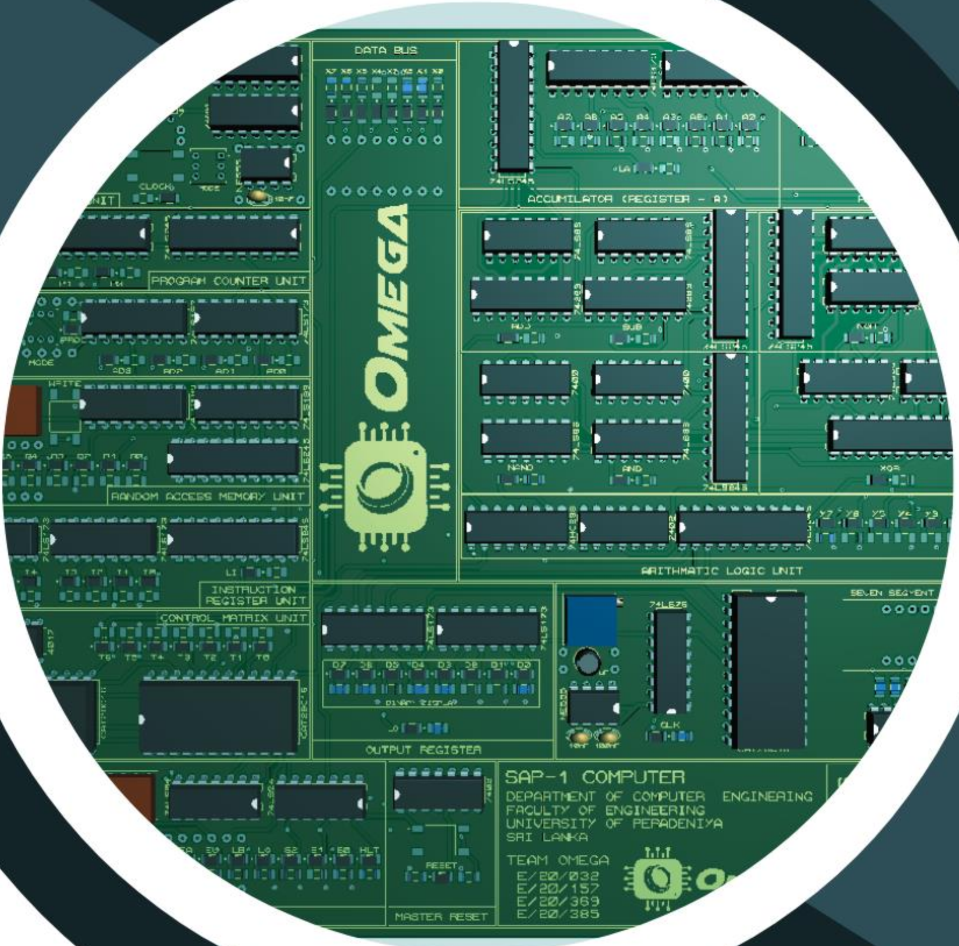
OMEGA

SAP – 1 COMPUTER

PROJECT REPORT



CO221: DIGITAL DESIGN

08.09.2024

Table of Contents

Project Team

- S.M.B.G. Janakantha
- A.M.N.C. Bandara
- G.S. Seneviratne
- D.H. Sriyarathna

Supervisor

- Dr. Kamalanath Samarakoon

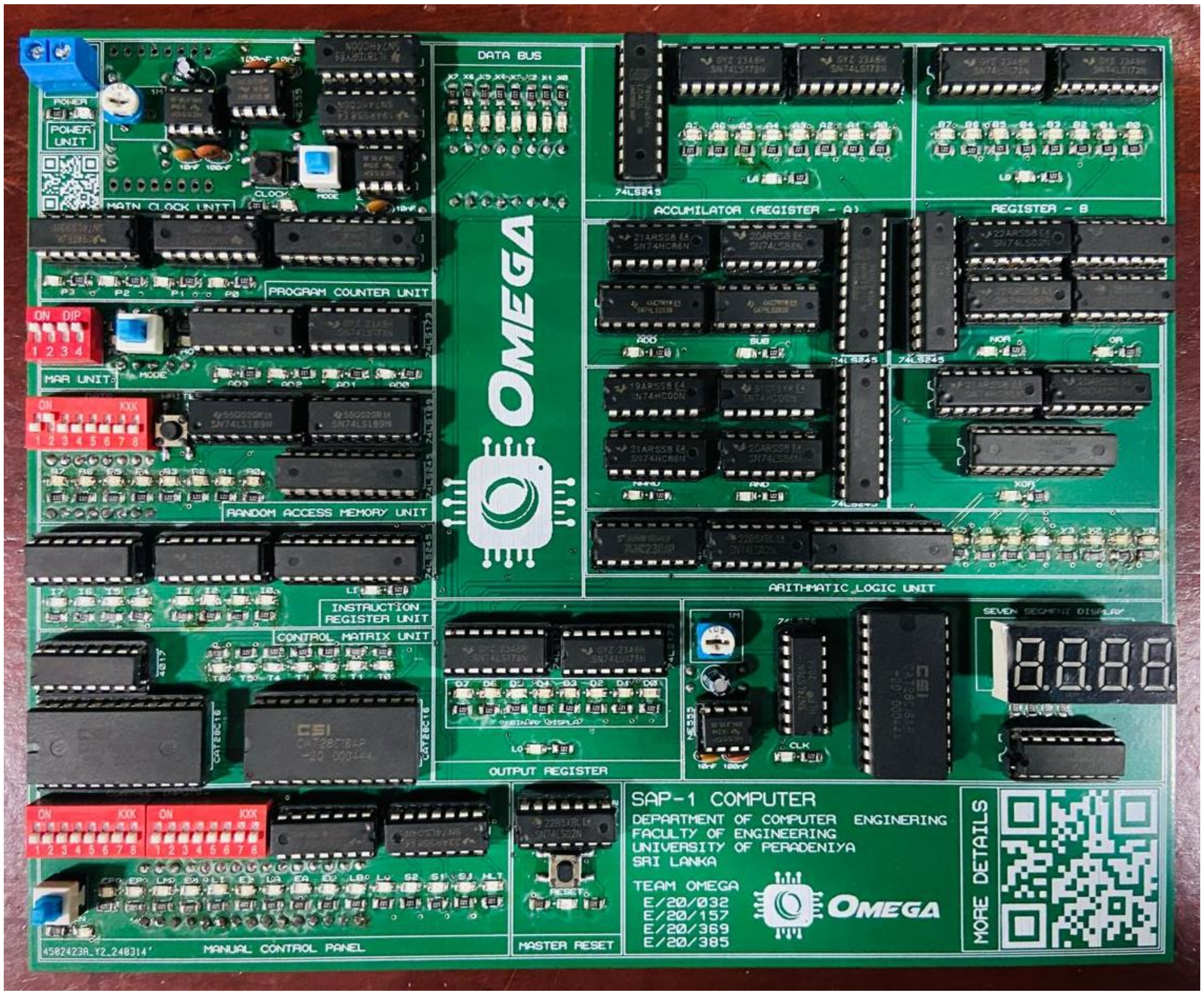


Figure 1: SAP-1 Computer

Introduction

SAP-1, standing for "Simple As Possible," is the first step in understanding the core principles of computer architecture. The SAP-1 model is a minimalistic design that introduces fundamental concepts of how a computer processes information. Although simple, SAP-1 lays the groundwork for more complex systems by demonstrating basic instruction execution, data movement, and control flow.

SAP-1 has a limited instruction set, usually comprising a few basic operations like LOAD, ADD, SUB, IN, OUT, and HLT (halt). This helps beginners understand how instructions are fetched, decoded, and executed in a sequential manner.

