

Booting with SD card & QSPI

20th March 2023

0. 디자인 설명

> SD 카드 또는 QSPI로 FPGA 부팅하기

- SD카드 포트와 QSPI를 활성화하여 SD 카드에 넣은 펌웨어를 통해 SDK 프로젝트를 스스로 실행하도록 하거나 SPI로 펌웨어를 직접 부팅하도록 함

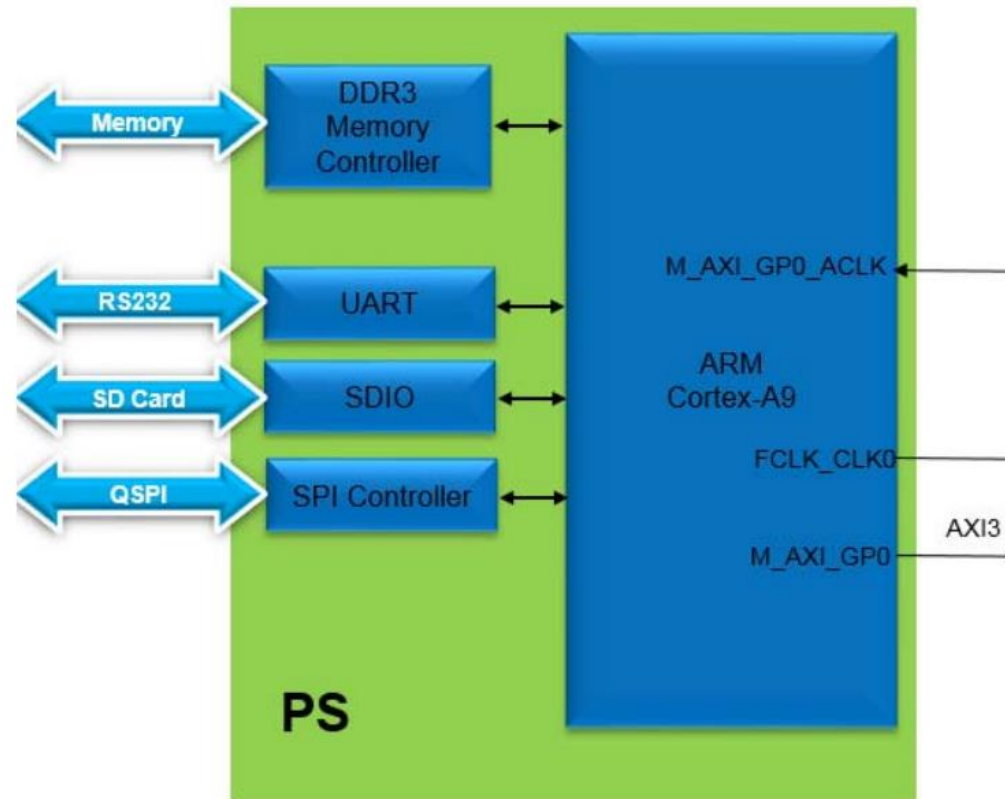
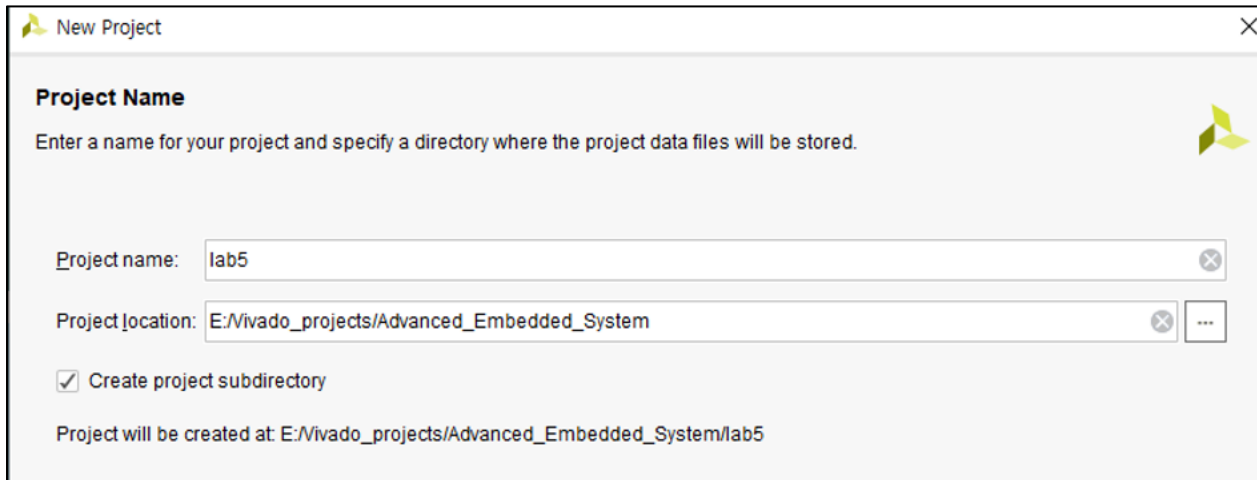


Figure 1. Completed Design

1. IP 설계

> Project 생성하기

- 새로운 **RTL Project** 생성 (project name: **lab5** / target board: **Zybo-Z7-20**)



New Project

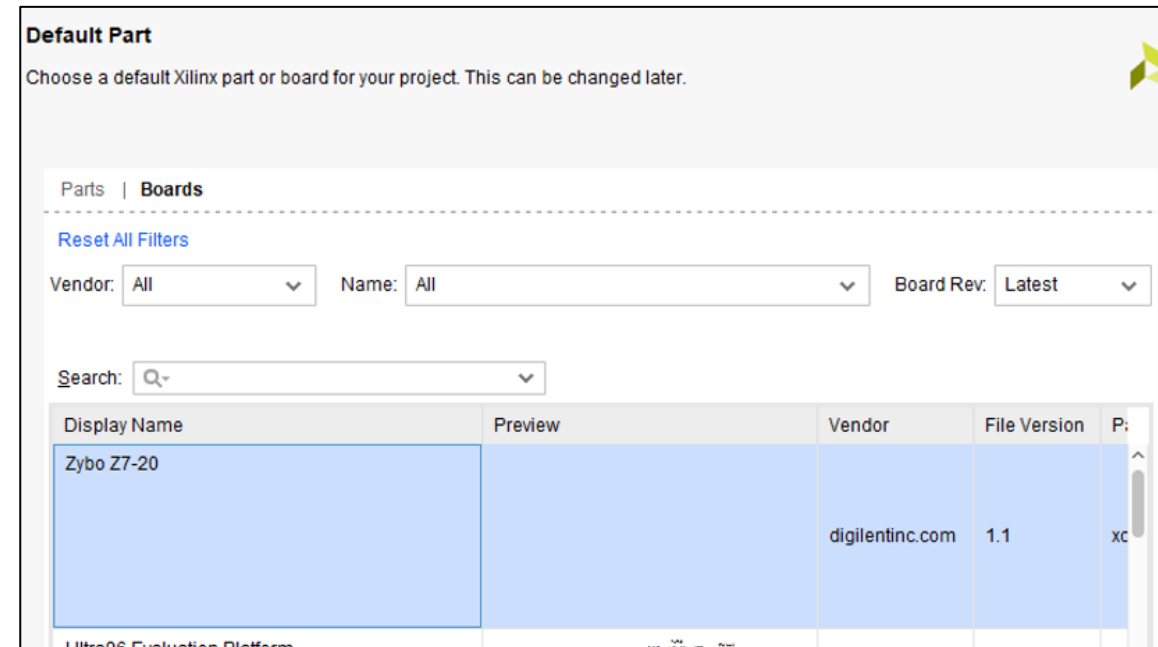
Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

Project name: lab5

Project location: E:/Vivado_projects/Advanced_Embedded_System

☒ Create project subdirectory

Project will be created at: E:/Vivado_projects/Advanced_Embedded_System/lab5



Default Part
Choose a default Xilinx part or board for your project. This can be changed later.

Parts | **Boards**

[Reset All Filters](#)

Vendor: All Name: All Board Rev: Latest

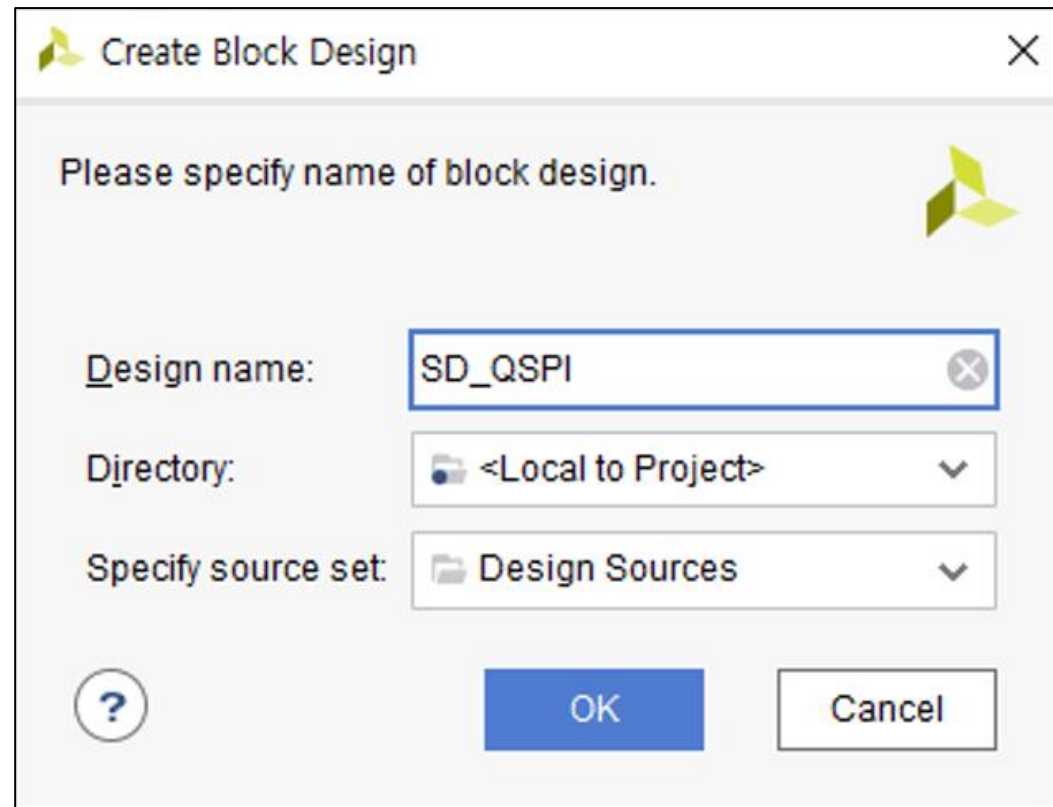
Search: Q

Display Name	Preview	Vendor	File Version	P
Zybo Z7-20		digilentinc.com	1.1	xc7z020
Ultra96 Evaluation Platform				

1. IP 설계

> Block Design 생성하기

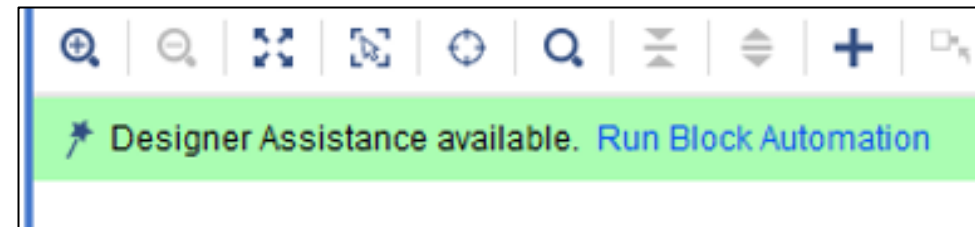
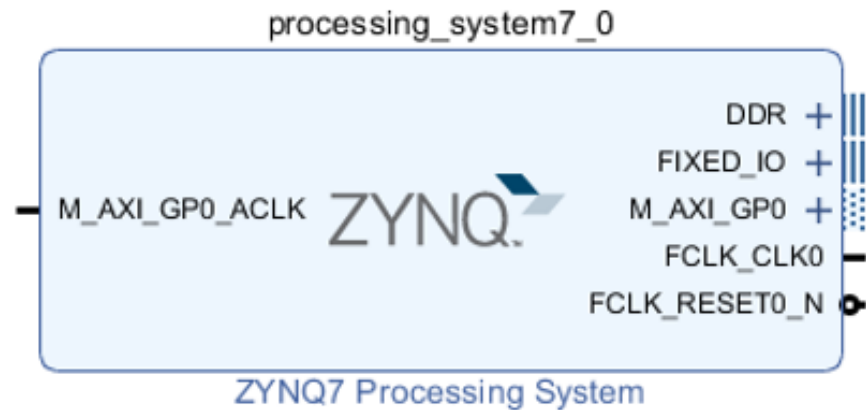
- 새로운 **Block Design** 생성하기 (이름은 아무거나)



1. IP 설계

> Block Design 설정

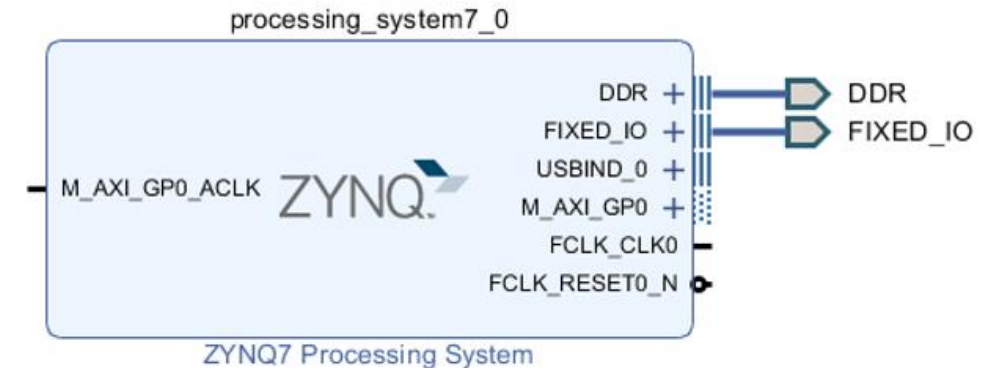
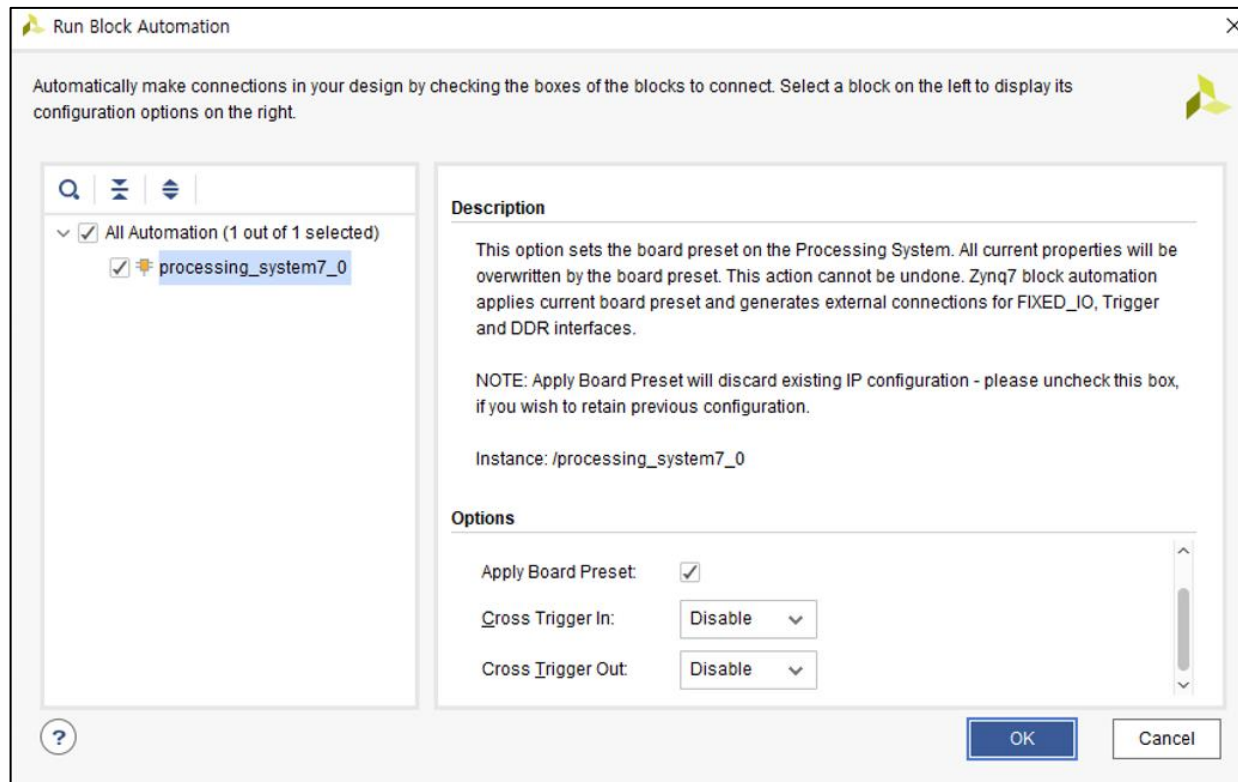
- ZYNQ7 Processing System 블록 불러오기 (Add IP)
- 불러온 후 Run Block Automation 실행



