University of Moratuwa
Faculty of Engineering
Department of Electronic & Telecommunication Engineering

EN2031 - Fundamentals of Computer Organization and Design

ISA and Microarchitecture Design

Group Members	
Index No	**Name**
210097M	A.N.T.Silva
210031H	A.A.W.L.R.Amarasinghe
210218M	H.M.S.I.Herath

1. (a) List all the required instructions with proper format showing opcode and the operands and the meaning of each instruction.

Opcode	Operand	Meaning
ADD	RS,RD	RD←RD+RS Add the data in the registers RD and RS and store the result in RD
SUB	RS,RD	RD←RD-RS Subtract the data in registers RD and RS and store the result in RD
AND	RS,RD	RD←RD AND RS Perform bitwise AND for the data in registers RD and RS and store them in RD
OR	RS,RD	RD←RD OR RS Perform bitwise OR for the data in registers RD and RS and store them
PUSH	RS	DM[R7]←RS Push the data of RS register into the memory location pointed by the R7 register
POP	-	Pop from the stack
LOAD	RS,RD	RD←DM[RS] Load the data in the memory location pointed by RS into RD
LOADI	RD, imm[7:0]	Load the 8 bit constant into register RD
STORE	RS,RD	DM[RD]←RS Store the data of register RD in the memory location pointed by RS
MOVE	RS,RD	RD←RS Copy the data of register RS into RD
LOADIN	RS,RD	RD←RS Copy the data of register RS into RD
BRN	Offset	PC←PC+Offset

BREQ	RS,Offset	If equal PC←PC+Offset
BRLEQ	RS,Offset	If less than or equal PC←PC+Offset
BRGEQ	RS,Offset	If greater than or equal, PC←PC+Offset

• ALU instructions : ADD, SUB, AND, OR
• Control flow instructions : BRN, BREQ, BRLEQ, BRGEQ
• LOAD instructions : LOAD, LOADI, LOADIN
• STORE instructions : STORE
• MOVE instructions : MOVE
• Stack instructions : PUSH, POP

(b) Develop the Instruction Formats required to support the above-mentioned instructions and provide justifications for your decisions.

We categorise instructions into 4 types as register(R), Immediate(I), memory access(R) and stack operation(S) types.

- Register Instructions

AND, SUB, OR, ADD, MOV

func	Rs[10:8]	Rd[7:5]	Opcode[0:4]

- Immediate Instructions

LOADI, LOADUI, STOREI

Imm[15:8]	Rd[7:5]	Opcode[0:4]

- Memory Access Instructions

LOAD, STORE,

	Radd[10:8]	Ra[7:5]	Opcode[0:4]

- Stack Operations

POP, PUSH BREQ, BRLEQ,BRGEQ, JUMP

offset	Ra[7:5]	Opcode[0:4]

2. Propose an appropriate encoding scheme for the opcodes that will facilitate easy decoding of the instructions.

We used binary encoding scheme where each instruction is assigned a unique binary opcode value. In our design there are 16 instructions. We use first two bits in the opcode for defining the type of the instruction.
00 - register type
01 - immediate type
10 - memory access instructions
11 - stack operations

R type contains ADD, SUB, OR, MOV, AND. For these instructions last 3 bits of operand decodes which ALU command will run.

