

User's Guide

MSPM33 Bootloader



ABSTRACT

The MSPM33 Bootloader (also known as BSL) provides a method to program and verify the device memory (Flash) through multiple serial interfaces including UART and I2C.

Table of Contents

1 Introduction.....2

 1.1 Overview of BSL Features.....2

 1.2 Terminology.....2

 1.3 Additional resources.....2

2 BSL Architecture.....3

 2.1 Design.....3

 2.2 BSL Invocation.....4

 2.3 Memory.....5

 2.4 BSL Non-Main Configuration.....6

 2.5 Changing BSL Configuration.....12

3 Bootloader Protocol.....15

 3.1 Packet Format.....15

 3.2 BSL Protocol.....15

 3.3 Bootloader Core Commands.....16

 3.4 Bootloader Core Response.....23

 3.5 Bootloader Security.....26

4 Sample Program Flow with Bootloader.....27

5 Revision History.....29

Trademarks

All trademarks are the property of their respective owners.

1 Introduction

1.1 Overview of BSL Features

The bootstrap loader (BSL) provides a method to program or verify the device memory through a standard UART or I2C serial interface.

Key features of the BSL that are accessible through the serial interface include:

- Programming and erasing flash memory
- Can return a 32-bit CRC of a code or data region (1KB minimum region size) to verify programming
- Can enable code or data read-out (disabled by default)
- Can return a firmware version number via a pointer to the main flash
- Can specify a hardware invoke GPIO
- Access is always protected with a 256-bit password
- Configurable security alert handling for resisting brute force attacks
- ROM-based interface plugins for UART, I2C with auto-detection capability
- UART baud rate configuration with multiple options
- Configurable interface pins for UART and I2C

1.2 Terminology

Bootloader (BSL) - Boot routine used to load data to the device memory

Bootcode (BCR) - Startup Routine that runs after BOOTRST, to configure the device for executing application

BCR configuration - Configuration structure that contains all user configurable parameters for Bootcode, which resides in non-main flash memory

BSL configuration - Configuration structure that contains all user configurable parameters for Bootloader, which resides in non-main flash memory

1.3 Additional resources

1. Technical Reference Manual
 - a. [MSPM33 C-Series Microcontrollers](#)

2 BSL Architecture

2.1 Design

The bootloader will be invoked by Bootcode when a valid bootloader invocation condition is detected. It will be invoked only if the bootloader is enabled in the BSL mode field of the BCR configuration.

Once the bootloader is started, it first executes the "Init" phase where the initial checks of the BSL configuration are done and the device is configured for bootloader operations.

Next the bootloader enters the "Interface Autodetection" phase. In this phase the BSL configures all the available BSL interfaces, if registered. The BSL then polls for the data through all the interfaces one by one. When a valid [connection packet](#) is received in one of the interfaces, that interface will be considered as active interface for further communication and all other interfaces will be disabled. Interface discovery is done for 4 seconds, if no interface has been detected, the device is put in STANDBY mode.

Next the BSL enters the "Command reception" phase. In this phase BSL will be waiting in an infinite loop for command from the host. When a valid command is received, the command is processed and the response from the BSL core is sent back to the host. Then it goes back to the loop and waits for the next command and so on. If the 'Start Application' command is received, the bootloader will trigger a system reset, after which the Bootcode is executed, and the application is invoked. This phase also has a timeout of 4 seconds. If there is no valid command received, the bootloader is locked, and it enters Sleep mode.

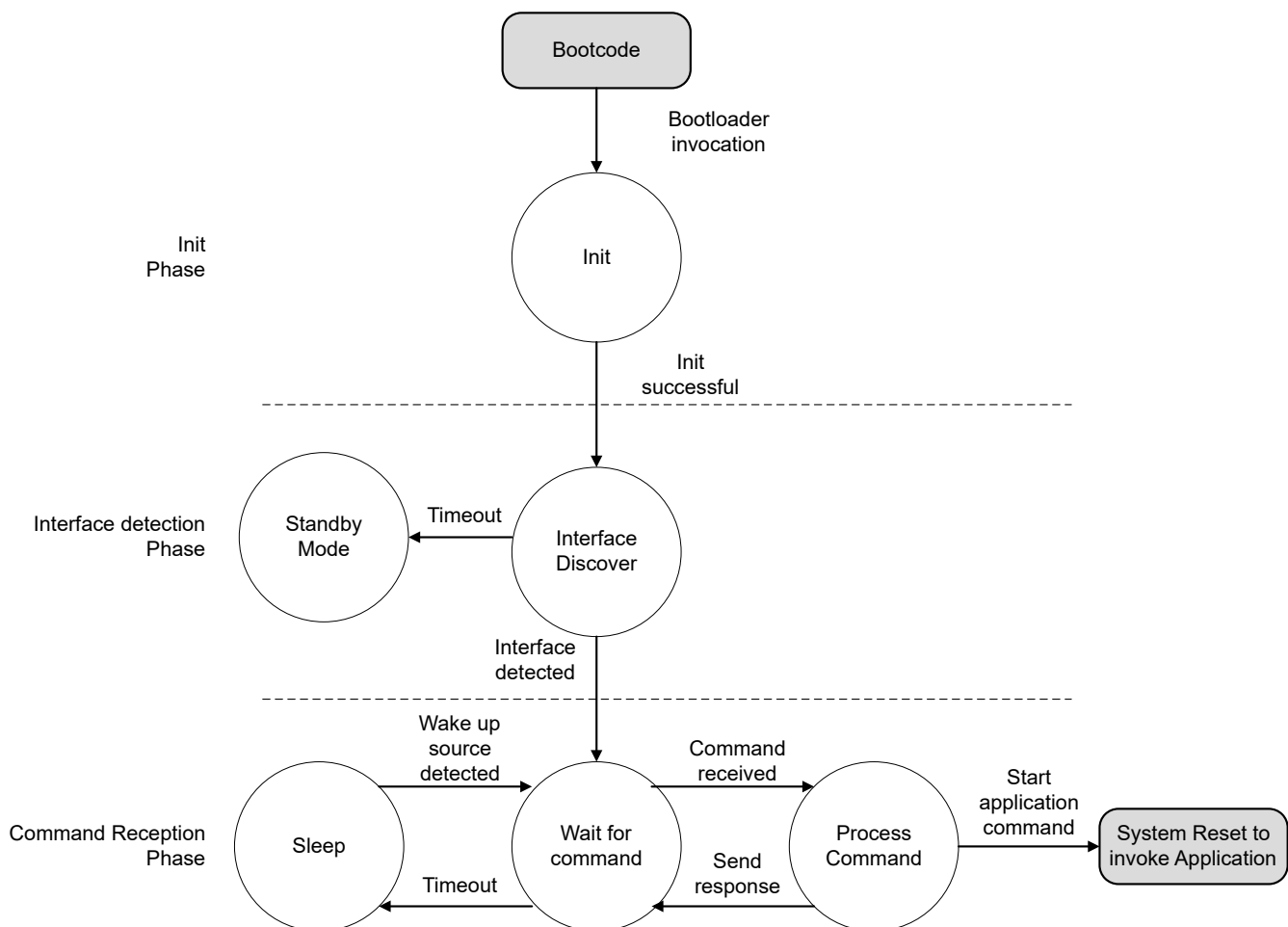


Figure 2-1. BSL Architecture

2.1.1 Timeout Feature

Bootloader will time out when no activity is detected and enters low-power mode to save power.

This has been implemented at the following two phases.

