EE2080 Microprocessor Systems Laboratory

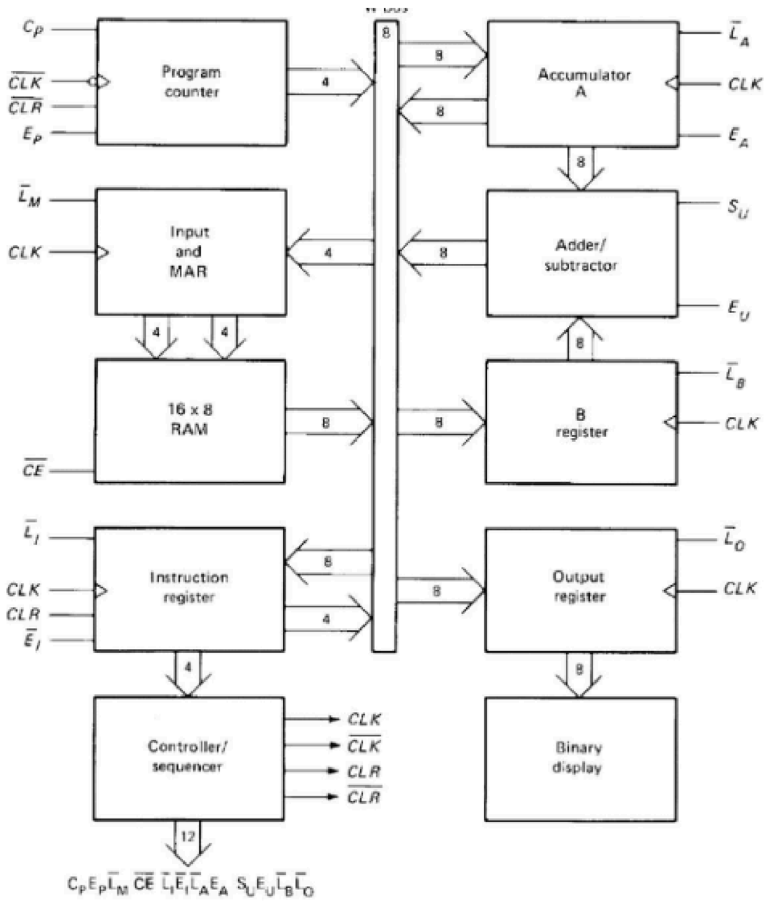
8-bit Microcomputer Using Logisim

Abstract and description

Develop an 8-bit microcomputer on Logisim–Evolution using basic components.
a. Evaluation of individual components such as program counter, registers, random access memory (RAM), arithmetic logic units (ALUs) and output interfacing like seven segment display.
b. Evaluation of an integrated 8-bit microcomputer.
The 8 bit microcomputer developed here can do basic addition and subtraction and it will display the output in a 7 segment display. The assigned operations for each unit are performed during their respective clock cycles.
The experiment is performed in Logisim – Evolution is a free, open-source and cross-platform educational software for designing and simulating digital logic circuits.

Introduction

The Architecture



We can see various blocks like Program Counter, MAR, RAM, Instruction Register, CU, Accumulator, ALU, Registers and Display. Each block has its own control pins and it will be activated through the clock cycles from CU. The output of each block is not connected directly to the 8-bit bus. This will create short circuits between the bus and block. So to avoid it, we connected 3 state buffers to the outputs. For the block with bit output, we used a splitter to connect it to the bus.

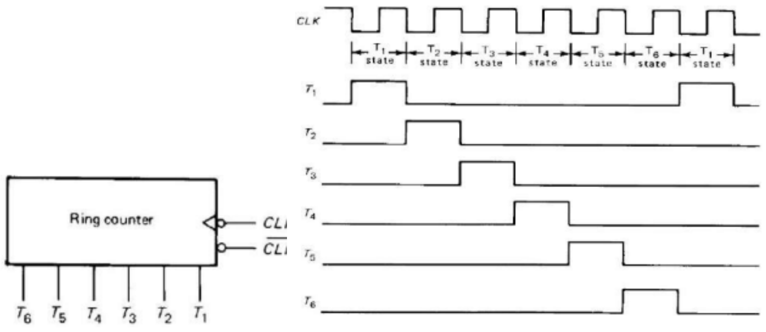
The Instructions and the Operations

We assign a set of 5 instructions - LDA, ADD, SUB, OUT, HLT.

Mnemonic	Op Code	Operation
LDA	0000	Loads the RAM data into the accumulator.
ADD	0001	Adds the data in the RAM to the data already present in the accumulator.
SUB	0010	Subtract the data in RAM from the accumulator
OUT	1110	Results in loading of data from the accumulator to the output register
HLT	1111	Stops the process

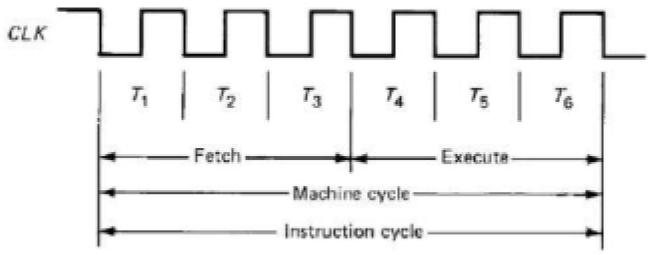
The basic operations of the microprocessor consist of six timing states. The first three states are for the fetch cycle will fetch data/instructions from main memory then store them in other memory areas. The next three states are the execution cycle which executes according to the instructions. The ring counter inside the CU facilitates in achieving the six states.