Specification:

- 8-bit Bus
- 4-bit Program Counter
- 4-bit Memory Address Register (MAR)
- 16-byte Memory
- 8-bit Instruction Register (IR)
- 6-cycle micro-instructions with 16-bit words
- 8-bit Accumulator
- 8-bit B register
- 8-bit Output Register
- Arithmetic and Logical unit (ALU)
  - Full Adder / Subtractor
  - XOR
  - AND / NAND
  - OR / NOR
- Seven Segment Display Output

Design Procedure

The design of the SAP-1 (Simple As Possible) computer involves a modular approach where each component plays a crucial role in the computer's overall functionality. Each module was independently designed, tested, and finally integrated onto a single Printed Circuit Board (PCB). The design was carefully structured to ensure that the computer operates efficiently and maintains the integrity of data flow across the system.

Modules

- Clock module
- Program Counter
- Arithmetic and Logical Unit (ALU)
    - Full Adder/Subtractor
    - AND / NAND
    - OR / NOR
    - XOR
- Output register
- Registers
    - Register A
    - Register B
    - Instruction Register
- Memory Address Register (MAR)
- Seven Segment Display
- RAM
- Control sequencer

All the modules were designed by using Proteus and fabricated on a 210mm * 180mm multi-layer PCB board

Clock Module

The computer’s Clock Module is used to synchronize all operations (there are some asynchronized operations within some modules, eg: ALU, even though most of the control instructions were synchronized).

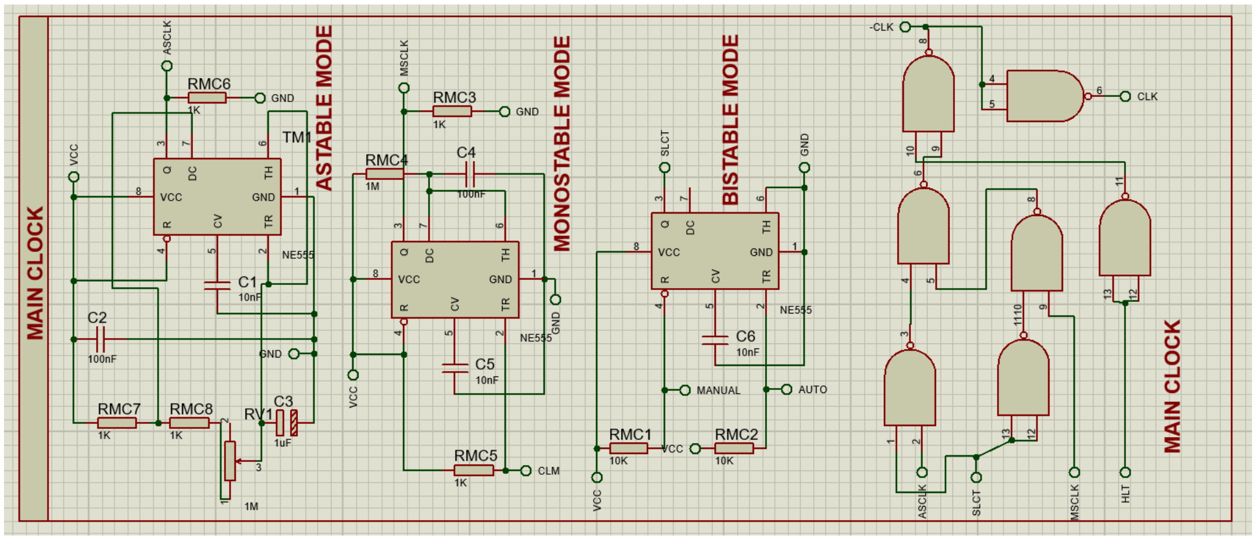


Figure 2: Main Clock

There are two main modes in a clock module.

- **A-stable mode**

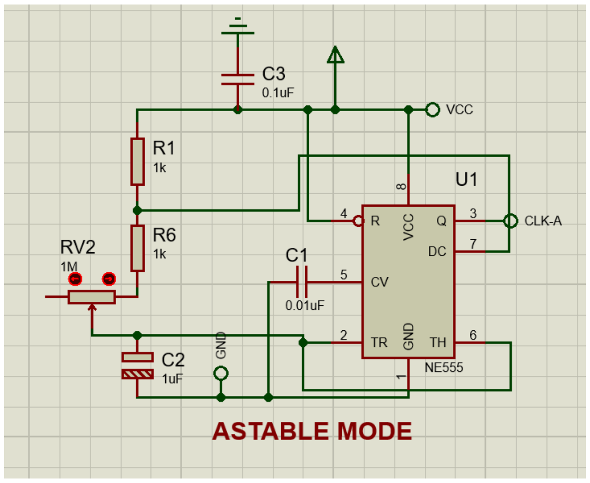


Figure 3- A-stable mode design

In Astable mode, a clock module does not have any stable states. Instead, it continuously oscillates between two states, generating a periodic waveform (usually square).

- Behavior: The circuit continuously switches between high and low states without any external trigger.
- Output: Produces a continuous square wave signal.
- Application: Used for generating clock pulses, timing signals, and waveform generation.

A 555 timer IC configured in Astable mode will produce a continuous square wave signal, which can be used as a clock signal for digital circuits.

- **Mon stable mode**

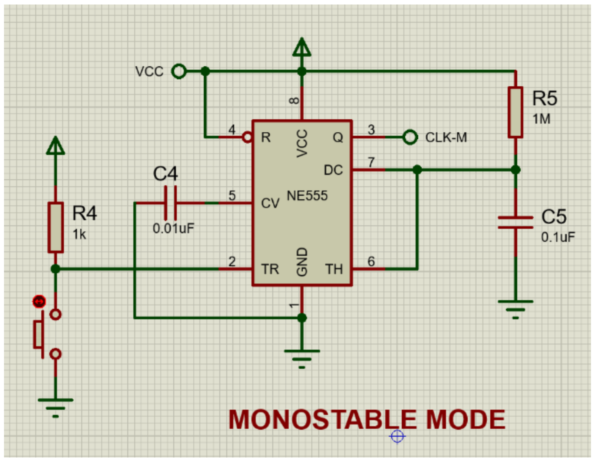


Figure 4 - Monostable mode

In monostable mode, a clock module has one stable state and one unstable state. When triggered by an external signal, it temporarily switches to the unstable state and then returns to the stable state after a predetermined period.

- Behavior: The circuit stays in the stable state until it receives an external trigger. After being triggered, it switches to the unstable state for a specific duration before returning to the stable state.
- Output: Generates a single pulse of a defined width when triggered.
- Application: Used in timing applications where a single output pulse is required, such as timers, delay circuits, and pulse width modulation (PWM).

A 555 timer IC configured in monostable mode will generate a single output pulse of a specified duration when triggered, useful for creating time delays or single-shot pulses.

- **Bi stable mode**

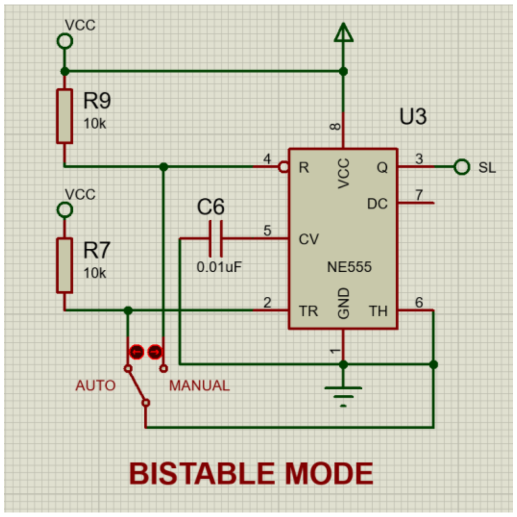


Figure 5: Bistable Mode

In bistable mode, a clock module has two stable states. The circuit can switch between these states based on external inputs and will remain in either state indefinitely until an input causes it to switch.

- Behavior: The circuit remains in one of two stable states until an external input triggers a change to the other state.
- Output: Maintains a stable high or low output until an input causes a state change.
- Application: Used in applications requiring memory storage, such as flip-flops, latches, and data storage elements in digital circuits.

Halt status

To halt the computer programmatically, the halt state can be used. Normally the halt input is a low state and it is inverted by a NOT gate

Program Counter

The program is stored in the RAM as listed instructions. To execute them, the instructions must be fetched from the RAM. So, the program counter is used to keep track of which address is where. It contains the address, or the location of the instruction being executed at the current time.

