ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

Course: EEE 4308 (Digital Electronics)

Project: SAP-1 Architecture-based 8-bit Computer Design and Implementation

Group Name: CMTT

Group Members:

- Rayan Hossain Khan (190021214)
- Saadat Mahmud (190021307)
- Khandaker Fidak M. Rahman (190021311)

SAP-1

SAP stands for Simple as Possible. It is a simple design which covers the basics of computer operation like data storage, data transfer and arithmetic operations.

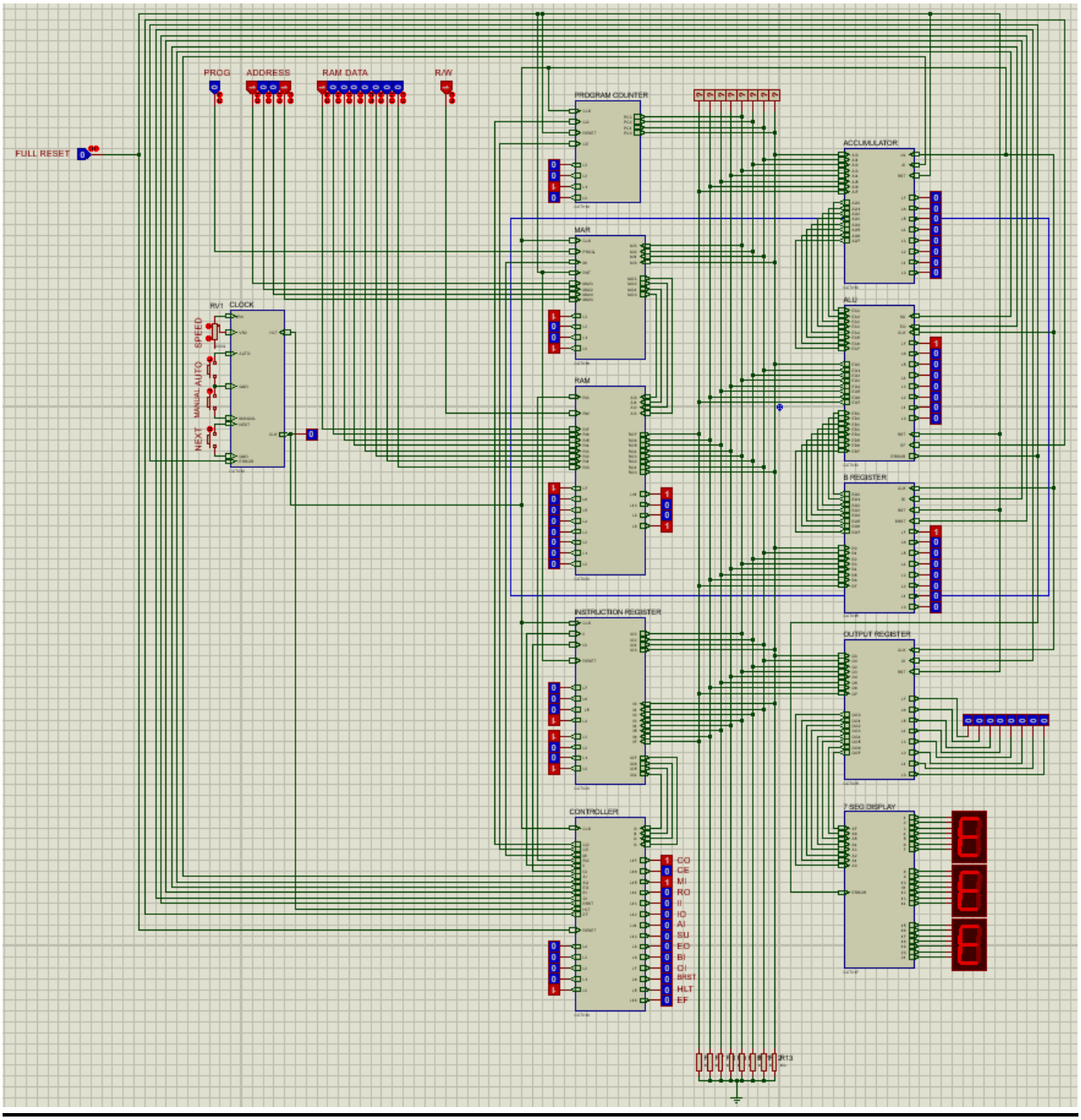
Objective

The objective of this project is to design a simple 8-bit computer based on the architecture of SAP-1 using combinational and sequential circuits. The computer has the following main components:

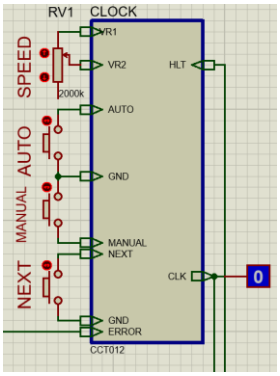
- An 8-bit BUS
- Clock Module
- 16*8-bit RAM
- Program Counter
- Input Unit
- Memory Address Register (MAR)
- Arithmetic Logic Unit (ALU)
- A-register and B-register
- Instruction Register
- Controller-sequencer
- Output Register

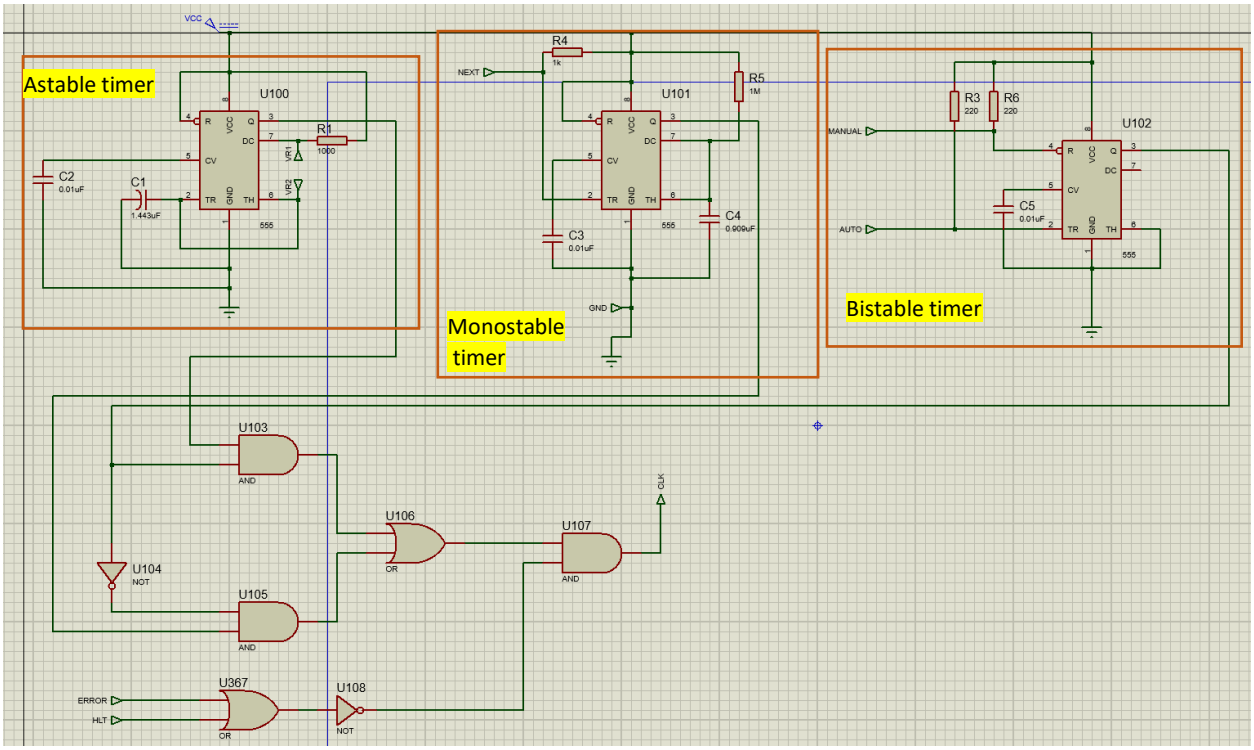
The main functions of the computer are:

- Load data
- Add
- Subtract
- Output
- Halt


PROG
ADDRESS
RAM DATA
R/W
FULL RESET
PROGRAM COUNTER
ACCUMULATOR
MAR
ALU
RAM
B REGISTER
INSTRUCTION REGISTER
OUTPUT REGISTER
CONTROLLER
7 SEG DISPLAY
RV1
CLOCK
SPEED
MANUAL-AUTO
NEXT
CO
CE
MI
RO
II
IO
AI
SU
EO
BI
OI
BRST
HLT
EF

CLOCK MODULE:





The clock module is the most important and primary component of the SAP-1. It provides the clock pulse for all the other components and determines when they will run or stop. There are two modes for this module- Auto and Manual. Depending on our choice, it can run on either stage. We can also control its speed in the auto mode using a variable resistor.

Related topics:

- 555 timer IC: It is an IC used as timer, delay, pulse generation, and oscillator applications.
- Astable timer: It has no stable states because it changes continuously between high and low voltage. We can calculate its time period by the following:
    - $T_{high} = \ln(2)*(R_1+R_2)*C = 0.693* (1000k+1k)\ \Omega*1.443uF = 1.001$ s
    - $T_{low} = \ln(2)*R_2*C = 0.693*1000k\Omega*1.443uF = 1.0002$ s
- Monostable timer: It has a single stable state. It starts with 0V. When triggered, 5V is obtained for a period of time, then again returns to 0V. . We can calculate its time period by the following:
    - $T = 1.1*R*C = 1.1*0.909uF*1M\Omega = 0.9999$ s
- Bistable timer: It is stable in both the stages. With each triggering, the output changes between 0V and 5V.

