

Final Project. 디지털 시계 만들기

2018 Fall Logic Design LAB

Department of Computer Science and Engineering

Seoul National University

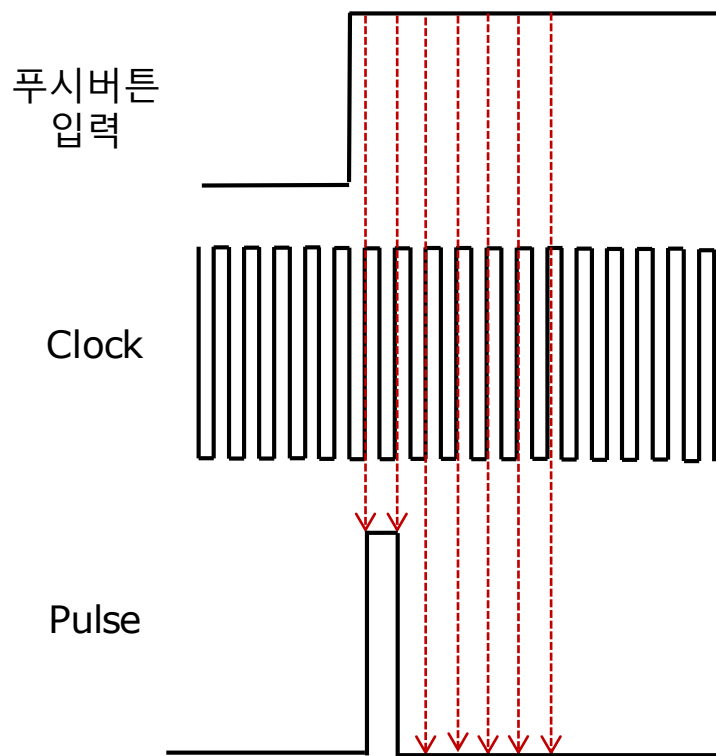
1. Pulse Generator
2. 디지털 시계 기본 구현 사항 설명
3. 추가 구현을 위한 힌트
4. Programmable Watch 추가 구현 설명

Pulse Generator

- 높은 주파수의 clock에 synchronous한 FSM 설계
 - 푸시 버튼 입력이 **여러 번** 반영되는 문제가 발생

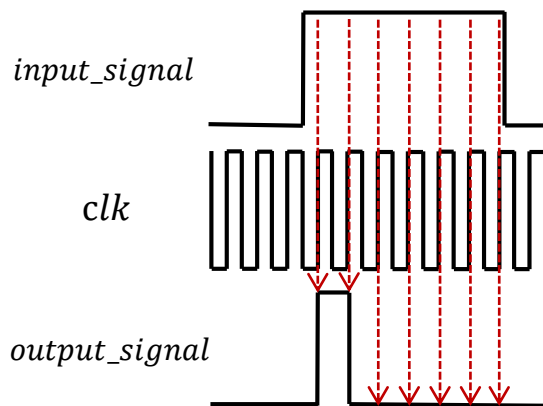
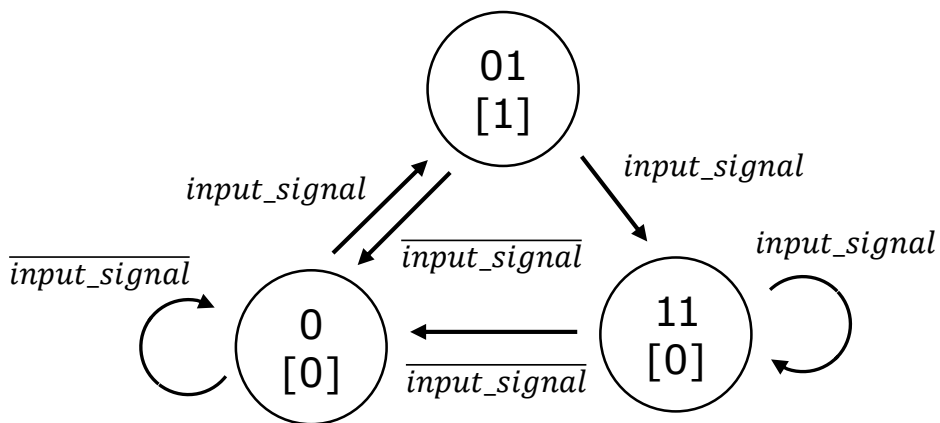
- Pulse Generator

- 긴 시간 동안의 푸시 버튼 입력을 한 cycle 동안의 입력으로 변환



Pulse Generator FSM

- 다음과 같은 FSM의 설계로 문제를 해결



```

module PulseGenerator (
    input clk,
    input input_signal,
    input reset,
    output output_signal
);
    parameter s_0 = 2'd0;
    parameter s_11 = 2'd1;
    parameter s_01 = 2'd2;

    reg[1:0] state; reg[1:0] next_state;

    assign output_signal = (state == s_01) ? 1'b1 : 1'b0;

    always @(*) begin
        case (state)
            s_0:
                next_state = (input_signal == 1'b1) ? s_01 : s_0;
            s_11:
                next_state = (input_signal == 1'b1) ? s_11 : s_0;
            s_01:
                next_state = (input_signal == 1'b1) ? s_11 : s_0;
            default:
                next_state = s_0;
        endcase
    end

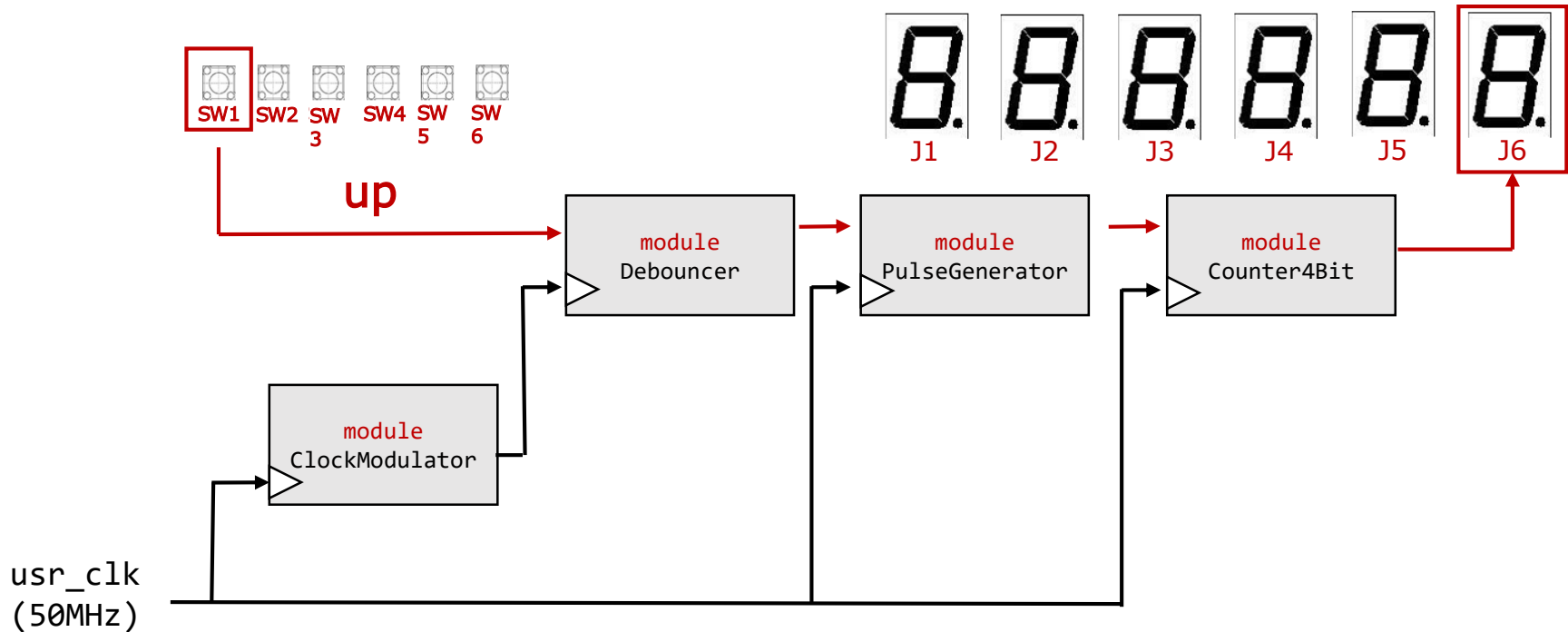
    always @(posedge clk) begin
        state <= (reset == 1'b1) ? s_0 : next_state;
    end
endmodule
  
