# COL215P Assignment 1

Submitted by:

Chinmay Mittal 2020CS10336

Dhruv Tyagi 2020CS10341

Department of Computer Science and Engineering

Indian Institute of Technology, Delhi

# 1 Design I/O

## 1.1 Input

We can consider our design as having 3 inputs: start/continue button, pause button and reset button. We also have a $10^8$ Hz clock at our disposal.

## 1.2 Output

The output is 7 bits for cathode, 4 bits for anode and one bit for the decimal point. We have also added one additional output to a LED light that is ON when the clock is running.

# 2 Component Design

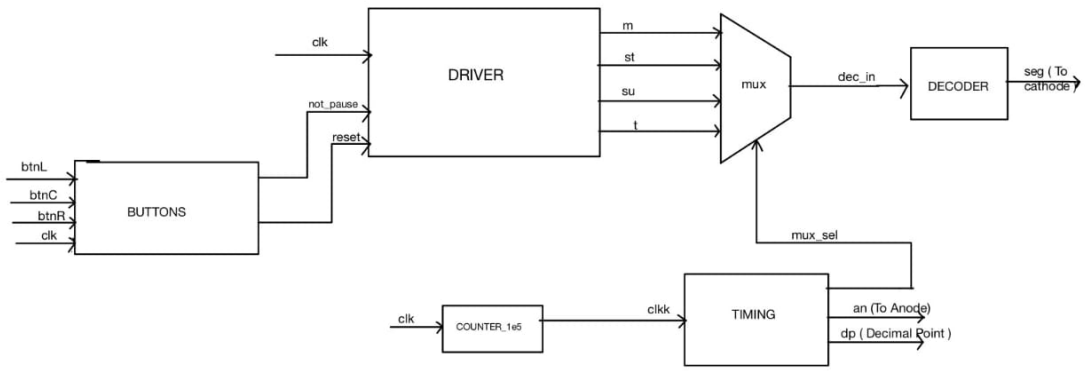


Figure 1: Block diagram displaying Components in glue logic

The main modules in our stop watch are the following:

1. Buttons
2. Driver
3. Timing circuit

4. Multiplexer
5. Decoder
6. counter_1e5

## 2.1 Buttons

We have a component called 'buttons' that is used to receive the input from the 3 available buttons. Each signal is then sent to a debounce module (explained below) to clean the signal. These buttons are used to set up the status of 2 flags, namely not_pause and reset. These flags are sent to the driver module to control the current clock value and whether to increment it or not.

### 2.1.1 Debounce Module

The debounce module is a component of the buttons module described above. It comprises of 2 D Flip Flops and cleans the signal from the button directly by using a clock of $10^5$ Hz (slow_clk) for the D Flip Flops. The input from the button is directly sent to the first Flip Flop. The output of the first flip flop $S_1$ is then sent to the second flip flop. Output from the second flip flop is $S_2$.

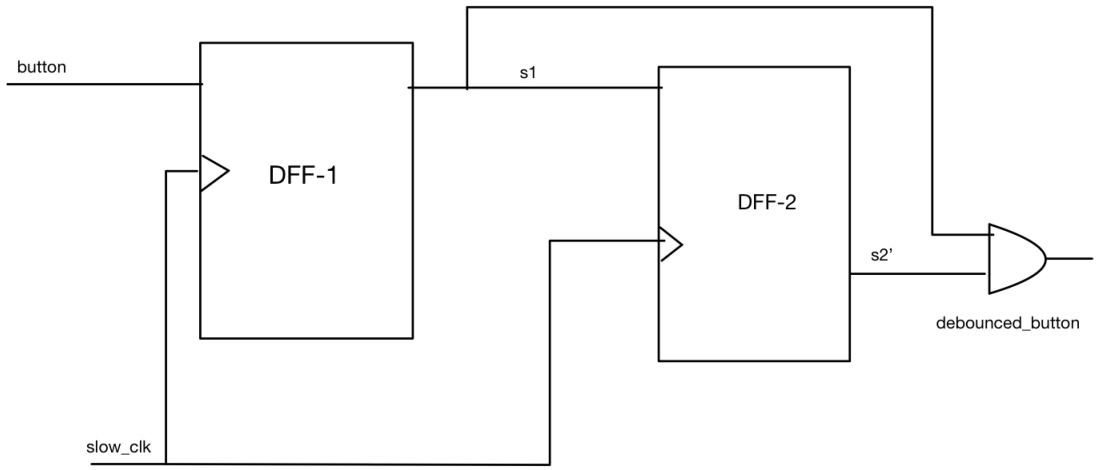


Figure 2: Debounce Module

The final output of the module is $S_1\overline{S_2}$. This can be seen from the waveform diagram.