Features:

- Auto: Pressing this button, the clock runs continuously with 1s high and 1s low until a HALT command is obtained or the Manual button is pressed again.
- Manual: This button is for stopping the continuous Auto mode of the clock.
- Next: In the manual mode, pressing this button ends a positive pulse for 1s, thus, running the entire circuit for 1 positive clock pulse.
- Speed: The speed can be increased by decreasing the resistance of the potentiometer and vice versa.

Inputs:

- Error: When an Error is obtained in the ALU, the clock is stopped.
- HLT: The HLT command can also stop the clock through this input.

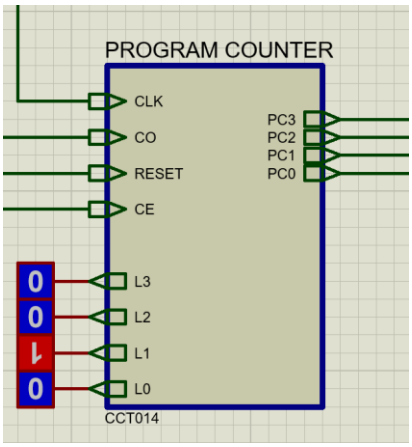
Outputs:

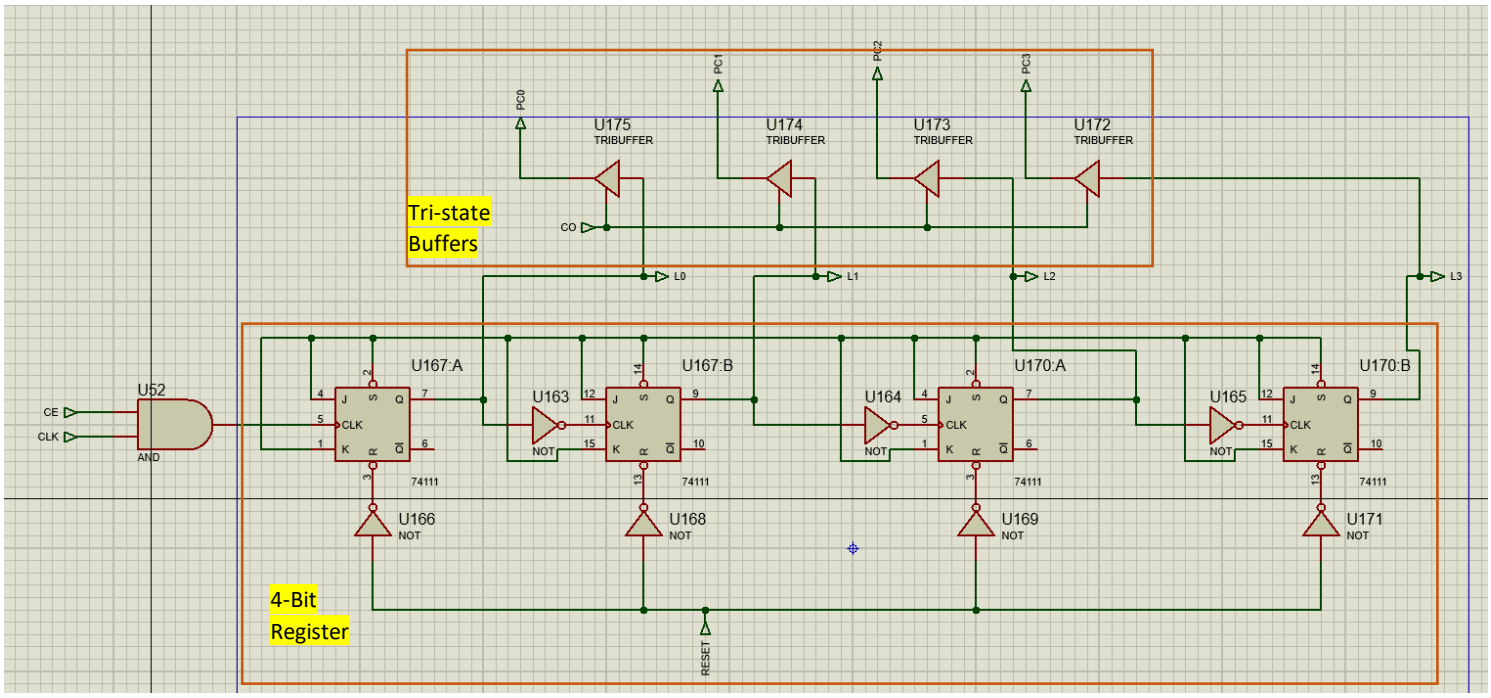
- CLK: This is the main output which reaches all the other components of the circuit and provides them with clock pulses.

Parts used:

- 555 timer IC
- Capacitor
- Resistor
- Variable resistor: For controlling the speed of the pulses
- AND gate
- OR gate
- NOT gate
- Logic probe: For viewing the state of the clock. It is used instead of a LED.
- VCC
- Ground
- Push Buttons: For selecting a mode and running the clock

Program Counter:





The program counter is basically a 4-bit binary asynchronous counter which can count from 0000 to 1111 which is 0 to 15. So, we can put as many as 16 commands in the Ram. The commands of the Ram are then serially run with the help of this program counter.

Inputs:

- CLK: Clock pulse from the Clock Module
- CO: Counter out- Puts the value of program counter into the BUS
- CE: Counter Increment- Increases the value of counter by 1
- RESET: Resets the value of counter to 0

Outputs:

- PC0 to PC3: This are connected to the BUS through 4 tri-state buffers.

Parts Used:

- Tri-state buffers: Used for controlling the flow of output into the BUS
- Master-Slave JK Flip Flop
- AND gates
- NOT gates
- Logic Probes: For showing the current count of the counter

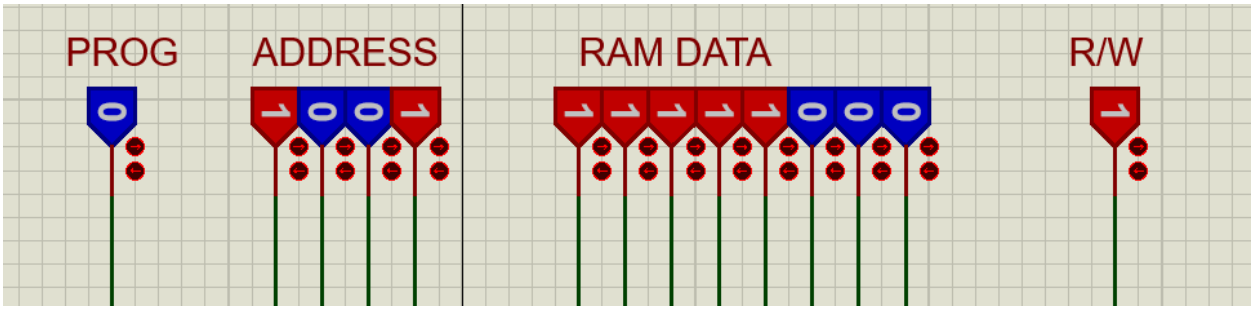
Truth table:

STAGE	PC3	PC2	PC1	PC0
Initial	0	0	0	0
2^{nd}	0	0	0	1
3^{rd}	0	0	1	0
4^{th}	0	0	1	1
5^{th}	0	1	0	0
6^{th}	0	1	0	1
7^{th}	0	1	1	0
8^{th}	0	1	1	1
9^{th}	1	0	0	0
10^{th}	1	0	0	1
11^{th}	1	0	1	0
12^{th}	1	0	1	1
13^{th}	1	1	0	0
14^{th}	1	1	0	1
15^{th}	1	1	1	0
16^{th}	1	1	1	1

Changes for CLK, CE, CO and RESET:

CLK	CE	CO	RESET	Change	Output
X	X	0	1	0000	NO
X	X	1	1	0000	YES
↑	1	0	0	Next stage	NO
↑	1	1	0	Next stage	YES
↑	0	0	0	Same stage	NO
↑	0	1	0	Same stage	YES

INPUT UNIT:



This part is used to input commands and data into the RAM into the required address before the Clock starts. PROG must be 1 before entering the data, that is, programming mode must be enabled. After every data input in their definite addresses, R/W is witched off and then on to write that data.

Parts used:

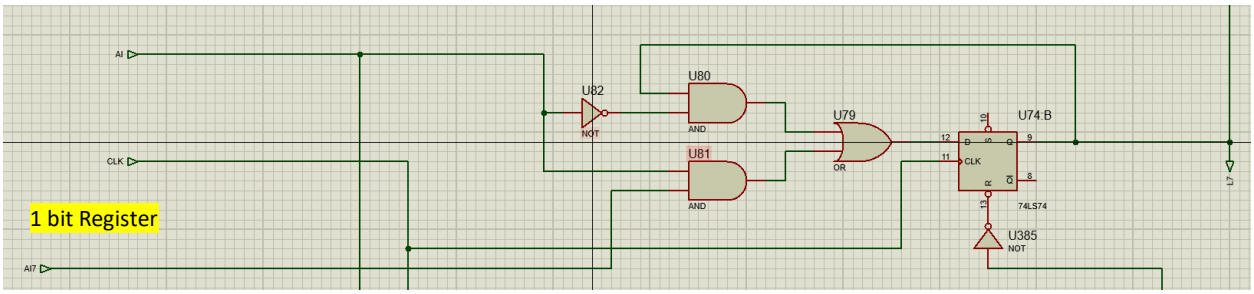
- Logic States: For inputting either 1 or 0 into the MAR/RAM

Inputs:

- PROG: For choosing between inputs from BUS and inputs from Input Unit
- ADDRESS: This 4-Bit address of Ram directs the data to any of the 16 places
- RAM DATA: This 8-Bit data contains the command and data for computing
- R/W: Turning this off and then on again on a specific address will input the 8-bit data on that specific address of the Ram.