1. IP 설계

> Block Design 설정

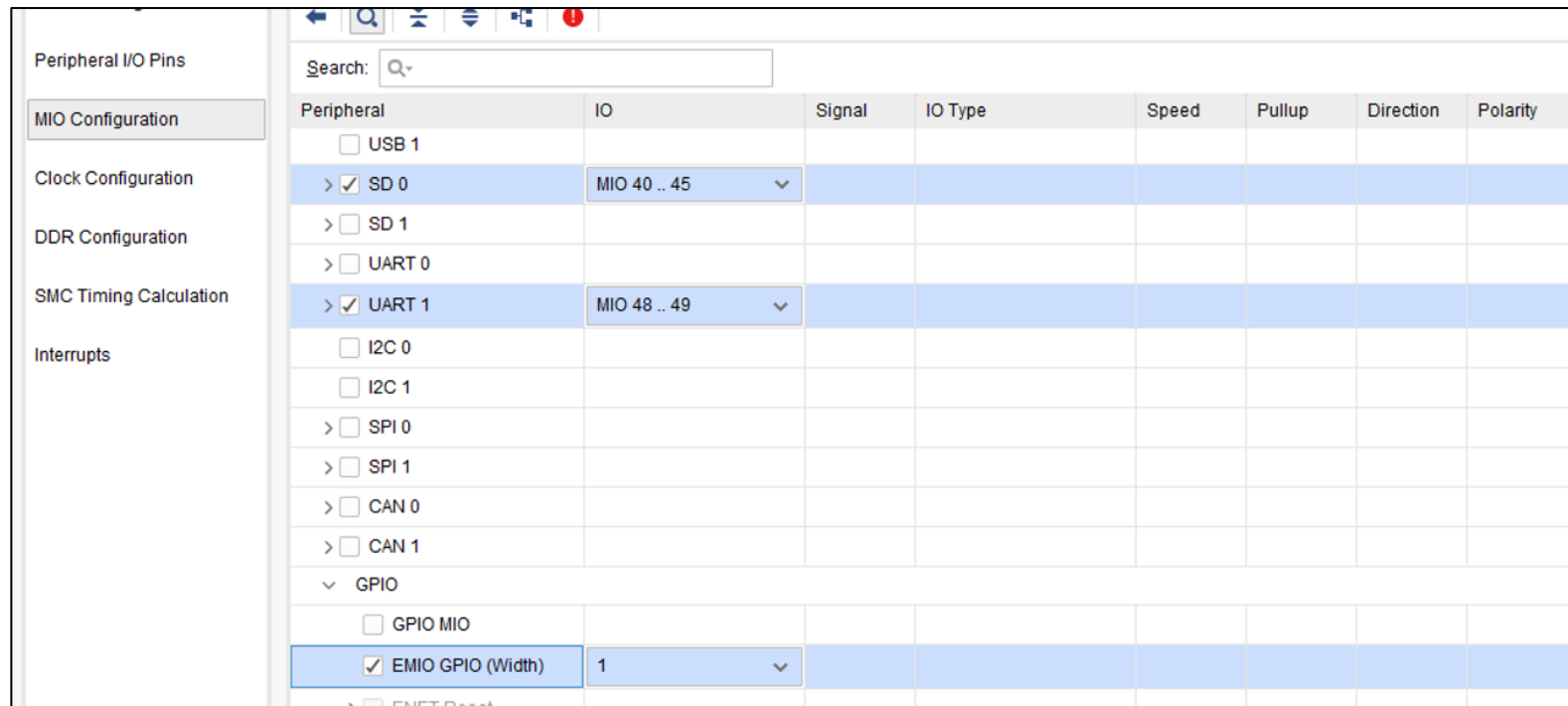
- 따로 설정 없이 그대로 **Run Block Automation** 실행
- 실행 하면 오른쪽 그림과 같이 됨



1. IP 설계

> Block Design 설정

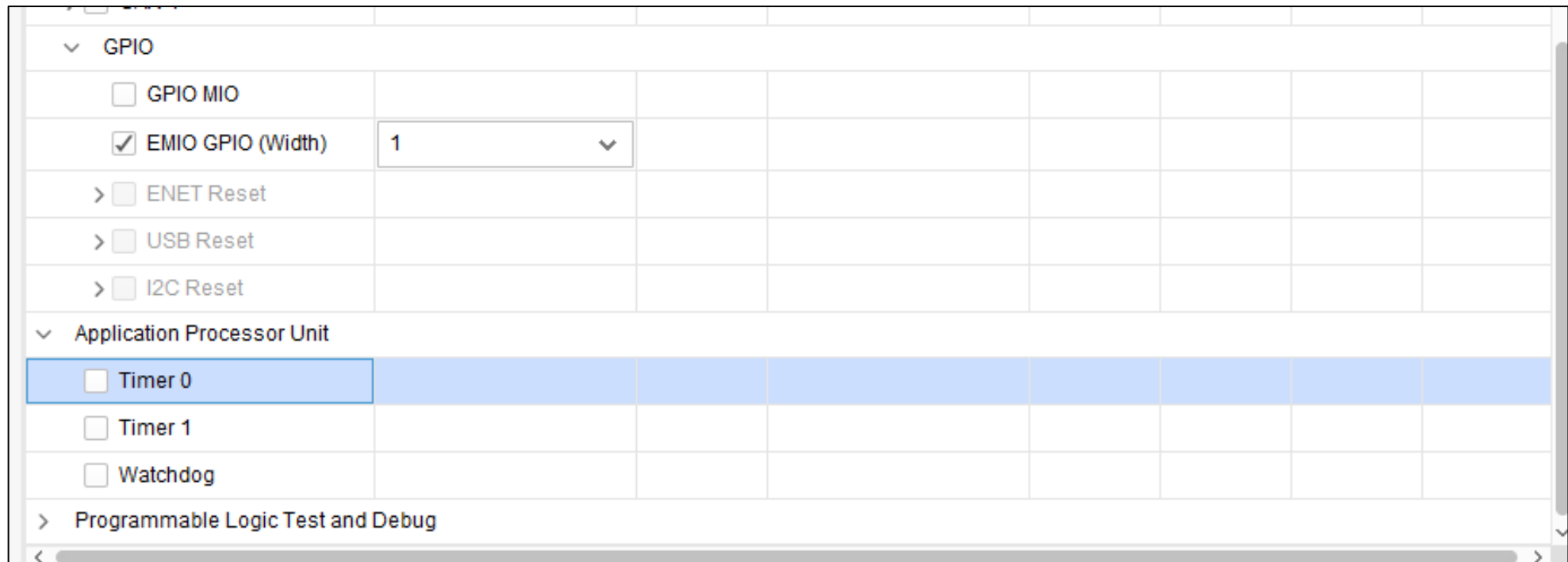
- **ZYNQ7 Processing System** 블록을 더블 클릭 > MIO Configuration 클릭
- I/O Peripherals에서 **SD 0, UART 1**만 남기고 나머지는 체크 해제
- GPIO > **EMIO GPIO**의 width를 1로 설정



1. IP 설계

> Block Design 설정

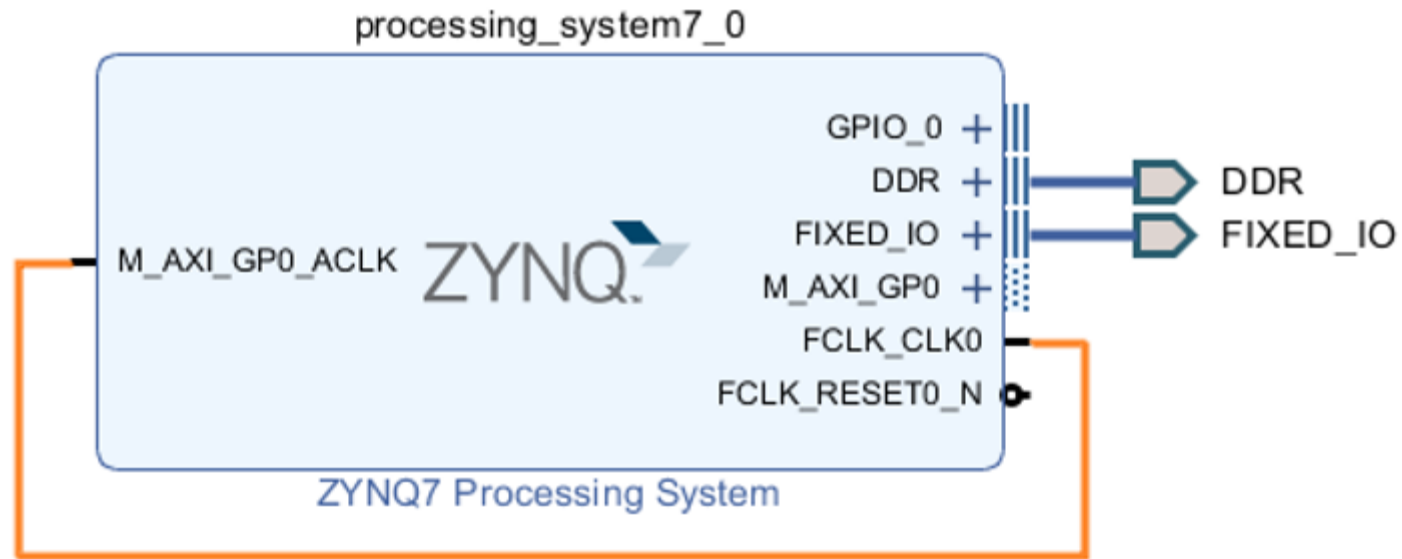
- Application Processor Unit에서 **Timer 0**를 체크 해제로 두기
- 이후 OK 클릭



1. IP 설계

> Block Design 설정

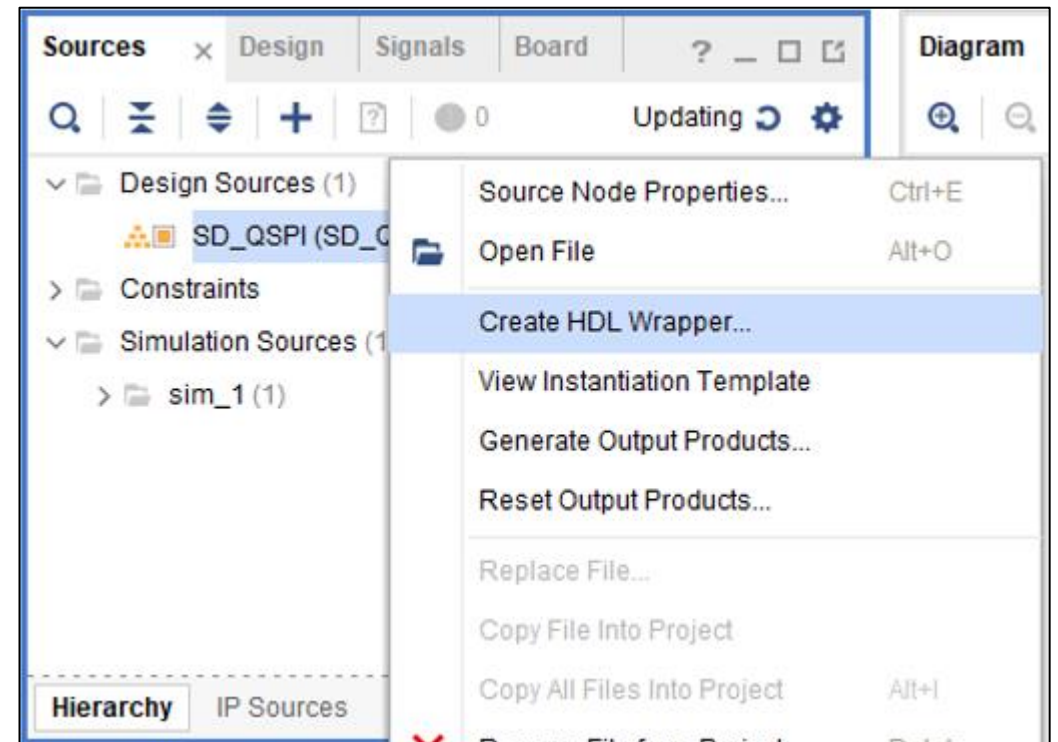
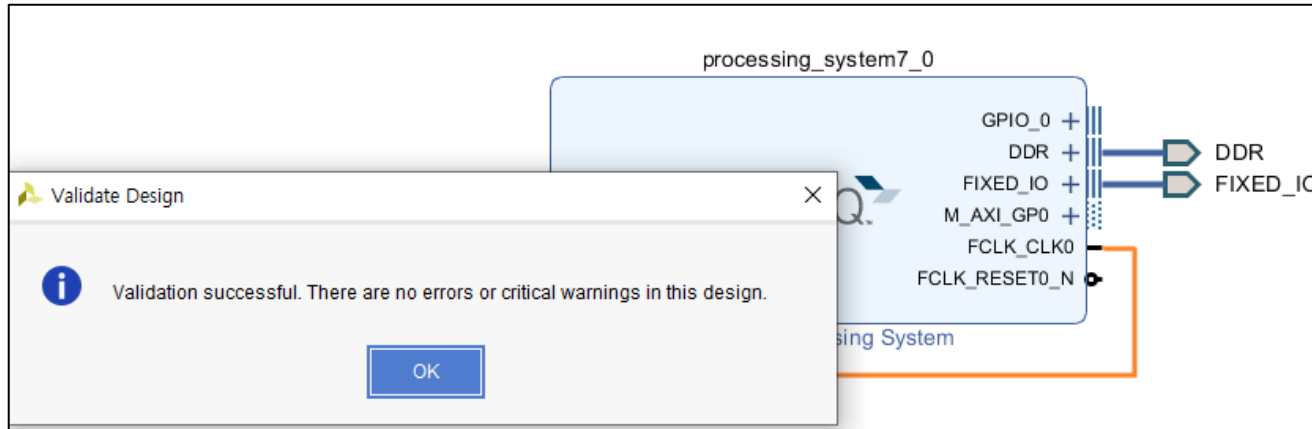
- M_AXI_GP0_ACLK와 FCLK_CLK0 사이를 연결



