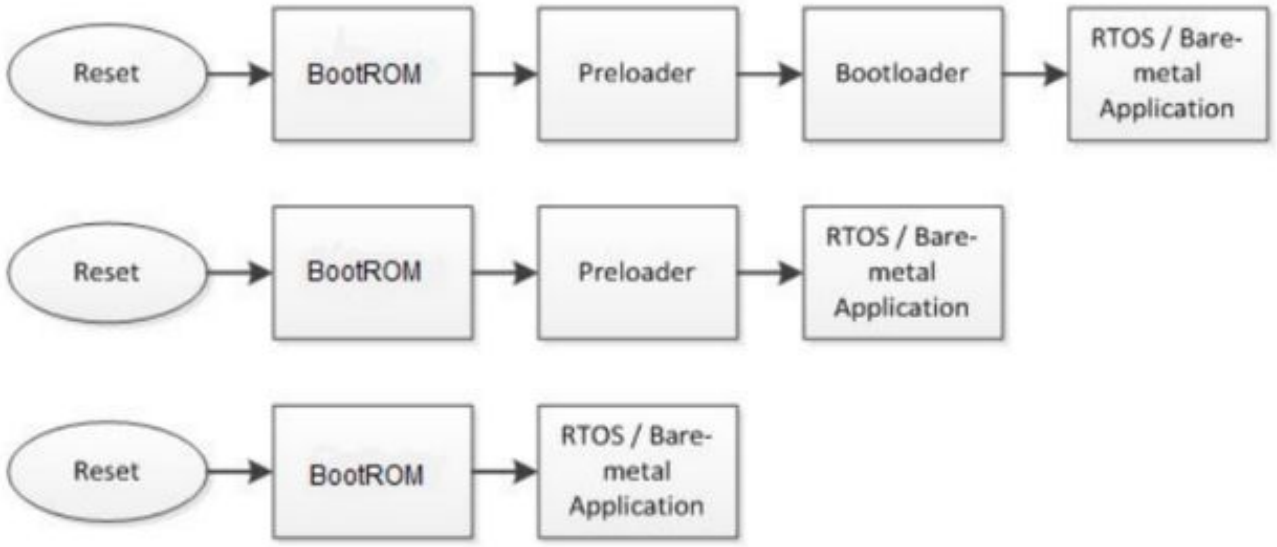
부트로더

Hancheol Cho

부트로더란?

- 펌웨어가 실행되기전에 초기화기능 수행
  - DDR 메모리 초기화
  - Linux 커널 로딩
- 펌웨어 업데이트 수행
  - 원하는 통신 방식으로 JTAG 같은 툴 없이 펌웨어 업데이트 진행



부트로더 종류

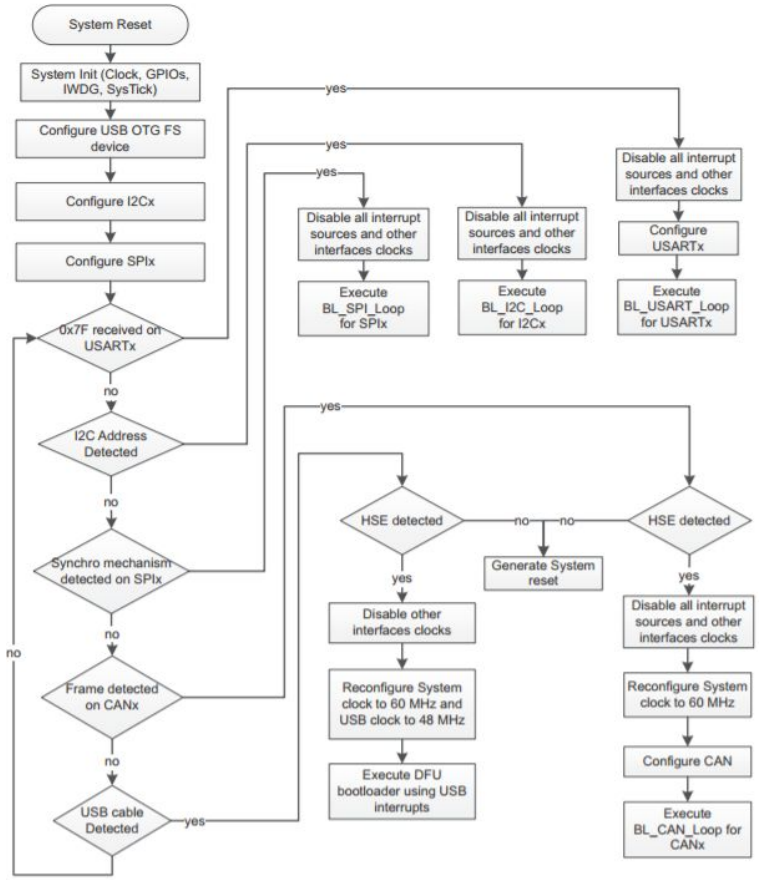
- On-Chip 부트로더
  - MCU에 기본으로 내장된 부트로더로 별도의 부트로더를 제작하지 않더라도 사용 가능 (STM BOOTLOADER 등)
- 사용자 부트로더
  - 직접 제작한 부트로더로 다양한 프로토콜과 통신 방식 지원 가능

