User's Guide

MSPM0 부트로더

추상

MSPM0 부트로더(BSL이라고도 함)는 UART 또는 I2C(표준 직렬 인터페이스)를 통해 장치 메모리(플래시 및 RAM)를 프로그래밍하고 확인하는 방법을 제공합니다.

목차

상표

1 BSL 기능 개요

BSL(부트스트랩 로더)은 표준 UART 또는 I2C 직렬 인터페이스를 통해 장치 메모리를 프로그래밍하거나 확인하는 방법을 제공합니다.

직렬 인터페이스를 통해 액세스할 수 있는 BSL의 주요 기능은 다음과 같습니다.

- 플래시 메모리 프로그래밍 및 삭제
- 코드 또는 데이터 영역의 32비트 CRC(최소 영역 크기 1KB)를 반환하여 프로그래밍을 확인할 수 있습니다
- 코드 또는 데이터 읽기를 활성화할 수 있습니다(기본적으로 비활성화됨)
- 주 플래시에 대한 포인터를 통해 펌웨어 버전 번호를 반환할 수 있습니다
- 하드웨어 호출 GPIO를 지정할 수 있습니다
- 액세스는 항상 256비트 암호로 보호됩니다
- 무차별 암호 대입 공격에 대한 보안 경고 처리를 구성할 수 있습니다
- 새 인터페이스를 플래시 플러그인으로 추가할 수 있습니다
- 사용자 지정 부트로더를 사용할 수 있습니다

2 용어

부트로더(BSL) - 장치 메모리에 데이터를 로드하는 데 사용되는 부트 루틴

BCR(bootcode) - BOOTRST 후에 실행되는 시작 루틴으로, 애플리케이션을 실행하도록 장치를 구성합니다

BCR 구성 - 기본 플래시 메모리가 아닌 메모리에 상주하는 Bootcode에 대한 사용자 구성 가능한 모든 매개 변수를 포함하는 구성 구조입니다

BSL 구성 - 기본 플래시 메모리가 아닌 메모리에 상주하는 부트로더에 대한 사용자 구성 가능한 모든 매개 변수를 포함하는 구성 구조입니다

3 BSL 아키텍처

3.1 설계

유효한 부트로더 호출 조건이 감지되면 부트로더가 부트코드에 의해 호출됩니다. BCR 구성의 BSL 모드 필드에서 부트로더가 활성화된 경우에만 호출됩니다.

부트로더가 시작되면 먼저 "Init" 단계를 실행합니다. 여기서 BSL 구성의 초기 확인이 수행되고 장치가 부트로더 작동에 맞게 구성됩니다.

다음으로 부트로더는 "인터페이스 자동 감지" 단계로 진입합니다. 이 단계에서 BSL은 사용 가능한 모든 BSL ROM 인터페이스 및 플래시 플러그인 인터페이스(등록된 경우)를 구성합니다. 그러면 BSL은 모든 인터페이스를 통해 데이터를 하나씩 폴링합니다. 인터페이스 중 하나에서 유효한 연결 패킷이 수신되면 해당 인터페이스는 추가 통신을 위한 활성 인터페이스로 간주되고 다른 모든 인터페이스는 비활성화됩니다. 인터페이스 검색은 10초 동안 수행되며, 감지된 인터페이스가 없으면 장치는 대기 모드로 전환됩니다.

다음으로 BSL은 "명령 수신" 단계로 진입합니다. 이 단계에서 BSL은 호스트의 명령을 무한 루프에서 대기하게 됩니다. 유효한 명령이 수신되면 명령이 처리되고 BSL 코어의 응답이 호스트로 다시 전송됩니다. 그런 다음 다시 루프로 돌아가 다음 명령을 기다리는 등의 작업을 수행합니다. 'Start Application' 명령이 수신되면 부트로더가 시스템 재설정을 트리거하고, 그 후에 부트코드가 실행되고 애플리케이션이 호출됩니다. 또한 이 단계의 시간 초과는 10초입니다. 유효한 명령이 수신되지 않으면 부트로더가 잠기고 절전 모드로 들어갑니다.

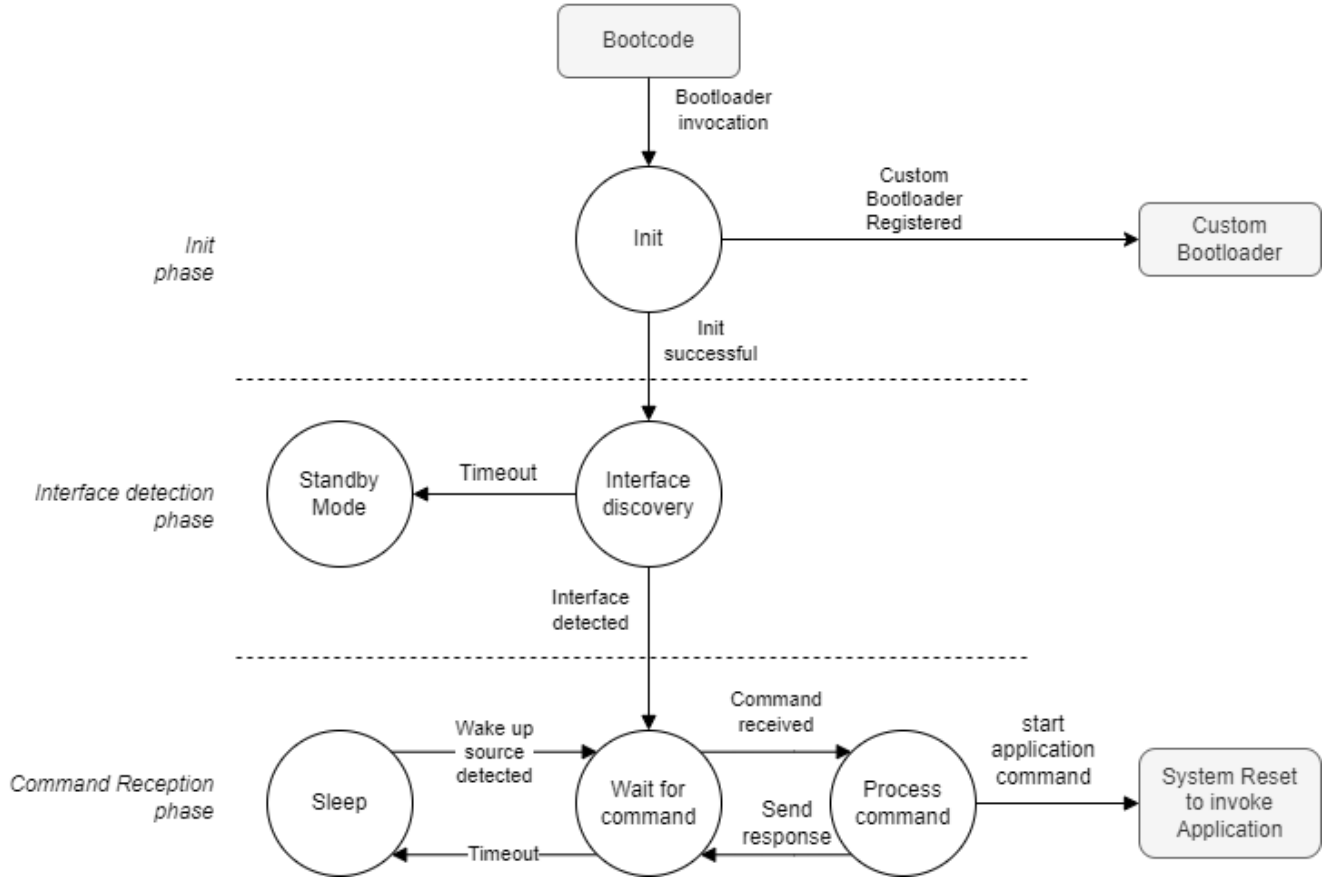

그림 3-1. BSL 아키텍처

3.1.1 타임아웃 기능

아무런 활동이 감지되지 않으면 부트로더가 시간 초과되고 전력 절약을 위해 저전력 모드로 들어갑니다.

이는 다음 두 단계에서 구현되었습니다.

1. 인터페이스 자동 감지

2. 명령 수신

3.1.1.1 인터페이스 자동 감지

인터페이스 감지 단계에서 인터페이스를 통해 10초 동안 유효한 연결 명령이 수신되지 않으면 부트로더가 대기 모드로 들어갑니다.

POR은 이 상태를 벗어나 BSL 호출 조건을 다시 생성하여 부트로더를 사용해야 합니다.

3.1.1.2 명령 수신

명령 수신 단계에서 10초 동안 유효한 명령이 수신되지 않으면 부트로더는 절전 모드로 들어갑니다. 디바이스의 절전 모드를 해제하려면 활성 인터페이스에서 데이터 트랜잭션이 발생해야 합니다.

공격 표면을 줄이기 위해 부트로더는 절전 모드로 들어가기 전에 잠깁니다. 따라서 저전력 모드에서 해제한 후에는 256비트 BSL 암호(부트로더 잠금 해제 명령 참조)를 전송하여 부트로더를 다시 잠금 해제해야 합니다.

NOTE

시간 초과 검사를 위한 부트로더 사용자 LFCLK. 애플리케이션이 외부 클럭을 LFCLK의 소스로 구성하는 경우 애플리케이션 BSL 요청을 통해 호출될 때 BSL에서 동일한 클럭을 사용합니다.

3.2 BSL 호출

BSL 호출 조건이 충족되고 BCR 구성 TRM에서 부트로더가 활성화된 경우 부트 코드에서만 부트로더를 호출할 수 있습니다.

