COMPUTER SYSTEMS ASSIGNMENT 1 – DIGITAL STOPWATCH

Name: Pulkit Pannu
Student ID: 104093910
Unit Code: COS10004
Lab Session: Thursday 8:30am – 10:30am
Lab Tutor: Kafil Uddin

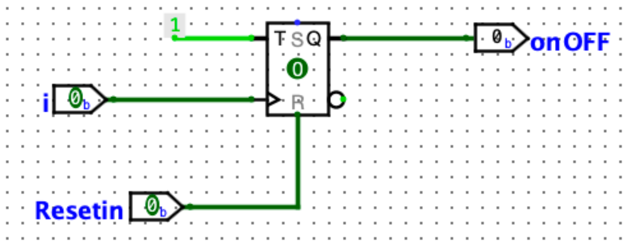
INTRODUCTION

The aim of this project was to "implement the functionality for a simple stopwatch interface" using the Logisim Evolution software. The assignment required us to complete as many of the seven stages explained in the assignment and implement those stages in our circuit to make our stopwatch work. I have completed all of the stages of this assignment with as precise functionality of the stopwatch as I was able to comprehend from the assignment description.

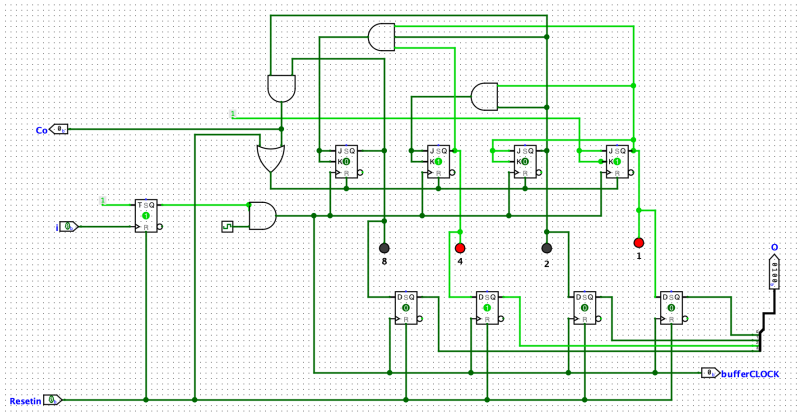
To achieve this, I have used subcircuits to implement specific components in the main interface of my circuits. My main circuit consists of subcircuits for counters, stacks, an encoder and a multiplexer, all made from logic gates and flip flops to implement the functionality of my stopwatch. I have named my subcircuits according to the stage they represent.

DESIGN OUTINE

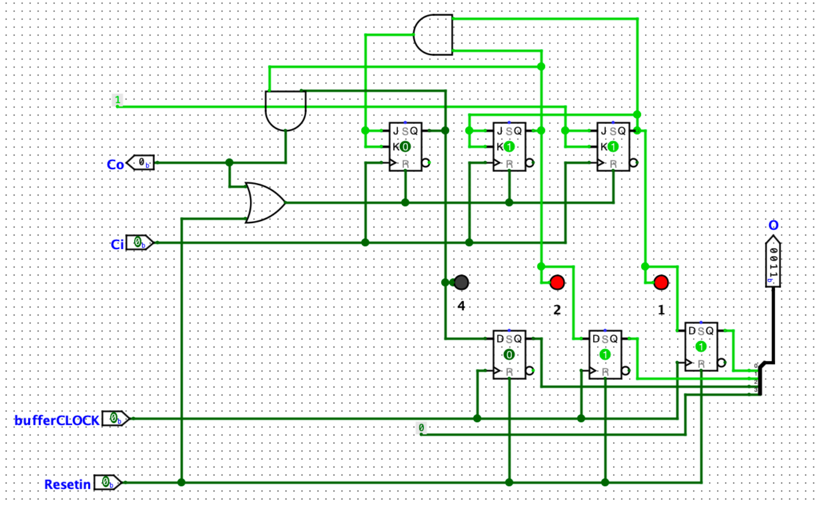
Stage 1 – The aim for this stage was to make a circuit that turns on the "clock started" LED when the "Start/Stop" button was pressed. To achieve this, I used a toggle flip flop with the button pulse as its clock input and a constant with high value as the T input. The Q output was connected to the LED and the reset input R was connected to the reset button pulse. This allowed us to toggle between the on and off state with every button pulse.