2. Bitstream 생성 및 SDK 실행

> Validation 확인 후 HDL Wrapper 생성

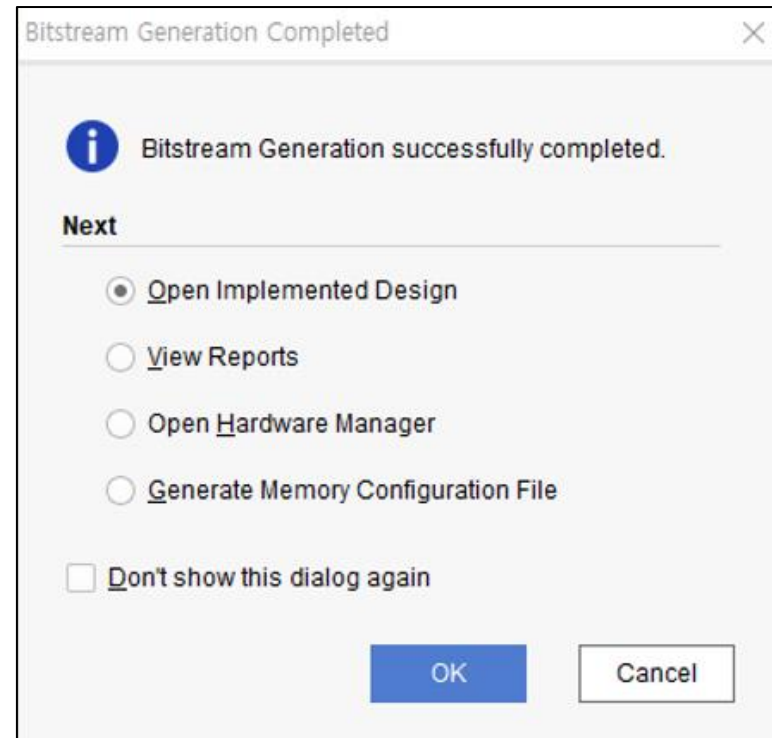
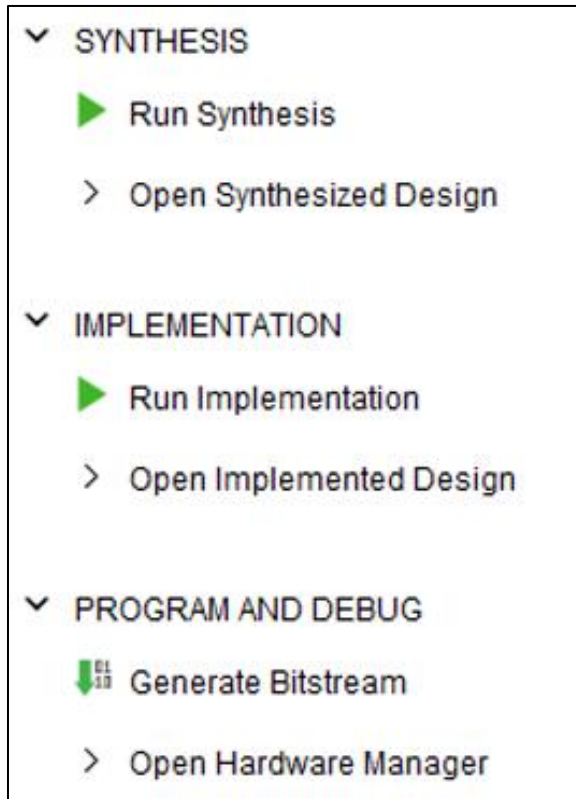
- **Validate Design**에서 문제 없으면 해당 디자인의 **HDL Wrapper** (Verilog file) 생성



2. Bitstream 생성 및 SDK 실행

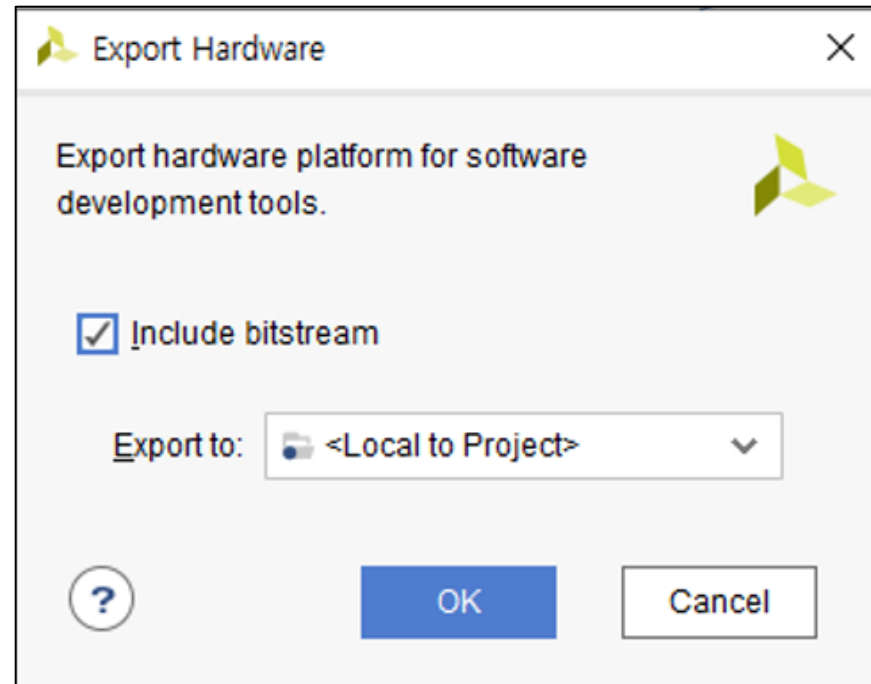
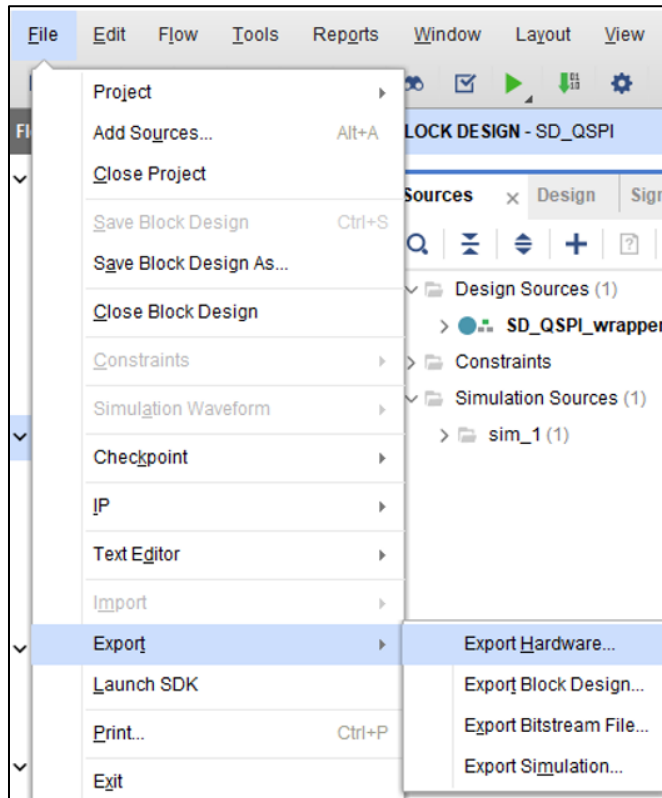
> Bitstream 생성

- **Generate Bitstream** 클릭 (Synthesis부터 자동으로 진행됨)



2. Bitstream 생성 및 SDK 실행

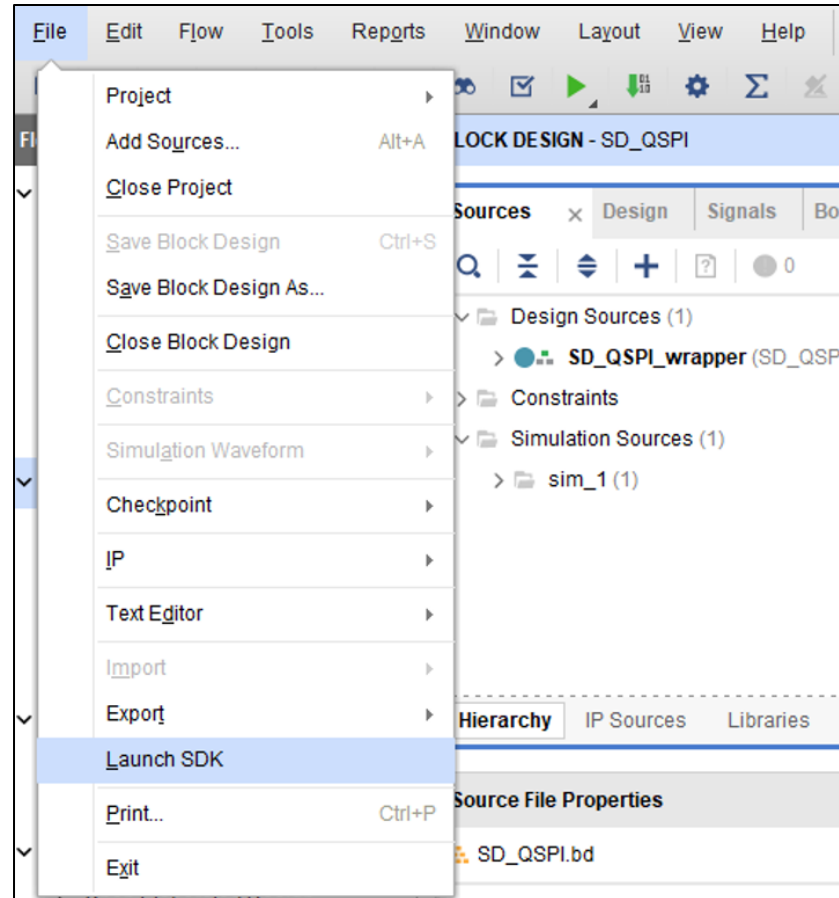
- > Hardware Export 후 SDK 실행
- File > Export > Export Hardware 클릭
- Include bitstream 반드시 체크



2. Bitstream 생성 및 SDK 실행

> Hardware Export 후 SDK 실행

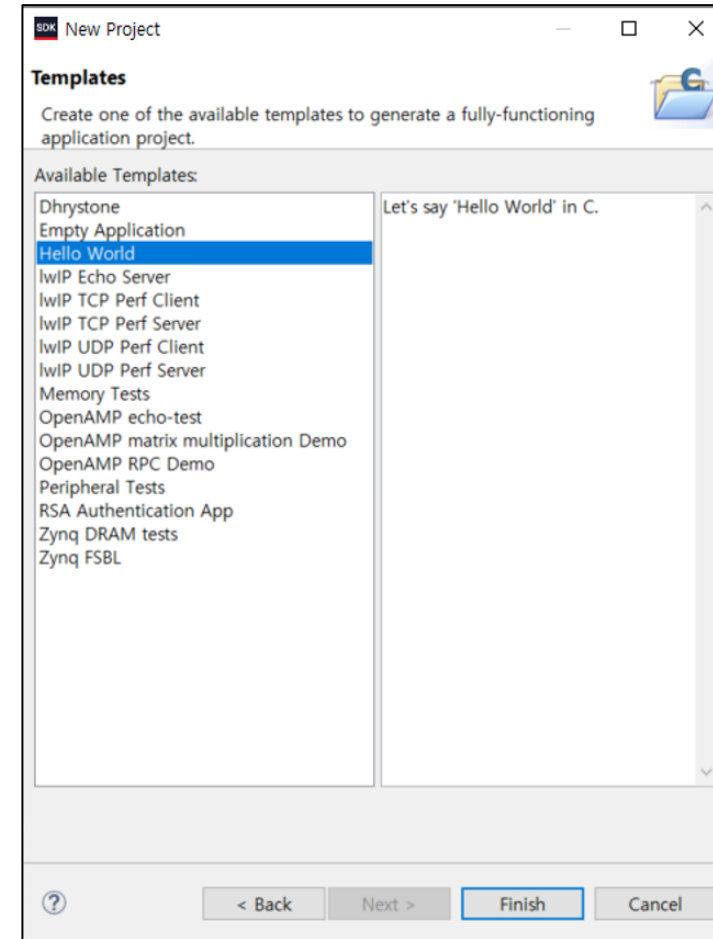
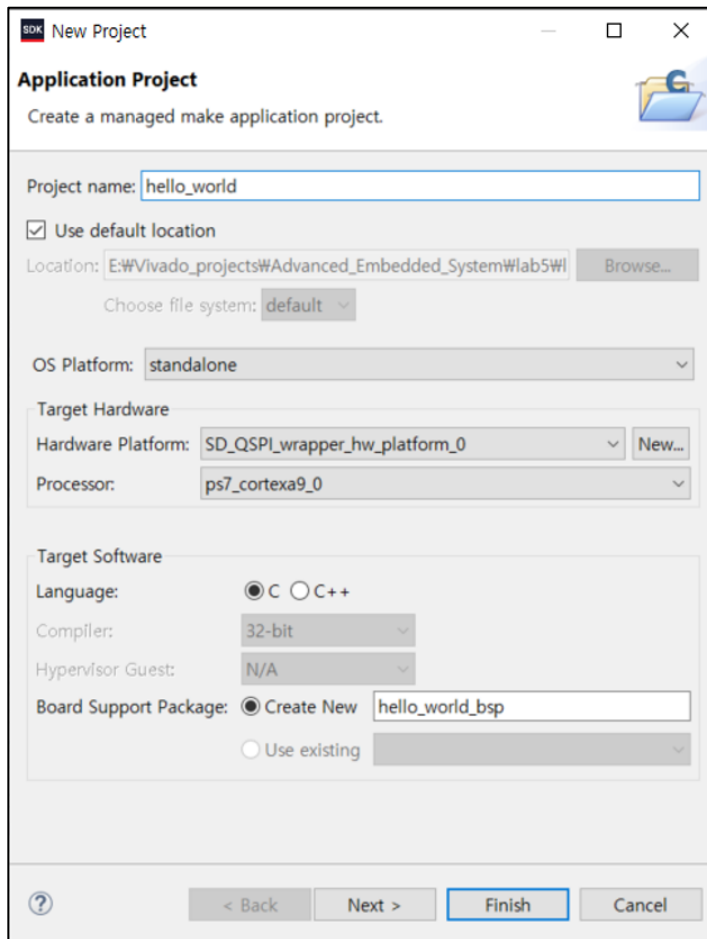
- File > Launch SDK 클릭 -> 자동으로 SDK 열림



