CSE 306 (Computer Architecture Sessional)

Experiment No: 04

Name of the experiment:

8-bit MIPS Pipelined Execution

Group No.	**06**
Section	A1
Department	CSE
Group Members	1705026 1705027 1705028 1705029 1705030
Date of Performance:	02-06-2021
Date of Submission:	04-06-2021

Introduction:

The objective of the given assignment was to design 8-bit MIPS Pipelined Execution which implements a subset of multiple MIPS instruction and those can be overlapped in execution. Comparing to Single Cycle implementation, Pipeline implementation is efficient. In Pipeline, all the stage take a single clock cycle that is long enough to accommodate the slowest operation. Each instruction takes 5 clock cycle to be executed. Average time of Pipeline mplementation is less than Single Cycle implementation. In Pipeline, five stages: instruction fetch (IF), instruction decode (ID), execution and address calculation (EX), data memory access (MEM), and write back (WB) are designed. These components are made of registers which store values of registers and control bits. Instruction memory, data memory, register file, ALU, control unit, five pipeline registers, and a forwarding unit were the main components of the processor. Some additional components such as multiplexers, adder, gates were used to design main components of the processor.

Block diagram:

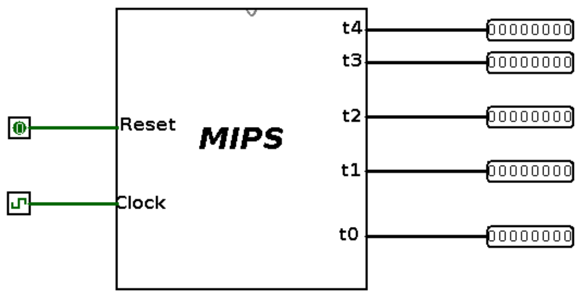

Figure: Block diagram of 8 bit MIPS Pipeline Execution

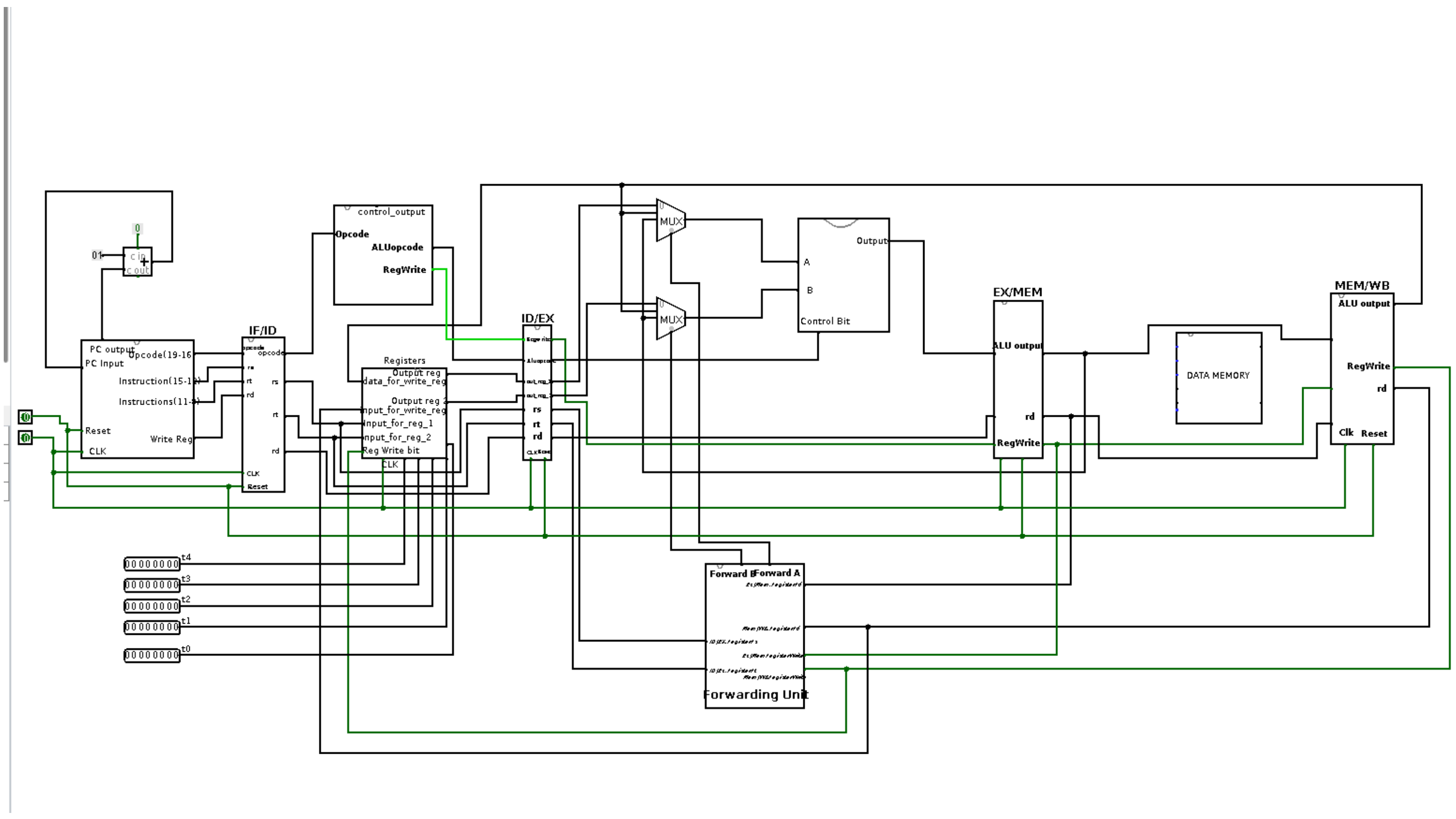

Figure: 8- bit MIPS Pipeline Execution

Block Diagram of Pipeline Registers:

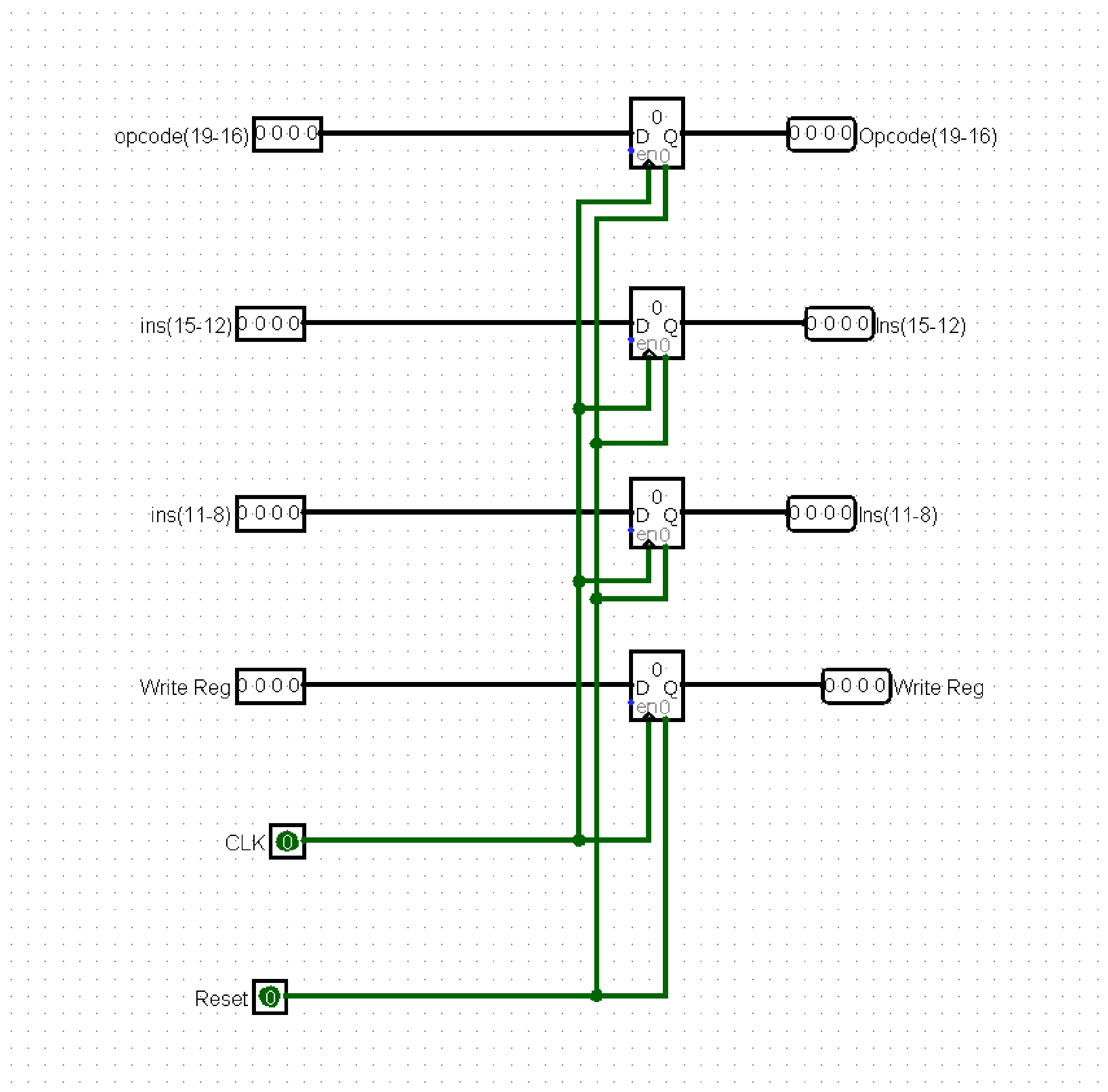

Figure: IF/ID Pipeline Register

