UM-SJTU Joint Institute

VE370

# Project 2

## Single Cycle and Pipeline MIPS CPU

Group: 45

Name: Tao Yitong ID: 516370910081
Name: Wang Ren ID: 516370910177
Name: Wang Shuhan ID: 516370910228
Date: November 23, 2018

# Objective

Model both single cycle and pipelined implementation of MIPS computer in Verilog that support a subset of MIPS instruction set including:

- The memory-reference instructions load word `lw` and store word`sw`
- The arithmetic-logical instructions `add`, `addi`, `sub`, `and`, `andi`, `or`, and `slt`
- The jumping instructions branch equal `beq`, branch not equal `bne` and jump `j`

# Top Level Block Diagram

## Single Cycle

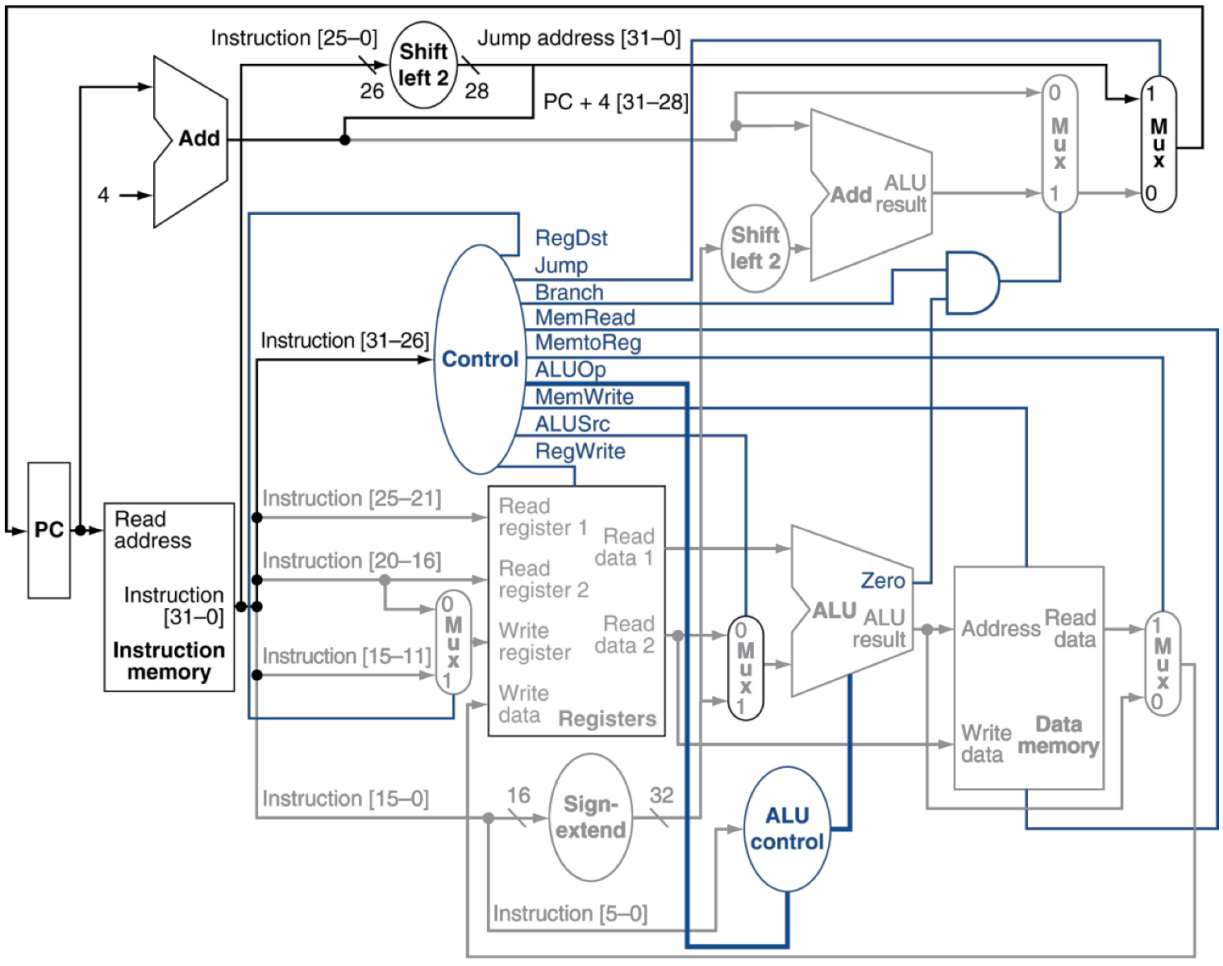


Figure 1:

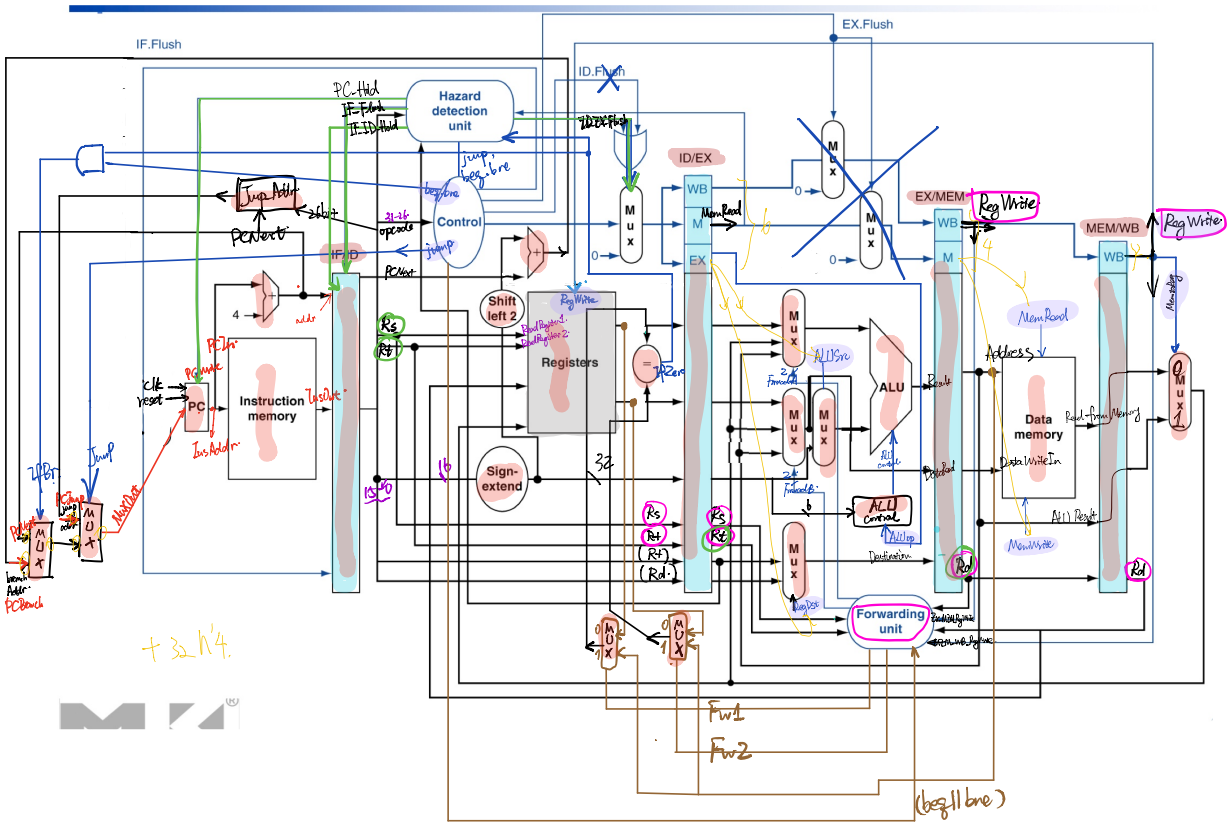
IF.Flush
EX.Flush
ID.Flush
Hazard detection unit
PC_Hold
IF_Flush
IF_ID_Hold
Control
Shift left 2
Registers
Sign-extend
Instruction memory
PC
ID/EX
EX/MEM
MEM/WB
WB
M
EX
ALU
ALU control
Data memory
Forwarding unit
RegWrite
MemRead
Fw1
Fw2
(beq || bne)

Figure 2:

### Pipeline

## Design of Components

We focus on the components for pipeline since the single cycle case is simple.

