ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)
ORGANISATION OF ISLAMIC COOPERATION (OIC)
DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

PHASE- A

TEAM NAME: **BOOLEAN AUTOCRATS**

TEAM MEMBERS:

Name	IDs
1. K.M. Sirazul Monir	200021247
2. Shafin Ibnul Mohasin	200021244
3. Md. Muhidur Rahman	200021242
4. Sk Asad Al Abir	200021331
5. Tahmid Hasan Muttaky	200021310

Supervisor Name: Ahmad Shafiullah
Lecturer, EEE

Introduction:

For phase A of SAP-1, we have 3 modules to construct:

1. Clock pulse generator.
2. Program counter.
3. Input unit and memory address register (MAR).

Clock pulse generator basically generates clock pulse which is used to run the whole computer. Program counter counts from 0 to 15 using binary and sends it to the bus. Memory address register takes the counting and outputs it to other modules. We may use enable, reset etc as inputs in program counter or memory address register to enable or reset their behavior as per our necessity.

Designing and implementation:

Clock pulse generator:

This module generates clock pulse which is basically a square wave signal consisting of 5V at peak and 0v at troughs.

Architecture of the module:

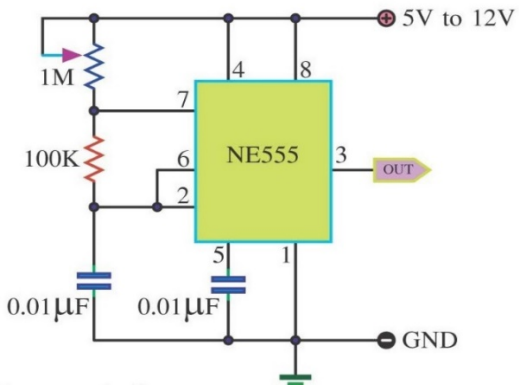


Fig.01

For astable clock, we used 555 timer IC to generate this clock pulse. This chip basically consists of two comparator, one flip flop and three 5k resistors hence the name 555 timer IC. For the resistors we used 1k and 100k respectfully and the capacitor we used was 1uf. These resistors and capacitor values will determine the clock speed. Later we will use variable resistor to increase or decrease the clock speed. But for now, let's go with the fixed values. These gives us around 3.8hz clock speed which we calculated using the formula provided in the datasheet. We connected the output of this clock to an LED.

For the monostable clock, we used another 555 IC chip and a trigger. By giving an input trigger, we can manually generate a clock pulse signal which can be used for debugging. We connected the output of this monostable clock to another LED.

For the bistable clock, we used yet another 555 timer ic and a 2-position switch (SPDT) to switch between astable and monostable. The purpose of this bistable generator is to choose between the astable and monostable multivibrator clock. we used an LED to indicate the final output coming from the bistable clock.

Program counter:

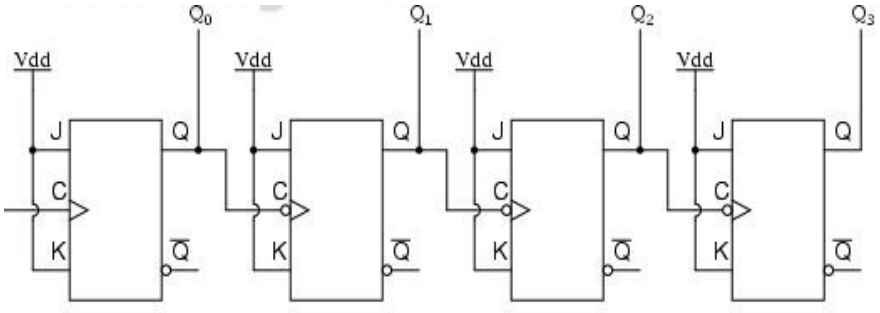


Fig.02

We used 4 j-k flipflops (2x 74HC107 DUAL JK FLIPFLOP IC) and 4 tri state buffers. As we know, by shorting J and K of the j-k flip-flop, we get a output at Q which is half the input signal. We used this behavior of the j-k flipflop and fed the Q output to another j-k flipflop and repeated it two more times. Thus, we get a binary counter. Now, to control the output we used a TRI state buffer by which we can control if the output will reach to the bus or not.

Input Unit & Memory address register (MAR):

MAR is basically 4 D- flipflops with 4 tri state buffers. We used an IC (74HC273 OCTA D flipflop) which has these built-in for convenience & we used AND gate to give the clock pulse & control this MAR functions simultaneously.
And we built input unit using a number of switches that will be used to give inputs to different modules all over the computer.

The difference between them is- Program counter holds the memory address of the next instruction to be fetched from main memory whereas memory address register (MAR) - holds the address of the current instruction that is to be fetched from memory, or the address in memory to which data is to be transferred.

Software simulation:

We used Proteus 8 for simulation. First, we built astable multivibrator. The simulation shows output through an LED.

