Bangladesh University of Engineering and Technology

July 4, 2021

CSE-306: Computer Architecture Sessional

Assignment on 8-bit MIPS Pipelined Execution

Submitted By
Section: B1
Group: 03
1705068
1705069
1705070
1705071
1705074

Contents

1 Introduction

MIPS is a reduced computer instruction set architecture which defines the interface between a user and a microprocessor. In this offline, we designed a 8-bit processor that supports pipelined datapath for a small subset of MIPS instruction set. Each intruction is divided into five different stages:

- Instruction Fetch (IF)
- Instruction Decode (ID)
- Execution and Address Calculation (EX)
- Data Memory Access (MEM)
- Write Back (WB)

and the length of the clock cycle is equals the maximum time to execute and single stage so for each instruction up to five clock cycle is needed.

In this assignment, we designed a 8-bit processor that supports only R-type instruction set.

2 Instruction Set

Instruction ID	**Instruction Type**	**Instruction**	**Category**
A	R	add	Arithmetic
B	R	sub	Arithmetic
C	R	and	Logic
D	R	or	Logic

Table 1: Instruction Set

3 Instruction Format

R-type	OpCode 4-bits	Src Reg1 4-bits	Src Reg2 4-bits	Dst Reg2 4-bits	Shift Amount 4-bits

Table 2: R-type Instruction Set

4 Pipelined Datapath

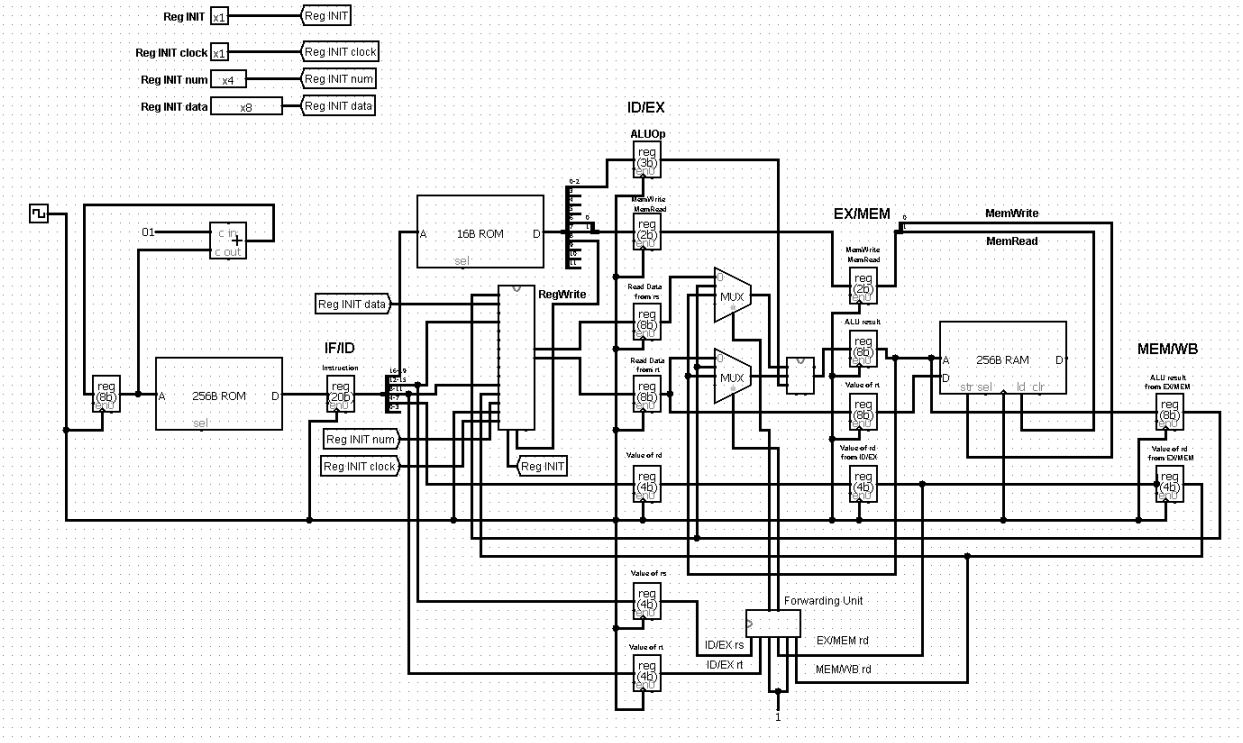

Figure 1: Block diagram of Pipelined Datapath

Our pipelined datapath consists of a program counter register PC, a instruction memory ROM, a control unit ROM, a register file, a forwarding unit, an 8bit ALU, a data memory RAM and four pipeline registers. It also includes some basic components like multiplexers, comparators, adder and basic gates.
The pipeline registers are as follows:

- IF/ID Register between instruction memory and register file
- ID/EX Register between register file and ALU & forwarding unit
- EX/MEM Regsiter between ALU & forwarding unit and data memory
- MEM/WB Regsiter between data memory and Register file

5 Pipeline Registers

We have four pipeline registers.

- IF/ID Register
- ID/EX Register
- EX/MEM Register
- MEM/WB Register

5.1 IF/ID Register

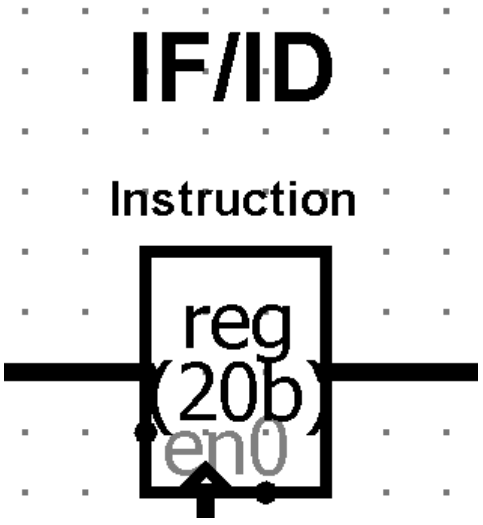

Figure 2: Block diagram of IF/ID Register

The size of the IF/ID register is **20 bits**. It consists of only 1 register.It stores the instruction value in each clock cycle.

5.2 ID/EX Register

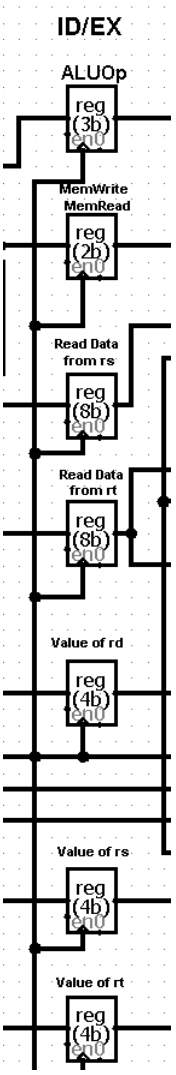

Figure 3: Block diagram of ID/EX Register

The size of the ID/EX register is **33 bits**. It consists of 7 registers of different size of data bits .It stores the ALUOp control bit, MemRead and MemWrite control bits, data from rs register, data from rt register, value of rd, rs, rt registers in each clock cycle.

5.3 EX/MEM Register

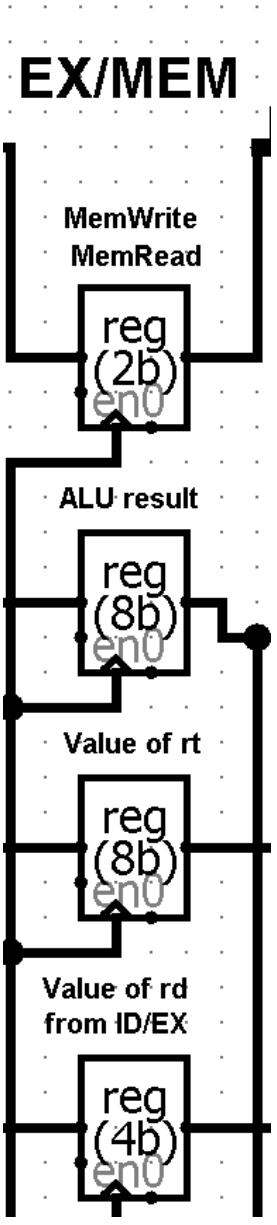

Figure 4: Block diagram of EX/MEM Register

The size of the EX/MEM register is **22 bits**. It consists of 4 registers of different size of data bits .It stores the MemRead and MemWrite control bits from ID/EX register, result from ALU, value of rt register, value of rd register from ID/EX register in each clock cycle.

5.4 MEM/WB Register

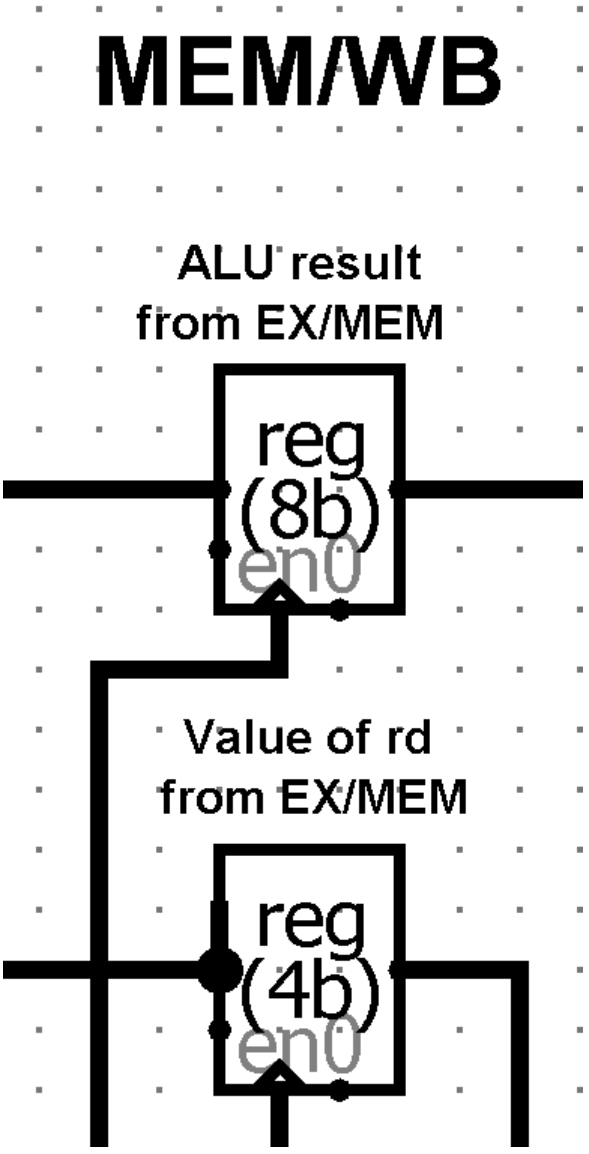

Figure 5: Block diagram of MEM/WB Register

The size of the MEM/WB register is **12 bits**. It consists of 2 registers of different size of data bits .It stores the result from ALU from EX/MEM register, value of rd register from EX/MEM register in each clock cycle.

6 Forwarding Unit

The forwarding unit is designed to eliminate 3 kinds of hazard:

- EX Hazard
- MEM Hazard
- Double Data Hazard

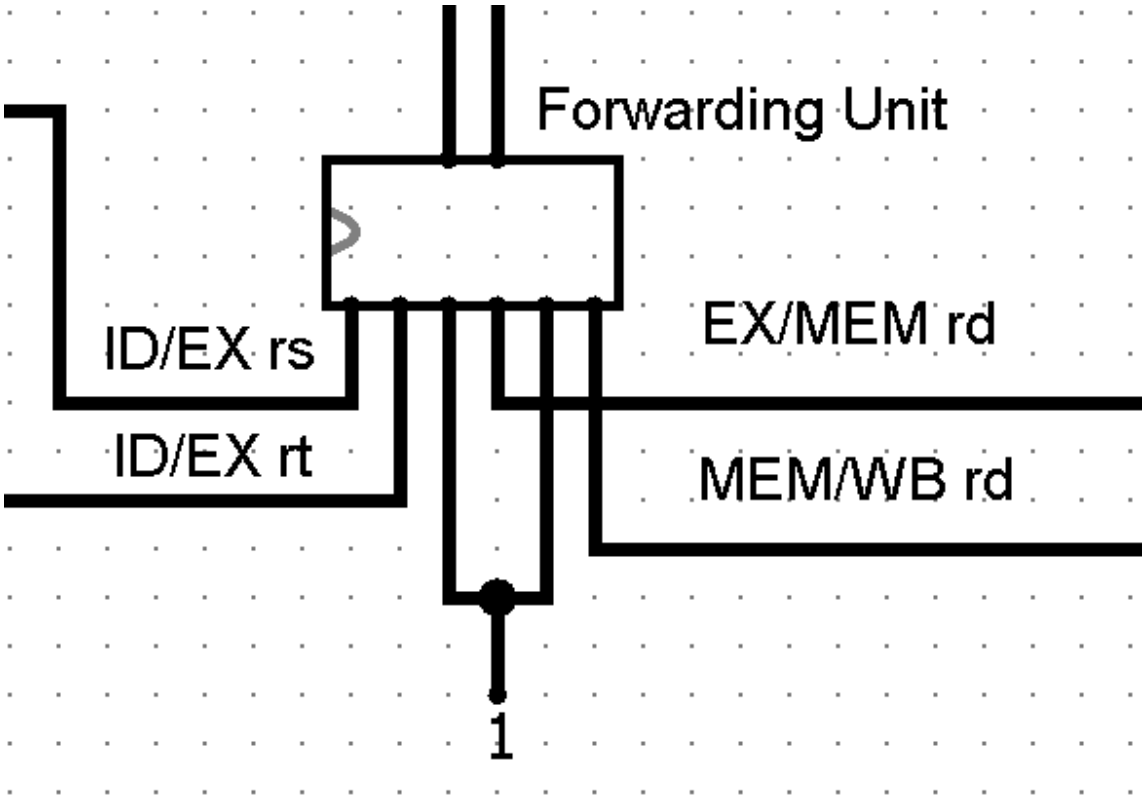

Figure 6: Block diagram of Forwarding Unit

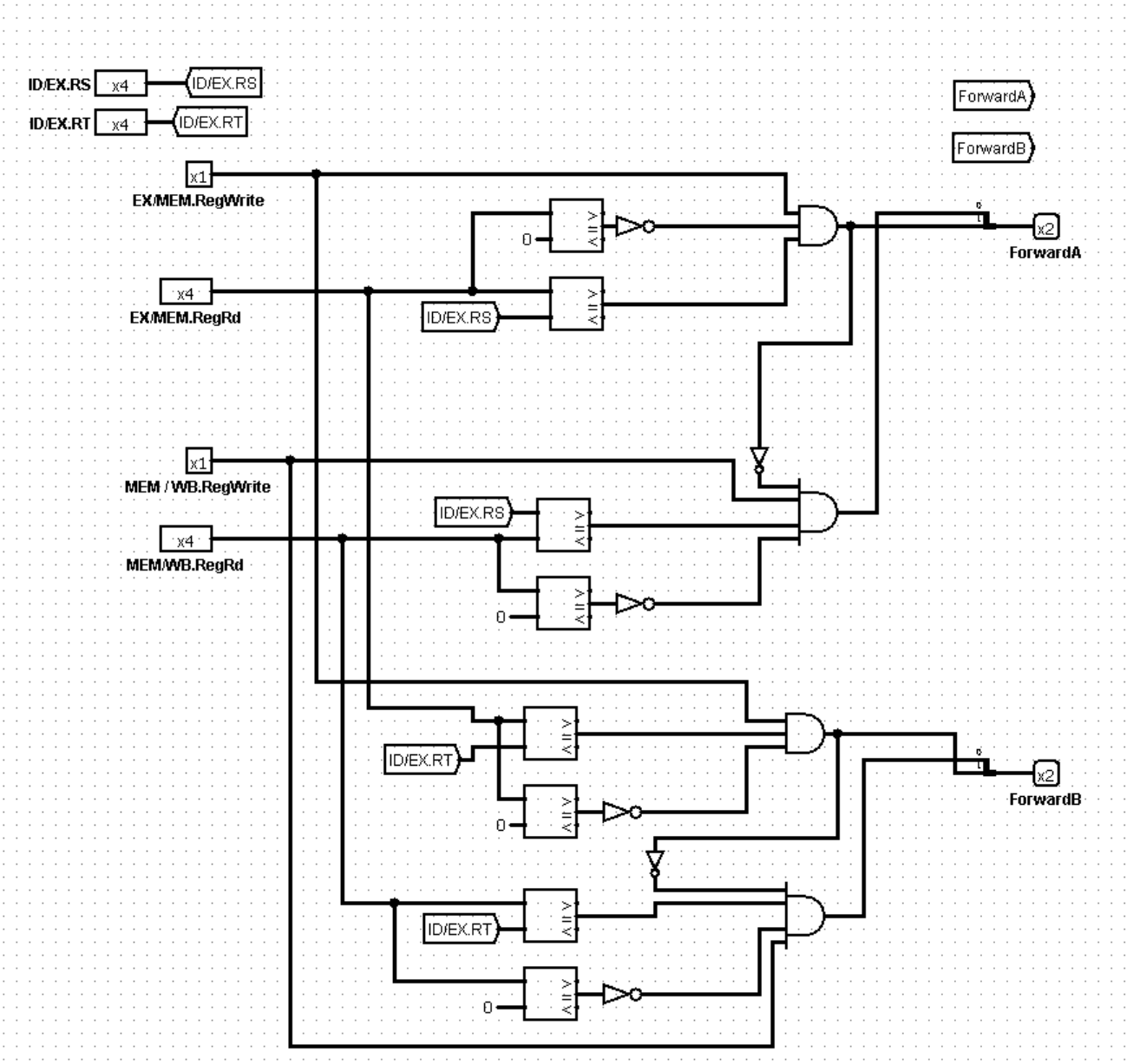

Figure 7: Forwarding Unit

6.1 EX Hazard

Occurs when the dependent instruct is in the EX stage and the prior instruction is in MEM stage.

The conditions that are used both for detecting hazards and the control signal that are used to forward data from registers are given below:

Algorithm 1: EX Hazard

if *EX/MEM.RegWrite and* $EX/MEM.RegisterRd \neq 0$ *and*

EX/MEM.RegisterRd = ID/EX.RegisterRs **then**
| ForwardA = 10

if *EX/MEM.RegWrite and* $EX/MEM.RegisterRd \neq 0$ *and*

EX/MEM.RegisterRd = ID/EX.RegisterRt **then**
| ForwardB = 10

6.2 MEM Hazard

When the dependent instruction is in the EX stage and the prior instruction is in WB stage.

The conditions that are used both for detecting hazards and the control signal that are used to forward data from registers are given below:

Algorithm 2: MEM Hazard

if *MEM/WB.RegWrite and* $MEM/WB.RegisterRd \neq 0$ *and*

MEM/WB.RegisterRd = ID/EX.RegisterRs and not (EX/MEM.RegWrite

and $EX/MEM.RegisterRd \neq 0$ *and EX/MEM.RegisterRd =*

ID/EX.RegisterRs) **then**
| ForwardA = 01

if *MEM/WB.RegWrite and* $MEM/WB.RegisterRd \neq 0$ *and*

MEM/WB.RegisterRd = ID/EX.RegisterRt and not (EX/MEM.RegWrite

and $EX/MEM.RegisterRd \neq 0$ *and EX/MEM.RegisterRd =*

ID/EX.RegisterRt) **then**
| ForwardB = 01

6.3 Double Data Hazard

Occurs when the dependent instruction is in EX stage and it depends on two prior instructions, one of which, is in MEM stage, and the other is in WB stage.

For double data hazard, the result is forwarded from the MEM stage because the result in the MEM stage is the more recent result.

7 Simulator

Logisim 2.7.1

8 Discussion

In this offline, we designed a 8-bit processor that supports pipelined datapath only for R-type instruction set which is a subset of MIPS instruction set.

As, each instruction is divided into five stages and length of clock cycle is same as maximum time to execute a single stage so each instruction takes upto 5 clock cycle.

While designing the processsor, we have used minimum number of ICs. We have used a 8-bit Arithmetic Logic Unit(ALU), a Register file, Ram, Rom, control unit, five pipeline registers, a forwarding unit and some logic gate. We also designed in a way that data bus and address bus are multiplexed and each bus has exactly 8-bits.

Above all, we have tried to make our design as simple as possible.