***Registers

Register is the prime component of the SAP-1. A 1-bit register can save 1 bit of data, which is either 1 or 0. Several registers are used together to make a bigger register which in turn make up the Accumulator, B-register, MAR, Input register, Output Register. Even the RAM is just a collection of 128 registers. It has 8 registers in 16 Columns. Thus, we can save 8-Bit data or a word in any of the 16 addresses. So, the basic of a register is discussed below:



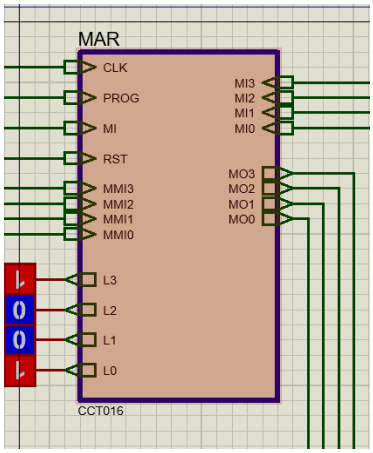
Basically, it consists of a single D Flip-flop which stores data whenever the enabler is active and it receives a positive clock pulse. It is to be noted that all the D flipflops used here are positive edge triggered. The combinational circuit used before the input of the D flipflop is simply a mux. It has two cases:

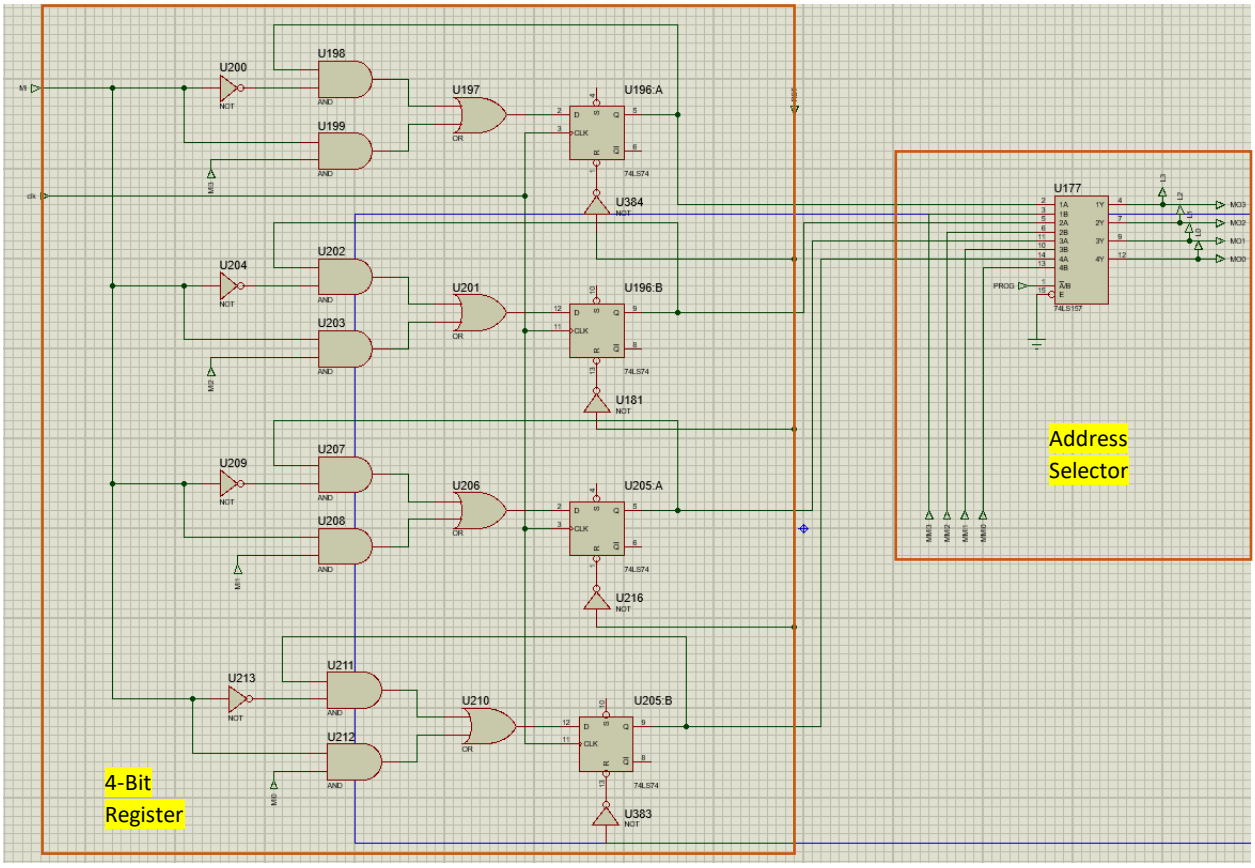
- Enabler is active: Stores new data
- Enabler is off: Cycles the stored data back to the flipflop

So, any number of registers can be used together to form a set of parallel register. They have a common enabler and a common clock input. Along with those, we can give a common reset input. Tri-state buffers can be also used in the outputs of the registers to control the output flow.

Clk	Load (enabler)	D	Q
↑	0	X	Previous input
↑	1	X	X

MAR and Address Selector:





MAR stands for Memory Address Register. It holds the 4-bit address of the RAM and utilizes it in the next clock pulse to retrieve the data of that particular address in the RAM. It has two types of inputs:

- Manual Input: Given by the user in programming mode for storing commands and data in the RAM
- From BUS: Obtained from the program counter sequentially from 0000 to run the commands stored.

Here, the address selector determines the input based on the run/prog switch. For PROG=0, it takes the BUS input and in PROG=1, it takes the manual input.

Parts used:

- For Register:
    - D Flip Flop
    - OR Gate
    - AND gate
    - NOT Gate
- For Address Selector:
    - Quadruple 1 of 2 data Selector (MUX):
- Logic Probes

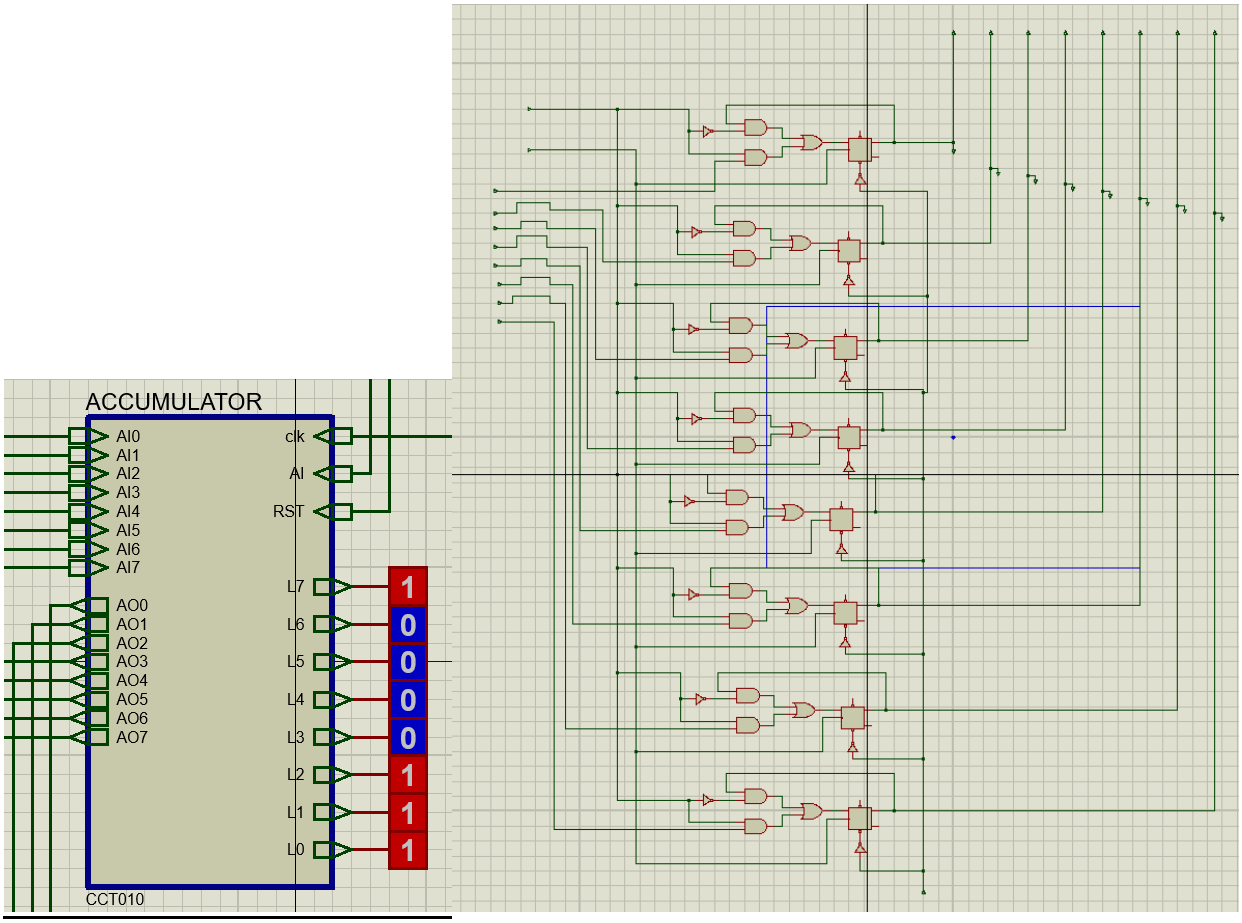
Inputs:

- CLK: For positive edge triggering
- PROG: Choosing between programming and run mode
- MI: For input Control
- RST: Reset
- MMI0 to MMI3: Manual inputs from Input Unit
- MI0 to MI3: Inputs coming from Program Counter through bus

Outputs:

- MO0 to MO3: To RAM

ACCUMULATOR:



It is the primary register for operations and storing result. It is a simple 8-bit register which stores a 8-bit data from the RAM which is later used for any arithmetic operation. Also, the result from the ALU is stored here.