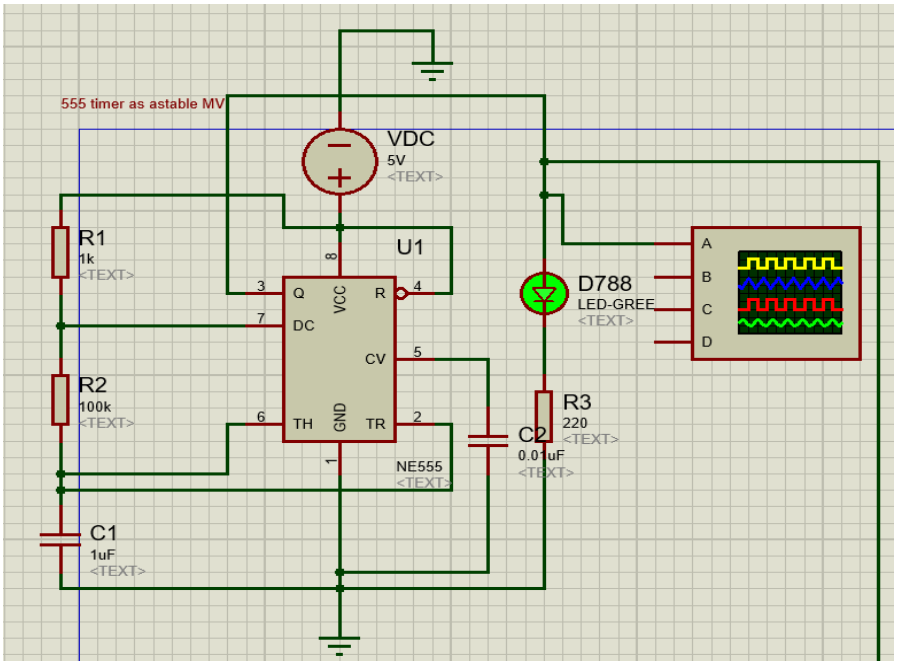


Fig.03: Astable multivibrator

We also verified the output pulses using Oscilloscope:

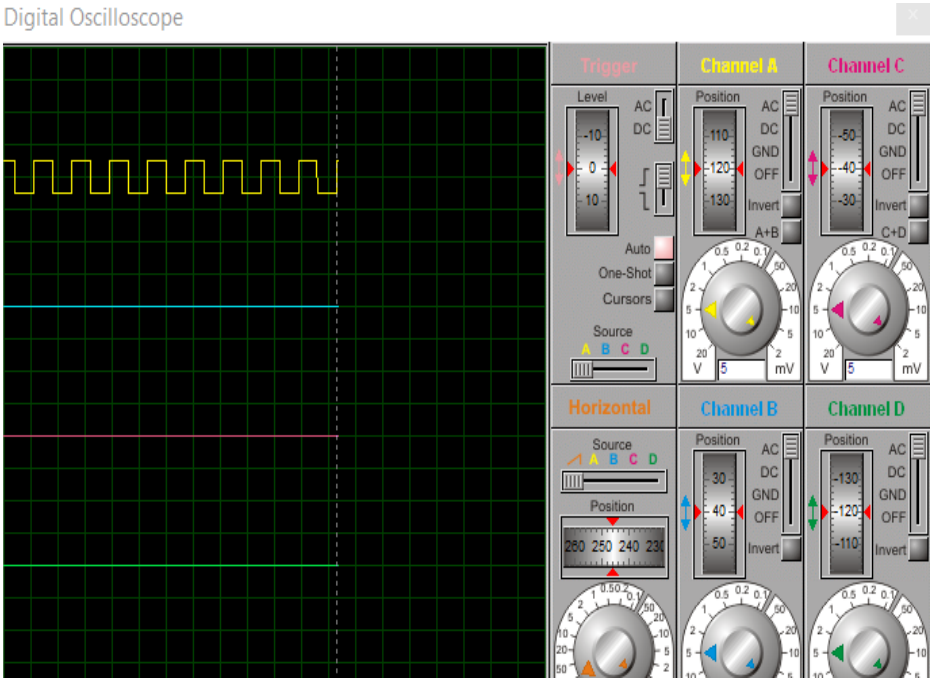


Fig.04: Oscilloscope output

After that we constructed monostable clock. The output is shown in an oscilloscope.