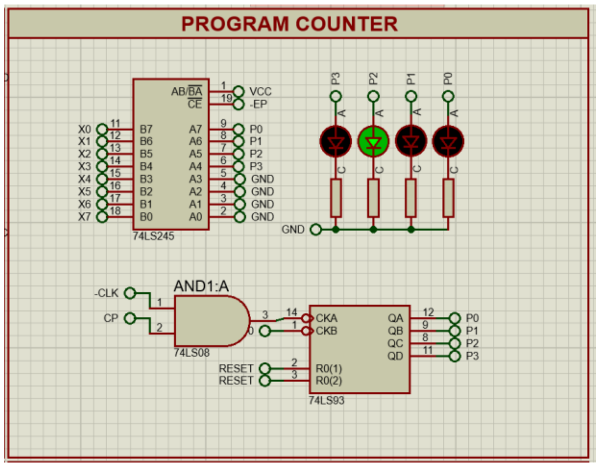


Figure 6: Program counter

This is a 4-bit counter as the memory has a 4-bit address. This works similarly to register A or register B because it stores a value, and sends and receives values from the bus. The program counter contains 3 control signals.

- **Program Count Out (CO):** Sends the stored value to the bus.
- **Jump (Program Counter In):** Normally, the program counter increments sequentially starting from 0. This signal is used to jump to a different address or loop. When activated, it takes the 4 least significant bits from the bus (since it's a 4-bit counter) and assigns them to the program counter. This sets the next address to be executed.
- **Count Enable (Increment):** This increases the program counter once per instruction cycle, not every clock cycle (since an instruction may require multiple clock cycles). When this signal is enabled, the counter increments on each clock cycle to fetch the next instruction, then the signal is disabled to pause further incrementing while the current instruction is executed.

Arithmetic and Logical Unit

SAP-1 is capable of doing below arithmetic and logical operations. Register A and B is used to feed data to ALU

- Full Adder/Subtractor
- AND / NAND
- OR / NOR
- XOR
- MUX

- Components and Architecture

  The ALU was constructed using basic logic gates and a cascaded 4-bit adders. The design allows for the performance of basic arithmetic operations such as addition and subtraction, along with logical operations like AND, OR, XOR, and their negated versions (NAND, NOR). The use of multiplexers (MUX) enables the selection of different operations based on control signals from the Control Sequencer.

- **Full Adder/Subtractor**

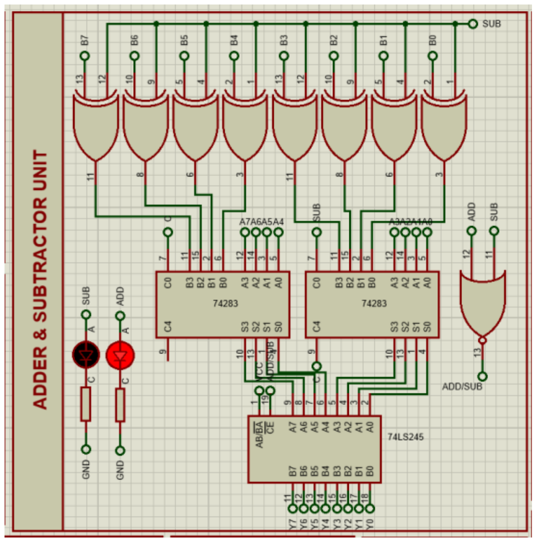


Figure 7: Adder and Subtractor

- **AND / NAND**

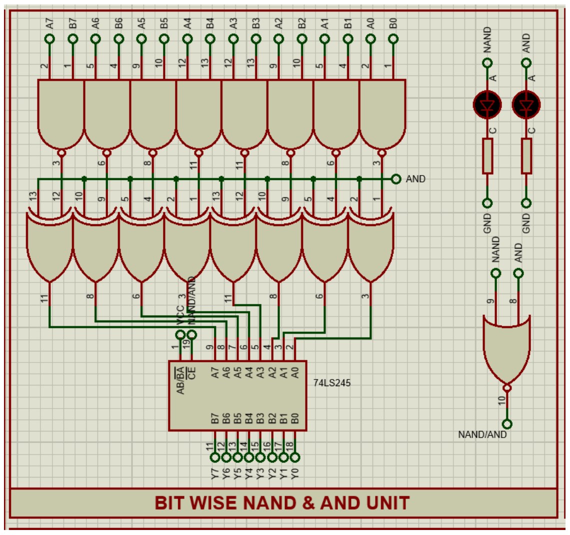


Figure 8: Bit wise Nand & and Unit

- **OR / NOR**

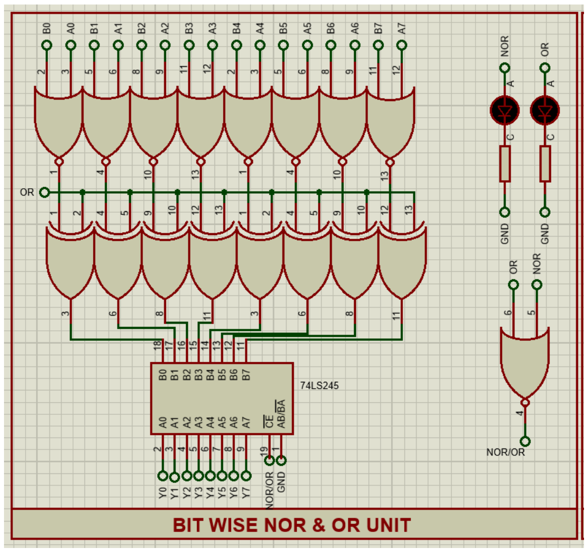


Figure 9: Bit wise NOR & OR unit

- **XOR**

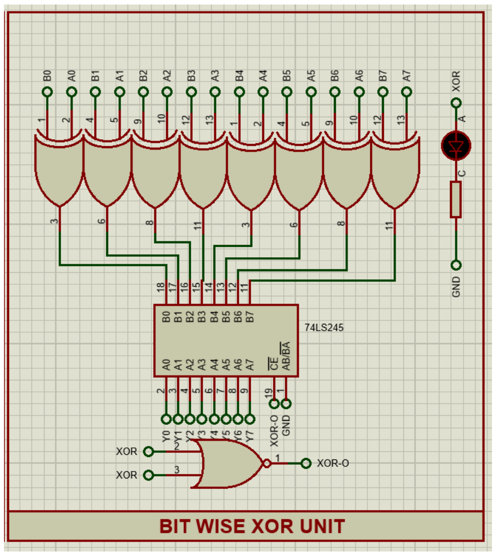


Figure 10: Bit wise XOR unit

Output Register

It uses two 74LS173 4-bit registers, which together form an 8-bit register for holding data. These registers are clocked by a signal (CLK), which controls when data is latched into the register.

The stored data is displayed using a set of 8 LEDs (Z0-Z7), where each LED corresponds to one bit of the output. This register is essential for storing intermediate or final results before passing them on to the seven segment display.

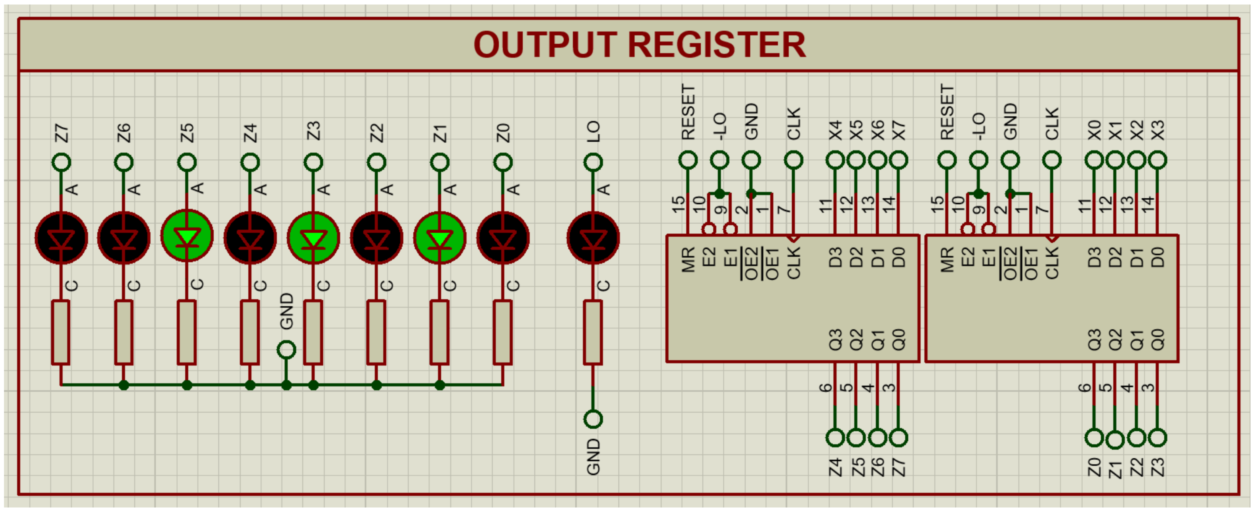


Figure 11: Output Register

Registers

The SAP-1 computer is designed with three similar 8-bit register modules: Register A, Register B, and the Instruction Register.