```

실습 – 4-bit binary up counter 구현

- 목표
 - 4-bit binary up counter를 50MHz clock에서 동작하도록 구현
- 실험 내용
 - ClockModulator, Debouncer, PulseGenerator 구현
 - 구현한 모듈들을 이용해 4-bit binary up counter 구현
 - SNU Logic Design 보드에 올려 동작 검증
- 제출 사항: 없음

실습 – 4-bit binary up counter 구조

- 푸시 버튼을 한 번 누를 때 마다 카운터가 1 씩 증가하도록 구현



기말프로젝트: 디지털 시계 디자인

디지털 시계 디자인

■ 목표

- 이론 시간과 실습 시간에 학습한 내용을 활용해 디지털 시계를 구현

■ 기본 구현

- 시계 기능

- 현재 시간을 정확히 1초 마다 증가시킨다.
- 임의의 시간과 분으로 시간을 설정한다.

- 알람 기능

- 알람시간 (시/분 단위)을 설정한다.
- 알람으로 설정해놓은 시간이 되면 시계 기능에서 이를 표시한다.

- 스톱워치 기능

- 1/100초 단위까지 측정하는 스톱워치를 구현한다.

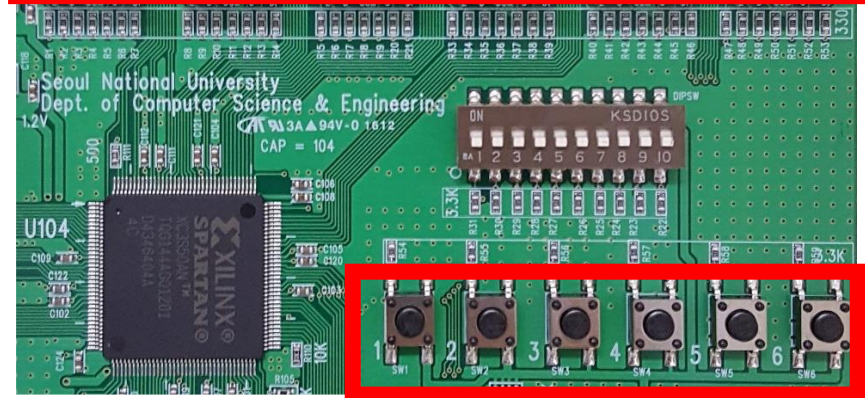
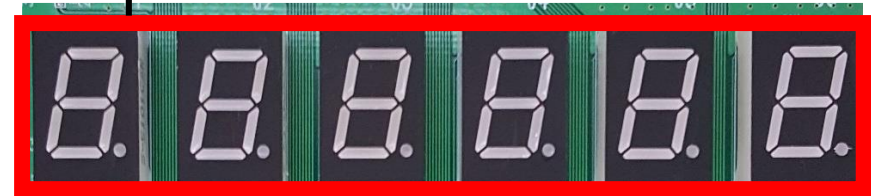
디지털 시계 디자인: 추가 구현

- 자유롭게 기본 구현 사항에 더해 구현 가능한 부분을 확장하여 구현
- 예시
 - Programmable Watch
 - 다른 보드에 미리 Program된 일련의 Instruction을 읽고 이를 입력값으로 동작
 - 디스플레이 관련 추가 구현
 - 디스플레이 밝기 조절
 - 스톱워치 lap 저장 기능
 - 진행 중인 스톱워치를 중단하지 않고 2개 이상의 lap을 저장
 - 그 외의 다양한 구현 가능
- 추가 구현 시 새로 구현된 기능이 기능적으로 새로운 구현이 아니거나 난이도가 낮은 경우 추가 구현으로 인정하지 않음 (예. 시간의 16진수 표시)

입출력 개요

- 7-segment LED (J1, 2, 3, 4, 5, 6)
 - 기본적으로 시간을 10진수로 표시
 - 추가 구현 시 모드 표현 등에 사용 가능
- 푸시 스위치 (SW1~6)
 - 6개의 스위치 중 5개는 지정된 기능 수행
 - SW1(MODE): 모드 변경 버튼
 - SW2(SET): 설정 혹은 확인 버튼
 - SW3(OP1): 모드에 따른 기능1 수행
 - SW4(OP2): 모드에 따른 기능2 수행
 - SW6(RESET): 디지털 시계의 RESET
 - SW5: 지정된 기능이 없어 추가 구현 시 자유롭게 사용 가능
- 6개의 색상 LED(D1 ~ D6), DIP 스위치를 자유롭게 구현에 사용