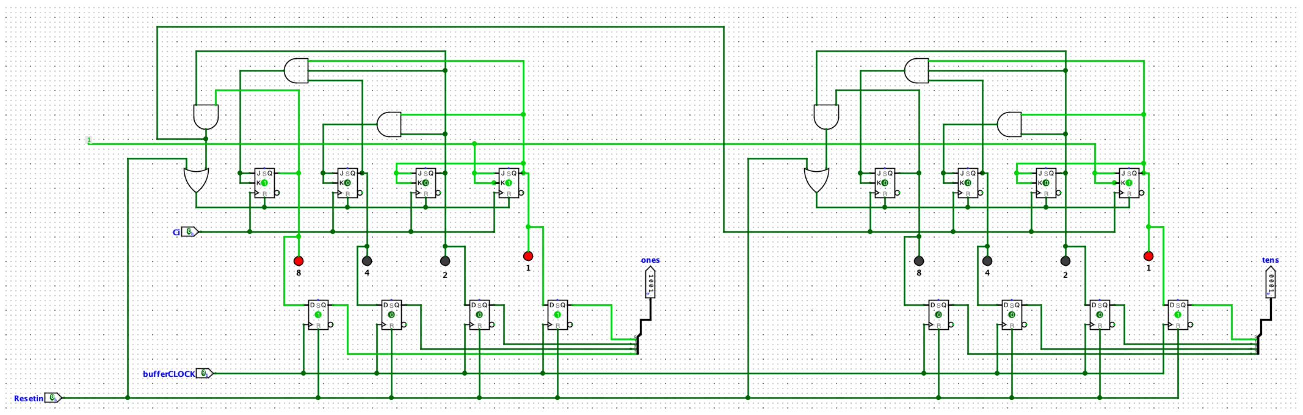
Stage 2 – The aim for this stage was to implement a single digit 'seconds' display that counts up to 9 and then resets back to 0 and so on when the "Start/Stop" button is pressed. To achieve this, I used a four bit, common clock, J-K flip flop counter with D flip flops used as buffers to prevent the illegal state from showing up on the display when the counter resets back to zero after 9.



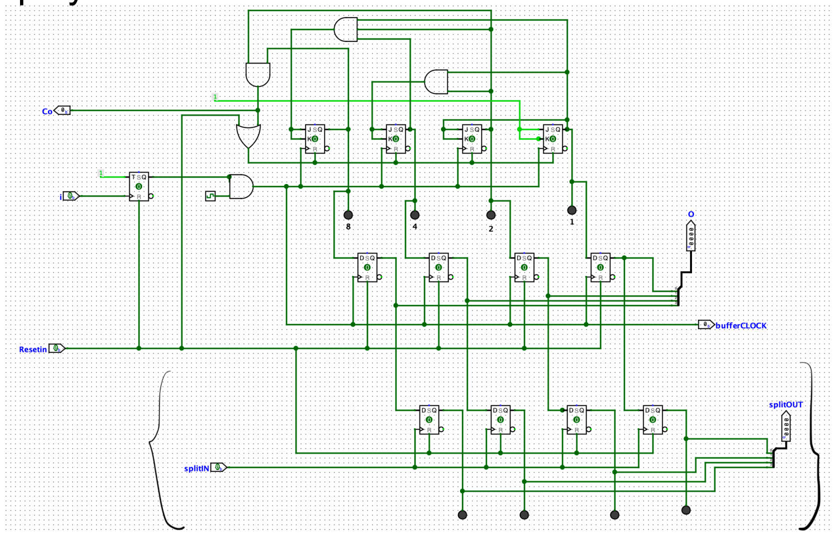
Stage 3 – The aim for this stage was to implement a full two digit ‘seconds’ display that counts up to 59 and then resets back to 00. To achieve this, I used a three bit common clock counter, resetting at 6, with the clock input for the J-K flip flops as the reset pulse of stage 2 because we only want an input in the tens place of the seconds display only when the ones place resets from 0 to 9 and back to 0. This was the critical insight for this assignment as one needs to understand that the clock inputs of flip flop don’t always require actual clocks as their input, it rather depends on the situation as to how many “clock pulses” is required for the correct implementation of our circuit.