The Verilog code of each components will be provided in the Appendix.

### IF Stage

This part is to handle the normal +4 PC or PC after branch and draw the corresponding Instruction Memory to IF/ID reg.

We have `mux_pc_in_1`, `mux_pc_in_2`, `PC` and `Instruction_Mem`

### ID Stage

This stage detects all the control hazards, decode control signal and get value from the register file.

We have `Hazard` `Control` `control_out_mux` `Register_File` `Sign_Extended` and `Comparator`

### EX Stage

In this stage we handle data hazard and provide the 'right' value to ALU.

We have 2 `MUX321` 2 `MUX221` `ALU` `ALU_control` `Fowarding_unit`

### MEM Stage

In this stage we read or write the data memory and provide result back to early stages when needed.

We have `Data_Mem`

### WB Stage

In this stage we provided the written back data back to the Register_File.

## Control and Data Hazard

### Ex Stage:

1. Data Hazard

Forwarding Unit:

**The first kind of forwarding:**

There is a forwarding unit in EX stage, taking in Register address inputs from ID/EX, EX/MEM, MEM/WB pipeline registers and control signals to tell whether the register whose content is needed is the destination register of the former instruction in later stage. The way to detect EX hazard(forwarding from EX/MEM to EX):

```
if(((EX_MEM_RegWrite) && (EX_MEM_RegisterRd!=5'b0))
                  && (EX_MEM_RegisterRd==ID_EX_RegisterRs)) ForwardA= 2'b10;

if(((EX_MEM_RegWrite) && (EX_MEM_RegisterRd!=5'b0))
                 && (EX_MEM_RegisterRd==ID_EX_RegisterRt)) ForwardB= 2'b10;
```

The way to detect MEM hazard (and not EX hazard)(forward from MEM/WB to EX):

```
else if((((MEM_WB_RegWrite) && (MEM_WB_RegisterRd!=5'b0))
                  && (MEM_WB_RegisterRd==ID_EX_RegisterRs))&& (!(((EX_MEM_RegWrite)
                  && (EX_MEM_RegisterRd!=5'b0))&& (EX_MEM_RegisterRd==ID_EX_RegisterRs))))
                 ForwardA= 2'b01;

else if((((MEM_WB_RegWrite) && (MEM_WB_RegisterRd!=5'b0))
                 && (MEM_WB_RegisterRd==ID_EX_RegisterRt)) && (!(((EX_MEM_RegWrite)
                 && (EX_MEM_RegisterRd!=5'b0))
                 && (EX_MEM_RegisterRd==ID_EX_RegisterRt))))
                 ForwardB= 2'b01;
```

**The second kind of forwarding:**

Another forwarding condition is when the fording unit taking register addresses from IF/ID and EX/MEM pipeline registers and control signals, to tell whether the output of register read value should be forward with the value calculated in the EX ALU and passed through EX/MEM pipeline register.

This forwarding will happen when the ID instruction is Branch category, like beq/ben, since the fetched register value from register file is used for comparation immediately.

The way to detect hazard when forwarding for beq/bne is needed:

```
if(EX_MEM_RegisterRd!=5'b0 && EX_MEM_RegWrite &&(bne|| beq)
                  && (EX_MEM_RegisterRd==IF_ID_RegisterRs)) Fw1= 1'b1;

if(EX_MEM_RegisterRd!=5'b0 && EX_MEM_RegWrite &(bne||beq)
                   && (EX_MEM_RegisterRd==IF_ID_RegisterRt)) Fw2= 1'b1;
```

Forwarding MUX in EX: Two additional muxs used for the firsts kind of forwarding in EX stage, with the select signals generated in forwarding unit, control the inputs of ALU sources, which allows forwarding from both MEM and WB stages.

There are two more MUXs used for forwarding in ID stage, which will be introduced later.

## ID Stage

1. Control Hazard: Control hazards are experienced when dealing with jumping between instructions. Here we solved beq/bne hazard in ID stage. When a branch hazard is detected, the content of IF/ID pipeline register should be flushed, and the target address is sent to PC at the same time. Control hazard is detected and solved in the hazard detection unit in ID stage.

The way to detect beq/bne/jump hazard:

```
assign IF_Flush=(PC_Hold==1'b0 && (jump||(bne&&IfEqual==1'b0)||(beq&&IfEqual==1'b1)))?1:0;
```

1. Data Hazard: Hazard detection Unit: There are several different conditions when hazard is experienced and need to be solved:

a. The most common Load-Use hazard Load-Use hazard appears when the value of a register is needed but it is not fetched from the memory yet. It can be detected through comparing

```
((ID_EX_MemRead==1'b1)&&((ID_EX_RegisterRt==IF_ID_RegisterRs) ||(ID_EX_RegisterRt==IF_ID_RegisterRt)))
```

When a load-use hazard is detected, several things should be done: Flush the ID stage (flushing Control signals are enough); Hold the PC & Hold the IF/ID register content. In this way , the whole ID stage can be cleared and a NOP instruction is inserted in ID, without changing other stages or causing data lose.

```
assign ID_EX_Flush=PC_Hold;
assign IF_ID_Hold=PC_Hold;
```

b. Load-Use when facing Branch&MEM-Hazard:

Two total stalls are needed if the branch instruction is next to `lw` instruction, only when they are 2 stages away, conflict can be avoided.

```
|| ((beq || bne) && ID_EX_MemRead==1'b1 && ((ID_EX_RegisterRt==IF_ID_RegisterRs)
              ||(ID_EX_RegisterRt==IF_ID_RegisterRt)))
|||((beq ||bne) &&  EX_MEM_MemRead==1'b1 && ((EX_MEM_RegisterRd==IF_ID_RegisterRs)
              ||(EX_MEM_RegisterRd==IF_ID_RegisterRt)))
```

c. branch is next to the instruction needing to write back. Both Rs and Rt should be considered as the possible destination where conflict happens, so ID$EX$RegDst is needed to tell which one should be compared with.

```
 ID_EX_RegWrite==1'b1 && (((ID_EX_RegDst!=5'b0)&&((ID_EX_RegisterRd==IF_ID_RegisterRs)||(ID_EX_Register
```

Forwarding MUX in ID: The MUXs of second kind of forwarding is set in ID stage to select input data to ID/EX register or comparing them to tell whether to branch. Register content from register file and value forwarding from EX/MEM register , which is the result of ALU calculation is selected with the selection signal generated by Forwarding unit in EX stage.

