	0x17	(A0)	(A1)	(A2)	-	-	Yes
Load PC 32	Yes	0x27	(A0)	(A1)	(A2)	(A3)	-	Yes
TX Data Block	Yes	0x18	(A0)	(A1)	(A2)	-	Length (low byte), Length (high byte)	Yes
TX Data Block 32	Yes	0x28	(A0)	(A1)	(A2)	(A3)	Length (low byte), Length (high byte)	Yes
TX BSL Version	Yes	0x19	-	-	-	-	-	Yes
Change Baud Rate	No	0x52	-	-	-	-	D1	No

A0, A1, A2, A3

Address bytes. These represent the 4 bytes from LSB to MSB, respectively, of a 32 bit address.

D1...Dn

Data bytes 1 through n.

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.

-

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

5.3.1 RX Data Block

Table 11 shows the RX data block format.

Table 11. RX Data Block

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
RX Data Block	Yes	0x10	(A0)	(A1)	(A2)	-	D1…Dn	Yes

Description

The BSL core writes bytes D1 through Dn starting from the location specified in the address fields. This command has been kept for backward compatibility with MSP430 BSL and will allow to address the lower 24 bits of the device.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x10

Command Address

Address where the received data should be written.

Command Data

Command consists of the data D1 through Dn to be written. The command consists of n bytes, where n has maximum 256.

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Write data 0x76543210 to address 0x0001:0000:

Header	Length	Length	CMD	A0	A1	A2	D1	D2	D3	D4	CKL	CKH
0x80	0x08	0x00	0x10	0x00	0x00	0x01	0x10	0x32	0x54	0x76	0x93	0xCA

BSL response for a successful data write:

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.2 RX Data Block 32

Table 12 shows the RX data block 32 format.

Table 12. RX Data Block 32

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
RX Data Block 32	Yes	0x20	(A0)	(A1)	(A2)	(A3)	D1…Dn	Yes

Description

The BSL core writes bytes D1 through Dn starting from the location specified in the address fields. This command allows the BSL to address the device with the full 32 bit range.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x20

Command Address

Address where the received data should be written.

Command Data

Command consists of the data D1 through Dn to be written. The command consists of n bytes, where n has maximum 256.

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Write data 0x76543210 to address 0x0001:0000:

Header	Length	Length	CMD	A0	A1	A2	A3	D1	D2	D3	D4	CKL	CKH
0x80	0x09	0x00	0x20	0x00	0x00	0x01	0x00	0x10	0x32	0x54	0x76	0x66	0x96

BSL response for a successful data write:

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.3 RX Password

Table 13 shows the RX password format.

Table 13. RX Password

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
RX Password	No	0x21	-	-	-	-	D1…D256	Yes

Description

The BSL core receives the password contained in the packet and unlocks the BSL protected commands if the password matches the top 256 bytes in the BSL interrupt vector table (located between addresses 0x0 to 0xFF). When an incorrect password is given, a Boot Override factory reset is initiated. This means all code in flash main memory is erased, but not Information Memory.

When a mass erase is performed, the password is always be 0xFF for all bytes. This is commonly used to gain access to an empty device or to load a new application to a locked device without password.

Protection

This command is not password protected.

Command

0x11

Command Address

N/A

Command Data

The command data is 256 bytes long and contains the device password.

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Unlock a blank device:

Header	Length	Length	CMD	D1	D2	D3	D4	D5	D6
0x80	0x01	0x01	0x21	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D17	D18	D19	D20	D21	D22	D23	D24	D25	D26
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D27	D28	D29	D30	D31	D32	D33	D34	D35	D36
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D37	D38	D39	D40	D41	D42	D43	D44	D45	D46
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D47	...	...	...	...	...	...	..	...	D250
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

D251	D252	D253	D254	D255	D256	CKL	CKH
0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xAD	0x08

BSL response for a successful password:

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.4 Erase Sector

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Erase Sector	Yes	0x12	(A0)	(A1)	(A2)	-	-	Yes

Description

All code flash (Main or Information flash) in the MSP432 in the sector corresponding to the address is erased except for areas defined as IP Protected secure zones. This function does not erase RAM.

Protection

This command is password protected.

Command

0x12

Command Address

Any address (24 bits) in the sector for which erase is to be performed.