1. Interface autodetection
2. Command reception

2.1.1.1 Interface Autodetection

In Interface detection phase, if there is no valid connection command received for 4 seconds over any interface, the bootloader enters STANDBY mode.

POR is required to get out of this state and use Bootloader by creating the BSL invocation condition again.

2.1.1.2 Command Reception

In command reception phase, if no valid command is received for 4 seconds, the bootloader will enter SLEEP mode. To wake up the device from SLEEP mode, a data transaction should happen on the active interface.

Bootloader will be locked before entering Sleep mode to reduce attack surface. Hence after waking up from the low power mode, Bootloader needs to be unlocked again by sending the 256 bit BSL Password (See [Unlock Bootloader command](#)).

2.2 BSL Invocation

Bootloaders shall be invoked only by the Bootcode, when any of the BSL invocation conditions are met and if Bootloader is enabled in BCR configuration.

When Fast boot mode is enabled in BCR configuration, bootloader can be invoked only by Debug mailbox command and Application request. Other checks are skipped to save execution time.

In the **HS-FS (High Secure – Field Securable)** state, GPIO-based invocation, Application Request, and Debug Mailbox are enabled. In the **HS-KP (High Secure – Key Provisioning)** state, only **GPIO-based invocation** is enabled; Application Request and Debug Mailbox invocation are disabled to maintain security during key provisioning. Once the device transitions to **HS-SE (High Secure – Security Enforced)**, the bootloader is completely disabled, and no BSL invocation is possible, ensuring maximum protection of customer code.

2.2.1 Application Request

To invoke the bootloader from an application, set the RESETLEVEL as BOOTLOADERENTRY and trigger reset through the RESETCMD register. This sequence causes a system reset, and the Bootcode is executed and the bootloader is invoked.

Because a system reset is issued, all the peripheral configurations are reset while exiting the application.

2.2.2 GPIO Based Invocation

GPIO used for BSL invocation can be configured in the BSL Configuration in Non-main flash.

Fresh devices will have the TI programmed default pin detail in BSL configuration.

GPIO pin based invocation can be disabled in the BCR configuration. It is enabled by default.

GPIO should be asserted before POR, and the state should be maintained for at-least T_start ms after POR. Then the GPIO pin state can be de-asserted.

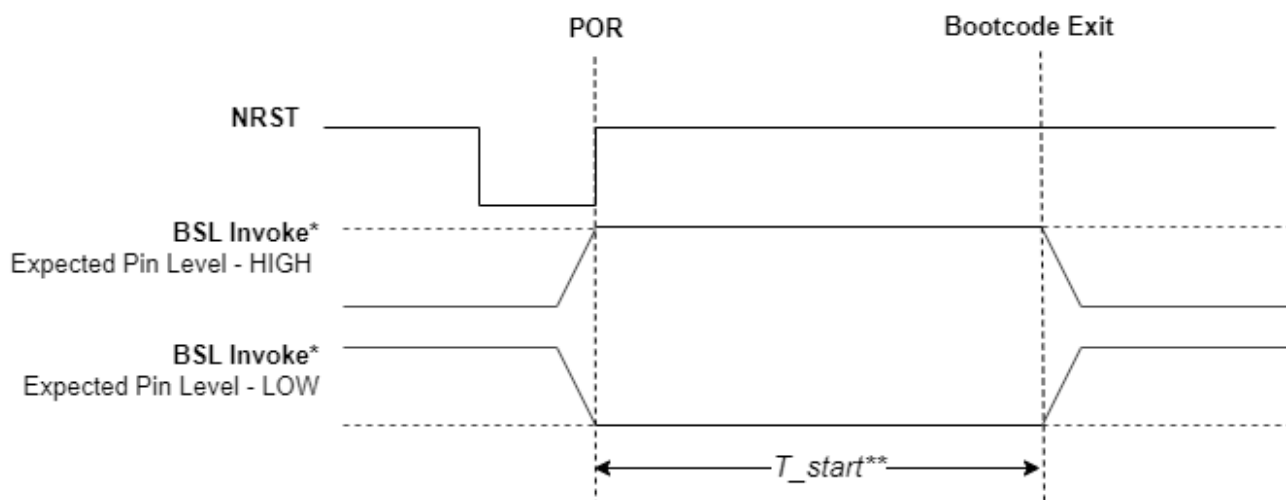


Figure 2-2. Invocation From GPIO

* - GPIO pin to be used as 'BSL Invoke' and 'Expected Pin Level' can be configured in BSL configuration

** - T_{start} refers to the Cold boot startup time, specified in the device specific data sheet

Note

If pin based BSL invocation is enabled, the configured GPIO pin should be either pulled high or low. It should not be left floating, which could cause unexpected BSL execution.

2.2.3 Debug Mailbox Command

When debug interface is available, Bootloader invocation command can be sent through the Debug Subsystem Mailbox (DSSM).

For more details on the DSSM command usage, please refer to [MSPM33C_Technical_Reference_Manual](#) DSSM commands section.

2.2.4 Other BSL Invocation Methods

2.2.4.1 Pre-Boot Application Verification

In the BCR configuration, if application CRC verification is enabled, the Bootcode verifies the CRC for the given application memory range. If the CRC is incorrect, the bootloader is invoked. In this case the bootloader is invoked only if enabled. Otherwise, the Bootcode logs a catastrophic error and causes boot failure.

2.2.4.2 Blank Device Handling

Unlike previous devices, when a blank device is detected (empty flash memory), the Bootcode does not automatically invoke the bootloader. Instead, it copies a while (1) loop to SRAM and waits for a debugger to be connected within 10 seconds. If no debugger connection is established within this timeframe, the device enters low power mode. This approach provides a more flexible development experience when working with blank devices.

2.3 Memory

2.3.1 SRAM Memory Usage

The MSPM33 BSL dynamically allocates SRAM memory based on the available SRAM size in the device. The SRAM memory layout explains the memory used for bootloader's operation.

The SRAM Memory layout explains the memory used for bootloader's operation.

- Data and Stack section - Used by BSL for its operation. While exiting the bootloader, these sections of the SRAM are cleared.
- Variable Buffer Space - Buffer space used for storing the data packets received /sent during the BSL communication

The SRAM memory allocation follows this structure:

1. BSL Buffer Start Address: Calculated as the end address of the SRAM data section used for BootROM operation, aligned to the next 8-byte boundary.
2. BSL Buffer End Address: Calculated as the RAM end address minus the stack size.
3. Available Buffer Space: The space between BSL Buffer Start Address and BSL Buffer End Address is divided into two equal parts:
 - a. RX Buffer: Used for receiving data packets, starts at BSL Buffer Start Address
 - b. TX Buffer: Used for transmitting data packets, starts at BSL Buffer Start Address + Buffer Size

The maximum buffer size is limited to 32KB (0x7FFF bytes) due to the BSL protocol having 2 bytes for defining length. Even though the MSPM33 has 256KB of SRAM, only 32KB will be used for each buffer (RX and TX).

The SRAM memory allowed for read and write access by the host is BSL Buffer Start Address to [SRAM end address – Stack Size], where SRAM end address is determined by the SRAM memory available in each device. Since the same SRAM space is shared with variable buffer space, it has chances of getting overwritten during an SRAM write/read operation.

2.4 BSL Non-Main Configuration

BSL configuration in Non-main Flash memory allows certain parameters used by BSL to be customized by the user. The configuration is stored in Non-main Flash memory starting at offset 0x1C00.

2.4.1 BSL Configuration ID

The BSL_CONFIG_ID is a predetermined signature ID that identifies this as a Bootloader configuration structure. This 32-bit field serves as an identifier for the BSL configuration.

