Microprocessor Lab Project

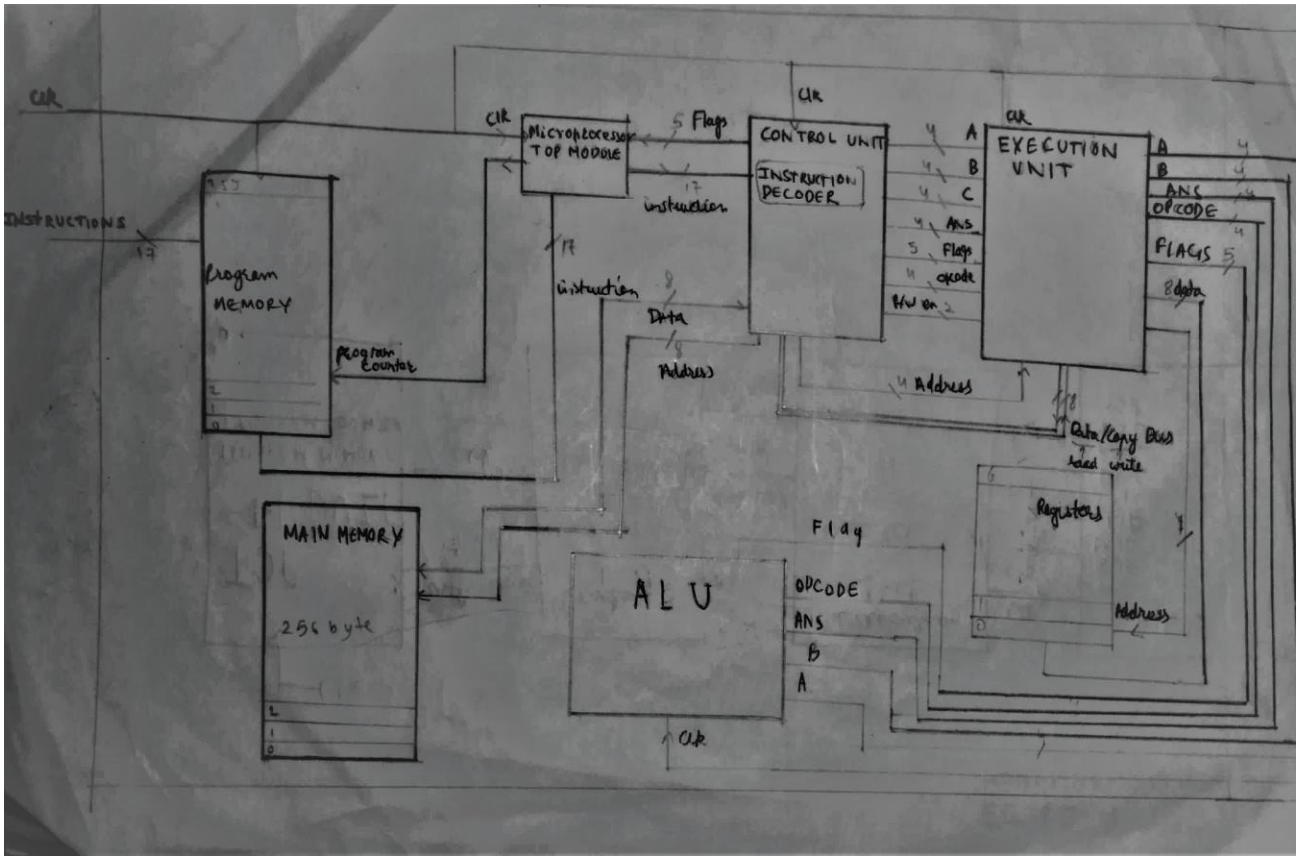
EE16B117 MOHAMMED KHANDWAWALA

EE16B115 M V SAINATH

Instructions to use-

- Open program.txt in the project folder to write the code.
- Run program.py (python assembling script) which writes the program in instruction code in def.v
- Now project is good to simulate the program.

Microprocessor Block Diagram-



Module Description-

- Input to the microprocessor is the clock and 17-bit instruction set.
- This is then processed in the top module microprocessor. It has Program Memory where the instructions are stored. Program Counter points at the instruction to be called in the Program Memory and gets incremented after each instruction (Unless Jump Instruction). For jump it checks relevant flag and jumps to the instruction assigned.
- Control Unit: The instruction from the top module is received here and decoded. Control Unit also has access to main memory. Based on the decoding and the type of operation read enable and write enable is set. Address and Data Bus is also set.

- Execution Unit: This receives in input opcode, operands for ALU, read enable, write enable, address line, data line. It also has register memory. It can perform read write operation in the registers. And passes the Alu operations to ALU.
- ALU receives operand and opcode, performs operations and sets the flag.

Specifications-

OPCODE	**OPERATION/SYNTAX*(format to write in program.txt)**	
00001	ADD	ADD R1,R2,R3;
00010	SUBTRACT	SUB R1,R2,R3;
00011	MULTIPLY	MUL R1,R2,R3;
00100	COMPARE	CMP R1,R2;
00101	INCREMENT	INC R1;
00110	DECREMENT	DEC R1;
00111	LEFT SHIFT	LSL R1;
01000	RIGHT SHIFT	RSL R1;
01001	MODULO	MOD R1;
01010	AND	AND R1,R2,R3;
01011	OR	OR R1,R2,R3;
01100	XOR	XOR R1,R2,R3;
01101	LOAD	LD R1,#address;
01110	LOAD IMIDIATE	LDI R1,#value;
01111	STORE	STR R1,#address;
10000	MOV	MOV R1,R2;
10001	LOAD INDIRECT of second argument)	LDIN R1,R2;(store in address of value

FOR JUMP USE 11111 as opcode followed by flag no. to check and program counter value to be assigned after the Jump.

SPECIFICATIONS

- Instruction 17-bit (5-bit opcode followed opcode specific bits)
- Program Memory 32 17-bits registers
- Main Memory 256 Byte (256 8-bit registers)

- 5-bit Flags
- (FLAG [0] = Carry/Borrow/overflow; FLAG [1] Shift out; FLAG [2] Zero Output; FLAG [3] Zero Output; FLAG [4] A<B ; FLAG [5] A==B)
- Opcode 5-bits
- Register File 6 registers 8-bit
- Address Line 8-bit
- Data Line 8-bit

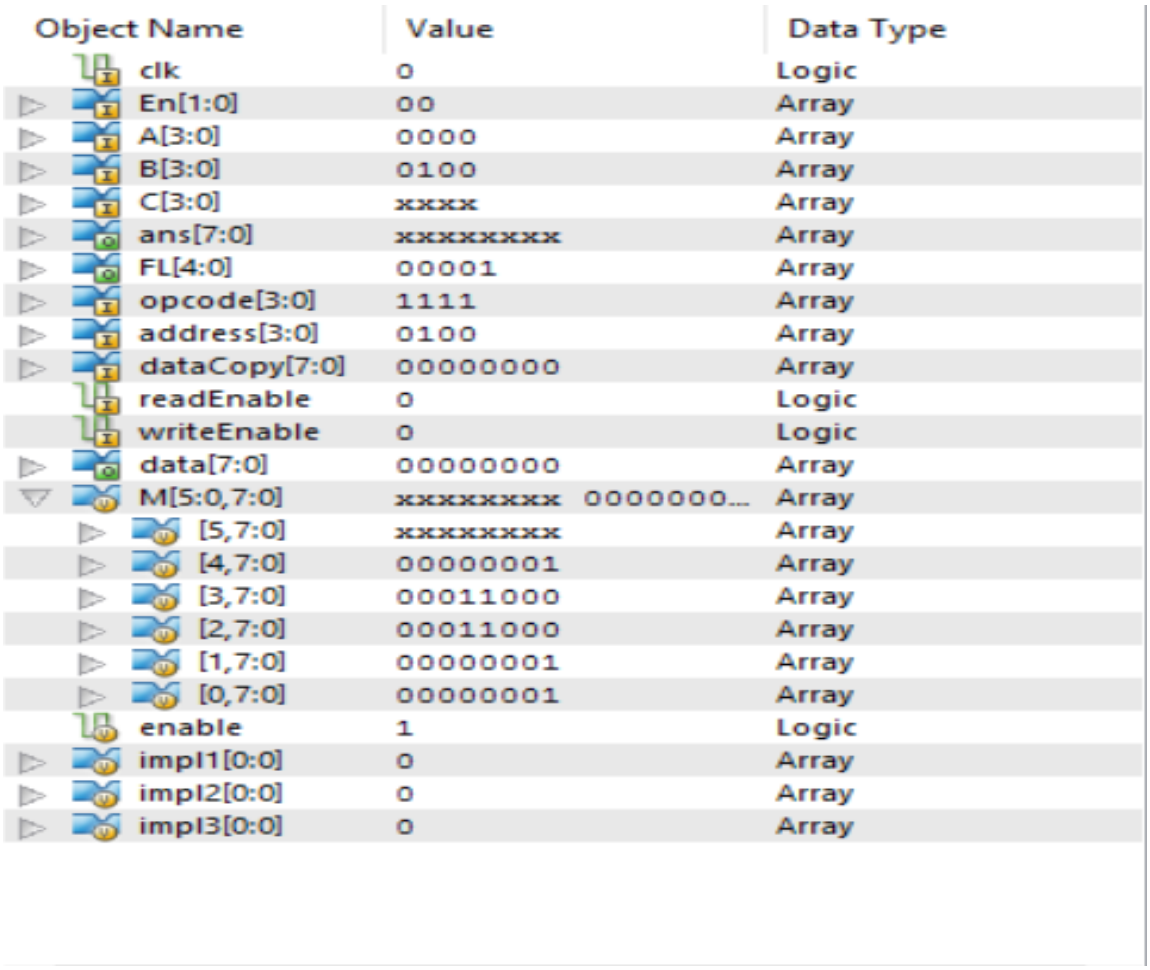
Screenshot For FACTORIAL 4 PROGRAM

```
LDI R1,#4;
LDI R2,#1;
LDI R4,#1;
label:
MUL R1,R2,R3;
MOV R3,R2;
SUB R1,R4,R0;
MOV R0,R1;
CMP R1,R4;
BEQ label;
```

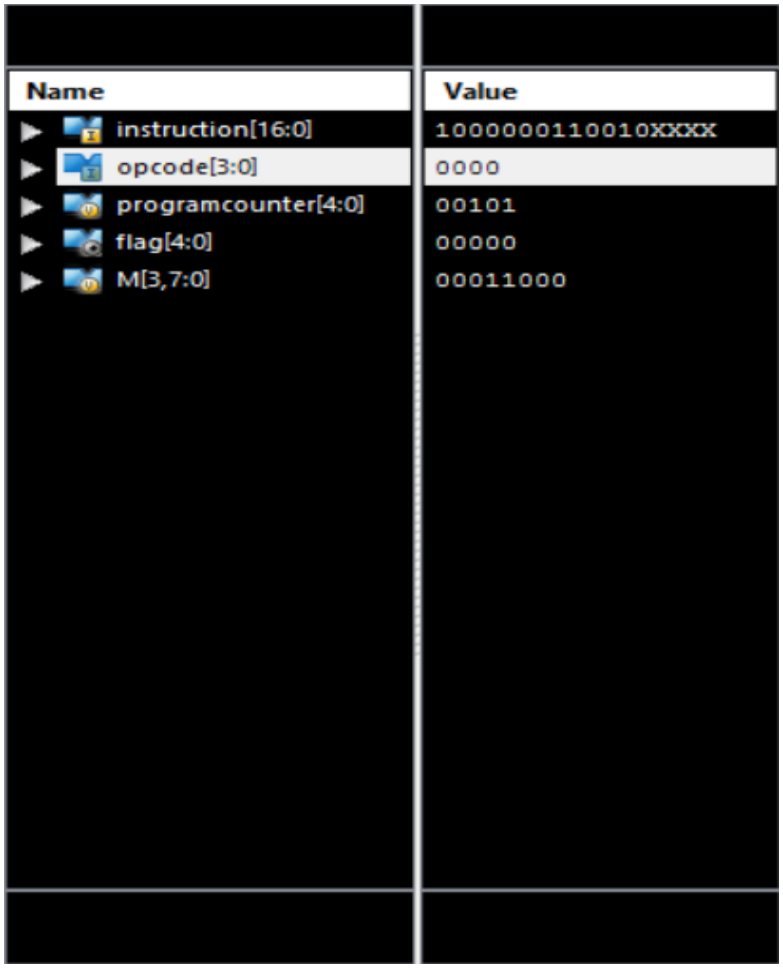
IN 17-bit Instruction Code

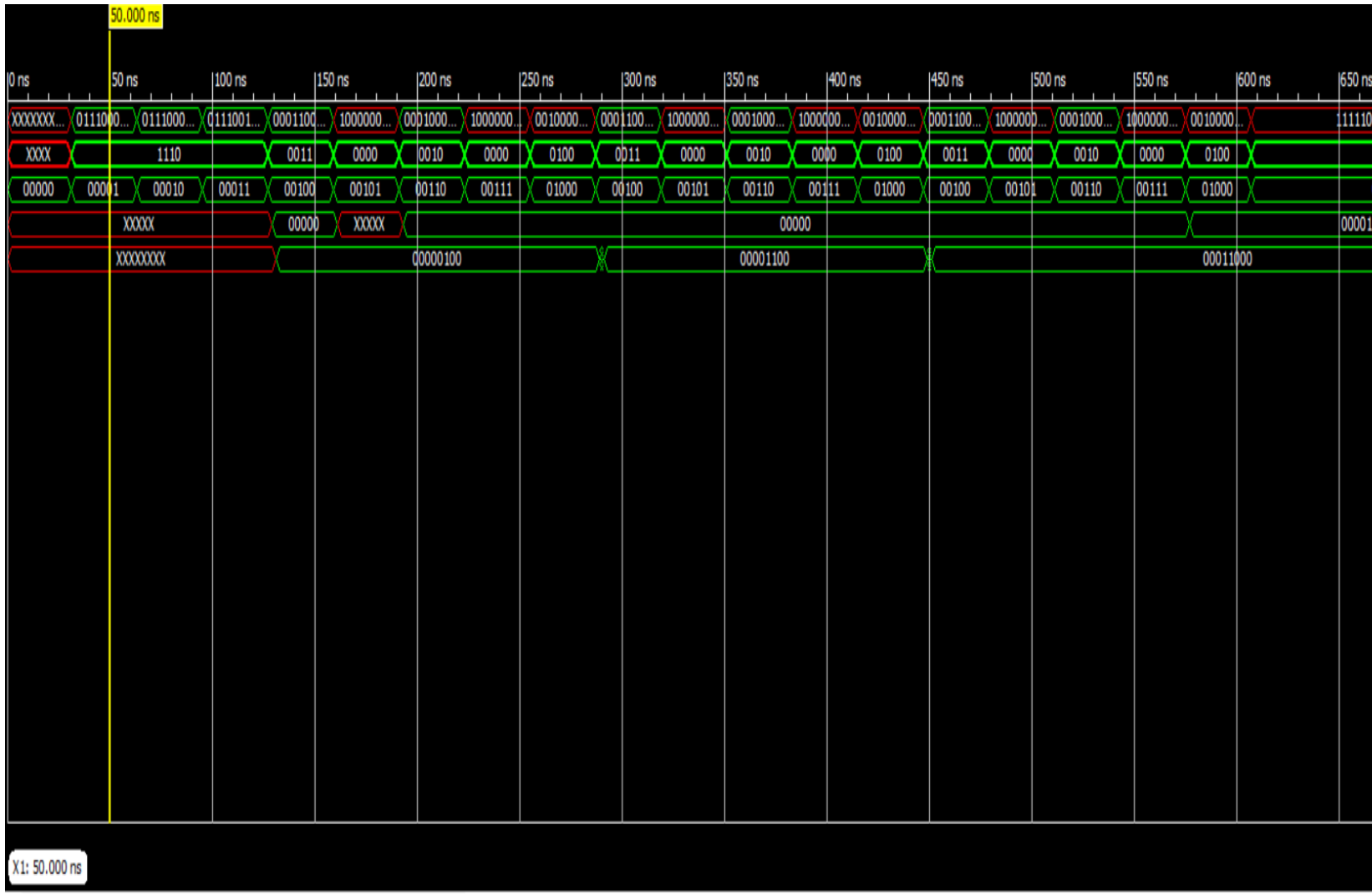
```
17'b01110000100000100
17'b01110001000000001
17'b01110010000000001
17'b00011000100100011
17'b1000000110010xxxx
17'b00010000101000000
17'b1000000000001xxxx
17'b0010000010100xxxx
17'b1111100000100xxxx
```

4! Value 24(00011000) stored in R3.



Object Name	Value	Data Type
clk	0	Logic
En[1:0]	00	Array
A[3:0]	0000	Array
B[3:0]	0100	Array
C[3:0]	xxxx	Array
ans[7:0]	xxxxxxxx	Array
FL[4:0]	00001	Array
opcode[3:0]	1111	Array
address[3:0]	0100	Array
dataCopy[7:0]	00000000	Array
readEnable	0	Logic
writeEnable	0	Logic
data[7:0]	00000000	Array
M[5:0,7:0]	xxxxxxxx 0000000...	Array
[5,7:0]	xxxxxxxx	Array
[4,7:0]	00000001	Array
[3,7:0]	00011000	Array
[2,7:0]	00011000	Array
[1,7:0]	00000001	Array
[0,7:0]	00000001	Array
enable	1	Logic
impl1[0:0]	0	Array
impl2[0:0]	0	Array
impl3[0:0]	0	Array


Name
Value
instruction[16:0]
1000000110010XXXX
opcode[3:0]
0000
programcounter[4:0]
00101
flag[4:0]
00000
M[3,7:0]
00011000


50.000 ns
0 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns
XXXXXXX
0111000
0111000
0111001
0001100
1000000
0001000
1000000
0010000
0001100
1000000
0001000
1000000
0010000
0001100
1000000
0001000
1000000
0010000
111110
XXXX
1110
0011
0000
0010
0000
0100
0011
0000
0010
0000
0100
0011
0000
0010
0000
0100
00000
00001
00010
00011
00100
00101
00110
00111
01000
00100
00101
00110
00111
01000
00100
00101
00110
00111
01000
XXXXX
00000
XXXXX
00000
00001
XXXXXXXX
00000100
00001100
00011000
X1: 50.000 ns