## Instruction Implementation

## SSD and Top Module

The code of the above two will be provided in the Appendix.

## RTL Schematic

## Textual Result

This will be put in the Appendix.

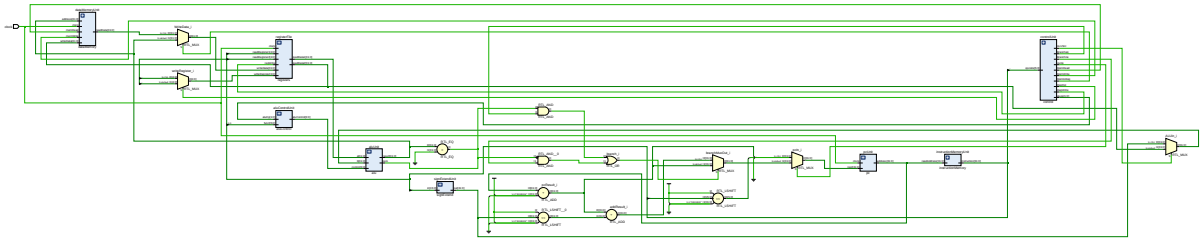

Figure 3:

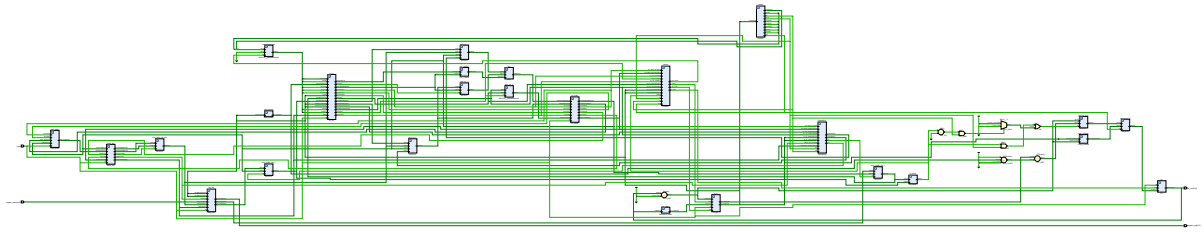

Figure 4:

# Conclusion and Discussion

Pipeline design accelerates nearly 5 times of performance, but at a cost of 10 times of sweat to handle hazard problems.

# Appendix

## Textual Result

### Single Cycle

```
VCD info: dumpfile test.lxt opened for output.
Time:           0, clock = 0, PC = 0x00000000
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
----------------------------------------------------------
Time:           1, clock = 1, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
```

```
------------------------------------------------------------
Time:           1, clock = 0, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           2, clock = 1, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           2, clock = 0, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           3, clock = 1, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           3, clock = 0, PC = 0x0000000c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           4, clock = 1, PC = 0x00000010
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           4, clock = 0, PC = 0x00000010
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
```

```
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           5, clock = 1, PC = 0x00000014
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           5, clock = 0, PC = 0x00000014
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           6, clock = 1, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           6, clock = 0, PC = 0x00000018
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           7, clock = 1, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           7, clock = 0, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           8, clock = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
```

```
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           8, clock = 0, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           9, clock = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:           9, clock = 0, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          10, clock = 1, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          10, clock = 0, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          11, clock = 1, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          11, clock = 0, PC = 0x0000002c
```

```
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          12, clock = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          12, clock = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          13, clock = 1, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          13, clock = 0, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          14, clock = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          14, clock = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
```

```
------------------------------------------------------------
Time:          15, clock = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          15, clock = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          16, clock = 1, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          16, clock = 0, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          17, clock = 1, PC = 0x00000044
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          17, clock = 0, PC = 0x00000044
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          18, clock = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
```

```
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          18, clock = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          19, clock = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          19, clock = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          20, clock = 1, PC = 0x0000007c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          20, clock = 0, PC = 0x0000007c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          21, clock = 1, PC = 0x00000080
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          21, clock = 0, PC = 0x00000080
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
```

```
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          22, clock = 1, PC = 0x00000084
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          22, clock = 0, PC = 0x00000084
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          23, clock = 1, PC = 0x00000088
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          23, clock = 0, PC = 0x00000088
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          24, clock = 1, PC = 0x0000008c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
Time:          24, clock = 0, PC = 0x0000008c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000
------------------------------------------------------------
** VVP Stop(0) **
```

```
** Flushing output streams.
** Current simulation time is 500 ticks.
> finish
** Continue **
```

**Pipeline**

```
close_sim
INFO: [Simtcl 6-16] Simulation closed
launch_simulation
INFO: [Vivado 12-5682] Launching behavioral simulation in 'D:/VE270/Projects/1456/1456.sim/sim_1/behav/
INFO: [SIM-utils-51] Simulation object is 'sim_1'
INFO: [SIM-utils-54] Inspecting design source files for 'pipeline_tb' in fileset 'sim_1'...
INFO: [USF-XSim-97] Finding global include files...
INFO: [USF-XSim-98] Fetching design files from 'sim_1'...
INFO: [USF-XSim-2] XSim::Compile design
INFO: [USF-XSim-61] Executing 'COMPILE and ANALYZE' step in 'D:/VE270/Projects/1456/1456.sim/sim_1/beha
"xvlog --incr --relax -prj pipeline_tb_vlog.prj"
INFO: [USF-XSim-69] 'compile' step finished in '3' seconds
INFO: [USF-XSim-3] XSim::Elaborate design
INFO: [USF-XSim-61] Executing 'ELABORATE' step in 'D:/VE270/Projects/1456/1456.sim/sim_1/behav/xsim'
Vivado Simulator 2018.1
Copyright 1986-1999, 2001-2017 Xilinx, Inc. All Rights Reserved.
Running: D:/VE270/Vivado/2018.1/bin/unwrapped/win64.o/xelab.exe -wto cd926085497343a88d05fbc02a87146d --
Using 2 slave threads.
Starting static elaboration
Completed static elaboration
INFO: [XSIM 43-4323] No Change in HDL. Linking previously generated obj files to create kernel
INFO: [USF-XSim-69] 'elaborate' step finished in '3' seconds
INFO: [USF-XSim-4] XSim::Simulate design
INFO: [USF-XSim-61] Executing 'SIMULATE' step in 'D:/VE270/Projects/1456/1456.sim/sim_1/behav/xsim'
INFO: [USF-XSim-98] *** Running xsim
   with args "pipeline_tb_behav -key {Behavioral:sim_1:Functional:pipeline_tb} -tclbatch {pipeline_tb.t
INFO: [USF-XSim-8] Loading simulator feature
Vivado Simulator 2018.1
Time resolution is 1 ps
source pipeline_tb.tcl
# set curr_wave [current_wave_config]
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave w
#   }
# }
# run 1000ns
Texual result of pipeline:
============================================================
Time:           0, CLK = 0, PC = 0x00000000
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
```

```
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           0, CLK = 1, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           1, CLK = 0, PC = 0x00000004
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           1, CLK = 1, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           2, CLK = 0, PC = 0x00000008
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           2, CLK = 1, PC = 0x0000000c
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           3, CLK = 0, PC = 0x0000000c
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           3, CLK = 1, PC = 0x00000010
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
```

```
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000000
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           4, CLK = 0, PC = 0x00000010
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           4, CLK = 1, PC = 0x00000014
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000000, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           5, CLK = 0, PC = 0x00000014
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           5, CLK = 1, PC = 0x00000018
[$s0] = 0x00000000, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           6, CLK = 0, PC = 0x00000018
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           6, CLK = 1, PC = 0x0000001c
[$s0] = 0x00000020, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
```

```
Time:           7, CLK = 0, PC = 0x0000001c
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           7, CLK = 1, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           8, CLK = 0, PC = 0x00000020
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           8, CLK = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           9, CLK = 0, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:           9, CLK = 1, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000000, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          10, CLK = 0, PC = 0x00000024
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
```

```
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          10, CLK = 1, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          11, CLK = 0, PC = 0x00000028
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          11, CLK = 1, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          12, CLK = 0, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          12, CLK = 1, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          13, CLK = 0, PC = 0x0000002c
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          13, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000057, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          14, CLK = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          14, CLK = 1, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          15, CLK = 0, PC = 0x00000030
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          15, CLK = 1, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0xfffffff9
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          16, CLK = 0, PC = 0x00000034
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          16, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          17, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
```

```
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          17, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          18, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          18, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000000, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          19, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          19, CLK = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          20, CLK = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
```

```
Time:          20, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          21, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          21, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          22, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          22, CLK = 1, PC = 0x00000044
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          23, CLK = 0, PC = 0x00000044
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          23, CLK = 1, PC = 0x00000048
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
```

```
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          24, CLK = 0, PC = 0x00000048
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          24, CLK = 1, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          25, CLK = 0, PC = 0x00000038
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          25, CLK = 1, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000000
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          26, CLK = 0, PC = 0x0000003c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          26, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          27, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
```

```
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          27, CLK = 1, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          28, CLK = 0, PC = 0x00000040
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          28, CLK = 1, PC = 0x0000007c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000001, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          29, CLK = 0, PC = 0x0000007c
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          29, CLK = 1, PC = 0x00000080
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          30, CLK = 0, PC = 0x00000080
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
Time:          30, CLK = 1, PC = 0x00000084
[$s0] = 0x00000037, [$s1] = 0x00000037, [$s2] = 0x00000037
```

```
[$s3] = 0x00000020, [$s4] = 0x00000000, [$s5] = 0x00000000
[$s6] = 0x00000000, [$s7] = 0x00000000, [$t0] = 0x00000020
[$t1] = 0x00000037, [$t2] = 0x00000000, [$t3] = 0x00000000
[$t4] = 0x00000000, [$t5] = 0x00000000, [$t6] = 0x00000000
[$t7] = 0x00000000, [$t8] = 0x00000000, [$t9] = 0x00000000
================================================
INFO: [USF-XSim-96] XSim completed. Design snapshot 'pipeline_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:07 ; elapsed = 00:00:09 . Memory (MB): peak = 1911.074 ; gain =
```