Usage: This field identifies the structure as a BSL configuration structure. It is a fixed value that should not be modified by users.

Returned in Get Device Info: The BSL_CONFIG_ID is included in the response to the Get Device Info command (CMD_GET_IDENTITY, 0x19). This allows the host to identify the BSL configuration version or type.

2.4.2 BSL Interface Pins (BLINTERFACE_PINS)

The BLINTERFACE_PINS structure configures the physical pins and pin function multiplexer selections for the BSL communication interfaces (UART and I2C). This configuration allows the BSL to use specific pins for communication with the host.

UART Interface Pins (0x80101C04)

1. UART Receive Pin Number (Bits 7...0)
 - a. Physical pin number for receiving data through UART
2. UART Receive Pin Function (Bits 15...8)
 - a. Function multiplexer selection for the receive pin
3. UART Transmit Pin Number (Bits 23...16)
 - a. Physical pin number for transmitting data through UART
4. UART Transmit Pin Function (Bits 31...24)
 - a. Function multiplexer selection for the transmit pin

I2C Interface Pins (0x80101C08)

1. I2C Data Pin Number (Bits 7...0)

- a. Physical pin number for I2C data line (SDA)
2. I2C Data Pin Function (Bits 15...8)
 - a. Function multiplexer selection for the data pin
3. I2C Clock Pin Number (Bits 23...16)
 - a. Physical pin number for I2C clock line (SCL)
4. I2C Clock Pin Function (Bits 31...24)
 - a. Function multiplexer selection for the clock pin

Reserved Pin Information (0x80101C0C)

2.4.3 BSL Invoke Pin Configuration (BSLPIN_INVOKE)

The BSLPIN_INVOKE structure configures the GPIO pin used to trigger BSL invocation. This allows the BSL to be invoked by setting a specific pin to a specific state during device startup.

1. Pin Configuration and Trigger Level (Bits 7...0)
 - a. Defines the pin configuration and expected trigger level for BSL invocation
 - b. Bit Fields
 - i. PINCM Index (Bits 6...0): Specifies the pin configuration mode index
 - ii. Trigger Level (Bit 7): Specifies the pin state that triggers BSL invocation
 1. 0 = BSL is invoked when pin is LOW
 2. 1 = BSL is invoked when pin is HIGH
 - iii. Default Value: 0xA8
 1. PINCM Index = 0x28 (40)
 2. Trigger Level = 1 (HIGH)
2. Pin Selection (Bits 15...8)
 - a. Defines which physical pin is used for BSL invocation
 - b. Bit Fields
 - i. Pin Number (Bits 4...0): Specifies which pin number within the selected port
 - ii. Port Index (Bits 6...5): Specifies which GPIO port the pin belongs to
 - iii. Reserved (Bit 7): Not used in the current implementation
 - c. Default Value: 0x12
 - i. Pin Number = 0x12 (18)
 - ii. Port Index = 0 (Port A)

The BSL invoke pin configuration allows the device to enter BSL mode when a specific GPIO pin is set to a specific state during device startup. By default, the BSL is invoked when Pin 18 of Port A is set to HIGH during device startup.

To invoke the BSL using the configured pin:

1. Set the specified pin (default: Pin 18 of Port A) to the specified state (default: HIGH)
2. Power on the device or perform a reset
3. Maintain the pin state for at least T_start milliseconds after power-on reset
4. The pin state can be de-asserted after T_start milliseconds

Note

If pin-based BSL invocation is enabled, the configured GPIO pin should be either pulled high or low according to the configuration. It should not be left floating, which could cause unexpected BSL execution.

2.4.4 Memory Readout Configuration

The READOUT field controls whether the BSL allows memory readout operations. When enabled, the Memory Read Back command can be used to read the contents of flash and SRAM memory. When disabled, the Memory Read Back command will return an error.

Default Value: BL_CFG_READBACK_EN (0xAABB)

Valid Values:

1. BL_CFG_READBACK_EN (0xAABB): Enable memory readout
2. All other values: Memory readout disabled

By default, memory readout is disabled for security reasons. It should only be enabled in developing environments or when memory readout is specifically required.

Even when memory readout is enabled, the BSL must still be unlocked with the correct password before the Memory Read Back command can be used. This provides an additional layer of security to prevent unauthorized access to the memory contents.

2.4.5 BSL Password

Memory Address: 0x80101C14 - 0x80101C30

The PASSWORD field stores the SHA-256 hash (32 bytes) of the password used for BSL authentication. This password is required to unlock the bootloader and access protected BSL commands.

Default Value: The hash of a string of all 0xFF bytes

1. Word 0 (0x80101C14): 0x761396AF
2. Word 1 (0x80101C18): 0x5F63720F
3. Word 2 (0x80101C1C): 0x5A4AB4BD
4. Word 3 (0x80101C20): 0x9FC3630A
5. Word 4 (0x80101C24): 0xF930AF12
6. Word 5 (0x80101C28): 0x5CEEA650
7. Word 6 (0x80101C2C): 0x88E11B97
8. Word 7 (0x80101C30): 0x51409CE8

Usage

The BSL password is used to protect access to critical BSL commands that can modify the device memory or read sensitive information. When the BSL is invoked, it starts in a locked state. To unlock the BSL and access protected commands, the host must send the Unlock Bootloader command (CMD_UNLOCK_BSL, 0x21) with the correct password.

The password authentication process works as follows:

1. The host sends the Unlock Bootloader command with a 32-byte password.
2. The BSL calculates the SHA-256 hash of the received password.
3. The BSL compares the calculated hash with the stored password hash in the BSL configuration.
4. If the hashes match, the BSL is unlocked and protected commands become available.
5. If the hashes do not match, the BSL remains locked and returns a BSL_PASSWORD_ERROR.

The BSL includes several security features to protect against password attacks:

1. **Sleep Delay:** If an incorrect password is sent, the device enters sleep mode for 2 seconds and does not accept any commands during this period. This makes brute force attacks more time-consuming.
2. **Security Alert:** If an incorrect password is sent 3 times, a security alert action is taken based on the SECURITY_ALERT_LEVEL configuration. This can include:
 - a. Factory Reset: Erases all flash memory
 - b. Disable Bootloader: Permanently disables the bootloader
 - c. Do Nothing: No action is taken

3. **Password Backup:** Before performing a factory reset, the BSL backs up the password to SRAM. This allows the BSL to be unlocked after a factory reset, even if the password field in non-main flash is erased.

Changing the BSL Password

There are multiple methods to change the BSL password

1. **Using BSL Commands**
 - a. Unlock the BSL with the current password.
 - b. Calculate the SHA-256 hash of the new password.
 - c. Update the PASSWORD field in the BSL configuration with the new hash.
 - d. Calculate the new CRC for the BSL configuration.
 - e. Program the updated BSL configuration to non-main flash using the Program Data command.
2. **Using SysConfig and Code Composer Studio (CCS)** The SysConfig tool provides a Non-Main Configurator that can generate a complete non-main configuration. If the non-main flash area is not protected, you can use the Flash Loader in Code Composer Studio to directly modify the BSL password.
 - a. Open the SysConfig tool in CCS or as a standalone application.
 - b. Navigate to the Non-Main Configurator section.
 - c. Configure all the BSL settings, including the PASSWORD field with the hash of the new password.
 - d. Generate the non-main configuration binary.
 - e. Program the generated binary to the non-main flash area using CCS Flash Loader or BSL commands.

