CSE306 - Computer Architecture Sessional

Assignment 3

8-bit MIPS Pipelined Execution

Submitted by:

Group 2

1705062 Jawad Ul Kabir

1705063 Md. Mahfuzur Rahman Rifat

1705065 Taseen Mubassira

1705066 Ataf Fazledin Ahamed

1705067 Nishat Farhana Purbasha

4 July, 2021

1 Introduction

Microprocessor without Interlocked Pipe-lined Stages (MIPS) is a Reduced Instruction Set Computer (RISC) Instruction Set Architecture (ISA). In this particular version of MIPS, only one instruction format (R-type) has been implemented along with a pipelined datapath.

Pipelining is a technique that is used to achieve parallelism among the instructions in a sequential form. Pipeline is introduced in MIPS processor to achieve performance and ensure the proper usage of resources. By decreasing the number of stages between two instructions, pipelining increases the number of simultaneously executable instructions. However, pipelining does not reduce the time to complete a single instruction- rather it improves the instruction *throughput*.

The pipeline datapath is separated into five stages:

- **IF:** Instruction fetch
- **ID:** Instruction decoder
- **EX:** Execution and address calculation
- **MEM:** Data memory access
- **WB:** Write back

Each of these stages will take one clock cycle to execute. So, a single instruction will take 5 clock cycles to be fully executed and 5 instructions will be able to execute partially by that time.

Opcode	Src Reg 1	Src Reg 2	Dst Reg	Shft Amnt
4-bits	4-bits	4-bits	4-bits	4-bits

ID	**Name**	**Type**	**Opcode**	**Control**	**Control (Hex)**
A	add	R	0001	100000 000 001	801
G	or	R	0010	100000 010 001	811
C	sub	R	1100	100000 001 001	809
E	and	R	1110	100000 011 001	819

We will consider three types of data hazard can occur. These hazards can be solved by **forwarding**.

- **EX Hazard:** Occurs when the dependent instruction is in the EX stage and the prior instruction is in MEM stage.
- **MEM Hazard:** Occurs when the dependent instruction is in the EX stage and the prior instruction is in WB stage.
- **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.