- **Register A** (Accumulator) and **Register B** are reserved for storing the two input numbers that will be sent to the Arithmetic and Logic Unit (ALU) for processing.
    - These registers hold the operands for arithmetic and logical operations, such as addition or subtraction, performed by the ALU.

- The **Instruction Register** holds the current instruction being executed. This register receives data from the RAM and stores it temporarily while the instruction is decoded and executed.

Each of these register modules is built using two 4-bit D-type registers and a Octal Bus Transceiver. D-type registers are synchronous flip-flops used to store binary data, clocked by the system clock. The output from each register can be connected or disconnected from the main data bus through an 8-bit Octal Bus Transceiver, which ensures smooth communication and prevents data conflicts on the bus.

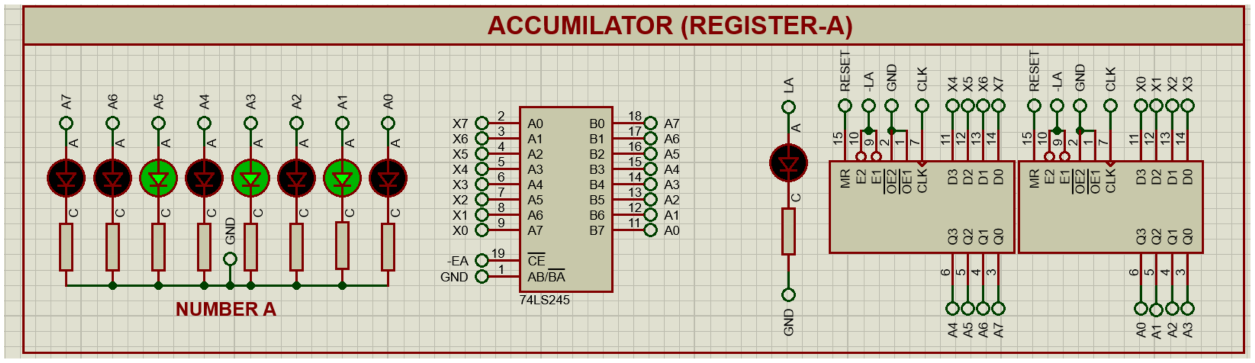


Figure 12: Register A

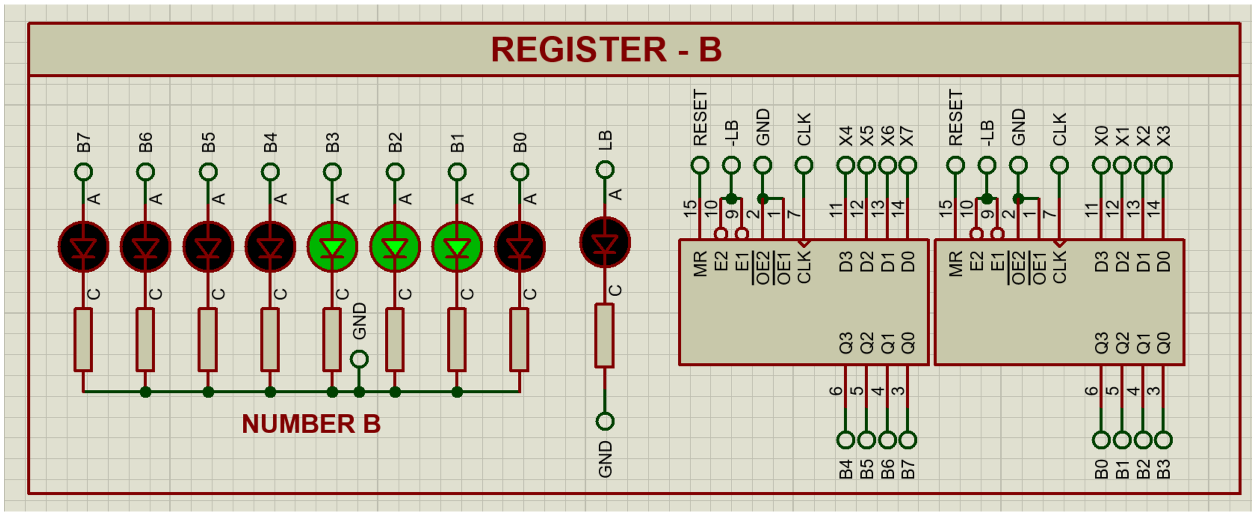


Figure 13: Register B

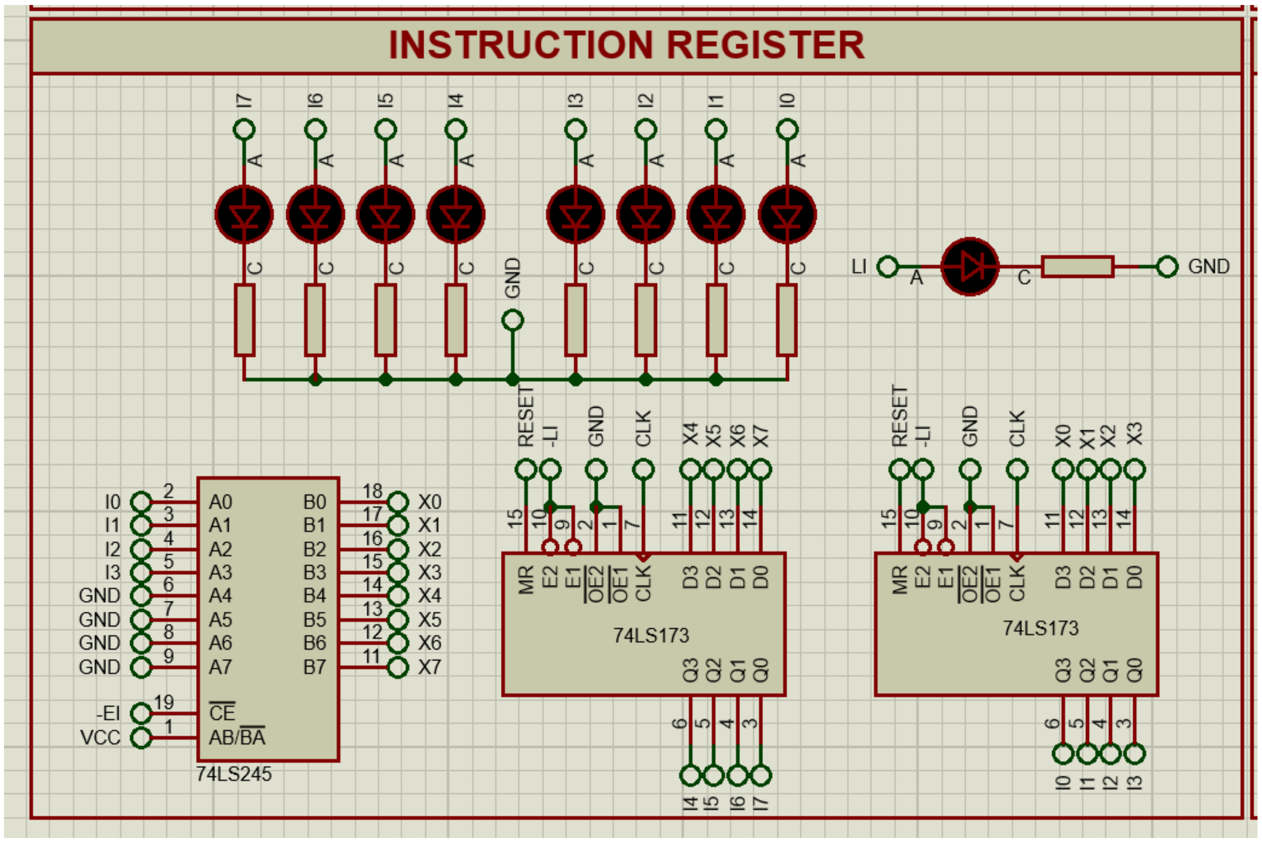


Figure 14: Instruction Register

Memory Address Register

The Memory Address Register of the SAP-1 computer, responsible for holding the memory addresses used to access data during instruction execution. The MAR consists of key components like multiplexers, flip-flops, and visual indicators (LEDs) that help manage and display the current address being accessed.