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Initiate a Erase sector at address 0x0020:0000:

Header	Length	Length	CMD	A0	A1	A2	CKL	CKH
0x80	0x04	0x00	0x12	0x00	0x00	0x20	0x6D	0x56

BSL response (successful operation):

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.5 Erase Sector 32

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Erase Sector 32	Yes	0x22	(A0)	(A1)	(A2)	(A3)	-	Yes

Description

All code flash (Main or Information flash) in the MSP432 in the sector corresponding to the address is erased except for areas defined as IP Protected secure zones. This function does not erase RAM.

Protection

This command is password protected.

Command

0x22

Command Address

Any address (32 bits) in the sector for which erase is to be performed.

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Initiate a Erase sector at address 0x0020:0000:

Header	Length	Length	CMD	A0	A1	A2	A3	CKL	CKH
0x80	0x05	0x00	0x12	0x00	0x00	0x20	0x00	0x33	0x57

BSL response (successful operation):

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.6 Mass Erase

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Mass Erase	No	0x15	-	-	-	-	-	Yes

Description

All code flash in the MSP432 is erased except for areas defined as IP protected secure zones. This function does not erase Information Memory and RAM.

Protection

This command is not password protected.

Command

0x15

Command Address

N/A

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

See Table 10 for more information on BSL core responses.

Command Example

Initiate a mass erase:

Header	Length	Length	CMD	CKL	CKH
0x80	0x01	0x00	0x15	0x64	0xA3

BSL response (successful operation):

ACK	Header	Length	Length	CMD	MSG	CKL	CKH
0x00	0x80	0x02	0x00	0x3B	0x00	0x60	0xC4

5.3.7 Reboot Reset

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Reboot reset	No	0x25	-	-	-	-	-	No

Description

This command is used to initiate a reboot-reset into the MSP432 system.

Protection

This command is not password protected.

Command

0x25

Command Address

N/A

Command Data

N/A

Command Returns

None. The devices resets after successfully receiving the command.

Command Example

Initiate a reboot reset into MSP432:

Header	Length	Length	CMD	CKL	CKH
0x80	0x01	0x00	0x25	0x37	0x95

BSL Response:

None

5.3.8 CRC Check

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
CRC Check	Yes	0x16	(A0)	(A1)	(A2)	-	Length (low byte), Length (high byte)	Yes

Description

The MSP432 device 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.

Refer to the CRC chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) for more details on the CRC that is used.

This command is retained from MSP430 for backward compatibility and is capable of address device memory with up to 24 bits of address.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x16

Command Address

Address to begin the CRC check.

Command Data

The 16-bit length of the CRC check. D1 is the low byte of the length, and D2 is the high byte of the length.

Command Returns

BSL acknowledgment and a BSL core response with the CRC value.

See Table 10 for more information on BSL core responses.

Command Example

Perform a CRC check from address 0x0000:4400 to 0x0000:47FF (size of 1024):

Header	Length	Length	CMD	A0	A1	A2	D1	D2	CKL	CKH
0x80	0x06	0x00	0x16	0x00	0x44	0x00	0x00	0x04	0x9C	0x7D

BSL response where 0x55 is the low byte of the calculated checksum and 0xAA is the high byte of the calculated checksum:

ACK	Header	Length	Length	CMD	D1	D2	CKL	CKH
0x00	0x80	0x03	0x00	0x3A	0x55	0xAA	0x12	0x2B

5.3.9 CRC Check 32

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
CRC Check 32	Yes	0x26	(A0)	(A1)	(A2)	(A3)	Length (low byte), Length (high byte)	Yes

Description

The MSP432 device 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.

Refer to the CRC chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) for more details on the CRC that is used.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x26

Command Address

Address to begin the CRC check.

Command Data

The 16-bit length of the CRC check. D1 is the low byte of the length, and D2 is the high byte of the length.

Command Returns

BSL acknowledgment and a BSL core response with the CRC value.

See Table 10 for more information on BSL core responses.