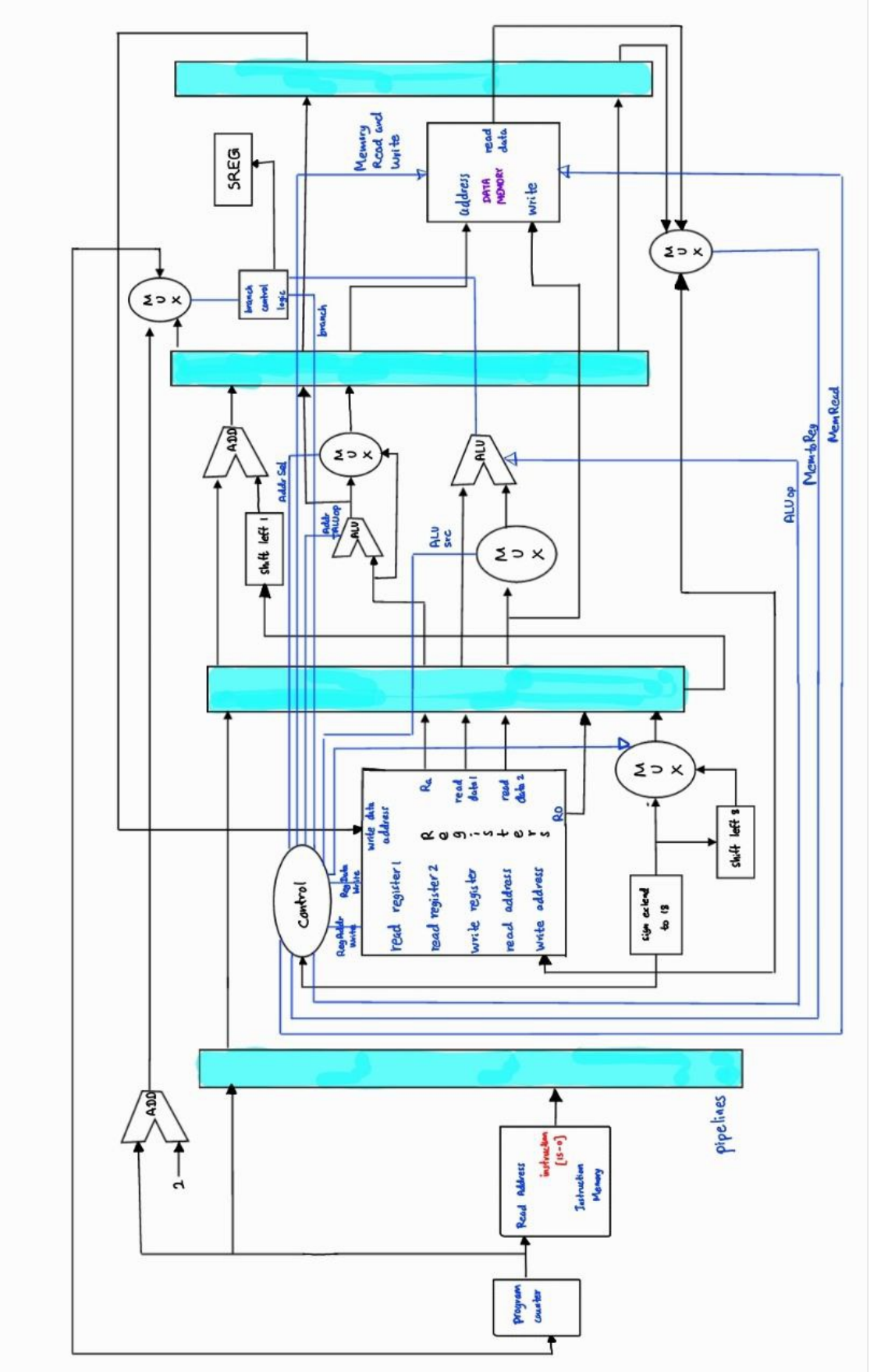
Immediate type instructions are encoded by next 3 bits for the instructions with immediate operands.

Third bit of the memory access instructions determines whether instructions are memory read or memory write etc.

Here's an example of how the opcode encoding scheme could be used to encode some of the instructions in the instruction set.

Instruction	Operands	Opcode	RS	RD1,RD2
ADD	RS,RD1,RD2	00001	3 bits	3 bits
SUB	RS,RD1,RD2	00010	3 bits	3 bits
AND	RS,RD1,RD2	00011	3 bits	3 bits
OR	RS,RD1,RD2	00100	3 bits	3 bits
PUSH	RS	11101	3 bits	3 bits
POP	-	11110	-	-
LOAD	RS,RD	10111	3 bits	3 bits
LOADI	RS,IMM	01000	3 bits	8 bits
LOADIN	RS,RD	01001	3 bits	3 bits
STORE	RS,RD	10010	3 bits	3 bits
MOVE	RS,RD	00011	3 bits	3 bits

3. Draw a clearly labelled datapath with all functional elements for your chosen design approach. Show all the control signals for the datapath elements.



4. i) Explain your chosen design approach for the microarchitecture: Hardwired or Microprogrammed with clear justification.

For this design approach we used a hardwired approach. It needed to achieve higher speed. As it is not going to use as a general purpose processor which runs operating system, less customizations are needed.

As we are implementing RISC-V similar 16 bit length ISA we can implement a high speed simple design through hardwired approach because it is possible to decode the required instructions with a 16 bit wide instruction length.

ii) Design the controller for the datapath that you have designed above, giving reasons for your decisions. Clearly illustate the controller showing the inputs and outputs.

Controller design is shown under the part 3) and the reasons for the decisions are as follow. Hardwired control unit is used and decoded control signals are send from the control units for muxes to decode further and to required blocks.

Main control paths are ALU enable control signals, memory read enable signal, memory write enable signal, branch control signal, write or read enable signals for registers. For decoding further control signals are sent to multiplexers.