## Verilog Code

### ALU.v

```
`timescale 1ns / 1ps

module ALU( input [3:0] control,
    input [31:0] a, b,
    output reg [31:0] result);

    always @(control, a, b) begin
        case (control)
            4'b0000: result = a & b;
            4'b0001: result = a | b;
            4'b0010: result = a + b;
            4'b0110: result = a - b;
            4'b0111: result = (a < b) ? 1 : 0;
            4'b1100: result = ~(a | b);
            default: result = 32'b0;
        endcase
    end

endmodule
```

### ALU_Control.v

```
`timescale 1ns / 1ps


module ALU_Control(input [5:0] funct,
               input [1:0] ALUop,
               output reg [3:0] control );

               always @(funct, ALUop) begin
               case (ALUop)
               2'b00: control=4'b0010; //lw/sw(add)
               2'b01: control=4'b0110;//branch(subtrsct)
               2'b10: case(funct)
                      6'b100000: control=4'b0010;//add
                      6'b100010: control=4'b0110;//subtract
                      6'b100100: control=4'b0000;//AND
                      6'b100101: control=4'b0001;//OR
```

```
                    6'b101010: control=4'b0111;//slt
                    default: control=4'b0000;
                    endcase
                2'b11: control=4'b0000;
               default:control=4'b0000;
               endcase
               end


endmodule
```

### Branch_Address.v

```
`timescale 1ns / 1ps

module Branch_Address(

    );
endmodule
```

### ClockDivider.v

```
`timescale 1ns / 1ps

module ClockDivider(
    input clock,
    output clk500Hz
);
reg[27:0] counter = 28'd0;
parameter DIVISOR = 28'd200000;
always @(posedge clock) begin
    counter <= counter + 28'd1;
    if(counter >= (DIVISOR-1))
        counter <= 28'd0;
end
assign clk500Hz = (counter<DIVISOR/2)?1'b0:1'b1;

endmodule // ClockDivider
```

### Control.v

```
`timescale 1ns / 1ps

module Control(input [5:0] opcode,
               output reg
               jump,
               Branch,
               Bne,
               MemRead,
               MemtoReg ,
               MemWrite,
               ALUSrc,
```

```
        RegWrite,
        RegDst
        ,output reg [1:0] ALUOp);

            initial begin
                RegDst      = 1'b0;
                jump        = 1'b0;
                Branch    = 1'b0;
                Bne    = 1'b0;
                MemRead     = 1'b0;
                MemtoReg    = 1'b0;
                MemWrite    = 1'b0;
                ALUSrc      = 1'b0;
                RegWrite    = 1'b0;
                ALUOp       = 2'b00;
            end

    always @(opcode)begin
        case(opcode)
        6'b000000://add, sub, and, or, slt
        begin
        RegDst= 1'b1;
        Branch= 1'b0;
        Bne=1'b0;
        MemRead= 1'b0;
        MemtoReg= 1'b1;
        MemWrite= 1'b0;
        ALUSrc= 1'b0;
        RegWrite= 1'b1;
        jump= 1'b0;
        ALUOp= 2'b10;
        end
        6'b001000://addi
        begin
        RegDst= 1'b0;
        Branch= 1'b0;
        Bne=1'b0;
        MemRead= 1'b0;
        MemtoReg= 1'b1;
        MemWrite= 1'b0;
        ALUSrc= 1'b1;
        RegWrite= 1'b1;
        jump= 1'b0;
        ALUOp= 2'b00;
        end
         6'b001100://andi
        begin
        RegDst= 1'b0;
        Branch= 1'b0;
        Bne=1'b0;
        MemRead= 1'b0;
        MemtoReg= 1'b1;
        MemWrite= 1'b0;
        ALUSrc= 1'b1;
```

```
RegWrite= 1'b1;
jump= 1'b0;
ALUOp= 2'b11;
end
6'b000100:// beq
begin
RegDst= 1'b1;
Branch= 1'b1;
Bne=1'b0;
MemRead= 1'b0;
MemtoReg= 1'b0;
MemWrite= 1'b0;
ALUSrc= 1'b0;
RegWrite= 1'b0;
jump= 1'b0;
ALUOp= 2'b01;
end
6'b000101:// bne
begin
RegDst= 1'b1;
Branch= 1'b0;
Bne=1'b1;
MemRead= 1'b0;
MemtoReg= 1'b0;
MemWrite= 1'b0;
ALUSrc= 1'b0;
RegWrite= 1'b0;
jump= 1'b0;
ALUOp= 2'b01;
end
6'b100011://lw
begin
RegDst= 1'b0;
Branch= 1'b0;
Bne=1'b0;
MemRead= 1'b1;
MemtoReg= 1'b0;
MemWrite= 1'b0;
ALUSrc= 1'b1;
RegWrite= 1'b1;
jump= 1'b0;
ALUOp= 2'b00;
end
6'b101011://sw
begin
RegDst= 1'b0;
Branch= 1'b0;
Bne=1'b0;
MemRead= 1'b0;
MemtoReg= 1'b0;
MemWrite= 1'b1;
ALUSrc= 1'b1;
RegWrite= 1'b0;
jump= 1'b0;
```

```
                ALUOp= 2'b00;
                end
                6'b000010://jump
                begin
                RegDst= 1'b0;
                Branch= 1'b0;
                Bne=1'b0;
                MemRead= 1'b0;
                MemtoReg= 1'b0;
                MemWrite= 1'b0;
                ALUSrc= 1'b0;
                RegWrite= 1'b0;
                jump= 1'b1;
                ALUOp= 2'b00;
                end
                default:
                begin
                RegDst= 1'b0;
                Branch= 1'b0;
                Bne=1'b0;
                MemRead= 1'b0;
                MemtoReg= 1'b0;
                MemWrite= 1'b0;
                ALUSrc= 1'b0;
                RegWrite= 1'b0;
                jump= 1'b0;
                ALUOp= 2'b00;
                end
                endcase
               end

endmodule
```