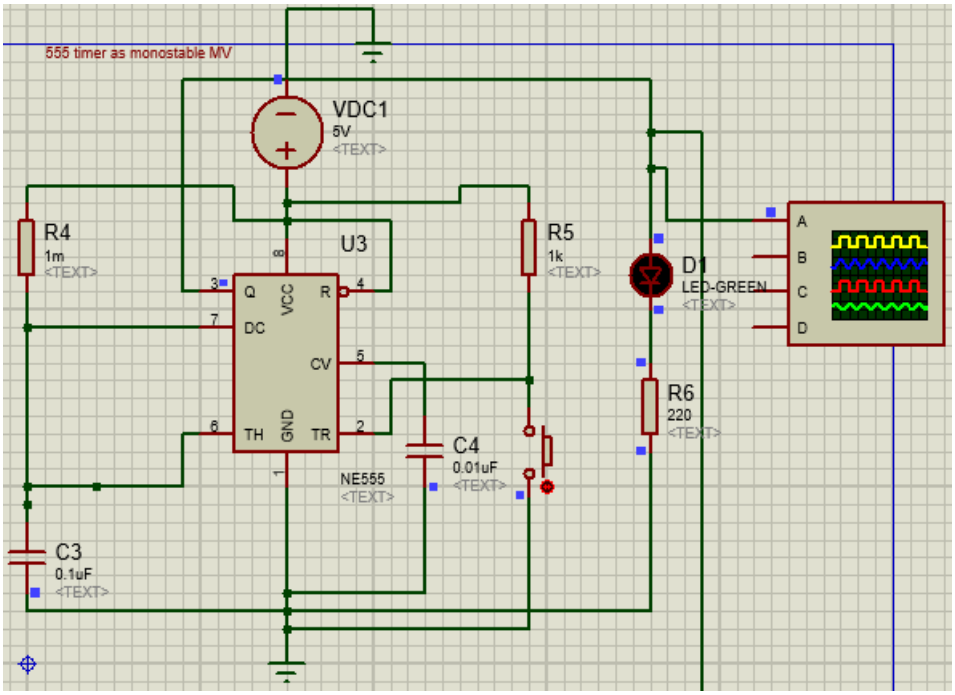


Fig.05: Monostable multivibrator

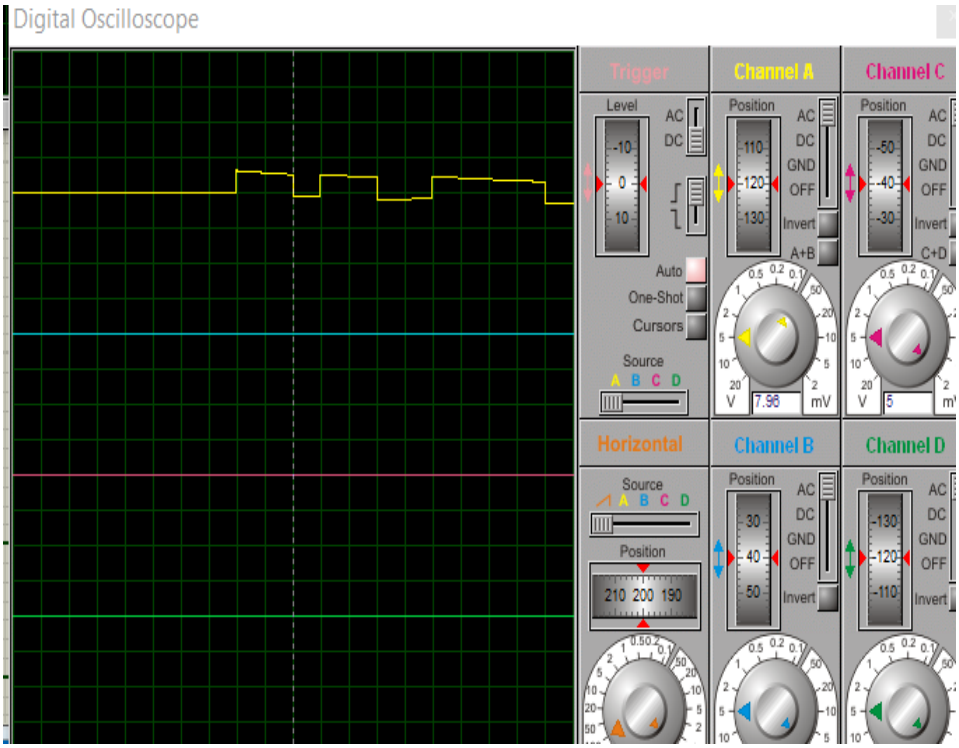


Fig.06: Oscilloscope output

After that we constructed bistable multivibrator and output is shown in the oscilloscope.

