Lab 3. 타이머/PWM IP 구현 및 레지스터 설정

지금까지 실습한 Wishbone bus 인터페이스 과제를 바탕으로 타이머 등의 대표적인 IP 모듈을 코딩하고, 모의실험으로 동작의 타당성을 확인.

실습 1. [따라하기] 타이머 IP 모듈 생성

1. 32bit 카운터를 하나 생성하여 매 클럭 마다 1씩 업 카운트. 카운트 값이 미리 정해 둔 값에 도달하면 0부터 다시 업 카운트. 이 동작을 무한히 반복.
2. 타이머 IP 모듈에 출력 단자(oTimer)를 1개 추가하고, 1의 카운트 값이 설정 값에 도달하면 1클럭 사이클 동안 high pulse가 출력되도록 설정.
3. 미리 정해 두는 값은 32bit register에 저장되는데, 이 레지스터에는 32'h0200_1000의 주소를 할당. Address decoding은 centralized 방식을 적용하며, 최대 4개의 32bit 레지스터를 생성 가능하도록 16 byte 주소 공간을 할당. Lab02의 디지털 출력 포트 module을 수정하면 편리. 결국, 32'h0200_1000의 레지스터를 설정하여 oTimer 단자로 출력되는 1 클럭 사이클 펄스의 발생 주기를 조절할 수 있음.
4. 모의 실험을 통하여 동작 확인.
5. 기능을 추가한다. 타이머 동작 ON/OFF를 설정하고, 타이머 출력을 pulse 또는 toggle을 선택할 수 있는 옵션 설정 레이스터를 32'h0200_1004에 추가. (bit 0: ON/OFF, bit 1: pulse/toggle
6. Reset value 등은 parameter로 설정.
7. 모의 실험을 통하여 동작을 확인.

실습 2. [연습과제] PWM(Pulse Width Modulation) IP 모듈 생성

1. PWM은 구형파인데, 구형파 주기는 고정한 채로 high 구간의 길이를 다르게 조절한 파형을 생성하는 PWM module을 코딩.
2. 12 bit UP Counter를 생성하고, MCLK로 계속 카운트. 12'h000 ~ 12'hFFF 카운트 후 0으로 overflow 후 count up 무한 반복.
3. 12bit duty register를 생성하고, 주소 32'h0200_2000를 이용하여 read/write하는 기능 구현. Address decoding은 centralized 방식을 적용하며, 16 byte 주소 공간을 할당.
4. 12bit Up Counter와 12bit duty register의 값을 비교하여 카운터 값이 크면 low, 카운터 값이 작으면 high를 출력하는 코드 작성. 출력은 oPWM 단자로 출력.
5. 12bit duty register는 read/write 모두 가능해야 하며, 12 bit duty register의 reset 직후 초기값은 12'h800으로 설정하여 초기의 PWM 파형은 high와 low 구간 길이가 같도록 설정. Reset value 등은 parameter로 처리.
6. 32'h0200_2000 주소로 duty register 값을 설정하면 출력 파형의 high/low 구간 길이가 변경됨을 모의실험으로 확인.
7. PWM 객체를 1개 더 생성하여, 주소는 32'h0200_3000을 사용하고, 코딩 후 모의실험으로 동작 확인.

제출 대상: 압축한 코드, 모의실험 화면 캡쳐하여 제대로 동작함을 보고서로 제출(waveform을 알아볼 수 있도록 캡쳐)