Parts Used:

- For 1 bit Register:
  - D Flip- Flop
  - AND Gate
  - OR gate
  - NOT gate

- Logic Probes

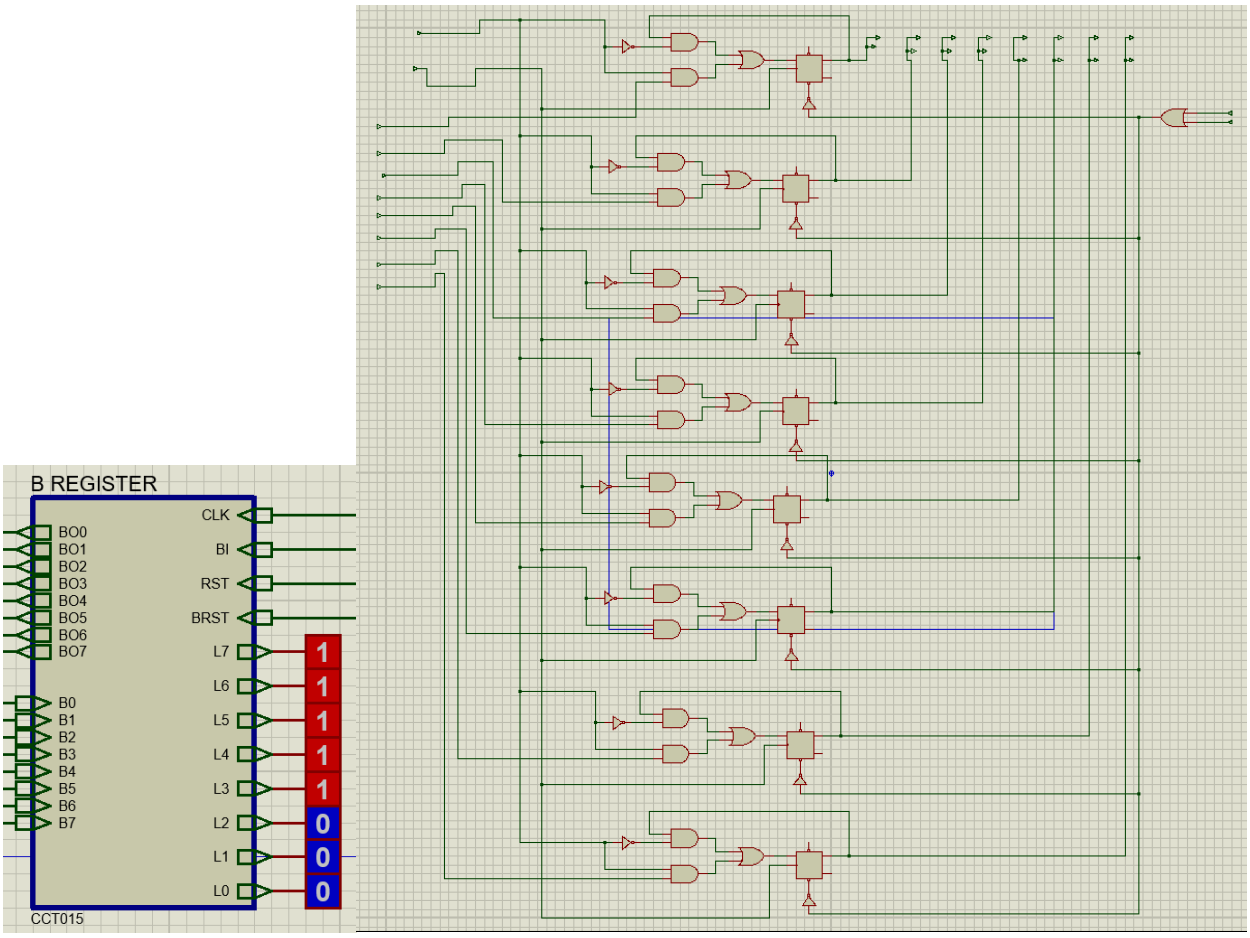
Inputs:

- AI0 to AI7: From RAM through BUS
- CLK
- AI: To control the inputs
- RST: Reset

Outputs:

- AO0 to AO7: Directly connected to the ALU without any buffer

B REGISTER:



It is the secondary register or temporary register which is very much similar to the A register. It holds the data for the second part of any arithmetic operation. The outputs are directly connected to the ALU.

Parts Used:

- For 1 bit Register
    - D Flip- Flop
    - AND Gate
    - OR gate
    - NOT gate
- Logic Probes

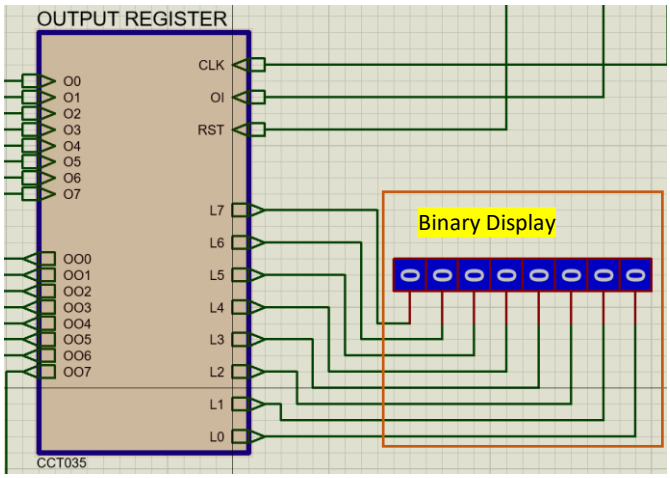
Inputs:

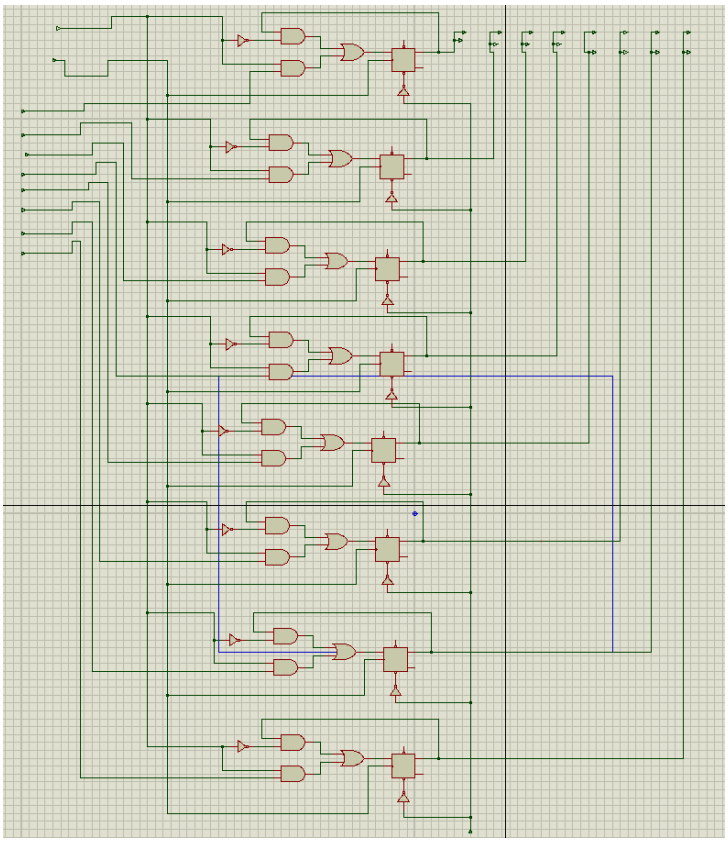
- B0 to B7: From RAM through BUS
- CLK
- BI: Controls input
- RST: Reset
- BRST: Control bit which is activated with the OUT command. It resets the B register so that the value in A register and ALU becomes same.

Outputs:

- BO0 to BO7: To ALU directly

Output Registers:





This 8-bit register stores the result from the A-register when the OUT command is obtained. Then it shows the output in a binary display and also transfers the output to the 7-segment display.