BCR 구성에서 빠른 부팅 모드가 활성화되면 사서함 디버그 명령 및 애플리케이션 요청에 의해서만 부트로더를 호출할 수 있습니다. 실행 시간을 절약하기 위해 다른 검사를 건너뜁니다.

3.2.1 빈 장치

부트코드는 스택 포인터(0x00000000) 및 재설정 벡터(0x00000004) 주소의 지우기 상태를 확인하여 빈 장치를 감지합니다. 두 플래시 메모리 주소가 모두 비어 있는 경우 부트로더가 호출됩니다.

3.2.2 애플리케이션 요청

애플리케이션에서 부트로더를 호출하려면 RESETLEVEL을 BOOTLOADERENTRY로 설정하고 RESETCMD 레지스터를 통해 RESET을 트리거합니다. 이 시퀀스는 시스템을 재설정하고, 부트코드가 실행되고 부트로더가 호출됩니다.

시스템 재설정이 실행되므로 애플리케이션을 종료하는 동안 모든 주변 기기 구성이 재설정됩니다.

3.2.3 GPIO 기반 호출

BSL 호출에 사용되는 GPIO는 비 메인 메모리의 BSL 구성 TRM에서 구성할 수 있습니다.

새로운 장치는 BSL 구성에서 TI가 프로그래밍한 기본 핀 세부 정보를 갖게 됩니다.

GPIO 핀 기반 호출은 BCR 구성에서 비활성화할 수 있습니다. 기본적으로 활성화되어 있습니다.

GPIO는 POR 전에 어설션되어야 하며 상태는 POR 후 최소 T_start ms 동안 유지되어야 합니다. 그런 다음 GPIO 핀 상태를 어설션 해제할 수 있습니다.

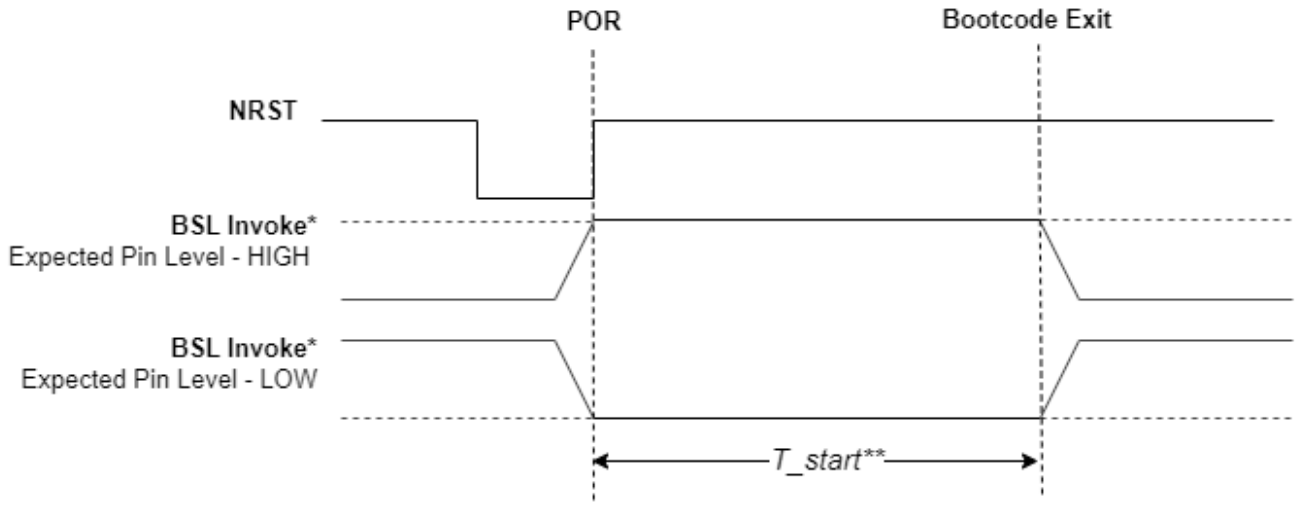

그림 3-2. GPIO에서 호출

** - GPIO 핀은 'BSL 호출'로 사용될 것이며 '예상 핀 레벨'은 BSL 구성에서 구성할 수 있습니다*

*** - T_start는 장치별 데이터 시트에 지정된 콜드 부팅 시작 시간을 나타냅니다*

NOTE

핀 기반 BSL 호출이 활성화된 경우 구성된 GPIO 핀을 하이 또는 로우 상태로 끌어내야 합니다. 예상치 못한 BSL 실행이 발생할 수 있으므로 부동 상태로 두어서는 안 됩니다.

3.2.4 사서함 디버그 명령

디버그 인터페이스를 사용할 수 있는 경우 DSSM(디버그 하위 시스템 사서함)을 통해 부트로더 호출 명령을 전송할 수 있습니다.

DSSM 명령 사용에 대한 자세한 내용은 MSPM0x_Technical_Reference_Manual DSSM 명령 섹션을 참조하십시오.

3.2.5 기타

3.2.5.1 부팅 전 애플리케이션 확인

BCR 구성에서 애플리케이션 CRC 검증이 TRM으로 활성화된 경우, 부트 코드는 지정된 애플리케이션 메모리 범위에 대해 CRC를 확인합니다. CRC가 올바르지 않으면 부트로더가 호출됩니다. 이 경우 부트로더는 활성화된 경우에만 호출됩니다. 그렇지 않으면 부팅 코드가 치명적인 오류를 기록하고 부팅 오류를 유발합니다.

3.3 메모리

3.3.1 SRAM 메모리 사용량

SRAM 메모리 레이아웃에는 부트로더의 작동에 사용되는 메모리가 설명되어 있습니다.

- 데이터 및 스택 섹션 - BSL에서 작업을 위해 사용합니다. 부트로더를 종료하면 SRAM의 이러한 섹션이 지워집니다.
- 가변 버퍼 공간 - BSL 통신 도중 수신/전송된 데이터 패킷을 저장하는 데 사용되는 버퍼 공간입니다

호스트에서 읽기 및 쓰기 액세스가 허용되는 SRAM 메모리는 [SRAM 종료 주소- 0x120]의 BSL 버퍼 시작 주소입니다. 여기에서 SRAM 종료 주소는 각 장치에서 사용 가능한 SRAM 메모리에 따라 결정됩니다. 동일한 SRAM 공간이 가변 버퍼 공간과 공유되므로 SRAM 쓰기/읽기 작업 중에 덮어 쓰일 가능성이 있습니다.

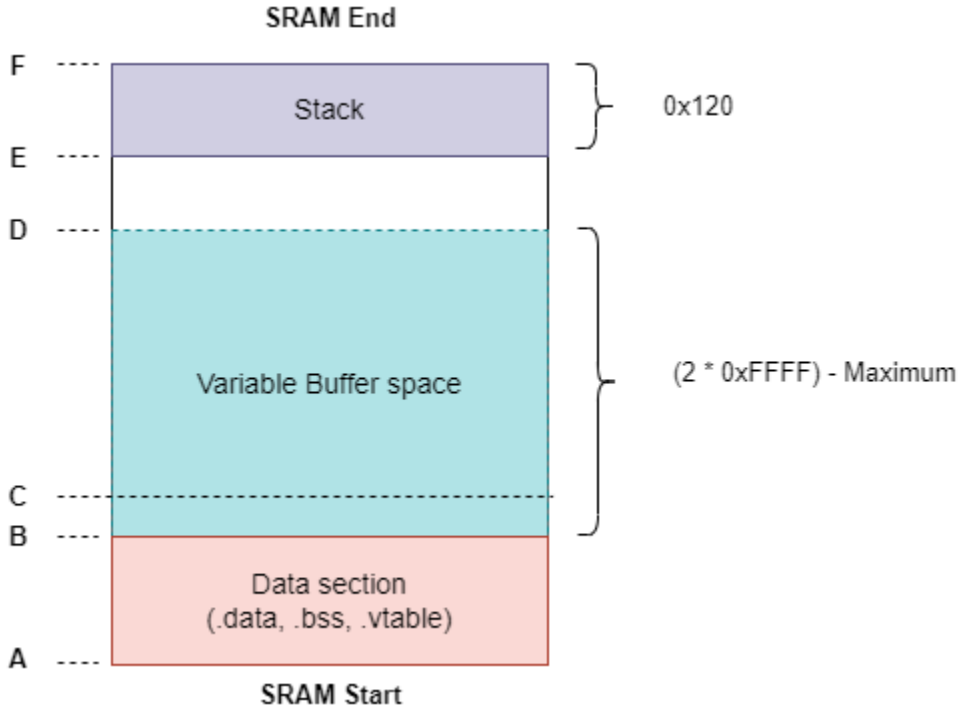

그림 3-3. SRAM 사용량

A - SRAM 시작 주소(0x20000000)

B - 등록된 플래시 플러그인 인터페이스가 없는 경우 'Get Device Info' 명령 응답에서 알 수 있는 'BSL 버퍼 시작 주소'

C - 'Get Device Info' 명령 응답에서 알 수 있는 'BSL 버퍼 시작 주소'. 등록된 플래시 플러그인 인터페이스가 없는 경우 'B'와 동일합니다

D-BSL 버퍼 끝 주소 = 'BSL 버퍼 시작 주소' + (2 * 'BSL 최대 버퍼 크기'). 여기서 BSL 버퍼 시작 주소 및 BSL 최대 버퍼 크기는 'Get Device Info' 명령 응답에서 알 수 있습니다

E - 스택 할당의 시작 주소(E-0x120). 'BSL 최대 버퍼 크기'가 0xFFFF보다 작을 때 'D'와 동일합니다.

F - 장치에서 사용할 수 있는 SRAM 메모리의 끝 주소입니다. 자세한 내용은 장치별 데이터 시트를 참조하십시오.