2 Complete Block Diagram of the Pipelined Datapath

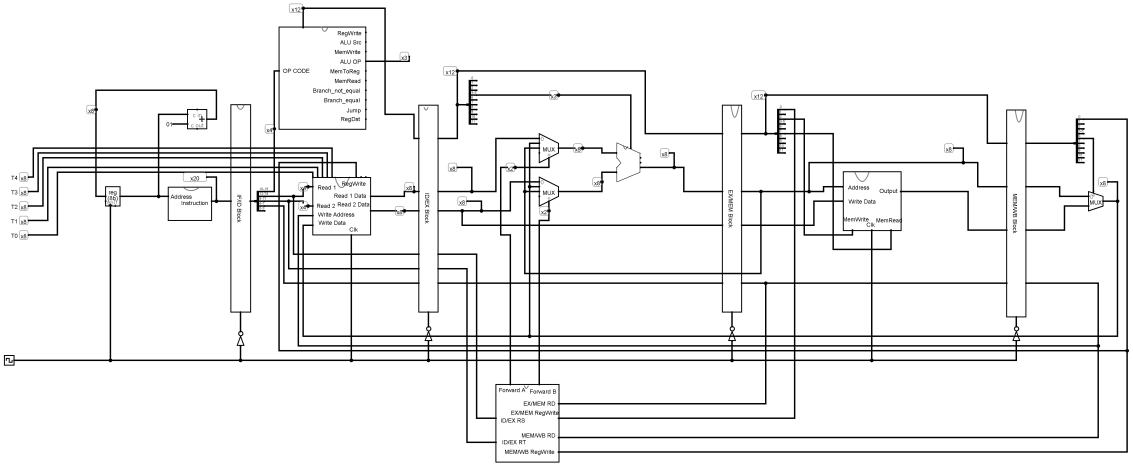


Figure 1: Block Diagram of Pipelined Datapath

3 Block Diagrams and Size of Pipeline Registers

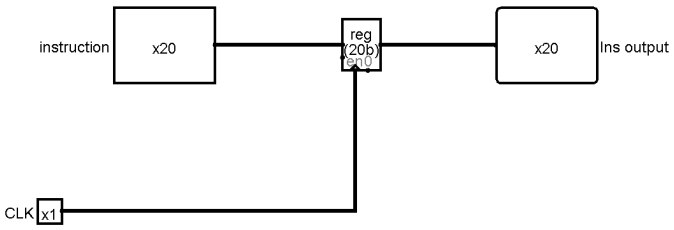


Figure 2: Block Diagram of IF/ID Block

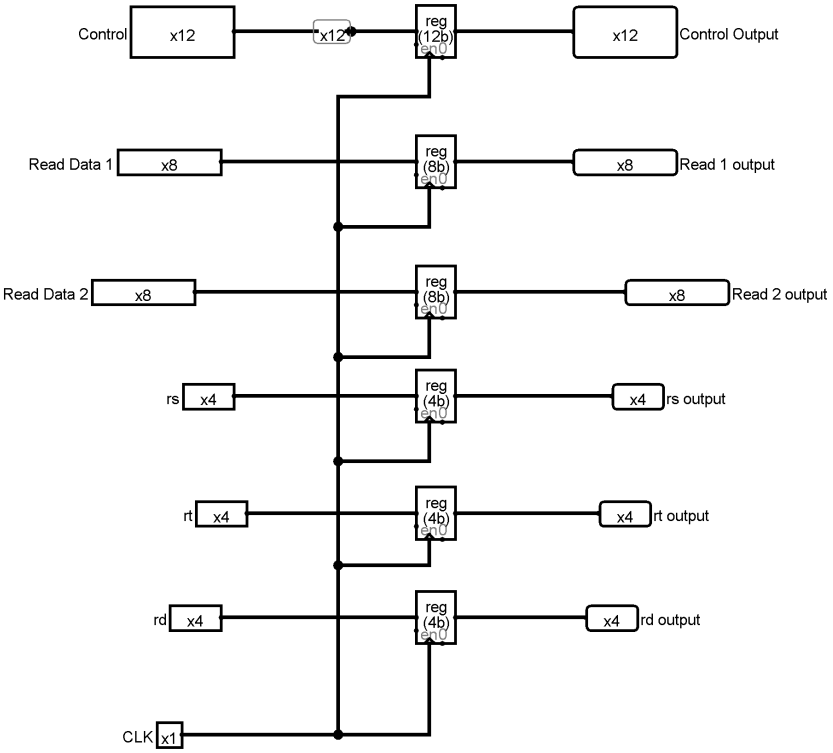


Figure 3: Block Diagram of ID/EX Block

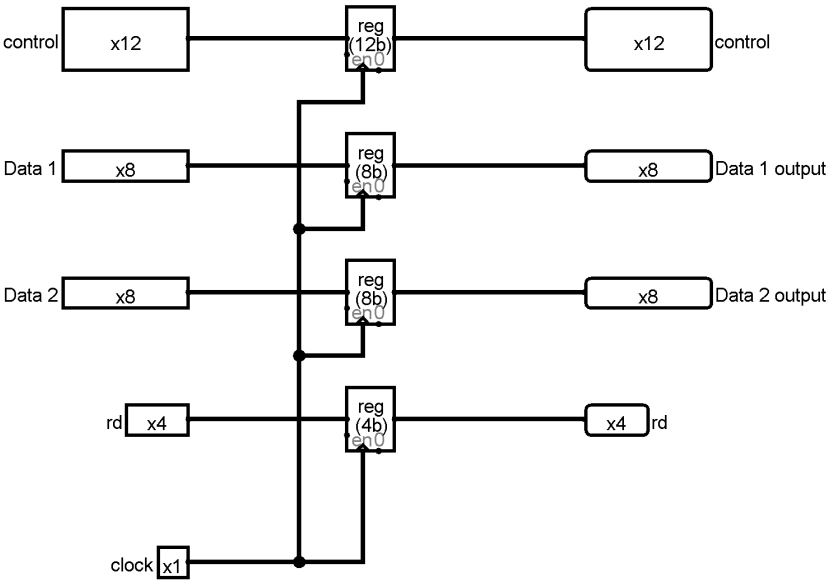


Figure 4: Block Diagram of EX/MEM Block

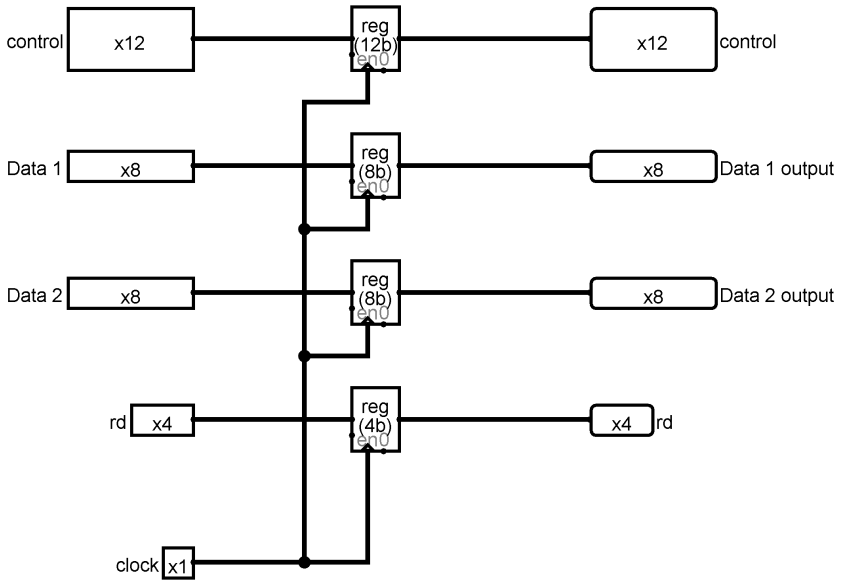


Figure 5: Block Diagram of MEM/WB Block

Pipeline Register	Number of Bits
IF/ID	20
ID/EX	40
EX/MEM	32
MEM/WB	32

Table 1: Size of Pipeline Registers

4 Mechanism and Block diagram of Forwarding Unit

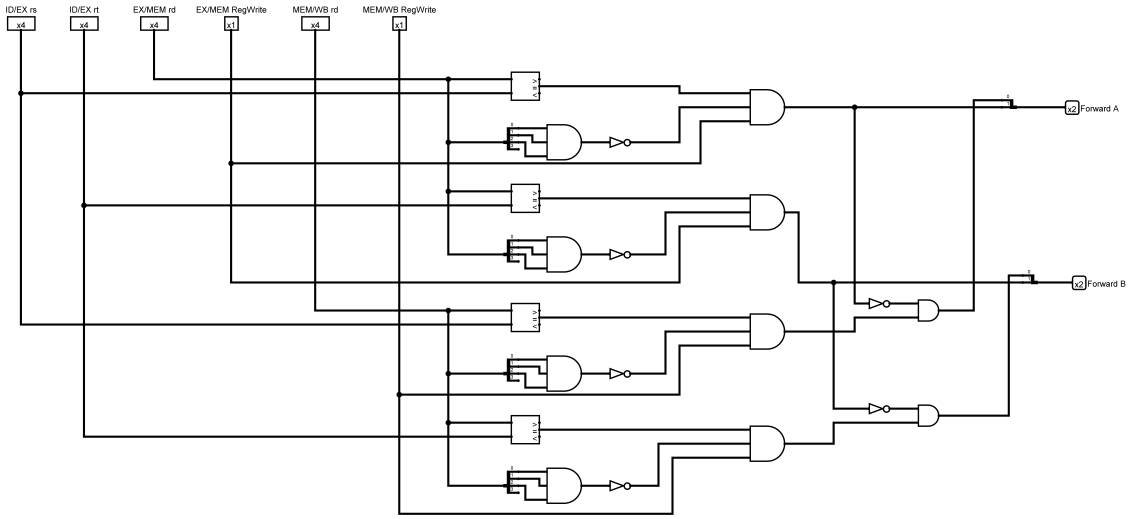


Figure 6: Block Diagram of Forwarding Unit

In this forwarding unit, we resolve data hazard using forwarding technique. Here, we have set flags `ForwardA & ForwardB` by the following conditions:

- **EX Hazard:**

```
if ((EX/MEM.RegWrite == 1)
and (EX/MEM.RegisterRd != 0)