3. Hello World 프로젝트 생성

> “hello_world” application project 생성 및 설정

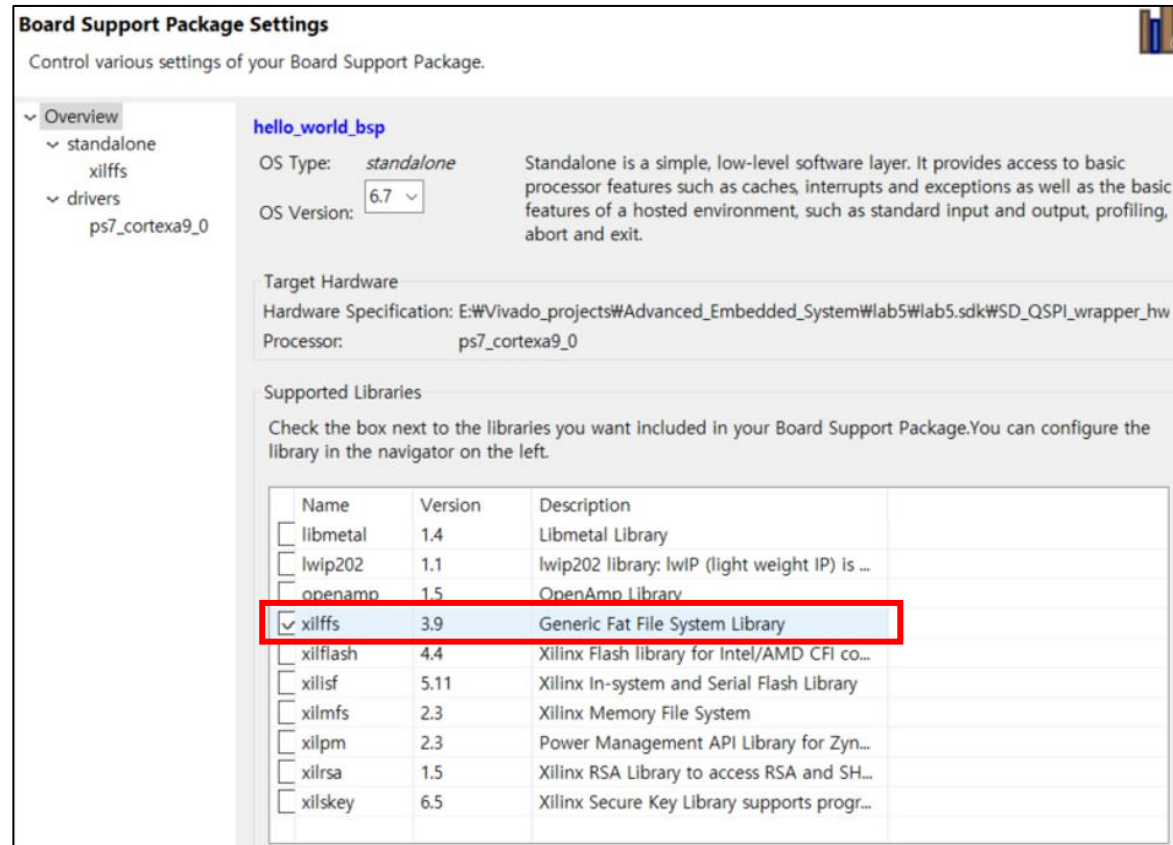
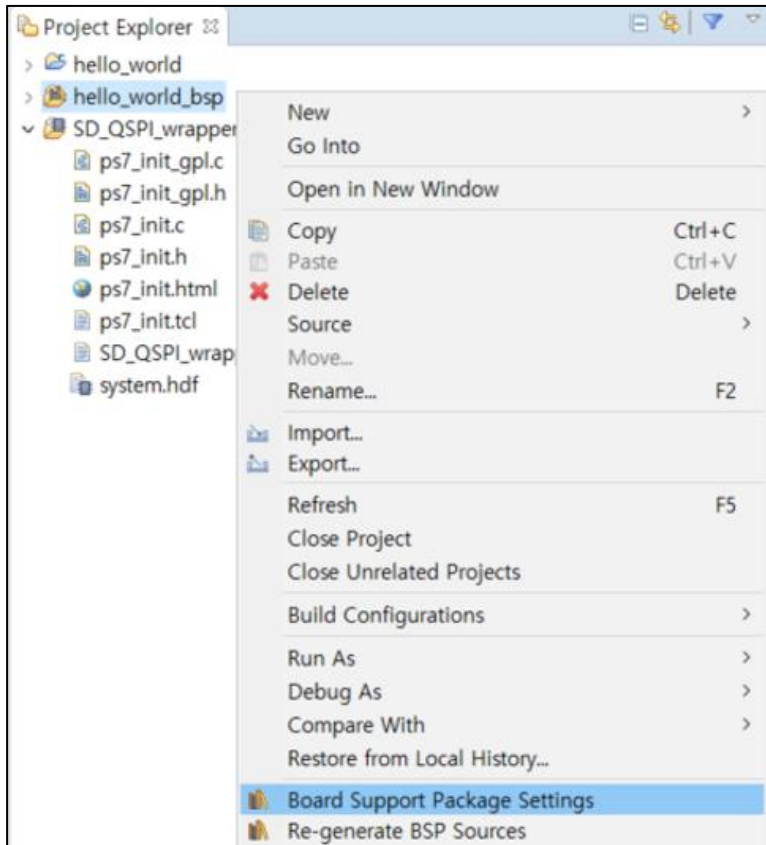
- “hello_world”라는 이름으로 project 생성 -> Next 클릭 -> **Hello World** 선택 -> Finish



3. Hello World 프로젝트 생성

> “hello_world” application project 생성 및 설정

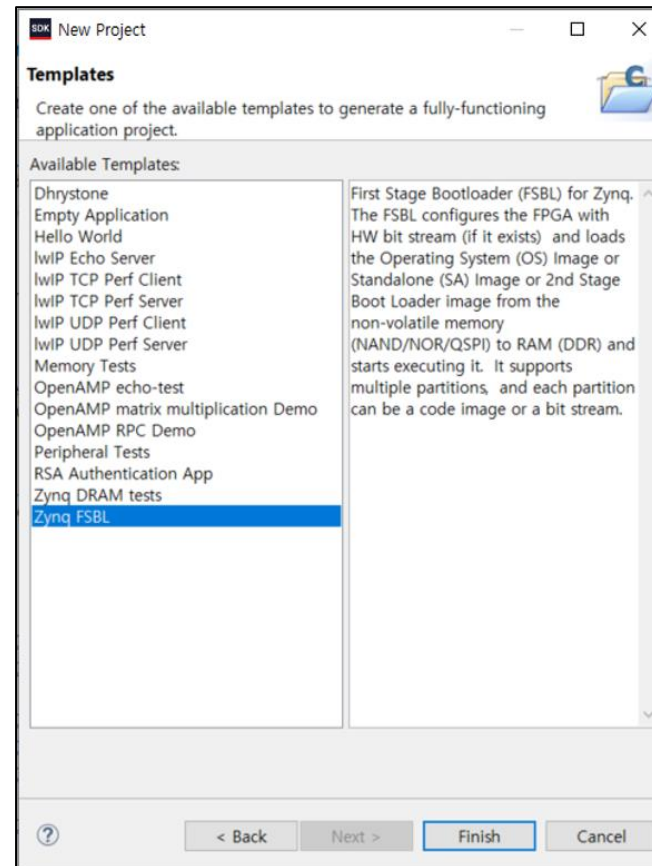
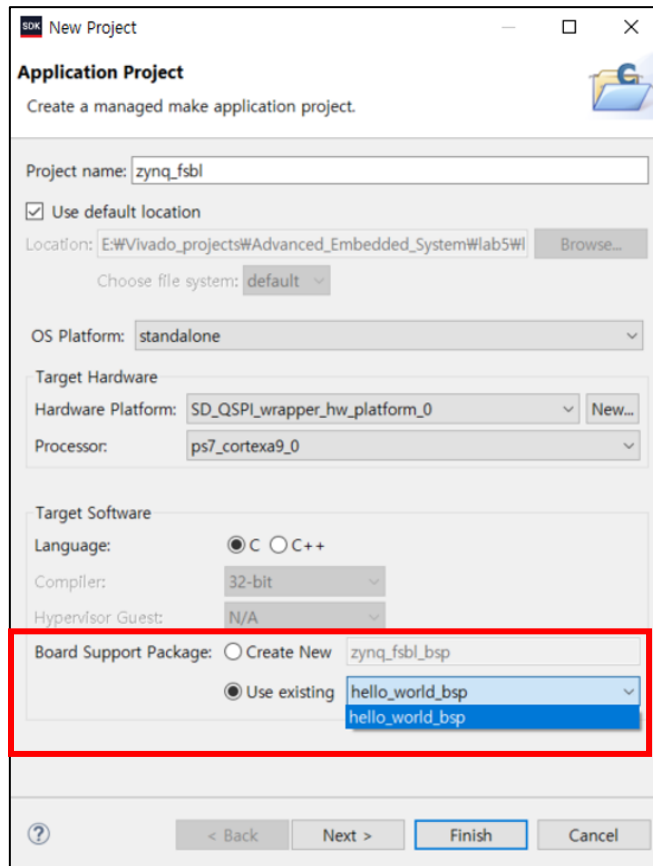
- 생성된 **hello_world_bsp**를 오른쪽 클릭 > **Board Support Package Settings**
- Supported Libraries에서 **xilffs** 선택 > OK 클릭



4. FSBL 프로젝트 생성

> Zynq FSBL (first stage bootloader) application project 생성 및 설정

- “zynq_fsbl”이라는 이름으로 project 생성 -> Board Support Package 항목에 **Use existing** > **hello_world_bsp** 선택 -> Next 클릭 -> **Zynq FSBL** 선택 -> Finish



5. Bootloader image 생성

> Image 저장 폴더 생성

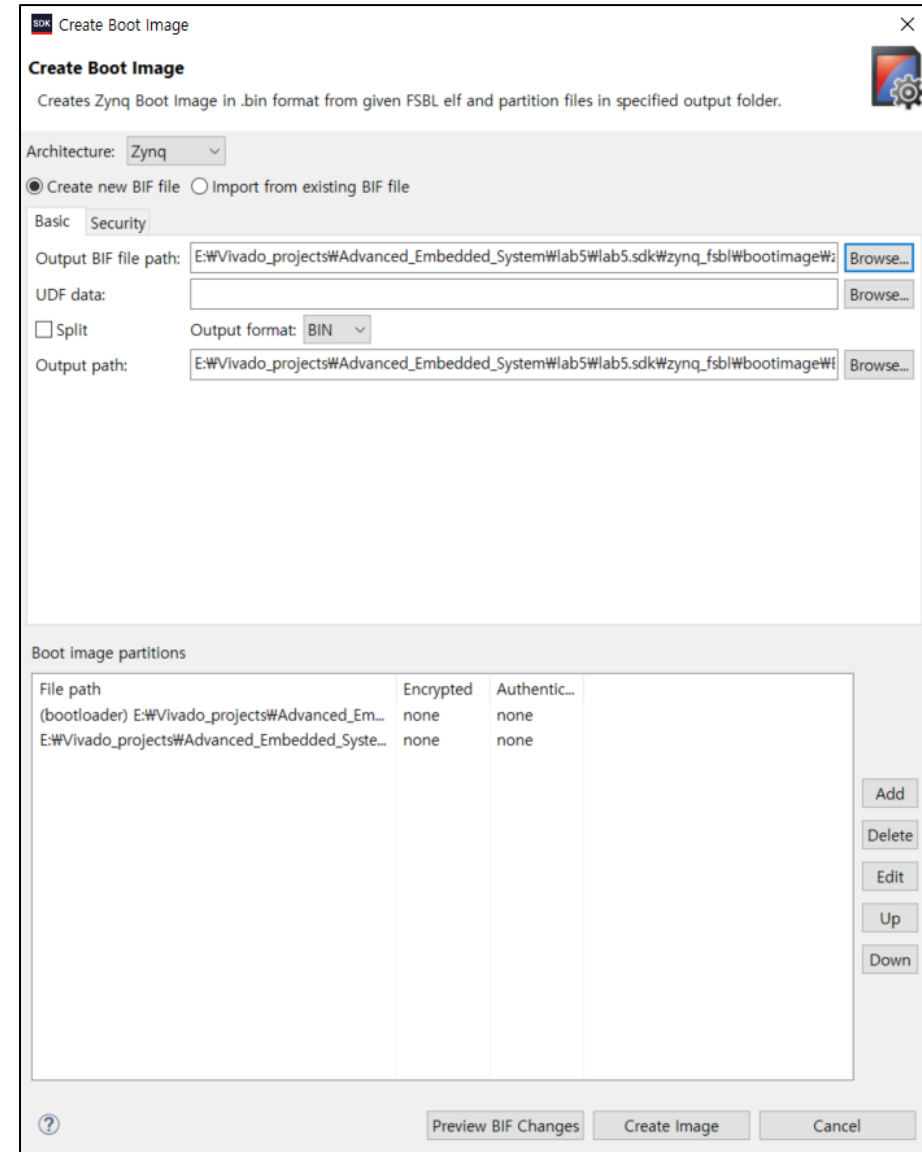
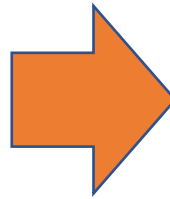
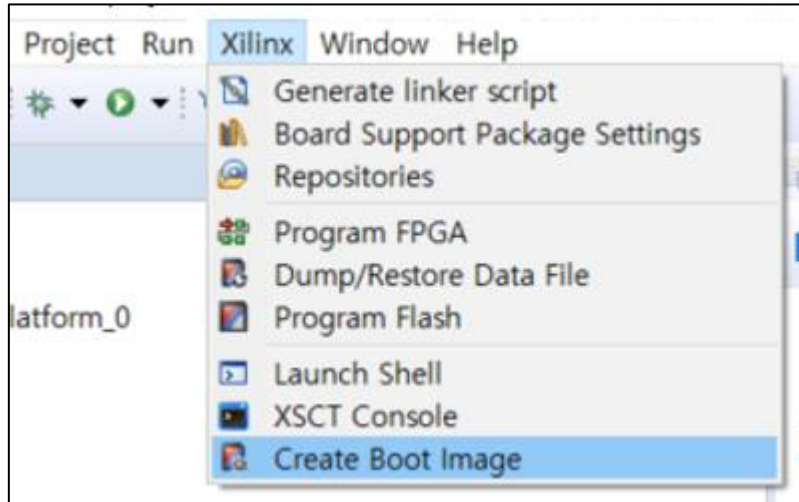
- Bootloader 파일을 저장할 폴더를 생성 (image)

이름	수정한 날짜	유형	크기
image	2023-03-02 오후 4:59	파일 폴더	
lab5.cache	만든 날짜: 2023-03-02 오후 4:59 폴더가 비어 있습니다.	파일 폴더	
lab5.hw		파일 폴더	
lab5.ip_user_files	2023-03-02 오후 3:37	파일 폴더	
lab5.runs	2023-03-02 오후 3:57	파일 폴더	
lab5.sdk	2023-03-02 오후 4:25	파일 폴더	
lab5.sim	2023-03-02 오후 3:37	파일 폴더	
lab5.srcs	2023-03-02 오후 3:39	파일 폴더	
lab5.xpr	2023-03-02 오후 3:59	Vivado Project File	10KB