Parts Used:

- For 1 bit Register:
  - D Flip- Flop
  - AND Gate
  - OR gate
  - NOT gate
- Logic Probes

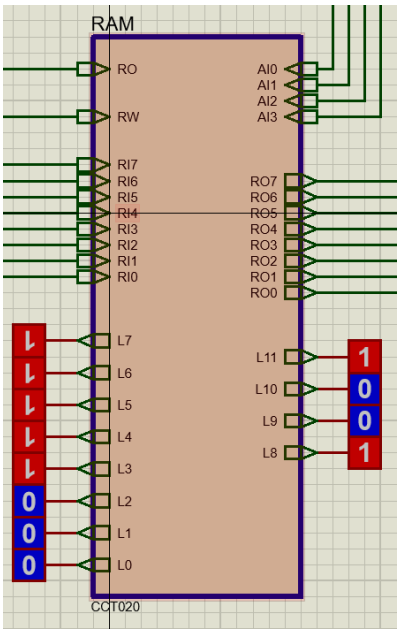
Inputs:

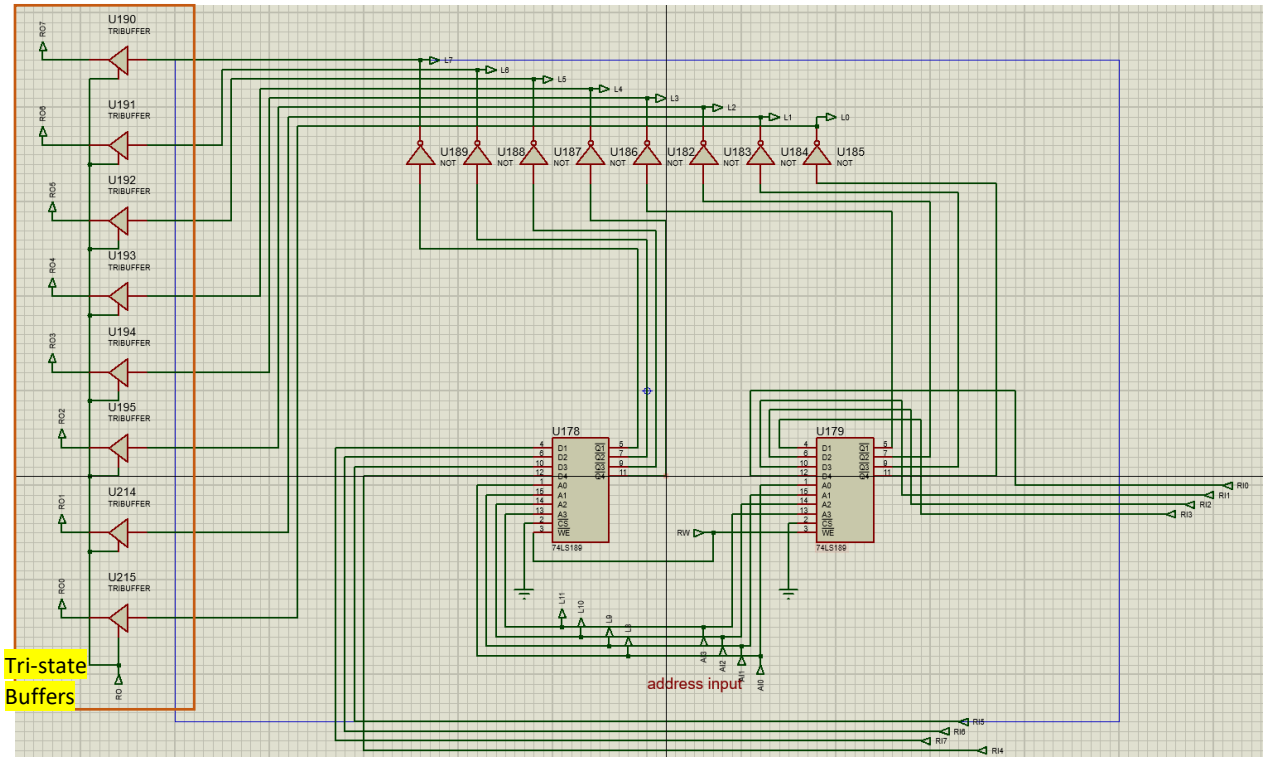
- O0 to O7: From A-register through ALU and BUS
- CLK
- OI: Controls the Input
- RST: Reset

Outputs:

- OO0 to OO7: To 7-segment display

RAM:





The RAM is the main storage component of the SAP. It stores the initial commands and the data. Then it supplies that information to the rest of the parts and helps in completing the operations. Here, we used two 64-bit ram modules.

Each one of them contains 16 addresses and 4 bit per address. So, in together we have 16 addresses with 8 bits in each. The RAM has two modes:

- Read: When the computer runs, it supplies with the available data in this mode
- Write: While in programming mode, the RAM can be re-written in this mode

Since our 74LS189 IC gives inverted output, we used 8 NOT gates to correct them. Then 8 tri-state buffers were used to control the output flow to the BUS.

Parts Used:

- 74LS189 (64 Bit RAM)
- Tri-State Buffers
- NOT Gate
- Ground

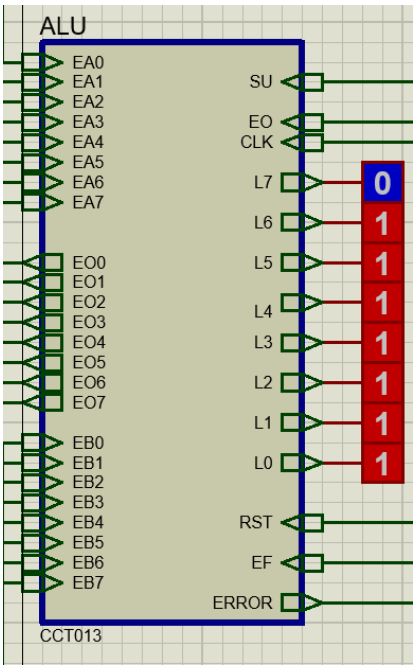
Inputs:

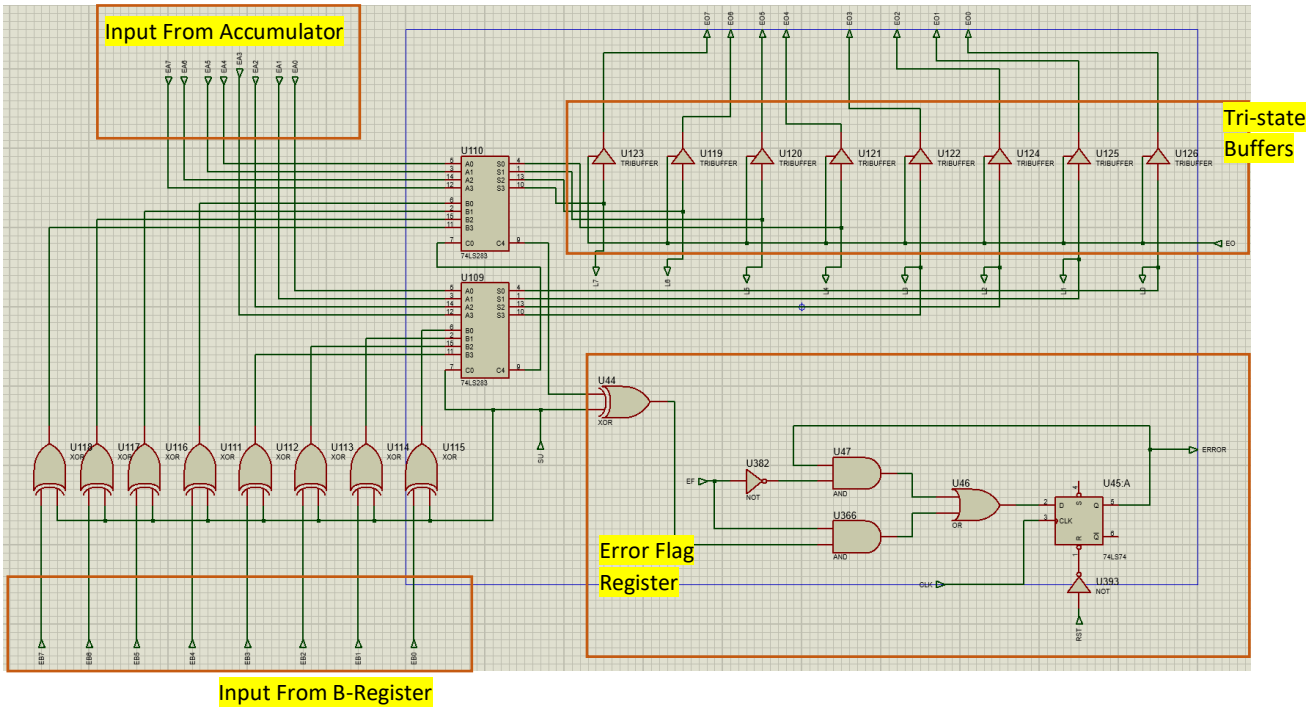
- RO: Controls the output flow through tri-state buffers
- RW: Switches between read and write mode
- A0 to A3: Address input from MAR
- RI0 to RI7: Data inputs from Input Unit

Outputs:

- RO0 to RO7: Data outputs to BUS

ALU:





ALU stands for Arithmetic and Logic Unit. It performs the arithmetic and logic operations as commanded by the instructions from the RAM. It basically receives two 8-bit data from A- register and B-register and then adds or subtracts them depending on the command. So, two 4-bit binary full adders are used here.

For subtraction, XOR gates are used to do 2's complement. Then it is added with the 1st number to find the result. The following truth table can be observed to understand the operation of ALU:

A0-A7	B0-B7	SU	EO0-EO7	9th bit	ERROR
00000000	00000000	0	00000000	0	No
00000001	00000001	0	00000010	0	No
…	…	0	…	0	No
11111111	00000001	0	00000000	1	Yes
11111111	00000000	1	11111111	1	No
11111111	11111111	1	00000000	1	No
…	…	1	…	1	No
00000000	00000001	1	11111111	0	Yes

So, the 4-bit binary adder does these operations bit by bit on the two numbers. We can see that an error is obtained whenever SU and 9B are different. But when they are equal, the result can be shown. This is the property of an XOR gate. So, we connected SU and 9B through an XOR gate and save the ERROR flag at the ADD-5 and SUB-5 states. If the flag is 1, then all other operations are halted and ERROR is shown in the 7-segment display.