and (EX/MEM.RegisterRd == ID/EX.RegisterRs))
{ ForwardA = 10 }


if ((EX/MEM.RegWrite == 1)
and (EX/MEM.RegisterRd != 0)
and (EX/MEM.RegisterRd == ID/EX.RegisterRt))
{ ForwardB = 10 }
```

- **MEM Hazard:**

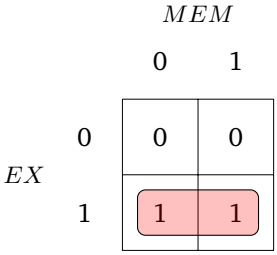
```
if ((MEM/WB.RegWrite == 1)
and (MEM/WB.RegisterRd != 0)
and not (
      (EX/MEM.RegWrite)
      and (EX/MEM.RegisterRd != 0)
      and (EX/MEM.RegisterRd != ID/EX.RegisterRs)
)
and (MEM/WB.RegisterRd == ID/EX.RegisterRs))
{ ForwardA = 01 }


if ((MEM/WB.RegWrite == 1)
and (MEM/WB.RegisterRd != 0)
and not (
      (EX/MEM.RegWrite)
      and (EX/MEM.RegisterRd != 0)
      and (EX/MEM.RegisterRd != ID/EX.RegisterRt)
)
and (MEM/WB.RegisterRd == ID/EX.RegisterRt))
{ ForwardA = 01 }
```

It is to note that the Double Data Hazard is resolved due to the conditions imposed in the third conditions for both MEM Hazards' if clause.

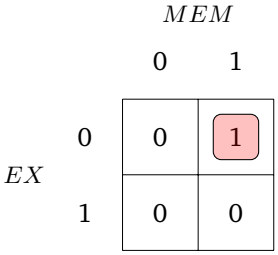
EX Hazard	MEM Hazard	Forward
0	0	00
0	1	01
1	0	10
1	1	10

Table 2: Truth Table for Forward Bits

K-Maps:



Forward MSB
$MSB = EX$



Forward LSB
$LSB = EX'.MEM$

5 Simulator Info

Logisim - Windows (Version 2.7.1)

6 Discussion

- A ROM was used as instruction memory which had 8-bit address. Each access to the instruction memory provided a 20-bit MIPS instruction.
- This MIPS architecture supports pipelined datapath for a subset (`add`, `sub`, `or`, `and`) of MIPS instruction set.
- We handled all the three hazards (EX hazard, MEM hazard and Double Data hazard) in our design. A **Forwarding unit** was implemented to handle the hazards.
- As there are 5 stages in the datapath, the first instruction takes 5 clock cycles to execute. The rest of the instructions require 1 clock cycle each.
- In the current design, when compared with no pipelining, it generally takes longer to execute one single instruction. However, the number of instructions executed in unit time, or in other words, the throughput of the system, increases significantly. This is the improvement that we get in a pipelined system.