**Data_Mem.v**

```
`timescale 1ns / 1ps

module Data_Mem(input [31:0] address,
                input [31:0] Write_data,
                input MemWrite, MemRead,clk,
                output [31:0]Read_data);
         reg [31:0]Data[0:31];
         wire [31:0]a;
         assign a=address >> 2;
         integer i;

initial begin
             for (i = 0; i < 32; i=i+1) begin
                 Data[i] = 32'b0;
             end
         end

         always @(posedge clk) begin
```

```
                if(MemWrite==1'b1) Data[a]=Write_data;
                else Data[a]=Data[a];
                end

                assign Read_data=(MemRead==1'b1)? Data[a]:32'b0;

endmodule
```

**EX_MEM.v**

```
`timescale 1ns / 1ps

module EX_MEM(input clk,
              input MemRead,MemtoReg,MemWrite
              ,RegWrite,
              input [31:0] ALUResultAddr,DataWriteIn,
              input[4:0] RegisterDst,

              output reg MemReadM,MemtoRegM,MemWriteM
              ,RegWriteM,
              output reg [31:0] ALUResultAddrM,DataWriteInM,
              output reg [4:0] RegisterDstM);

              initial begin
              MemReadM<=1'b0;
              MemtoRegM<=1'b0;
              MemWriteM<=1'b0;
              RegWriteM<=1'b0;
              ALUResultAddrM<=32'b00000000000000000000000000000000;
              DataWriteInM<=32'b00000000000000000000000000000000;
              RegisterDstM<=5'b00000;
              end

              always @(posedge clk) begin
                  MemReadM<=MemRead;
                  MemtoRegM<=MemtoReg;
                  MemWriteM<=MemWrite;
                  RegWriteM<=RegWrite;
                  ALUResultAddrM<=ALUResultAddr;
                  DataWriteInM=DataWriteIn;
                  RegisterDstM<=RegisterDst;
 //                 end
                  end

endmodule
```

**Forwarding.v**

```
`timescale 1ns / 1ps

module Forwarding(input  MEM_WB_RegWrite ,EX_MEM_RegWrite,bne,beq,

                  input [4:0] MEM_WB_RegisterRd,ID_EX_RegisterRs
```

```
                ,EX_MEM_RegisterRd,ID_EX_RegisterRt
                ,IF_ID_RegisterRs,IF_ID_RegisterRt,
  output reg [1:0] ForwardA,ForwardB,
  output reg  Fw1,Fw2
  );

initial begin
ForwardA=2'b00;
ForwardB=2'b00;
Fw1=1'b0;
Fw2=1'b0;
end

  always @(*)
  begin
  if(((EX_MEM_RegWrite) && (EX_MEM_RegisterRd!=5'b0))
   && (EX_MEM_RegisterRd==ID_EX_RegisterRs)) ForwardA= 2'b10;
  else if((((MEM_WB_RegWrite) && (MEM_WB_RegisterRd!=5'b0))
   && (MEM_WB_RegisterRd==ID_EX_RegisterRs))&& (!(((EX_MEM_RegWrite)
   && (EX_MEM_RegisterRd!=5'b0))&& (EX_MEM_RegisterRd==ID_EX_RegisterRs))))
  ForwardA= 2'b01;
  else ForwardA<= 2'b00;
  end

  always @(*)
  begin
  if(((EX_MEM_RegWrite) && (EX_MEM_RegisterRd!=5'b0))
  && (EX_MEM_RegisterRd==ID_EX_RegisterRt)) ForwardB= 2'b10;
  else if((((MEM_WB_RegWrite) && (MEM_WB_RegisterRd!=5'b0))
  && (MEM_WB_RegisterRd==ID_EX_RegisterRt)) && (!(((EX_MEM_RegWrite)
  && (EX_MEM_RegisterRd!=5'b0))
  && (EX_MEM_RegisterRd==ID_EX_RegisterRt))))
  ForwardB= 2'b01;
  else ForwardB<= 2'b00;
  end

   always @(*)
   begin
   if(EX_MEM_RegisterRd!=5'b0 && EX_MEM_RegWrite &&(bne|| beq)
   && (EX_MEM_RegisterRd==IF_ID_RegisterRs)) Fw1= 1'b1;
   else Fw1=1'b0;
   end

    always @(*)
    begin
    if(EX_MEM_RegisterRd!=5'b0 && EX_MEM_RegWrite &(bne||beq)
    && (EX_MEM_RegisterRd==IF_ID_RegisterRt)) Fw2= 1'b1;
    else Fw2=1'b0;
    end


endmodule
```

**Hazard.v**

```
`timescale 1ns / 1ps

module Hazard( input ID_EX_MemRead,
                EX_MEM_MemRead,
                ID_EX_RegWrite,
                ID_EX_RegDst,
                jump,
                bne,
                beq,
                IfEqual,
               input [4:0] ID_EX_RegisterRt,
               ID_EX_RegisterRd,
                IF_ID_RegisterRs,
                IF_ID_RegisterRt,
                EX_MEM_RegisterRd,
               output PC_Hold,
               IF_ID_Hold,
               ID_EX_Flush,
               IF_Flush
);

     assign PC_Hold=(
               ((ID_EX_MemRead==1'b1)&&((ID_EX_RegisterRt==IF_ID_RegisterRs)
              ||(ID_EX_RegisterRt==IF_ID_RegisterRt)))
              || ((beq || bne) && ID_EX_MemRead==1'b1 && ((ID_EX_RegisterRt==IF_ID_RegisterRs)
              ||(ID_EX_RegisterRt==IF_ID_RegisterRt)))
              ||((beq ||bne) &&  EX_MEM_MemRead==1'b1 && ((EX_MEM_RegisterRd==IF_ID_RegisterRs)
              ||(EX_MEM_RegisterRd==IF_ID_RegisterRt)))
              ||((beq || bne)&& ID_EX_RegWrite==1'b1 &&
              (((ID_EX_RegDst!=5'b0)&&((ID_EX_RegisterRd==IF_ID_RegisterRs)||(ID_EX_RegisterRd==IF_ID_R
              ||((ID_EX_RegDst==5'b0)&&((ID_EX_RegisterRt==IF_ID_RegisterRs)||(ID_EX_RegisterRt==IF_ID_
               )? 1:0 ;

                              assign ID_EX_Flush=PC_Hold;
                              assign IF_ID_Hold=PC_Hold;
                              assign IF_Flush=(PC_Hold==1'b0 && (jump||(bne&&IfEqual==1'b0)||(beq&&IfEq
endmodule
```