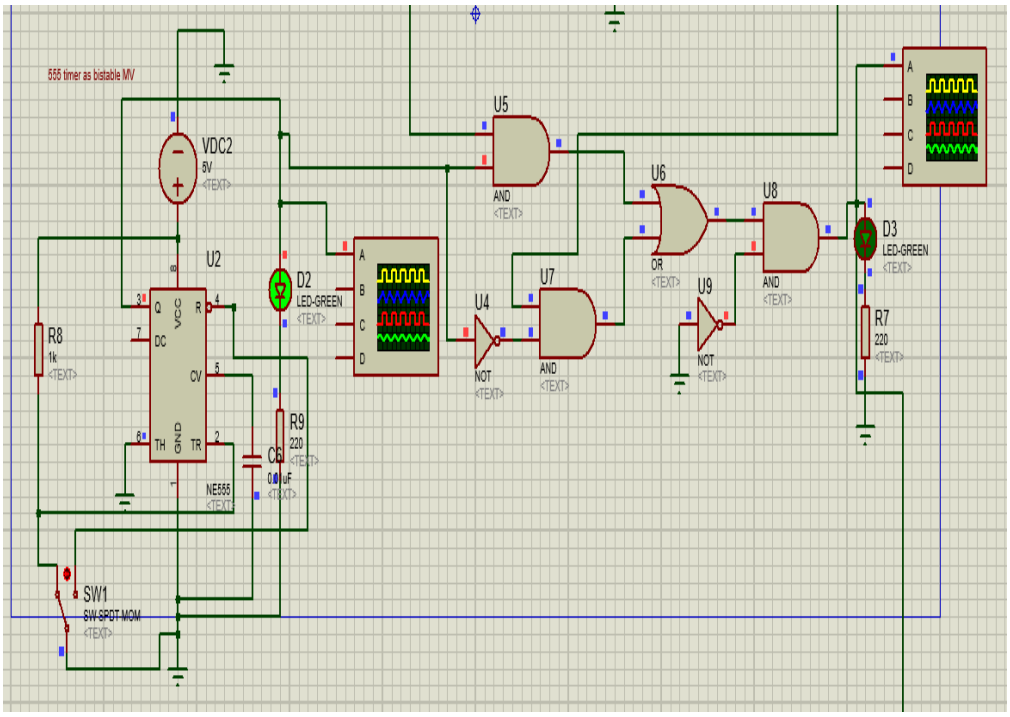


Fig.07: Bistable multivibrator

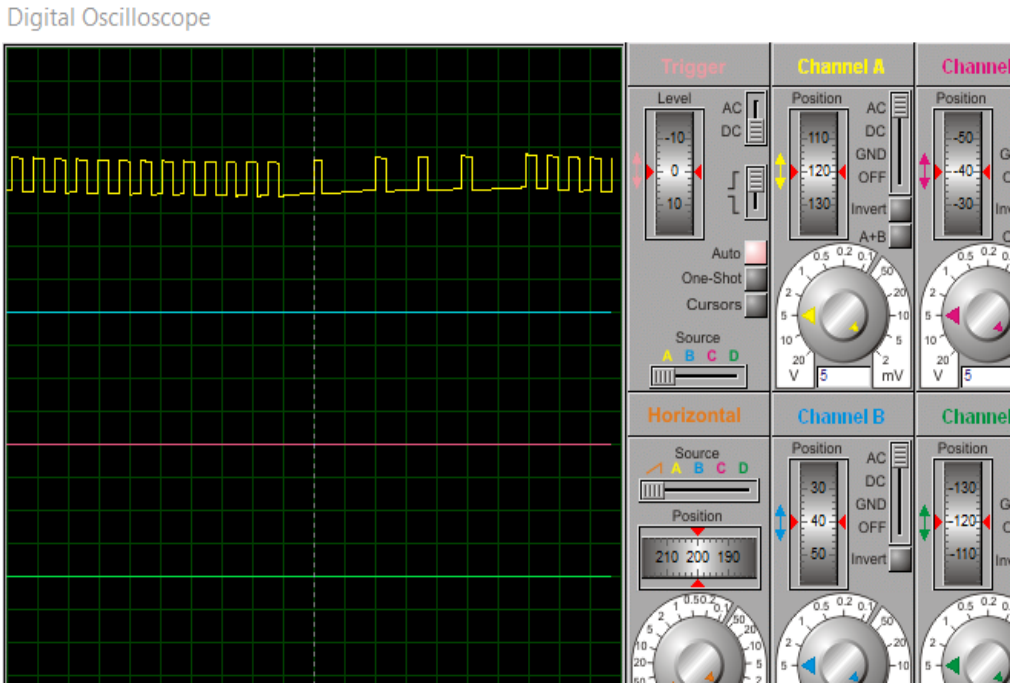


Fig.08: Oscilloscope output

Finally, we combined & synchronized all of them and observed the output:

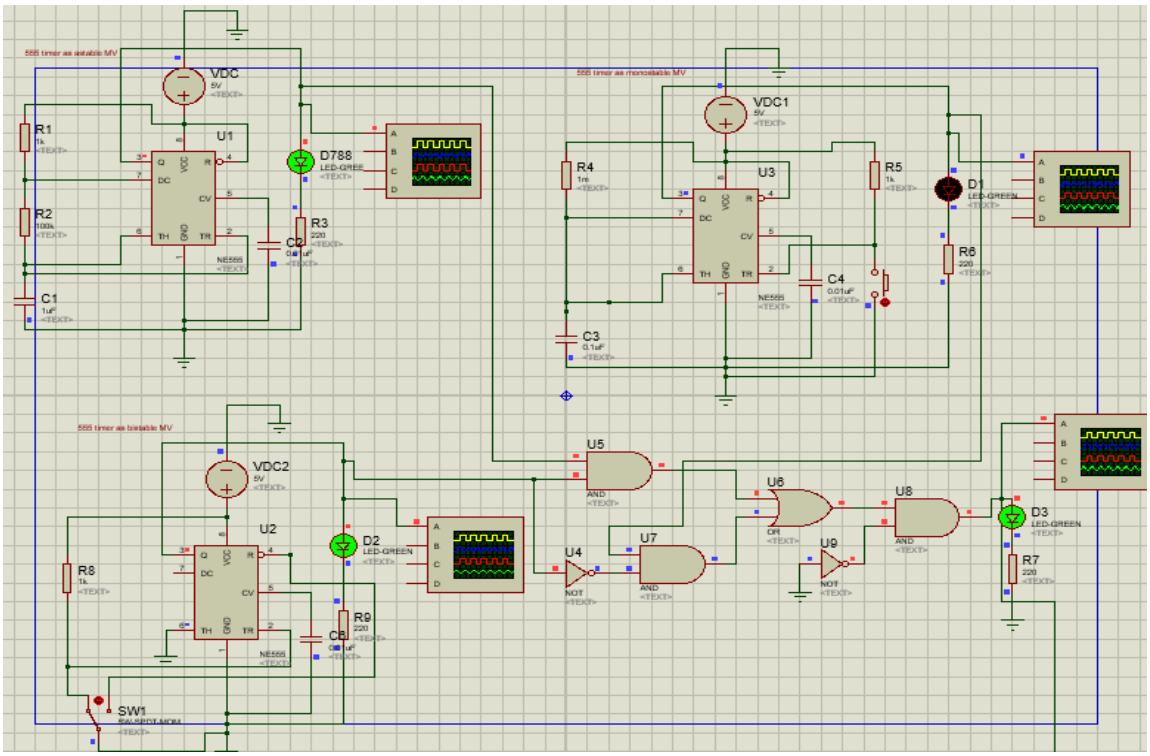


Fig.9: Clock pulse generator

Then we constructed the program counter. We built it separately and provided clock pulse separately. The output was shown using LED.

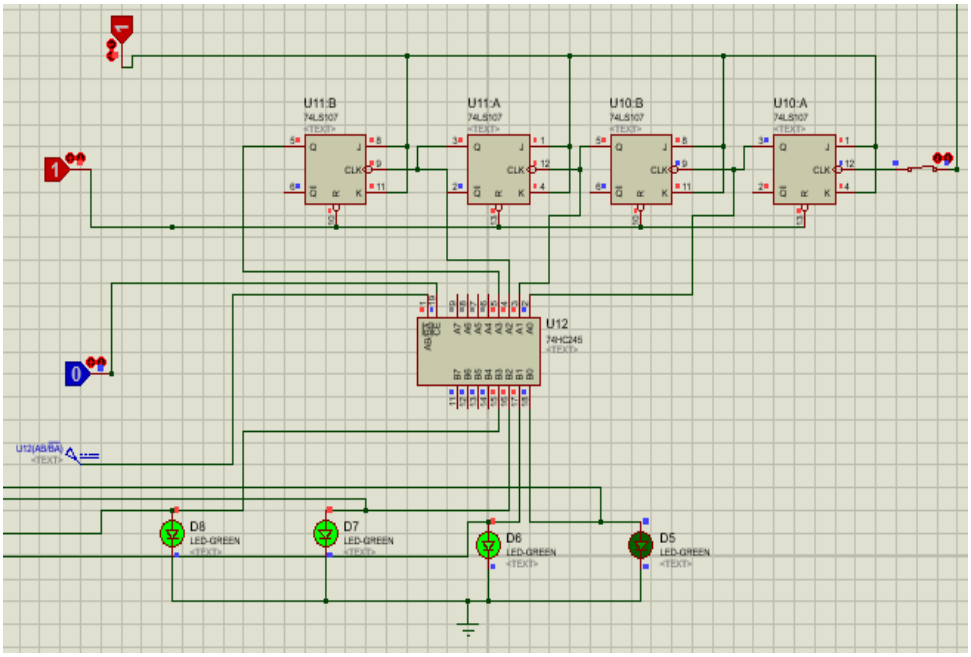


Fig.10: Program counter

After that we built the MAR. We used logic states to simulate the input and LED to simulate the output.

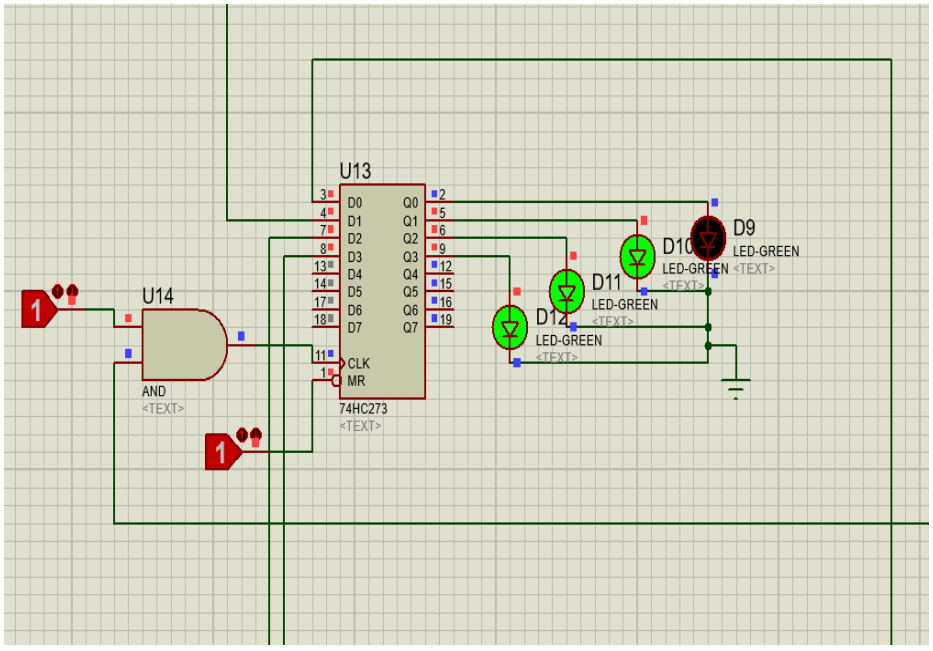


Fig.11: Memory address resister

Finally, we put them all together and tied all the clock pulse to the main clock pulse generator. This produced our desirable output.