5. Bootloader image 생성

> Boot Image 생성

- Xilinx > Create Boot Image 클릭



5. Bootloader image 생성

> Boot Image 생성

- (위쪽 영역)오른쪽의 Browse 버튼을 통해서 **Output BIF file path**와 **Output path**의 경로를 방금 생성한 image 폴더로 지정

Architecture: Zynq

☒ Create new BIF file ☐ Import from existing BIF file

Basic Security

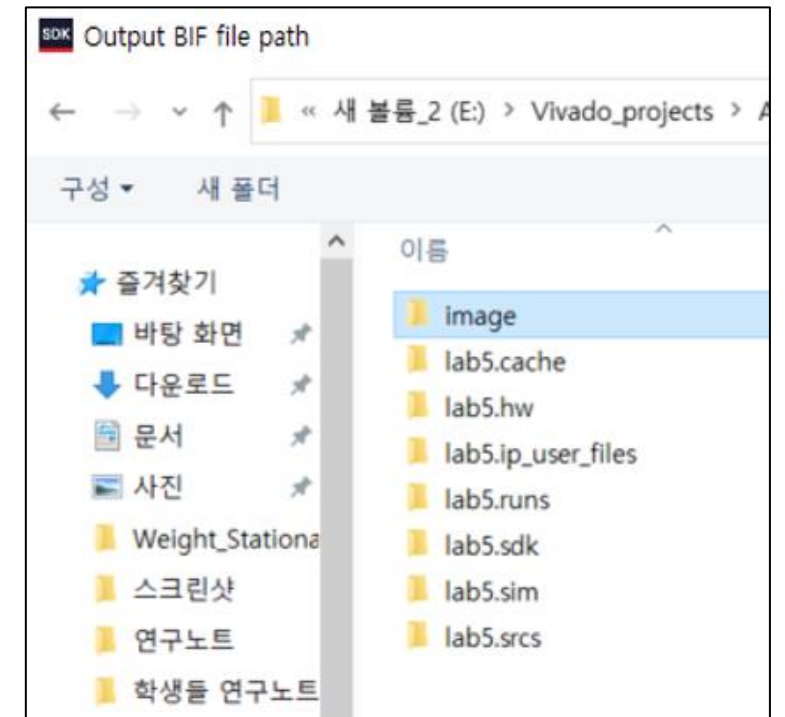
Output BIF file path: E:\Vivado_projects\Advanced_Embedded_System\lab5\image\zynq_fsbl.bif Browse...

UDF data: Browse...

☐ Split Output format: BIN

Output path: E:\Vivado_projects\Advanced_Embedded_System\lab5\image\BOOT.bin Browse...

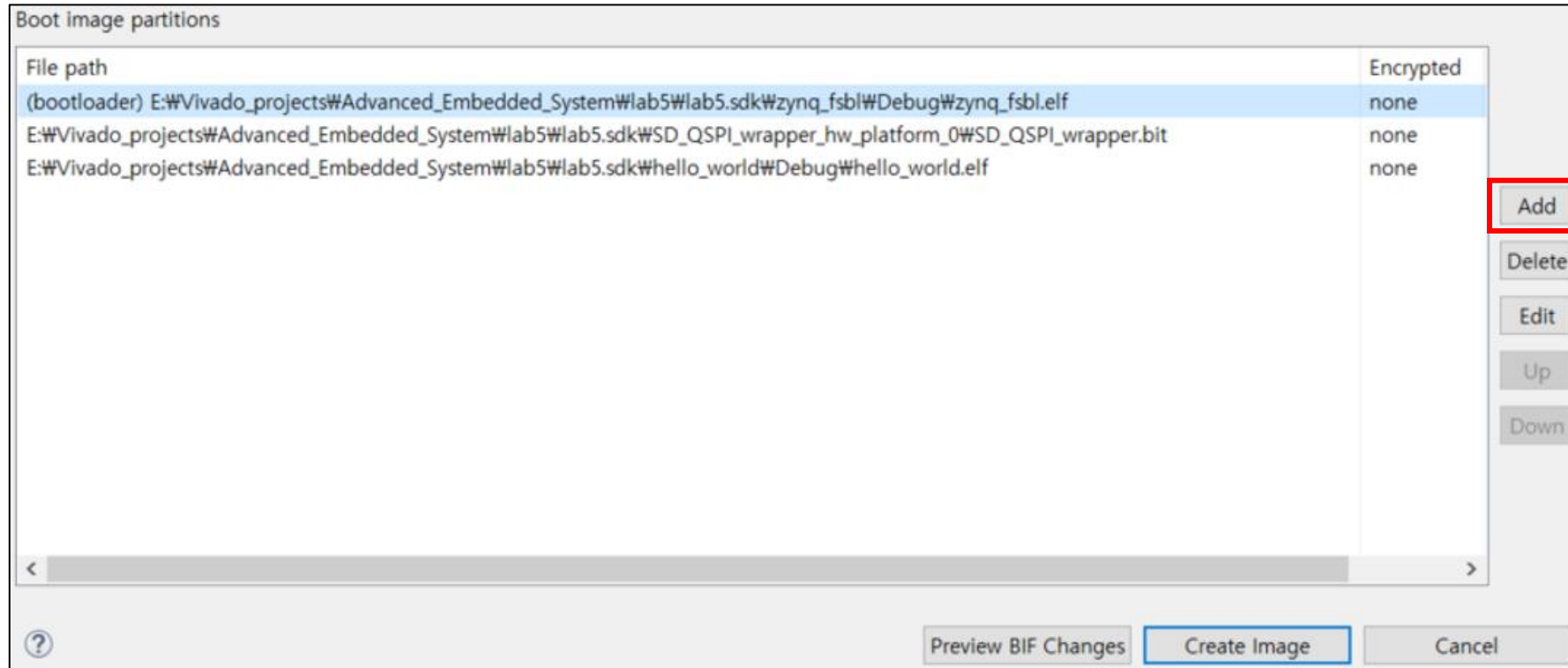
Boot image partitions



5. Bootloader image 생성

> Boot Image 생성

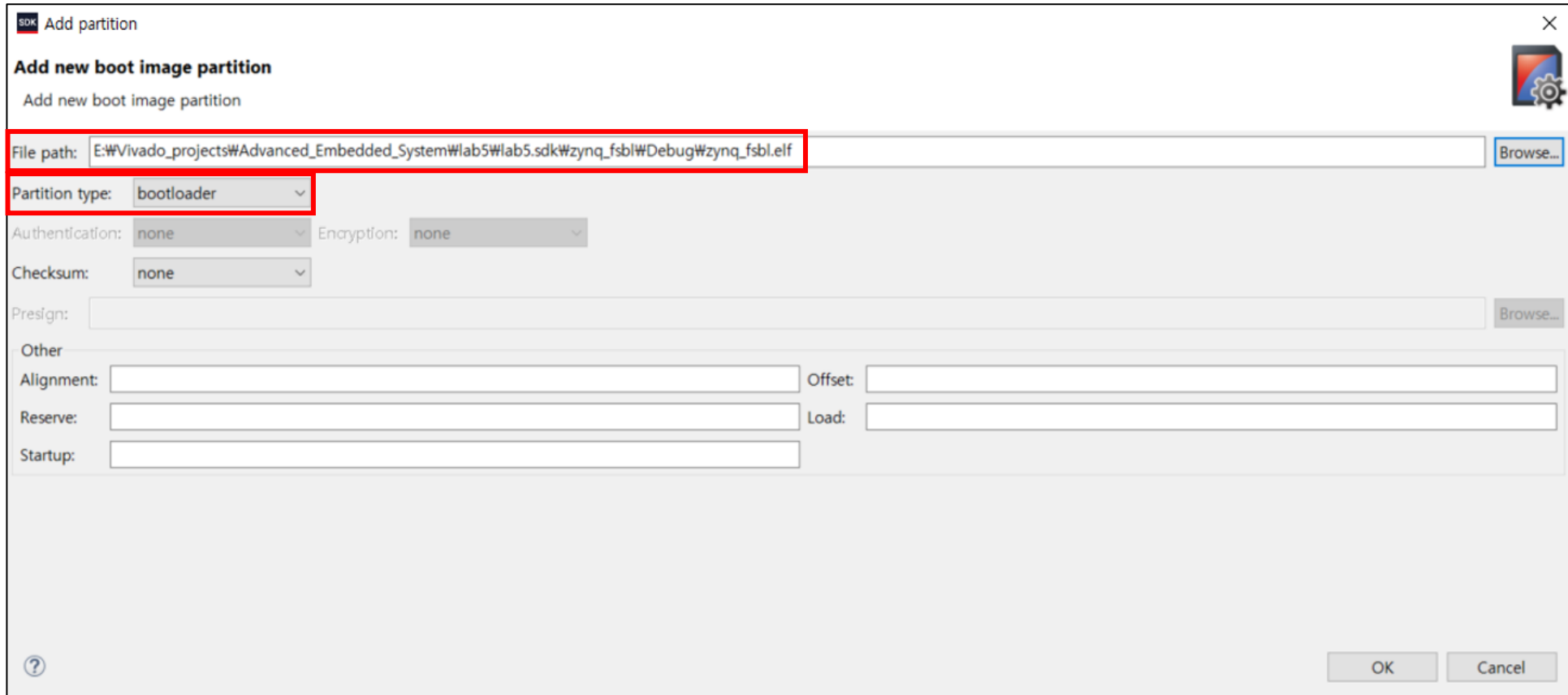
- (아래쪽 영역)오른쪽의 Add 버튼을 통해서 bootloader 파일과 datafile들을 등록 (bootloader 파일 1개, datafile 2개) (뒤에서 자세히 설명)



5. Bootloader image 생성

> Boot Image 생성

- **zynq_fsbl.elf** 파일을 다음 경로에서 찾아서 등록(lab5.sdk/zynq_fsbl/Debug 안에 있음)
- Partition type을 **bootloader**로 선택



SDK Add partition

Add new boot image partition

Add new boot image partition

File path: E:\Vivado_projects\Advanced_Embedded_System\lab5\lab5.sdk\zynq_fsbl\Debug\zynq_fsbl.elf Browse...

Partition type: bootloader

Authentication: none Encryption: none

Checksum: none

Presign: Browse...

Other

Alignment: Offset:

Reserve: Load:

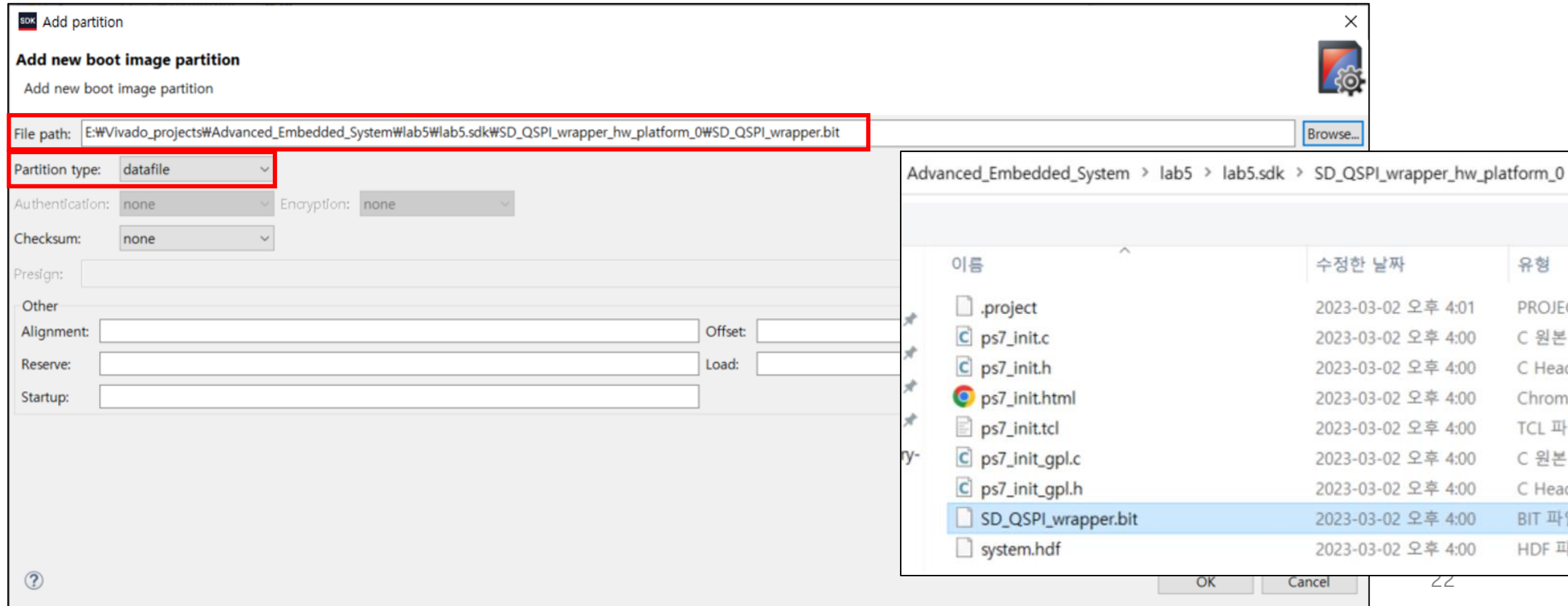
Startup:

OK Cancel

5. Bootloader image 생성

> Boot Image 생성

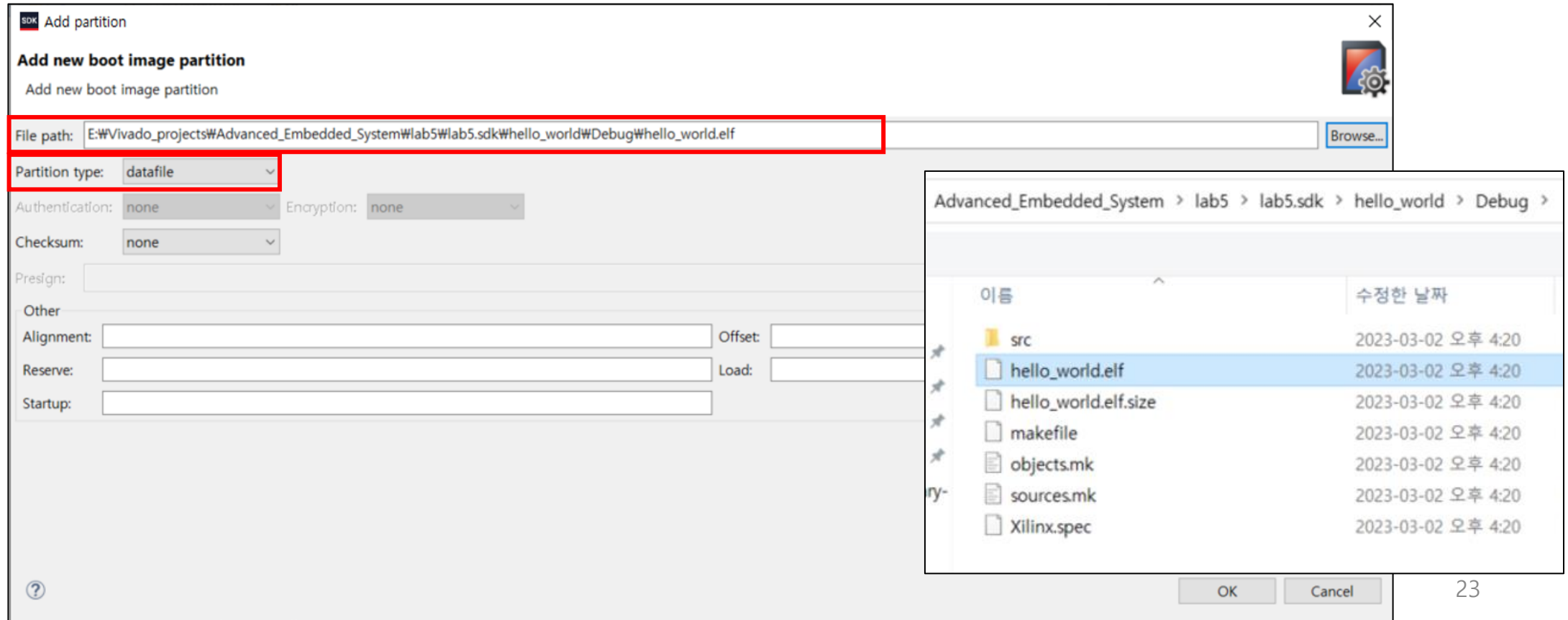
- **Bitstream** 파일을 다음 경로에서 찾아서 등록(lab5.sdk 이후 추가 경로는 그림 참조)
- Partition type을 **datafile**로 선택