- STM BOOTLOADER
  - STM32 에 기본 포함된 부트로더로 다양한 인터페이스 지원
  - BOOT 핀 상태로 부트로더 실행 모드 선택함

Boot mode selection		Boot area
BOOT	Boot address option bytes	
0	BOOT_ADD0[15:0]	Boot address defined by user option byte BOOT_ADD0[15:0] ST programmed value: Flash on ITCM at 0x0020 0000
1	BOOT_ADD1[15:0]	Boot address defined by user option byte BOOT_ADD1[15:0] ST programmed value: System bootloader at 0x0010 0000

부트로더 종류

- STM BOOTLOADER
  - 부트로더 부팅 과정



부트로더 구성요소

부트로더 구현시 유의사항

부트로더 구현 순서

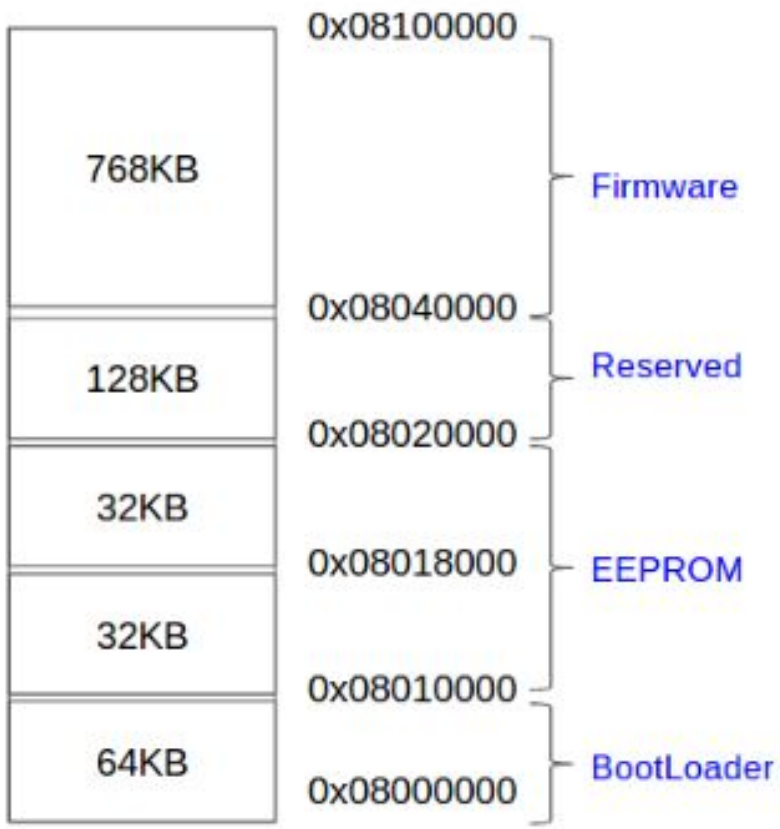
FLASH 메모리 맵

- STM32F746의 FLASH 메모리 구성
  - FLASH 메모리는 Sector 단위로 지워지기 때문에 Sector 단위로 부트로더 용량 고려 필요

Block	Name	Bloc base address on AXIM interface	Block base address on ICTM interface	Sector size
Main memory block	Sector 0	0x0800 0000 - 0x0800 7FFF	0x0020 0000 - 0x0020 7FFF	32 KB
	Sector 1	0x0800 8000 - 0x0800 FFFF	0x0020 8000 - 0x0020 FFFF	32 KB
	Sector 2	0x0801 0000 - 0x0801 7FFF	0x0021 0000 - 0x0021 7FFF	32 KB
	Sector 3	0x0801 8000 - 0x0801 FFFF	0x0021 8000 - 0x0021 FFFF	32 KB
	Sector 4	0x0802 0000 - 0x0803 FFFF	0x0022 0000 - 0x0023 FFFF	128 KB
	Sector 5	0x0804 0000 - 0x0807 FFFF	0x0024 0000 - 0x0027 FFFF	256 KB
	Sector 6	0x0808 0000 - 0x080B FFFF	0x0028 0000 - 0x002B FFFF	256 KB
	Sector 7	0x080C 0000 - 0x080F FFFF	0x002C 0000 - 0x02F FFFF	256 KB
Information block	System memory	0x1FF0 0000 - 0x1FF0 EDBF	0x0010 0000 - 0x0010 EDBF	60 Kbytes
	OTP	0x1FF0 F000 - 0x1FF0 F41F	0x0010 F000 - 0x0010 F41F	1024 bytes
	Option bytes	0x1FFF 0000 - 0x1FFF 001F	-	32 bytes