Block Diagrams and Timing Diagrams



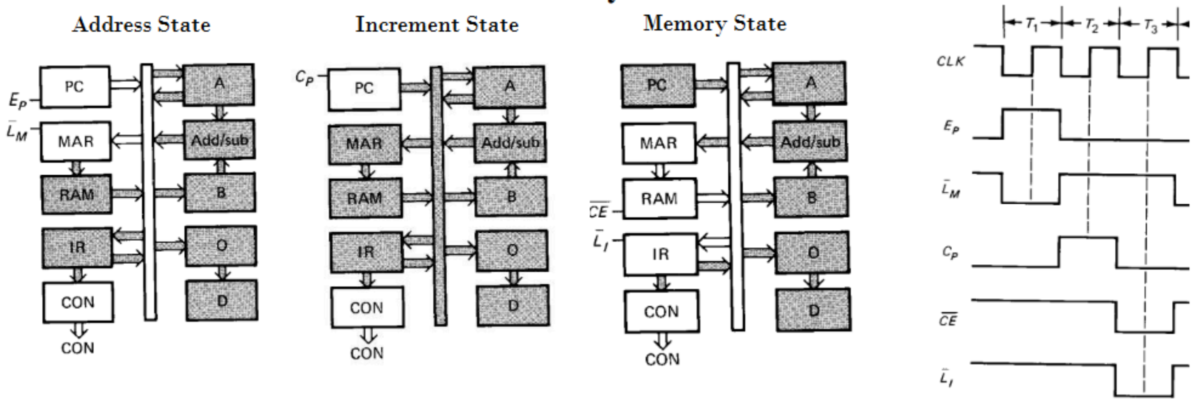
Clock and timing signals of counter

There are 3 states for each OpCode. In some operations, 3 states are not required, but for the sake of uniformity, we include 3 states for each operation.



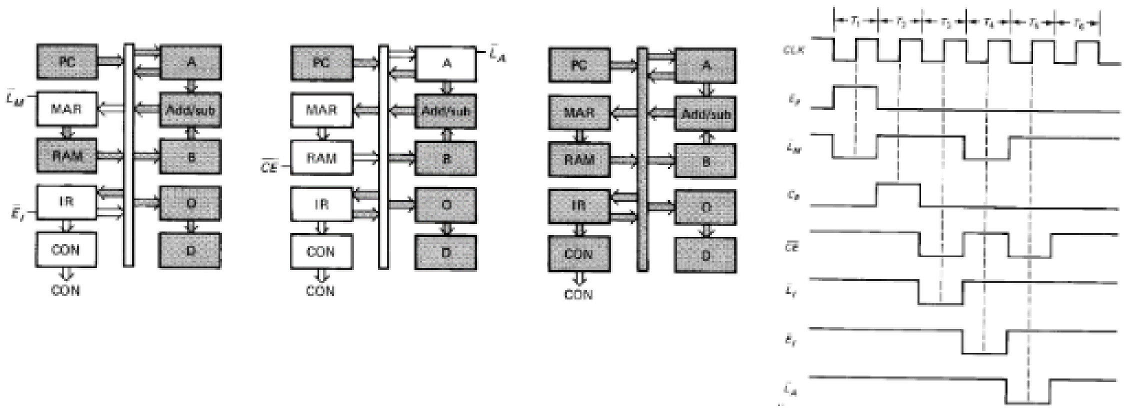
6 clock cycle(Fetch+Execute)

Fetch Cycle



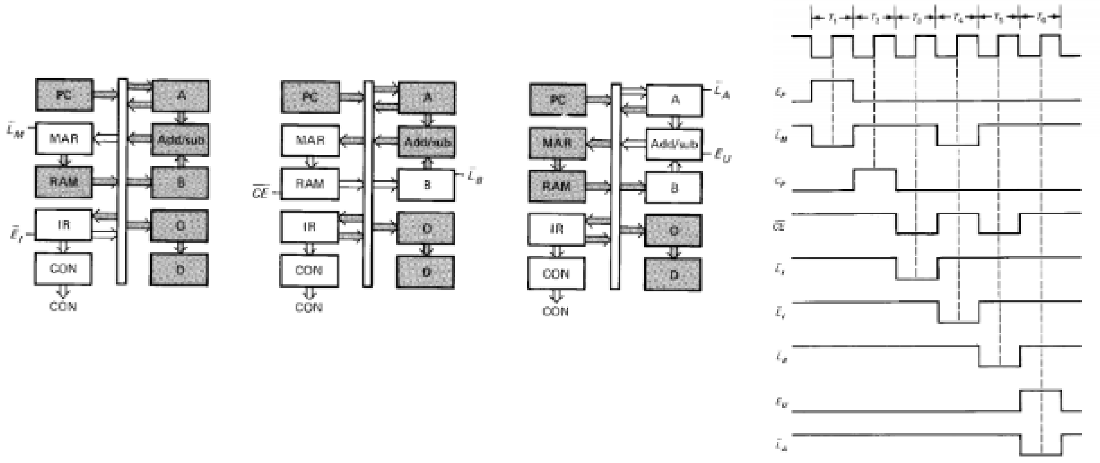
T1 state: The connection is established between PC and MAR by making Ep and Lm' high.
T2 state: Cp is on and the counter starts counting
T3 state: The data in MAR is directed to ROM and the data is transferred to IR. Corresponding pins are ON based on their functions.
The value from PC goes to MAR and it will point toward the respective address in the ROM.