섹션 B-C:

- BSL 구성에 등록될 때 플래시 플러그인 작업에 할당되는 데이터 섹션입니다.

섹션 C-D:

- 데이터 패킷을 저장하는 데 사용되는 버퍼 공간
- 최대 크기는 (2 * 0xFFFF)입니다.

섹션 C-E:

- BSL 명령을 통해 SRAM 읽기 및 쓰기 작업에 사용 가능한 메모리

3.4 BSL 구성

비메인 플래시 메모리의 BSL 구성을 통해 BSL에서 사용하는 특정 매개 변수를 사용자가 사용자 지정할 수 있습니다.

사용 가능한 구성에 대한 자세한 내용은 MSPM0 기술 참조 설명서 - 구성 메모리 섹션을 참조하십시오.

3.5 BSL 상태

BSL 상태는 BSL이 응답하지 않는 실행 중 발생한 오류에 대한 세부 정보를 제공합니다. 이 32비트 상태 정보는 특정 SRAM 주소(0x200000C0)에 저장되며 BCR 구성에서 디버그 액세스가 활성화된 경우 디버거를 통해 읽을 수 있습니다.

표 3-1. BSL 상태의 해석

바이트 4	바이트 3	바이트 2	바이트 1
예약됨	HW 오류	HW 오류 세부 정보	SW 오류

SW 오류

오류	설명
0x01	BSL 구성 CRC 오류

HW 오류

오류	설명	HW 오류 세부 정보
0x07	NMI 예외	NMIIDX - NMI 인터럽트 인덱스 레지스터 데이터

4 부트로더 프로토콜

4.1 패킷 형식

BSL 데이터 패킷은 계층 구조로 되어 있습니다. BSL 코어 명령에는 BSL이 처리할 실제 명령 데이터가 포함되어 있습니다. 표준 BSL 명령 외에도 PI 코드(주변 기기 인터페이스 코드)라고 하는 각 코어 명령 앞뒤에 래퍼 데이터가 있을 수 있습니다. 이 래퍼 데이터는 사용 중인 주변 장치 및 프로토콜별 정보이며 BSL 코어 명령의 올바른 전송을 지원하는 정보가 들어 있습니다. 래퍼와 코어 명령은 함께 BSL 데이터 패킷을 구성합니다

PI 코드	BSL 코어 데이터	PI 코드

4.2 UART 및 I2C BSL 프로토콜

UART 및 I2C BSL 프로토콜의 데이터 패킷은 다음과 같은 구조로 되어 있습니다.

- 헤더 바이트는 사용된 프로토콜과 패킷 유형(명령 또는 응답 패킷)을 나타냅니다.
- 길이 필드에는 BSL 코어 데이터의 크기가 바이트 단위로 표시됩니다.
- BSL 코어 데이터에는 명령에 필요한 명령/응답 ID 및 주소, 데이터가 포함되어 있습니다
- CRC32 필드에는 BSL 코어 데이터의 데이터에 대해 계산된 CRC가 포함되어 있습니다

PI 코드		BSL 코어 데이터	PI 코드
헤더(1바이트)	길이(2바이트)	BSL 코어 명령/응답	CRC32(4바이트)

코어 데이터 필드에 따라 데이터 패킷은 명령 패킷 또는 응답 패킷으로 분류됩니다.

명령 패킷은 BSL로 전송된 첫 번째 패킷입니다. 두 번째 패킷은 BSL에서 수신된 응답 패킷입니다. 응답 패킷에는 BSL 인식 및 BSL 코어 응답의 두 가지 구성 요소가 포함되어 있습니다. 이 두 가지 중, 전송되는 모든 명령어 패킷에 대해 BSL로부터 확인이 수신됩니다. 그러나 BSL 코어 응답은 명령마다 수신되지는 않습니다.

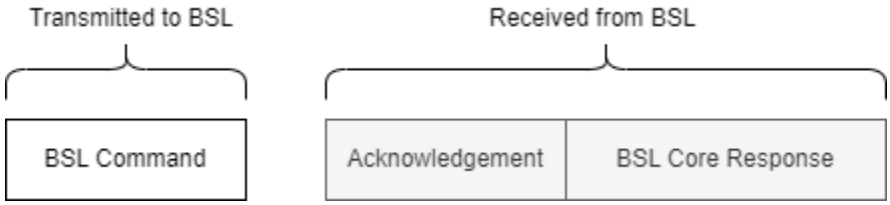

그림 4-1. BSL 프로토콜

4.2.1 BSL 승인

BSL 소프트웨어의 주변 기기 인터페이스 섹션은 BSL 데이터 패킷의 래퍼 섹션을 구문 분석합니다. 데이터 전송에 오류가 있으면 오류 메시지가 즉시 전송됩니다. 모든 데이터가 성공적으로 수신된 후에 ACK가 전송되며, 이는 명령이 올바르게 실행되었음을 의미하지 않고(또는 명령이 유효하다는 의미도 아님), 데이터 패킷이 올바르게 포맷되어 해석을 위해 BSL 코어 소프트웨어로 전달되었음을 의미합니다.

BSL 프로토콜은 전송된 모든 BSL 데이터 패킷 외에 싱글 바이트 확인으로 응답하도록 명령합니다. 표에는 BSL의 승인 응답이 나열되어 있습니다. ACK가 아닌 승인 바이트가 전송되면 BSL은 BSL 데이터 패킷을 보내지 않습니다. 호스트 프로그래머는 승인 오류를 확인하고 전송을 재시도해야 합니다.

데이터	의미
0x00	BSL_ACK(패킷 수신 성공)
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
0x56	BSL_ERROR_UNKNOWN_BAUD_RATE

4.2.2 주변 기기 구성

4.2.2.1 UART

UART는 다음과 같은 구성으로 사용할 수 있습니다.

- UART0이 사용됩니다
- 기본적으로 전송 속도는 9600 bps입니다. 전송 속도 변경 명령으로 업데이트할 수 있습니다.
- 데이터 폭: 8 bit
- 정지 비트: 1
- 패리티 없음
- RXD 및 TXD에 사용되는 핀은 BSL 구성에서 가져온 것입니다

4.2.2.2 I2C

BSL의 I2C 인터페이스는 I2C 타겟 역할을 할 수 있습니다. 호스트는 컨트롤러 역할을 하며 통신을 구동합니다.

- I2C0이 사용됩니다
- I2C 타겟 주소는 기본적으로 0x48입니다. BSL 구성에서 구성할 수 있습니다
- SCL 및 SDA 라인에 외부 풀업이 필요합니다
- SDA 및 SCL에 사용되는 핀은 BSL 구성에서 가져옵니다

NOTE

- BSL은 UART 및 I2C 인터페이스용 비메인에 구성된 핀의 세부 정보를 확인하지 않습니다. BSL은 핀 구성이 올바를 것으로 예상합니다.
- UART와 I2C 모두에 동일한 핀을 사용하지 마십시오.

4.2.2.3 CRC

데이터에 대한 CRC는 다음을 사용하여 계산해야 합니다.

- CRC32-ISO3309 다항식
- 비트 반전 구성
- 초기 시드 - 0xFFFFFFFF

4.3 부트로더 코어 명령

BSL 명령	보호	CMD	시작 주소	데이터(바이트)	BSL 코어 응답
CMD 연결	아니요	0x12	-	-	아니요
CMD 부트로더 잠금 해제	아니요	0x21	-	D1⋯D32(암호)	예
CMD 플래시 범위 지우기	예	0x23	A1...A4	A1...A4(끝 주소)	예
CMD 대량 삭제	예	0x15	-	-	예
CMD 데이터 프로그래밍	예	0x20	A1...A4	D1⋯Dn,	예
CMD 빠른 데이터 프로그래밍	예	0x24	A1...A4	D1⋯Dn,	아니요
CMD 메모리 다시 읽기	예	0x29	A1...A4	L1...L4	예
CMD 공장 초기화	예	0x30	-	D1...D16(암호)	예
CMD 장치 가져오기 정보	아니요	0x19	-	-	예
CMD 독립 실행형 검증	예	0x26	A1...A4	L1...L4	예
CMD 애플리케이션 시작	아니요	0x40	-	-	아니요
CMD 전송 속도 변경	아니요	0x52	-	D1(전송 속도 ID)	아니요

약자:

A1...A4

주소 바이트. 여기서 A1은 최하위 바이트입니다

D1...Dn

데이터 바이트, 여기서 'n'은 BSL 최대 버퍼 크기에 의해 제한됩니다

L1...L4

데이터 길이 바이트. 여기서 C1은 최하위 바이트입니다

4.3.1 Connection

조직

헤더	길이		CMD	CRC32			
0x80	0x01	0x00	0x12	C1	C2	C3	C4

설명

연결 명령은 특정 인터페이스(UART 또는 I2C)를 통해 호스트와 타겟 간의 연결을 설정하는 데 사용되는 첫 번째 명령입니다.

보호

아니요

명령 반환

BSL 승인만

예

호스트: 80 01 00 12 3A 61 44 DE

BSL: 00

4.3.2 장치 정보 가져오기

조직

헤더	길이		CMD	CRC32			
0x80	0x01	0x00	0x19	C1	C2	C3	C4

설명

이 명령은 데이터 트랜잭션에 사용할 수 있는 버전 정보와 버퍼 크기를 가져오는 데 사용됩니다.

보호

아니요

명령 반환

장치 정보가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 장치 정보를 참조하십시오.