Note

If you forget the BSL password, you may need to perform a DSSM factory reset to restore the default password. However, this will erase all flash memory, including the application code and configuration data. Alternatively, if the non-main flash area is not protected, you can use CCS with Flash Loader to reprogram the BSL configuration with a new password.

2.4.6 Application Revision Pointer

Memory Address: 0x80101C34

The APP_REV_POINTER field contains a pointer to the application version information stored in main flash memory. This allows the BSL to retrieve and report the application version when requested by the host.

Default Value: 0xFFFFFFFF (indicating no application version information is available)

Usage:

The APP_REV_POINTER is used by the BSL to locate and retrieve the application version information when the Get Device Info command (CMD_GET_IDENTITY, 0x19) is executed. This allows the host to query the version of the application currently installed on the device without having to load and execute the application.

When the Get Device Info command is executed, the BSL performs the following checks:

1. Verifies that the APP_REV_POINTER contains a valid flash address
2. Checks if the address is not blank (not 0xFFFFFFFF)
3. Ensures the address is 64-bit aligned
4. If all checks pass, reads an 8-byte (64-bit) value from the specified address as the application version

If any of these checks fail, the BSL returns 0 as the application version.

Implementation Requirements:

The APP_REV_POINTER field is maintained by the customer and requires the following implementation steps:

1. **Allocate a Section in Application Linker File:** The customer must allocate a dedicated section in the application linker file to store the application version information. This section should be 64-bit aligned and located in main flash memory.

2. **Define the Application Version:** The application version should be defined as a 64-bit (8-byte) value in the application code. This can be a structured value containing major, minor, and patch version numbers, or any other format that fits within 64 bits.
3. **Place the Version in the Allocated Section:** The application version value should be placed in the allocated section during the application build process.
4. **Update the APP_REV_POINTER:** The address of the allocated section must be stored in the APP_REV_POINTER field in the BSL configuration. This can be done using the methods described in the "Changing BSL Configuration" section.

Note

The APP_REV_POINTER is an optional feature. If not used, it should be left at its default value of 0xFFFFFFFF, which indicates that no application version information is available.

2.4.7 Security Alert Level

Memory Address: 0x80101C38

Bits: 15...0

Description: The SECURITY_ALERT_LEVEL field defines the action to take when a security violation is detected, such as multiple failed password attempts.

Default Value: 0xAABB (Factory Reset)

Valid Values:

1. BL_CFG_SECURITY_FACTORY_RESET (0xAABB): Perform a factory reset when a security violation occurs
2. BL_CFG_SECURITY_DISABLE_BSL (0xCCDD): Disable the bootloader when a security violation occurs
3. All other values: Take no action when a security violation occurs

Note

The SECURITY_ALERT_LEVEL is checked only after 3 consecutive incorrect password attempts. For the first and second incorrect attempts, the device enters sleep mode for 2 seconds regardless of this configuration.

2.4.8 UART Baud Rate

Memory Address: 0x80101C38

Bits: 31..16

The UART_BAUD_RATE field specifies the default baud rate used for UART communication in the BSL. This determines the communication speed when using the UART interface for BSL operations.

Valid Values:

1. BL_CFG_UART_BAUDRATE_4800 (0x1): 4800 bits per second
2. BL_CFG_UART_BAUDRATE_9600 (0x2): 9600 bits per second
3. BL_CFG_UART_BAUDRATE_19200 (0x3): 19200 bits per second
4. BL_CFG_UART_BAUDRATE_38400 (0x4): 38400 bits per second
5. BL_CFG_UART_BAUDRATE_57600 (0x5): 57600 bits per second
6. BL_CFG_UART_BAUDRATE_115200 (0x6): 115200 bits per second
7. BL_CFG_UART_BAUDRATE_1000000 (0x7): 1,000,000 bits per second
8. BL_CFG_UART_BAUDRATE_2000000 (0x8): 2,000,000 bits per second
9. BL_CFG_UART_BAUDRATE_3000000 (0x9): 3,000,000 bits per second

Changing the Baud Rate:

There are two ways to change the UART baud rate:

1. Permanent Change: Modify the UART_BAUD_RATE field in the BSL configuration using the methods described in the "Changing BSL Configuration" section. This changes the default baud rate used whenever the BSL is invoked.
2. Runtime Change: Use the Change Baud Rate command (CMD_CHANGE_BAUD_RATE) during a BSL session to temporarily change the baud rate. This change is effective only for the current BSL session and reverts to the default value when the device is reset.

Note

When using the UART interface for BSL communication, both the host and the device must be configured to use the same baud rate. If the baud rate is changed using the Change Baud Rate command, the host must also change its baud rate to match.

2.4.9 I2C Slave Address

Memory Address: 0x80101C3C

The I2C_SLAVE_ADDR field specifies the slave address used by the BSL when operating in I2C mode. External devices must use this address to communicate with the BSL over the I2C interface.

Bits: 15...0

Default Value: 0x48

The I2C_SLAVE_ADDR field sets the slave address that the BSL uses when communicating over the I2C interface. When the BSL is invoked and the I2C interface is selected, it configures the I2C peripheral to respond to this slave address.

2.4.10 Configuration CRC

Memory Address: 0x80101C4C

The CRC field contains a 32-bit Cyclic Redundancy Check value calculated over the entire BSL configuration structure (excluding the CRC field itself). This value is used to verify the integrity of the configuration data.

Default Value: 0xB4808AA4 (calculated based on default configuration values)

Valid Values: Calculated value based on the contents of the BSL configuration

The CRC field serves several important purposes:

1. Data Integrity Verification: Ensures that the BSL configuration data has not been corrupted or accidentally modified.
2. Validation of Configuration: Confirms that the BSL configuration structure is valid and complete.
3. Protection Against Errors: Helps prevent the BSL from using incorrect configuration values that could lead to unexpected behavior.

The CRC must be recalculated whenever any field in the BSL configuration is modified. This includes:

1. Initial Configuration: When the BSL configuration is first created or programmed.
2. Configuration Updates: When any field in the configuration is changed.
3. Factory Reset: After a factory reset, a new CRC is calculated for the default configuration.

An incorrect CRC in the BSL configuration will result in permanent loss as it results in catastrophic error will not be recoverable.

Note

When using tools like SysConfig or CCS to modify the BSL configuration, the CRC is typically calculated and updated automatically. However, when manually modifying the configuration using BSL commands or other methods, you must ensure that the CRC is correctly recalculated to avoid permanently damaging the device.

2.5 Changing BSL Configuration

The BSL configuration stored in Non-main Flash memory can be modified using several methods. This section describes the available approaches for updating BSL configuration parameters.