Execution Cycle – LDA



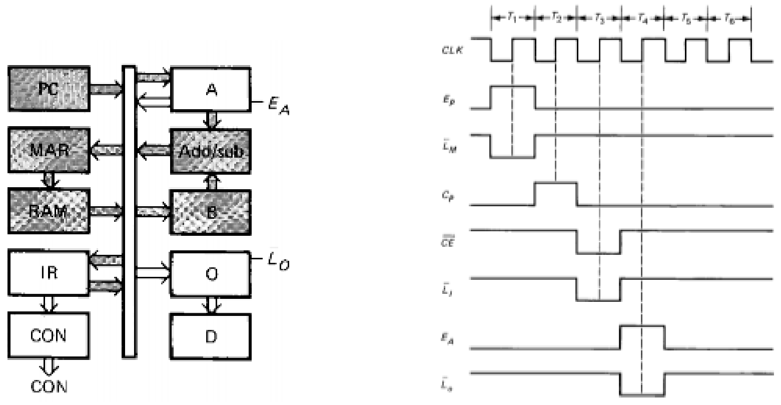
T4 state: The connection is established between IR and MAR. The memory address from IR is given to MAR to point on the assigned data.
T5 state: MAR points to the corresponding data and it will be send to Accumulator
T6 state: Idle state

Execution Cycle – ADD/SUB



T4 state: The connection is established between IR and MAR. The memory address from IR is given to MAR to point on the assigned data.
T5 state: MAR points to the corresponding data and it will be send to B register
T6 state: The data from Accumulator and B register is given to ALU where it performs necessary arithmetic operations.

Execution Cycle – OUT



T4 state of OUT

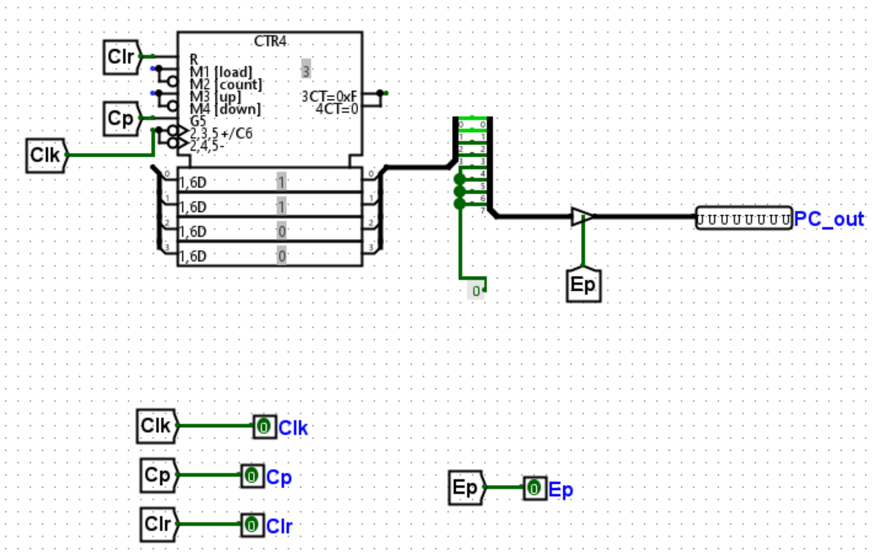
Even we take 3 cycles for the operations, effectively there is only one, i.e, T4 state
T4 state: The connection is established between OutRegister and Accumulator. The data from OutReg is passed to a display.
T5 state: Idle state
T6 state:Idle state

State Diagram

Instruction	State	Cp	Ep	Lm’	Ce’	Li’	Ei’	La’	Ea	Su	Eu	Lb’	Lo’	Hex Code
	T1	0	1	0	1	1	1	1	0	0	0	1	1	5E3
Fetch	T2	1	0	1	1	1	1	1	0	0	0	1	1	BE3
	T3	0	0	1	0	0	1	1	0	0	0	1	1	263
	T4	0	0	0	1	1	0	1	0	0	0	1	1	1A3
LDA	T5	0	0	1	0	1	1	0	0	0	0	1	1	2C3
	T6	0	0	1	1	1	1	1	0	0	0	1	1	3E3
	T4	0	0	0	1	1	0	1	0	0	0	1	1	1A3
ADD/SUB	T5	0	0	1	0	1	1	1	0	0	0	0	1	2E1
	T6	0	0	1	1	1	1	0	0	0/1	1	1	1	3C7/3CF
	T4	0	0	1	1	1	1	1	1	0	0	1	0	3F2
OUT	T5	0	0	1	1	1	1	1	0	0	0	1	1	3E3
	T6	0	0	1	1	1	1	1	0	0	0	1	1	3E3