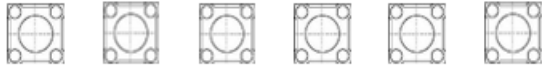
출력값(시간, 모드 등) 표시



입력값 받기

기본 구현: 디지털 시계의 초기 상태

■ 입력



- 어떤 상태에 있더라도 SW6(RESET)을 누르게 되면 초기 상태로 돌아온다.
- 디지털 시계의 초기 상태
 - 모드: 시계 모드
 - RESET이 눌러져 있는 동안: 00시 00분 00초 유지
 - 설정한 알람 시간 등 모든 설정 값 초기화
- 디지털 시계 동작 시 무조건 초기조건으로 RESET을 누르고 시작

기본 구현: 시간 표시법 (1)

- 시계 기능과 알람 기능

- 각각 현재의 시간과 알람 시간을 24시간 표현법과 12시간 표현법으로 표시
- Leading zero 표현

- 24시간 표현법

- 시, 분, 초 순서대로 2자리씩 출력



- 자정의 경우 오후와 같이 표현

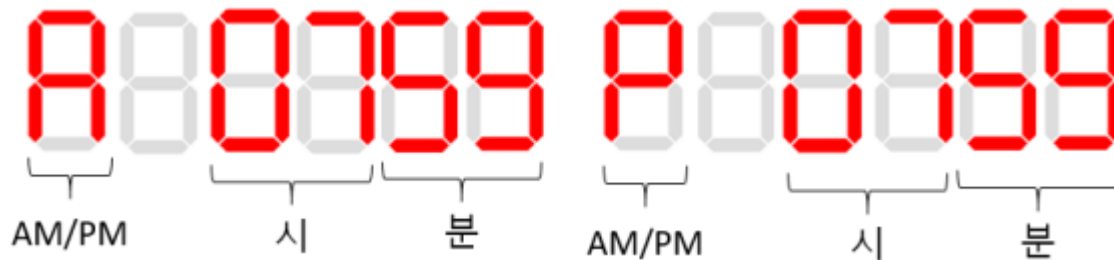
(00시 00분 00초)



기본 구현: 시간 표시법 (2)

- 12시간 표현법

- AM/PM, 시, 분 순서대로 출력하고 초 단위는 생략



- 정오는 PM 12시 00분, 자정은 AM 12시 00분으로 표현

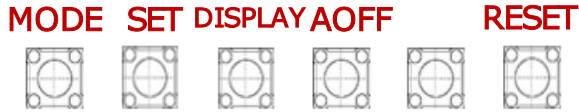


기본 구현: 시계 기능

■ 시계 표현 방법

- 시간 표현은 12시간/24시간 표시법으로 표현

■ 입력



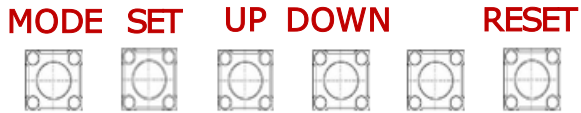
- MODE: 현재의 시계가 진행되는 상태에서 알람 설정 모드로 진입
- SET: 시계의 시간 설정으로 진입
- DISPLAY: 12/24시간 표시법 간의 변환
- AOFF: 알람이 울리고 있는 경우 알람을 끄
 - 울리지 않은 경우에는 아무 동작 없음
- RESET: 디지털 시계의 초기 상태로 돌아감

기본 구현: 시계 시간 설정 기능 (1)

■ 시계 시간 설정 표현 방법

- 시간 표현은 시간 설정으로 진입하기 전 시계의 표시 설정에 따라 표시
 - 24시간 표현법에서 진입할 경우 24시간 표현법으로 표시
 - 12시간 표현법에서 진입할 경우 12시간 표현법으로 표시

■ 입력



- MODE: 현재의 시계가 진행되는 상태에서 알람 설정 모드로 진입
- SET: 현재 설정 중인 항목을 저장 후, 다음 설정으로 이동
- UP: 현재 설정 중인 항목의 값을 올림 (예. 1시 → 2시)
- DOWN: 현재 설정 중인 항목의 값을 내림 (예. 2시 → 1시)
- RESET: 디지털 시계의 초기 상태로 돌아감

기본 구현: 시계 시간 설정 기능 (2)

■ 동작: 12시간 표현법

- 12시간 표현법: AM/PM - 시 - 분 순서대로 설정
- 현재 설정 중인 항목은 점멸

- UP/DOWN을 누르면 점멸하고 있는 부분만 변경
- 분 항목까지 설정하고 SET을 누르면 시계로 돌아옴
- 설정한 시간의 00초부터 정확하게 시작



시계 모드로 복귀

기본 구현: 시계 시간 설정 기능 (3)

- 동작: 12시간 표현법
 - 24시간 표현법: 시 - 분 순서대로 설정 (초는 00으로 고정)
 - 현재 설정 중인 항목은 점멸
 - UP/DOWN을 누르면 점멸하고 있는 부분만 변경됨
 - 분 항목까지 설정하고 SET을 누르면 시계로 돌아옴
 - 설정한 시간의 00초부터 정확하게 시작

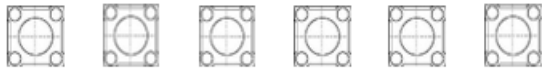


기본 구현: 알람 기능 (1)

■ 시계 표현 방법

- 시간 표현은 앞서 설명한 12시간/24시간 표현법으로 표현
- 시간표현 방식의 경우 직전의 시계 기능과 동일할 필요 없음

■ 입력 MODE SET DISPLAY CLEAR RESET



- MODE: 현재의 알람 기능이 지속되는 상태에서 스톱워치 모드로 진입
- SET: **알람 시간 설정**으로 진입
 - 알람 시간 설정 방식은 시계 시간 설정과 동일
- DISPLAY: 12/24시간 표시법 간의 변환
- CLEAR: 현재 설정되어 있는 알람을 취소
- RESET: 디지털 시계의 초기 상태로 돌아감

기본 구현: 알람 기능 (2)

■ 알람 시간 표현 방법 (12시간 표현법 가정)



알람이 없는 경우



AM/PM

시

분

알람이 있는 경우

■ 알람 표시 방법

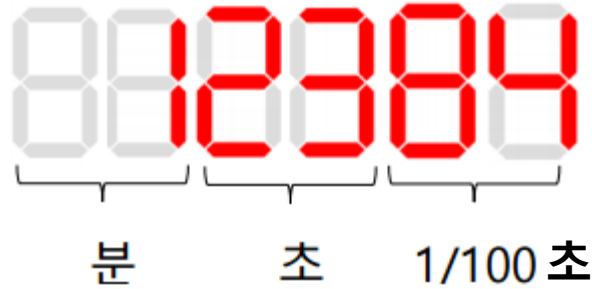
- 설정된 알람과 시계의 현재 시간이 일치할 경우 알람이 울림
- 시계 모드에서 AOFF를 눌러야만 알람이 정지함
- 시간이 지나도 AOFF가 눌러지지 않으면 계속 알람이 울림
- 알람이 울리는 것은 시계 모드에서 모든 글자가 점멸하는 방식으로 기본 구현



