CSE 306
COMPUTER ARCHITECTURE SESSIONAL

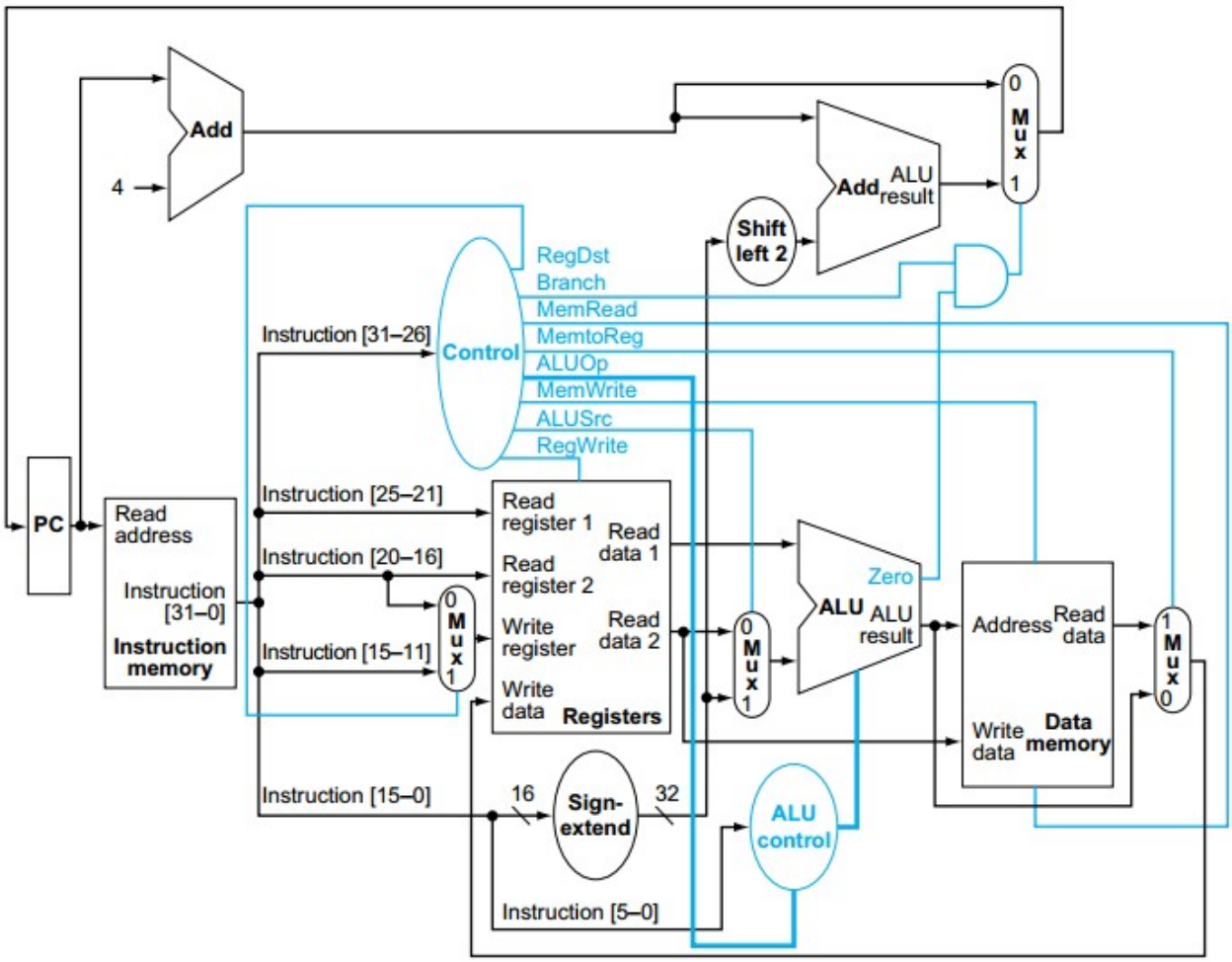
Assignment on 8-bit MIPS Design and Simulation

Group 5
Section : B2
L/T :3-1

Group Member's Std ID: 1705095
1705096
1705097
1705102
1705104

Introduction : In this assignment, we designed an 8-bit processor that implements the MIPS instruction set. It's data paths are 8-bit wide ,hence called 8-bit MIPS. In 1 clock cycle,only one instruction is executed. The main components of the processor are as follows: instruction memory, data memory, register file, ALU, and a control unit. Additional components such as multiplexors, adders are added.

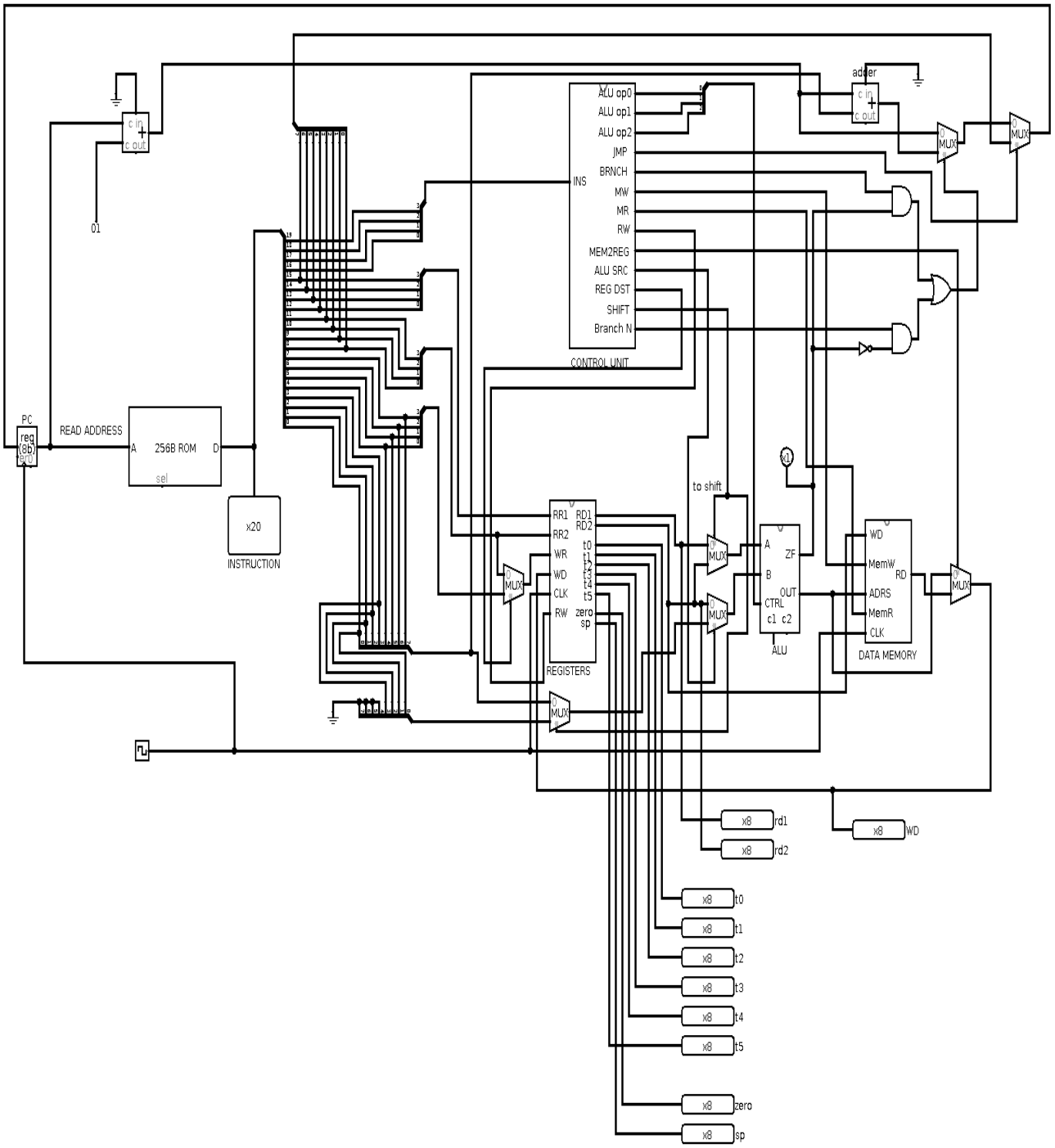
The concept of the diagram below is implemented in our design,



Instruction Set :

Instruction ID	Instruction	Opcode
P	j	0
A	add	1
G	or	2
I	sll	3
F	andi	4
O	bneq	5
E	and	6
H	ori	7
J	srl	8
N	beq	9
M	lw	10
D	subi	11
C	sub	12
L	sw	13
K	nor	14
B	addi	15

Complete Block diagram of 8-bit MIPS processor :



Block diagrams of the main components :

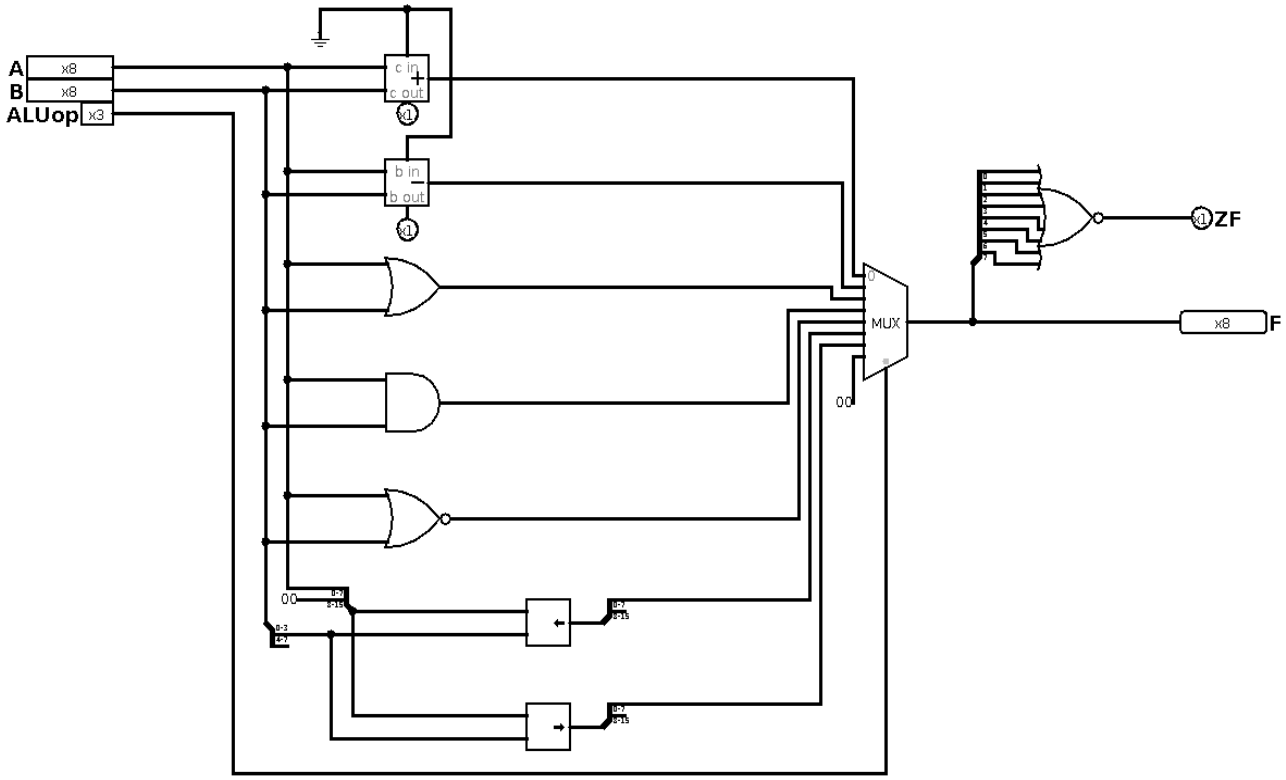


Fig 1: ALU

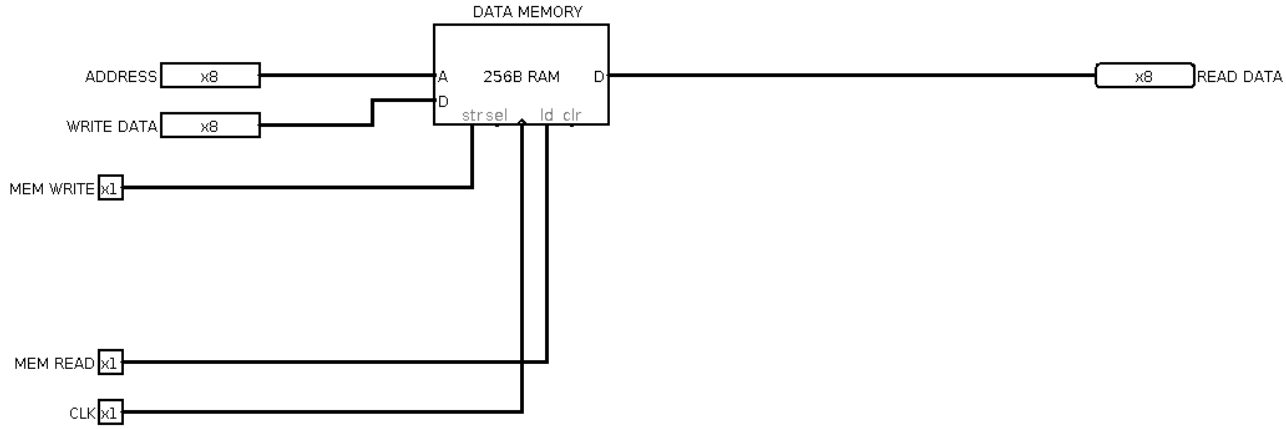


Fig 2 : Data Memory

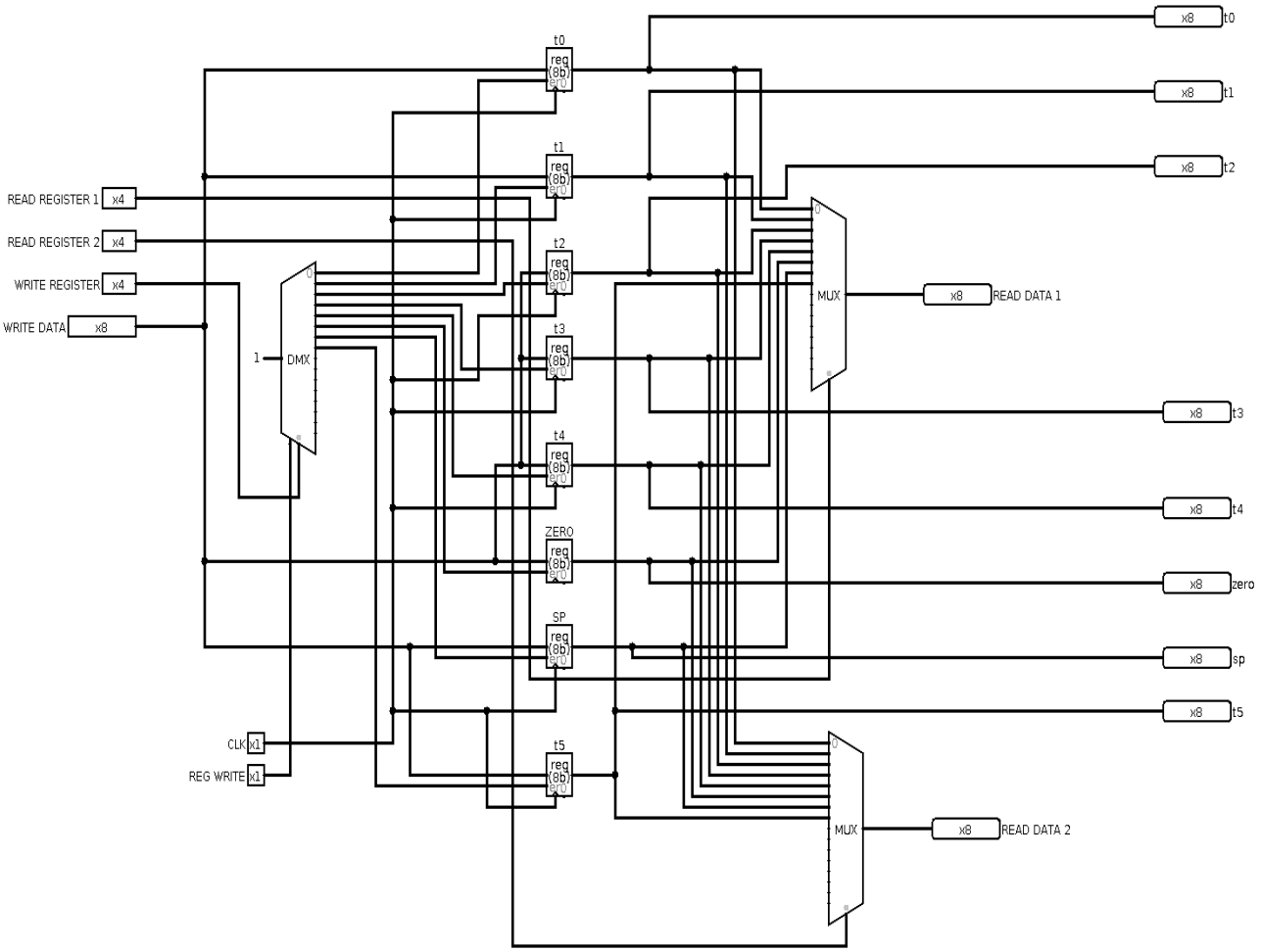


Fig 3 : Registers

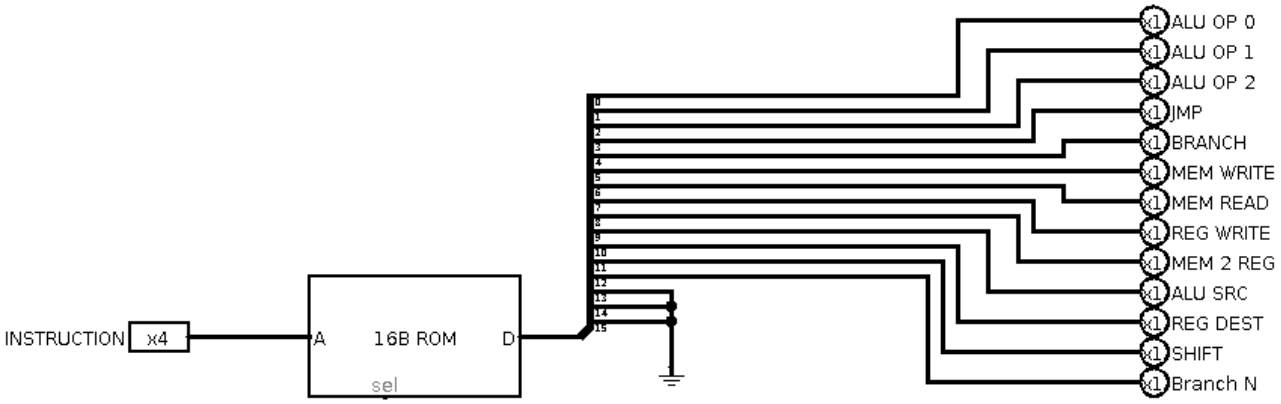


Fig 4 : Control Unit

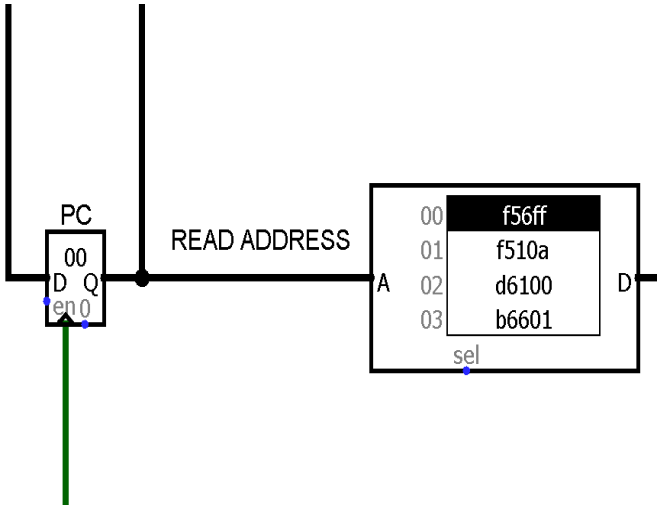


Fig 5 : Instruction Memory with Pc

Implementation of the push and pop instructions : We decoded push and pop instructions into basic load and store instructions. We used store instruction for push and load instruction for pop . For instance,

```
push $t1 ==>     sw $t1, 0($sp)
                 subi $sp, $sp,1
```

and,

```
pop $t1 ==>      addi $sp, $sp,1
                 lw $t1, 0($sp)
```

But for push with offset address ,we used both load and store instructions. For Example,

```
push 3($t0) ==>  lw $t5, 3($t0)
                 sw $t5, 0($sp)
                 subi $sp, $sp,1
```

ICs used with their count :

Control Unit:

Name	Count	IC No.
ROM	1	-

Data Memory:

Name	Count	IC No.
RAM	1	-

Register Unit:

Name	Count	IC No.
MUX	2	DM54LS450
DeMUX	1	748CT154
Registers	8	-

Main Circuit:

Name	Count	IC No.
MUX	7	748C157
NOT	1	7404
AND	2	7408
OR	1	7432
Register	1	-
Adder	2	7483
ROM	1	-

Discussion : We followed the design from the book “Computer Organization and Design” by David A. Patterson and John L. Hennessy. The ALU used in the design is imported from an external source. We used CPP code to convert the pseudo instructions into machine code according to the opcode given in the specification. We have used an extra control bit "BRANCH N" in CONTROL UNIT circuit. This handles the instruction on BNEQ. 1 is added to go to the next instruction whereas in the book, 4 is added. We have implemented SLL and SLR which increased the number of MUX in our circuit by 2.To implement conditional BNEQ we used extra AND gate, OR gate and NOT gate