Verilog Assignment 7
Instruction Format, Data Path, and Control Unit Design
Group 24

Gorantla Thoyajakshi - 21CS10026
Ashwin Prasanth - 21CS30009

1. Instruction format and Encoding

i. R-type instructions:

Opcode	Source Register 1 (Rs)	Source Register 2 (Rt)	Destination Register (Rd)	Shift Amount (shamt)	Function (funct)
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

Instruction	Usage	Opcode	Funct code
ADD	ADD Rd,Rs,Rt	000000	000001
SUB	SUB Rd,Rs,Rt	000000	000010
AND	AND Rd,Rs,Rt	000000	000011
OR	OR Rd,Rs,Rt	000000	000100
XOR	XOR Rd,Rs,Rt	000000	000101
NOT	NOT Rd,Rs,Rt	000000	000110
SLA	SLA Rd,Rs	000000	000111
SRA	SRA Rd,Rs	000000	001000
SRL	SRL Rd,Rs	000000	001001

ii. I-type instructions:

Opcode	Source Register 1 (Rs)	Destination Register (Rt)	Immediate Value
6 bits	5 bits	5 bits	16 bits

Instruction	Usage	Opcode
ADDI	ADDI Rs, Rt, Imm	000001
SUBI	SUBI Rs, Rt, Imm	000010
ANDI	ANDI Rs, Rt, Imm	000011
ORI	ORI Rs, Rt, Imm	000100
XORI	XORI Rs, Rt, Imm	000101
NOTI	NOTI Rs, Rt. Imm	000110
SLAI	SLAI Rs, Imm	000111
SRLI	SRLI Rs, Imm	001000
SRAI	SRAI Rs, Imm	001001
BR	BR Imm	001010
BMI	BMI Rs, Imm	001011
BPL	BPL Rs, Imm	001100
BZ	BZ Rs, Imm	001101
LD	LD Rt, Rs, Imm	001110
ST	ST Rt, Rs, Imm	001111
LDSP	LDSP SP, Rs, Imm	010000
STSP	STSP SP, Rs, Imm	010001
MOVE	MOVE Rt, Rs	010010
PUSH	PUSH Rs	010011
POP	POP Rt	010100
CALL	CALL Imm	010101

iii. Miscellaneous Instructions:

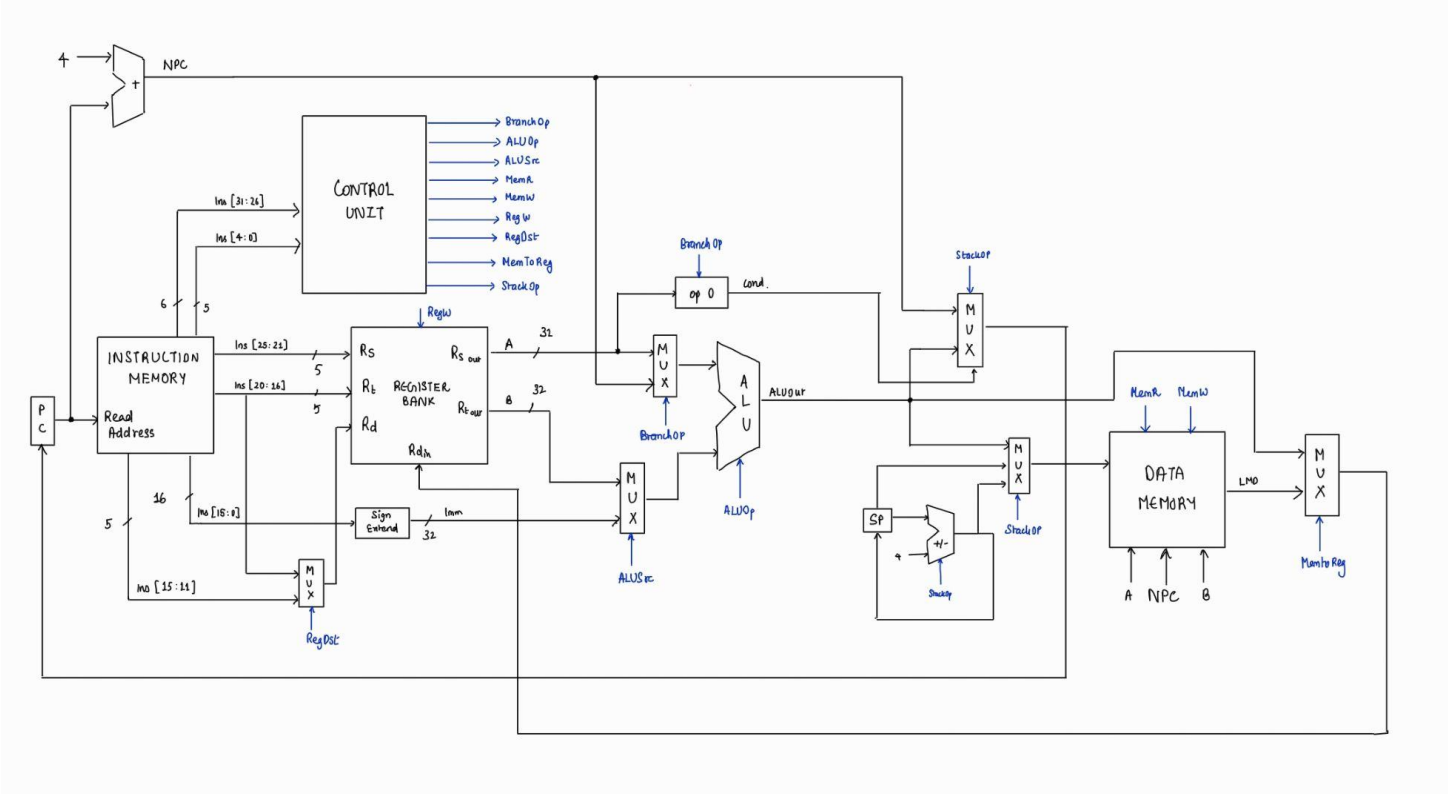
Opcode	Don’t care
6 bits	26 bits

Instruction	Usage	Opcode
HALT	HALT	010110
NOP	NOP	010111
RET	RET	011000

2. Register Usage Convention

Register	Function	Register Number
$R0	Hardwired to 0	0
$R1-$R15	Temporary registers	1-15
$SP	Stack Pointer	16
$PC	Programme counter	17

3. Datapath



4. Control Signals

The control signals used are as follows:

1. **BranchOp:** Denotes which branching operation is to be executed (0 if it's not a branching instruction)

Opcode	BranchOp	Meaning
001010	001	BR
001011	010	BPL
001100	011	BMI
001101	100	BZ
Other	000	Not a branching instruction

2. **ALUOp:** Either specifies the ALU Operation to be carried out, or specifies that the operation should be figured out from the ‘funct’ bits

Opcode	ALUOp	Meaning
000000	0000	Figure out from funct
000001 to 001001	0001 to 1001	Corresponding function (ADD, SUB,... SRA)
001010 to 010001	0001	ADD
Other	-	Not relevant (don’t care)

3. **ALUSrc:** Selects the second source operand for ALU. For R-type operations, B will be chosen. For I-type operations, Imm will be chosen. For other operations, it is irrelevant

Opcode	ALUSrc	Meaning
000000	0	Choose B
000001 to 01001, 010101	1	Choose Imm
Other	-	Not relevant (don’t care)

4. **MemR:** Enables memory read, used in Load, Pop and Ret

Opcode	MemR	Meaning
001110	1	Read from memory
010000	1	Read from memory
010100	1	Read from memory
011000	1	Read from memory
Other	0	No memory read

5. **MemW:** Enables memory write. Used in store, push and call

Opcode	MemR	Meaning
001111	1	Write to memory
010001	1	Write to memory
010011	1	Write to memory
010101	1	Write to memory
Other	0	No memory write

6. **RegW:** Enables write to register bank. Used in arithmetic operations, load, pop and move

Opcode	RegW	Meaning
000000	1	Write to register bank
000001 to 001001	1	Write to register bank
001110	1	Write to register bank
010100	1	Write to register bank
010000	1	Write to register bank
010010	1	Write to register bank
Other	0	No register bank access

7. **RegDst:** Specifies the destination register (Rt or Rd). It is Rd for R-type instructions, and Rt for I-type instructions (wherever it is required)

Opcode	RegDst	Meaning
000001 to 000110, 001110 to 010010, 010100	0	Rt is destination register
000000	1	Rd is destination register
Other	-	Rt/Rd is not used

8. **MemtoReg:** Determines whether value to be written to register comes from ALUOut or LMD. It is ALUOut for arithmetic instructions, and LMD for Load and Pop

Opcode	MemtoReg	Meaning
000000, 000001 to 001001	0	Value comes from ALUOut
001110, 010000, 010100	1	Value comes from LMD
Other	-	Not relevant (don’t care)

9. StackOp: Denotes which stack operation is to be executed (0 if it is not a stack instruction)

Opcode	StackOp	Meaning
010011	000	PUSH
010100	001	POP
010101	010	CALL
011000	011	RET
Other	000	Not a stack instruction