Parts used:

- XOR gates
- For error Flag Register:
    - D Flip-Flop
    - AND gates
    - OR Gate
    - NOT Gates
    - XOR gate
- Tri-State Buffers
- 74LS283: 4 Bit Binary Full Adders
- Logic Probes

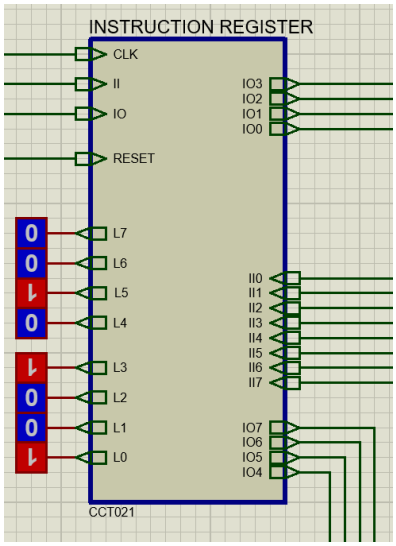
Inputs:

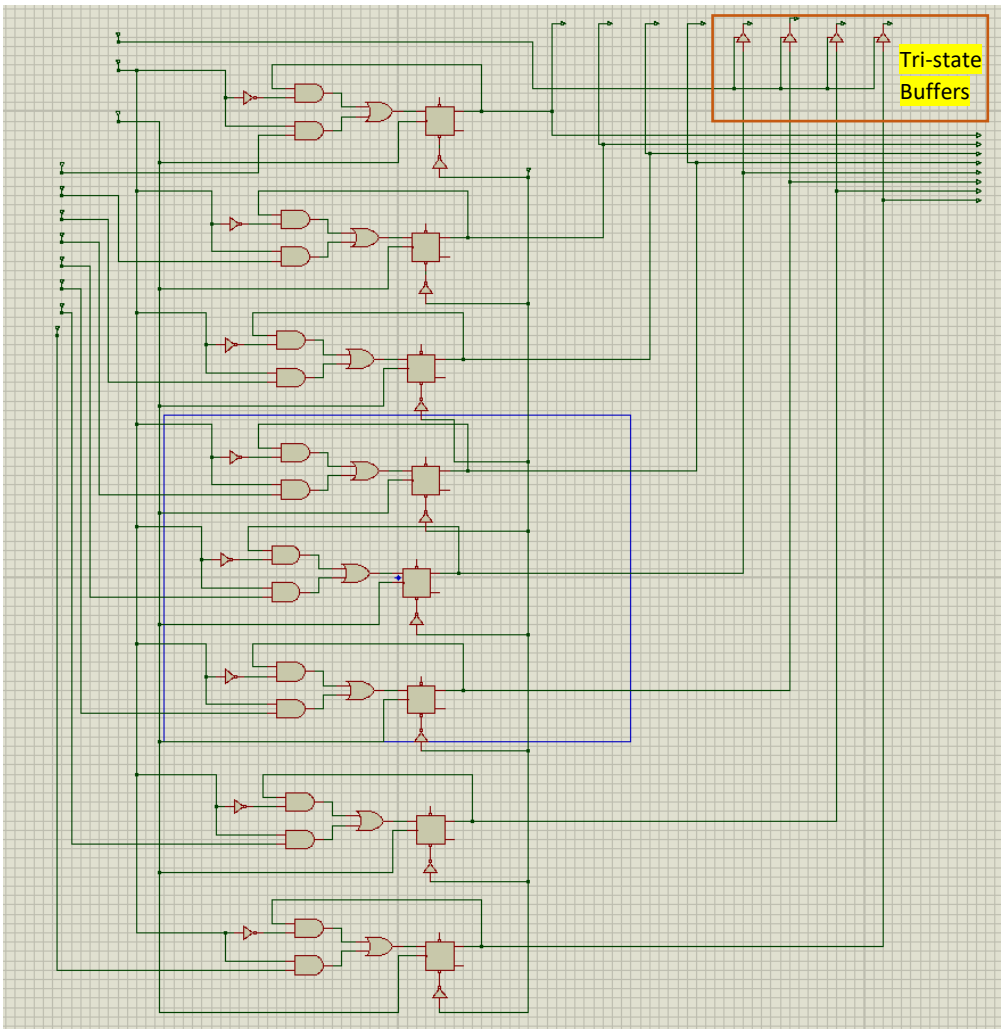
- EA0 to EA7: From Accumulator
- EB0 to EB7: From B-Register
- CLK
- RST: Reset
- EF: Error Flag from Controller
- EO: Sum out control Bit
- SU: Subtraction control Bit. 1 for SUB, 0 for ADD

Outputs:

- EO0 to EO7: To output register through BUS
- ERROR: To Clock and Output Register. Halts the Clock and shows ERROR on the 7-segment display

Instruction Register:





This is also an 8-bit register which receives 8-bit data from the RAM. These are the commands which are inserted into the RAM in the starting parts. Then the 8-bit is divided into 2 parts. The upper nibble is sent to the Controller. The lower nibble is again sent to the MAR through the BUS which is then sent to the RAM to retrieve the 8-bit data.

Upper nibble= OPCODE, connected directly

Lower nibble= Address of RAM, connected through Tri-state buffers

Parts Used:

- For 1 bit Register:
    - D Flip- Flop
    - AND Gate
    - OR gate
    - NOT gate
- Logic Probes
- Tri-state Buffers

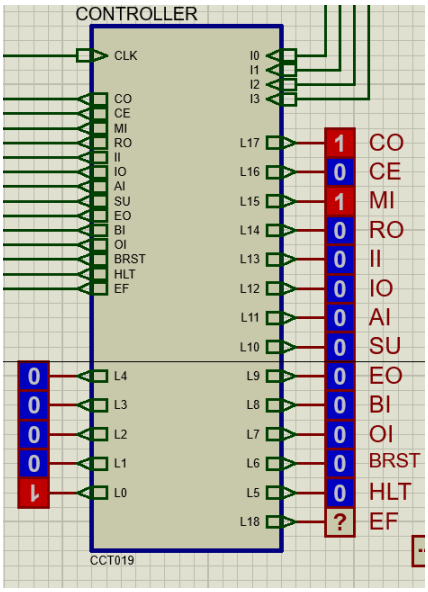
Inputs:

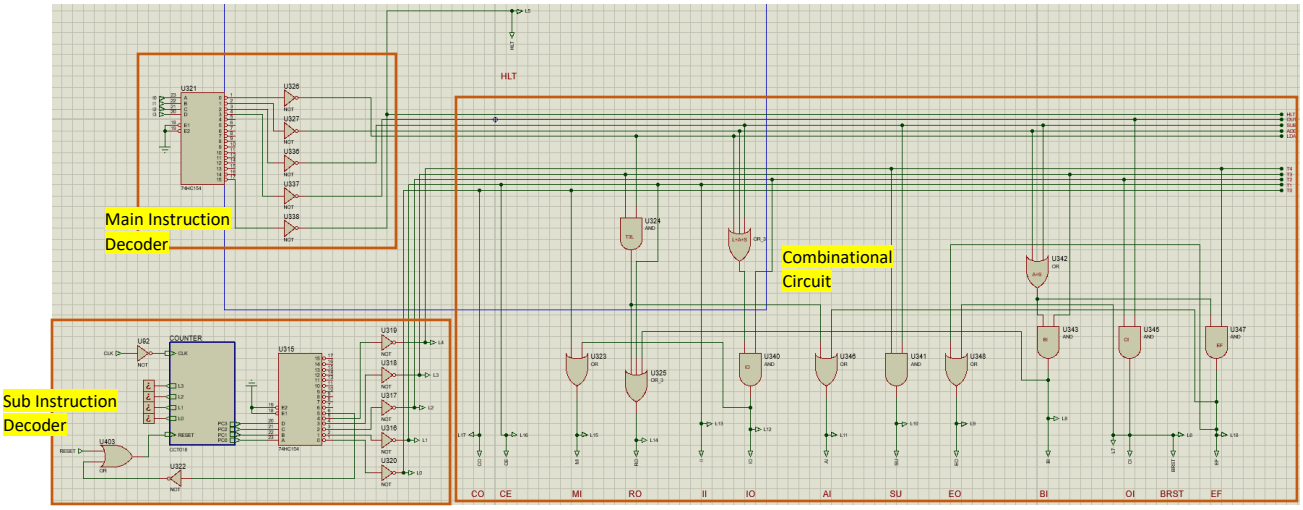
- II0 to II7: Inputs from RAM through BUS
- CLK
- II: Input control Bit
- IO: Output Control Bit
- RESET

Outputs:

- IO0 to IO3: To BUS
- IO4 to IO7: To controller

Controller-Sequencer:





Controller is the brain of the Sap. It decides which component will be switched on when. There are 14 control bits here connected to different parts. It I controlled by combinational circuits connected to two different parts. They are main instructions and sub-instructions. The main instructions are LDA, ADD, SUB, OUT, HLT and the sub-instruction are t0 to t4 for each of the

Parts Used:

- For decoding the main instructions:
    - 4 to 16 decoder: For decoding the commands from Instruction register
    - NOT Gates
    - Ground
- For decoding the sub-instructions:
    - 4-bit Synchronous counter:
        - Tri-state buffers
        - Master-Slave JK Flip Flop
        - AND gate
        - NOT gates
        - Logic Probes
    - NOT gates
    - Logic probes
    - 4 to 16 decoder: To decode the T-states 0 to 4. Then 5 is connected to reset
    - Ground:
- For combining the main instruction and micro-instructions:
    - AND Gates
    - OR Gates

Inputs:

- CLK
- I0 to I3: From Instruction register directly
- RESET

Commands:

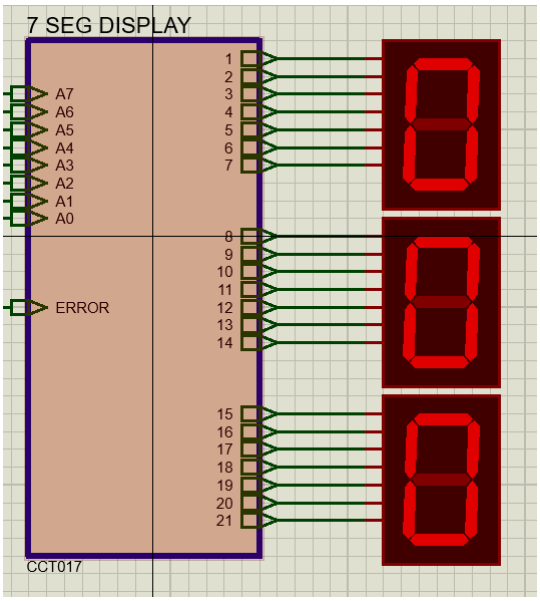
- LDA: 0000
- ADD: 0001
- SUB: 0010
- OUT: 0011
- HLT: 1111