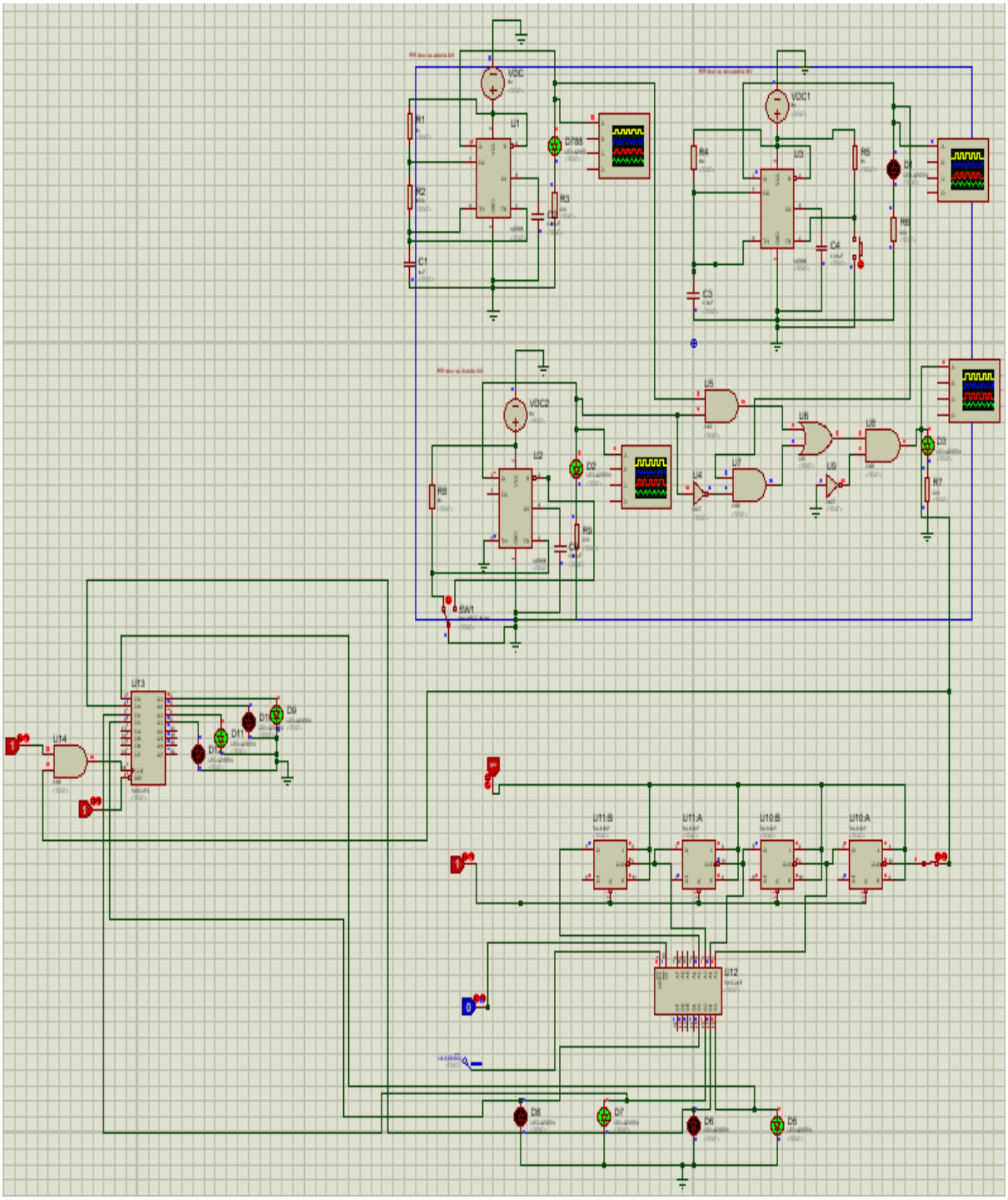


Fig.12: Complete schematic diagram

Hardware implementation:

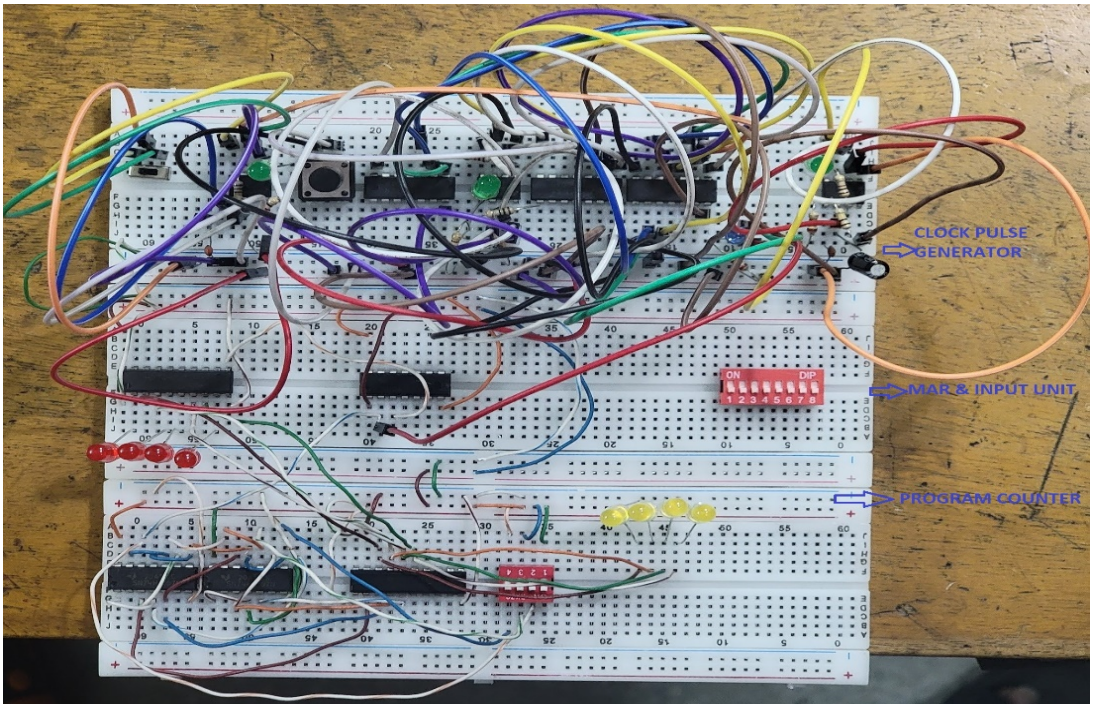


Fig.13: Complete hardware implementation using breadboard

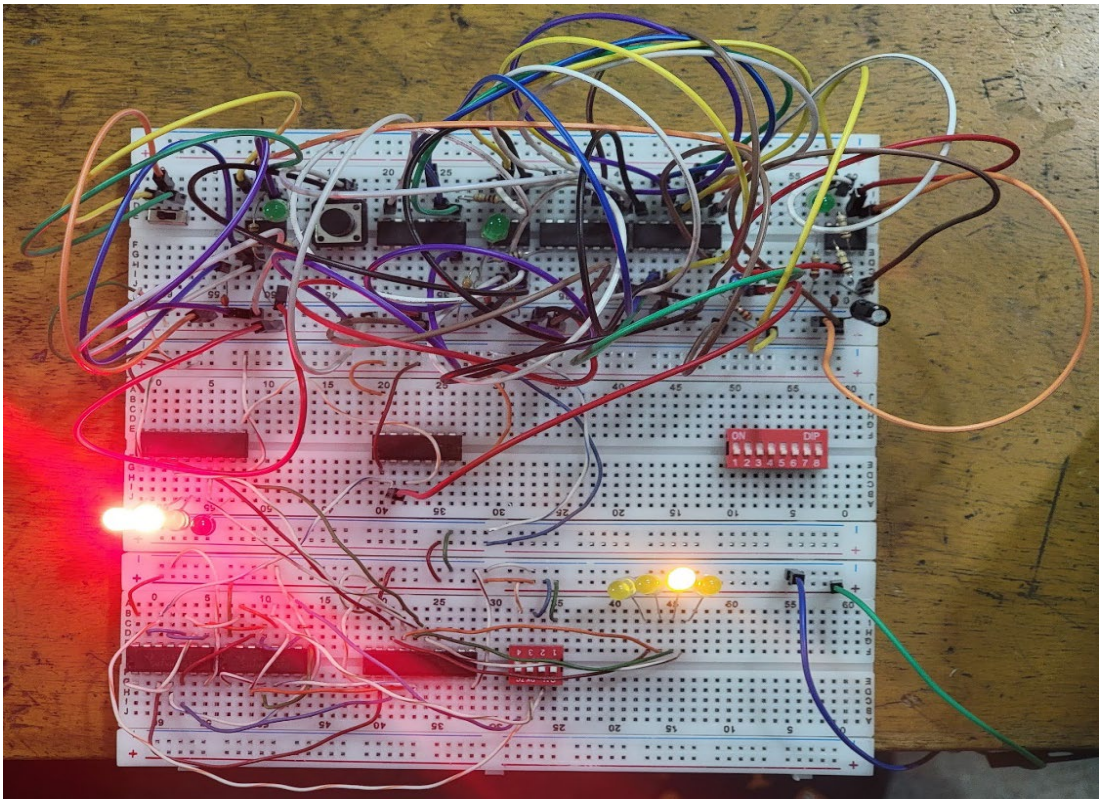


Fig.14: Complete module in function

For hardware, we used various ICs. Such as:
For clock pulse generator:

1. 3x 555 timer IC.
2. 1x 74hc08 AND GATE.
3. 1x 74HC32 OR GATE.
4. 1X 74HC04 NOT GATE.
5. Push button switch.
6. 2-position switch.
7. resistors, capacitors.
8. LEDs.

