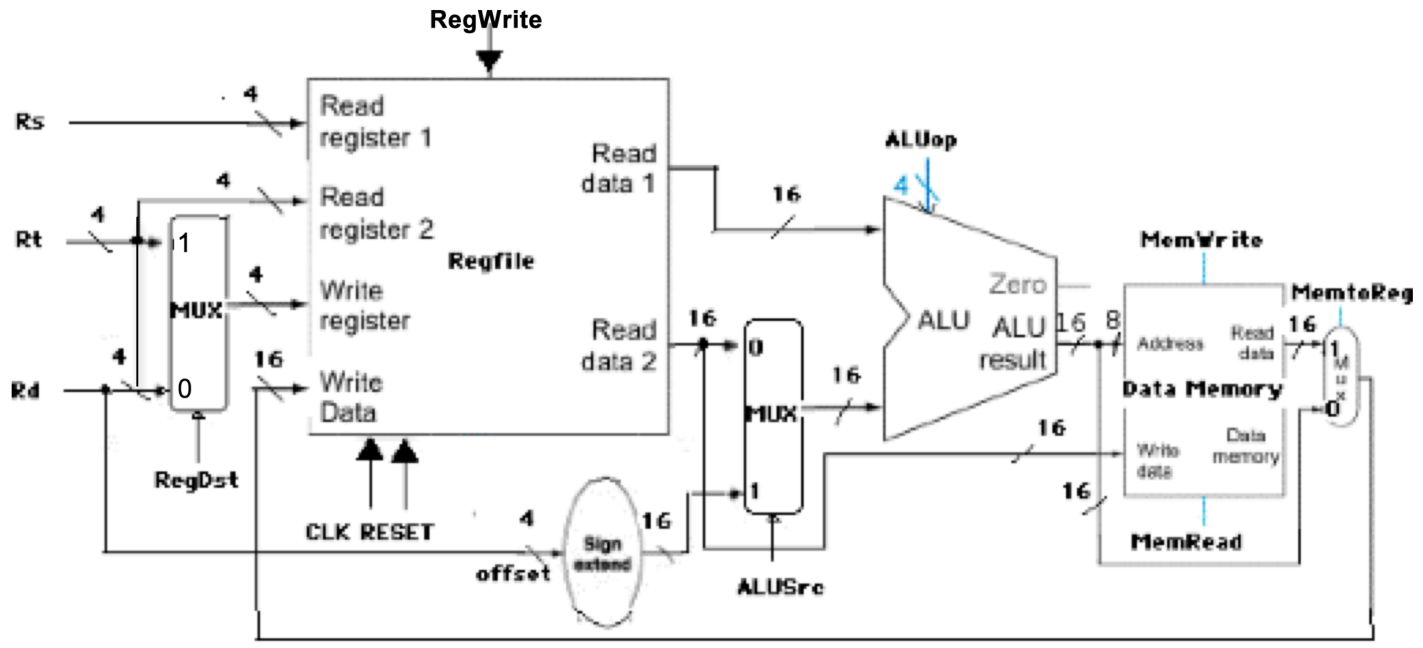
Assignment for Laboratory 10

Computer Science 240

Below is the full datapath for the Mini-MIPS computer from lab:



The control lines for the various multiplexers and Data Memory can be produced as a simple function of the opcode, as defined in the following table:

Instruction	Opcode	RegDst	RegWr	ALUSrc	MemRd	MemWr	MemtoReg
LW	0000	1	1	1	0	1	1
SW	0001	1	0	1	1	0	0
ADD	0010	0	1	0	1	1	0
SUB	0011	0	1	0	1	1	0
AND	0100	0	1	0	1	1	0
OR	0101	0	1	0	1	1	0
SLT	0110	0	1	0	1	1	0
BEQ	0111	0	0	0	1	1	0
JMP	1000	0	0	0	1	1	0

- **RegDst** (select line to MUX, chooses whether Rd or Rt goes to the Write data input on the register file)
  Selects destination register: if 0, the destination register is Rd. If 1, the destination register is Rt.
- **RegWr** (control line to Register File)
  If 1, writes the value on the Write data input to the register on the Write register input.
- **ALUSrc** (select line to MUX which chooses the source of the second ALU operand)
  If 0, the operand is the value of the Rt register. If 1, the operand is the sign-extended, lowest 4 bits of the instruction.
- **MemRd** (read control signal to data memory, active low)
  If 0, read the value of the address specified from Read Data
- **MemWr** (write control signal to data memory, active low)
  If 0, write a new data value at Write Data to the address specified
- **MemtoReg** (select line to MUX chooses the value to be written back to the register file)
  If 0, the value comes from the ALU. If 1, the value comes from data memory.