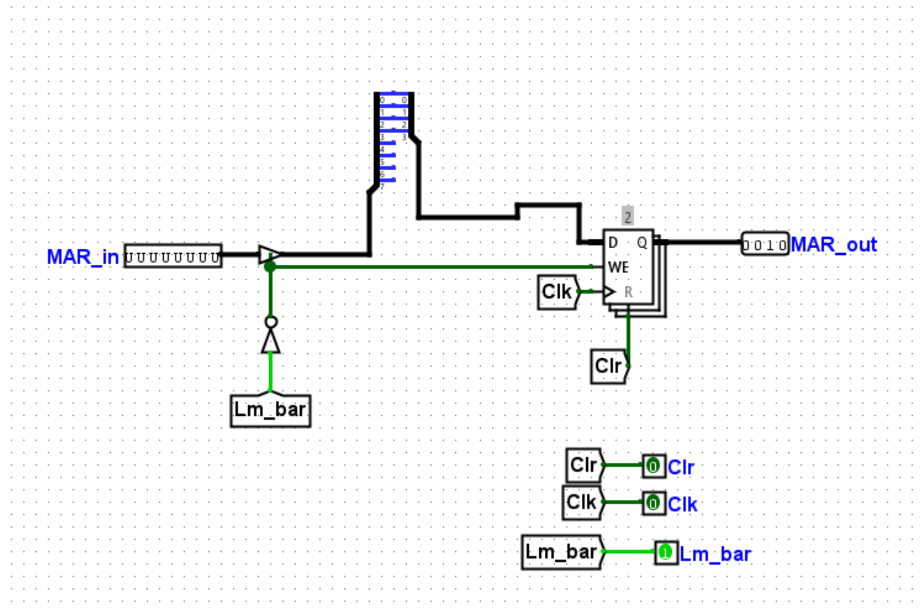
Implementation and Results

1.Program Counter



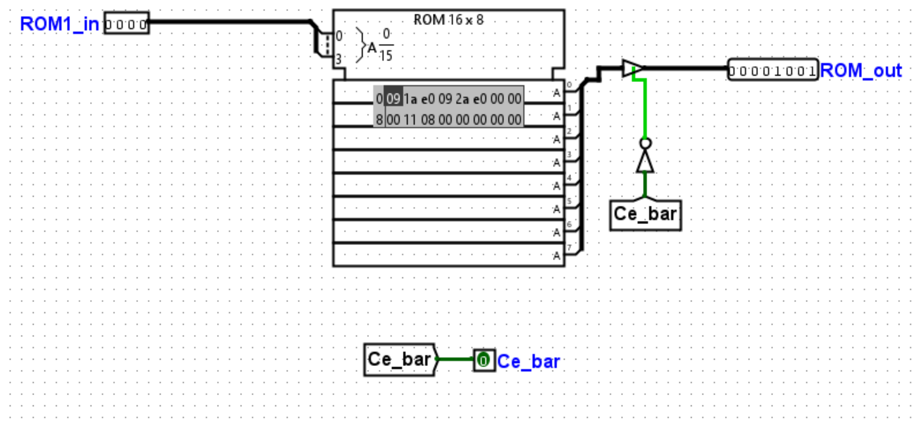
The above program counter is a falling edge clock counter. It has 4 bits,ie, it will count from 0 to 15 or in hexadecimal terms 0 to f. It is enabled by pin Cp and is connected to the 8 bit bus by a three state buffer Ep. Since the output given by PC is 4 bit, it is connected to a 8 bit wire splitter in which the first 4 MSB's are connected to ground.

2.Memory Address Register



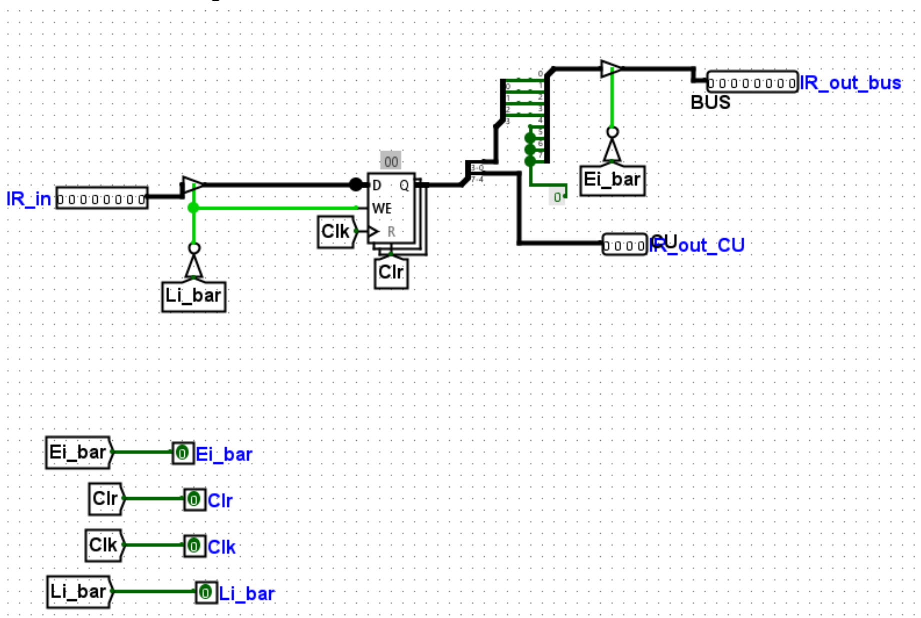
MAR functions as a pointer to the addresses of the 8 bit data in 16 address locations in ROM. Here, MAR is a 4 bit register that stores the 4 bit value from the bus when the clock is on and pin Lm'(the tri-state buffer connected directly to the bus) is 0. The MAR is directly connected to RAM so the instructions are directly transferred to RAM.