5. Bootloader image 생성

> Boot Image 생성

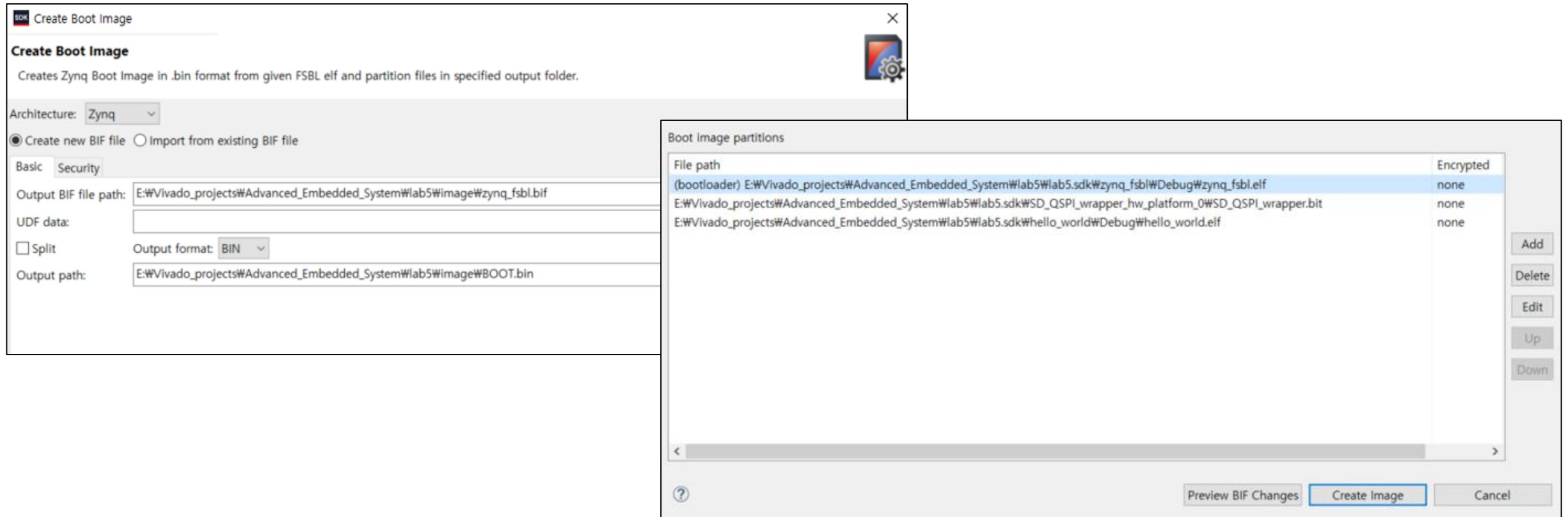
- **hello_world.elf** 파일을 다음 경로에서 찾아서 등록(lab5.sdk 이후 추가 경로는 그림 참조)
- Partition type을 **datafile**로 선택



5. Bootloader image 생성

> Boot Image 생성

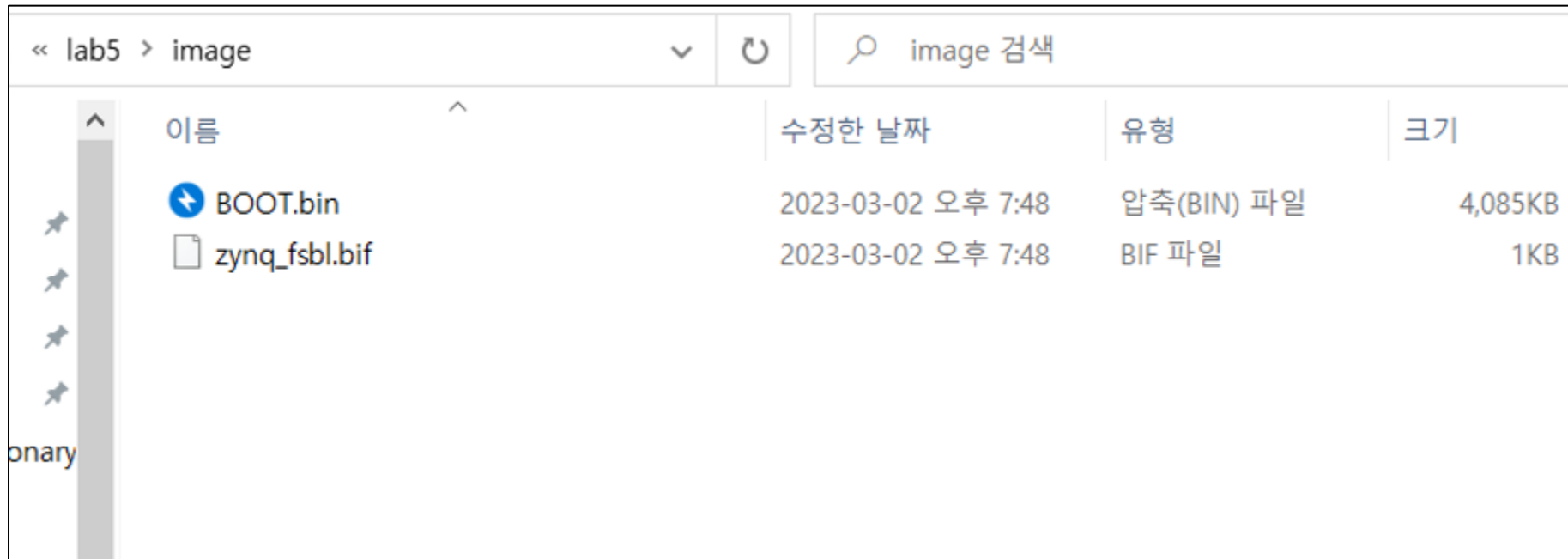
- 설정이 완료된 모습 (*File path에 저 3가지 외에 다른 것이 있어선 안됨)
- 제대로 설정했을 경우 아래에 **Create Image** 버튼이 활성화됨 -> 클릭





5. Bootloader image 생성

> Boot Image 생성

- Image 폴더로 가보면 다음과 같이 파일들이 만들어져 있음



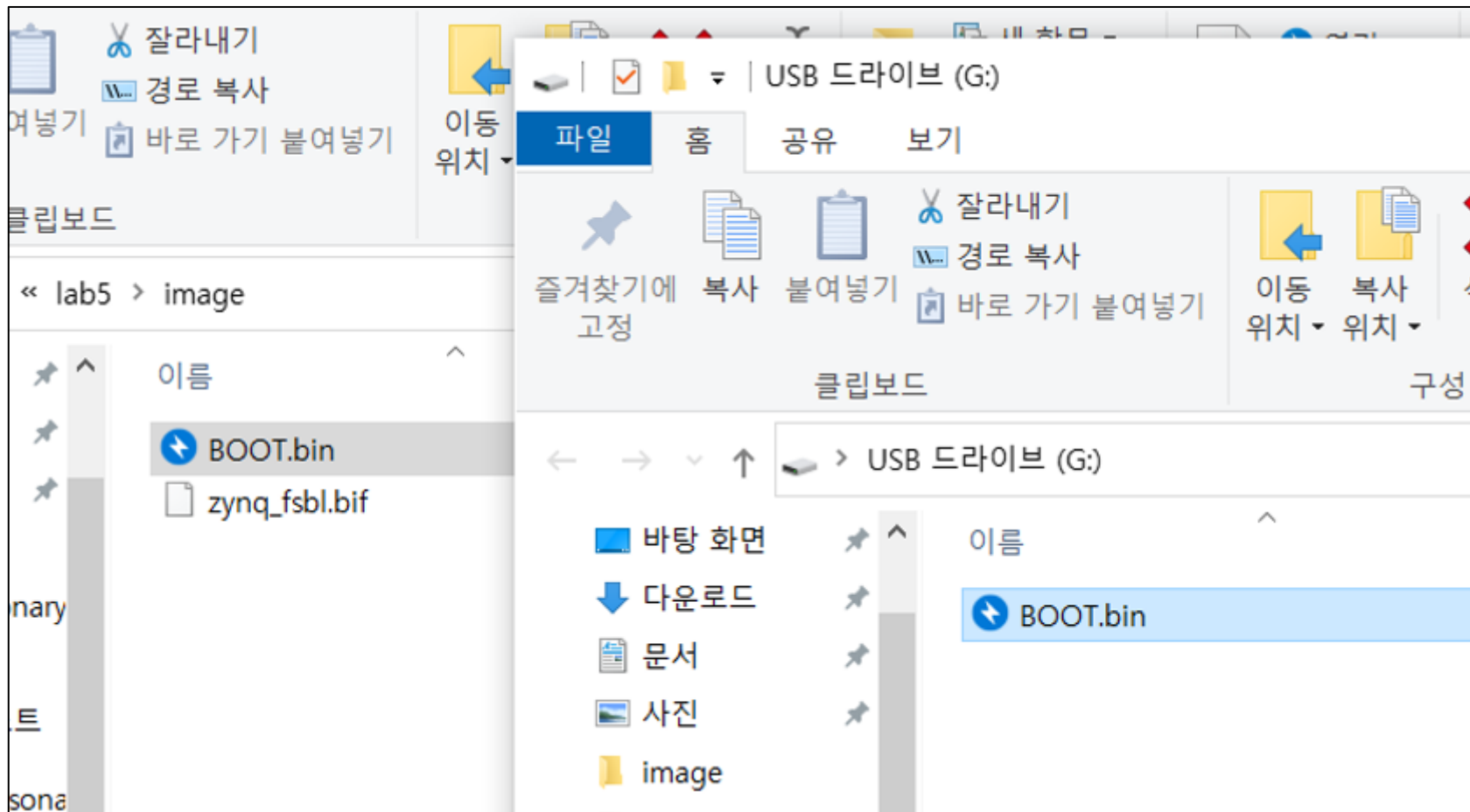
The screenshot shows a Windows File Explorer window with the address bar set to '<< lab5 > image'. The search bar contains 'image 검색'. The file list is displayed in a table format with columns for '이름' (Name), '수정한 날짜' (Date modified), '유형' (Type), and '크기' (Size). Two files are listed: 'BOOT.bin' and 'zynq_fsbl.bif'. 'BOOT.bin' is a compressed (BIN) file of 4,085KB, and 'zynq_fsbl.bif' is a BIF file of 1KB. Both files were last modified on 2023-03-02 at 7:48 PM. On the left sidebar, the 'image' folder is selected, and a portion of the 'dictionary' folder is visible below it.

이름	수정한 날짜	유형	크기
 BOOT.bin	2023-03-02 오후 7:48	압축(BIN) 파일	4,085KB
 zynq_fsbl.bif	2023-03-02 오후 7:48	BIF 파일	1KB