Command Example

Perform a CRC check from address 0x0000:4400 to 0x0000:47FF (size of 1024):

Header	Length	Length	CMD	A0	A1	A2	A3	D1	D2	CKL	CKH
0x80	0x07	0x00	0x26	0x00	0x44	0x00	0x00	0x00	0x04	0xF7	0xE6

BSL response where 0x55 is the low byte of the calculated checksum and 0xAA is the high byte of the calculated checksum:

ACK	Header	Length	Length	CMD	D1	D2	CKL	CKH
0x00	0x80	0x03	0x00	0x3A	0x55	0xAA	0x12	0x2B

5.3.10 Load PC

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Load PC	Yes	0x17	(A0)	(A1)	(A2)	-	-	Yes

Description

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

This command is retained from MSP430 for backward compatibility and is capable of address device memory with up to 24 bits of address.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x17

Command Address

Address to set the Program Counter.

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

Load PC always return success or a BSL locked status. This status is sent only if the main application returns to the BSL.

See Table 10 for more information on BSL core responses.

Command Example

Set program counter to 0x0000:4451:

Header	Length	Length	CMD	A0	A1	A2	CKL	CKH
0x80	0x04	0x00	0x17	0x51	0x44	0x00	0xBC	0x66

The BSL does not respond after the application gains control.

5.3.11 Load PC 32

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Load PC 32	Yes	0x27	(A0)	(A1)	(A2)	(A3)	-	Yes

Description

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

Protection

This command is password protected and fails if the password has not been sent.

Command

0x27

Command Address

Address to set the Program Counter.

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with the status of the operation.

Load PC always return success or a BSL locked status. This status is sent only if the main application returns to the BSL.

See Table 10 for more information on BSL core responses.

Command Example

Set program counter to 0x0000:4451:

Header	Length	Length	CMD	A0	A1	A2	A3	CKL	CKH
0x80	0x05	0x00	0x27	0x51	0x44	0x00	0x00	0x8E	0xBC

The BSL does not respond after the application gains control.

5.3.12 TX Data Block

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
TX Data Block	Yes	0x18	(A0)	(A1)	(A2)	-	Length (low byte), Length (high byte)	Yes

Description

The BSL transmits data starting at the command address and with size command data. This command initiates multiple packets if the size is greater than or equal to the buffer size. This command is retained from MSP430 for backward compatibility and is capable of address device memory with up to 24 bits of address.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x18

Command Address

Address to begin transmitting data from.

Command Data

The 16-bit length of the data to transmit. D1 is the low byte of the length, and D2 is the high byte of the length.

Command Returns

BSL acknowledgment and a BSL core response with *n* data packets where *n* is:

$$n = \text{ceiling}\left(\frac{length}{buffer\ size - 1}\right)$$

For example, if 512 bytes are requested, BSL sends two packets: the first packet with a length of 262 (1 CMD + 261 data bytes) and the second with a length of 252 (1 CMD + 251 data bytes).

See Table 10 for more information on BSL core responses.

Command Example

Transmit 4 bytes of data from address 0x0000:1C00:

Header	Length	Length	CMD	A0	A1	A2	D1	D2	CKL	CKH
0x80	0x06	0x00	0x18	0x00	0x1C	0x00	0x04	0x00	0x87	0x81

BSL response where D1..D4 are the data bytes requested:

ACK	Header	Length	Length	CMD	D1	D2	D3	D4	CKL	CKH
0x00	0x80	0x05	0x00	0x3A	0x11	0x33	0x55	0x77	0x90	0x55

5.3.13 TX Data Block 32

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
TX Data Block 32	Yes	0x28	(A0)	(A1)	(A2)	(A3)	Length (low byte), Length (high byte)	Yes

Description

The BSL transmits data starting at the command address and with size command data. This command initiates multiple packets if the size is greater than or equal to the buffer size.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x28

Command Address

Address to begin transmitting data from.

Command Data

The 16-bit length of the data to transmit. D1 is the low byte of the length, and D2 is the high byte of the length.

Command Returns

BSL acknowledgment and a BSL core response with *n* data packets where *n* is:

$$n = \text{ceiling}\left(\frac{length}{buffer\ size - 1}\right)$$

For example, if 512 bytes are requested, BSL sends two packets: the first packet with a length of 262 (1 CMD + 261 data bytes) and the second with a length of 252 (1 CMD + 251 data bytes).

See Table 10 for more information on BSL core responses.