3.Read Only Memory



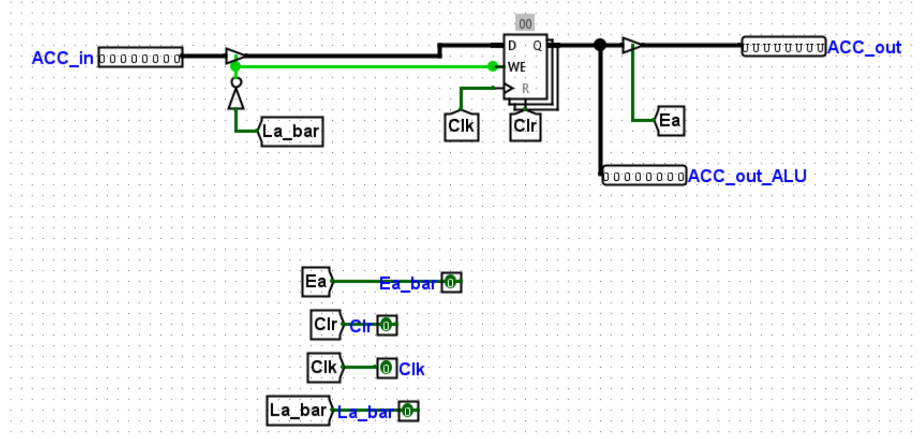
RAM is a memory device used to store data. Here, the size of RAM is 16*8,i.e. 16 addresses of 8 bits each. It can be connected to the bus using the buffer pin Ce'.

4.Instruction Register



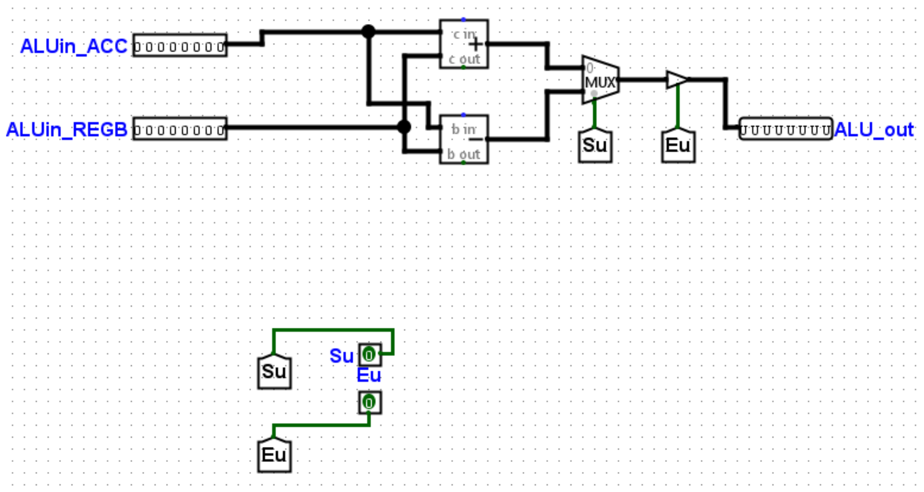
IR is an 8 bit register. It takes in 8 bit data from bus(, controlled by pin Li’), divides it into two 4 bits and directs first 4 MSBs to bus(, controlled by Ei’) and next 4 to CU.

5.Accumulator



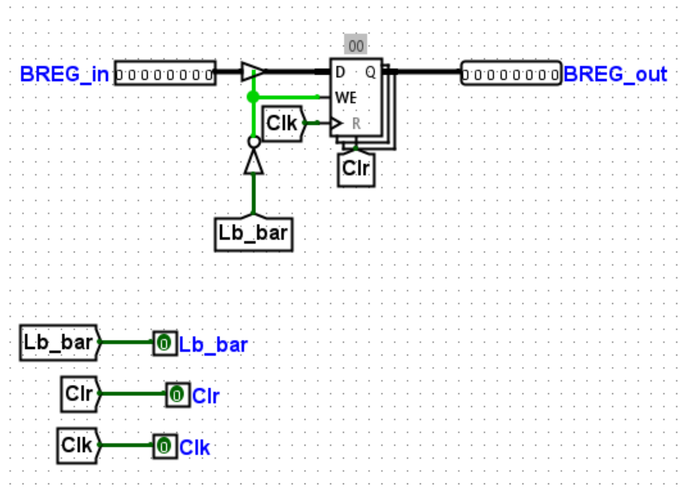
The accumulator takes in data from ROM and delivers it to the ALU. It is controlled by enable pins La’ and Ea.