6. Bootloader image를 SD카드에 넣기

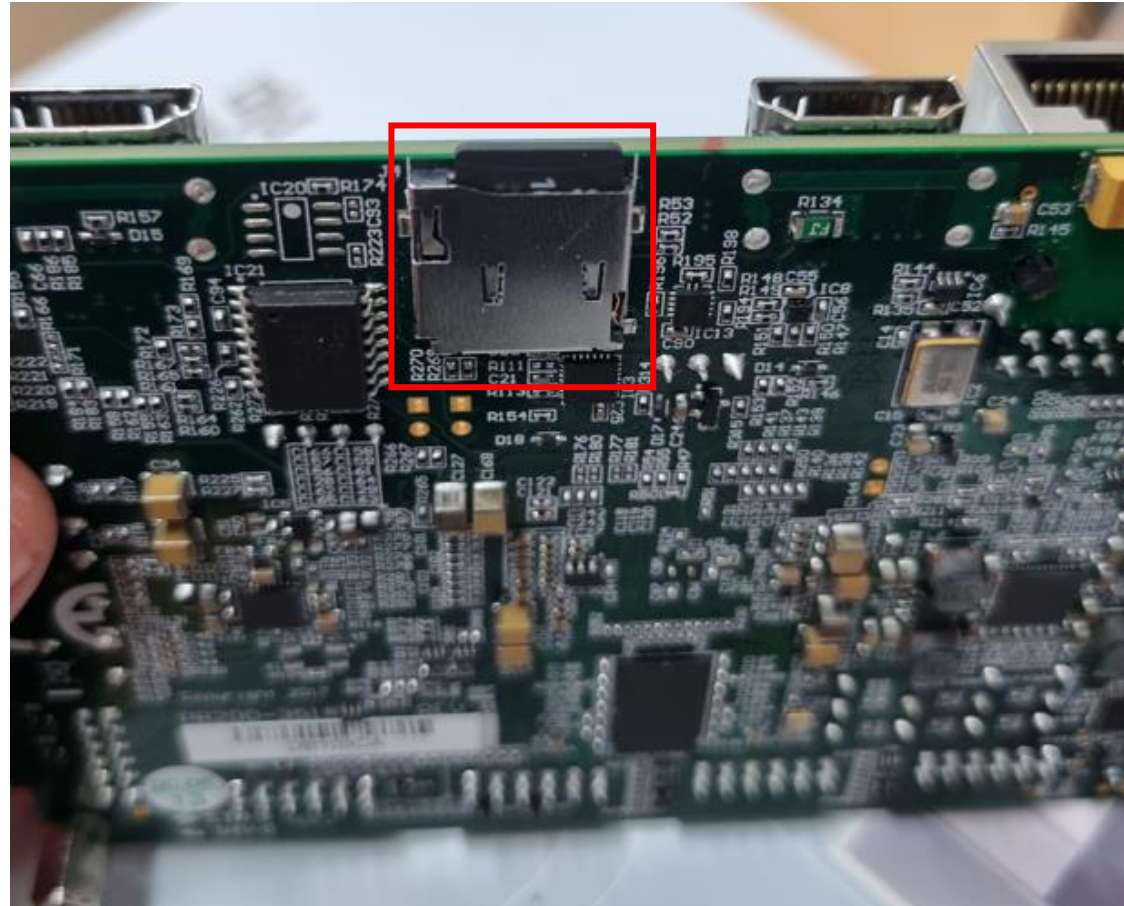
> SD카드에 펌웨어 복사

- SD카드를 PC에 연결한 후 SD카드에 **BOOT.bin** 파일을 복사



6. Bootloader image를 SD카드에 넣기

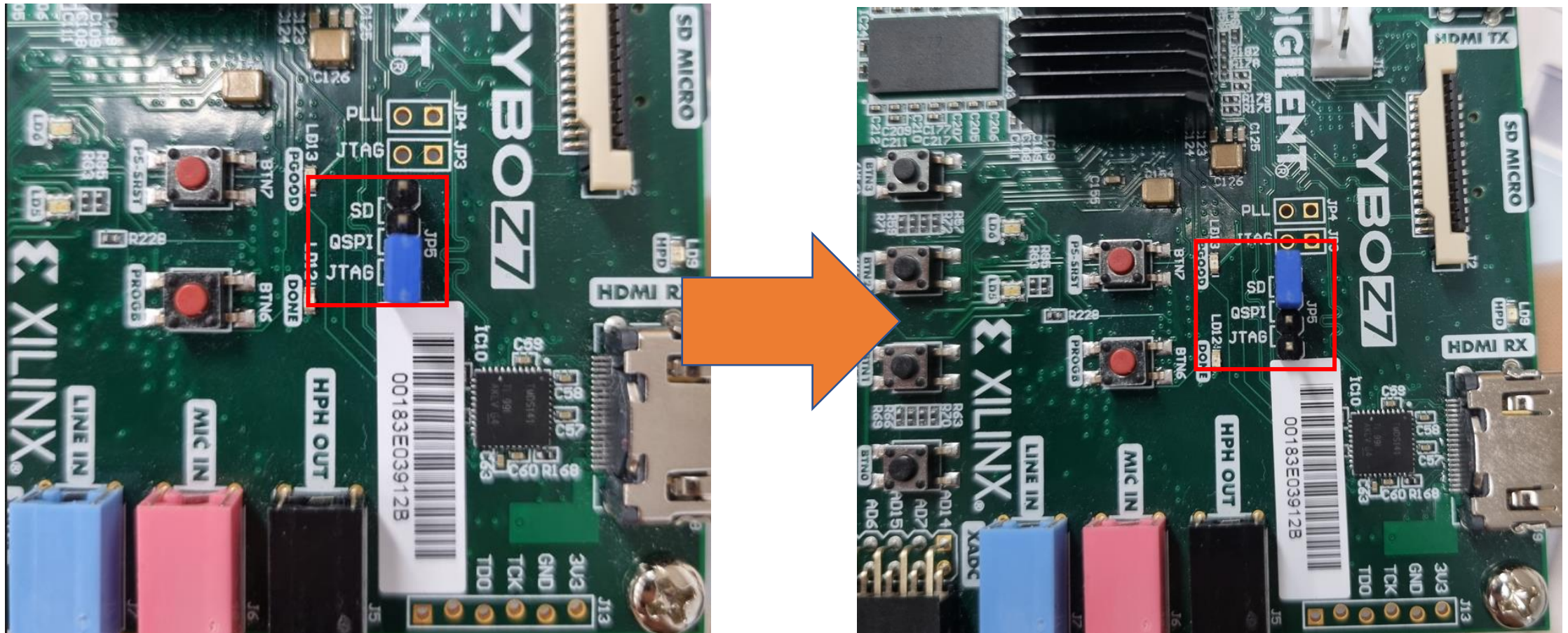
- SD카드를 보드 아래에 있는 슬롯에 장착



6. Bootloader image를 SD카드에 넣기

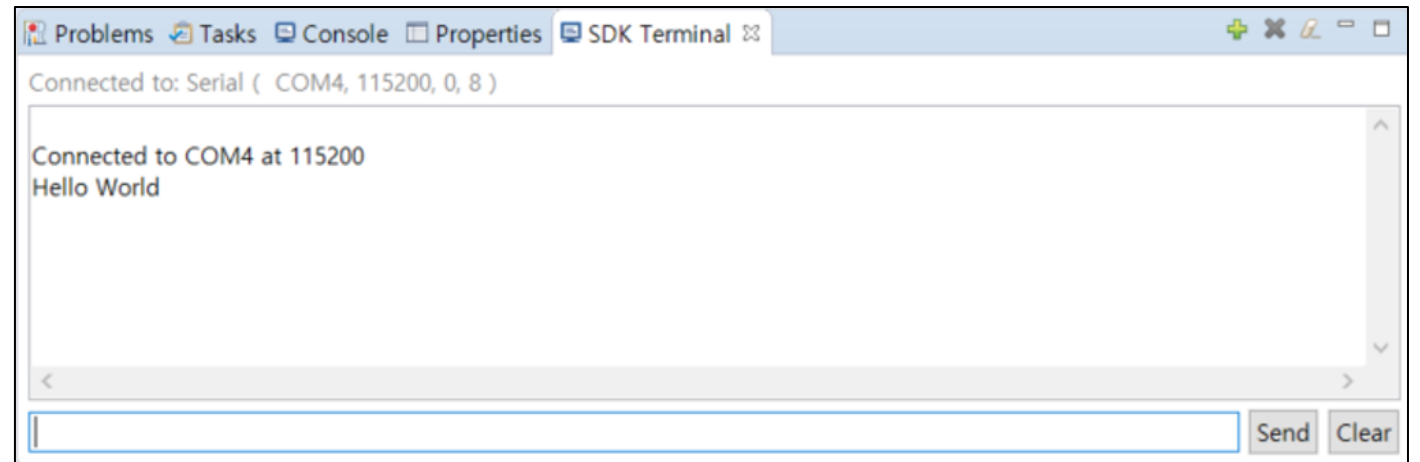
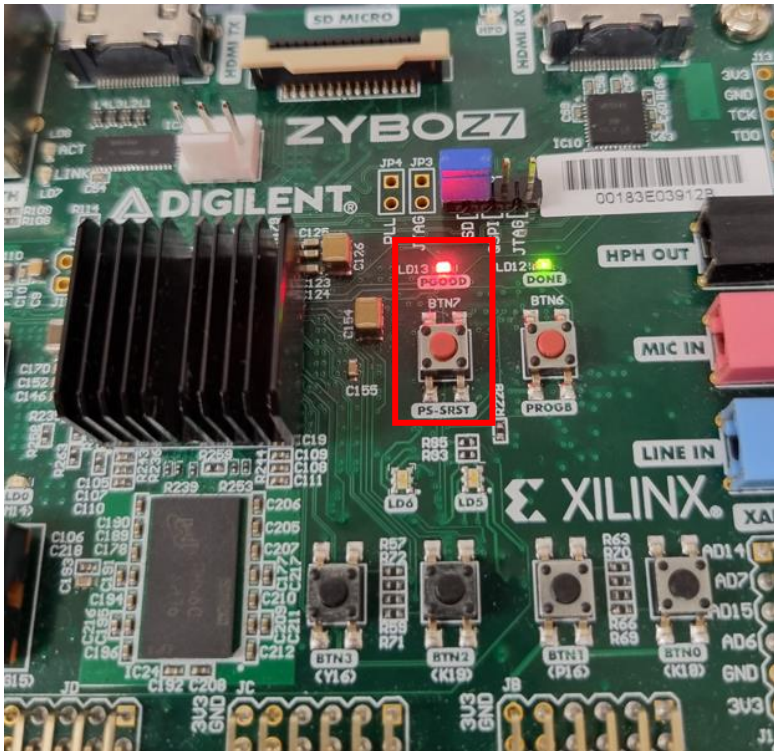
> SD카드를 보드에 삽입

- 점퍼를 바꿔 끼워서 JTAG 모드에서 **SD 모드**로 변경 (JP5)



6. Bootloader image를 SD카드에 넣기

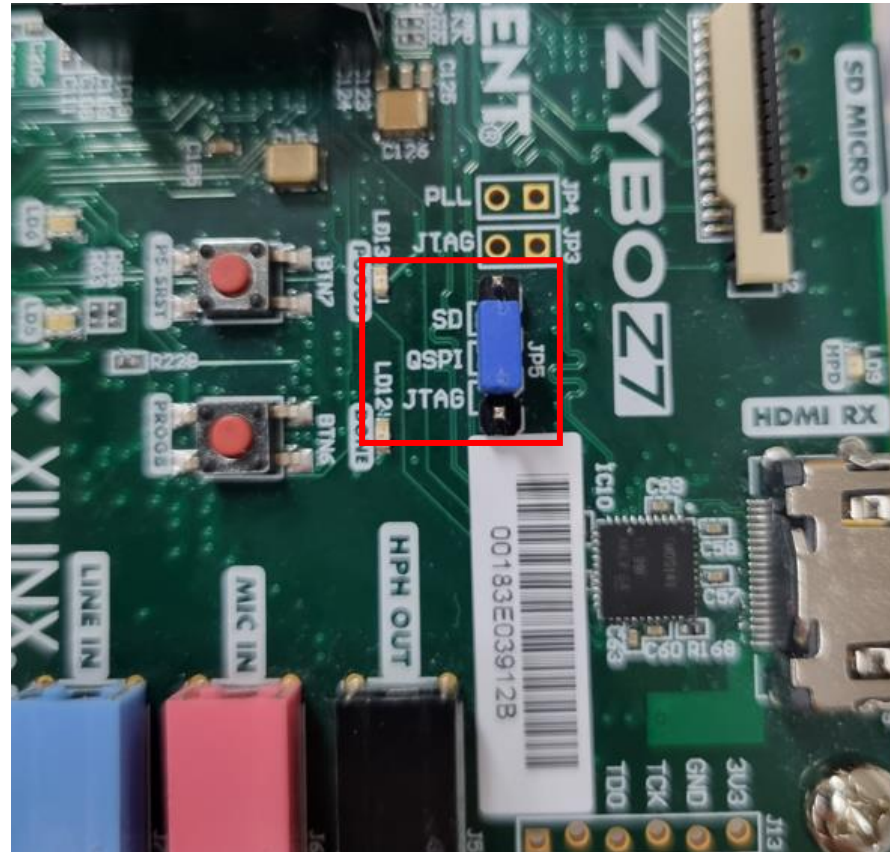
- > 보드를 실행하여 프로그램이 도는지 serial 출력으로 확인
 - 보드를 PC에 usb연결하고, terminal에 comport를 등록한 후, 전원을 켜면 보드가 자동으로 실행되면서 Hello World가 출력되는 것을 확인할 수 있음 (즉, hello_world.c가 자동 실행)
 - 만약 출력되지 않을 경우 **붉은 LED** 아래에 있는 **리셋버튼**을 누르면 재실행됨



7. Flash Writer를 이용하여 QSPI boot mode 사용하기

> 보드의 점퍼를 바꾸어 QSPI 모드로 변경

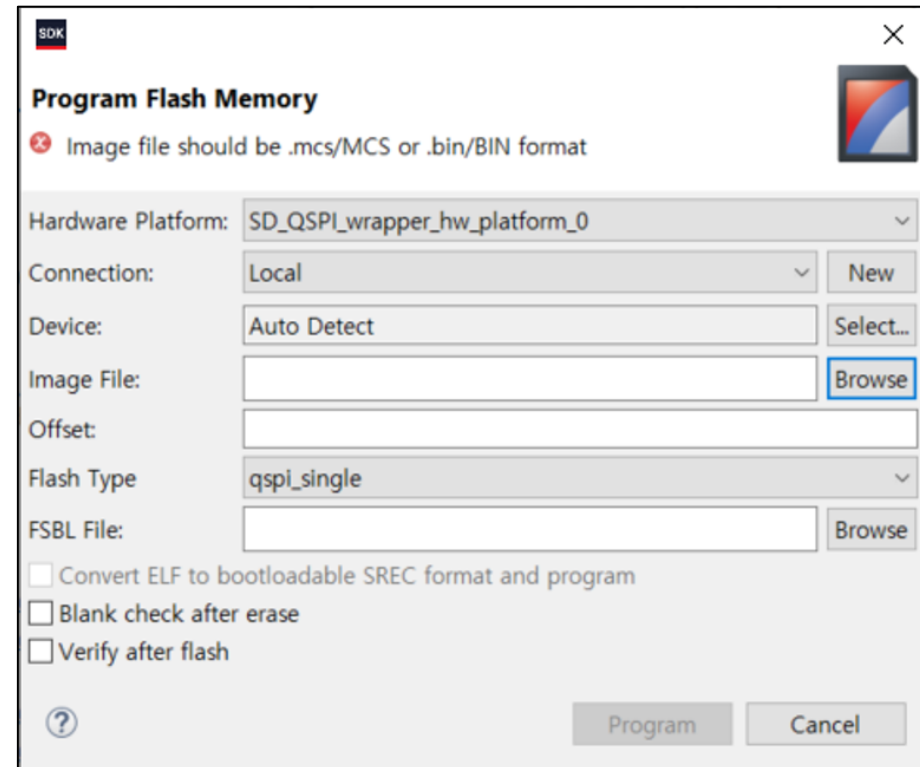
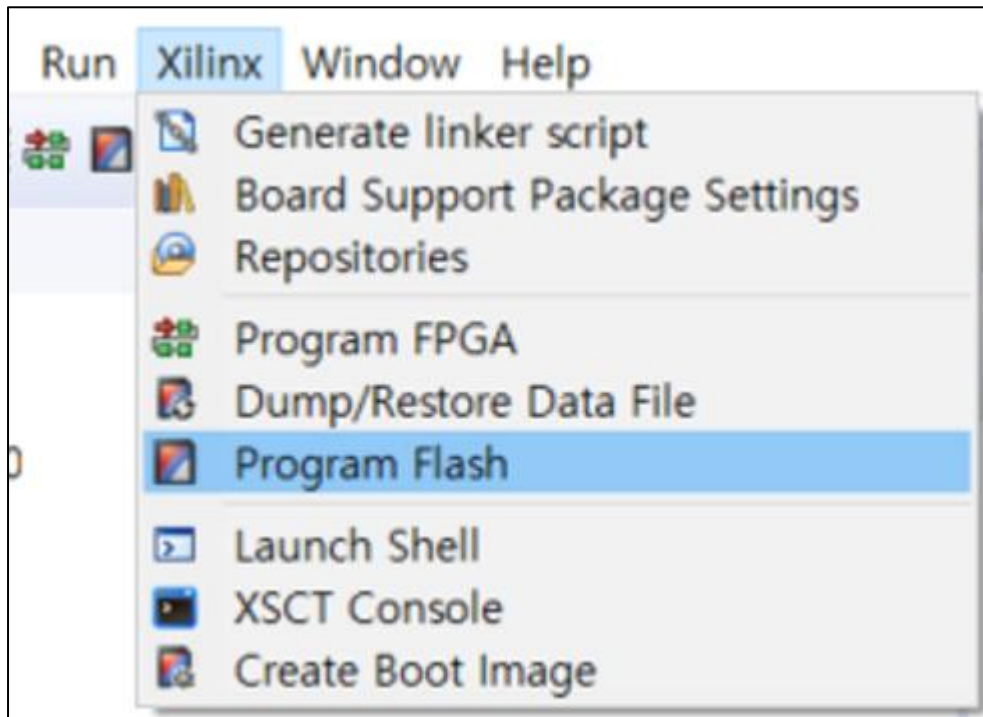
- 점퍼를 바꿔 끼워서 SD 모드에서 **QSPI** 모드로 변경 (JP5)
- 가운데 2개에 끼우면 됨