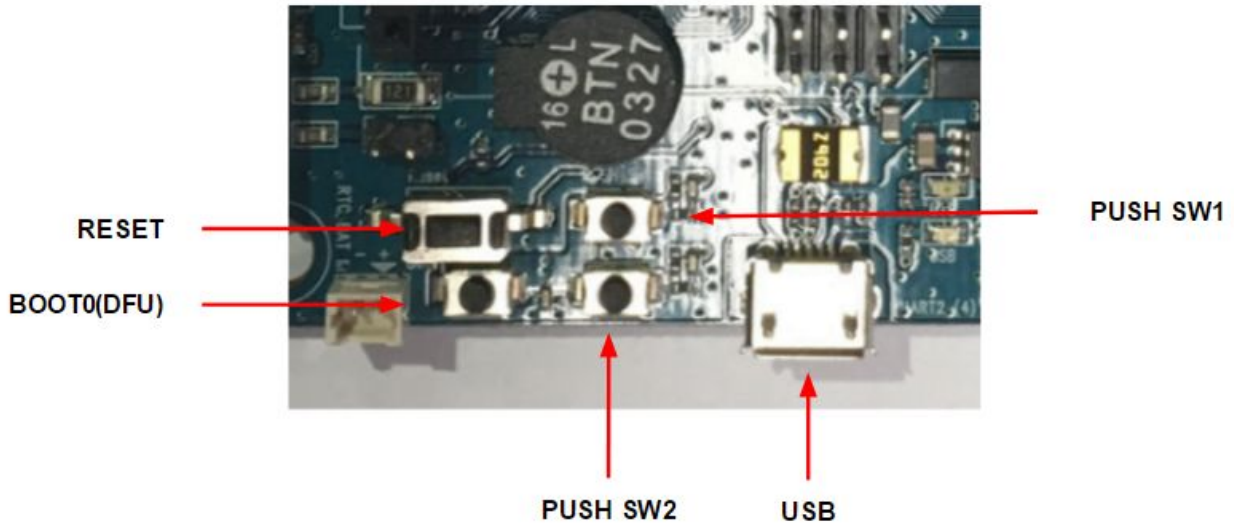
FLASH 메모리 맵

- FLASH 메모리의 영역 할당



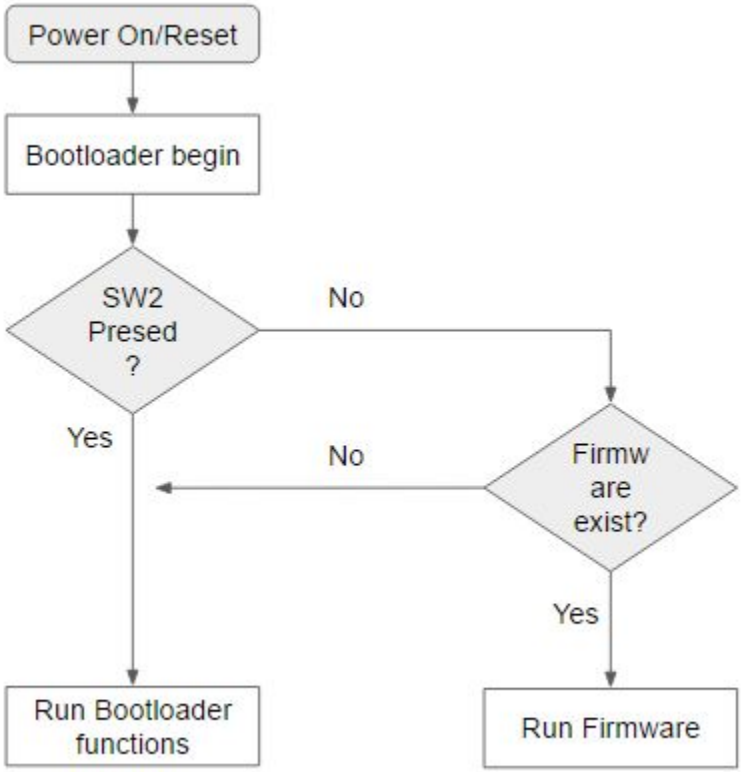
부트로더 실행

- 부트로더 실행을 위한 별도 버튼 할당
  - 리셋 혹은 전원 On시에 부트로더를 강제로 실행 할 수 있는 기능 필요 (리커버리 모드)
  - 일반적으로는 전원 On시 부트로더 실행 후 펌웨어로 점프함.