6.Arithmetic Logic Unit



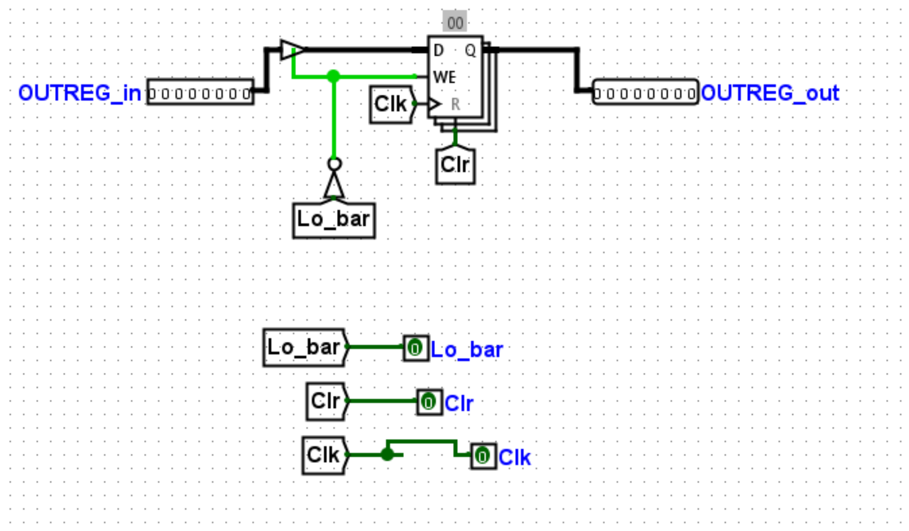
ALU is the unit which performs arithmetic operations like addition and subtraction. It comprises an adder and a subtractor that are connected to a multiplexer. By controlling the enabling pin in MUX, i.e, when Su pin is 0, it will perform add operation and when Su is 1, it performs subtraction. It has a tristate buffer Eu connected to the bus

7.Register B



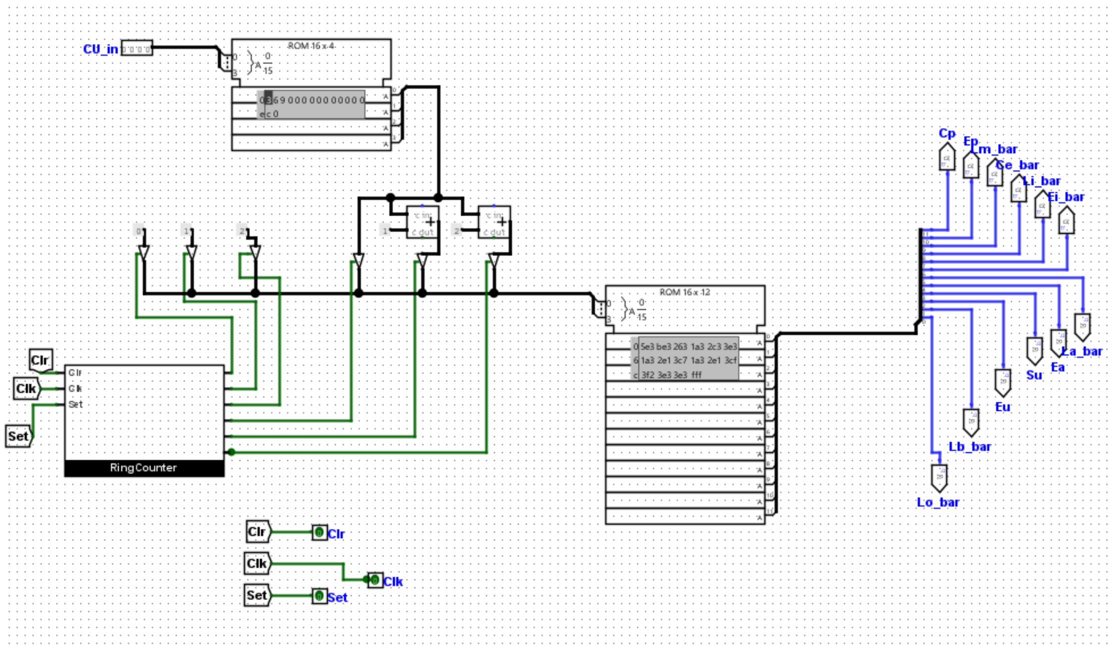
The B register takes the second value from ROM and then delivers it to the ALU for assigned operations. Enable pin is Lb'.

8.Out Register



Output Register is a register used for displaying the output data that we got after performing the logical/arithmetic operations.

9.Control Unit



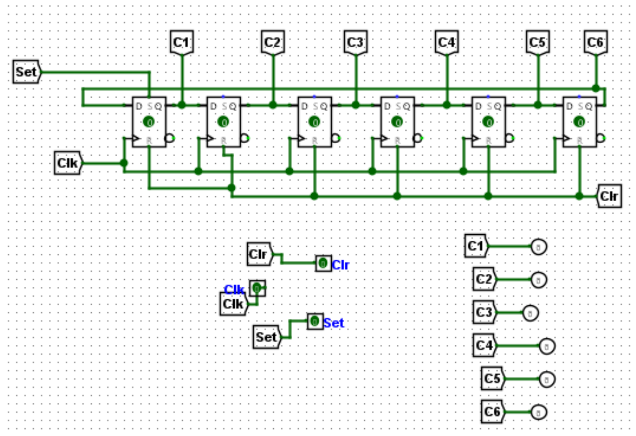
The CU is the main controlling unit of the whole circuit. It sends out instructions assigned to each block.