Command Example

Transmit 4 bytes of data from address 0x0000:1C00:

Header	Length	Length	CMD	A0	A1	A2	A3	D1	D2	CKL	CKH
0x80	0x07	0x00	0x28	0x00	0x1C	0x00	0x00	0x04	0x00	0x20	0x4F

BSL response where D1..D4 are the data bytes requested:

ACK	Header	Length	Length	CMD	D1	D2	D3	D4	CKL	CKH
0x00	0x80	0x05	0x00	0x3A	0x11	0x33	0x55	0x77	0x90	0x55

5.3.14 TX BSL Version

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
TX BSL Version	No	0x19	-	-	-	-	-	Yes

Description

BSL transmits its version information.

Protection

This command is password protected and fails if the password has not been sent.

Command

0x19

Command Address

N/A

Command Data

N/A

Command Returns

BSL acknowledgment and a BSL core response with its version number. The data is transmitted as it appears in memory with the following data bytes:

Version Byte	Data (Low Byte, High Byte)
BSL Vendor	D1, D2
Command Interpreter	D3, D4
API	D5, D6
Peripheral Interface	D7, D8
Build ID	D9, D10

See Table 10 for more information on BSL core responses.

Command Example

Request the BSL version:

Header	Length	Length	CMD	CKL	CKH
0x80	0x01	0x00	0x19	0xE8	0x62

BSL response (version 0011.2233.4455.6677.8899 of the BSL):

ACK	Header	Length	Length	CMD	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	CKL	CKH
0x00	0x80	0x0B	0x00	0x3A	0x00	0x11	0x22	0x33	0x44	0x55	0x66	0x77	0x88	0x99	0xCF	0x1D

5.3.15 Change Baud Rate

BSL Command	Protected	CMD	A0	A1	A2	A3	Data	BSL Core Response
Change Baud Rate	No	0x52	-	-	-	-	D1	No

Description

This command changes the baud rate for all subsequently received data packets. The command is acknowledged with either a single ACK or an error byte sent with the old baud rate before changing to the new one. No subsequent message packets can be expected.

Protection

This command is not password protected.

Command

0x52

Command Address

N/A

Command Data

Single byte, D1, that specifies the new baud rate to use.

D1	Baud Rate
0x01	9600
0x03	19200
0x04	38400
0x05	57600
0x06	115200

Command Returns

BSL acknowledgment

Command Example

Change baud rate to 115200:

Header	Length	Length	CMD	D1	CKL	CKH
0x80	0x02	0x00	0x52	0x06	0x14	0x15

BSL Response:

ACK
0x00

5.4 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 14. All numbers are in hexadecimal format.

Table 14. BSL Core Responses

BSL Response	CMD	Data
Data Block	0x3A	D1 ... Dn
BSL Version	0x3A	D1 ... D10
CRC Value	0x3A	DL, DH
Buffer Size	0x3A	NL, NH
Message	0x3B	MSG

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.

5.5 BSL Core Messages

Table 15 describes the BSL core messages

Table 15. BSL Core Messages

MSG	Meaning
0x00	Operation Successful
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.
0x07	Unknown Command. The command given to the BSL was not recognized.

5.6 UART Peripheral Interface (PI)

5.6.1 UART Peripheral Interface Wrapper

The MSP432 UART BSL protocol interface is implemented in multiple packets. The first packet is transmitted to the BSL device and contains the BSL Core Command and its wrapper. This has the format shown in Table 16 . The second packet is received from the BSL device and contains the BSL acknowledgment and the BSL Core Response if one is required by the BSL Core Command that was sent (see Table 8 for more information about what commands return a BSL Core Response). The BSL response is shown in Table 17.

Table 16. UART BSL Command

Header	Length	Length	BSL Core Command	CKL	CKH
0x80	NL	NH	See Table 10	CKL	CKH

Table 17. UART BSL Response

ACK	Header	Length	Length	BSL Core Response	CKL	CKH
ACK from BSL	0x80	NL	NH	See Table 15	CKL	CKH

The BSL acknowledgment indicates any errors in the first packet. If a response other than ACK is received, the BSL Core Response is not sent. It is important that the host programmer check this first byte to determine if more data will be sent.

CKL, CKH

CRC checksum high and low bytes. The checksum is computed on bytes in BSL core command section only. The BSL uses CRC-CCITT for the checksum and computes it using the MSP432 CRC module (see the CRC chapter of the MSP432P401R Family User's Guide (SLAU356) for more details about the CRC hardware).

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.