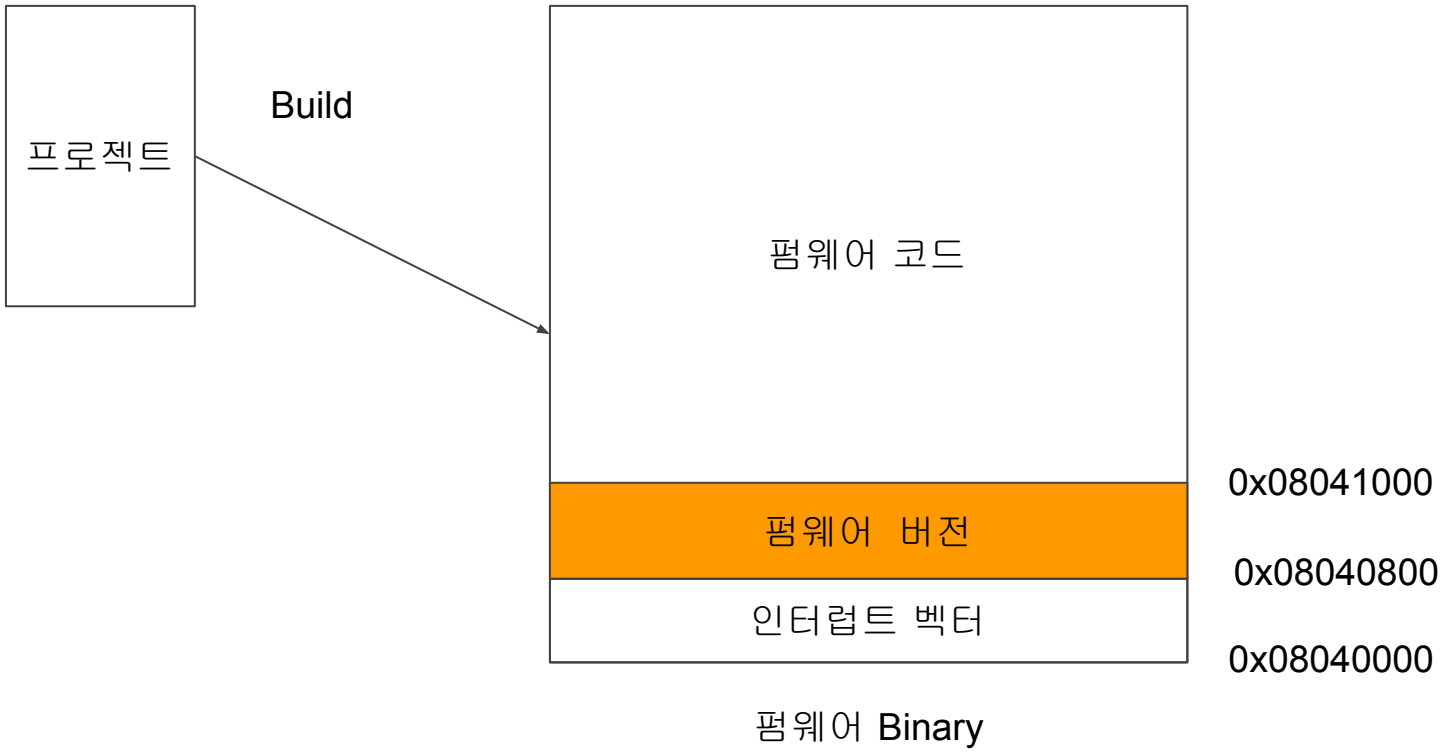
부트로더 실행

- 부팅 순서 정의



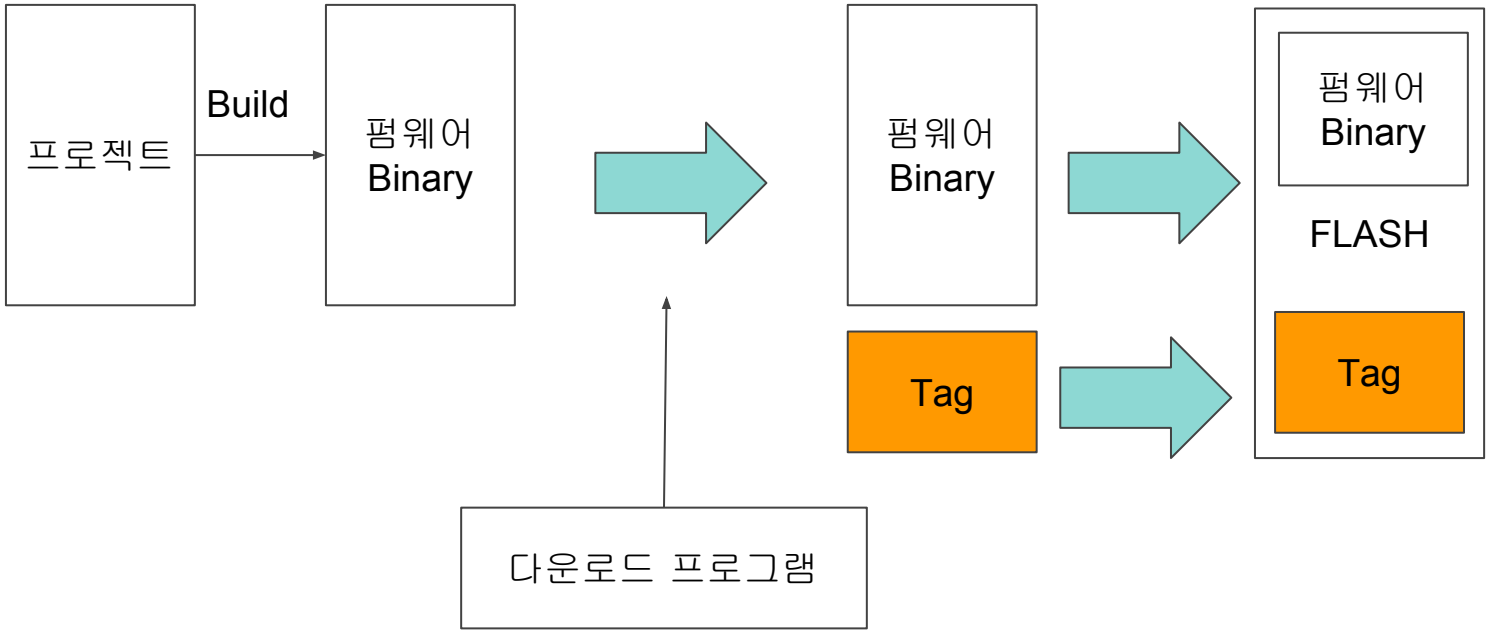
펌웨어 이미지 생성

- 부트로더에서 펌웨어 버전을 확인 할 수 있도록 특정 위치에 펌웨어 버전을 위치시킴



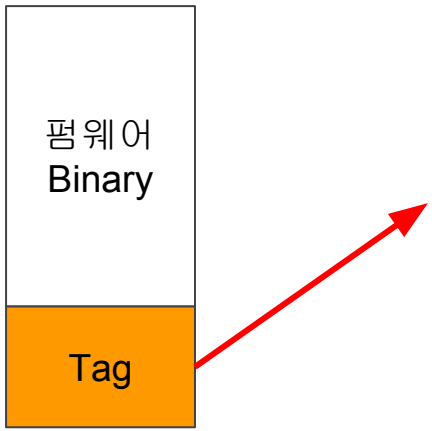
펌웨어 이미지 생성

- 프로젝트 빌드 후에 생성된 펌웨어 Binary에 대한 정보를 Tag정보로 Flash에 저장
  - Tag는 펌웨어 Binary의 유효성 검증용