For program counter:

1. 2x 74HC107 DUAL JK FLIPFLOP.
2. 74HC245 OCTA TRISTATE BUFFER.
3. LEDs.

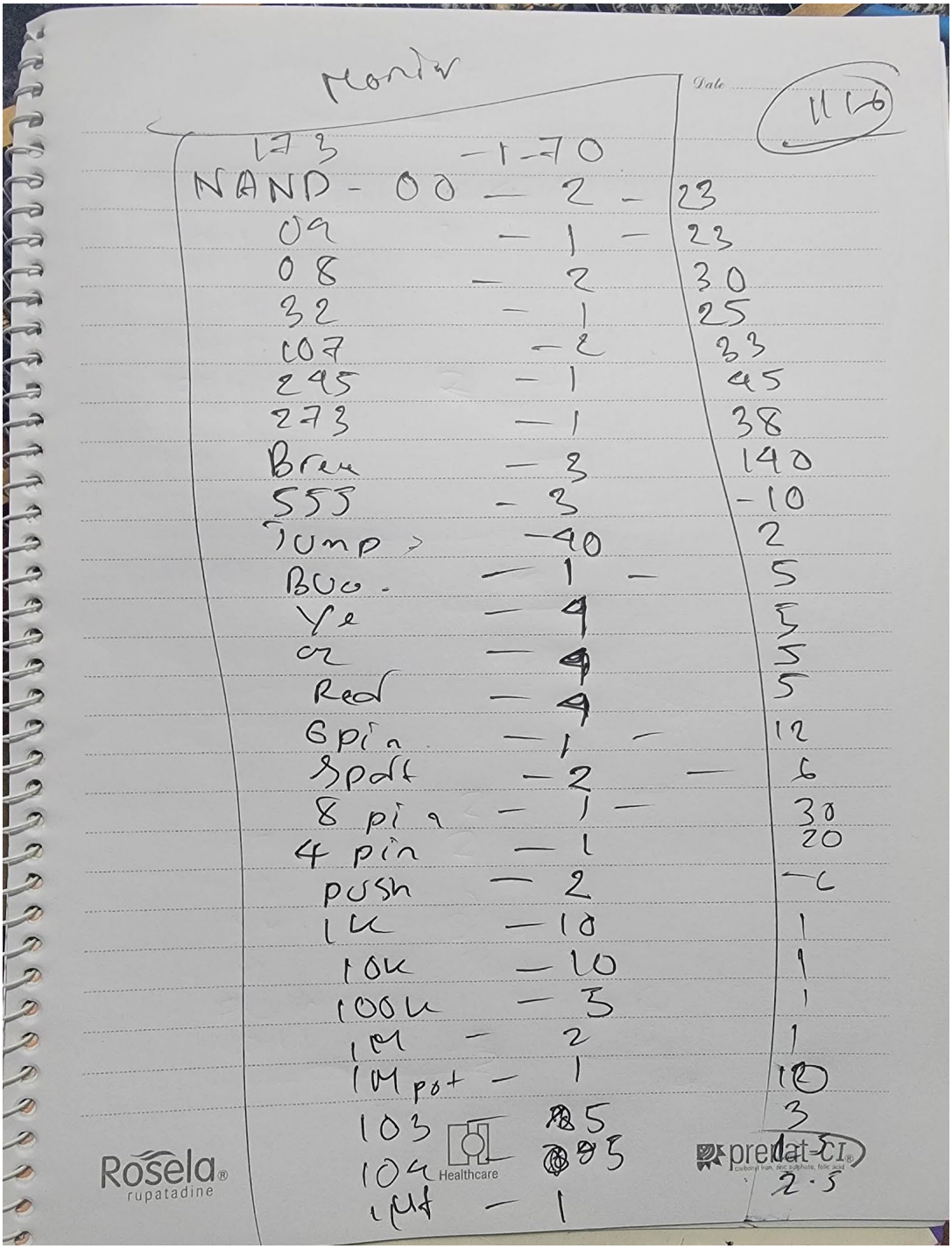
For MAR:

1. 74HC273 OCTA D flipflop.
2. 74HC08 AND gate.
3. LEDs.

We implemented them in the breadboard and used the manufacturers datasheet for the IC pinout diagrams. To power the whole circuit, we used 5V from USB port of power bank or mobile adapters. The outputs were observed by lighting up LEDs.

Logistics:

Here is a small list of hardware's and their prices:



Our total cost of week 1 was **1140BDT** and we had some components left for the upcoming weeks.

As we had a shortage of jumper wires, we had to make them ourselves by cutting and stripping Ethernet cables. We were unable to implement enable or reset switch as we did not find any two-position switch anywhere but we were able to replicate them by manually connecting them with 5V or ground.

Open discussion:

We observed an interesting phenomenon that is the output of the MAR unit is always 1 position lagging from the clock counter. After analyzing we understood that the counter always outputs the next instruction and that is why the MAR unit is lagging 1 behind it. Another interesting observation we made was about floating nodes. Whenever we had a floating node at input side that is neither 5V nor grounded, we saw random outputs. We fixed this by connecting them to appropriate inputs.

THE END