예

호스트: 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

4.3.3 부트로더 잠금 해제

조직

헤더	길이		CMD	데이터	CRC32			
0x80	0x21	0x00	0x21	D1...D32	C1	C2	C3	C4

설명

이 명령은 부트로더의 잠금을 해제하는 데 사용됩니다. 부트로더 잠금 해제 후에만 섹션 4.3에 나열된 보호된 명령을 모두 BSL에서 처리합니다.

호스트가 잘못된 암호를 전송하면 BSL이 오류 응답을 전송한 다음 장치가 2초 동안 절전 모드로 들어갑니다. 이 2초 동안 명령어를 수신 또는 처리하지 않는다. 2초 후 호스트는 다음 암호를 보낼 수 있습니다. 잘못된 암호를 3번 보내면 보안 경고 조치가 수행됩니다. 보안 경고 동작은 BSL 구성에서 구성할 수 있습니다. 자세한 내용은 섹션 4.5.1.1의 내용을 참조하십시오.

보호

아니요

데이터

BSL 구성 메모리에 저장된 32바이트 BSL 암호입니다. 새 장치는 모두 기본값인 0xFF를 가집니다.

명령 반환

작업 상태에 대한 메시지가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 BSL 코어 메시지 섹션을 참조하십시오.

예

호스트 80 21 00 21 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 02 AA F0 3D

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

4.3.4 데이터 프로그래밍

조직

헤더	길이		CMD	주소	데이터	CRC32			
0x80	L1	L2	0x20	A1...A4	D1...Dn	C1	C2	C3	C4

설명

프로그램 명령은 A1...A4부터 시작하는 메모리 주소에 데이터 D1~DN을 쓰는 데 사용됩니다. 이 명령어는 차단 쓰기를 수행합니다. 프로그래밍이 완료되면 메시지 응답이 호스트로 전송됩니다.

프로그래밍은 메인 플래시(애플리케이션 메모리), 비 메인 플래시(구성 메모리) 및 SRAM 메모리에 사용할 수 있습니다. 절대 주소 범위에 대한 자세한 내용은 장치별 데이터시트를 참조하십시오.

플래시 메모리는 프로그래밍하기 전에 호스트에 의해 삭제되어야 합니다. 주 플래시 영역 삭제에 대한 자세한 내용은 플래시 범위 삭제, 대량 삭제를 참조하십시오. 비 메인 플래시는 공장 초기화 명령으로만 삭제할 수 있습니다.

플래시 컨트롤러 특성으로 인해 플래시 프로그래밍의 시작 주소와 데이터 길이는 8바이트로 정렬되어야 합니다.

NOTE

호스트가 SRAM 메모리에 완전히 액세스할 수 없습니다. 자세한 내용은 섹션 3.3.1의 내용을 참조하십시오.

보호

예

주소

프로그래밍할 메모리 영역의 시작 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

데이터

지정된 주소에 쓸 데이터 바이트입니다. 전송할 수 있는 데이터의 최대 크기는 장치의 버퍼 크기에 의해 제한됩니다. 버퍼 크기는 장치 정보 가져오기 명령으로 알 수 있습니다.

명령 반환

작업 상태에 대한 메시지가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.1 섹션을 참조하십시오.

예

호스트: 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

4.3.5 빠른 데이터 프로그래밍

조직

헤더	길이		CMD	주소	데이터	CRC32			
0x80	L1	L2	0x24	A1...A4	D1...Dn	C1	C2	C3	C4

설명

빠른 데이터 프로그래밍 명령은 프로그래밍 프로세스 속도를 높이기 위해 비차단 쓰기를 수행한다는 점을 제외하면 데이터 프로그래밍 명령과 동일합니다. BSL은 프로그래밍이 성공적이었음을 나타내기 위해 이 명령에 BSL 코어 메시지 응답을 전송하지 않습니다.

보호

예

주소

프로그래밍할 메모리 영역의 시작 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

데이터

지정된 주소에 쓸 데이터 바이트입니다. 전송할 수 있는 데이터의 최대 크기는 장치의 버퍼 크기에 의해 제한됩니다. 버퍼 크기는 장치 정보 가져오기 명령으로 알 수 있습니다.

명령 반환

BSL 승인.

예

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

BSL: 00

4.3.6 데이터 다시 읽기

조직

헤더	길이		CMD	주소	데이터	CRC32			
0x80	0x09	0x00	0x29	A1...A4	L1...L4	C1	C2	C3	C4

설명

이 명령은 주소 A1...A4에서 시작하는 데이터를 판독하는 데 사용됩니다.

이 명령을 사용하여 데이터를 판독하려면 BSL 구성에서 판독이 활성화되어 있어야 합니다. BSL 구성에서는 기본적으로 비활성화되어 있습니다.

메인 플래시(애플리케이션 메모리), 비 메인 플래시(구성 메모리) 및 SRAM 메모리에 대해 데이터 판독이 가능합니다.

NOTE

호스트가 SRAM 메모리에 완전히 액세스할 수 없습니다. 자세한 내용은 SRAM 메모리 사용을 참조하십시오.

보호

예

주소

다시 읽을 메모리 영역의 시작 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

데이터

읽을 데이터의 크기(바이트 단위: L1...L4)입니다. 여기서 L1은 최하위 바이트입니다. 읽을 수 있는 데이터의 최대 크기는 장치의 버퍼 크기에 의해 제한됩니다. 버퍼 크기는 장치 정보 가져오기 명령에서 알 수 있습니다.

명령 반환

readback 명령이 유효한 경우 요청된 데이터가 있는 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.3의 내용을 참조하십시오.

readback 명령에 잘못된 주소/길이가 있거나 판독기가 비활성화된 경우 해당 오류는 BSL 승인 후 메시지 응답으로 전송됩니다.

예

호스트: 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 FF F6 2B A1 73

4.3.7 플래시 범위 삭제

조직

헤더	길이		CMD	주소	데이터	CRC32			
0x80	0x09	0x00	0x23	A1...A4(시작 주소)	A1...A4(끝 주소)	C1	C2	C3	C4

설명

플래시 범위 삭제 명령은 지정된 플래시 메모리 영역을 삭제하는 데 사용됩니다. 플래시는 섹터 단위로 삭제되므로(1KB) 더 작은 크기의 삭제는 불가능합니다.

시작 주소와 끝 주소가 다른 플래시 섹터에 있을 때 BSL은 이러한 주소가 포함된 섹터를 포함하여 시작 주소와 끝 주소 사이의 모든 플래시 섹터를 삭제합니다.

이 명령은 주 플래시 메모리만 지우는 데 사용할 수 있습니다. 비 메인 삭제는 불가능합니다.

끝 주소는 시작 주소보다 작을 수 없습니다.

보호

예

주소

삭제할 메모리 영역의 시작 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

데이터

삭제할 메모리 영역의 끝 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

명령 반환

작업 상태에 대한 메시지가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.1 섹션을 참조하십시오.

예

호스트: 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

4.3.8 대량 삭제

조직

헤더	길이		CMD	CRC32			
0x80	0x01	0x00	0x15	C1	C2	C3	C4

설명

대량 삭제 명령은 장치에서 사용 가능한 전체 메인 플래시 메모리(애플리케이션 메모리)를 삭제합니다.

BCR 구성 메모리의 대량 삭제 구성은 이 BSL 명령에 영향을 주지 않습니다.

플래시 영역이 BCR 구성 메모리에서 정적 쓰기 보호되어 있으면 해당 영역을 삭제할 수 없습니다.

보호

예

명령 반환

작업 상태에 대한 메시지가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.1의 내용을 참조하십시오.

예

호스트: 80 01 00 15 99 F4 20 40\

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

4.3.9 공장 초기화

조직

헤더	길이		CMD	데이터	CRC32			
0x80	L1	L2	0x30	D1...D16	C1	C2	C3	C4

설명

공장 초기화 명령은 전체 메인 플래시(애플리케이션) 메모리와 비 메인 플래시(구성) 메모리를 삭제합니다.

이 명령을 처리하는 경우 BCR 구성 메모리의 공장 초기화 구성에 영향을 받습니다.

Factory reset은

- 'Enabled'인 경우 암호 없이 허용됨
- 'Enabled with Password'인 경우 암호와 함께 허용됨
- 'Disabled'인 경우 허용되지 않음

플래시 영역이 BCR 구성 메모리에서 정적 쓰기 보호되어 있으면 해당 영역을 삭제할 수 없습니다.

보호

예

데이터

BCR 구성 메모리에 저장된 16바이트 공장 초기화 암호입니다. 기본 암호는 모두 0xFF입니다. 암호는 BCR 구성에서 공장 초기화가 "Enabled with Password"인 경우에만 필요합니다.

명령 반환

작업 상태에 대한 메시지가 포함된 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.1의 내용을 참조하십시오.

CAUTION

공장 초기화를 수행한 후 비 메인 구성이 복원될 때까지 시스템은 장치에 다시 액세스할 수 없는 잠재적인 잠금 상황에 매우 취약합니다.

예

호스트: 80 01 00 30 DE 20 24 0B

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

4.3.10 독립 실행형 검증

조직

헤더	길이		RSP	데이터	CRC32			
0x08	0x05	0x00	0x32	D1...D4	C1	C2	C3	C4

설명

이 명령은 지정된 메모리 범위에 저장된 데이터의 CRC를 확인하는 데 사용됩니다. 이를 통해 프로그래밍된 데이터를 더 빠르게 검증할 수 있습니다. 데이터 크기가 최소 1KB여야 합니다.

메인 플래시(애플리케이션 메모리), 비메인 플래시(구성 메모리) 및 SRAM 메모리에 대해 CRC 검증이 허용됩니다.

NOTE

호스트가 SRAM 메모리에 완전히 액세스할 수 없습니다. 자세한 내용은 섹션 3.3.1의 내용을 참조하십시오.