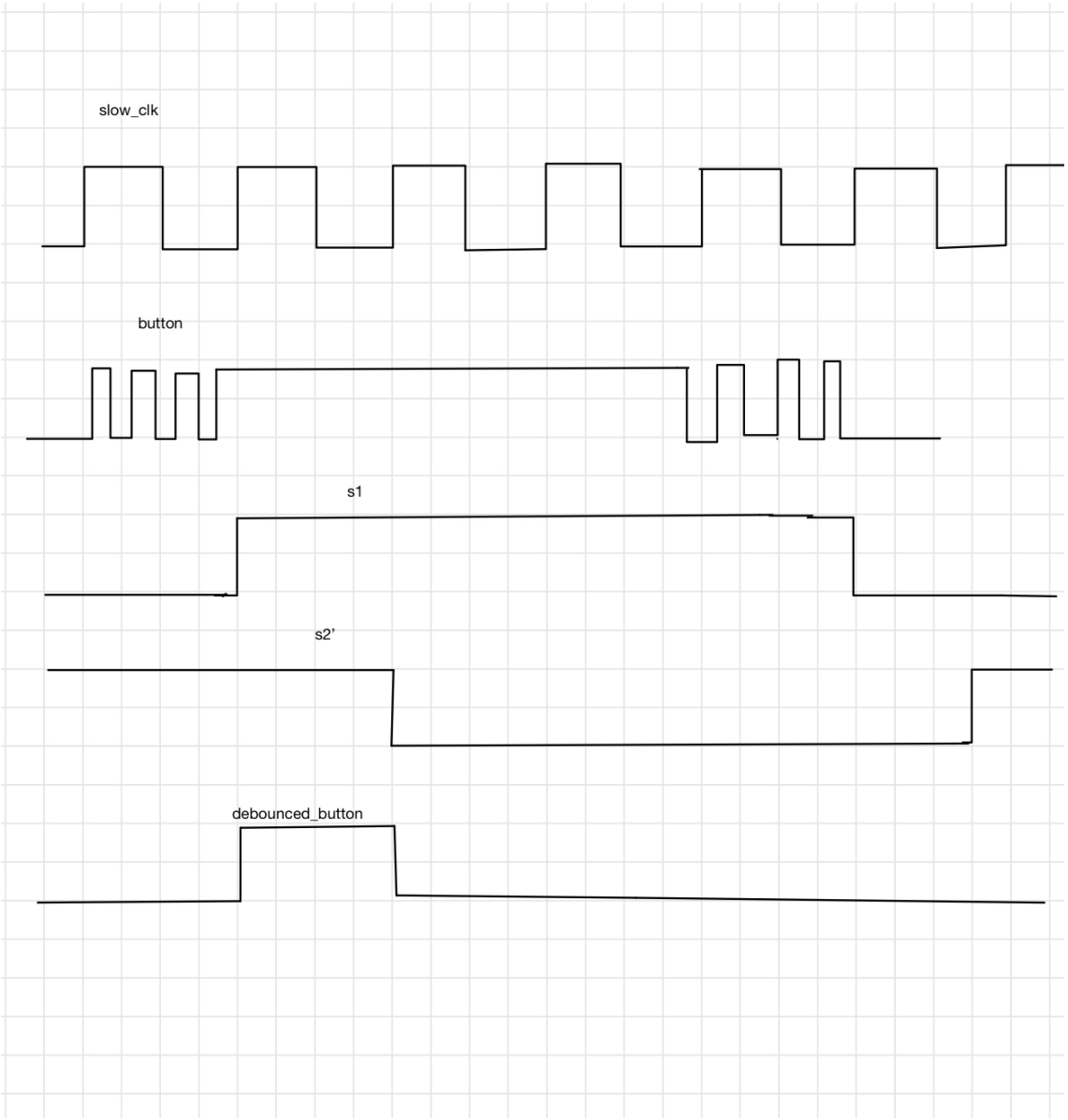


Figure 3: Waveform for debounce module

## 2.2 Driver

The driver module takes as input the two signals described above (not_pause and reset) and also system clock as input.
It consist of 4 counters, one for each digit to be displayed on the stopwatch. There is one additional counter.
*counter*_1$e7$ : This maintains a count which increments at each system clock edge when the count reaches $1e7$ it resets. We maintain a positive output for half of this duration and a negative output for the remaining duration, which creates a clock with a positive edge after every $1e7 * 1e-8 = 1e-1$ seconds. This is directly sent to the unit place counter of the stopwatch (that increment in one tenth of a second).
Each of these 4 counters are interlinked and the output ( becomes one when the digit resets and is zero otherwise) from the smaller significant digit