점멸


기본 구현: 스톱워치 기능

■ 스톱워치 표현 방법



- 앞의 두 자리는 분, 중간 두 자리는 초, 나머지 두 자리는 1/100초 단위
- 분/초 단위는 Leading Zero 제거, 1/100초 단위는 Leading Zero 미 제거

기본 구현: 스톱워치 기능

- 입력 MODE SET CLEAR RESET

 - MODE: 현재의 스톱워치가 진행되는 상태에서 시계 모드로 진입
 - SET: 스톱워치를 시작하거나 일시 정지
 - CLEAR: 스톱워치가 정지한 상태에서 시간을 00분 00.00초로 초기화
 - RESET: 디지털 시계의 초기 상태로 돌아감
- 동작
 - 59분 59.99초에 다다른 경우 더 이상 진행하지 않고 멈춤
 - 59분 59.99초에서 멈춘 경우 SET을 눌러도 스톱워치가 진행하지 않음

기본 구현

■ 기본 구현 시 참고사항

- 자세한 구현 세부는 ETL에 같이 공지된 스펙 문서를 참조
- 불충분하다고 생각하는 혹은 추가하고 싶은 스펙이 있다면 스스로 스펙을 정하고 이를 구현
- 스스로 정한 스펙이 있다면 이를 보고서에 작성하여 제출

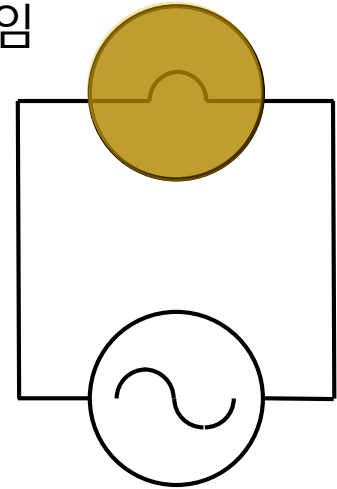
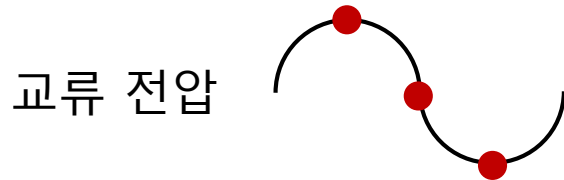
■ 권장사항

- 과제의 복잡도가 높으므로 **모듈화** 하여 구현을 시작할 것
- 각 모듈마다 **시뮬레이션**을 통해 동작을 확인할 것

추가 구현 힌트: 밝기 조절 (1)