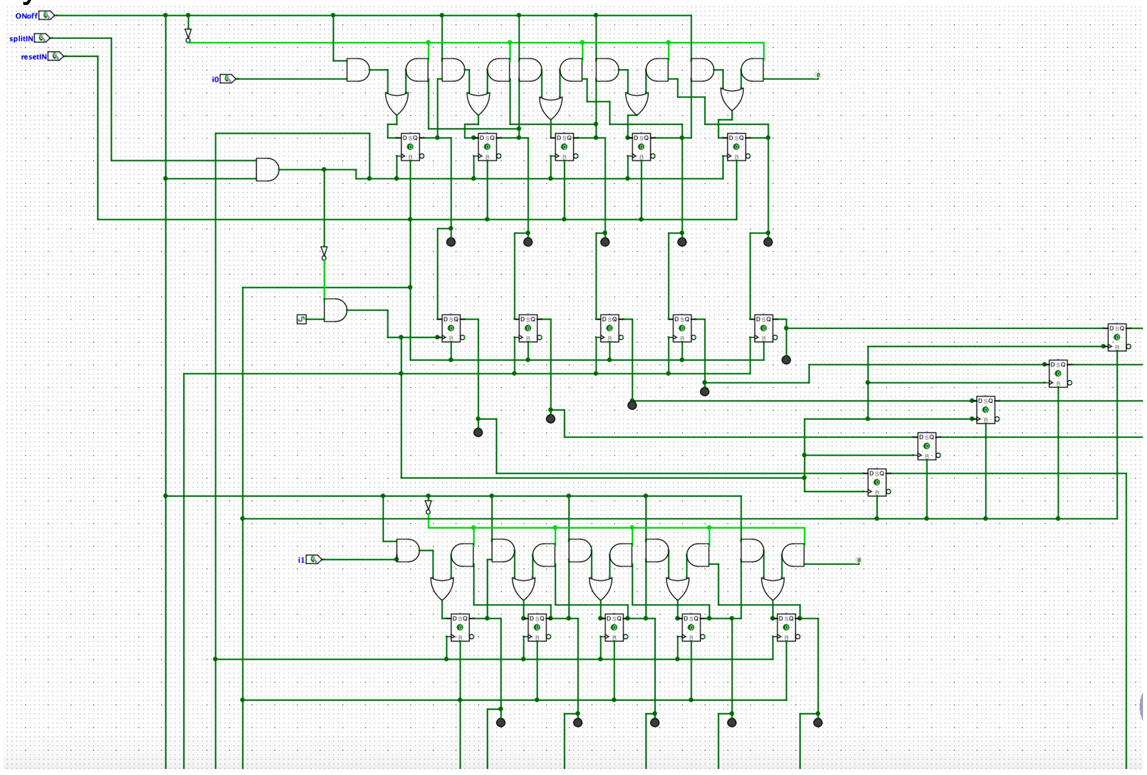
Stage 4 – This stage requires us to implement the full two digit ‘minutes’ display that counts up to 99 and then wraps back to 00. This stage uses the combined concept of the last two stages. I used two 4 bit counters to display the ones and the tens position of the minutes display. The ones place counter takes the clock input from the reset pin of the stage 3 counter, whereas, the tens place counter takes the clock input from the reset pin of the ones place counter. Moreover, for every counter I have used, the buffer flip flops takes in the same clock input from stage 2 to avoid an error in the stopwatch, i.e. when the stopwatch is paused when the ones place of the seconds display shows 9.