보호

예

주소

검증할 메모리 영역의 시작 주소입니다. A1...A4, 여기서 A1은 32비트 주소의 최하위 바이트입니다.

데이터

검증할 데이터의 크기(바이트 단위: L1...L4)입니다. 여기서 L1은 최하위 바이트입니다. 1KB <= 크기 <= 64KB.

명령 반환

요청된 메모리 영역에 대해 계산된 CRC 값을 가진 BSL 승인 및 BSL 코어 응답입니다. 자세한 내용은 섹션 4.4.5의 내용을 참조하십시오.

검증 명령에 잘못된 주소/길이가 있는 경우 해당 오류는 BSL 승인 후 메시지 응답으로 전송됩니다. 섹션 4.4.1의 내용을 참조하십시오.

예

호스트: 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

4.3.11 애플리케이션 시작

조직

헤더	길이		CMD	CRC32			
0x80	0x01	0x00	0x40	C1	C2	C3	C4

설명

애플리케이션 시작 명령을 실행하면 시스템 재설정이 실행되면서 부트로더가 종료되고 부트코드가 다시 실행되면서 애플리케이션이 시작됩니다.

보호

아니요

명령 반환

BSL 승인

예

호스트:80 01 00 40 E2 51 21 5B

BSL:00

4.3.12 전송 속도 변경

조직

헤더	길이		CMD	데이터	CRC32			
0x80	0x02	0x00	0x52	D1	C1	C2	C3	C4

설명

이 명령을 사용하여 UART 인터페이스의 전송 속도를 변경할 수 있습니다. 새로운 전송 속도는 이 패킷에 대한 BSL 승인을 전송한 후 적용됩니다.

BSL UART의 기본 전송 속도는 9600bps입니다.

NOTE

전송 속도를 업데이트한 후 Unlock Bootloader 명령과 함께 잘못된 BSL 암호를 전송하면 전송 속도 설정이 기본값으로 되돌아갑니다. 추가 통신은 기본 전송 속도로 이루어져야 합니다.

보호

아니요

데이터

표에 지정된 D1 전송 속도입니다.

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

명령 반환

BSL 승인

4.4 BSL 코어 응답

BSL 응답	RSP	데이터
메모리 다시 읽기	0x30	D1…Dn
장치 정보 가져오기	0x31	D1…D24
독립 실행형 검증	0x32	D1…D4
메시지	0x3B	MSG
자세한 오류	0x3A	D1..D3

약자:

MSG

요청된 작업의 결과를 설명하는 BSL 코어의 응답이 포함된 바이트입니다. 오류 코드이거나 작업 성공 확인일 수 있습니다. BSL이 데이터로 응답해야 하는 경우(예: 메모리, 버전, CRC 또는 버퍼 크기), 작업에 대한 응답이 성공적으로 수행되지 않으며 BSL 코어는 즉시 데이터를 전송합니다.

D1..Dn

데이터 바이트, 여기서 'n'은 BSL 최대 버퍼 크기에 의해 제한됩니다

4.4.1 BSL 코어 메시지

조직

헤더	길이		RSP	데이터	CRC32			
0x08	0x02	0x00	0x3B	MSG	C1	C2	C3	C4

설명

일부 명령의 경우 BSL은 처리된 명령의 상태를 나타내는 메시지 응답을 호스트에 보냅니다. 이 표에는 BSL에서 사용 가능한 모든 메시지가 나열되어 있습니다.

MSG	의미	가능한 원인(1)
0x00	작업 성공	
0x01	BSL 잠김 오류	BSL이 아직 부트로더 잠금 해제 암호 명령으로 잠금 해제되지 않았거나 BSL 잠금 해제 후에 명령 수신 단계에서 시간 초과가 발생했습니다
0x02	BSL 암호 오류	부트 로더 잠금 해제를 위한 잘못된 암호가 전송되었습니다.
0x03	여러 BSL 암호 오류입니다. 보안 경고 조치가 수행됩니다.	부트 로더 잠금 해제를 위한 잘못된 암호가 3번 전송되었습니다.
0x04	알 수 없는 명령	BSL에 지정된 명령이 유효한 명령으로 인식되지 않았습니다
0x05	잘못된 메모리 범위	지정된 메모리 범위가 잘못되었습니다.
0x06	잘못된 명령	BSL에 제공된 명령은 알려진 명령이지만 현재 잘못되어 처리할 수 없습니다.
0x07	공장 초기화 비활성화	BCR 구성에서 공장 초기화가 비활성화되었습니다
0x08	공장 초기화 암호 오류	BCR 구성이 '암호로 활성화됨'으로 공장 초기화되었을 때 올바르지 않거나 공장 초기화 명령과 함께 암호가 전송되지 않았습니다.
0x09	읽기 오류	BCR 구성에서 메모리 읽기가 비활성화되었습니다
0x0A	잘못된 주소 또는 길이 정렬	플래시 프로그래밍의 시작 주소 또는 데이터 길이가 8바이트로 정렬되지 않았습니다
0x0B	독립 실행형 검증의 길이가 잘못되었습니다	독립 실행형 검증을 위해 전송된 데이터 크기가 1KB 미만입니다

(1) 여기에 나열된 가능한 원인이 상태 또는 오류의 유일한 원인은 아닙니다. 여기에는 호스트에 의해 수정될 수 있는 오류의 가능한 소프트웨어 이유만 나열되어 있습니다.

4.4.2 자세한 오류

조직

헤더	길이		RSP	데이터	CRC32			
0x08	0x02	0x00	0x3A	D1..D3	C1	C2	C3	C4

설명

D1 - 오류 유형

D3,D2 - 오류 세부 정보

가능한 값

오류 유형		오류 세부 정보	
평가	설명	평가	설명
0xF0	플래시 오류	0xXX	FLASHCTL.STATCMD 레지스터 값을 포함합니다

4.4.3 메모리 다시 읽기

조직

헤더	길이		RSP	데이터	CRC32			
0x08	L1	L2	0x30	D1...Dn	C1	C2	C3	C4

설명

이 명령은 Readback 명령에 대한 응답으로 요청된 데이터를 반환합니다

데이터

Data D1..Dn, 여기서 'n'은 BSL 최대 버퍼 크기에 의해 제한됩니다.

4.4.4 장치 정보

조직

헤더	길이		RSP	데이터	CRC32			
0x08	0x19	0x00	0x31	D1...D24	C1	C2	C3	C4

설명

이 명령은 Get Identity 명령에 대한 응답으로 버전 정보와 BSL 버퍼 크기를 반환합니다

데이터

ID 바이트	데이터 바이트
명령 인터프리터 버전	[D02-D01]
빌드 ID	[D04-D03]
애플리케이션 버전	[D08-D05]
활성 플러그인 인터페이스 버전	[D10-D09]
BSL 최대 버퍼 크기	[D12-D11]
BSL 버퍼 시작 주소	[D16-D13]
BCR 구성 ID	[D20-D17]
BSL 구성 ID	[D24- D21]

애플리케이션 버전:

32비트 애플리케이션 버전은 BSL 구성에 지정된 주소에서 가져옵니다.

BSL 버퍼 크기:

전송/수신된 BSL 데이터 패킷을 저장하는 데 사용할 수 있는 RAM 데이터 버퍼 크기입니다.

예

호스트: 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

위의 주어진 응답에서,

명령 인터프리터 버전 - 0x0100

빌드 ID - 0x0100

애플리케이션 버전 - 0x00000000

활성 플러그인 인터페이스 버전 - 0x0001

BSL 최대 버퍼 크기 - 0x06C0

BSL 버퍼 시작 주소 - 0x20000160

BCR 구성 ID - 0x00000001

BSL 구성 ID - 0x00000001

4.4.5 독립 실행형 검증

조직

헤더	길이		RSP	데이터	CRC32			
0x08	0x05	0x00	0x32	D1...D4	C1	C2	C3	C4

설명

이 명령은 독립 실행형 검증 명령에 대한 응답으로 CRC 값을 반환합니다

데이터

요청된 메모리 영역에 대해 계산된 32비트 CRC 값입니다. D1...D4, 여기서 D1은 CRC32의 최하위 바이트입니다.

4.5 부트로더 보안

4.5.1 암호로 보호된 명령

메모리의 데이터에 직접 또는 간접적으로 액세스할 수 있는 모든 명령은 암호로 보호됩니다. 암호는 주 메모리가 아닌 메모리의 BSL 구성에서 구성할 수 있습니다.

잘못된 암호를 전송하면 장치는 2초 동안 절전 모드로 전환되며 이 기간 동안 무단 공격을 더 어렵게 하기 위해 어떠한 명령도 받지 않습니다. 잘못된 암호를 3회 전송할 경우 BSL이 보안 경고 조치를 취합니다.

4.5.1.1 보안 경고

보안 경고는 BSL 구성에서 다음 세 가지 모드 중 하나에 대해 구성할 수 있습니다.

1. 공장 초기화 - 공장 초기화는 전체 메인 및 비 메인 플래시 메모리를 삭제합니다. 삭제는 BCR 구성의 공장 초기화 모드와 상관없이 수행됩니다. 플래시의 특정 부분이 정적 쓰기 보호되어 있으면 BSL이 전체 플래시 메모리를 삭제할 수 없습니다.
2. 부트로더 비활성화 - BCR 구성의 부트로더를 비활성화하고 BSL에서 종료합니다. BCR 기본 구성이 부트로더를 활성화하기 위해 업데이트되지 않는 한 BSL에 더 이상 입력할 수 없습니다. 일반 메모리에 대해 정적 쓰기 보호가 활성화된 경우 BSL은 비활성화되지 않습니다.
3. 아무 것도 안 함 - 아무 조치도 취하지 않습니다.