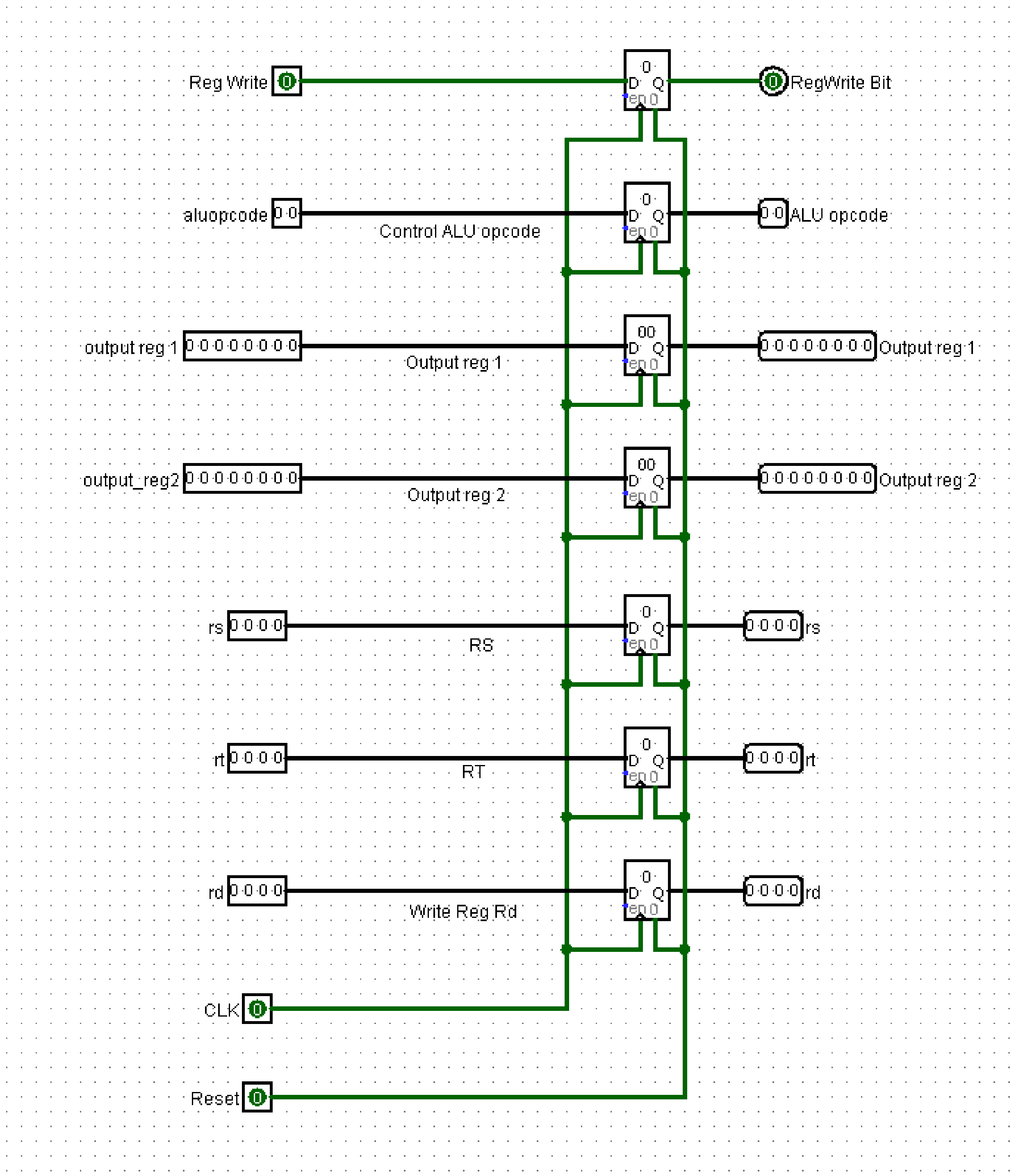

Figure: ID/EX Pipeline Register