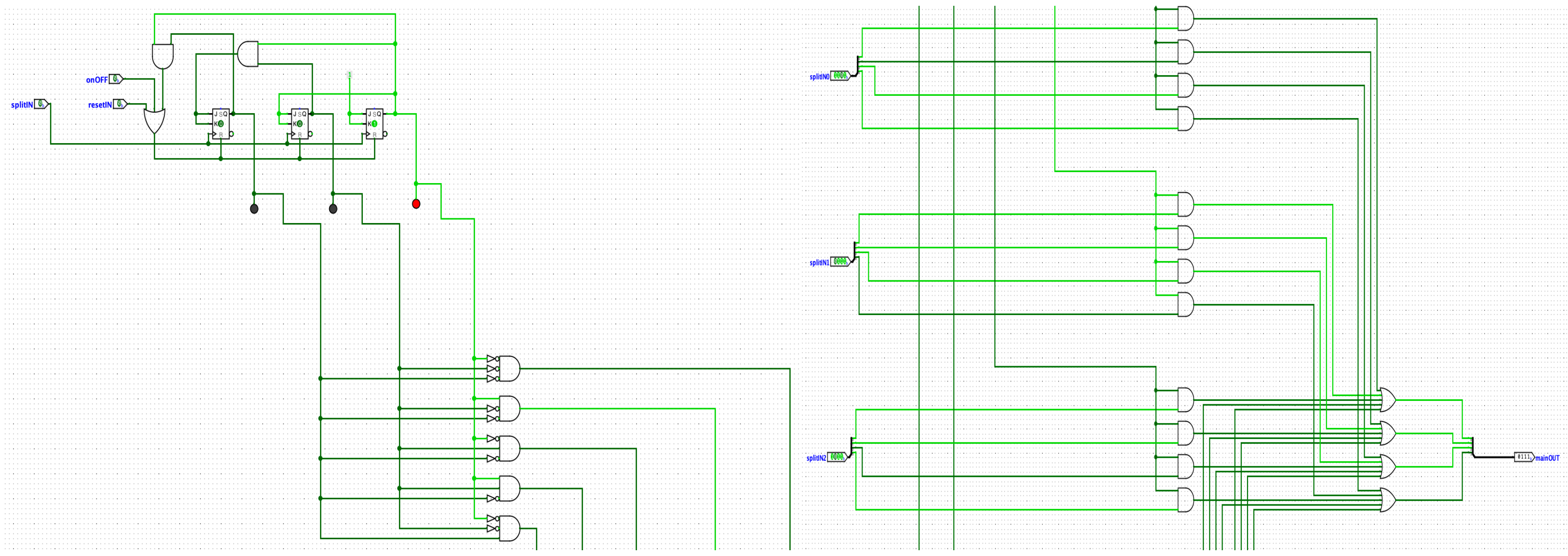
Stage 5 – The aim for this stage was to implement the split time display to store the current time displayed on the elapsed time display when the split button is pressed. To achieve this, I used D flip flops in my seconds and minutes counters to create a parallel circuit with the split button pulse as the clock input to store and display the current time on the split time display.



Stage 6 – For this stage, we have to record and store five different split times and ensure the most recent split time is displayed on the spilt display at all times. Moreover, when the clock is in the stop state, we need to loop through the split times from the most recent to the earliest by clicking the split button. To achieve this, I used a stack to store and represent the split times. To be precise, I used a 5 bit deep and four stacks wide stack, taking and storing a four bit number at time when the split button is pressed. We use buffer D flip flops to prevent any illegal states from showing up on the hex display.



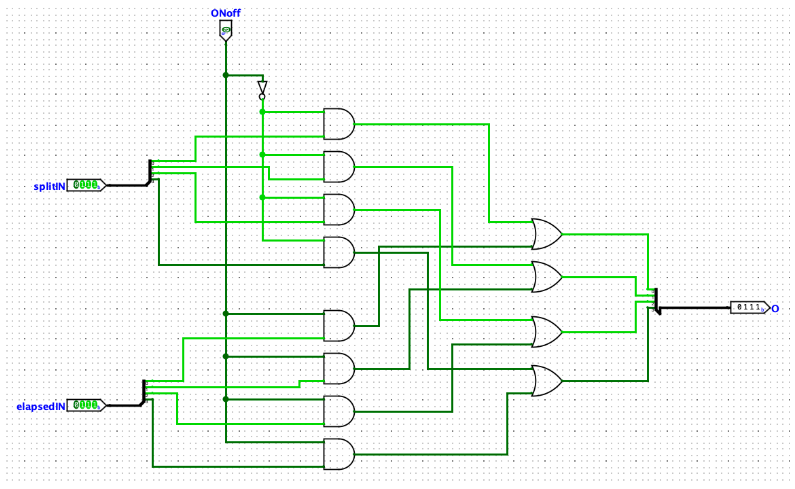
All the stored split times are connected to splitters and stored in output pins. I use a self-built encoder to choose from these output pins, which pin value I need to display and under what circumstances.