**ID_EX.v**

```
`timescale 1ns / 1ps

module ID_EX(input clk,
             input RegDst,MemRead
             ,MemtoReg,MemWrite,ALUSrc
             ,RegWrite,
             input [1:0] ALUOp,
             input [31:0] ReadData1,ReadData2,ExtendedIm,
             input[4:0] ReadRegister1, ReadRegister2
             ,Rt,Rd,
```

```
            output reg RegDstEX,MemReadEX
            ,MemtoRegEX,MemWriteEX,ALUSrcEX
            ,RegWriteEX,
            output reg [1:0] ALUOpEX,
            output reg [31:0] ReadData1EX,ReadData2EX
            ,ExtendedImEX,
            output reg[4:0] ReadRegister1EX
            ,ReadRegister2EX,RtEX,RdEX
            );

            initial begin
            RegDstEX<=1'b0;
            MemReadEX<=1'b0;
            MemtoRegEX<=1'b0;
            MemWriteEX<=1'b0;
            ALUSrcEX<=1'b0;
            RegWriteEX<=1'b0;
            ALUOpEX<=2'b00;
            ReadData1EX<=32'b00000000000000000000000000000000;
            ReadData2EX<=32'b00000000000000000000000000000000;
            ExtendedImEX<=32'b00000000000000000000000000000000;
            ReadRegister1EX<=5'b00000;
            ReadRegister2EX<=5'b00000;
            RtEX<=5'b00000;
            RdEX<=5'b00000;
            end
always @(posedge clk) begin
                RegDstEX<=RegDst;
                MemReadEX<=MemRead;
                MemtoRegEX<=MemtoReg;
                MemWriteEX<=MemWrite;
                ALUSrcEX<=ALUSrc;
                RegWriteEX<=RegWrite;
                ALUOpEX<=ALUOp;
                ReadData1EX<=ReadData1;
                ReadData2EX<=ReadData2;
                ExtendedImEX<=ExtendedIm;
                ReadRegister1EX<=ReadRegister1;
                ReadRegister2EX<=ReadRegister2;
                RtEX<=Rt;
                RdEX<=Rd;
                end
endmodule
```

**IF_ID.v**

```
`timescale 1ns / 1ps

module IF_ID(input clk,hold,Flush,
             input [31:0] Instr, Addr,
             output reg [31:0] out_Instr, out_Addr);
             always @(posedge clk) begin
            if(Flush==1'b1)
```

```
            begin
            out_Instr<=32'b0;
            out_Addr<=32'b0;
            end
            else if(hold==1'b1)begin
            out_Instr<=out_Instr;
            out_Addr<=out_Addr;
            end
            else begin
            out_Instr<=Instr;
            out_Addr<=Addr;
            end
            end
            initial begin
            out_Instr <= 32'b0;
            out_Addr <= 32'b0;
            end

endmodule
```

### If_Equal.v

```
`timescale 1ns / 1ps

module If_Equal(input [31:0] a,b,
                output wire IfEqual);

                 assign IfEqual = (a == b) ? 1'b1 : 1'b0;
endmodule
```

### Instruction_Mem.v

```
`timescale 1ns / 1ps

module Instruction_Mem(input [31:0] addr,
                       output wire [31:0]out_Instr);

wire [31:0] memory[0:31];
wire [31:0] A;
assign A=addr>>2;
assign out_Instr = memory[A];

//demo
assign memory[0] = 32'b00100000000010000000000000100000; //addi $t0, $zero, 0x20
assign memory[1] = 32'b00100000000010010000000000110111; //addi $t1, $zero, 0x37
assign memory[2] = 32'b00000001000010011000000000100100; //and $s0, $t0, $t1
assign memory[3] = 32'b00000001000010011000000000100101; //or $s0, $t0, $t1
assign memory[4] = 32'b10101100000010000000000000000100; //sw $s0, 4($zero)
assign memory[5] = 32'b10101100000010000000000000001000; //sw $t0, 8($zero)
assign memory[6] = 32'b00000001000010011000100000100000; //add $s1, $t0, $t1
assign memory[7] = 32'b00000001000010011001000000100010; //sub $s2, $t0, $t1
assign memory[8] = 32'b00010010001100100000000000001001; //beq $s1, $s2, error0
assign memory[9] = 32'b10001100000010001000000000000100; //lw $s1, 4($zero)
```

```
assign memory[10]= 32'b00110010001100100000000001001000; //andi $s2, $s1, 0x48
assign memory[11] =32'b00010010001100100000000000001001; //beq $s1, $s2, error1
assign memory[12] =32'b10001100000100110000000000001000; //lw $s3, 8($zero)
assign memory[13] =32'b00010010000100110000000000001010; //beq $s0, $s3, error2
assign memory[14] =32'b00000001001010001101000000101010; //slt $s4, $s2, $s1 (Last)
assign memory[15] =32'b00010010100000000000000000001111; //beq $s4, $0, EXIT
assign memory[16] =32'b00000010001000001001000000100000; //add $s2, $s1, $0
assign memory[17] =32'b00001000000000000000000000001110; //j Last
assign memory[18] =32'b00100000000010000000000000000000; //addi $t0, $0, 0(error0)
assign memory[19] =32'b00100000000010010000000000000000; //addi $t1, $0, 0
assign memory[20] =32'b00001000000000000000000000011111; //j EXIT
assign memory[21] =32'b00100000000010000000000000000001; //addi $t0, $0, 1(error1)
assign memory[22] =32'b00100000000010010000000000000001; //addi $t1, $0, 1
assign memory[23] =32'b00001000000000000000000000011111; //j EXIT
assign memory[24] =32'b00100000000010000000000000000010; //addi $t0, $0, 2(error2)
assign memory[25] =32'b00100000000010010000000000000010; //addi $t1, $0, 2
assign memory[26] =32'b00001000000000000000000000011111; //j EXIT
assign memory[27] =32'b00100000000010000000000000000011; //addi $t0, $0, 3(error3)
assign memory[28] =32'b00100000000010010000000000000011; //addi $t1, $0, 3
assign memory[29] =32'b00001000000000000000000000011111; //j EXIT

endmodule
```

### Jump_Address.v

```
`timescale 1ns / 1ps

module Jump_Address(input [25:0]JumpWhere26,
                    input [3:0] PCNext_4,
                    output [31:0] JumpAddress );
  assign JumpAddress={PCNext_4[3:0],JumpWhere26[25:0],2'b0};

endmodule
```