2.5.1 Reference

Prerequisites:

Factory Reset must be enabled in BCR configuration (either "Enabled" or "Enabled with Password")

Procedure:

1. Invoke the bootloader using one of the supported invocation methods (GPIO, Application request, Debug Mailbox, or SFI)
2. Establish connection using the Connection command (CMD_CONNECTION, 0x12)
3. Unlock the BSL using the Unlock Bootloader command (CMD_UNLOCK_BSL, 0x21) with the current password
4. Execute Factory Reset command (CMD_FACTORY_RESET, 0x30) to erase the non-main flash configuration
 - a. This erases both Main flash (application memory) and non-main flash (configuration memory)
 - b. Provide the factory with a reset password if BCR configuration requires it (default: all 0xFF)
 - c. After factory reset, the BSL password reverts to default (hash of all 0xFF)
5. Prepare the new BSL configuration structure and Calculate CRC32 for the BSL configuration:
 - a. Calculate CRC32-ISO3309 over the entire configuration structure (excluding the CRC field itself)
 - b. Use bit-reversed configuration with initial seed 0xFFFFFFFF
 - c. Configuration structure size: 76 bytes (from 0x80101C00 to 0x80101C4B)
 - d. Store the calculated CRC32 value in the CRC field (0x80101C4C)
6. Program the new BSL configuration to non-main flash using the Program Data command (CMD_PROGRAM_DATA, 0x20):
 - a. Start Address: 0x80101C00
 - b. Data: Complete BSL configuration structure (80 bytes total, including CRC)
 - c. Ensure data is 16-byte aligned as required by the Program Data command
7. Reset the device to apply the new configuration
8. Note The CRC field must be recalculated whenever any configuration field is modified • An incorrect CRC will result in catastrophic error and permanent device lockout • If Factory Reset is not enabled in BCR configuration, non-main flash programming will fail • Ensure the entire configuration structure is programmed, not just individual fields

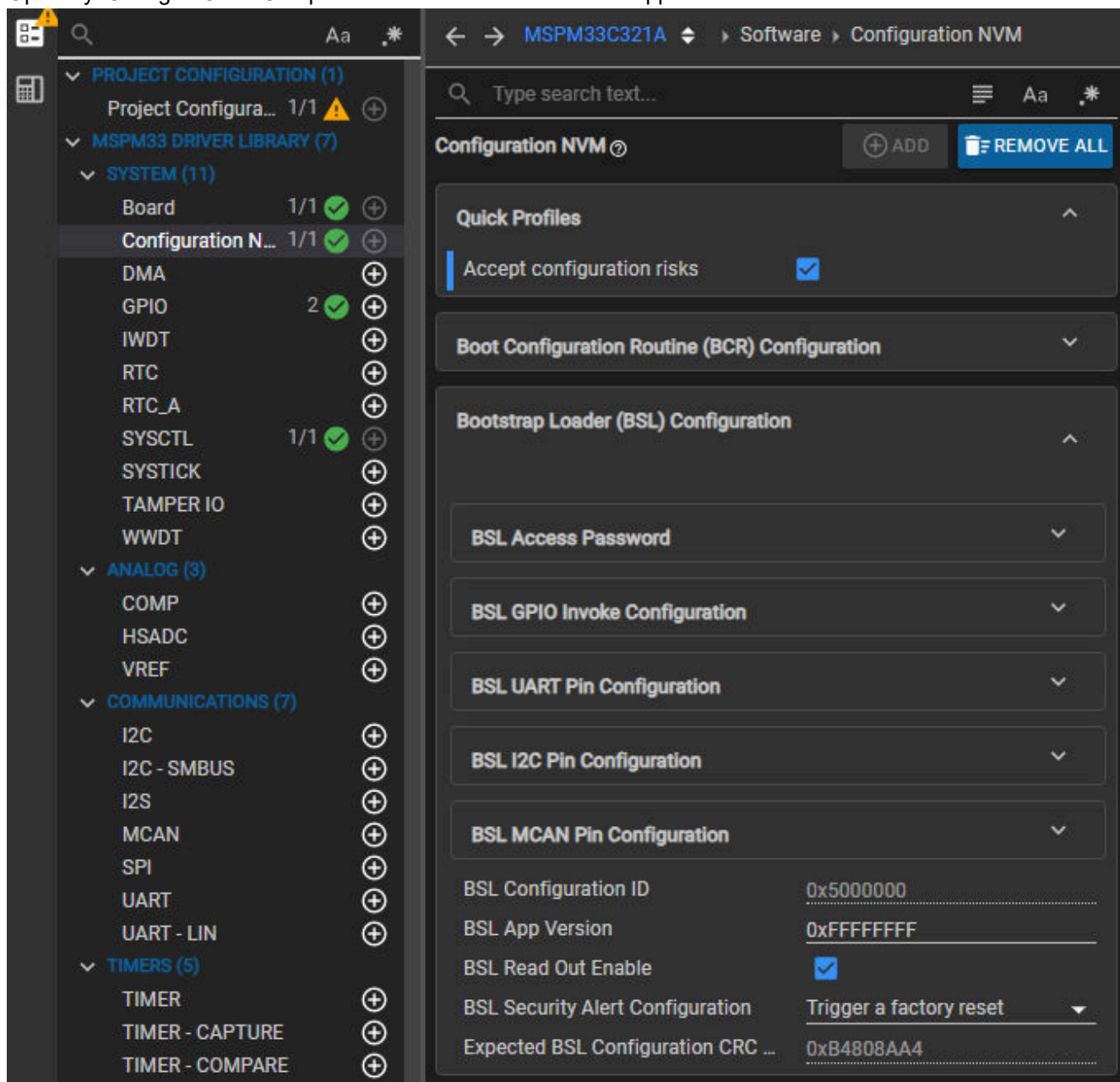
Note

- The CRC field must be recalculated whenever any configuration field is modified
 - An incorrect CRC will result in catastrophic error and permanent device lockout
 - If Factory Reset is not enabled in BCR configuration, non-main flash programming will fail
 - Ensure the entire configuration structure is programmed, not just individual fields
-

2.5.2 Using SysConfig and Code Composer Studio (CCS)

The SysConfig tool provides a graphical interface for configuring BSL parameters:

1. Open SysConfig in Code Composer Studio or as a standalone application



2. Navigate to the Non-Main Configurator section
3. Configure the desired BSL parameters:
 - a. Interface pins (UART, I2C)
 - b. BSL invoke pin configuration
 - c. Memory readout enable/disable
 - d. BSL password: Enter the pre-calculated 32-byte SHA-256 hash (not the plaintext password)
 - e. Application revision pointer
 - f. UART baud rate
 - g. I2C slave address
4. The tool automatically calculates and updates the CRC field based on all configured parameters
5. Generate the non-main configuration binary file
6. Program the generated binary to non-main flash using one of these methods:

- a. CCS Flash Loader (if non-main is not write-protected and debug access is available)
- b. BSL Program Data command (if Factory Reset is enabled in BCR configuration)
- c. Debug interface programming tools

Note

- SysConfig does NOT calculate the SHA-256 hash of passwords - you must provide the pre-calculated hash
 - The tool only calculates the CRC32 for the configuration structure integrity
 - Keep a secure record of both your plaintext password and its SHA-256 hash
-

2.5.3 Using Debug Interface

If the non-main flash area is not write-protected, the BSL configuration can be directly programmed using a debug probe:

1. Connect a debug probe (XDS110, etc.) to the device
2. Use Code Composer Studio Flash Loader or similar tool
3. Load the BSL configuration binary file
4. Program to non-main flash starting at address 0x80101C00

3 Bootloader Protocol

3.1 Packet Format

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 to 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 Data	PI Code
---------	---------------	---------

3.2 BSL Protocol

The data packets of the UART and I2C BSL protocol have the following structure.

- Header byte indicates the protocol used and the packet type (command or response packet).
- Length field contains the size of the BSL Core Data in bytes.
- BSL core data, contains the Command / Response ID and Address, data as needed by the command
- CRC32 field contains the CRC calculated for the data in BSL core data

PI Code		BSL Core Data	PI Code
Header (1 byte)	Length (2 byte)	BSL Core Command/Response	CRC32 (4 byte)

Based on the Core data field, the data packet is classified as either Command packet or response packet.

Command packet is the first packet which is transmitted to the BSL. The second packet is the Response packet which is received from the BSL. Response packet contains two components BSL acknowledgment and BSL Core response. In these two, Acknowledgment is received from BSL for every command packet sent. But the BSL core response is not received for every command.