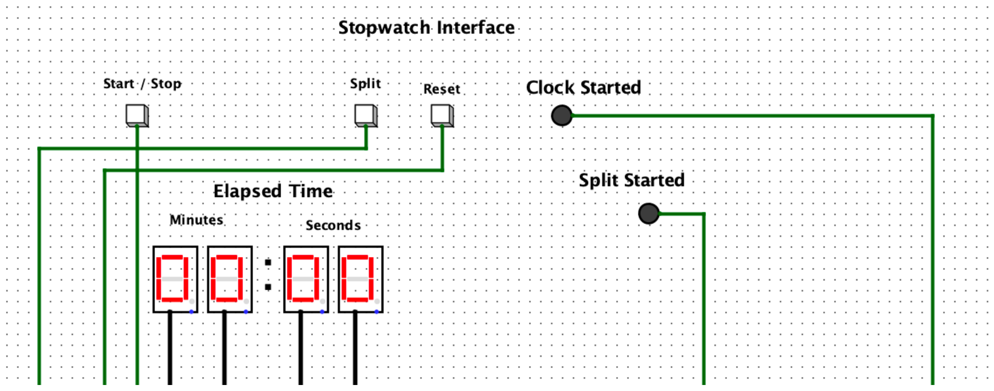
It is not an actual encoder but it does the work of an encoder in the sense that it chooses from multiple inputs with the help of selection inputs/pins and provides a single output. In this case, I had to choose among 5 split times to represent on the split time display. Basically, I had to loop through these split times, so I needed to loop through my selection inputs as well. Therefore, for the selection of these 5 split time values I used 3 bit counter, with the split button pulse as the clock inputs for the D flip flops, counting from 0 to 4 and then looping back to 0. I then arranged their outputs such that it allows only one split value to go through at a time. For this I used AND gates, OR gates and NOT gates.

I also used an “onOFF” pin for the reset input of the D flip flops because I was required to store the split times in the right order. With the use of this pin, I was able to achieve that when you have 5 different split times stored and you loop through them and leave a random split time on the split display and then you start the clock and store another split time, that split time is stored in the correct descending order, i.e. from the most recent split time to the earliest split time.

Another circuit that I used for this stage is a selector circuit that allows us to select which input pin to choose upon the condition. This is a kind of multiplexer. I used this circuit to show the most recent split time on the split time display when the stopwatch is in the on state and the encoder output value when the stopwatch is in the off state. I used this circuit as I was required to "Ensure the most recent split time is displayed on the "Split Time" display at all time".

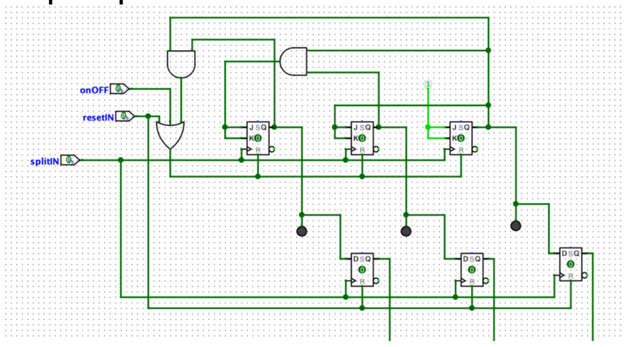


Stage 7 – For the final stage, we were supposed to display all times using only the 'elapsed time' display. This means that we have to show the five stored split times when the stopwatch is in the stop state and the split time is clicked, i.e. when the split time is not clicked, the display should, by default, show the elapsed time and when the split time is clicked, an LED , different from the 'clock started' LED, lights up indicating that the display is now showing the split time and the split button allows the user to loop through the split times. If the stopwatch is started again, then the display switches back to elapsed time and continue from where it was left before.

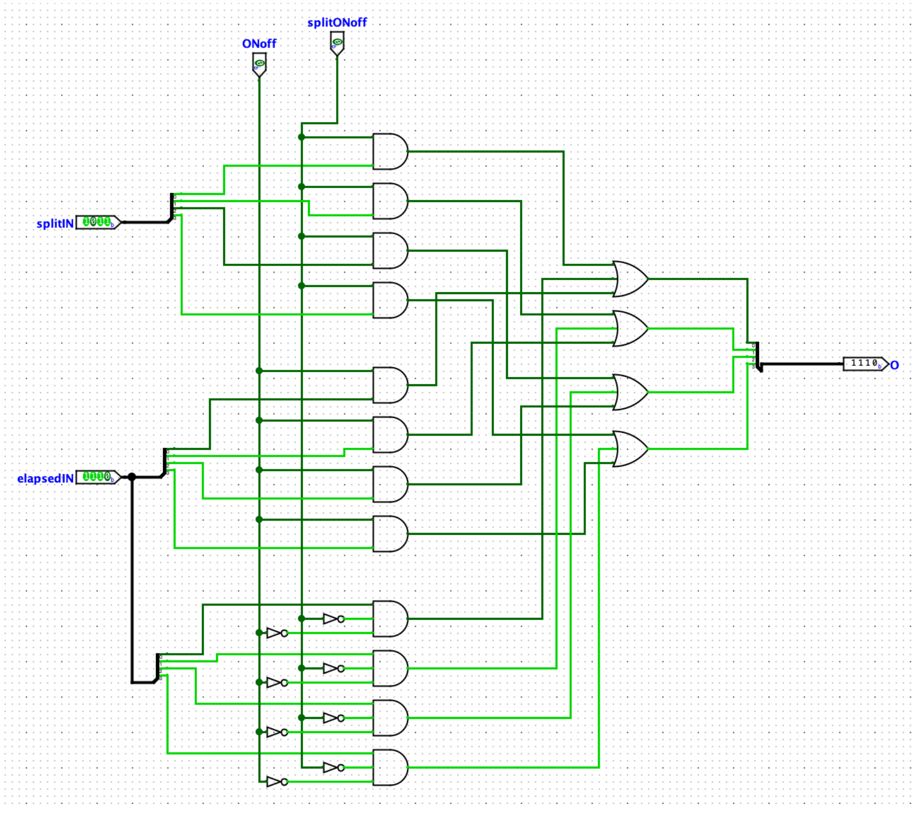


Stage 7 inherits most of the stage 6 subcircuits and makes a couple of changes to the other circuits. For this stage we have the same exact counters to count the elapsed time and the same exact stack to store and display the split times. The subcircuits that were modified were the encoder and the multiplexer.

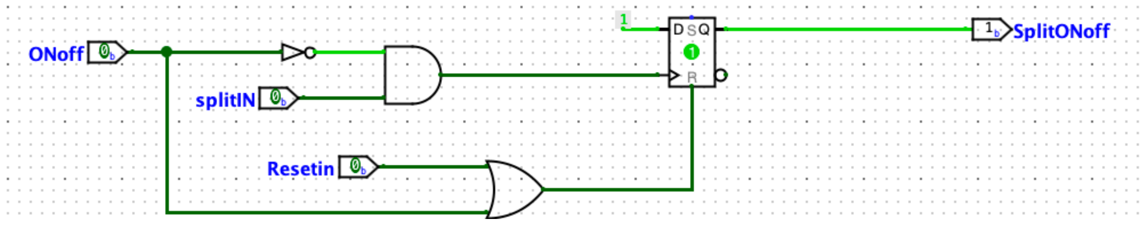
To the encoder, I added buffer gates to its outputs to delay the display of the split times as we had to show the elapsed time when the clock was stopped and the split time when the split button was clicked. I also made the counter count from 0 to 5 and then back to 0 by resetting the flip flop at 6.



For the multiplexer, we were now supposed to choose between the elapsed time and the split time to show on single display. I modified my circuit so that when the stopwatch is in the on state, the display will show the elapsed time and when the stopwatch is in the stop state and the split button is clicked, the display will show the split time. Moreover, I added another input to the multiplexer which is selected when both the buttons are not selected i.e. when the stopwatch is in the stop state and the split button is not clicked, the display will show the last elapsed time when the stop button was clicked.



I used another subcircuit for the split LED which only turns on when the stopwatch is in the stop state and the split button is pressed, indicating that the display is now showing split time. For this circuit I used a D flip flop because I needed to store the value of the input signal irrespective of the following signals. The D flip flop resets back to 0 when either the start button is pressed or when the reset button is pressed.



ASSUMPTIONS MADE

Assumption made for this assignment is that one clock pulse is equal to one second.

UNRESOLVED PROBLEMS

There are no unresolved problems with my circuits to the best of my knowledge.

REFERENCES

The reference for this report is:
https://swinburne.instructure.com/courses/49145/assignments/504471