NOTE

공장 초기화 보안 경고 구성의 경우 비 메인이 삭제되면 부트로더 암호가 기본값으로 돌아갑니다. 이를 방지하기 위해 비 메인을 쓰기 보호할 수 있습니다. 비 메인이 삭제된 상태로 남아 있으면 장치가 잠기고 장치에 다시 액세스할 수 없습니다.

4.5.2 BSL 입력

부트로더에 대한 항목은 부트코드를 통해서만 허용됩니다.

BCR 구성에서 부트로더를 비활성화할 수 있습니다.

5 부트로더를 사용한 샘플 프로그램 흐름

이 섹션에서는 부트로더를 통해 이미지를 로드하기 위해 BSL 호스트가 따르는 일반적인 순서에 대해 설명합니다. 이 샘플 시퀀스는 플래시 메모리를 삭제하고 그 안에 새 펌웨어를 프로그래밍합니다.

- 부트로더는 빈 장치 감지, 핀 기반 호출 또는 애플리케이션 요청을 통해 시작할 수 있습니다.
- 호출되면 원하는 인터페이스를 통해 BSL과 연결을 설정하는 연결 명령을 전송합니다.
- UART 인터페이스를 사용하는 경우 전송 속도를 더 높은 값으로 변경하여 추가 통신 속도를 높일 수 있으며, 이는 선택 사항입니다.
- 플래시 메모리를 완전히 지우려면 대량 삭제 명령을 사용합니다. 비 메인 플래시를 업데이트해야 하는 경우에만 공장 초기화 명령을 사용하십시오. 비 메인 플래시가 삭제되고 프로그래밍되지 않은 상태로 두면 장치가 잠기기 때문입니다.
- 펌웨어 이미지 프로그래밍
- 프로그래밍된 메모리 영역의 CRC 검증을 수행하여 프로그래밍된 데이터의 정확성을 확인합니다. 이 단계는 선택 사항입니다.
- 'Start application' 명령으로 애플리케이션을 시작할 수 있습니다.

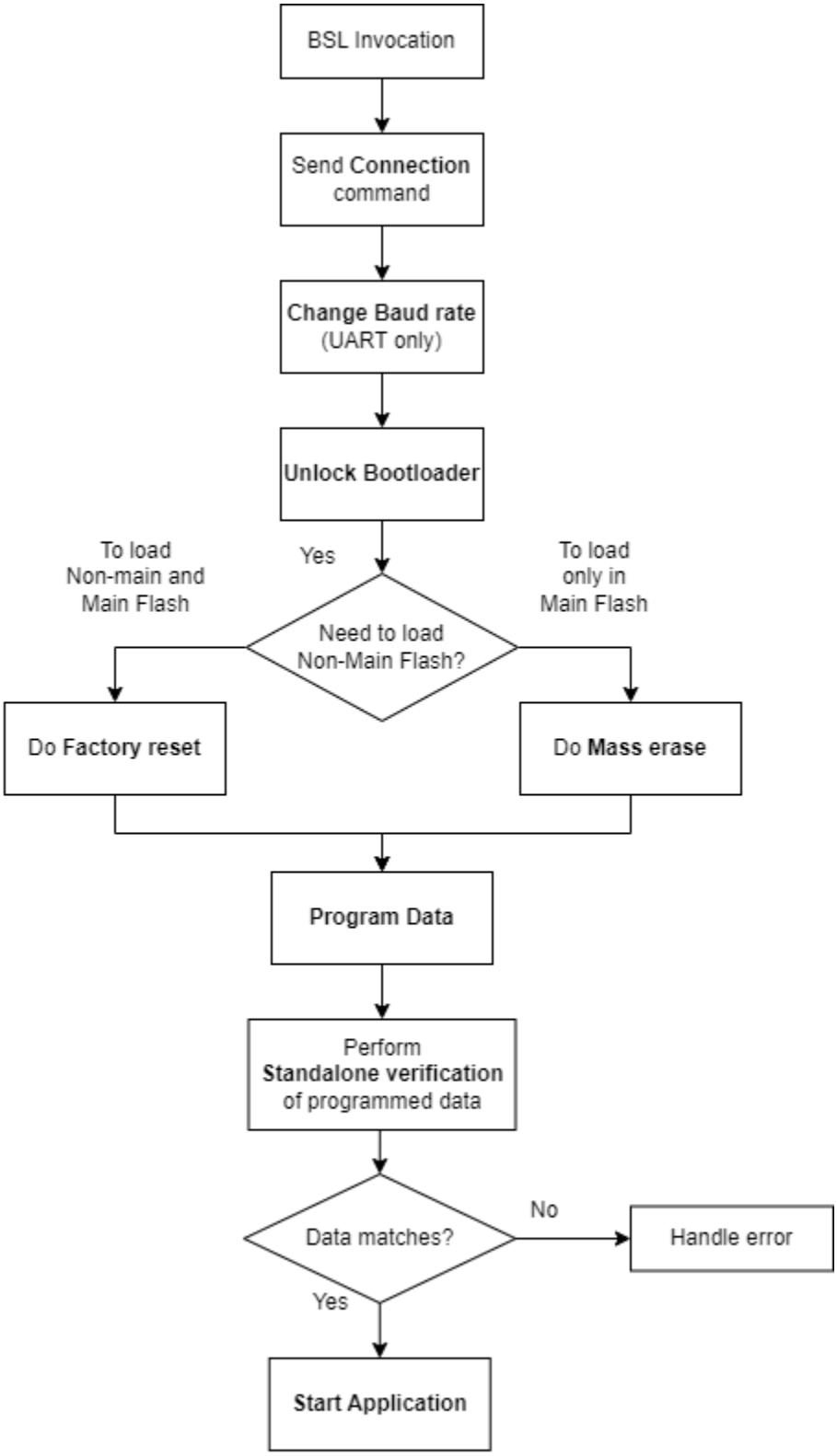

그림 5-1. BSL 호스트 시퀀스

6 보조 부트로더

ROM 부트로더는 사용자 지정 부트로더를 사용할 수 있는 옵션을 제공합니다. 이를 위해서는 메인 플래시 메모리에 사용자 지정 부트로더를 로드하고 비 메인 플래시의 BSL 구성에 등록합니다. 다음에 ROM BSL이 호출되면 보조 부트로더 구성 필드와 해당 필드를 확인하게 되며, 컨트롤은 ROM BSL로 다시 돌아가지 않을 것으로 예상됩니다.

BCR 구성의 BSL 모드 구성은 2차 BSL에도 적용할 수 있습니다. 이 설정을 비활성화하면 보조 BSL이 호출되지 않습니다.

부트로드 프로세스 중에 의도하지 않은 삭제가 발생하지 않도록 사용자 지정 BSL이 로드된 플래시 메모리 영역은 BCR 구성에서 쓰기 보호되어야 합니다. 보조 부트로더 사용 사례의 경우 비 메인 쓰기 보호는 선택 사항입니다. 그러나 삭제 후 보조 부트로더 포인터가 제대로 복원되어야 합니다.

NOTE

보조 부트로더를 삭제하면 장치가 잠길 수 있습니다. 따라서 보조 부트로더 메모리 영역을 작성해야 합니다.

6.1 보조 부트로더 예

보조 부트로더는 참조용 SDK 예제의 일부로 제공됩니다. 이 섹션에서는 이에 대해 자세히 설명합니다.

설명

이 샘플 보조 부트로더는 장치의 기본 BSL(ROM BSL)과 동일한 BSL 프로토콜 형식을 사용하여 메모리의 데이터 프로그래밍/확인을 지원하며 ROM BSL과 동일한 방식으로 호출할 수 있습니다.

다음과 같은 주요 기능을 지원합니다

- 데이터 프로그래밍
- 플래시 메모리 삭제
- 데이터 다시 읽기
- CRC 검증
- 애플리케이션 시작

USB를 사용하여 PHY와 통신합니다.

이 예에서는 보조 부트로더 구현과 등록에 대해 다룹니다. 따라서 이 이미지가 장치에 로드되면 장치의 기본 부트로더를 사용할 수 없습니다. 보조 부트로더만 활성화됩니다. 기본 부트로더를 사용하도록 장치를 되돌리려면 SWD_Factory_Reset 명령을 사용해야 합니다(하위 시스템 디버그 사서함을 통해 공장 초기화). .

사용 예

- UART_RX 및 UART_TX를 BSL 호스트에 연결합니다(UART를 사용하는 모든 마이크로컨트롤러).
- 예제를 컴파일하고 로드합니다.
- BSL 호출 핀 또는 기타 호출 메서드를 사용하여 BSL 호출 조건을 만듭니다.
- 호스트에서 **GetDeviceInfo** 명령을 보냅니다.
- 장치가 버전 정보 및 SRAM 버퍼 공간으로 응답해야 합니다.
- 마찬가지로 메모리에 있는 데이터를 프로그래밍하기 위해 삭제, 프로그램, 검증 명령을 전송합니다.