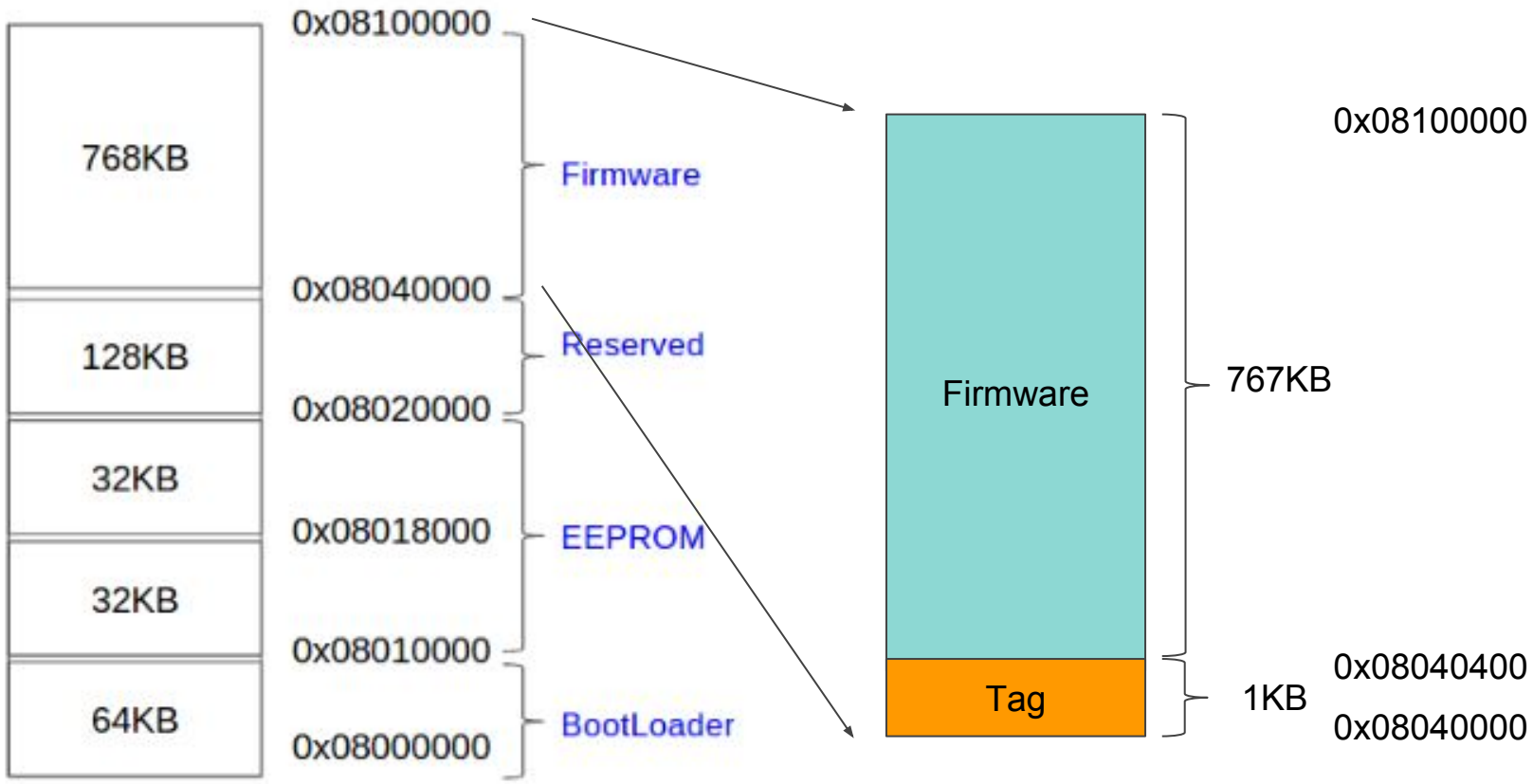
펌웨어 Tag 구성

- 펌웨어 Tag에는 펌웨어 유효성을 검증할 수 있는 데이터가 포함
  - 펌웨어 위치 및 크기와 CRC 혹은 체크섬 데이터를 이용하여 검증 함




```
typedef struct
{
  uint32_t tag;                          // 0x5555AAAA
  uint32_t date;                         // 0x20170410
  uint32_t firmware_offset;
  uint32_t firmware_offset_version;
  uint32_t firmware_length;
  uint32_t firmware_check_sum;
} firmware_header_t;
```

Tag를 포함한 Memory Map



메모리 정의

- hw_def.h 에 메모리 주소 정의
  - 메모리 주소를 정의하여 변경하기 쉽도록 적용

hw_def.h

```
...

#define FLASH_FW_SIZE                   (768*1024)  // 768KB