	CO	CE	MI	RO	II	IO	AI	SU	EO	BI	OI	BRST	HLT	EF
FETCH-1 (F1)	1		1											
FETCH-2 (F2)		1		1	1									
LDA-3 (L3)			1			1								
LDA-4 (L4)				1			1							
ADD-3 (A3)			1			1								
ADD-4 (A4)				1						1				
ADD-5 (A5)							1		1					1
SUB-3 (S3)			1			1								
SUB-4 (S4)				1						1				
SUB-5 (S5)							1	1	1					1
OUT-3 (O3)									1		1	1		
HLT-3 (H3)													1	

Outputs (Control Bits):

- CO (Counter Out) = F1= T_0
- CE (Counter Increment) = F2= T_1
- MI (MAR In) = F1+L3+A3+S3= F1+IO= T_0 + IO
- RO (RAM Out) = F2+L4+A4+S4= BI+T_1+T_3L
- II (Instruction In) = F2= T_1
- IO (Instruction Out) = L3+A3+S3=T_2(L+A+S)
- AI (Accumulator In) = L4+A5+S5= EF+T_3L
- SU (Subtraction) = S5= T_4S
- EO (Sum Out) = A5+S5+O3= EF+OI
- BI (B-Register In) = A4+S4= T_3(A+S)
- OI (Output Register In) = O3= T_2O
- BRST (B-Register Reset) = O3= OI
- HLT (Halt) = H3= T_2H
- EF (Error Flag) = A5+S5= T_4(A+S)

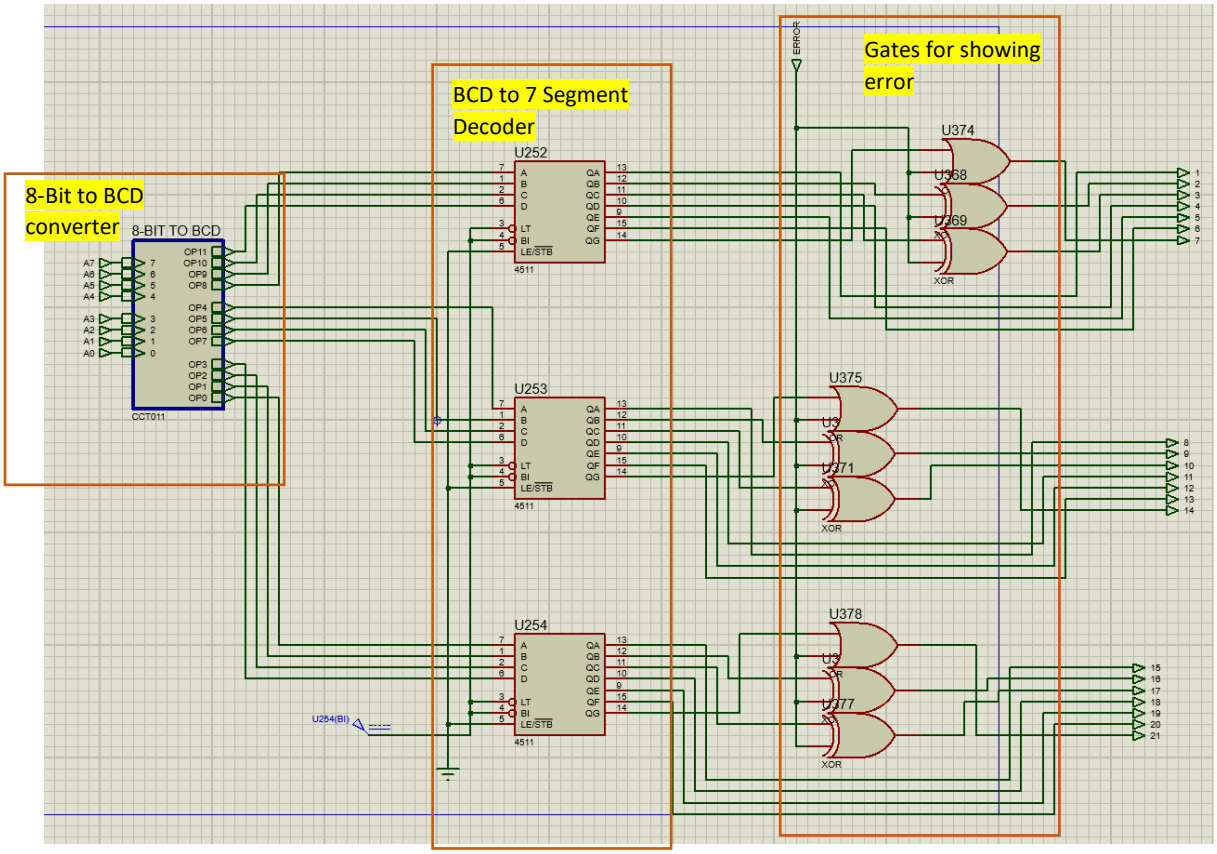
7- Segment Display:

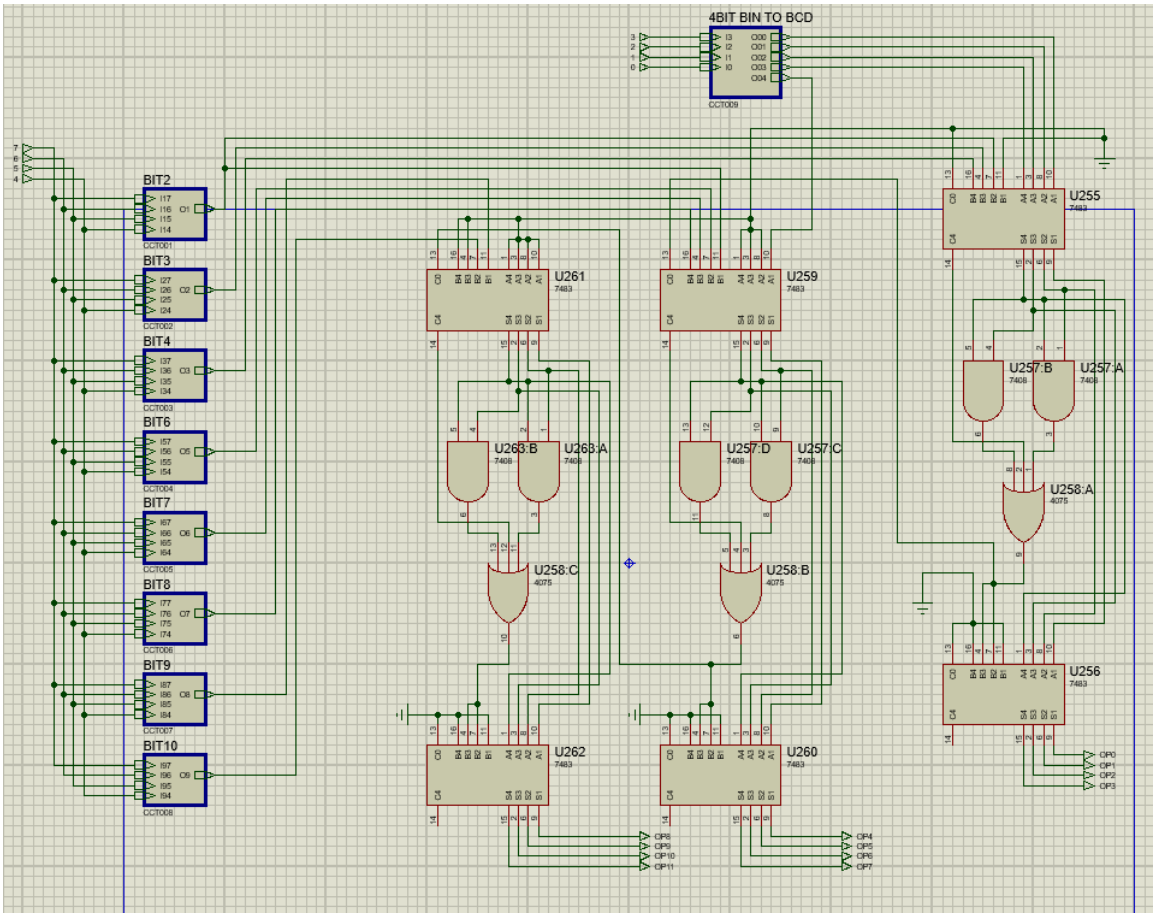


The 7-segment display is a digital display which shows the output in decimal in three different digits. If there is any error in ALU, it shows E in all the displays.

