DESIGNING 4 BIT MICROPROCESSOR

1. OBJECTIVE:

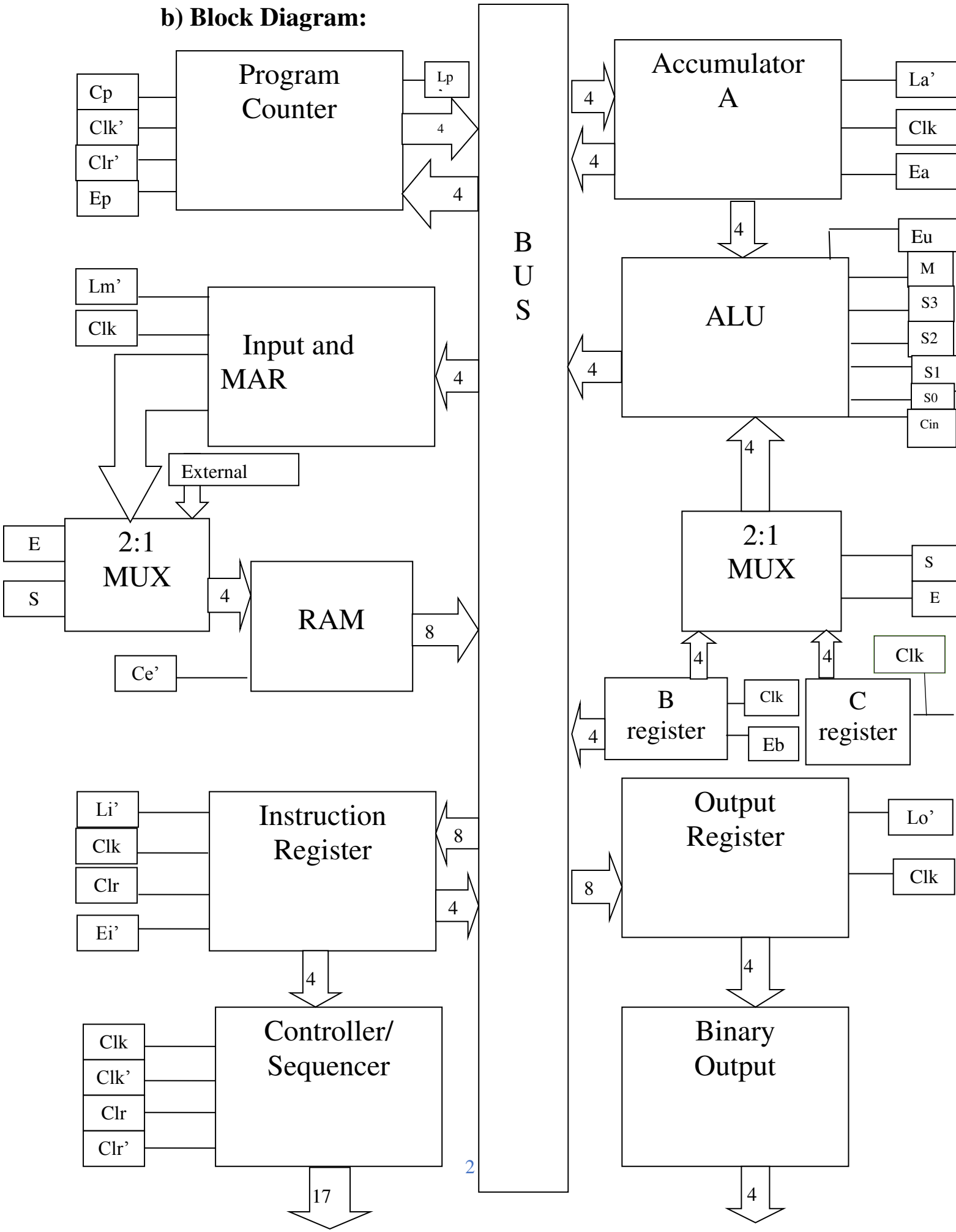
The main purpose of SAP (Simple As Possible) design is to introduce all the crucial ideas behind computer operation without burying in unnecessary detail. But even being a simple microprocessor, SAP covers many advanced concepts. The SAP we have designed is 4-bit bus organized computer where all registers are connected to a bus with the help of tri-state buffers which can perform nine instructions including arithmetic and logic operations. The address bus of our SAP is 4-bit and data bus is 8-bit. The primary purpose of the design is to develop a basic understanding of how a microprocessor works, interacts with memory and other parts of the system like input and output and to learn microprocessor design.

2. METHODOLOGY:

a) Design Step:

- Understanding and sketching out the architectural structure.
- Analysing the instructions and finding out the T states and nonetheless minimizing the number of states.
- Designing the control matrix according to the T states.
- Simulating the whole system in “Circuit Maker” software.
- Implementing the hardware of control matrix and other parts like- Program Counter(PC), Memory Address Register(MAR), Instruction Register(IR), Random Access Memory(RAM), Arithmetic Logic Unit(ALU), Registers etc. and finally merging them into one system.

b) Block Diagram:



c) T State:

Fetch Cycle:

T_1: MAR ← PC; E_P, $\overline{Lm}$

T_2: PC ← PC+1; IR ← M[MAR];C_P, $\overline{Ce}$, $\overline{L\imath}$

Execution Cycle:

1.LDA Routine:

T_3: MAR ← IR; $\overline{E\imath}$, $\overline{Lm}$

T4: A ← M[MAR]; $\overline{Ce}$, $\overline{La}$

2.ADD Routine:

T_3: A ← A+B; C_{in}, S_1, $\overline{La}$, E_u

T4: NOP

3.SUB Routine:

T_3: A ← A+C'+1; S_3, S_2, S_0, $\overline{La}$

T4: NOP

4.AND Routine:

T_3: A ← A^B; $\overline{La}$, E_U

T4: NOP

5.MOV Routine:

T_3: A ← B; $\overline{La}$, E_B

T4: NOP

6.JMP[M] Routine:

T_3: PC ←M[IR]; $\overline{Lp}$, $\overline{E\imath}$

T4: MAR ←PC

7.JZ[M] Routine:

T_3: PC ←M[IR]; $\overline{Lp}$, $\overline{E\imath}$

T4: MAR ←PC

8.OUT Routine:

T_3:OUT ← A; $\overline{Lo}$, E_A

T4: NOP

9.HLT Routine:

T_3: NOP

T4: NOP

d) Control Matrix:

Equations:

$E_p=T_1$
$C_p = T_2$

$\overline{Lp}=\overline{T3(JMP + ZJZ)}$
$\overline{Lm}=\overline{T1 + T3LDA + T4(JMP + ZJZ)}$

$L_I=T_2$
$E_I=T_3$ (JMP+ZJZ+LDA)

$L_A = T_4$ (LDA)+T_3(ADD+SUB+AND+MOV)
$E_A=T_3$out

$E_p = T_3$ (ADD+SUB+AND)
$S_3=\overline{SUB}$

$S_2=SUB$
$S_1=\overline{ADD}$

$S_0=\overline{SUB}$
$C_{in}=ADD$
$L_0=T3OUT$

Timing routines:

Fetch Cycle: This cycle is same for all the instructions.

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_1	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0
T_2	0	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0

Execution Cycle: This cycle is most of the time different for different instructions.

1. **LDA:-**

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	1	0
T_4	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0

2. **ADD:-**

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	1	1	0
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

3. **SUB:-**

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	0	0	1	0	0	1	0	0	0	1	0
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

4. AND:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	0	0	1	1	0	0	0	0	0	1	0
T_4	0	0	1	1	1	0	0	1	0	0	1	0	0	0	0	0	1	0

5. MOV:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

6. OUT:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

7. JMP:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

8. JZ:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0
T_4	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0

9. HLT:-

T	E_P	C_P	L_P	L_M	C_E	L_I	E_I	L_A	E_A	E_U	M	S_3	S_2	S_1	S_0	C_{in}	L_0	E_B
T_3	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0
T_4	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0

3. RECQUIRED IC:

a) Total IC:

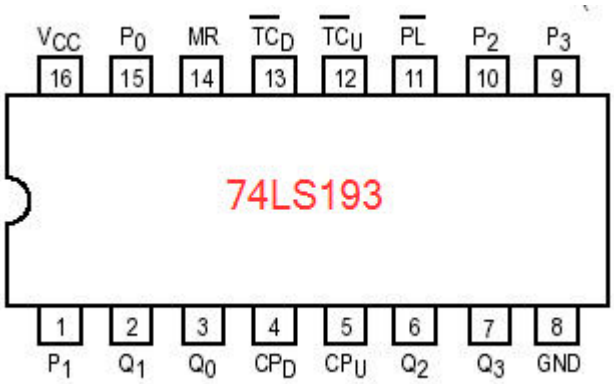
IC Number	IC Name	Amount
1. 74193	SYNCHRONOUS 4-BIT UP/DOWN COUNTERS	1
2. 74125	QUADRUPLE BUS BUFFERS (WITH 3-STATE OUTPUTS)	5
3. 74173	4-BIT D-TYPE REGISTER (WITH 3-STATE OUTPUTS)	7
4. 74157	QUAD 2-LINE to 1-LINE DATA SELECTORS/MULTIPLEXERS	2
5. 74181	4-BIT ARITHMATIC LOGIC UNIT	1
6. 6116	CMOS STATIC RAM 16K (2K x 8-BIT)	1
7. 74154	4-LINE to 16-LINE DECODERS	1
8. 74164	8-BIT SERIAL-IN, PARALLEL-OUT SHIFT REGISTER	1
9. 7404	HEX INVERTES	4
10. 4002	4-IN NOR	1
11. 7486	2-IN XOR	1
12. 7408	2-IN AND	3
13. 7432	2-IN OR	2
TOTAL		**30**

N.B.- We have minimized

1. 1 NAND gate by 1 AND and 1 NOT GATE
2. 3 NOR gate by 3 OR and 3 NOT GATE
3. 1 3-IN OR gate by 2 OR GATE

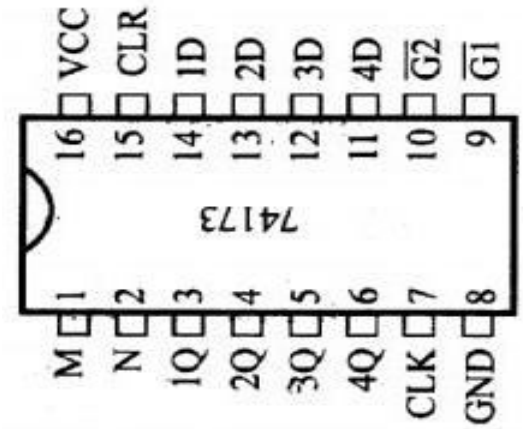
b) IC CONFIGURATION:

- **74193:**



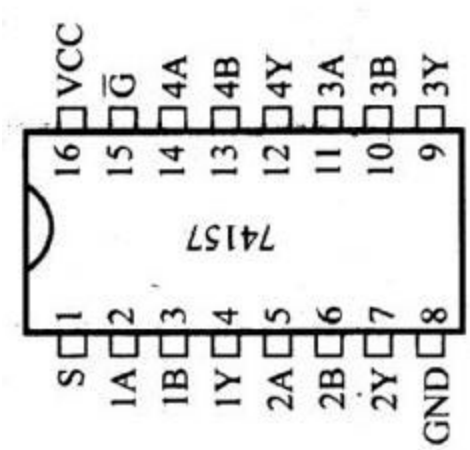
1	P_1	data input
2	Q_1	counter output
3	Q_0	counter output
4	CPD	count down clock input (low-to-high, edge-triggered)
5	CPU	count up clock input (low-to-high, edge-triggered)
6	Q_2	counter output
7	Q_3	counter output
8	GND	ground
9	P_3	data input
10	P_2	data input
11	PL	parallel load input (active low)
12	TCU	terminal count up (carry) output (active low)
13	TCD	terminal count down (borrow) output (active low)
14	MR	asynchronous master reset (active high)
15	P_0	data input
16	Vcc	supply voltage

- **74173:**



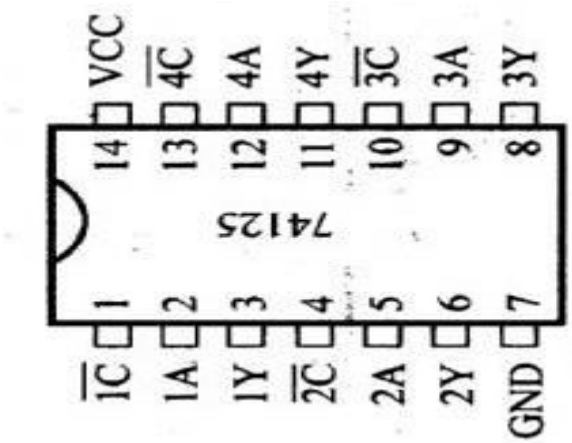
Pin	Symbol	Description
1	M	output enable input (active low)
2	N	output enable input (active low)
3	1Q	3-state flip-flop output
4	2Q	3-state flip-flop output
5	3Q	3-state flip-flop output
6	4Q	3-state flip-flop output
7	CLK	clock input (low-to-high, edge-triggered)
8	GND	ground
9	G_1	data enable input (active low)
10	G_2	data enable input (active low)
11	4D	data input
12	3D	data input
13	2D	data input
14	1D	data input
15	CLR	asynchronous master reset (active high)
16	Vcc	supply voltage

- **74157:**



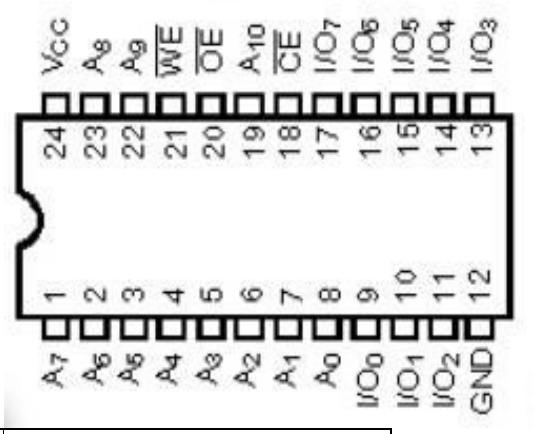
Pin	Symbol	Description
1	S	common data select input
2	1A	data input from source 0
3	1B	data input from source 1
4	1Y	multiplexer output
5	2A	data input from source 0
6	2B	data input from source 1
7	2Y	multiplexer output
8	GND	ground
9	3Y	multiplexer output
10	3B	data input from source 1
11	3A	data input from source 0
12	4Y	multiplexer output
13	4B	data input from source 1
14	4A	data input from source 0
15	G	enable input (active low)
16	Vcc	supply voltage

- **74125:**



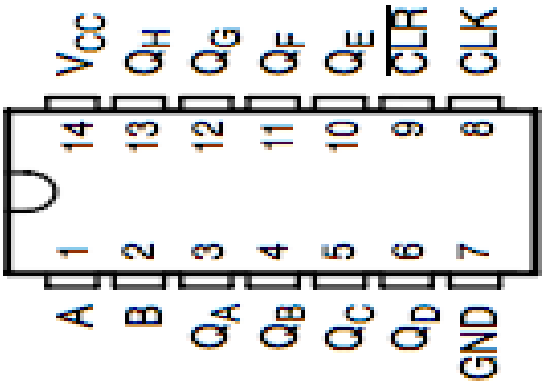
Pin	Symbol	Description
1	1C	output enable (active low)
2	1A	data input
3	1Y	data output
4	2C	output enable (active low)
5	2A	data input
6	2Y	data output
7	GND	ground
8	3Y	data output
9	3A	data input
10	3C	output enable (active low)
11	4Y	data output
12	4A	data input
13	4C	output enable (active low)
14	Vcc	supply voltage

- **6116 -2k RAM:**



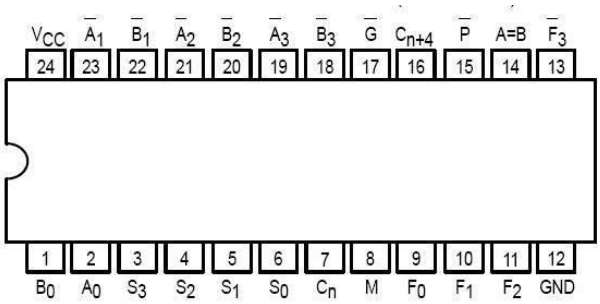
Pin	Symbol	Description
1	A7	Address inputs
2	A6	Address inputs
3	A5	Address inputs
4	A4	Address inputs
5	A3	Address inputs
6	A2	Address inputs
7	A1	Address inputs
8	A0	Address inputs
9	I/O0	Data Input/output
10	I/O1	Data Input/output
11	I/O2	Data Input/output
12	GND	Ground
13	I/O3	Data Input/output
14	I/O4	Data Input/output
15	I/O5	Data Input/output
16	I/O6	Data Input/output
17	I/O7	Data Input/output
18	CE	Chip Enable (active low)
19	A10	Address inputs
20	OE	Output Enable (active low)
21	WE	Write Enable (active low)
22	A9	Address inputs
23	A8	Address inputs
24	Vcc	Supply voltage

- **74164:**



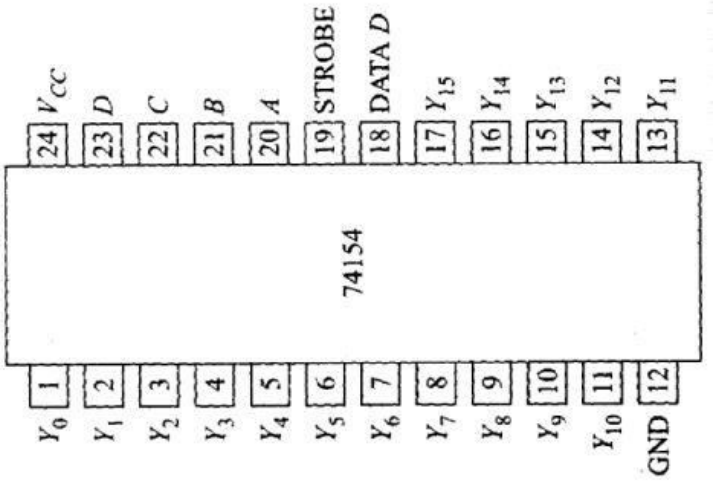
Pin	Symbol	Description
1	A	data input
2	B	data input
3	QA	output
4	QB	output
5	QC	output
6	QD	output
7	GND	ground
8	CP	clock input (low-to-high, edge-triggered)
9	CLR	master reset input (active low)
10	QE	output
11	QF	output
12	QG	output
13	QH	output
14	Vcc	supply voltage

- **74181:**



Pin	Symbol	Description
1	B0	Operand (Active LOW) Inputs
2	A0	Operand (Active LOW) Inputs
3	S3	Function — Select Inputs
4	S2	Function — Select Inputs
5	S1	Function — Select Inputs
6	S0	Function — Select Inputs
7	Cn	Carry Input
8	M	Mode Control Input
9	F0	Function (Active LOW) Outputs
10	F1	Function (Active LOW) Outputs
11	F2	Function (Active LOW) Outputs
12	GND	Ground
13	F3	Function (Active LOW) Outputs
14	A=B	Comparator Output
15	P	Carry Propagate (Active LOW) Output
16	Cn+4	Carry Output
17	G	Carry Generator (Active LOW) Output
18	B3	Operand (Active LOW) Inputs
19	A3	Operand (Active LOW) Inputs
20	B2	Operand (Active LOW) Inputs
21	A2	Operand (Active LOW) Inputs
22	B1	Operand (Active LOW) Inputs
23	A1	Operand (Active LOW) Inputs
24	Vcc	Supply voltage

- **74154:**



Pin	Symbol	Description
1	Y0	data output (active low)
2	Y1	data output (active low)
3	Y2	data output (active low)
4	Y3	data output (active low)
5	Y4	data output (active low)
6	Y5	data output (active low)
7	Y6	data output (active low)
8	Y7	data output (active low)
9	Y8	data output (active low)
10	Y9	data output (active low)
11	Y10	data output (active low)
12	GND	ground
13	Y11	data output (active low)
14	Y12	data output (active low)
15	Y13	data output (active low)
16	Y14	data output (active low)
17	Y15	data output (active low)
18	DATA D	enable input (active low)
19	STROBE	enable input (active low)
20	A	address input
21	B	address input
22	C	address input
23	D	address input
24	Vcc	supply voltage

5. DISCUSSION:

The discussion is based on the problems we critically faced with some major recommendations.

- We tried to minimize the T states to save clock periods so that operations run faster. The lesser the T states, the faster the microprocessor.
- We also minimized some ICs.
- It is wise to use ROM rather than RAM because the RAM is volatile and we have to write instructions every time after we switch off the power. It takes a lot of time.
- It was hard to find out a mistake after implementing the whole big circuitry and we had to rebuild the whole circuitry for several times. It should be done that the instructions are implemented one by one.
- The circuit must not be powered on for much time and kept in rest after testing for some times to avoid damage of ICs.
- Jumper wires are preferable as there is less voltage drop in jumpers.
- If LEDs are used it is wise to connect it via resistor or capacitor.
- Any part of the circuit may get shorted. So the contents of BUS should be checked for a regular interval.
- All the components of the trainer board are to be checked before starting the hardware implementation.
- We tried to make control matrix simpler.

SAP-1 is a 4-bit microprocessor which is the first stage in the evolution towards modern computers. The instruction sets of SAP-1 are very limited and simple. To make a 8-bit microprocessor, further extension of this project is needed to make. For example, there would have more instruction sets and the circuitry would be more complex than our project. That means it will take more time and energy to design and implement the hardware.