소프트웨어 파일 세부 정보

파일 이름	세부 정보
secondary_bsl.c	BSL 작동에 필요한 주변 기기를 초기화합니다. 통신 인터페이스에서 명령 패킷을 수신하여 명령 처리 계층으로 전달합니다. 또한 BSL 구성 메모리에 2차 부트로더를 등록하는 일도 수행합니다.
bsl_ci.c	명령 패킷을 해석하고, 명령을 처리하고, 응답을 호스트에 다시 보냅니다
bsl_ci.h	BSL 명령 및 응답의 정의가 포함되어 있습니다. bsl_ci.c의 함수 선언도 볼 수 있습니다.
bsl_uart.c	호스트와 BSL 코어 간의 통신을 처리합니다
bsl_uart.h	bsl_uart.c의 BSL 승인 및 함수 선언에 대한 정의가 포함되어 있습니다
ti_msp_dl_config.h	UART 핀, 사용된 주변 장치의 기본 주소 등과 같은 장치별 구성이 포함되어 있습니다
boot_config.h	BCR 및 BSL 구성 구조를 포함하고 있습니다

파일 이름	세부 정보
factory_config.c	SRAM 메모리 크기와 같이 출하 시 구성된 장치별 데이터를 가져오는 기능을 구현합니다.
factory_config.h	factory_config.c의 공장 구성 구조 및 함수 선언을 포함합니다
startup_mspm0x_ticlang	벡터 테이블, 재설정 처리기 및 기타 핸들러가 포함된 시작 파일입니다
mspm0x.cmd	보조 부트로더 이미지가 메모리 및 해당 이미지가 작동해야 하는 SRAM 영역에 있어야 하는 메모리 영역을 지정하는 링커 명령 파일입니다.

사용자화

이 예제는 보조 부트로더에 대한 레퍼런스 구현을 제공합니다. 필요에 따라 사용자 지정할 수 있습니다. BSL 코어 계층(secondary_bsl.c, bsl_ci.c) 또는 인터페이스 계층(bsl_uart.c)은 사용자 지정이 수행되는 주요 위치입니다.

따라야 할 단계

- 필요에 따라 코드를 수정합니다
- 변경이 완료되면 코드를 컴파일합니다
- BCR 구성에서 플래시 쓰기 보호 설정을 적절하게 수정합니다
- BCR 구성에 대한 CRC를 계산하고 새 CRC 값을 저장합니다
- 코드를 다시 컴파일합니다
- 사용자 지정된 BSL 이미지를 로드합니다

7 인터페이스 플러그인

ROM 부트로더는 ROM BSL 코어에 플래시 플러그인으로 사용자 지정 인터페이스 구현을 추가할 수 있는 옵션을 제공합니다. 이는 전체 BSL 코어를 다시 구현하지 않고 인터페이스를 사용자 지정할 수 있는 이점을 제공합니다.

이 기능을 사용하면 다음 중 하나를 수행할 수 있습니다.

- ROM BSL에서 사용할 수 없는 새 인터페이스를 자동 감지를 위한 인터페이스 목록에 추가할 수 있습니다. 예: SPI, CAN 등(또는)
- ROM 인터페이스 구현(UART/I2C)을 재정의할 수 있습니다

이 옵션을 사용하려면 플래시 플러그인 이미지를 메인 플래시에 로드하고 비 메인 플래시 메모리의 BSL 구성에 등록해야 합니다. MSPM0xx 기술 참조 설명서 - 구성 메모리(비 메인) 섹션을 참조하십시오.

또한 BCR 구성에서 플러그인이 로드되고 비 메인 구성 메모리가 로드되는 플래시 메모리 영역은 플래시 플러그인 및 비 메인 구성 메모리의 등록이 부트 로드 프로세스 중에 삭제되지 않도록 쓰기 보호되어야 합니다.

NOTE

메인 플래시에서 플래시 플러그인 영역을 삭제하면 장치가 잠길 수 있습니다. 따라서 쓰기 보호가 필요합니다.

7.1 구현

플러그인은 데이터 처리 및 BSL 호스트와의 통신을 처리해야 합니다. 이 인터페이스 플래시 플러그인은 다음 4개의 API를 통해 ROM BSL 코어와 결합됩니다.

- Init
- Receive
- Transmit
- Deinit

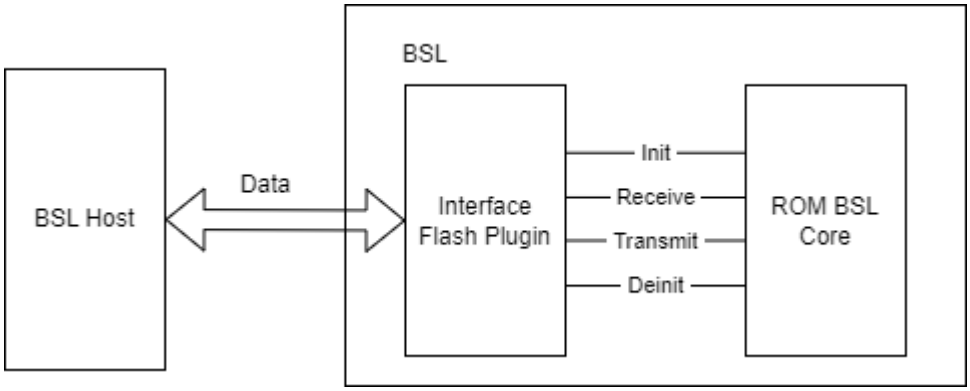

그림 7-1. 플러그인 구현

플래시 플러그인 이미지는 다른 애플리케이션과 마찬가지로 구축되어야 하며 주 플래시에 로드됩니다. 그러나 애플리케이션과 달리 시작 코드 또는 main 함수는 실행되지 않습니다. 위에 언급한 4개의 API만 비 메인 메모리에 BSL 구성에 등록된 후크를 통해 ROM BSL에서 호출됩니다.

7.1.1 Init

시제품

uint16_t init(uint8_t* buffer, uint16_t bufferSize);

buffer - BSL 코어에서 전송된 SRAM 데이터 버퍼에 대한 포인터입니다.

bufferSize - 데이터 버퍼로 사용할 수 있는 SRAM 메모리 크기의 1/2입니다. 두 개의 버퍼가 전송에 1을, 수신에 1을 사용하는 경우 한 버퍼의 주소는 **'buffer'**이고 다른 버퍼의 주소는 **'buffer+bufferSize'**입니다.

return - 16비트 플러그인 버전 정보를 반환합니다.

설명

Init 함수는 원하는 인터페이스를 구성하고 데이터 처리를 위한 매개 변수를 초기화하는 작업을 수행해야 합니다. 또한 사용되는 다른 전역 변수를 초기화하는 시작 코드가 실행되지 않으므로 해당 변수를 초기화하는 작업도 처리해야 합니다.

7.1.2 Receive

시제품

uint32_t receive(void);

return - 수신된 데이터 패킷의 32비트 시작 주소를 반환합니다. 데이터 패킷은 ROM BSL 프로토콜 섹션 4에 설명된 것과 동일한 형식을 가져야 합니다.

설명

수신 기능은 BSL 호스트에서 데이터 패킷을 읽는 것을 처리해야 합니다. 완전한 패킷이 수신되어 데이터의 정확성 확인(데이터의 CRC 검증)이 된 경우에만 BSL 코어와 패킷을 공유해야 합니다. 또한 주소는 데이터 패킷당 한 번만 공유해야 합니다. 이 함수가 BSL 코어에 의해 호출되는 경우, 수신된 데이터 패킷이 없거나 진행 중일 때 '0'을 반환해야 합니다.

데이터 패킷이 아무런 문제 없이 성공적으로 수신된 경우 ROM BSL 플러그인이 제공하는 것과 같이 호스트에 ACK가 전송됩니다(ROM BSL acknowledgments 참조). 문제가 발생할 경우 NACK를 통해 호스트로 보고되어야 하며 패킷은 ROM BSL 코어와 공유되어서는 안 됩니다.

7.1.3 Transmit

시제품

uint8_t send(uint8_t* data, uint16_t length);

data - 호스트로 전송할 BSL 코어 응답 패킷에 대한 포인터입니다. 섹션 4 섹션에 설명된 것과 동일한 형식을 갖습니다.

length - CRC 4바이트의 길이를 제외한 BSL 코어 응답 패킷의 길이입니다.

return - 전송이 성공하면 '1'을 반환합니다. 전송에 실패한 경우 '0'입니다.

설명

transmit은 BSL 코어 응답 패킷을 호스트에 전송해야 합니다. 또한 패킷에 대한 CRC를 계산하고 추가하는 작업도 처리해야 합니다. 전송 API는 데이터를 완전히 전송한 후에만 반환해야 합니다.

7.1.4 Deinit

시제품

bool deinit(void);

Return

deinit가 완료되면 'True'를 반환합니다.

설명

이 함수는 인터페이스 구성을 재설정하고, 등록된 경우 인터럽트 처리기를 등록 해제합니다.

7.1.5 중요 참고

플래시 플러그인을 개발하는 동안 유의해야 할 중요 사항

1. 플래시 플러그인을 개발하는 동안 다음 사항을 고려해야 합니다.
2. 플래시 플러그인이 로드되는 주 플래시 메모리 영역은 정적 쓰기 방지되어야 합니다
3. 모든 글로벌 변수는 'Init' 기능으로 초기화되어야 합니다
4. 4개의 플러그인 API에 대한 함수 프로토타입은 BSL 사용 설명서에 지정되어 있어야 합니다.
5. SRAM 메모리 사용량
    a. VTOR - SRAM 시작(0x20000000). 인터럽트를 사용하는 경우 ROM BSL이 해당 주소 공간을 사용하기 때문에 VTOR는 SRAM의 시작 위치에 배치해야 합니다
    b. 스택 시작 주소 - 장치에서 사용 가능한 SRAM 메모리의 끝입니다
    c. 스택 크기 - ROM BSL 스택 크기를 초과하면 안 됩니다
    d. 데이터 섹션(.data, .bss) - 장치에 등록된 플래시 플러그인이 없는 경우 Get Device Info 명령이 반환하는 'BSL 버퍼 시작 주소'는 데이터 섹션의 시작 주소여야 합니다.

e. 데이터 섹션 크기 - 데이터 섹션(.data, .bss)에서 소비되는 크기는 BSL 비기본 구성 메모리에 구성해야 합니다.