#define FLASH_FW_ADDR_START             0x08040000
#define FLASH_FW_ADDR_END               (FLASH_FW_ADDR_START + FLASH_FW_SIZE)

...
```

통신 프로토콜

- 시리얼 통신 프로토콜 정의 - 송신 패킷
  - PC에서 MCU가는 패킷으로 명령을 전송함

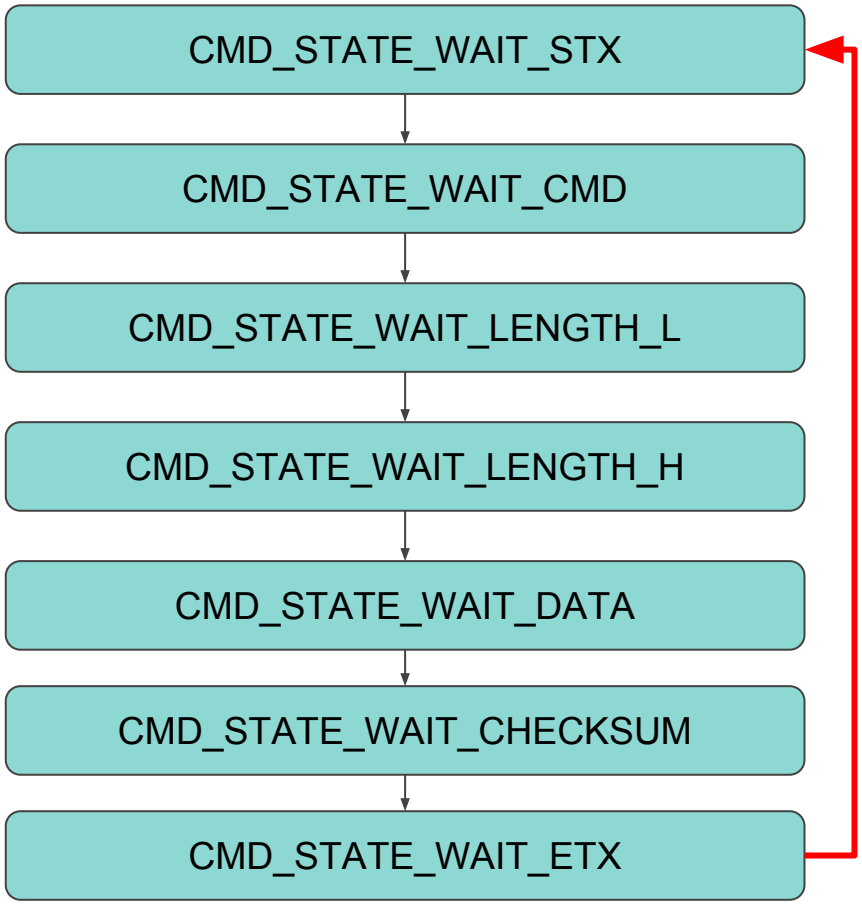
STX	Cmd	Option	LENGTH_L	LENGTH_H	Data[0]	...	Data[n]	CheckSum	ETX
0x02	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	CheckSum	0x03

- 시리얼 통신 프로토콜 정의 - 수신 패킷
  - MCU에서 PC로 보내는 패킷으로 송신 패킷을 수신시 그에 대한 응답으로 보냄

STX	Cmd	Error	LENGTH_L	LENGTH_H	Data[0]	...	Data[n]	CheckSum	ETX
0x02	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	0x00 ~ 0xFF	CheckSum	0x03

