Lab #7: Run and Debug IP

2018-04-22
2016-17101 김종범
2014-16511 박성관

- HW IP: Behavior of the shifter IP based on design sources

• myip_v1_0.v

이 ip에서는 Axi Slave Bus Interface의 Port들과 User Port들을 input/output port로 선언하고, 해당 port들을 myip_v1_S00_AXI.v에서 정의된 모듈에 이어주는 역할을 한다. 특히, User port에는 BRAM의 clock, 주소, write data, read data, enable, reset, write enable 신호 등이 연결되어 있다.

• myip_v1_S00_AXI.v

이 ip에서는 Axi Slave Bus Interface의 구현과, User가 구현한 부분이 포함되어 있다. 특히, 유저가 구현한 부분은 코드의 가장 아랫부분에서 확인할 수 있으며, 주요 동작은 다음과 같다.

1. S_AXI_ACLK의 posedge마다 상태(bram_state)를 변경한다. 기본적으로는 BRAM_IDLE 상태에 있으며, magic_code (32'h5555)를 확인하면 BRAM_READ -> BRAM_WAIT -> BRAM_WRITE의 과정을 4회 반복하고 다시 BRAM_IDLE 상태로 돌아간다.
2. BRAM_READ 상태에서는 데이터를 읽어오도록, BRAM_WRITE 상태에서는 데이터를 쓰도록 BRAM에 연결된 output port를 설정한다. BRAM_WRITE 상태에서 쓰는 데이터인 BRAM_WRDATA는 BRAM_RDDATA를 shift한 결과가 되도록 설정한다.
3. 4개의 주소에 대해 해당 연산을 수행하는 과정은 bram_counter 변수를 이용해 구현되었다. bram_counter는 0에서 3까지 증가하며 해당 위치의 주소를 READ/WRITE 동작에서 참조하도록 작성되어 있으며, 3이 되었을 때 동작이 종료되었다는 표시를 통해 BRAM_IDLE 상태로 돌아갈 수 있게 한다.

위 소스(mark debug 할 때와 같은 경로를 source 탭에서 찾으면 파일을 찾을 수 있음) 를 참고하여, waveform으로 출력한 세 개의 변수를 비롯하여 주요 변수로 생각되는 것들이 어떤 식으로 변화하는 지를 묘사. 그를 통해 해당 IP가 input을 받았을 때 어떤 output을 출력하게 되는 지를 설명

- HW System: Composition of the block design of Lab7

Processing System은 Application Processing Unit와 Main memory를 가지며, External interface logic for peripherals를 지원한다.
AXI bus는 Processing System과 Programmable logic을 연결하는 인터페이스를 제공한다
myip는 Lab7에서 새롭게 추가한 IP이다. myip에는 shifter가 연결되어있다.
BRAM은 FPGA안에 있는 RAM구조를 가진 블럭이다.
block design의 각 block이 어떤 일을 하고, 따라서 전체 block design이 어떤 일을 하는 지를 묘사

- SW: Role of each partitions of the SD card and program codes.

• 1st partition(1.1GB partition)

하드웨어 디자인 언어를 컴파일 하여 bitstream화 한 결과를 zynq.bit라는 이름으로 가지고 있다. FPGA는 이 bitstream을 해석하여 하드웨어의 동작을 결정한다.

또는 Lab시간에서처럼 HOST에서 BOARD로 파일을 옮기고 싶을 때, 저장소로도 사용 가능한다. (root의 sdcard폴더에서 확인 가능하다.)

• 2nd partition(15GB partition)

장치를 부팅시킬 때 쓰는 메모리 partition이다.

• main.c

void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset)이다.
의미 : mmap은 파일이나 장치를 메모리에 대응시키거나 푸는 함수이다.
Argument 설명
addr : 커널에게 어디에 맵핑하면 좋을지 제안하는 값으로 보통 0(NULL)을 사용한다.
length : 대응시킬 바이트의 양, prot : 맵핑에 원하는 메모리 보호 정책,
flags : 맵핑 유형과 동작 구성요소, fd : 지정된 파일, offset : 맵핑의 시작 기준점.

- **Makefile**

make 또는 make all로 gcc main.c && sudo ./a.out를 실행할 수 있다.
gcc main.c를 하면 기본적으로 a.out 실행파일이 생성되므로, main.c에 대한 실행파일이 실행된다.

Tutorial 1, 2의 스크린샷