The **74LS157 multiplexer** is used to selects between different address inputs (AD0-AD3) based on the mode switch (MD). This allows the SAP-1 to switch between program control and manual address input. The selected address is passed to the **74LS173 register**, which stores and latches the address value. Clock signals (CLK) control when the address is updated, while output enables (OE1, OE2) manage when the address is passed to the data bus.

LED indicators show the current address values for each bit (A0-A3), providing a visual confirmation of the address being accessed. The address can also be manually set using the switch block labeled "ADDRESS." This setup ensures that the SAP-1 can effectively manage memory access during both program execution and manual operation modes.

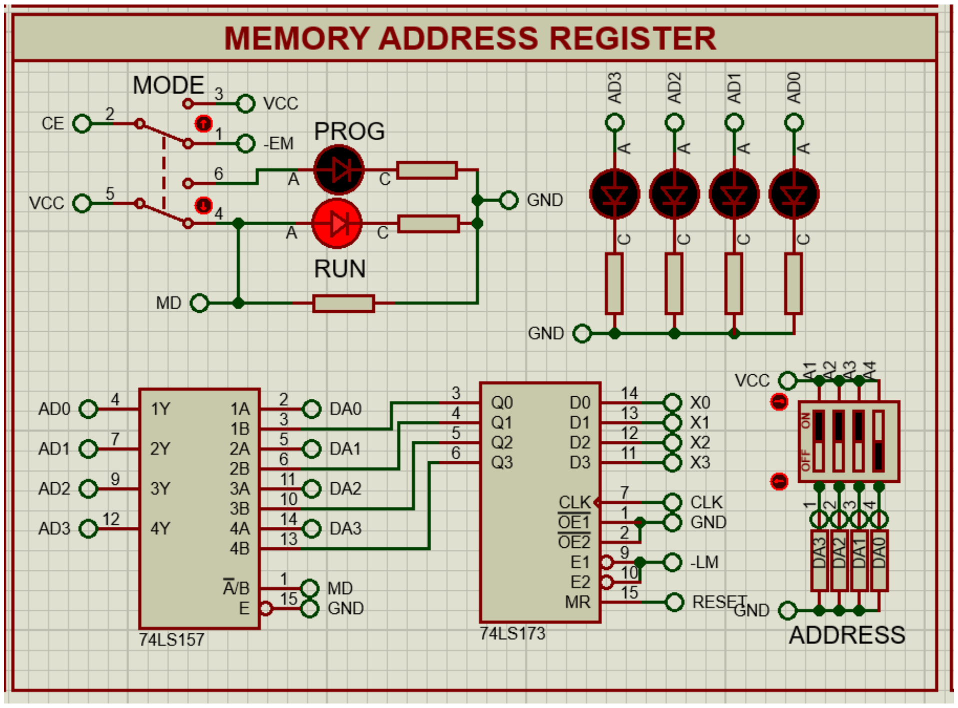


Figure 15: Memory address register

Seven Segment Display

The Seven Segment display is controlled by an EEPROM (CAT28C16), which stores the data to be shown, with the data lines from the memory driving the display segments. A 74HC238 decoder is used to select which digit of the display is active at any given time, allowing the SAP-1 to cycle through the four digits.

A 555 timer generates a clock signal, which is fed to two JK flip-flops (74LS76) that manage the multiplexing of the display digits. The flip-flops ensure that only one digit is active at a time, giving the impression of a stable output. The combination of these components allows the SAP-1 to output its results in a readable format on the seven-segment display.

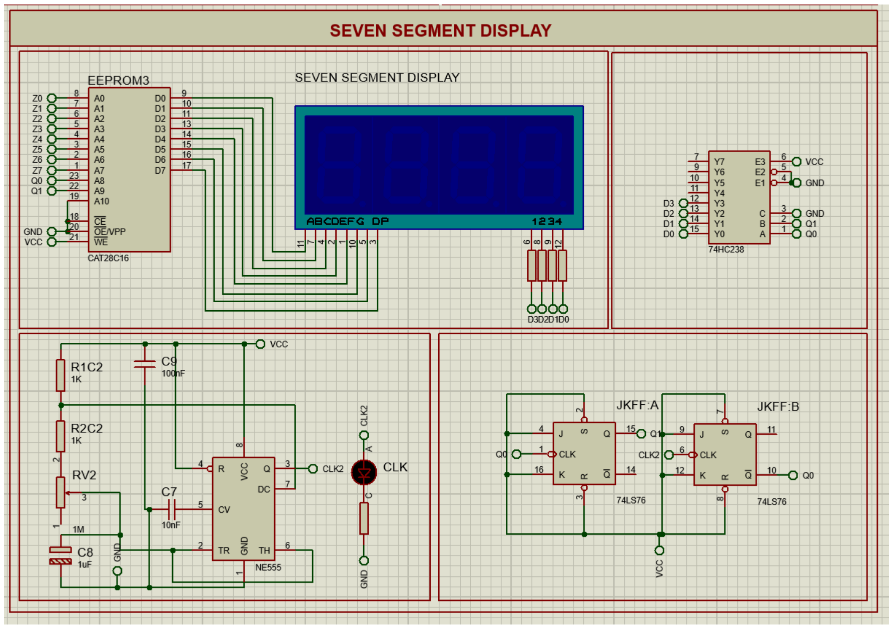


Figure 16: Seven Segment Display

RAM

The RAM (Random Access Memory) in the SAP-1 is a 16-byte memory unit used for storing both instructions and data. The memory address register (MAR) determines the specific memory location to be accessed, while the data is read from or written to the memory based on the control signals. The RAM is crucial for fetching instructions during program execution and temporarily holding data for processing by the ALU.

The system includes a set of LEDs (R0-R7) that visually indicate the current state of the data being stored or accessed from the RAM. A switch block (DATA) is used to input or modify data values. A "WRITE" control circuit ensures that the data inputted through the switches is written to the selected memory location when the write signal is enabled. This setup allows the SAP-1 to store and retrieve data during program execution.