is used as input ( rising_edge of the input is used to update the digit ) to the more significant digit. The driver module outputs these 4 digits as $4-$ bit vectors which are sent to the multiplexer unit 'mux'. If not_pause is zero than the digits in all counters do not update. If reset becomes one then all digits are reset to zero.

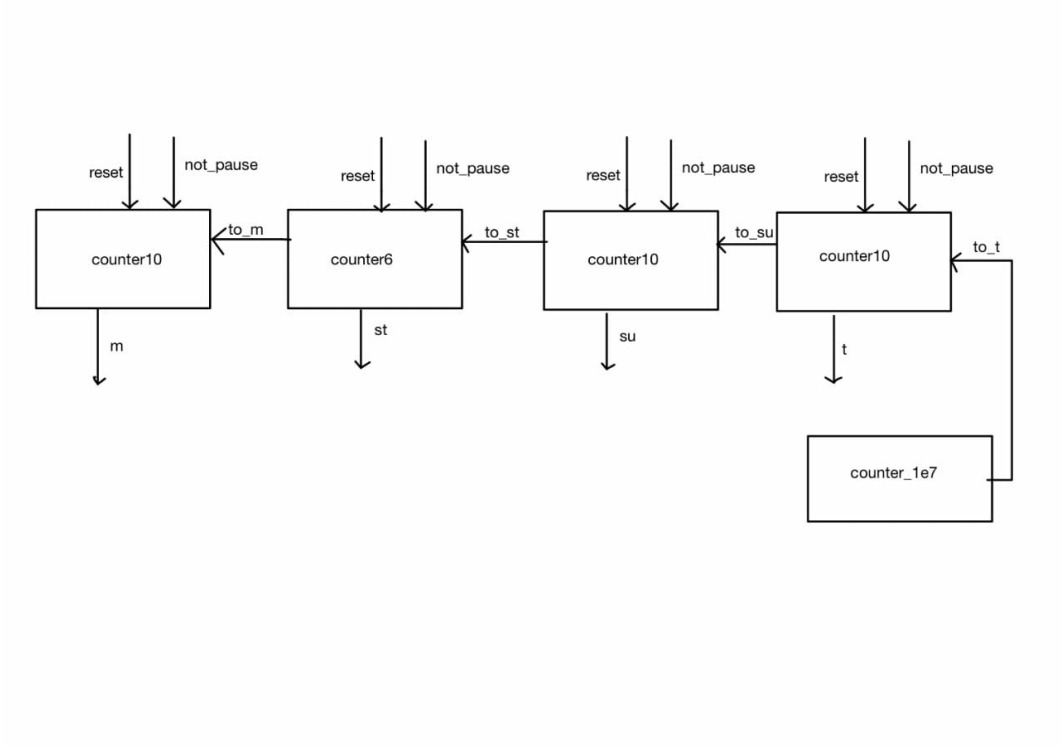


Figure 4: Driver

### 2.3 Timing Circuit and Multiplexer

These components work as mentioned in the assignment. The multiplexer decides which of the four digits is to be displayed as per the timing circuit. The timing circuit alternates among the 4 bits at a frequency of $10^3$ Hz. The timing circuit also decides which decimal point is to be switched on. The rapid cycling of the timing circuit ensures that the digits refresh fast enough for the human eye so that all digits and decimal points appear displayed at once.