Each Rs,Rt,Rd operand represents either a register number or an offset. When it is a register number, the contents of the register specified by the number is used in the operation. When it is an offset, the offset itself is used in the operation.

For each set of inputs, predict the ALU result, and answer the questions.

Assume each instruction produces results that are used by the subsequent instructions.

ALUop	Rs Rt Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	1 1 5	1	0	0	1	1	0	2

Explain the operation and result:
ADD R1 R1 R5

R1 (1) + R1 (1) = 2 to R5

R5 = 2

ALUop	Rs Rt Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	**0 5 0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**

Assume this is a **SW** instruction. What tells you this?
MemWrite = 0, which is only on a SW instruction

SW R5,0(R0)

Explain the operation and result: what value will be stored into what address?
ALU calculates the address to be accessed
Address = R0 + offset 0 = 0
Contents of R5 (2) to address 0

Address 0 = 2

ALUop	Rs Rt Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	5 1 0	0	1	1	1	0	0	2

Again, assume a **SW** instruction.

Explain the operation and result: what value will be stored into what address?
SW R1,0(R5)

ALU calculates the address to be accessed
Address = R5 (2) + offset 0 = 2
Contents of R1 (1) to address 2

Address 2 = 1

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	0	3	0	1	1	1	0	1	1	0

This time, assume a **LW** instruction. What tells you this?
MemRead = 0, which is only on a LW instruction

LW R3,0(R0)

Explain the operation and result: what value is loaded from what address into what register?
ALU calculates the address to be accessed
Address = R0 + offset 0 = 0
Contents of Address 0 (2) to R3

R3 = 2

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	3	4	0	1	1	1	0	1	1	2

Again, assume a **LW** instruction.
LW R4,0(R3)

Explain the operation and result: what value is loaded from what address into what register?
ALU calculates the address to be accessed
Address = R3 (2) + offset 0 = 2
Contents of Address 2 (1) to R4

R4 = 1

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	3	5	2	1	0	0	1	1	0	4

Explain the operation and result.
ADD R3,R5,R2

R3 (2) + R5(2) = 4 to R2

R2 = 4

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
7	3	2	6	1	0	0	1	1	0	1

Explain the operation and result.
SLT R3, R2,R6

If R3 (2) < R2 (4), R6 = 1 (otherwise R6 = 0)

R6 = 1

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
1	2	6	7	1	0	0	1	1	0	5

Explain the operation and result.
OR R2,R6,R7

R2 (0100) or R6 (0001) = 0101 (5) to R7

R7 = 5

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
6	2	1	2	1	0	0	1	1	0	3

Explain the operation and result.
SUB R2,R1,R2

R2 (4) – R1(1) = 3 to R2

R2 = 3

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
0	2	7	4	1	0	0	1	1	0	1

Explain the operation and result.
AND R2,R7,R4

R2 (0011) and R7 (0101) = 0001 (1) to R4

R4 = 1

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	**3**	**7**	**2**	**0**	**1**	**1**	**1**	**0**	**0**	**4**

Explain the operation and result.
SW R7,2(R3)

ALU calculates the address to be accessed
Address = R3 (2) + offset 2 = 4
Contents of R7 (5) to address 4

address 4 = 5

ALUop	Rs	Rt	Rd	RegWrite	RegDst	ALUSrc	MemRead	MemWrite	MemtoReg	**ALU result**
2	3	8	0	1	1	1	0	1	1	2

Explain the operation and result.
LW R8,0(R3)

ALU calculates the address to be accessed
Address = R3 (2) + offset 0 = 2
Contents of Address 2 (1) to R8

R8 = 1

Assuming all of the previous intstructions have been executed in order, record the final values of

Registers 0 – 8:
R0: 0
R1: 1
R2: 3
R3: 2
R4: 1
R5: 2
R6: 1
R7: 5
R8: 1

Data stored at memory addresses 0, 2, and 4:
Addr 0: 2
Addr 2: 1
Addr 4: 5