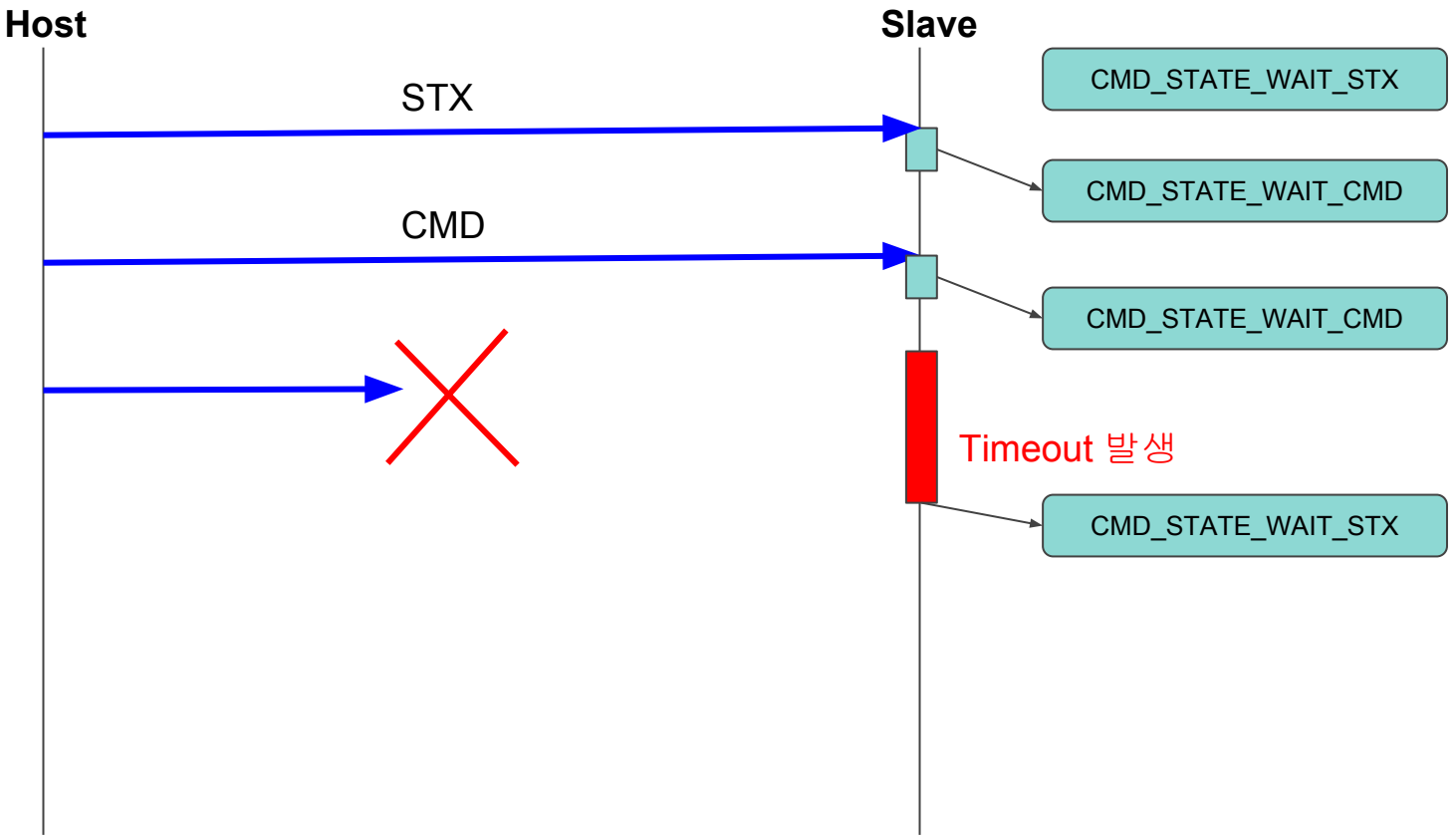
프로토콜 데이터 수신 처리

- 수신되는 데이터를 1바이트씩 받아서 처리함.
  - 상태머신을 이용하여 각 상태를 정의하고 패킷을 확인함.