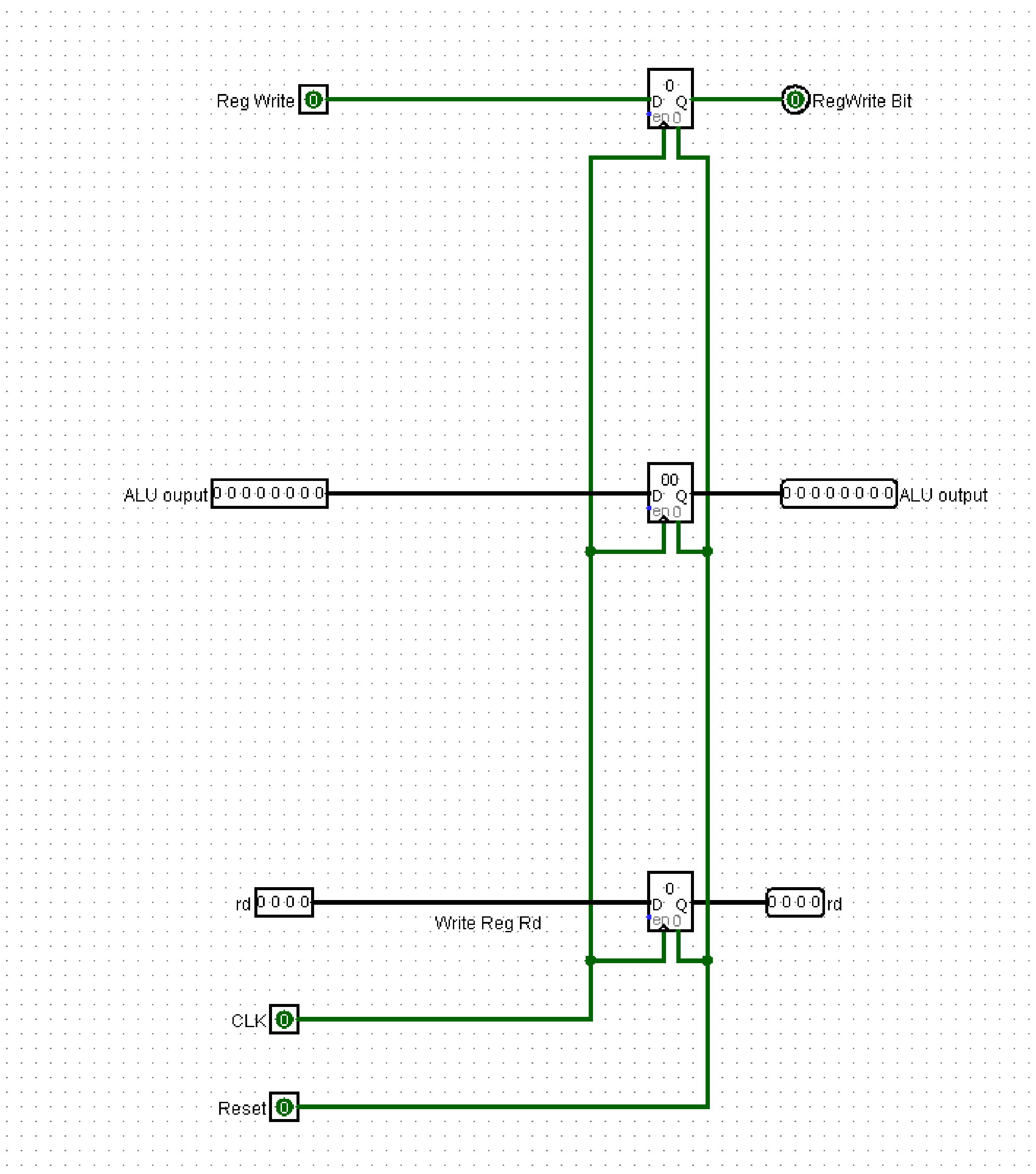

Figure: EX/MEM Pipeline Register

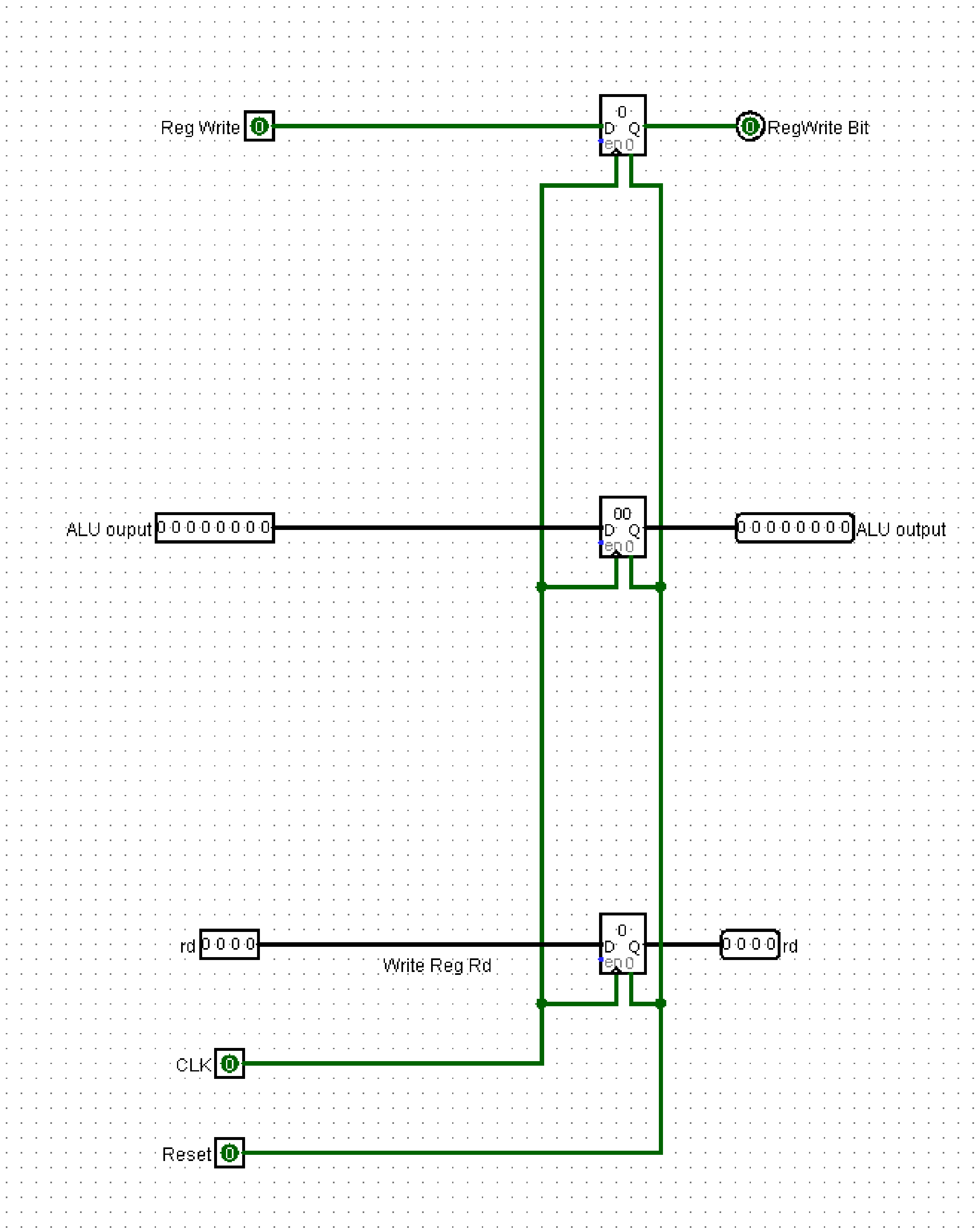

Figure: MEM/WB Pipeline Register

Size of Pipeline Registers:

Pipeline Register	Size (Bit)
IF/ID	16
ID/EX	31
EX/MEM	13
MEM/WB	13

Mechanism and Block Diagram of Forwarding Unit:

A forwarding unit is used to handle three types of data hazards.

1. **EX Hazard:** Example of this type of hazard is

   add \$t1, \$t2, \$t3……(i)

   sub \$t4, \$t1, \$t2……(ii)

Here the rd of instruction (i) in EX/MEM is equal to rs of instruction (ii) in ID/EX. So this is an EX hazard. To deal with this hazard, in the forwarding unit, it is checked if the rs or rt of ID/EX is equal to the rd of EX/MEM. If they are equal and it is not the zero register, also the Register write control bit in the EX/MEM register is 1, then the forwarding unit generates the value 10 so that the value of the 1st input to the ALU is given from the ALU output of EX/MEM rather than the output reg 1 of ID/EX register.

2. **MEM Hazard:** Example of this type of hazard is

   add \$t1, \$t2, \$t3……(i)

   sub \$t4, \$t1, \$t2……(ii)

   or \$t4, \$t1, \$t3……...(iii)

Here the rd of instruction (i) in MEM/WB is equal to rs of instruction (iii) in ID/EX. So this is an MEM hazard. To deal with this hazard, in the forwarding unit, it is checked if the rs or rt of ID/EX is equal to the rd of MEM/WB. If they are equal and it is not the zero register, also the Register write control bit in the MEM/WB register is 1, then the forwarding unit generates the value 01 so that the value of the 1st input to the ALU is given from the ALU output of MEM/WB rather than the output reg 1 of ID/EX register.

3. **Double Data Hazard:** Example of this type of hazard is

add $t1, $t2, $t3…….(i)

sub $t4, $t1, $t2…….(ii)

or $t4, $t1, $t3……...(iii)

add $t4, $t4, $t3……(iv)

Here the rd of instruction (ii) in MEM/WB and rd of instruction (iii) in EX/MEM both are equal to rs of instruction (iv) in ID/EX. Both EX hazard and MEM hazard occur here, so it is a double data hazard. As in this case the latest value of the first input of the ALU is in the ALU output of EX/MEM so in the forwarding unit, MEM hazard was considered only when there is no EX hazard. To deal with this hazard the the forwarding unit generates the value 10 so that the value of the 1st input to the ALU is given from the ALU output of EX/MEM rather than the output reg 1 of ID/EX register.

Discussions:

In this assignment, an 8-bit processor that supports pipelined datapath for the R-format of MIPS instruction set was designed using Logisim. Each instruction was divided into five stages: instruction fetch (IF), instruction decode (ID), execution and address calculation (EX), data memory access (MEM), and write back (WB). The main components of the processor are an instruction memory, a data memory, a register file, an 8 bit ALU, a control unit, four pipeline registers and a forwarding unit for handling hazard. A ROM was used as the instruction memory. A separate ROM was used for control unit and the control signals were mapped against the opcode of the individual instructions. An 8 bit ALU that takes two 8 bit inputs was used. Here, a separate register was kept as PC.

Without pipelining, only one instruction was carried out in a clock cycle. And the length of the clock cycle was the length of the longest instruction among the instruction set. But with pipelining different stages of different instructions were carried out in a single clock cycle. This improved the throughput but introduced data hazards.

To deal with data hazards, a forwarding unit was used which took care of three types of data hazards.

Using minimal number of ICs in order to reduce complexity was the priority while designing the circuit. Outputs of some of the intermediate gates and ICs were reused to minimize the circuit even more.

The connections were made carefully and the components were placed at fair distances. Messiness was avoided as much as possible to ensure higher readability. Suitable labels were used at different parts of the circuit to make the functions of individual parts more understandable. Finally, the circuit was tested several times to make sure it did not have any sort of error.