### MEM_WB.v

```
`timescale 1ns / 1ps

module MEM_WB(input clk,
              input MemtoReg,RegWrite,
              input [31:0] Data2Write, ALUResult,
              input [4:0] RegisterDst,

              output reg MemtoRegWB,RegWriteWB,
              output reg [31:0] Data2WriteWB, ALUResultWB,
              output reg [4:0] RegisterDstWB);

              initial begin
              MemtoRegWB<=1'b0;
              RegWriteWB<=1'b0;
              Data2WriteWB<=32'b00000000000000000000000000000000;
              ALUResultWB<=32'b00000000000000000000000000000000;
              RegisterDstWB<=5'b00000;
```

```
            end

             always @(posedge clk) begin
            MemtoRegWB<=MemtoReg;
            RegWriteWB<=RegWrite;
            Data2WriteWB<=Data2Write;
            ALUResultWB<=ALUResult;
            RegisterDstWB<=RegisterDst;
                end
endmodule
```

**MUX221.v**

```
`timescale 1ns / 1ps

module MUX221
            #(parameter bits=32)
             (  input sel,
                input [bits-1:0] a, b,
                output reg [bits-1:0] out );

    always @(sel, a, b) begin
        case (sel)
            1'b0: out = a;
            1'b1: out = b;
            default: out = 0;
        endcase
    end

endmodule
```

**MUX321.v**

```
`timescale 1ns / 1ps

module MUX321
            #(parameter bits=16)
            (input [1:0] sel,
             input [bits-1:0] a, b, c,
             output reg [bits-1:0] out );
            always @(sel, a, b, c) begin
                    case (sel)
                        2'b00: out = a;
                        2'b01: out = b;
                        2'b10: out =c;
                        default: out = 0;
                    endcase
                end
endmodule
```

**PC.v**

```
`timescale 1ns / 1ps

module PC( input clk,hold,
           input [31:0] next,
           output reg [31:0] address);
        initial begin
           address<=32'b0;
           end
    always @(posedge clk) begin
    if(hold==1'b1)  address<=address;
    else address <= next;
        end


endmodule
```

**Reg__File.v**

```
`timescale 1ns / 1ps

module Reg_File(input clk ,RegWrite,
              input [4:0] ReadRegister1, ReadRegister2,WriteReg , Reg_switch,
              input [31:0] WriteData,
              output wire [31:0] read_data1, read_data2, Reg_Read
);
        reg[31:0] registers[0:31];
    integer i;
        assign read_data1 = registers[ReadRegister1];
        assign read_data2 = registers[ReadRegister2];

    initial begin
            for (i = 0; i < 32; i=i+1) begin
                registers[i] = 32'b0;
            end
        end

        always@(negedge clk)
        begin
            if(RegWrite)
                registers[WriteReg] = WriteData;
        end
        assign Reg_Read=registers[Reg_switch];

endmodule
```

**Reg *Value*Fetch.v**

```
`timescale 1ns / 1ps

module Reg_Value_Fetch(
```

```
    input [4:0] switch_register,
    output [31:0] Read_Reg
    );

endmodule
```

**RingCounter.v**

```
`timescale 1ns / 1ps

module RingCounter(anode,clk_500);
    output [3:0] anode;
    input clk_500;

    reg [3:0] anode=4'b0111;

    always @(posedge clk_500)begin
 begin
        if (anode==4'b0111) anode<=4'b1011;
        else if (anode==4'b1011) anode<=4'b1101;
        else if (anode==4'b1101) anode<=4'b1110;
        else if (anode==4'b1110) anode<=4'b0111;
        else anode<=4'b0111;
        end
    end

endmodule
```

**SSD.v**

```
`timescale 1ns / 1ps

module SSD(
    input [31:0] PC_value,reg_value,
    input clock,reg_0_pc_1,
    output [3:0]an,
    output reg [6:0]ca
);
wire clock500Hz;
reg [4:0]in_an_1,in_an_2,in_an_3,in_an_4;
reg [6:0]Ca1,Ca2,Ca3,Ca4;
ClockDivider ClkDivider500Hz(clock,clock500Hz);
RingCounter RingShow(an,clock500Hz);
always@(*)begin
if(reg_0_pc_1==1'b0)begin
in_an_1<=reg_value[3:0];
in_an_2<=reg_value[7:4];
in_an_3<=reg_value[11:8];
in_an_4<=reg_value[15:12];
end
else
begin
in_an_1<=PC_value[3:0];
```

```
in_an_2<=PC_value[7:4];
in_an_3<=PC_value[11:8];
in_an_4<=PC_value[15:12];
end
end
          always @ (in_an_1) begin
    case (in_an_1)
          4'b0000: Ca1=7'b0000001;
            4'b0001: Ca1=7'b1001111;
            4'b0010: Ca1=7'b0010010;
            4'b0011: Ca1=7'b0000110;
            4'b0100: Ca1=7'b1001100;
            4'b0101: Ca1=7'b0100100;
            4'b0110: Ca1=7'b0100000;
            4'b0111: Ca1=7'b0001111;
            4'b1000: Ca1=7'b0000000;
            4'b1001: Ca1=7'b0000100;
            4'b1010: Ca1=7'b0001000;
            4'b1011: Ca1=7'b1100000;
            4'b1100: Ca1=7'b0110001;
            4'b1101: Ca1=7'b1000010;
            4'b1110: Ca1=7'b0110000;
            4'b1111: Ca1=7'b0111000;
            default Ca1=7'b1111111;
     endcase
end

always @ (in_an_2) begin
    case (in_an_2)
          4'b0000: Ca2=7'b0000001;
            4'b0001: Ca2=7'b1001111;
            4'b0010: Ca2=7'b0010010;
            4'b0011: Ca2=7'b0000110;
            4'b0100: Ca2=7'b1001100;
            4'b0101: Ca2=7'b0100100;
            4'b0110: Ca2=7'b0100000;
            4'b0111: Ca2=7'b0001111;
            4'b1000: Ca2=7'b0000000;
            4'b1001: Ca2=7'b0000100;
            4'b1010: Ca2=7'b0001000;
            4'b1011: Ca2=7'b1100000;
            4'b1100: Ca2=7'b0110001;
            4'b1101: Ca2=7'b1000010;
            4'b1110: Ca2=7'b0110000;
            4'b1111: Ca2=7'b0111000;
            default Ca2=7'b1111111;
     endcase
end

always @ (in_an_3) begin
    case (in_an_3)
          4'b0000: Ca3=7'b0000001;
            4'b0001: Ca3=7'b1001111;
            4'b0010: Ca3=7'b0010010;
```

```
            4'b0011: Ca3=7'b0000110;
            4'b0100: Ca3=7'b1001100;
            4'b0101: Ca3=7'b0100100;
            4'b0110: Ca3=7'b0100000;
            4'b0111: Ca3=7'b0001111;
            4'b1000: Ca3=7'b0000000;
            4'b1001: Ca3=7'b0000100;
            4'b1010: Ca3=7'b0001000;
            4'b1011: Ca3=7'b1100000;
            4'b1100: Ca3=7'b0110001;
            4'b1101: Ca3=7'b1000010;
            4'b1110: Ca3=7'b0110000;
            4'b1111: Ca3=7'b0111000;
            default Ca3=7'b1111111;
     endcase
end

always @ (in_an_4) begin
    case (in_an_4)
         4'b0000: Ca4=7'b0000001;
           4'b0001: Ca4=7'b1001111;
           4'b0010: Ca4=7'b0010010;
           4'b0011: Ca4=7'b0000110;
           4'b0100: Ca4=7'b1001100;
           4'b0101: Ca4=7'b0100100;
           4'b0110: Ca4=7'b0100000;
           4'b0111: Ca4=7'b0001111;
           4'b1000: Ca4=7'b0000000;
           4'b1001: Ca4=7'b0000100;
           4'b1010: Ca4=7'b0001000;
           4'b1011: Ca4=7'b1100000;
           4'b1100: Ca4=7'b0110001;
           4'b1101: Ca4=7'b1000010;
           4'b1110: Ca4=7'b0110000;
           4'b1111: Ca4=7'b0111000;
           default Ca4=7'b1111111;
     endcase
end
always@(an)begin
   case(an)
   4'b0111:ca=Ca1;
   4'b1011:ca=Ca2;
   4'b1101:ca=Ca3;
   4'b1110:ca=Ca4;
   endcase
end

endmodule
```