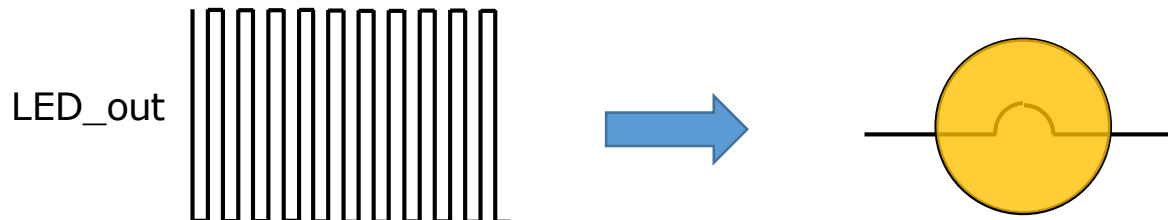
■ 교류 전구

- 60 Hz 220V에 연결하면 1 주기(1/60초)에 2번씩 반짝임
- 사람의 눈에는 점멸을 반복하는 것처럼 보이지 않는다



■ 기본 아이디어

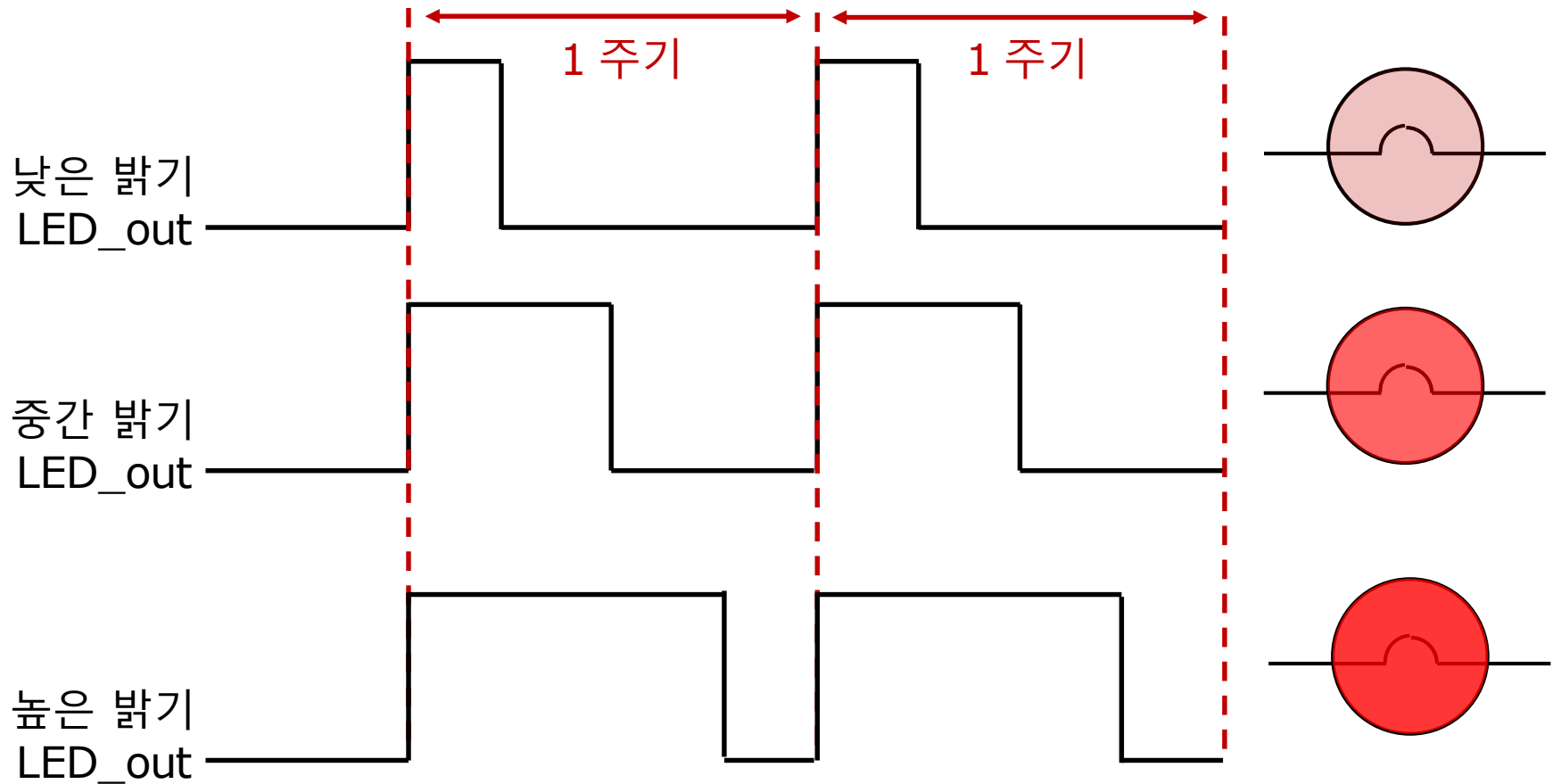
- LED를 빠르게 깜빡이도록 하면 계속 켜져 있는 것처럼 보임



추가 구현 힌트: 밝기 조절 (2)

- 밝기를 조절하는 방법

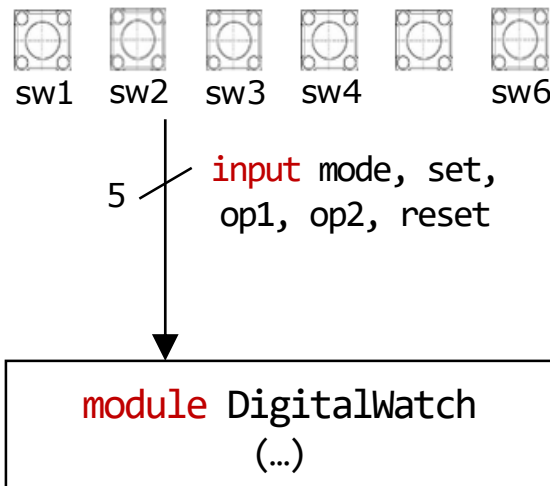
- 충분히 빠른 주파수의 신호에서 켜져 있는 부분의 비율을 높이면 밝아짐



