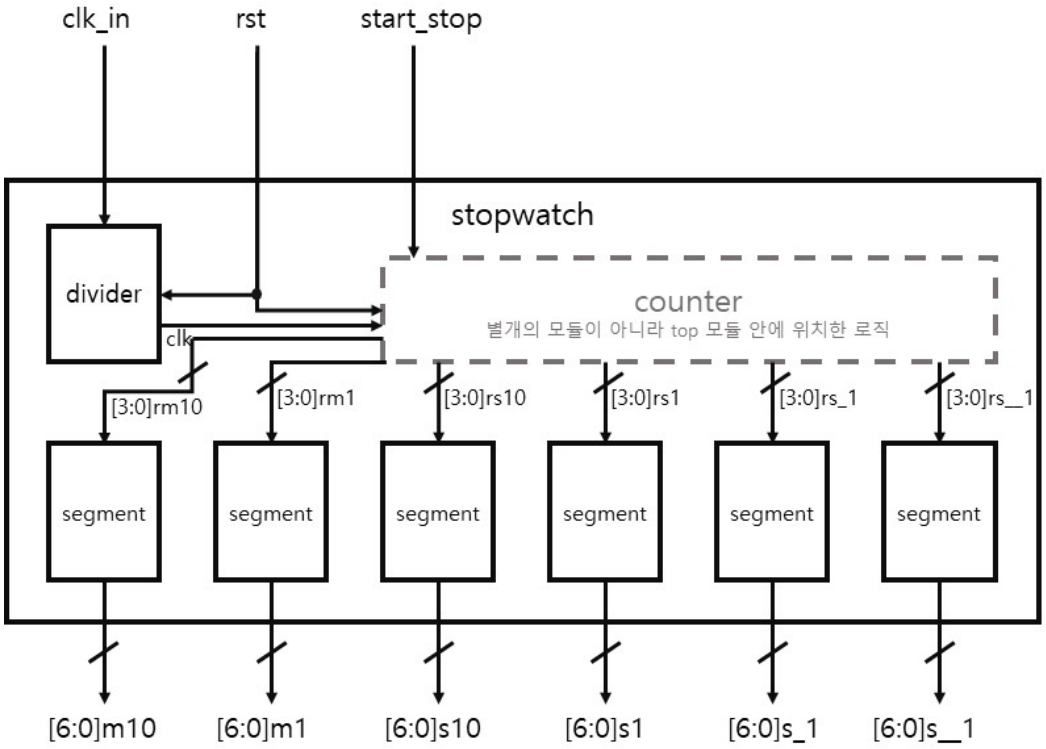
논리설계 보고서

1. 과제 3개 중 선택 항목
1) Digital Clock
2) Stopwatch

2. 설계 구조(각 설계 블럭도, 간단한 내용 기술)
1) Digital Clock

2) Stopwatch



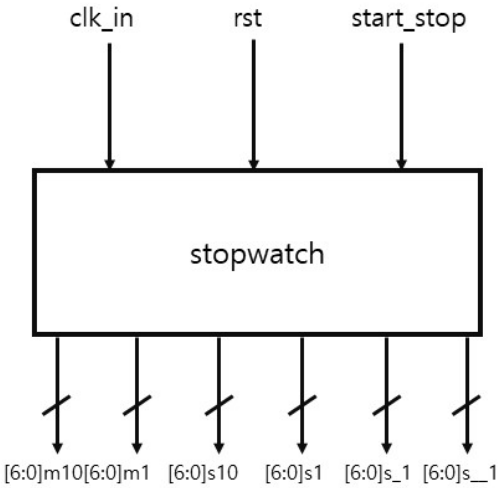
stopwatch : Top Module로, 하위 모듈을 연결하고, 여기에 더해 카운터를 구현하는 코드를 지니고 있다. 카운터를 별개 모듈로 분리하지 않은 이유는, 그렇게 되면 각 시간 단위에 대해 분주하고 카운트하는 모듈이 필요해 지나치게 모듈의 수가 많아지기 때문이다.

divider : 50MHz 입력을 0.01초(100Hz) 클락으로 분주하는 모듈이다.

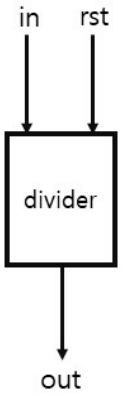
segment : 입력으로 받은 숫자들을 DE10 보드의 Seven Segment에 맞게 변환하는 모듈이다.