### 2.4 Decoder

The decoder uses the 4 bit input corresponding to a digit and outputs the values of the cathodes accordingly. The minimized logic of the decoder is as follows:

```
A <= a'b'c'd + bc'd'
B <=  bc'd + bcd'
C <= b'cd'
D <= a'b'c'd + bc'd' + bcd
E <= d + bc'
F <=  a'b'd + b'c + cd
G <=  a'b'c' + bcd
```

Figure 5: Minimized Logic for Decoder

### 2.5 counter_1e5

This module implements a counter which increments at each rising edge of the system clock. When the value of the counter reaches $1e5$ we reset the count. During half this duration we maintain the output to be positive and during the other half we maintain the output to be negative which creates a clock with a positive edge every $1e-8*1e5 = 1e-3$ i.e of frequency $1e3$ Hz. This is sent to the timing circuit to control the refresh of the display.

## 3 Working

The following picture shows the start/continue, pause and reset buttons in the board. The working is as expected: push the start button to start. The clock starts from the time displayed on it just before the start button was pushed. Pause button pauses the clock so that the clock continues to display the time at which the button was pressed. Reset button sets all the digits to 0 and now the clock is in a paused state until the start button is pressed.

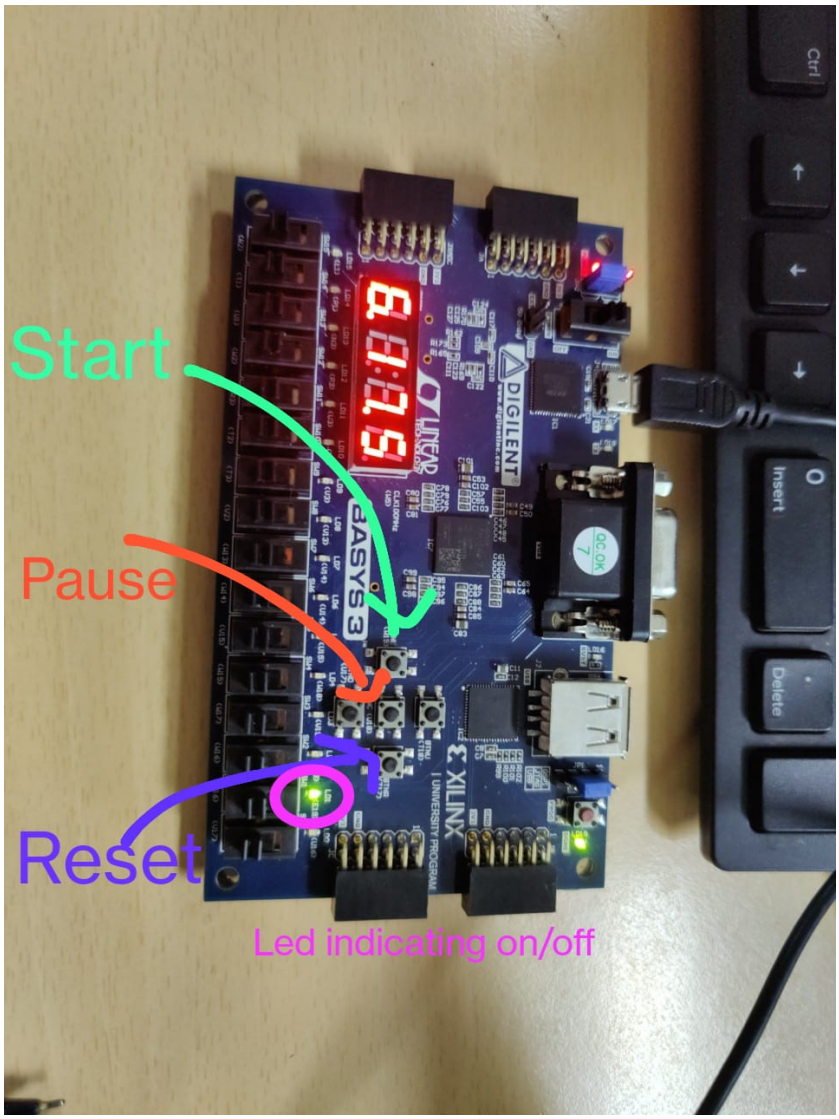


Figure 6: Stopwatch on basys board