Working Principle of the 7-segment display:

- The 8-bit result is obtained from the output register
- It is expressed as the sum of upper 4-bit and lower 4-bit
- The upper 4-bit can express decimal numbers in multiples of 16 as shown in the black truth table while the lower 4-bit can express 0 to 15 as shown in the red one.
- The with binary adders, a BCD adder is made with a checker circuit. The checker circuit adds 6 to the result if the result is more than 9
- The upper 4-bits and lower 4-bits are then added in the BCD adders, digit-wise and for each digit, 4 outputs are obtained
- These 4 outputs are connected with a BCD to 7-segment decoder. Thus, 3 decoders for 12 outputs
- Then theses 21 outputs are connected to the displays
- The error bit is designed in such a way that only the bars for showing “E” will light up when the ERROR signal is active


8-Bit to BCD converter
8-BIT TO BCD
BCD to 7 Segment Decoder
Gates for showing error
ERROR
U252
U253
U254
U254(BI)
4511
U374
U368
U369
U375
U371
U378
U377
XOR


4BIT BIN TO BCD
BIT2
BIT3
BIT4
BIT6
BIT7
BIT8
BIT9
BIT10
U261
U259
U255
U257:B
U257:A
U263:B
U263:A
U257:D
U257:C
U258:A
U258:C
U258:B
U256
U262
U260

Parts Used:

- 7-segment display
- BCD to 7-segment decoder
- XOR, OR, AND, NOT Gates
- Binary adder

Inputs:

- A0 to A7: From Output register
- ERROR: from ALU, for showing Error

Outputs:

- 1 to 21: To 7-segment displays

Example:

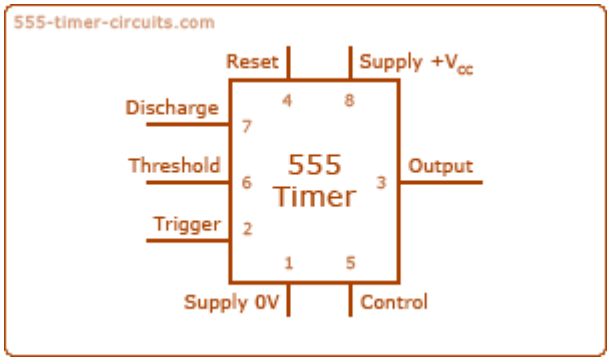
0101 0011
= 0101 0000 + 0000 0011

				Digit 3 in BCD				Digit 2 in BCD				Digit 1 in BCD				
I7 (128)	I6 (64)	I5 (32)	I4 (16)	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	16
0	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	32
0	0	1	1	0	0	0	0	0	1	0	0	1	0	0	0	48
0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	64
0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	80
0	1	1	0	0	0	0	0	1	0	0	1	0	1	1	0	96
0	1	1	1	0	0	0	1	0	0	0	1	0	0	1	0	112
1	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	128
1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	144
1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	160
1	0	1	1	0	0	0	1	0	1	1	1	0	1	1	0	176
1	1	0	0	0	0	0	1	1	0	0	1	0	0	1	0	192
1	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	208
1	1	1	0	0	0	1	0	0	0	1	0	0	1	0	0	224
1	1	1	1	0	0	1	0	0	1	0	0	0	0	0	0	240

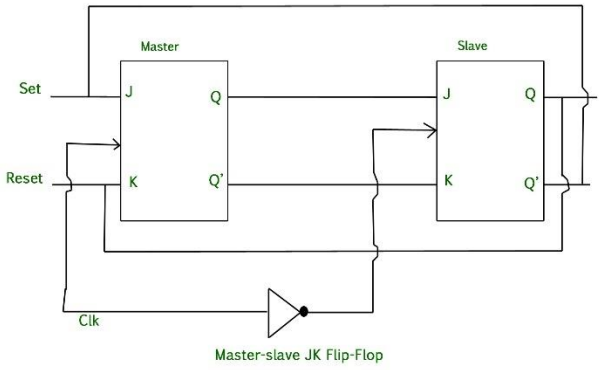
Input				Output				
I0	I1	I2	I3					
0	0	0	0	0	0	0	0	0
0	0	0	1	0				1
0	0	1	0	0				0
0	0	1	1	0				1
0	1	0	0	0				0
0	1	0	1	0				1
0	1	1	0	0				0
0	1	1	1	0				1
1	0	0	0	0				0
1	0	0	1	0				1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

ICs used:

- 555 Timer IC: It is used for the clock module. There are basically three equal resistances inside the IC which divides the voltage in three parts. Then by comparing it with the trigger and threshold input with a comparator, capacitors are charged and discharged simultaneously and pulses are provided accordingly.

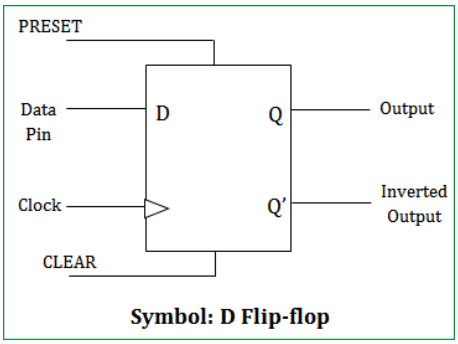


- 74111: It is a master-slave JK flip-flop used in constructing the counters.



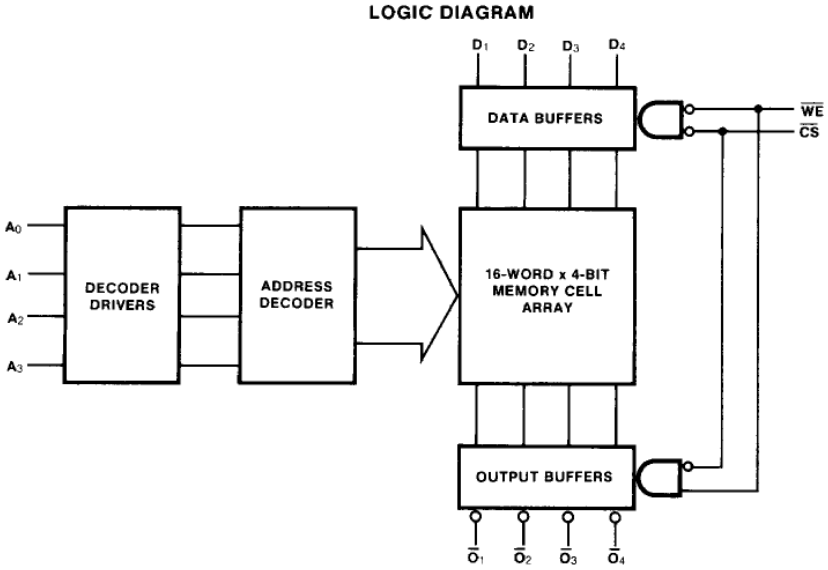
Master-slave JK Flip-Flop

- 74LS74: D Flip-Flops used in making registers.

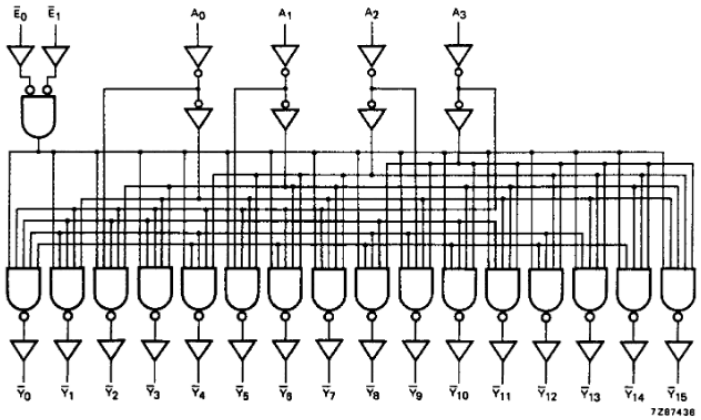


Symbol: D Flip-flop

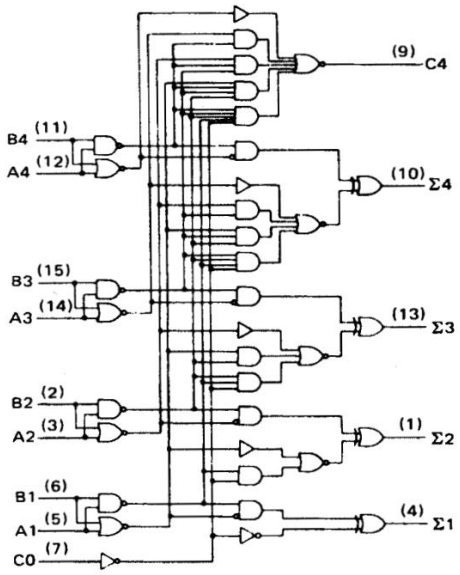
- 74LS189: 64-bit RAM IC used in constructing 128-bit RAM



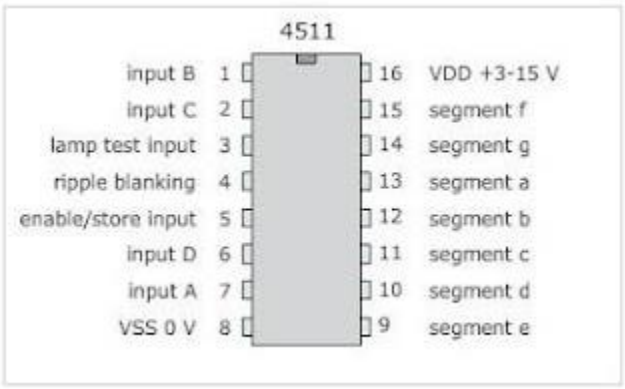
- 74HC154: A 4-to-16 decoder used in the controller-sequencer



- 74LS283: 4-bit binary full adders with fast carry used in ALU



- 4511: BCD to 7-segment decoder



- 7483: Similar to the 74LS283 IC which is a 4-bit binary full adder with fast carry.

Improvements:

Several improvements were made in the project like

- Reducing the T-states by one stage in each instruction
- Using a 7-segment display to show the output in decimal
- Optimization in the Controller-sequencer by reducing the number of gates as much as possible
- Showing error in the display whenever a 9-bit or negative result is obtained

Problems Faced:

While performing the project there were some problems faced by us which can be avoided if known before. Some of them were:

- Keeping the order of MSB and LSB same in each component
- Taking care of the active low and active high inputs
- Understanding the relation between Controller-sequencer and the other components so that the computer can run swiftly between the negative and positive clock pulses
- Connecting the separate components in the BUS
- Logic contentions

Discussion:

- Outcomes:
- Limitations:
- Future Development: