Computer Organisation and Architecture Laboratory

Verilog Assignment 7

Group 17

Tanishq Prasad (21CS30054)
Krish Khimasia (21CS10037)

Instruction Set Classification(based on operations) :-

- Arithmetic, Logic and Shift
- Load and Store
- Jump and branch
- Stack Handling
- Miscellaneous

Instructions are either **R-type**, **J-type** or **I-type**.

Common Elements:-

- All the instructions are encoded in 32 bits.
- The registers are addressed using 5 bits, that is, $2^5 = 32$ registers possible.

R type:-

OPCODE	Source Reg 1	Source Reg 2	Dest Reg	Shift Amount	Func CODE
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

The first 6 bits will be reserved for the opcode.
The next 5 bits will be reserved for the source register 1.
The next 5 bits will be reserved for the source register 2.
The next 5 bits will be reserved for the destination register.
The next 4 bits would be redundant.
The next 1 bit will be used for the shift amount.
The next 6 bits will be reserved for the function code.

Max shift amount = 1

I type:-

OPCODE	Source Reg 1	Dest Reg	Immediate Value
6 bits	5 bits	5 bits	16 bits

The first 6 bits will be reserved for the opcode.
The next 5 bits will be reserved for the source register (or destination register, depending upon operation).
The next 5 bits will be reserved for the destination register.
The next 16 bits will be reserved for immediate value.

Range of immediate: -32768 to 32767

J type:-

OPCODE	Immediate Value
6 bits	26 bits

The first 6 bits will be reserved for the opcode.
The next 26 bits will be used for the immediate value.

Range of immediate: -33554432 to 33554431

List of Operations:-

Operation	OPCODE (in decimal)	Func CODE (in decimal)
ADD	0	0
SUB	0	1
AND	0	2
OR	0	3
XOR	0	4
NOT	0	5
SLA	0	6
SRA	0	7
SRL	0	8
ADDI	1	NA
SUBI	2	NA
ANDI	3	NA
ORI	4	NA
XORI	5	NA
NOTI	6	NA
SLAI	7	NA
SRAI	8	NA
SRLI	9	NA
LD	10	NA
ST	11	NA
LDSP	12	NA
STSP	13	NA

BR	14	NA
BMI	15	NA
BPL	16	NA
BZ	17	NA
PUSH	18	NA
POP	19	NA
CALL	20	NA
RET	21	NA
MOVE	22	NA
HALT	23	NA
NOP	24	NA

Control Unit:-

OPCODE	ALUop	ALUsource	WriteReg	MemWrite	MemRead	MemReg	Branch	Jump	MoveReg	HaltPC	RegDest	WriteSP	ReadSP	UpdateSP	SPMux	RetMem	PM4	RetPC
0	0	1	1	0	0	0	00	0	0	0	1	0	0	0	0	0	0	0
1	1	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
2	2	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
3	3	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
4	4	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
5	5	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
6	6	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
7	7	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
8	8	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
9	9	0	1	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0
10	1	0	1	0	1	1	00	0	0	0	0	0	0	0	0	0	0	0
11	1	0	0	1	0	x	00	0	0	0	0	0	0	0	0	0	0	0
12	1	0	1	0	1	1	00	0	0	0	0	0	0	0	0	0	0	0
13	1	0	0	1	0	x	00	0	0	0	0	0	0	0	0	0	0	0
14	x	x	0	0	0	x	00	1	0	0	0	0	0	0	0	0	0	0
15	x	x	0	0	0	x	01	0	0	0	0	0	0	0	0	0	0	0
16	x	x	0	0	0	x	10	0	0	0	0	0	0	0	0	0	0	0
17	x	x	0	0	0	x	11	0	0	0	0	0	0	0	0	0	0	0
18	1	x	0	1	0	0	00	0	0	0	x	1	1	0	1	0	0	0

OPCODE	ALUop	ALUsource	WriteReg	MemWrite	MemRead	MemReg	Branch	Jump	MoveReg	HaltPC	RegDest	WriteSP	ReadSP	UpdateSP	SPMux	RetMem	PM4	RetPC
19	1	x	1	0	1	0	00	0	0	0	1	1	1	0	0	0	1	0
20	1	x	0	1	0	0	00	1	0	0	x	1	1	1	1	0	0	0
21	1	x	0	1	0	0	00	0	0	0	x	1	1	0	1	1	1	1
22	x	x	1	0	0	x	00	0	1	0	0	0	0	0	0	0	0	0
23	x	x	x	x	x	x	00	0	0	1	x	x	x	x	x	x	x	x
24	x	x	x	x	x	x	00	0	0	x	x	x	x	x	x	x	x	x

ALUop - This combined with the function code lets the ALU control decide what operation to choose. Code according to ALU Control.

ALUsource - Immediate or register for second operand of ALU.

WriteReg - WriteReg tells whether to write or not.

MemWrite and MemRead - Whether to read/write from memory or not.

MemReg - What to write to the register, the ALU result (at 0) or the memory read (at 1) .

Branch - Whether branch has been enabled or not. If 00 it has not been enabled. 01 for MIR, 10 for BPL, 11 for BZ.

Jump - Used to choose the 26-bit immediate data for BR and CALL.

MoveReg - Used to decide whether we write data from another register or the data which depends on MemReg.

HaltPC - Used during HALT instruction.

RegDest - 1 if we're giving a destination register.

WriteSP - 1 when we want to write to the stack pointer.

ReadSP - 1 when we want to read from the stack pointer.

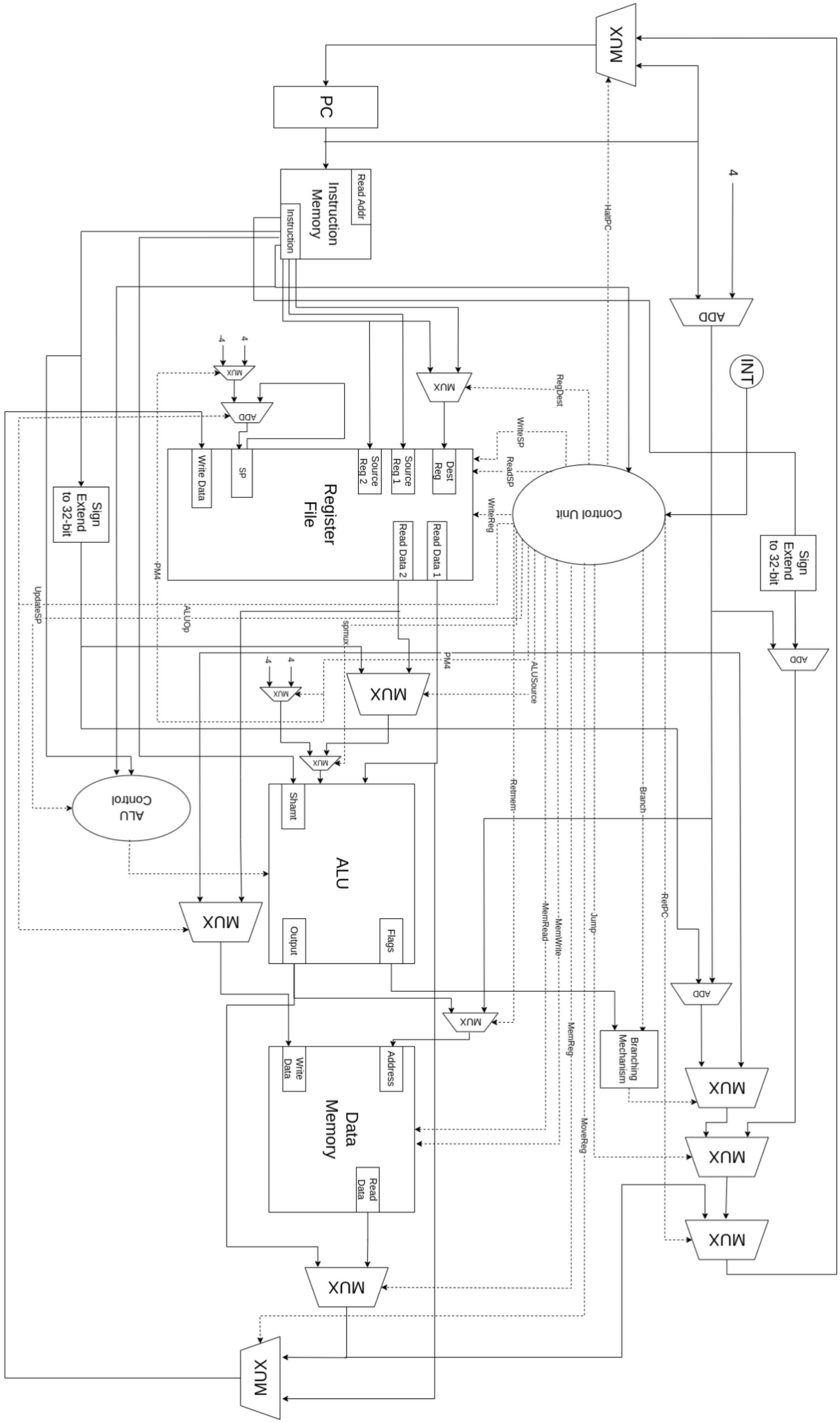
UpdateSP - 1 when we are updating the value of Mem[SP] with PC+4.

SPMux - To choose between +4/-4 (at 1) and source register 2/immediate data (at 0).

RetMem - When 1, it uses PC as the address for memory read.

PM4 - To choose between +4 (at 1) and -4 (at 0) for adding to SP.

RetPC - Used to choose the new value of PC between PC+4 (at 1) and Mem[PC] (at 0).

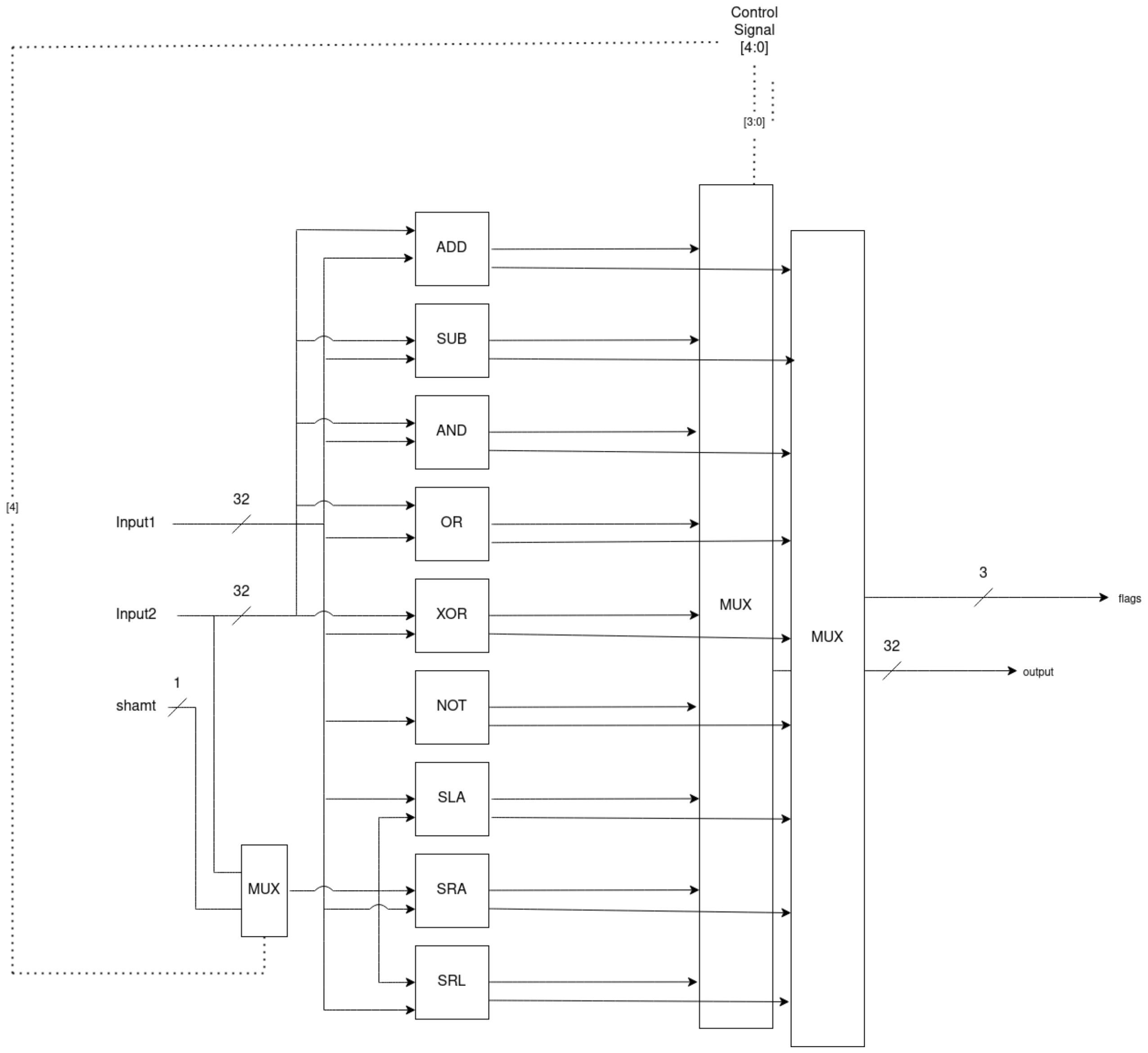

MUX
PC
4
Instruction Memory
Read Addr
Instruction
ADD
INT
HaltPC
RegDest
WriteSP
ReadSP
WriteReg
Control Unit
Register File
Dest Reg
Source Reg 1
Source Reg 2
SP
Write Data
Read Data 1
Read Data 2
-4
4
MUX
ADD
Sign Extend to 32-bit
Sign Extend to 32-bit
PM4
UpdateSP
ALUOp
spmux
PM4
ALUSource
MUX
ADD
ALU Control
ALU
Shamt
Output
Flags
Retmem
Branch
RetPC
MemRead
MemWrite
Jump
MemReg
MoveReg
Branching Mechanism
Data Memory
Address
Write Data
Read Data
MUX

ALU Control:-

Gets input from the main control unit and the function code and decides the necessary operation for the ALU. Generates a 4 bit control signal (explained on the next page).

ALUop (from control unit)	Function Code	Control Signal
0	0	0
0	1	1
0	2	2
0	3	3
0	4	4
0	5	5
0	6	6
0	7	7
0	8	8
1	x	0
2	x	1
3	x	2
4	x	3
5	x	4
6	x	5
7	x	6
8	x	7
9	x	8

ControlSignal[3:0]
- if 0000 then add
- if 0001 then sub
- if 0010 then and
- if 0011 then or
- if 0100 then xor
- if 0101 then not
- if 0110 then sla
- if 0111 then sra
- if 1000 then srl



Branch Mechanism:-

Branch (from control unit)	Flags (from ALU)	Control Signal
00	xxx	0
01	1xx	1
10	0xx	1
11	xx1	1