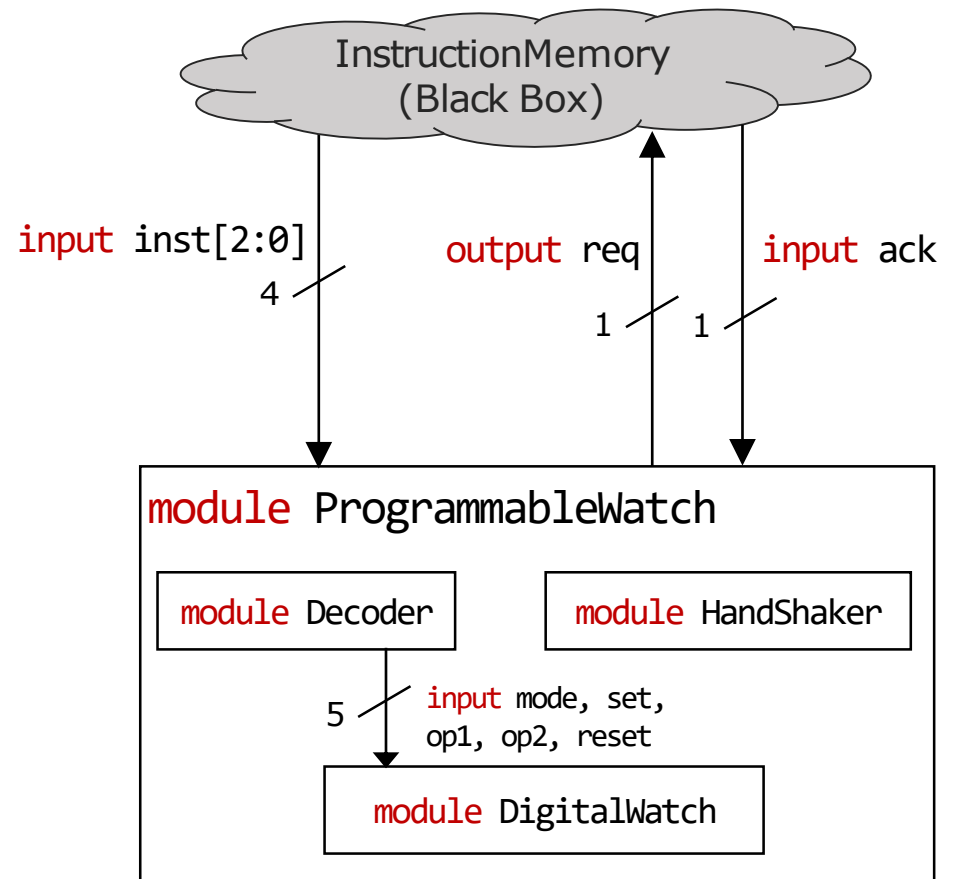
추가 구현: Programmable Watch

- 입력 값을 Push button이 아닌, 레지스터에 미리 저장한(programming된)값에서 받아 차례로 수행하는 시계

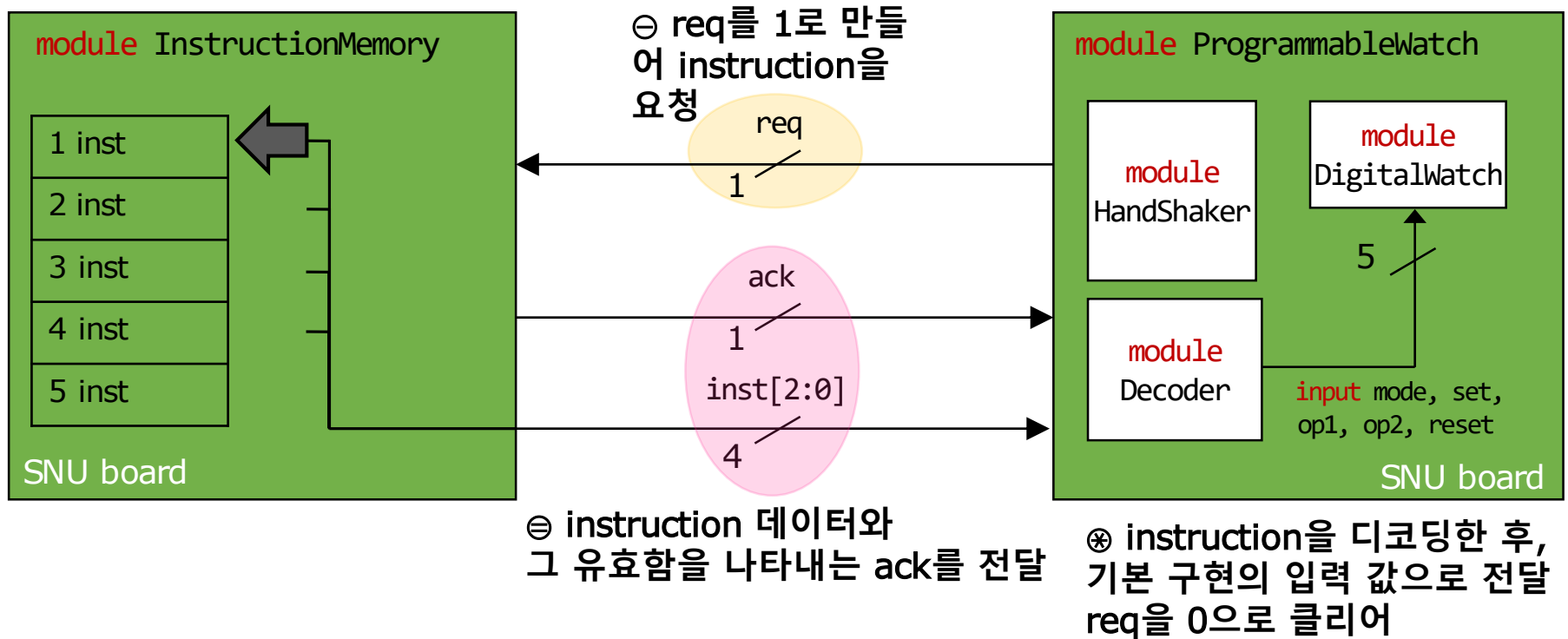
기본 구현



Programmable Watch 구현

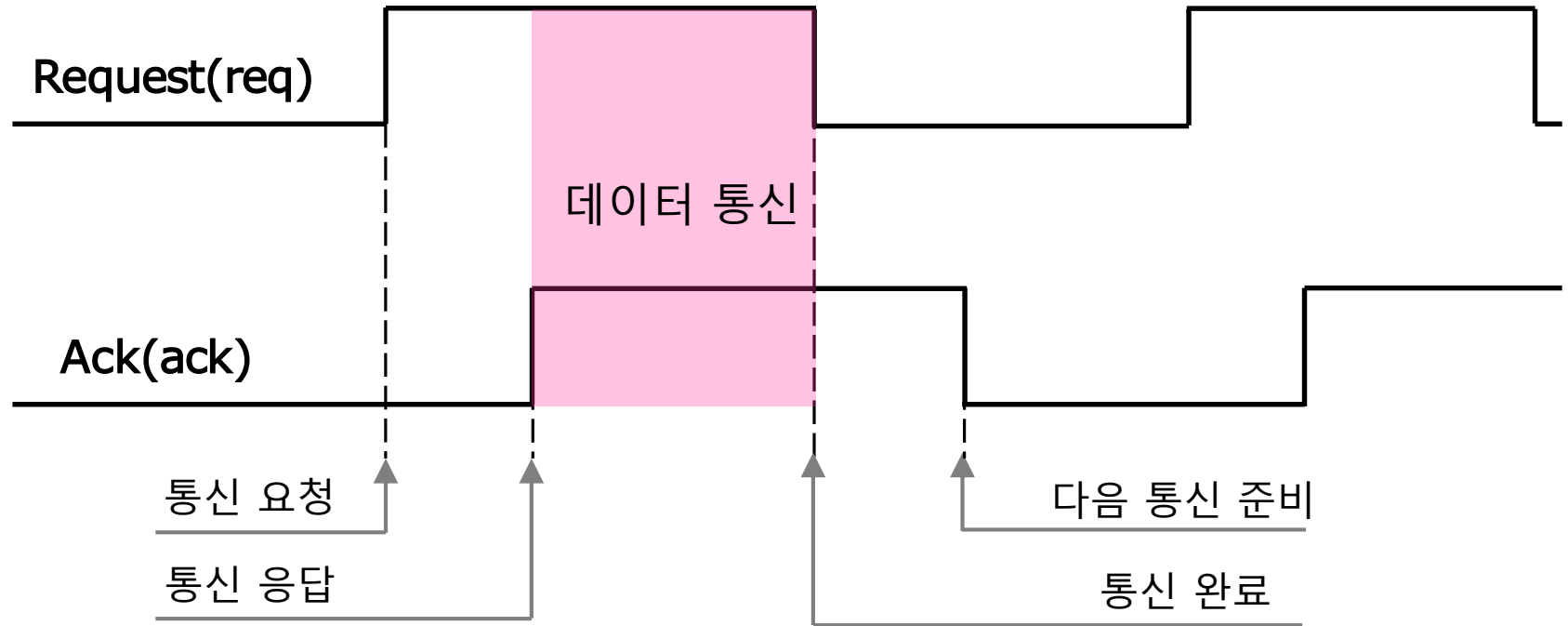


Programmable Watch: 시나리오



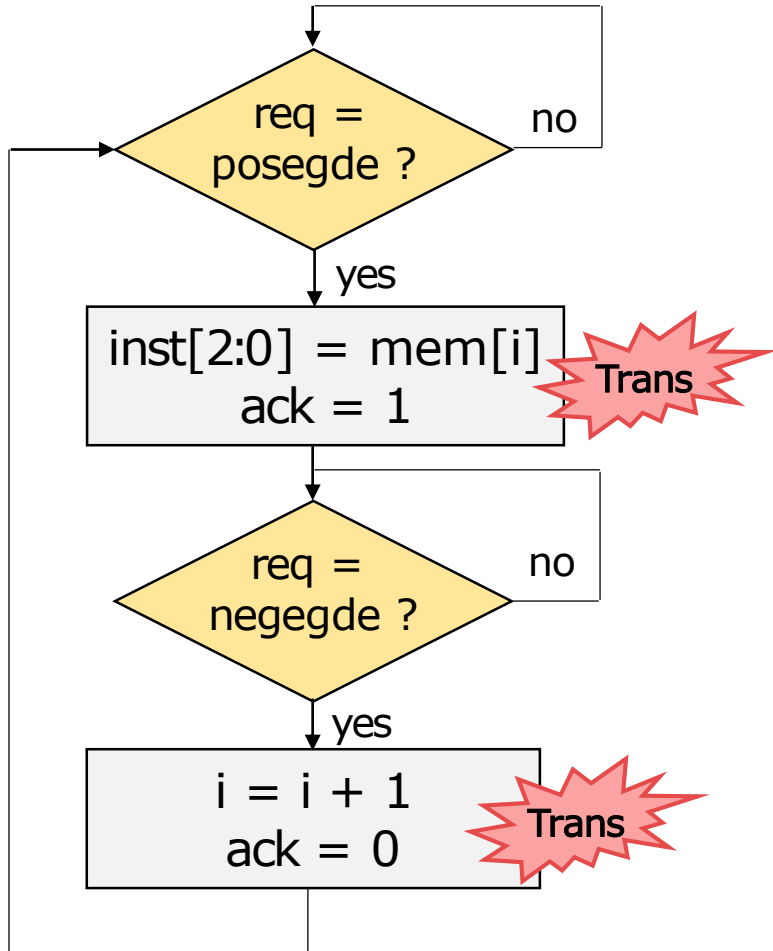
- 외부 보드(InstructionMemory)와 통신하여 Instruction 데이터를 받아오고, 이를 디코딩하여 DigitalWatch의 입력 값으로 전달한다.
 - req, ack : 4-cycle handshaking에 사용되는 와이어
 - inst[2:0] : Instruction 데이터를 전달받는 와이어

Programmable Watch: 4-Cycle Handshaking

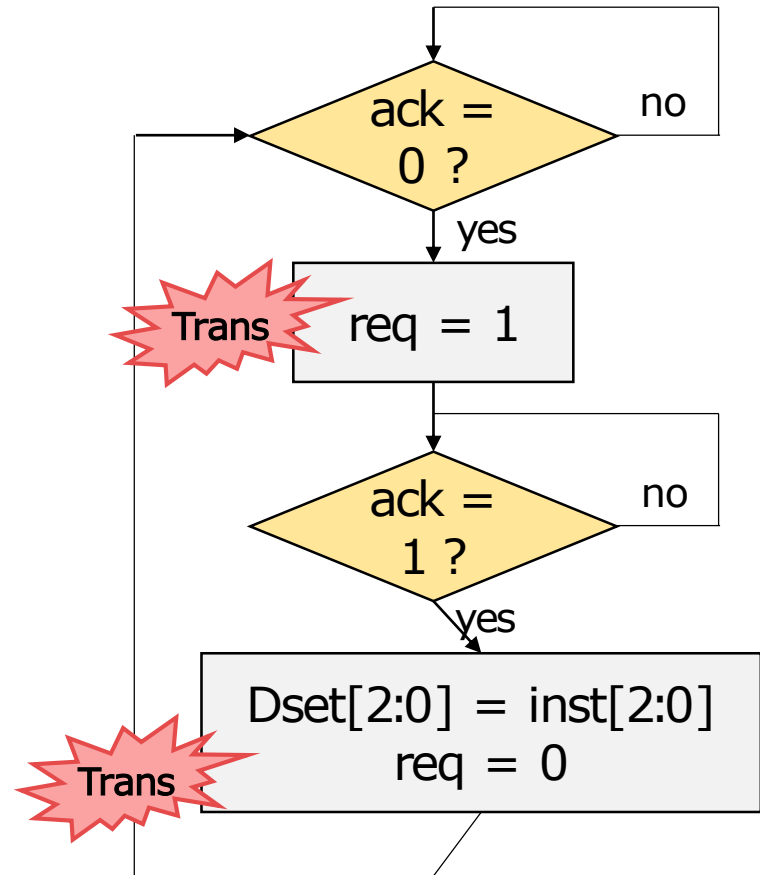


- Asynchronous한, 두 모듈이 올바른 데이터 통신을 위해 사용하는 프로토콜
- 데이터 통신을 위해서, 총 4번의 Transition이 발생함 (Request: 2번, Ack: 2번)

Programmable Watch: Handshaking 순서도



InstructionMemroy

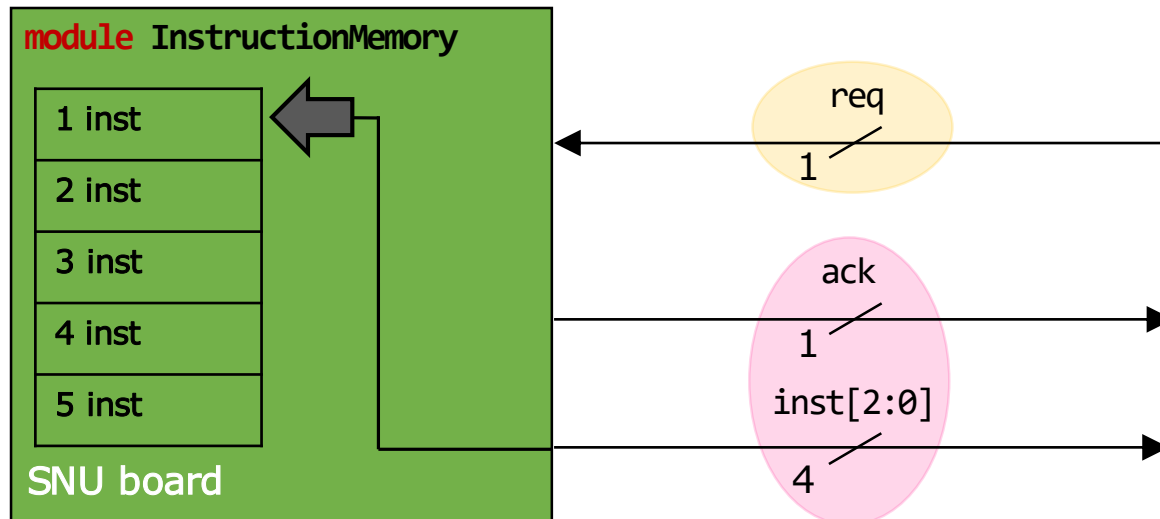


ProgrammableWatch

Programmable Watch: 모듈 (1)