Figure 3-1. BSL Protocol

3.2.1 BSL Acknowledgment

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](#) lists the 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 must check the acknowledgment error and retry transmission.

Data	Meaning
0x00	BSL_ACK (Packet received successfully)
0x51	BSL_ERROR_HEADER_INCORRECT
0x52	BSL_ERROR_CHECKSUM_INCORRECT
0x53	BSL_ERROR_PACKET_SIZE_ZERO
0x54	BSL_ERROR_PACKET_SIZE_TOO_BIG
0x55	BSL_ERROR_UNKNOWN_ERROR

Data	Meaning
0x56	BSL_ERROR_UNKNOWN_BAUD_RATE

3.2.2 Peripheral Configuration

3.2.2.1 UART

UART is enabled with following configuration:

- UC12 is used
- Baud rate 9600 bps by default. It can be updated by Change Baud Rate command
- Data width: 8 bit
- Stop bits: 1
- No parity
- Pins used for RXD and TXD are taken from the BSL configuration

3.2.2.2 I2C

The I2C interface in the BSL can act as the I2C target. The host acts as a controller and drives the communication.

- UC15_0 is used
- The I2C target address is 0x48, by default. It can be configured in BSL Configuration
- External pullup is needed for SCL and SDA lines
- Pins used for SDA and SCL are taken from the BSL configuration

3.2.2.3 CRC

CRC for the data must be calculated with:

- CRC32-ISO3309 polynomial
- Bit reversed configuration
- Initial seed - 0xFFFFFFFF

3.3 Bootloader Core Commands

3.3.1 Connection

Structure

Header	Length		CMD	CRC32			
0x80	0x01	0x00	0x12	C1	C2	C3	C4

Description

Connection command is the first command used to establish the connection between the Host and the Target through a specific interface (UART or I2C).

Protected

No

Command Returns

Only [BSL Acknowledgment](#)

Example

Host: 80 01 00 12 3A 61 44 DE

BSL: 00

3.3.2 Get Device Info

Structure

Header	Length		CMD	CRC32			
0x80	0x01	0x00	0x19	C1	C2	C3	C4

Description

The command is used to get the version information and buffer size available for data transaction.

Protected

No

Command Returns

BSL Acknowledgment and BSL core response with device information. See [Device info](#) for more details.

Example

Host: 80 01 00 19 B2 B8 96 49

BSL:00 08 19 00 31 00 01 00 01 00 00 00 00 01 00 C0 06 60 01 00 20 01 00 00 00 01 00 00 00 49 61 57 8C

3.3.3 Unlock Bootloader

Structure

Header	Length		CMD	Data	CRC32			
0x80	0x21	0x00	0x21	D1...D32	C1	C2	C3	C4

Description

The command is used to unlock the bootloader. Only after bootloader unlock, all the protected commands listed in [Section 3.3](#) are processed by the BSL.

If the host sends an incorrect password, the BSL sends the error response, then device enters sleep mode for 2 seconds. During this 2-second window, no command is received or processed. After 2 seconds, the host can send the next password. If an incorrect password is sent 3 times, a security alert action is taken. The security alert action is configurable in the BSL configuration. See [Section 3.5.1.1](#) for more details.

Protected

No

Data

32 byte BSL password stored in the BSL configuration memory. Fresh device would have the default value all 0xFF.

Command Returns

BSL Acknowledgment and BSL core response with Message about the Status of operation. See section [BSL Core Message](#) for more details.

Example

[illegible]

BSL: 00 08 02 00 3B 00 38 02 94 82

3.3.4 Program Data

Structure

Header	Length		CMD	Address	Data	CRC32			
0x80	L1	L2	0x20	A1...A4	D1...Dn	C1	C2	C3	C4

Description

This command is used to program data to a specific memory address. The data can be programmed to main flash memory, info flash memory (non-main) or data bank flash memory.

Important Requirements:

- The start address must be 16-byte aligned (address % 16 == 0)
- The data length must be 16-byte aligned (length % 16 == 0)
- For programming to info flash memory (non-main), Factory Reset must be enabled in the BCR configuration

Protected

Yes

Address

Start address of the memory region to be programmed. A1...A4, where A1 is the Least Significant Byte of the 32 bit address.

Data

Data bytes to be written in the specified address. Maximum size of the data that can be sent is limited by the Buffer size of the device. Buffer size is known from Get Device Info [command](#).

Command Returns

BSL Acknowledgment and BSL core response with Message about the Status of the operation. See section [Section 3.4.1](#) for more details.

Example

Host: 80 0D 00 20 00 00 00 00 00 00 00 04 00 00 00 08 7A DC AE B8

BSL: 00 08 02 00 3B 00 38 02 94 82

3.3.5 Program Data Fast

Structure

Header	Length		CMD	Address	Data	CRC32			
0x80	L1	L2	0x24	A1...A4	D1...Dn	C1	C2	C3	C4

Description

The Program Data Fast command is identical to the Program Data command, except that this command performs the non-blocking write, to speed up the programming process. To this command BSL will not send the BSL core message response to indicate if the programming was successful.

Protected

Yes

Address

Start address of the memory region to be programmed. A1...A4, where A1 is the least significant byte of the 32 bit address.

Data

Data bytes to be written in the specified address. Maximum size of the data that can be sent is limited by the Buffer size of the device. Buffer size is known from Get Device Info [command](#).

Command Returns

BSL Acknowledgment.

Example

Host: 80 0D 00 24 00 01 00 00 01 02 03 04 05 06 07 08 72 10 2A 18

BSL: 00

3.3.6 Readback Data

Structure

Header	Length		CMD	Address	Data	CRC32			
0x80	0x09	0x00	0x29	A1...A4	L1...L4	C1	C2	C3	C4

Description

This command is used to readout the data starting from address A1...A4.

Readout should be enabled in the BSL configuration, to readout the data using this command. It is disabled by default in the BSL configuration.

Data readout is allowed for Main flash (application memory), Non-main Flash (configuration memory) and SRAM memory.

Note

SRAM memory is not fully accessible by the Host. See [SRAM memory usage](#) for more details.

Protected

Yes

Address

Start address of the memory region to be read back. A1...A4, where A1 is the least significant byte of the 32 bit address.

Data

Size of the data to be read in bytes, L1...L4, where L1 is the least significant byte. Maximum size of the data that can be read is limited by the Buffer size of the device. Buffer size is known from [Get Device Info command](#).

Command Returns

BSL Acknowledgment and BSL core response with requested data, if the readback command is valid. See [Section 3.4.3](#) for more details.

If the readback command, had invalid address / length or if the readout was disabled, the corresponding error will be sent as the Message Response following the BSL acknowledgment.

Example

Host: 80 09 00 29 00 0C 00 00 08 00 00 00 32 9D B0 35

BSL: 00 08 09 00 30 FF FF FF FF FF FF FF F6 2B A1 73

3.3.7 Flash Range Erase

Structure

Header	Length		CMD	Address	Data	CRC32			
0x80	0x09	0x00	0x23	A1...A4(Start Address)	A1...A4(End Address)	C1	C2	C3	C4

Description

The Flash range erase command is used to erase the specified flash memory region. Flash is erased sector wise (2 KB), erasing lesser size is not possible.

When the Start and End address reside in different flash sectors, the BSL erases all the flash sectors in between the Start and End address, including sectors that contains these addresses.

This command can be used to erase only the Main Flash memory. Non-main erase is not possible.

End address should never be less than Start address.

Protected

Yes

Address

Start address of the memory region to be erased. A1...A4, where A1 is the Least Significant Byte of the 32 bit address.