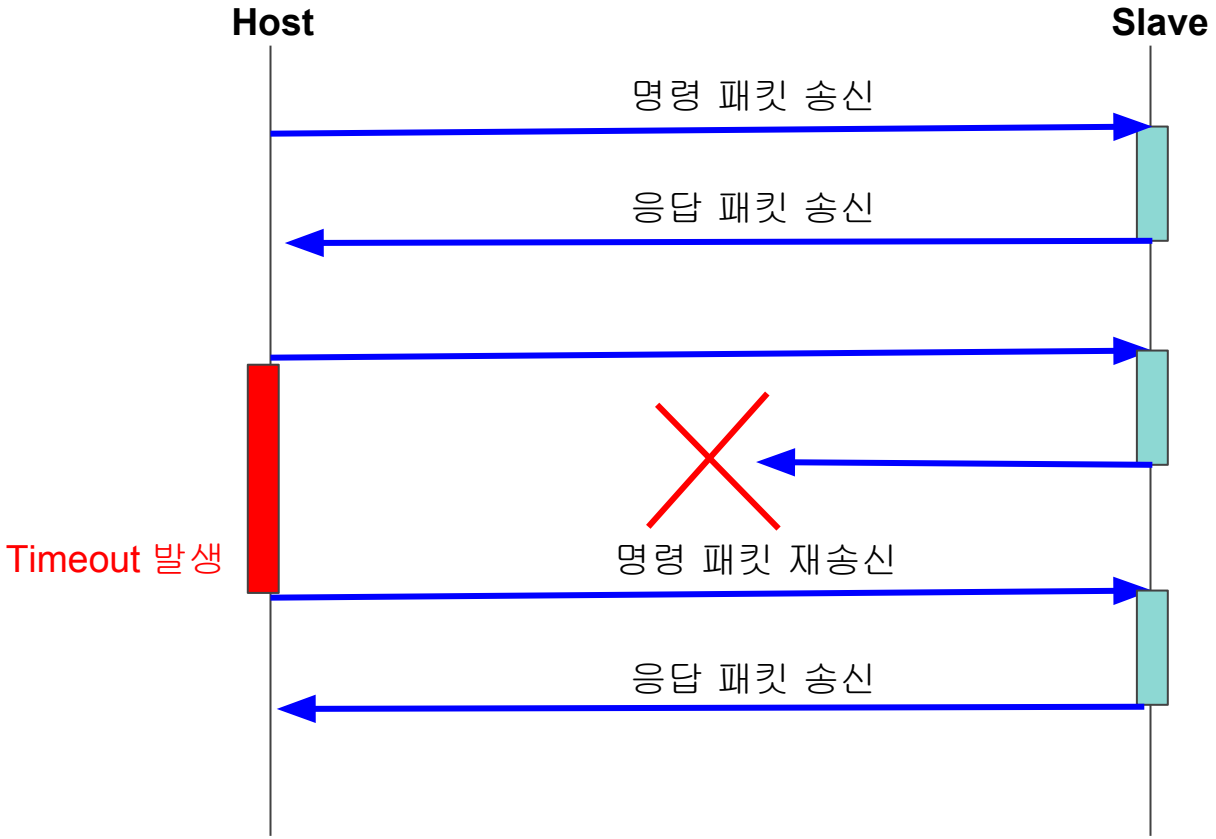
1바이트 송/수신 예외 처리

- Host->Slave로 전송하는 바이트 사이의 타임아웃 발생 처리



패킷 송/수신 타임아웃

- Host->Slave로 전송하는 명령어 패킷에 대한 응답 타임아웃 발생 처리



부트로더 명령어 구성

- 부트로더를 위한 명령어
  - BOOT_CMD_READ_VERSION
    - 현재 버전을 읽는다.
  - BOOT_CMD_READ_BOARD_NAME
    - 보드 이름을 읽는다.
  - BOOT_CMD_FLASH_ERASE
    - FLASH에 원하는 영역을 지운다.
  - BOOT_CMD_FLASH_WRITE
    - FLASH에 원하는 데이터를 Write한다.
    - 한번에 Write할 수 있는 양은 통신 버퍼 크기로 제한된다.
  - BOOT_CMD_JUMP_TO_FW
    - 펌웨어로 점프한다.

```
#define BOOT_CMD_READ_VERSION          0x00
#define BOOT_CMD_READ_BOARD_NAME       0x01
#define BOOT_CMD_FLASH_ERASE           0x02
#define BOOT_CMD_FLASH_WRITE           0x03
#define BOOT_CMD_JUMP_TO_FW            0x08
```

명령어 처리 순서

- cmdReceivePacket함수에서 통신 프로토콜 분석 후 정상 패킷 수신
- 수신된 패킷의 명령어에 따라 해당 함수를 수행함

cmdReceivePacket()

bootProcessCmd()

```
void bootProcessCmd(cmd_t *p_cmd)
{
  switch(p_cmd->rx_packet.cmd)
  {
    case BOOT_CMD_READ_VERSION:
      bootCmdReadVersion(p_cmd);
      break;

    case BOOT_CMD_READ_BOARD_NAME:
      bootCmdReadBoardName(p_cmd);
      break;

    case BOOT_CMD_FLASH_ERASE:
      bootCmdFlashErase(p_cmd);
      break;

    case BOOT_CMD_FLASH_WRITE:
      bootCmdFlashWrite(p_cmd);
      break;
```

명령어 처리 순서

```
void apInit(void)
{
  cmdInit(&cmd_boot);
  cmdBegin(&cmd_boot, _DEF_UART1, 115200);
}

void apMain(void)
{
  uint32_t pre_time;


  pre_time = millis();
  while(1)
  {
    if (cmdReceivePacket(&cmd_boot) == true)
    {
      bootProcessCmd(&cmd_boot);
    }

    if (millis()-pre_time >= 100)
    {
      pre_time = millis();
      ledToggle(_DEF_LED1);
    }
  }
}
```

펌웨어 다운로드 순서

- 명령어를 이용해서 PC용 Downloader 프로그램과 통신으로 다운로드 진행