5.7 I²C Peripheral Interface (PI)

5.7.1 I²C Peripheral Interface Wrapper

The MSP432 I²C BSL protocol interface is implemented in multiple packets. The first packet is sent as a write request to the BSL slave address and contains the BSL Core Command and its wrapper. This has the format shown in Table 18. The second packet is sent as a read request to the BSL slave address and contains the BSL acknowledgment and the BSL Core Response if one is required by the BSL Core Command that was sent (see Section 5.3 for more information about what commands return a BSL Core Response). Table 19 shows the format of this BSL response.

Table 18. I²C BSL Command

I²C	Header	Length	Length	BSL Core Command	CKL	CKH
S/A/W	0x80	NL	NH	See Table 10	CKL	CKH

Table 19. I²C BSL Response

I²C	ACK	Header	Length	Length	BSL Core Response(1)	CKL	CKH	I²C
S/A/R	ACK from BSL	0x80	NL	NH	See Table 15	CKL	CKH	STOP

(1) BSL Core Response is not always included.

The BSL acknowledgment indicates any errors in the first packet. If a response other than ACK is received, the BSL Core Response is **not** sent. It is important that the host programmer check this first byte to determine if more data will be sent.

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 MSP432 CRC module specification (see the CRC chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) 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.

S/A/W

I²C start sequence sent by the host programmer to the MSP432 BSL slave device. This sequence specifies that the host would like to start a write to the device with the specified slave address. See the eUSCI I²C mode chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) for more details on I²C communication.

S/A/R

I²C start or re-start sequence sent by the host programmer to the MSP432 BSL slave device. This sequence specifies that the host would like to start a read from the device with the specified slave address. This does not need to be a re-start, the host programmer can send a stop followed by another start. See the eUSCI I²C mode chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) for more details on I²C communication.

STOP

I²C stop bit indicating the end of an I²C read or write.

5.8 SPI Peripheral Interface

5.8.1 SPI Peripheral Interface

The MSP432 SPI BSL protocol interface is implemented in multiple packets. The first packet is sent as a write request to the BSL slave address and contains the BSL Core Command and its wrapper. This has the format shown in Table 20. The second packet is sent as a read request to the BSL slave address and contains the BSL acknowledgment and the BSL Core Response if one is required by the BSL Core Command that was sent (see Section 5.3 for more information about what commands return a BSL Core Response). Table 21 shows the format of this BSL response.

Table 20. SPI BSL Command

Header	Length	Length	BSL Core Command	CKL	CKH
0x80	NL	NH	See Table 10	CKL	CKH

Table 21. SPI BSL Response

ACK	Header	Length	Length	BSL Core Response[1]	CKL	CKH
ACK from BSL	0x80	NL	NH	See Table 15	CKL	CKH

(1) BSL Core Response is not always included.

The BSL acknowledgment indicates any errors in the first packet. If a response other than ACK is received, the BSL Core Response is **not** sent. It is important that the host programmer check this first byte to determine if more data will be sent.

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 MSP432 CRC module specification (see the CRC chapter of the *MSP432P4xx Technical Reference Manual* (SLAU356) 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.

5.9 BSL Messages

The peripheral interface section of the BSL 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.

The BSL protocol dictates that every BSL data packet sent is responded to with a single byte acknowledgment in addition to any BSL data packets that are sent. Table 22 lists all possible acknowledgment responses from the BSL. If an acknowledgment byte other than ACK is sent, the BSL does not send any BSL data packets. The host programmer needs to check the acknowledgment error and retry transmission.

Table 22. Error Messages

Data	Meaning
0x00	ACK
0x51	Header incorrect. The packet did not begin with the required value of 0x80.
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. (UART only).

6 Low-Power Mode Support

The MSP432 BSL supports low-power of operation. If the device starts up into the BSL, and the BSL does not detect any user input within 10 seconds, the BSL puts the device into the lowest power mode of the device, LPM4.5. After the device BSL enters the LPM4.5 mode of operation, only a pulse on the device $\overline{\text{RST}}$ pin can bring the device back up and go into the BSL.

7 MSP432P4xx BSL Version Information

Table 23. MSP432P4xx BSL Information

BSL Information	Value
Device	MSP432P401R
BSL version	0000.0002.0003.0102.0003
Buffer size for core commands	262 bytes

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

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

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license 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 significant portions 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. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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

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