Data

End address of the memory region to be erased. A1...A4, where A1 is the Least Significant Byte of the 32 bit address.

Command Returns

BSL Acknowledgment and BSL core response with Message about the Status of the operation. See [Section 3.4.1](#) for more details.

Example

Host: 80 09 00 23 00 01 00 00 FF 03 00 00 2B E6 BE D8

BSL: 00 08 02 00 3B 00 38 02 94 82

3.3.8 Mass Erase

Structure

Header	Length		CMD	CRC32			
0x80	0x01	0x00	0x15	C1	C2	C3	C4

Description

The mass erase command erases the complete Main Flash memory (application memory) available in the device.

The Mass erase configuration in BCR configuration memory doesn't affect this BSL command.

When a flash region is static write protected in BCR configuration memory, the region can't be erased.

Protected

Yes

Command Returns

BSL Acknowledgment and BSL core response with Message about the Status of the operation. See [Section 3.4.1](#) for more details.

Example

Host: 80 01 00 15 99 F4 20 40

BSL: 00 08 02 00 3B 00 38 02 94 82

3.3.9 Factory Reset

Structure

Header	Length		CMD	Data	CRC32			
0x80	L1	L2	0x30	D1...D16	C1	C2	C3	C4

Description

The factory reset command erases the complete Main Flash (application) memory and Non-main flash (configuration) memory.

Processing this command, is impacted by the factory reset configuration in the BCR configuration memory.

Factory reset is

- Allowed without password, if 'Enabled'
- Allowed with password, if 'Enabled with Password'
- Not allowed, if 'Disabled'

When a flash region is static write protected in BCR configuration memory, the region can't be erased .

Protected

Yes

Data

16 byte factory reset password stored in the BCR configuration memory. The default password is all 0xFF. Password is required only when Factory reset is "Enabled with Password" in BCR configuration.

Command Returns

BSL Acknowledgment and BSL core response with Message about the Status of the operation. See [Section 3.4.1](#) for more details.

CAUTION

After doing the factory reset, until the Non-main configuration has been restored, the system is highly vulnerable to a potential lockout situation in which it is impossible to get access to the device again.

Example

Scenario 1: Factory Reset without Password

Host: 80 01 00 30 DE 20 24 0B

BSL: 00 08 02 00 3B 00 38 02 94 82

Scenario 2: Factory Reset with Password

Host : 80 11 00 30 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 8A 28 EA DC

BSL : 00 08 02 00 3b 00 38 02 94 82

3.3.10 Standalone Verification

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x05	0x00	0x32	D1...D4	C1	C2	C3	C4

Description

This command is used to verify the CRC of the data stored at the given memory range. This enables faster verification of the programmed data. It requires the data size to be minimum 1kB.

CRC verification is allowed for Main flash (application memory) and Non-main Flash (configuration memory)

Protected

Yes

Data

Size of the data to be verified in bytes, D1...D4, where D1 is the least significant byte. 1kB <= size <= 64 KB.

Command Returns

BSL Acknowledgment and BSL core response with CRC value calculated for the requested memory region. Refer [Section 3.4.5](#) for more details.

If the verification command, had invalid address / length, the corresponding error will be sent as the Message Response following the BSL acknowledgment. Refer [Section 3.4.1](#).

Example

Host: 80 09 00 26 00 00 00 20 00 04 00 00 A0 97 D5 2E

BSL: 00 08 02 00 3B 05 B7 F6 FE F2

3.3.11 Start Application

Structure

Header	Length		CMD	CRC32			
0x80	0x01	0x00	0x40	C1	C2	C3	C4

Description

Start application command, issues a system reset, which causes Bootloader exit, rerun of Bootcode, which in turn starts the application.

Protected

No

Command Returns

BSL Acknowledgment

Example

Host: 80 01 00 40 E2 51 21 5B

BSL: 00

3.3.12 Change Baud Rate

Structure

Header	Length		CMD	Data	CRC32			
0x80	0x02	0x00	0x52	D1	C1	C2	C3	C4

Description

The command can be used to change the baud rate of the UART interface. The new baud rate will take effect after sending out the BSL acknowledgment for this packet.

The default baud rate of the BSL UART is 9600 bps.

Note

After updating the baud rate, if a wrong BSL password is sent with the Unlock Bootloader command, the baud rate settings will be reverted to default value. Further communication should happen in the default baud rate.

Protected

No

Data

D1 Baud rate as specified in the table.

ID	Baud Rate (bps)
1	4800
2	9600
3	19200
4	38400
5	57600
6	115200
7	1000000
8	2000000
9	3000000

Command Returns

BSL Acknowledgment

Example

Host: 80 02 00 52 03 6C 83 A2 AF

BSL: 00

3.4 Bootloader Core Response

BSL Response	RSP	Data
Memory read back	0x30	D1...Dn
Get Device Info	0x31	D1...D24
Standalone verification	0x32	D1...D4
Message	0x3B	MSG
Detailed Error	0x3A	D1..D3

Abbreviations:

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. 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..Dn

Data bytes, where 'n' is limited by the BSL maximum buffer size

3.4.1 BSL Core Message

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x02	0x00	0x3B	MSG	C1	C2	C3	C4

Description

For some commands, BSL sends the message response to the host, which indicates the status of the processed command. The table lists all the possible messages from the BSL.

MSG	Meaning	Possible Cause ⁽¹⁾
0x00	Operation Successful	

MSG	Meaning	Possible Cause ⁽¹⁾
0x01	BSL Locked error	The BSL is not yet unlocked with Bootloader unlock password command or After BSL unlock, a timeout would have occurred in the command reception phase
0x02	BSL password error	Incorrect password has been sent to unlock bootloader.
0x03	Multiple BSL password error. Security Alert action is taken.	Incorrect password has been sent to unlock bootloader 3 times.
0x04	Unknown Command	The command given to the BSL was not recognized as valid command
0x05	Invalid memory range	The given memory range is invalid.
0x06	Invalid command	The command given to the BSL is a known command but it is invalid at that time instant and cannot be processed.
0x07	Factory reset disabled	Factory reset is disabled in the BCR configuration
0x08	Factory reset password error	Incorrect or no password has been sent with factory reset command, when the BCR configuration has factory reset 'Enabled with password'
0x09	Read out error	Memory read out be disabled in BCR configuration
0x0A	Invalid address or length alignment	Start address or data length for the flash programming is not 8-byte aligned
0x0B	Invalid length for standalone verification	Data size sent for standalone verification is less than 1KB

(1) Possible causes listed here, are not the only reason for the status or error. It only lists the probable software reasons for the error, that can be corrected by the host.

3.4.2 Detailed Error

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x02	0x00	0x3A	D1..D3	C1	C2	C3	C4

Description

D1 - Error Type

D3,D2 - Error Details

Possible Values

Error type		Error Details	
Value	Description	Value	Description
0xF0	Flash Error	0xFF	Contains FLASHCTL.STATCMD register value

3.4.3 Memory Readback

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x02	0x00	0x30	D1...Dn	C1	C2	C3	C4

Description

The command returns the requested data in response to the Readback [command](#)

Data

Data D1..Dn where 'n' is limited by the BSL maximum buffer size.

3.4.4 Device Info

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x19	0x00	0x31	D1...D24	C1	C2	C3	C4

Description

The command returns the version information and BSL buffer size in response to the Get Identity [Command](#)

Data

Identity Byte	Data Byte
Command Interpreter version	[D02-D01]
Build ID	[D04- D03]
Application version	[D08-D05]
Active Plug-in interface version	[D10-D09]
BSL Max buffer size	[D12-D11]
BSL Buffer Start address	[D16-D13]
BCR Configuration ID	[D20-D17]
BSL Configuration ID	[D24- D21]

Application version:

32 bit application version is taken from the address specified in BSL configuration

BSL Buffer size:

RAM data buffer size available to store the BSL data packets that are sent/received.