7.2 플래시 플러그인 유형

플래시 플러그인에 사용되는 통신 인터페이스 유형은 BSL 구성 메모리의 BSLPLUGINCFG.PLUGINTYPE 필드에 구성됩니다(섹션 8의 기술 참조 설명서 참조).

플러그인 유형에 ROM 인터페이스 유형(UART 또는 I2C)이 있으면 플래시 플러그인이 ROM 구현을 재정의합니다. ROM에서 사용할 수 없는 새 유형(예: SPI)이 있는 경우에는 플러그인이 새 인터페이스로 추가됩니다.

표 7-1. 플러그인 유형 값

X.	인터페이스	플러그인 유형
1	UART	0x1000
2	I2C	0x2000
3	기타 인터페이스	0xXXXX

그림 7-2에 나와 있는 것처럼 플래시 플러그인 유형을 확인한 후 업데이트된 인터페이스 목록을 자동 감지에 사용합니다. 플래시 플러그인 등록 후에도 해당 인터페이스가 재정의되지 않으면 ROM 인터페이스를 사용할 수 있습니다.

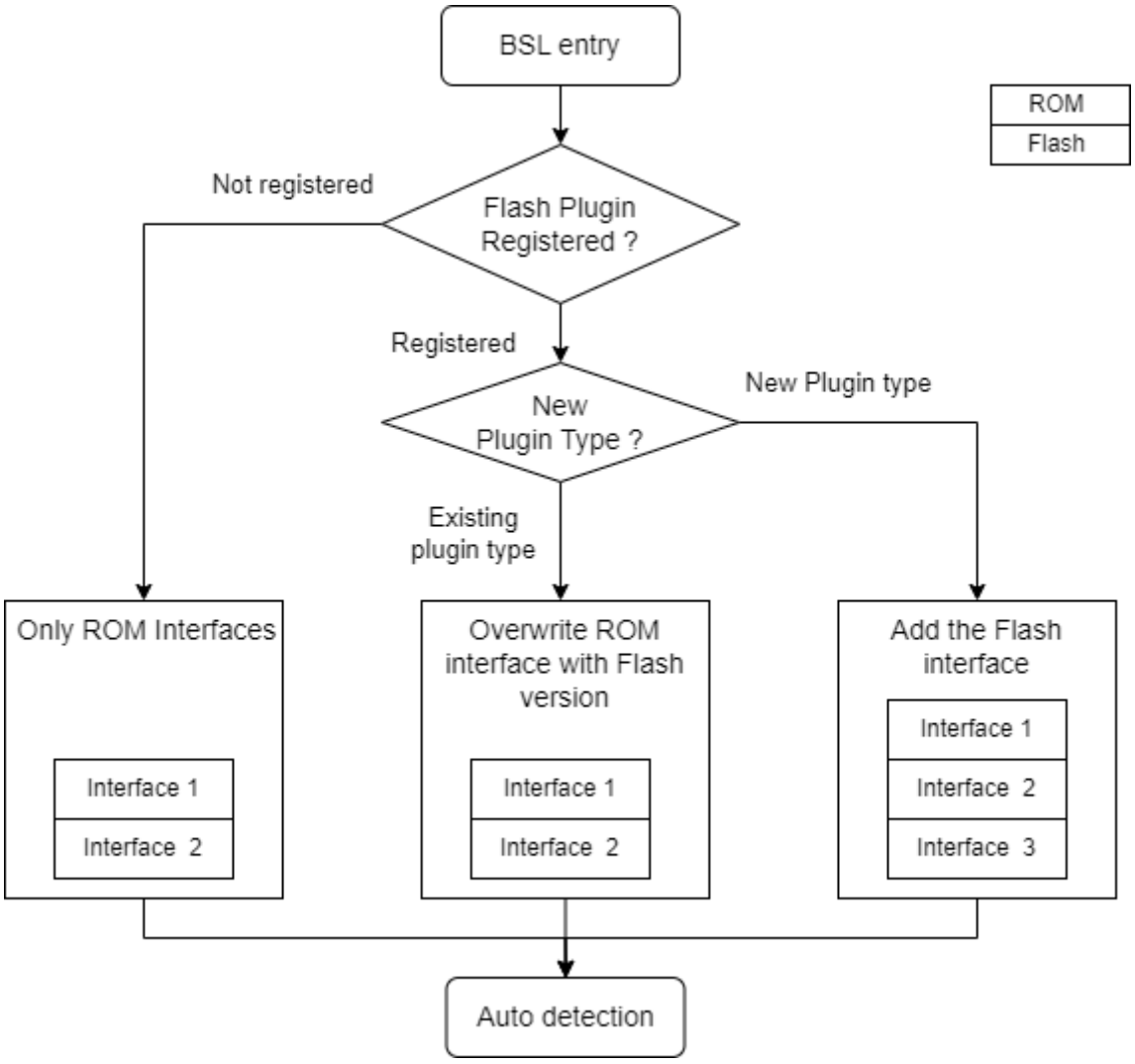

그림 7-2. 플래시 플러그인

7.3 기존 인터페이스 재정의

이 기능을 사용하면 사용자가 기존 ROM BSL 인터페이스, UART 및 I2C를 플래시 버전으로 재정의할 수 있습니다. 이 경우 플래시 플러그인이 로드되면 ROM 버전이 비활성화되며 사용할 수 없습니다. ROM 인터페이스를 사용하도록 디바이스를 되돌리려면 SWD_Factory_Reset(디버그 서브시스템 사서함을 통해 공장 재설정)을 사용합니다.

7.3.1 UART 인터페이스 플래시 플러그인 예

UART 통신을 사용하는 샘플 플래시 플러그인은 참조할 수 있도록 SDK 예제 SDK의 일부로 제공됩니다. 이 섹션에서는 이에 대해 자세히 설명합니다.

설명

UART 인터페이스 플래시 플러그인은 다음 4개의 API 후크를 통해 BSL 호스트와 ROM BSL 간 데이터 트랜잭션을 처리합니다.

- BSL_PI_UART_init
- BSL_PI_UART_receive
- BSL_PI_UART_send
- BSL_PI_UART_deinit

UART 플래시 플러그인은 주로 필요할 때 사용자 지정 구현으로 ROM BSL UART 인터페이스를 재정의하는 데 사용됩니다.

사용 예

- UART_RX 및 UART_TX를 BSL 호스트에 연결합니다(UART 인터페이스를 사용하는 모든 마이크로컨트롤러).
- 예제를 컴파일하고 로드합니다.
- BSL 호출 핀 또는 기타 호출 메서드를 사용하여 BSL 호출 조건을 만듭니다.
- 호스트에서 **연결** 명령을 전송합니다. 성공하면 BSL 승인이 수신됩니다.
- 호스트에서 **GetDeviceInfo** 명령을 전송합니다.
- BSL은 UART 인터페이스 플래시 플러그인 버전 정보로 응답합니다.
- 마찬가지로 메모리에 있는 데이터를 프로그래밍하기 위해 삭제, 프로그램, 검증 명령을 전송합니다.

소프트웨어 파일 세부 정보

파일 이름	세부 정보
bsl_uart.c	호스트와 BSL 코어 간의 통신을 처리합니다. 4개의 인터페이스 API Init, Receive, Send 및 Deinit을 정의합니다.
bsl_uart.h	bsl_uart.c의 BSL 승인 및 함수 선언에 대한 정의가 포함되어 있습니다
ti_msp_dl_config.h	UART 핀, 클록 구성 등과 같은 장치별 구성이 포함되어 있습니다.
boot_config.h	BCR 및 BSL 구성 구조를 포함하고 있습니다
startup_mspm0x_ticlang	기본 처리기 함수 정의만 포함하는 시작 파일입니다. 일반적인 시작 파일과는 달리 인터럽트 벡터 테이블 또는 재설정 처리기가 없습니다. 이러한 기능은 플래시 플러그인에서 사용되지 않으며 메모리 소비를 줄이기 위해 제거되기 때문입니다.
mspm0x.cmd	플래시 플러그인 이미지가 메모리와 SRAM에서 상주하는 메모리 영역을 지정하는 링커 명령 파일입니다.

사용자화

이 예에서는 플래시 플러그인에 대한 레퍼런스 구현을 제공합니다. 필요에 따라 사용자 지정할 수 있습니다. 인터페이스 플래시 플러그인 API가 중요한 변경 영역입니다.

따라야 할 단계:

- 필요에 따라 플래시 플러그인 API를 수정합니다
- 변경이 완료되면 코드를 컴파일합니다
- API 포인터가 업데이트된 경우 BSL 구성에 대한 CRC를 계산합니다
- BCR 구성에서 플래시 쓰기 보호 설정을 적절하게 수정합니다
- BCR 구성에 대한 CRC를 계산합니다
- 새 CRC를 BCR 및 BSL 구성에 저장합니다.
- 코드를 다시 컴파일합니다.
- 플래시 플러그인 이미지를 로드합니다

8 참고 문헌

1. MSPM0 G 시리즈 80MHz 마이크로컨트롤러 기술 레퍼런스 매뉴얼
2. MSPM0 L 시리즈 32MHz 마이크로컨트롤러 기술 레퍼런스 매뉴얼
3. MSPM0 SDK

9 개정 내역

참고: 이전 개정판의 페이지 번호는 현재 버전의 페이지 번호와 다를 수 있습니다

날짜	개정	참고
2023년 2월	*	초기 발매

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), 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 will 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 or other applicable terms available either on 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.

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

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