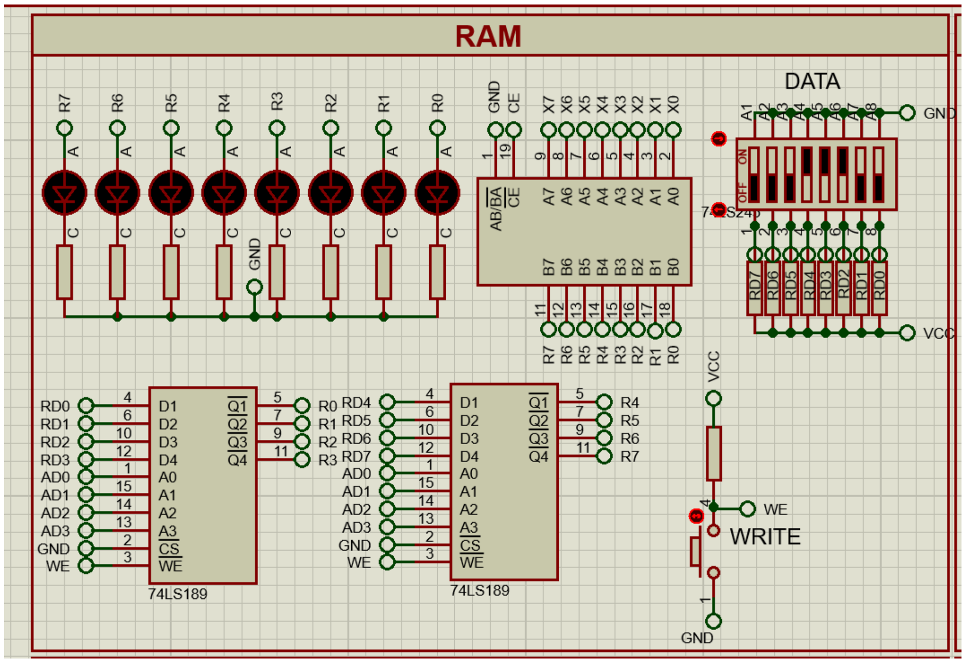


Figure 17: RAM

Control Sequencer

The **Control Matrix** (Control Sequencer) of SAP-1 computer, as shown in the diagram, is a critical component responsible for generating control signals that dictate the operation of the entire system.

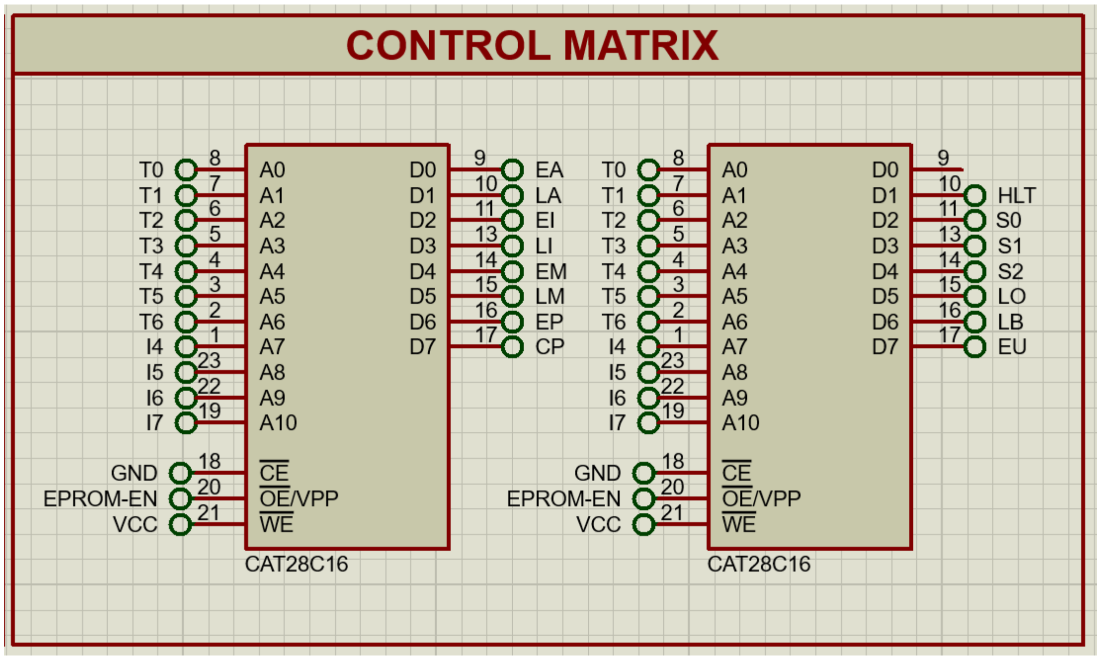


Figure 18: Control Matrix

1. **CAT28C16 EPROM Chips**: These chips store the control logic for each clock cycle and instruction. The **EPROM** serves as a programmable memory that contains pre-defined control signals for the processor.

    - The left chip's address lines (A0-A10) represent the time steps of the instruction cycle and the states (T0-T7) within each machine cycle.
    - The right chip also has address lines connected to specific functions such as halt (HLT), signal flags (S0-S2), load (LO), enable (EU), and more, which are critical for instruction sequencing.

2. **Address and Data Lines**:

    - Each EPROM has a set of **address lines (A0-A10)**, which correspond to the different time steps (T0-T7) and instruction inputs (I0-I7). These address lines select a specific word in the EPROM that defines the control signals to be generated at each time step.
    - The **data lines (D0-D7)** are the output of the EPROM, where each bit represents a specific control signal that will be activated or deactivated depending on the address.

3. **Control Signals**:

   - The **T-lines (T0-T7)** represent the different phases of an instruction cycle, which include fetching, decoding, and execution. These lines, in combination with the instruction inputs, select the correct control word from the EPROM.
   - The control signals such as **EA, LA, EI, LI, EM, LM, EP, CP, HLT, S0-S2, LO, LB, EU** are derived from the EPROM's output. Each signal enables or disables specific operations within the SAP-1, such as loading data into the accumulator, enabling the bus, incrementing the program counter, or halting the system.

4. **EPROM Control Pins**:

   - **CE (Chip Enable)** and **OE (Output Enable)** control when the EPROM is active and ready to output data.
   - **WE (Write Enable)** is typically used when programming the EPROM, which would not be necessary during normal operation of the SAP-1 since the EPROM contains pre-programmed control logic.

Control Signals

The **Control Input Module**, is responsible for managing the various control signals that influence the overall operation of the SAP-1 computer.

Key Components:

1. **Clock Section**:

   - **MCLK** and **CLK Mode**: This section manages the system clock, which is crucial for synchronizing the operations of the SAP-1. The **MCLK (Manual Clock)** allows the user to step through the clock cycles manually, while the **CLK Mode** switch toggles between **manual** and **automatic** modes. In **manual** mode, the user can control each clock pulse, and in **auto** mode, the clock runs continuously at a set frequency.
   - The **LED indicator** connected to the clock signal blinks, showing the clock's state (whether it's pulsing or halted).

2. **DIP Switches:**

   This section provides control over enabling and disabling the EPROM. The DIP switch is used to enable specific signals, including the EPROM itself (**EPROM-EN**), and other components. The DIP enable control ensures that only specific parts of the SAP-1 circuitry are active during a given phase, allowing you to fine-tune the operations.

3. **Control Signals**
   - **EA**: Enable Accumulator
   - **LA**: Load Accumulator
   - **EP**: Enable Program Counter
   - **EM**: Enable Memory
   - **LM**: Load Memory
   - **LI**: Load Instruction
   - **EI**: Enable Instruction
   - **HLT (Halt)**: This switch halts the clock, effectively stopping the processor.
4. **Logic Gates** :

   The **NOT gates** (inverters) are shown connected to some of the control lines, including **LA, EA, EU, LB, LO, EP, EM, LM, LI, EI**.

5. The **LED indicators** give a clear status of which control signals are currently active, providing feedback for debugging or educational purposes.

You can find the actual Control signals used in the each instructions which are programed in EPROM using the following link

https://github.com/Bimsara-Janakantha/SAP1_Computer_Design/tree/main/EEPROM%20program%20and%20OP Code

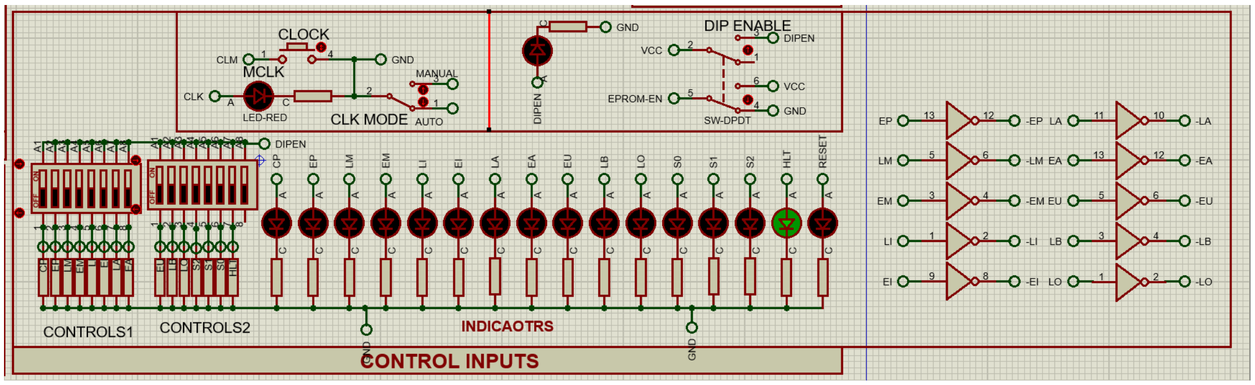


Figure 19: Control Inputs

Assembly Language Instructions:

The SAP-1 operates using a simple set of assembly language instructions. These instructions are written in binary code and stored in the RAM. Examples of instructions include:

- **LDA (Load Accumulator):** Loads a value from a specific memory address into the accumulator.
- **ADD:** Add the value in the B register to the value in the accumulator.
- **SUB:** Subtracts the value in the B register from the value in the accumulator.
- **OUT:** Transfers the value from the accumulator to the output register.
- **HLT (Halt):** Stops the execution of the program.