Example

Host: 80 01 00 19 B2 B8 96 49

BSL: 00 08 19 00 31 00 01 00 01 00 00 00 00 01 00 C0 06 60 01 00 20 01 00 00 00 01 00 00 00 49 61 57 8C

In the above given response,

Command Interpreter version - 0x0100

Build ID - 0x0100

Application version - 0x00000000

Active Plug-in interface version - 0x0001

BSL Max buffer size - 0x06C0

BSL Buffer Start address - 0x20000160

BCR Config ID - 0x00000001

BSL Config ID - 0x00000001

3.4.5 Standalone Verification

Structure

Header	Length		RSP	Data	CRC32			
0x08	0x05	0x00	0x32	D1...D4	C1	C2	C3	C4

Description

The command returns the CRC value in response to the standalone verification command

Data

32 bit CRC value calculated for the requested memory region. D1...D4, where D1 is the least significant byte of the CRC32.

3.5 Bootloader Security

3.5.1 Password Protected Commands

All the commands that can access the data in the memory directly or indirectly, are password protected. The password is configurable in the BSL configuration in Non-main memory.

When an incorrect password is sent, device will sleep for next 2 seconds and does not accept any command during this period to make brute force attacks harder. When incorrect password is sent for 3 times, Security alert action is taken by the BSL.

3.5.1.1 Security Alert

A security alert can be configured to any of the following three modes, in the BSL configuration.

1. **Factory Reset**- Factory reset erases the complete main and non-main flash memories. The erase is done irrespective of the factory reset mode in the BCR configuration. If certain parts of the flash are static write protected, the BSL cannot erase the complete flash memory.
2. **Disable Bootloader** - Disables the bootloader in the BCR configuration and exits from BSL. No further entry to BSL is possible, unless the BCR non-main configuration is updated to enable the bootloader. The BSL is not disabled if static write protection is enabled for non-main.
3. **Do Nothing** - Does not take any action.

Note

For the factory reset security alert configuration, if non-main is erased, the bootloader password returns to the default. To prevent this, non-main can be write protected. If non-main is left in the erased state, the device is locked and it is not possible to gain access to the device again.

3.5.2 BSL Entry

Entry to Bootloader is allowed only through the Bootcode.

Bootloader can be disabled in BCR configuration.

4 Sample Program Flow with Bootloader

This section explains the typical sequence followed by the BSL host for loading the image through the Bootloader. This sample sequence erases the flash memory and programs new firmware in it.

- Bootloader shall be started through one of the following methods:
 - Pin based invocation (GPIO)
 - Application request
 - Debug Mailbox command
- Once it is invoked, send the connection command to establish connection with the BSL through the desired interface.
- If UART interface is used, the baud rate can be changed to a higher value, to speed up further communication and is optional.
- To erase the flash memory completely use the Mass erase command. Only when there is a need to update non-main flash use factory reset command. Because if the non-main flash is erased and left unprogrammed the device is locked.
- Program the firmware image
- Do the CRC verification of the programmed memory region to check the correctness of data programmed. This is an optional step.
- Application can be started with 'Start application' command.

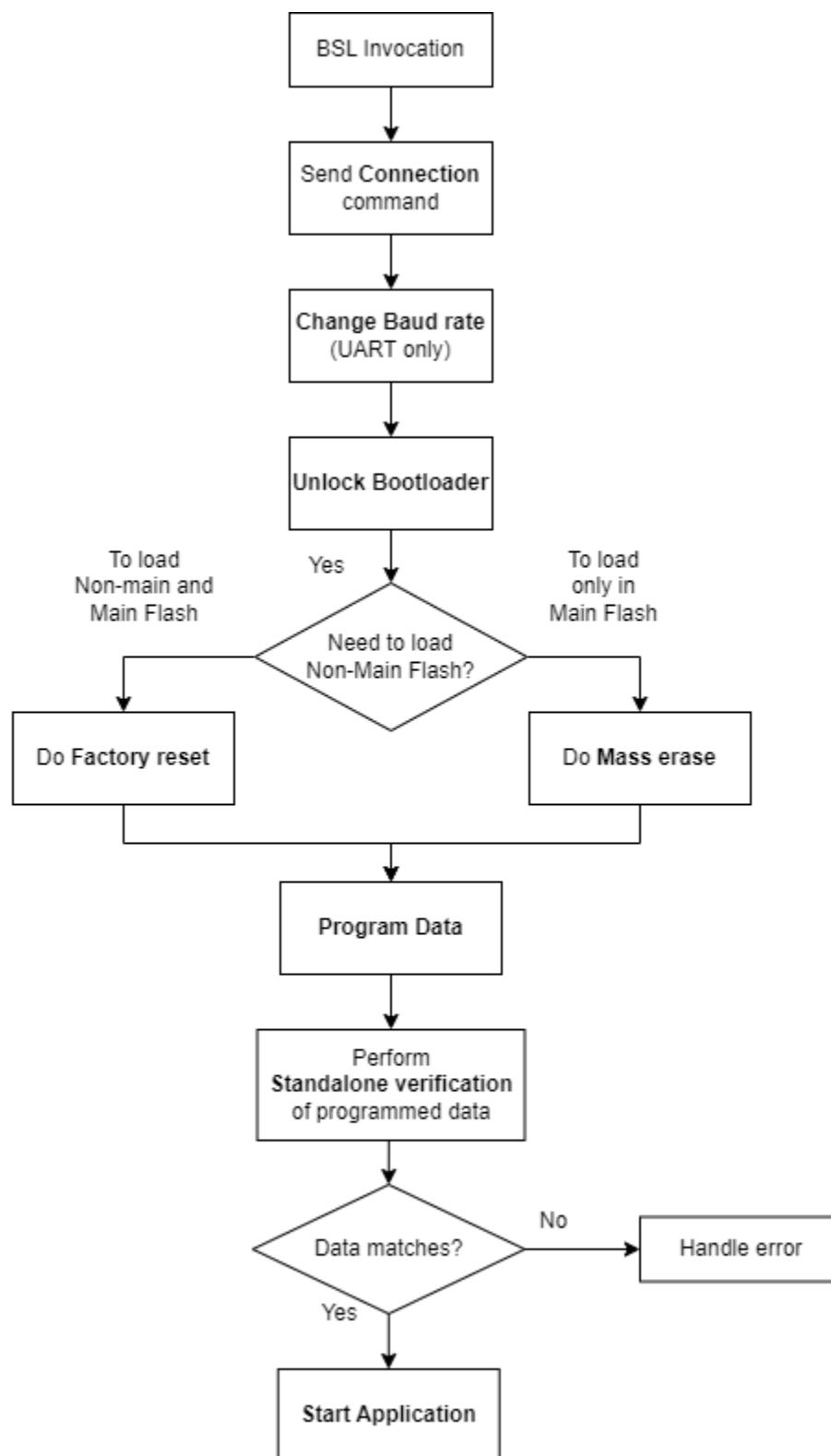


Figure 4-1. BSL Host Sequence

5 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

DATE	REVISION	NOTES
December 2025	*	Initial Release

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to [TI's Terms of Sale](#), [TI's General Quality Guidelines](#), or other applicable terms available either on [ti.com](https://www.ti.com) or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Copyright © 2025, Texas Instruments Incorporated

Last updated 10/2025