Detailed explanation of CU

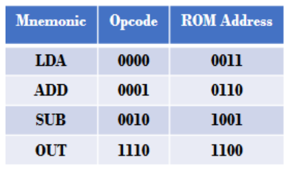
CU comprises of :

- Ring counter- Here it is constructed using D flip flops. The output of the last flip-flop is connected to the input of the first flip-flop. The main point of this is that it circulates a single one (or zero) bit around the ring.There is a set pin connected at preset of the first flip flop. It will activate the counter. The Logisim version is shown below.

Here it count upto 6, ie, we will get 6 states inside CU

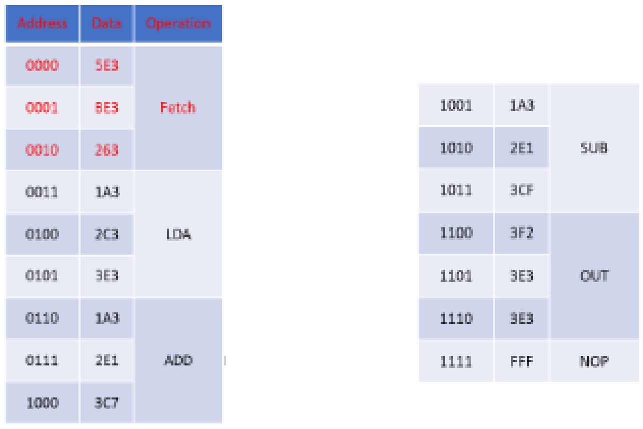


- Op code ROM(16X4)- This ROM is used as a memory for the operational codes so that for each assigned code, we will write corresponding data to the address.
- Control ROM(16X12)- This ROM contains Hexadecimal Codes assigned for each operation.



Mnemonic	Opcode	ROM Address
LDA	0000	0011
ADD	0001	0110
SUB	0010	1001
OUT	1110	1100

ROM for OpCodes



Address	Data	Operation
0000	5E3	Fetch
0001	BE3	
0010	263	
0011	1A3	LDA
0100	2C3	
0101	3E3	
0110	1A3	ADD
0111	2E1	
1000	3C7	
1001	1A3	SUB
1010	2E1	
1011	3CF	
1100	3F2	OUT
1101	3E3	
1110	3E3	
1111	FFF	NOP

ROM for CU

- The arithmetic logic circuits- We can see that there are 6 tristate buffers and 2 adders