OPCODES FOR OMEGA SAP-1 COMPUTER

INSTRUCTION	OPCODE
LDA	0000
ADD	0001
SUB	0010
XOR	0011
AND	0100
OR	0101
NAND	0110
NOR	0111
OUT	1110
HLT	1111

Printed Circuit Board

The Printed Circuit Board (PCB) is the platform on which all the modules of the SAP-1 computer are mounted. It is designed to integrate the clock module, program counter, registers, ALU, RAM, control sequencer, and output register into a cohesive unit of size 210mm * 180mm. The PCB layout was carefully planned to minimize signal interference and ensure proper connections between components. After designing the schematic, the PCB was fabricated and assembled, with testing conducted to verify the functionality of the entire system.

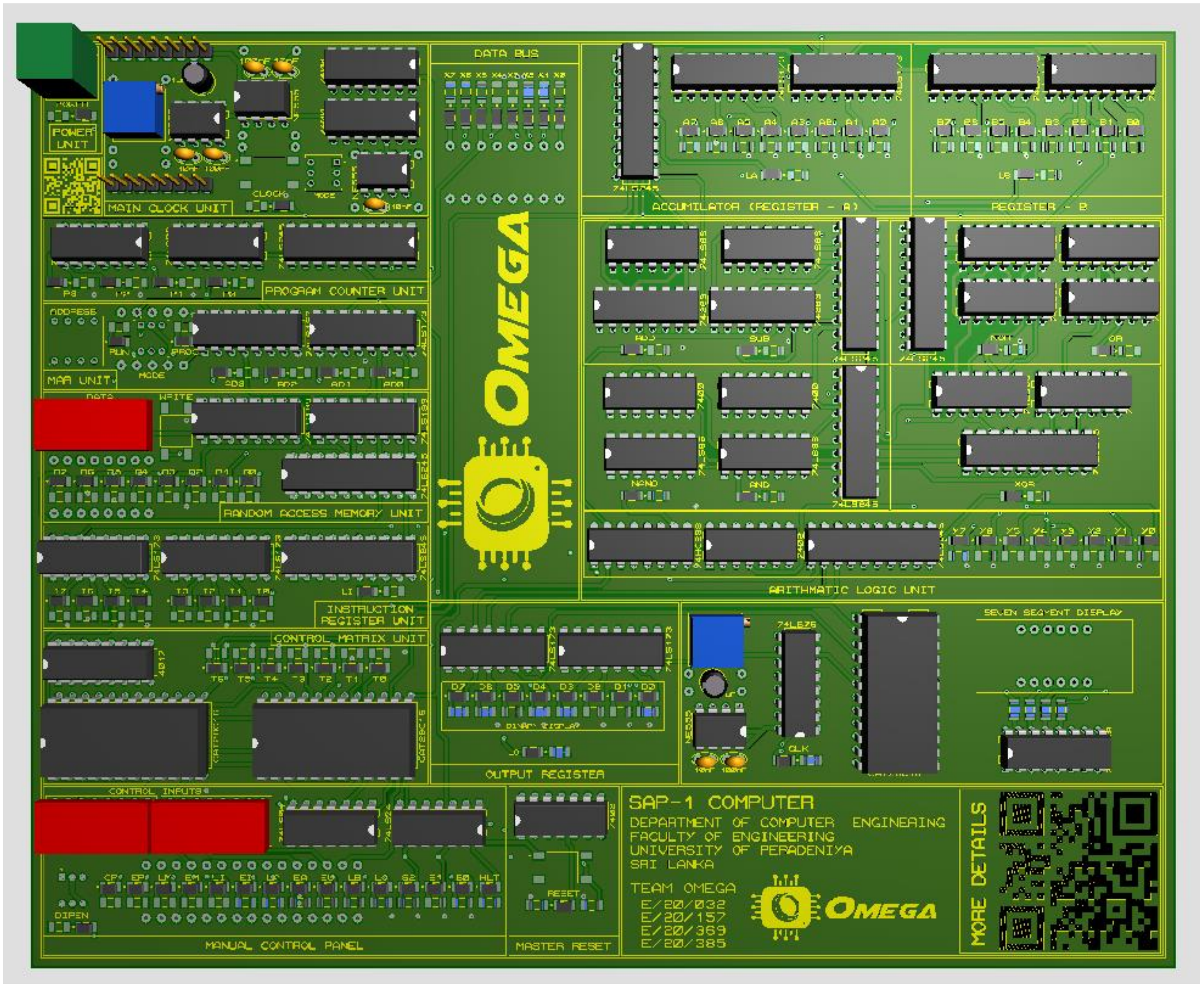


Figure 20: Printed Circuit Board

Budget

Item No.	Bill No.	Component	Unit Price	Quantity	Total
01	1	PCB Fabrication			20,169.65
02	2	Tax (Shipping /VAT)			7,464.00
03	3	Courier charge(Bill 03)			480.00
04	8	74LS00	30.00	5	150.00
05	8	74LS02	30.00	4	120.00
06	6	74LS04	50.00	5	250.00
07	4	74HC08	95.00	1	95.00
08	3	74LS76	120.00	2	240.00
09	3	74LS83	120.00	4	480.00
10	3	74LS86	50.00	10	500.00
11	4	74LS93	60.00	1	60.00
12	4	74LS157	60.00	1	60.00
13	7	74LS173	94.25	10	942.50
14	7	74LS189	226.90	5	1,134.50
15	8	74HC238	57.50	2	115.00
16	4	74LS245	26.00	10	260.00
17	9	74LS283	150.00	4	600.00
18	7	CAT28C16	512.40	10	5,124.00
19	8	CD4017	30.00	1	30.00
20	5	NE555	25.00	4	100.00
21	5	7 Segment Display	56.00	2	112.00
22	4	LED GREEN (SMD)	2.25	34	76.50
23	5	LED-RED (SMD)	2.50	40	100.00
24	5	LED-YELLOW (SMD)	2.25	40	90.00
25	8	Switch-DPDT	80.00	3	240.00
26	8	Self-locking Push Button	10.00	1	10.00
27	3	Dip Switch_4	30.00	2	60.00
28	5	Dip Switch_8	34.00	5	170.00
29	8	Resistor 1k	1.00	45	45.00
30	5	Resistor 1.2k (SMD)	1.50	120	180.00
31	3	1M Variable Resistor	8.00	4	32.00
32	4	Terminal Block I2	22.00	1	22.00
33	3	Capacitor 10nF	2.00	5	10.00
34	3	Capacitor 100nF	2.00	5	10.00
35	3	Capacitor 1µF	2.00	2	4.00
36	8	IC Bases (8 pin)	5.00	4	20.00
37	8	IC Bases (14 pin)	10.00	20	200.00
38	8	IC Bases (16 pin)	10.00	10	100.00
39	5	IC Bases (20 pin)	7.00	10	70.00
40	5	IC Bases (24 pin)	17.00	4	68.00
41	5	Female Headers	35.00	1	35.00
		TOTAL			40,029.15

Results

The SAP-1 project was successfully completed with all modules functioning as intended. The integration of the modules onto a single PCB resulted in a compact and operational SAP-1 computer. Testing confirmed that the clock module synchronized operations correctly, the program counter accurately tracked instruction execution, and the ALU performed arithmetic and logical operations as expected. The assembly language instructions were executed smoothly, with the output correctly displayed on the output register.