7. Flash Writer를 이용하여 QSPI boot mode 사용하기

> Flash Writer로 Program 실행

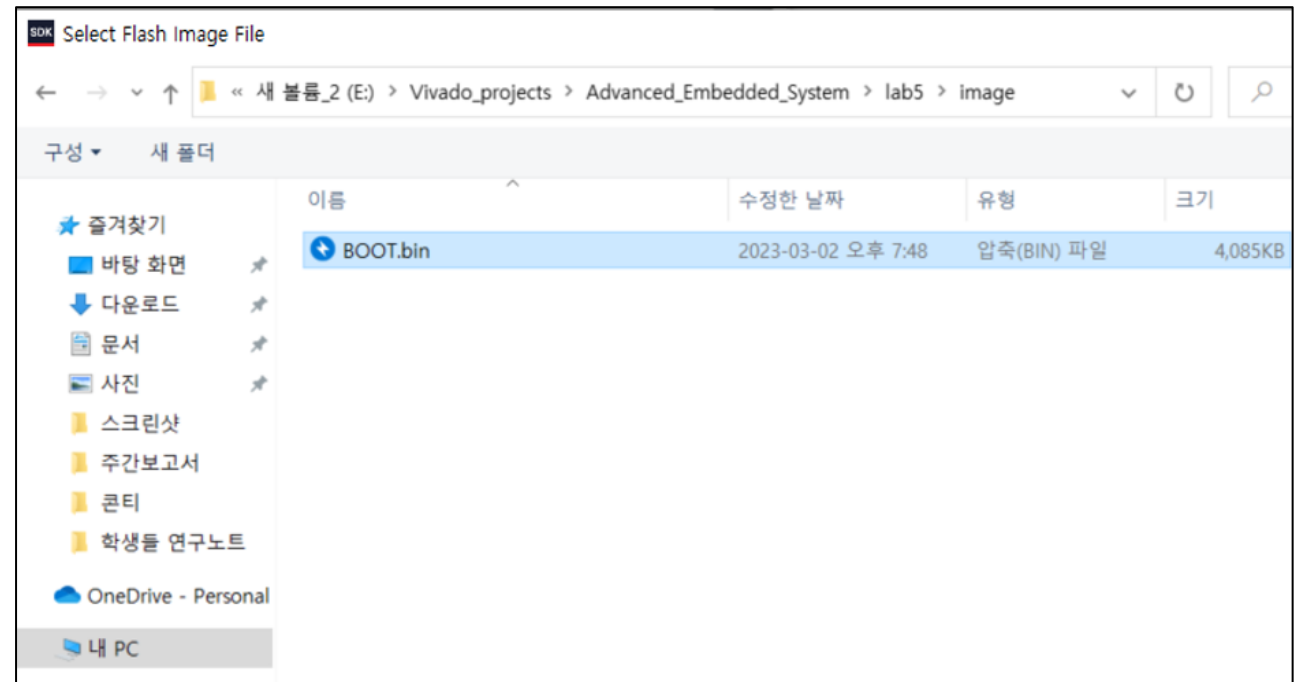
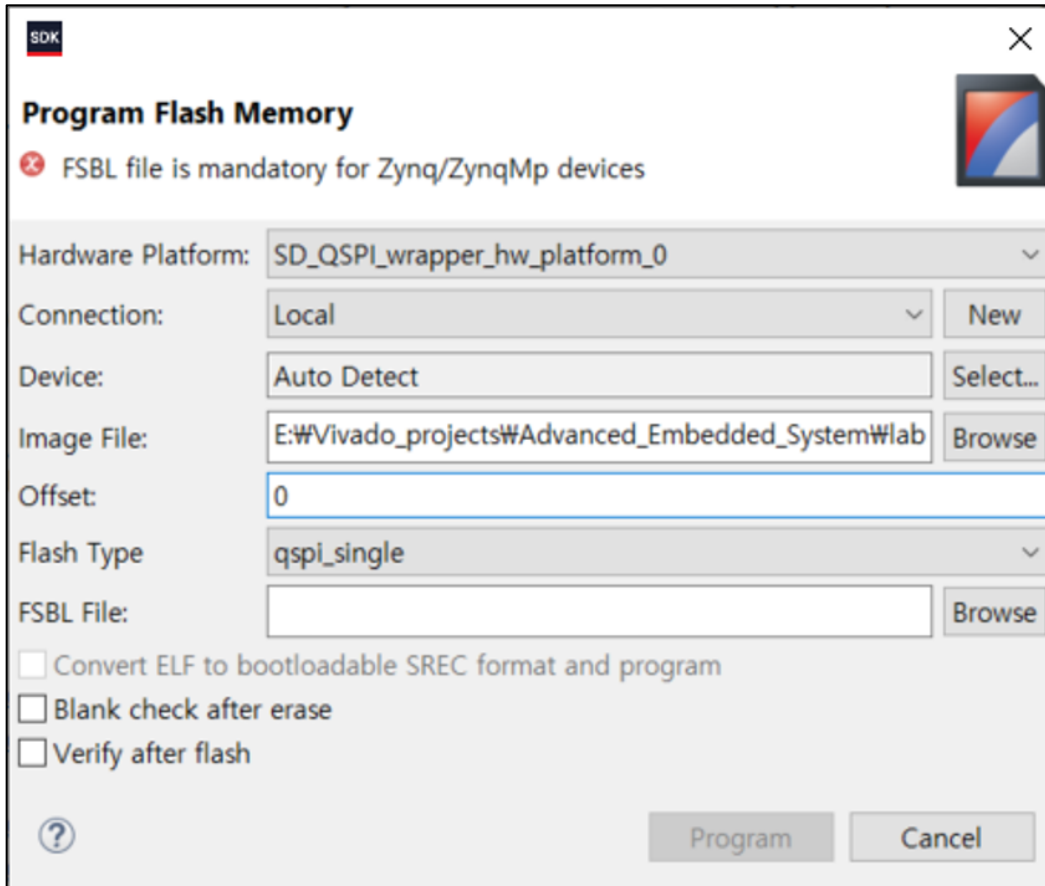
- 보드를 usb로 PC에 연결한 후 Xilinx > **Program Flash** 클릭



7. Flash Writer를 이용하여 QSPI boot mode 사용하기

> Flash Writer로 Program 실행

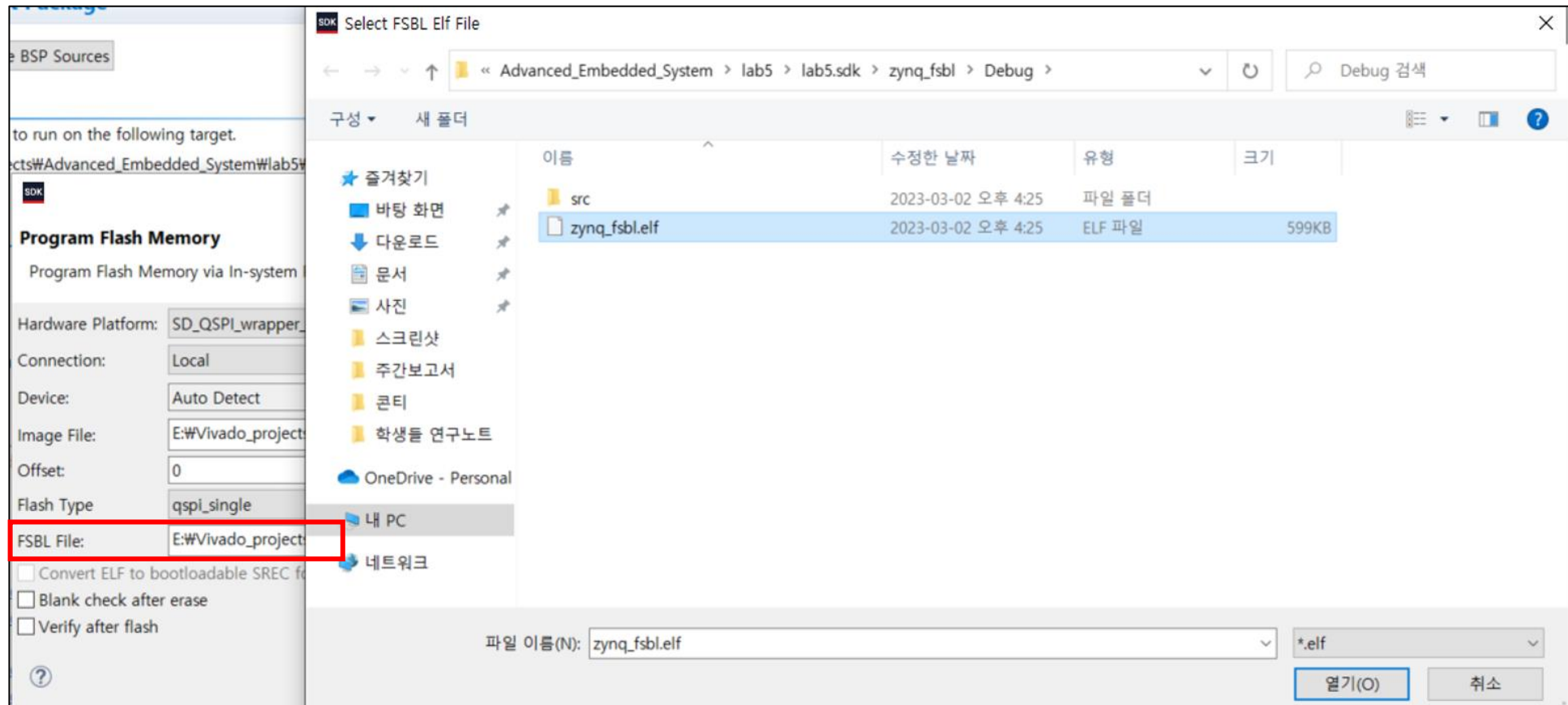
- Image File에 BOOT.bin 등록 & Offset = 0으로 설정



7. Flash Writer를 이용하여 QSPI boot mode 사용하기

> Flash Writer로 Program 실행

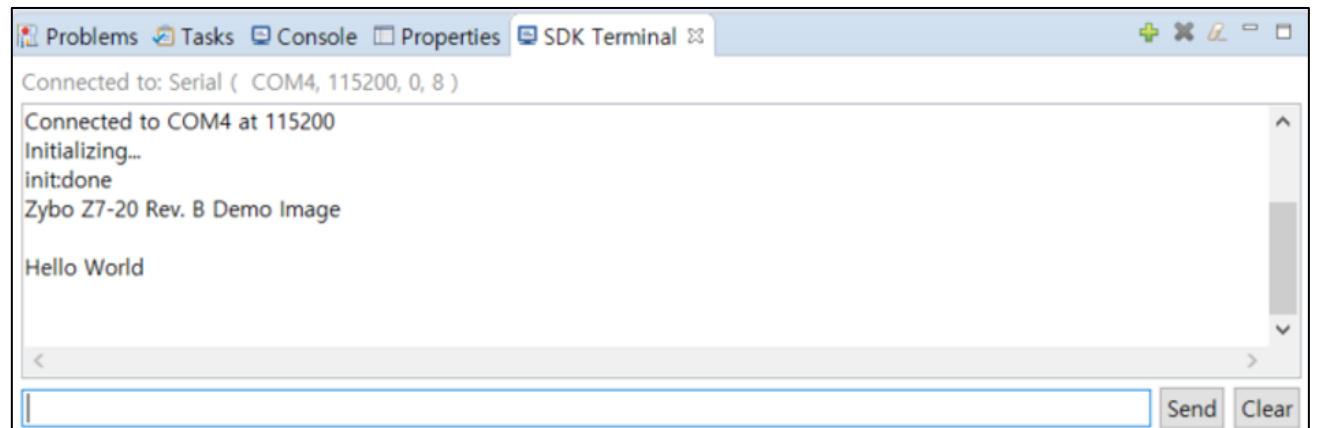
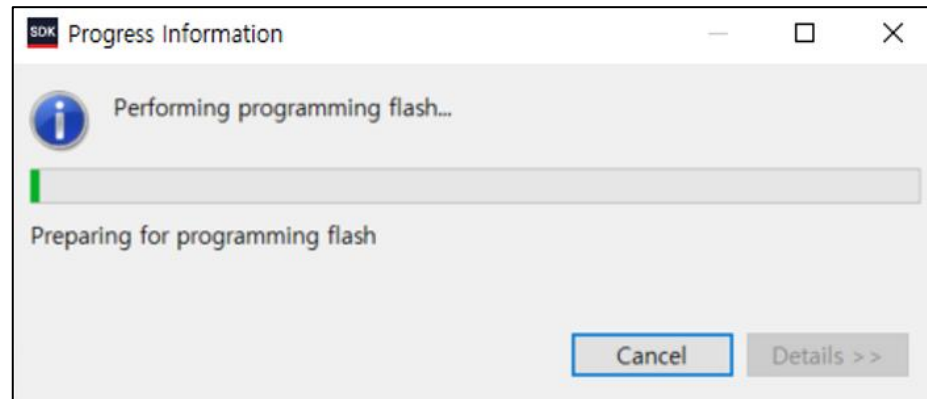
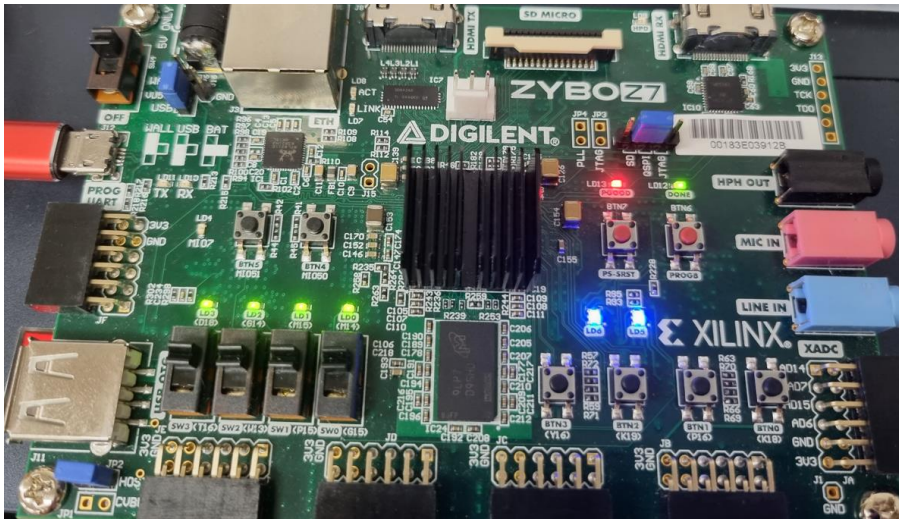
- FSBL 파일까지 등록하면 Program 버튼이 활성화됨 (**zynq_fsbl.elf**)



7. Flash Writer를 이용하여 QSPI boot mode 사용하기

> Flash Writer로 Program 실행

- Program을 완료하면 보드가 실행됨 (Hello World 출력되지 않으면 comport 재연결 하거나, 리셋버튼 누르면 됨)



감사합니다