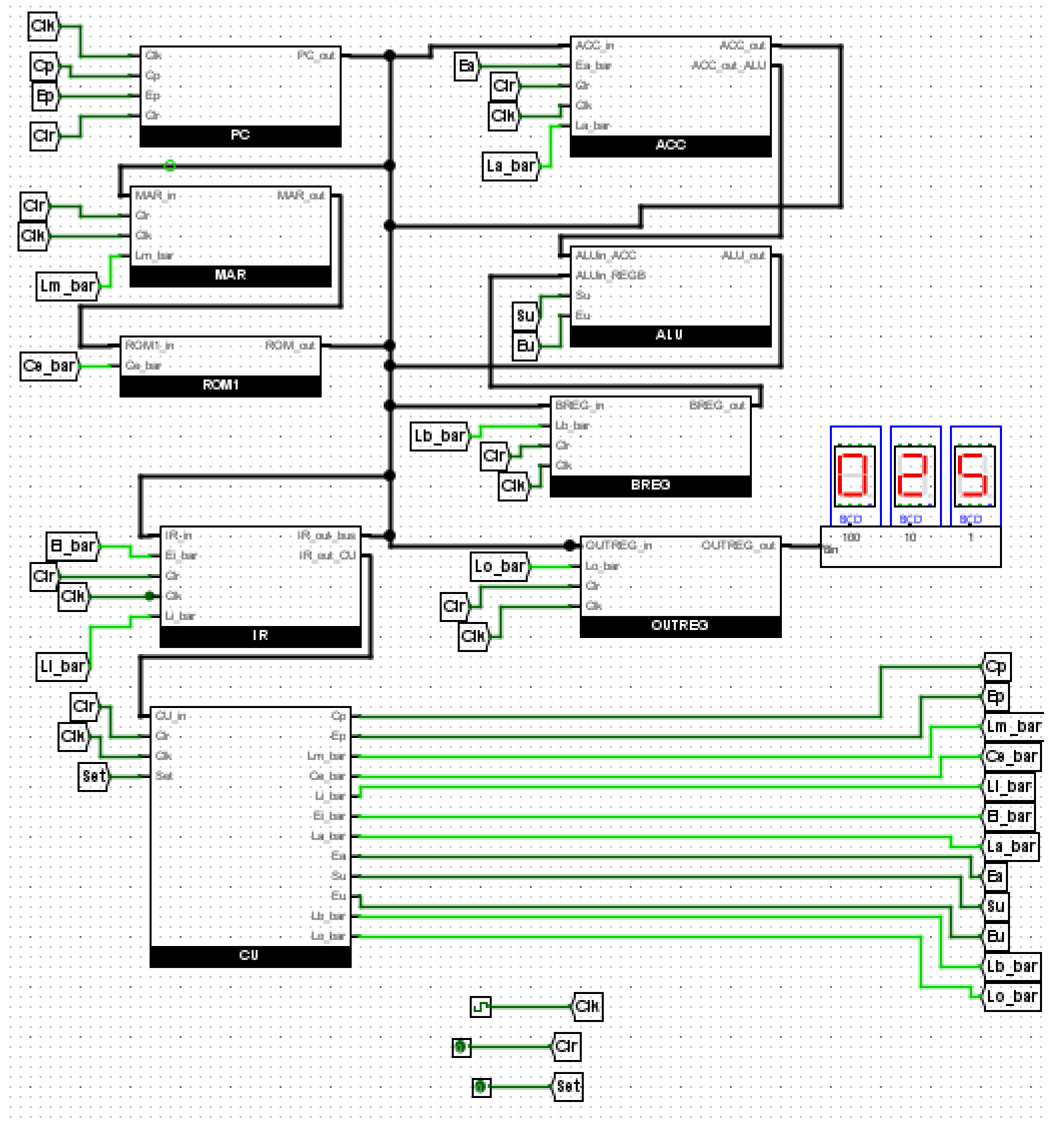
Pointing to CU ROM

1. First 3 counts
When the clock gives the first clock output, the first tristate buffer activates and 0X0001(hexa) will go to CU ROM. This continues for count 2 and 3. In count 2, it will

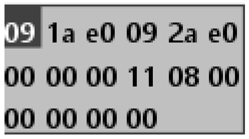
activate the 2nd tristate buffer and 2 will go to CU ROM . In count 3, it will activate the 3rd tristate buffer and 3 will go to CU ROM.

2. Next 3 counts
   From the 4th count, it will activate 4th tristate and the data from Op ROM, under the instructions from IR, will point toward the respective location in CU ROM. In Op ROM we can see 3, 6, 9 which are addresses for corresponding Op codes. The same will happen to 5th and 6th tristate buffers under 5th and 6th counts respectively with a small difference of addition of 1 and 2 in corresponding buffers

<u>Fully Integrated Block</u>



The operations that are performed in this microcomputer are :

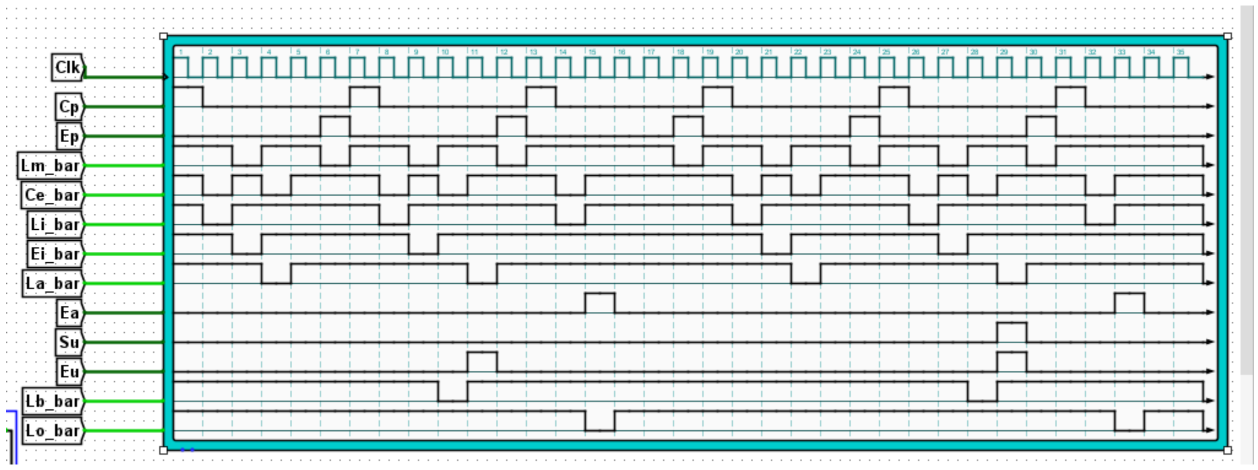


```
09 1a e0 09 2a e0
00 00 00 11 08 00
00 00 00 00
```

Data inside ROM

1. 09: LDA 9- Load the data from 9th position in ROM
2. 1a: ADD a- Add the data to B register from 10(a)th position in ROM
3. e0: OUT 0- Display the output
4. 09: LDA 9- Load the data from 9th position in ROM
5. 2a: SUB a- Subtract the data which is located at 10(a)th position in ROM
6. e0: OUT 0- Display the output

<u>Note:</u>
We can see the output of ADD in 16th cycle and SUB in 34th cycle. The reason is that for the first ADD, FLFAFO(Fetch,Load,Fetch,Add,Fetch,Out) will be performed, i.e, there are 18 cycles but since the last 2 states are idle, we can see output in 16th state itself.
Similarly after the 18th state, it will take another 18 cycles, i.e 36 cycles. Since the last 2 states are idle, we can see the SUB output in the 34th cycle.
The recordings from oscilloscope is attached below



<u>Conclusion And Key Learnings</u>

- The CU is the ultimate controller of the whole system.
- Fetch is performed before every execution cycle
- Even if some cycles are not necessary, it is performed for regularity
- Became familiar with Logisim