Conclusions and Future Works

The SAP-1 project successfully demonstrated the principles of computer architecture and the step-by-step process involved in building a simple computer. Future improvements could include expanding the instruction set, increasing the memory capacity, and enhancing the ALU to perform more complex operations. Additionally, the development of a more sophisticated control sequencer could allow for more intricate program execution, paving the way for further exploration into computer design.

References

1. Malvino, A. P. (1990). Digital Computer Electronics: An Introduction to Microcomputers.

2. Building an 8-bit breadboard computer!, Ben Eater Available at, https://www.youtube.com/playlist?list=PLowKtXNTBypGqImE405J2565dvjafglHU

3. NuwanJ. (n.d.). GitHub - NuwanJ/peraSAP-I: Pera SAP-I GitHub. https://github.com/NuwanJ/peraSAP-I

Appendix

Schematics Diagrams:

Schematic diagrams of additional modules design in our SAP 1 computer are listed below

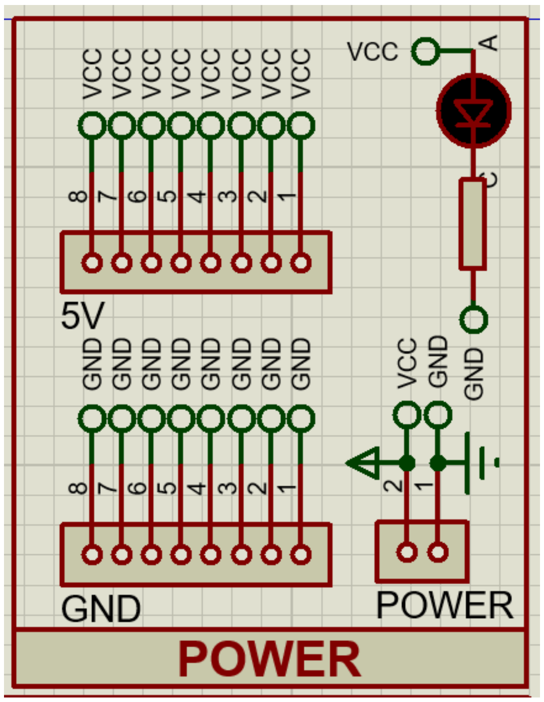


Figure 20: Power Unit

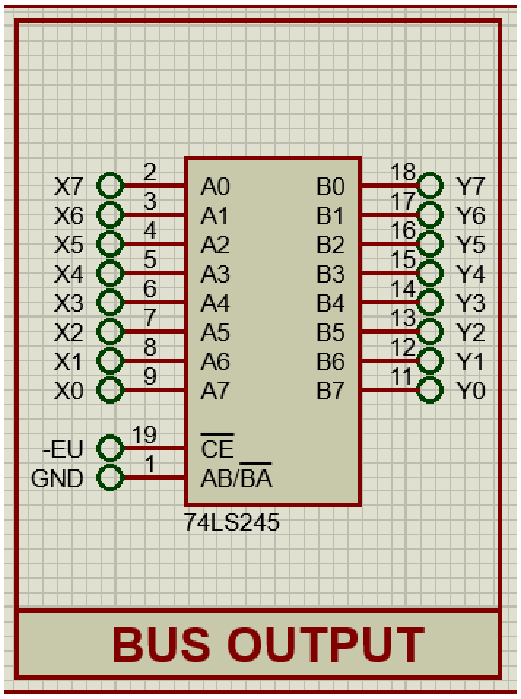


Figure 21: Bus output

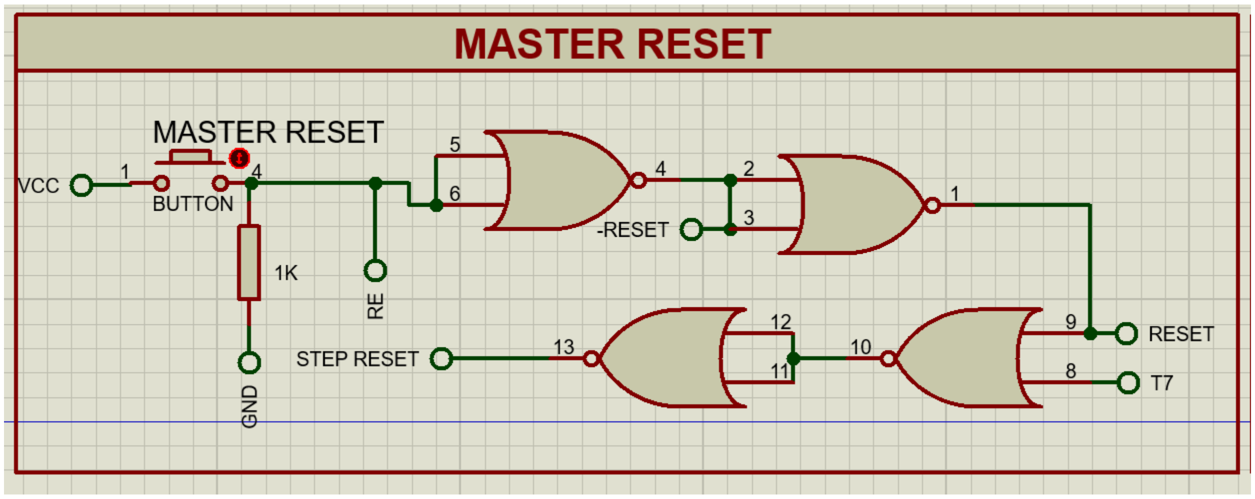


Figure 21: Master Reset

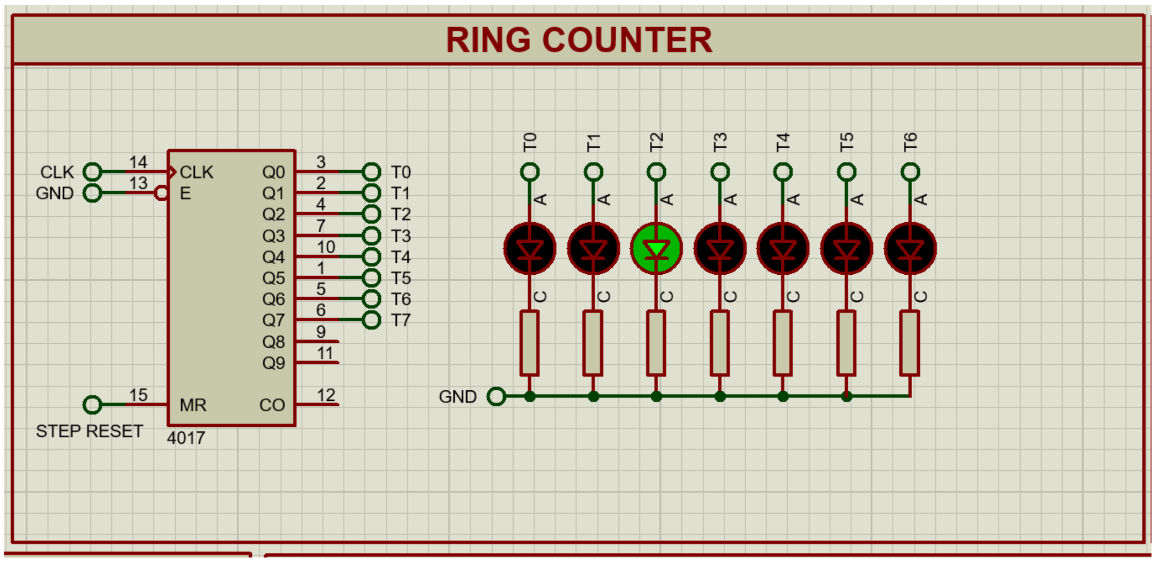


Figure 23: Ring Counter

Project Repository:

https://github.com/Bimsara-Janakantha/SAP1_Computer_Design/tree/main