COL215P Assignment 1

Stopwatch

Ujjwal Mehta | Entry No. - 2020CS10401

Somaditya Singh | Entry No. - 2020CS10389

In this assignment we made a stopwatch with a start/pause as well as reset button which can run on basys3 FPGA board. In order to build this stopwatch we created some independent modules which were then combined to creating stopwatch. The description of these modules as well as their code explanation is given in the next section.

Main Logic Of Working Code

In the working of code the parameters taken from the basys3 board are `clk, btnC, btnU, btnL` (explained in detail in the debouncing module) and using the buttons we are starting/pausing or reseting our stopwatch and then using the clock we update our select lines(details in timer module) which are then fed to the multiplexer to display the apporpriate digit at the appropriate anode.

Seven Segment Decoder

This module is created in order to display a digit from $0-9$ on the seven segment cathode of the basys3 board and the logic for this module is present in seven_segment.vhd and in order to create this combinationally we used the Karnaugh Map diagram on the truth table created by seeing the cathode positioning to deduce the combinational logic. It takes `unsigned(3 downto 0)` as the digit input and outputs the `std_logic_vector(6 downto 0)` which corresponds to the seven segment cathode display. The minimized logic is as described below:

```
seg(0) = (not(a) and not(b) and not(c) or (b and c and d)

seg(1) = (not(b) and c) or (c and d) or (not(a) and not(b) and d)

seg(2) = d or (b and not(c))

seg(3) = (b and not(c) and not(d)) or (b and c and d) or (not(a) and not(b) and
```

```
not(c) and d)

seg(4) = not(b) and a and not(d)

seg(5) = (b and not(c) and d) or (b and c and not(d))

seg(6) = (b and not(c) and not(d)) or (not(a) and not(b) and not(c) and d)
```

Here `a, b, c, d` are the bits of the input digit, decreasing in significance from left to right.

Timer

This module is created in order to change the refresh the digits that are displayed on the stopwatch and for that we created a modulo 10^5 counter i.e. on every 1 ms (10^5 clock cycles), we change the select lines and the anode to be displayed which means that the digit period is 1 ms, and the refresh period is 4 ms(for all the 4 digits). The logic of this module is present in timer.vhd file and for making such a modulo 10^5 counter we are taking `clock` as an input and we defined an internal signal counter which is increased at every `rising_edge` of clock and is reset when its value reaches 10^5 meanwhile shifting our anode to the next digit to be displayed.

Modulo N Counter

These modules logic is present in the module1.vhd, module2.vhd, module3.vhd, module4.vhd and here these modules are used to increment the digits of our stopwatch where module1.vhd is incrementing the minutes digit at every minute ($6 * 10^9$ clock cycles), module2.vhd is incrementing the tenth of seconds digit at every 10 seconds(10^9 clock cycles), module3.vhd is incrementing the seconds digit at every 1 second(10^8 clock cycles) and module4.vhd is incrementing the tenth of a second digit at every 0.1 seconds (10^7 clock cycles). The logic used to create such counters is similar in all the counter modules where we define an internal signal counter which is increased at every `rising_edge` of input `clock` (period of which is 10 nanoseconds) and then it gets reset when its value reaches the corresponding required counter value though this whole thing will work when the `enable_watch` (taken as an input to the modulo counters) is 1(else there will be no counter increment).

Multiplexer

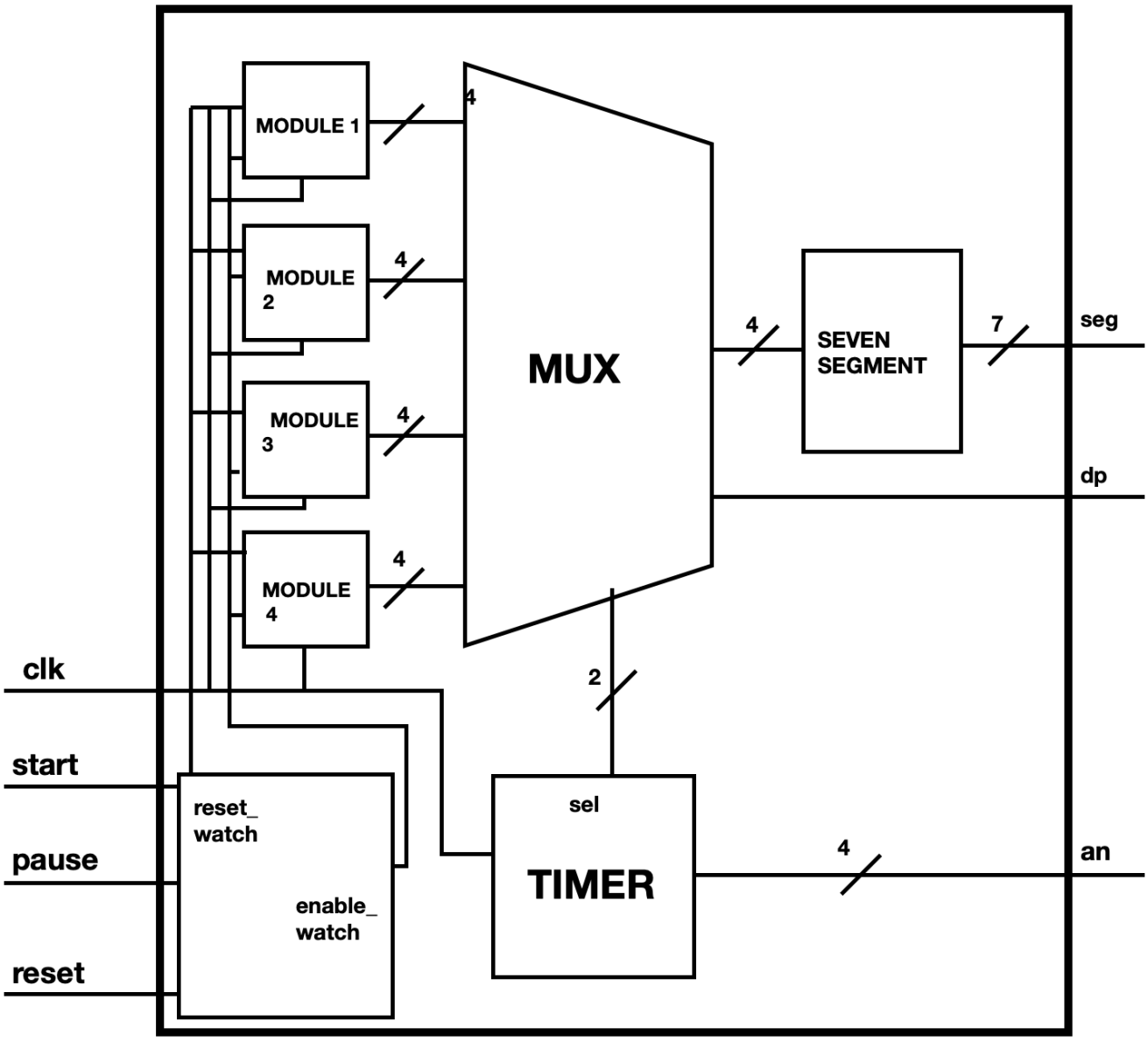
The logic of this multiplexer module is present in multiplexer.vhd file and we have designed this multiplexer using its combinational circuit logic where the input ports are `digit1, digit2, digit3, digit4` which are `unsigned(3 downto 0)` and select lines `sel` and

this will output the `digit_out` (and `dp` also) which is one the digit between `digit1, digit2, digit3, digit4` using `sel` value.

Debouncing Module for Push Button

The logic of this module is used to connect all the above modules along with incorporating the start/pause and reset buttons (the debouncing module) is present in the StopWatch.vhd file and the main logic used to connect the start/pause and reset buttons is that we are taking an enable_watch and reset_watch from the basys3 board buttons whose corresponding bit will change to 1 when that button is pressed, hence we are again maintaining a counter to slow the clock(check for enable_watch after every 10^5 clock cycles) so that we can detect the corresponding enable_watch or reset_watch is set to 1 and then with the help of muliplexer we display the corresponding digit at the corresponding position. The input ports of this module are `clk` , `btnC` (for start/ continue), `btnU` (for stop) , `btnL` (for reset) and this in turn will output us the `an, seg, dp` (anode, seven segment and decimal point).

Block Diagram for our design


MODULE 1
MODULE 2
MODULE 3
MODULE 4
4
4
4
4
MUX
4
SEVEN SEGMENT
7
seg
dp
clk
start
pause
reset
reset_ watch
enable_ watch
2
sel
TIMER
4
an