■ InstructionMemory

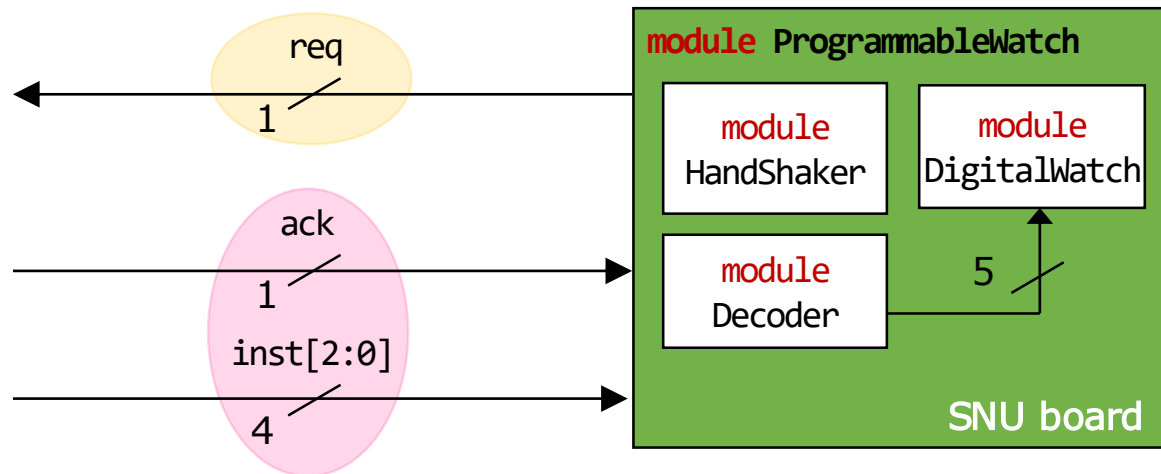
- 일련의 Instruction을 레지스터에 저장하고 있는 모듈
- req가 Posedge일 때, 데이터를 inst[2:0]로 넘겨주고, ack를 1로 set함
- req가 Negedge일 때, ack를 0으로 Clear하여 다음 통신을 기다림
- Black box 모듈로 구현할 필요 없음 (조교가 외부 보드에 미리 구현한 모듈)



Programmable Watch: 모듈 (2)

■ HandShaker

- req와 ack 와이어를 통해, 4-Cycle handshaking 통신을 진행하는 모듈
- ack가 Clear일 때, req를 1로 Set하여 데이터를 요청함
- inst[2:0]을 통해 데이터를 전달 받고, req을 0으로 Clear함



Programmable Watch: 모듈 (3)

■ Decoder

- inst[2: 0]을 통해, 읽어 들인 Instruction 데이터를 디코딩하여 DigitalWatch의 입력 값으로 바꾸는 모듈
- 총 7종류의 Instruction이 Binary encode로 전달됨
- 아래의 Binary encode는 인터페이스이며, 수정이 불가능하고 꼭 준수해야함

parameter inst_delay	= 3'b000	// Watch에 값을 입력하지 않음
parameter inst_reset	= 3'b001	// Watch의 reset을 수행
parameter inst_mode	= 3'b011	// Watch의 mode change를 수행
parameter inst_set	= 3'b100	// Watch의 set을 수행
parameter inst_op1	= 3'b101	// Watch의 op1을 수행
parameter inst_op2	= 3'b110	// Watch의 op2을 수행
parameter inst_stop	= 3'b111	// Handshaker가 통신을 멈춤

Programmable Watch: 보드 연결 방법

■ 두 Board 사이의 통신

- req, ack를 이용하여 4-Cycle handshaking 통신함
- 요청한 Instruction 데이터를 inst[2: 0]을 통해 전달받음
- UCF 파일에서 통신에 사용될 변수들을 I/O 포트와 매핑

```
NET "inst[0]" LOC = P110;  
NET "inst[1]" LOC = P111;  
NET "inst[2]" LOC = P112;  
NET "ack"     LOC = P113;  
NET "req"     LOC = P114;
```

(UCF 파일에서 I/O포트와의 매핑)



(케이블을 통해 두 보드를 연결한 모습)

추가 구현 힌트: Programmable Watch

■ 주의사항

- Programmable Watch에서는 사용자 입력을 처리하는 모듈을 제거 해야함
- 이처럼 기본구현에서 사용되는 모듈을 제거해야함으로, “**별도의 프로젝트**”에서 Programmable Watch를 구현 해야함
- 통신 프로토콜(4-Cycle handshaking)과 Binary encode를 꼭 준수할 것
- 동작 검증은 InstructionMemory 모듈이 있는 조교의 보드와 연결하여 진행



평가: 배점

■ 기본 구현

기능	기능	배점
시계 기능	1초 단위의 시/분/초 시계 기능	70 점
알람 기능	시계의 시간에 따라 알람을 설정/울리는 기능	70 점
스톱워치 기능	1/100초 단위의 스톱워치 기능	60 점
총 점		200 점

■ 추가 구현 (최대 추가 점수: 50점)

- Programmable Watch 구현 시, 50점 모두 부여
- 이외의 추가 구현은 난이도와 구현 개수에 따라 30점 내에서 차등 부여
- 추가 구현으로 얻은 점수는 **가산 점으로 총점 200점에 더하는 방식으로 평가됨**

구현의 예시

- 기본 구현과 추가 구현의 동작 영상

- <https://www.youtube.com/watch?v=JTQANSKILgY>

(기본 구현)

- <https://www.youtube.com/watch?v=UuR7pJNGfDk>

(추가 구현: Programmable Watch)

보고서 제출 및 평가 방법

- 제출 기한: 12월 18일 오후 6시 30분 전까지 ETL에 업로드
- 프로젝트 폴더와 보고서를 압축하여 제출
 - 보고서
 - 구현 사항, 모듈 설명, 추가 구현 설명/스펙 포함 (4장 내외)
예) 모듈 연결, state minimization 등 설계 사항
 - 프로젝트 폴더
 - Cleanup Project Files 하여 제출
- 동작 검사 기한
 - **12월 18일 화요일** 오후 6시 30분
 - 장소: 하드웨어 실습실
 - 제출 기한 내에 수시로 검사 가능 (단, 조교와 약속을 잡을 것)
 - **성적 기한 사정 상 Bonus Day 사용 불가**
 - **Copy 적발 시 F**

Help Desk

- Q & A Session

- 시간: 12월 13일 목요일 오후 6시 30분 ~ 8시
- 장소: 하드웨어 실습실
- **Q & A Session 중 출석 확인 없음 (12월 5일은 출석 확인)**

- Visiting Q & A Session

- 조교 e-mail (tas0218@davinci.snu.ac.kr)로 약속을 잡은 뒤
연구실에 직접 방문하여 조교에게 도움 요청
(사전 연락 없이 방문 시 조교가 없을 수도 있음)
- 장소: 302동 315-2호 임베디드 시스템 연구실 (하드웨어 실습실 맞은 편)