3. 시스템 동작 설명(각 블럭도, 내용 기술)

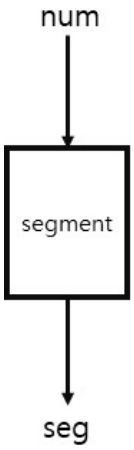
1) Digital Clock
2) Stopwatch



stopwatch
- 입력 : rst, clk_in, start_stop
- 출력 : [6:0]m10, [6:0]m1, [6:0]s10, [6:0]s1, [6:0]s_1, [6:0]s__1
- Top Module이기 때문에 하위 모듈들을 연결하고 있다. 또한, 내부에 분주기와 카운터가 구현되어 있는데, 각 분주기는 divider가 분주한 100Hz 클락을 카운터에 맞는 클락으로 분주하고, 각 카운터는 해당 클락에 맞게 카운팅을 하고 있다. 전술한 것처럼, 지나치게 모듈이 많아지는 것을 막기 위해 카운터를 분리하지 않고 구현하였다.



divider
- 입력 : rst, in
- 출력 : out
- 입력받은 50MHz 클락(in)을 100Hz 클락(out)으로 분주하는 분주기이다. rst는 리셋 신호이다. 스톱워치를 구현하는 데에 들어간 모든 분주기는 if문으로 카운트를 0으로 되돌리는 대신 모듈러 연산을 사용하였다. Non-Blocking으로 if문을 작성했을 때는 조건문의 판정이 덧셈과 동시에 일어나기 때문에 카운트할 값에서 1을 뺐지만, 모듈러 연산을 활용하면 덧셈 이후에 나누게 되므로 1을 뺄 필요가 없다.



segment
- 입력 : [3:0]num
- 출력 : [6:0]seg
- 입력받은 숫자(num)를 그 숫자에 상응하는 Seven Segment 값(seg)로 변환하여 출력한다. num의 최대 범위는 0에서 9의 숫자이므로 4비트이다.

4. 설계 코드 및 결과(코드, 시뮬레이션, 사진 등)

1) Digital Clock

2) Stopwatch

코드

```
// stopwatch.v (top module)
module stopwatch(rst, clk_in, start_stop, m10, m1, s10, s1, s_1, s__1);
input rst, clk_in, start_stop;
output [6:0] m10, m1, s10, s1, s_1, s__1;
reg [3:0] rm10, rm1, rs10, rs1, rs_1, rs__1;
reg [3:0] tm10, tm1, ts10, ts1, ts_1, ts__1;
reg [15:0] p;
wire clk;

segment u0(rm10, m10);
segment u1(rm1, m1);
segment u2(rs10, s10);
segment u3(rs1, s1);
segment u4(rs_1, s_1);
segment u5(rs__1, s__1);

divider u6(rst, clk_in, clk);
```

```
always@(posedge clk, negedge rst) begin
  if (!rst) begin
    rm10 <= 7'b000_0000;
    rm1 <= 7'b000_0000;
    rs10 <= 7'b000_0000;
    rs1 <= 7'b000_0000;
    rs_1 <= 7'b000_0000;
    rs__1 <= 7'b000_0000;
    tm10 <= 4'b0000;
    tm1 <= 4'b0000;
    ts10 <= 4'b0000;
    ts1 <= 4'b0000;
    ts_1 <= 4'b0000;
    ts__1 <= 4'b0000;
    p <= 15'b000_0000_0000_0000;
  end
  else if (start_stop) begin
    ts__1 <= (ts__1+1)%10;
    rs__1 <= ts__1;
    p <= (p+1)%60000;
    if (!(p%10)) begin
      ts_1 <= (ts_1+1)%10;
      rs_1 <= ts_1;
    end
    if (!(p%100)) begin
      ts1 <= (ts1+1)%10;
      rs1 <= ts1;
    end
    if (!(p%1000)) begin
      ts10 <= (ts10+1)%6;
      rs10 <= ts10;
    end
    if (!(p%6000)) begin
      tm1 <= (tm1+1)%10;
      rm1 <= tm1;
    end
    if (!p) begin
      tm10 <= (tm10+1)%6;
      rm10 <= tm10;
```

```
    end
  end
end

endmodule
```

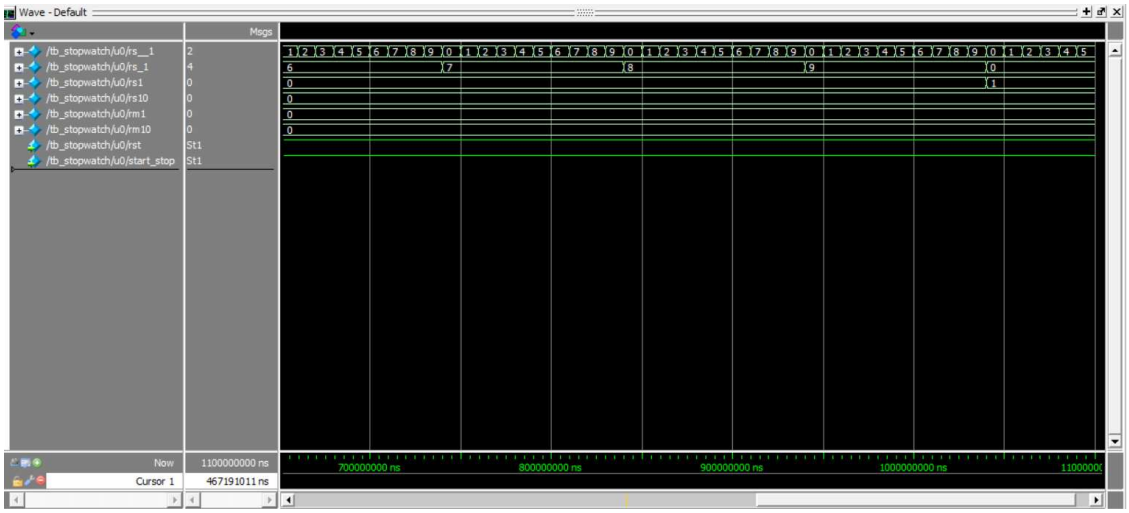
```
// divider.v
module divider(rst, in, out);
input in, rst;
output reg out;
reg [17:0] cnt;

always@(posedge in, negedge rst) begin
  if (!rst) begin
    out <= 0;
    cnt <= 0;
  end
  else begin
    cnt <= (cnt+1)%250000;
    if (!cnt) out <= !out;
  end
end

endmodule
```

```
// segment.v
module segment(num, seg);
  input [3:0] num;
  output reg [6:0] seg;

  always@(num) begin
    case(num)
      4'b0000 : seg = 7'b1000000;
      4'b0001 : seg = 7'b1111001;
      4'b0010 : seg = 7'b0100100;
      4'b0011 : seg = 7'b0110000;
      4'b0100 : seg = 7'b0011001;
      4'b0101 : seg = 7'b0010010;
      4'b0110 : seg = 7'b0000010;
      4'b0111 : seg = 7'b1111000;
```

```
        4'b1000 : seg = 7'b0000000;
        4'b1001 : seg = 7'b0010000;
      endcase
   end

endmodule
```

```
// tb_stopwatch.v
// this is testbench, only used in simulation
`timescale 1ns/1ns
module tb_stopwatch();
reg rst, clk, start_stop;
wire [6:0] m10, m1, s10, s1, s_1, s__1;

stopwatch u0(rst, clk, start_stop, m10, m1, s10, s1, s_1, s__1);

initial begin
   rst <= 0;
   clk <= 0;
   start_stop <= 0;

   #15_000_000; rst <= 1;
   #25_000_000; start_stop <= 1;
   //forever #5_000_000 clk <= !clk;

//   #15; rst <= 1;
//   #35; start_stop <= 1;
//   forever #5 clk <= !clk;

      forever #10 clk <= !clk;
end

endmodule
```

시뮬레이션



영상(별도 첨부)
- 영상 1 : 정상 작동 확인
- 영상 2 : 분 단위 작동을 위한 분주 시간 가속, 정지 및 리셋 기능 확인

5. 결론 및 검토

5.1 목표대비 달성도

1) Digital Clock

2) Stopwatch

목표(리셋, 시작/정지 기능을 가진 10분, 1분, 10초, 1초, .1초, .01초 단위의 스톱워치) 전체를 완전히 구현해내었다.

5.2 Review

1) Digital Clock

2) Stopwatch

분주기 내부 카운터 구현에 if문을 사용하는 것이 번거로웠는데, 모듈러 연산을 통해서 조금 더 편하게 짤 수 있었던 것 같다. 각 세그먼트 시간 단위에 필요한 카운터가 분주하는 단위가 각각 달라서(이 프로젝트의 예처럼 동기식으로 구현하면 모두 다르고, 비동기식으로 구현하더라도 6분주를 해야 하는 경우와 10분주를 해야 하는 경우가 있음) 별개 모듈로 분리하기가 번거로워 Top Module 내부에서 전체 시간 단위를 각각 카운트하도록 구현하였다. 분주하는 단위를 매개변수로 넣을 수 있도록 구현하면 편리할 것 같다는 생각이 들었다. 찾아보니 각 모듈에 매개변수를 주어 인스턴스마다 다른 값을 넣는 게 가능하다고 되어 있었는데, 수업의 범위를 벗어나기도 해서 해당 방법은 사용하지 않고 각각 구현하였다.

6. 팀원 역할

학번	이름	역할
		Stopwatch 설계 및 보고서 작성
		Digital Clock 설계 및 보고서 작성