**Shift*Left*2.v**

```
`timescale 1ns / 1ps
module Shift_Left_2(input [31:0] before,
```

```
                    output wire [31:0] after );
                  assign after=before[29:0]+2'b00;

endmodule
```

**Sign_Extend.v**

```
`timescale 1ns / 1ps

module Sign_Extend(input [15:0] small_In ,
                   output wire [31:0] big_Out);
                  assign big_Out={{16{small_In[15]}},small_In[15:0]};
endmodule
```

**TopModule.v**

```
`timescale 1ns / 1ps

module TopModule(
    input [4:0]reg_Sel,
    input reg_0_pc_1,
    input clk,clock_Push,

    output [6:0] ca,
    output [3:0] an
   );
    wire [31:0] PC_value,reg_value;
    SSD ssd( PC_value,reg_value,clk,reg_0_pc_1,an,ca);
    Pipeline pipeline( clock_Push, reg_Sel, PC_value, reg_value);
endmodule
```

**pipeline.v**

```
`timescale 1ns / 1ps


`include "PC.v"
`include "Instruction_Mem.v"
`include "IF_ID.v"
`include "Hazard.v"
`include "Reg_File.v"
`include "ID_EX.v"
`include "EX_MEM.v"
`include "MEM_WB.v"
`include "ALU.v"
`include "ALU_Control.v"
`include "MUX221.v"
`include "MUX321.v"
`include "Jump_Address.v"
`include "Sign_Extend.v"
`include "If_Equal.v"
`include "Data_Mem.v"
```

```
`include "Control.v"
`include "Forwarding.v"


module Pipeline(
        input clock,
        input [4:0] register_switch,
        output [31:0] pc_out,
        output [31:0] register_out);

        wire
                        id_ex__in__RegDst,
                        id_ex__in__MemRead,
                        id_ex__in__MemtoReg,
                        id_ex__in__MemWrite,
                        id_ex__in__ALUSrc,
                        id_ex__in__RegWrite,
                        id_ex__out__RegDst,
                        id_ex__out__MemRead,
                        id_ex__out__MemtoReg,
                        id_ex__out__MemWrite,
                        id_ex__out__ALUSrc,
                        id_ex__out__RegWrite;
        wire [1:0]
                        id_ex__in__ALUOp_2,
                        id_ex__out__ALUOp_2;
        wire [31:0]
                        id_ex__in__ExtendedIm_32,
                        id_ex__in__ReadData1_32,
                        id_ex__in__ReadData2_32,
                        id_ex__out__ExtendedIm_32,
                        id_ex__out__ReadData1_32,
                        id_ex__out__ReadData2_32;
        wire [4:0]
                        id_ex__in__ReadRegister1_5,
                        id_ex__in__ReadRegister2_5,
                        id_ex__in__Rt_5,
                        id_ex__in__Rd_5,
                        id_ex__out__ReadRegister1_5,
                        id_ex__out__ReadRegister2_5,
                        id_ex__out__Rt_5,
                        id_ex__out__Rd_5;

        // ====================================================

        wire
                        ex_mem__in__MemRead,
                        ex_mem__in__MemtoReg,
                        ex_mem__in__MemWrite,
                        ex_mem__in__RegWrite,
                        ex_mem__out__MemRead,
                        ex_mem__out__MemtoReg,
                        ex_mem__out__MemWrite,
                        ex_mem__out__RegWrite ;
```

```
wire [31:0]
                ex_mem__in__ALUResult_32,
                ex_mem__in__ReadData_32,
                ex_mem__out__ALUResult_32,
                ex_mem__out__ReadData_32 ;
wire [4:0]
                ex_mem__in__RegisterDst,
                ex_mem__out__ReadRegister1,
                ex_mem__out__ReadRegister2,
                ex_mem__out__RegisterDst;

// ================================================

wire
                mem_wb__in__MemtoReg,
                mem_wb__in__RegWrite,
                mem_wb__out__MemToReg,
                mem_wb__out__RegWriteWB;
wire        [31:0]
                mem_wb__out__ReadFromMemory_32,
                mem_wb__out__ALUResult_32;
wire        [4:0]
                mem_wb__out__Rd;


wire [31:0]         PcNext,
                PCBranch;
wire IfBr;

wire [31:0]         mux_pc_in_1__out_32;
wire [31:0]         jump_address__out__data_32;
    wire [31:0]         mux_pc_in_2__out_32;
    wire                 pc__in__hold;
    wire [31:0]     pc__in__next_32;
    wire [31:0]     pc__out__address_32;
wire [31:0]         ins_mem__out__ins_32;
    wire
                  if_id__in__hold,
                    if_id__in__flush;
    wire [31:0]
                    if_id__in__addr_32,
                    if_id__out__PCNext_32,
                    if_id__out__ins_32;
    wire
                control__out__jump,
                control__out__branch,
                control__out__bne,
                control__out__MemRead,
                control__out__MemtoReg,
                control__out__MemWrite,
                control__out__ALUSrc,
                control__out__RegWrite,
                control__out__RegDst;
```

```
        wire [1:0]            control__out__ALUOp;
    wire                 hazard__out__pc_hold,
                    hazard__out__if_id_hold,
                    hazard__out__id_ex_flush,
                    hazard__out__if_flush ;
        wire                 if_equal__out__if_zero;
    wire         [7:0]         control__out__combined;
    wire                 mux_control_out__in__id_ex_flush;
        wire [7:0]          mux_control_out__out__combined;
    wire [31:0]         sign_extend__out__data_32;
    wire                 reg_file__in__RegWrite;
        wire [4:0]
                         reg_file__in__read_addr_1_5,
                        reg_file__in__read_addr_2_5,
                         reg_file__in__write_addr_2_5
                         ;
        wire [31:0]
                        reg_file__in__write_data_32,
                         reg_file__out__read_data_1_32,
                        reg_file__out__read_data_2_32
                    ;
    wire                 Fw1;
        wire [31:0]         mux_regfile_out_1__out__data_32;
        //wire [31:0]         ex_mem__out__address_32;
    wire Fw2;
       wire [31:0]         mux_regfile_out_2__out__data_32;
    wire [31:0]        mux_ex_1__out__data_32;
        wire [1:0]         mux_ex_1__sel__ForwadA;

        wire [31:0]         mux_mem_wb_out__out__data_32;
    wire [1:0]         mux_ex_2__sel__ForwadB;
        wire [31:0]         mux_ex_2__out__data_32;
    wire [31:0]         mux_ex_3__out__data_32;
    wire                 mux_ex_4__in__RegDst;
        wire [31:0]         mux_ex_4__out__data_32;
        wire [3:0]         alu__in__alu_control;
        wire [31:0]         alu__out__data_32;
    wire [31:0]        data_mem__out__Data_32;
    wire [5:0]         alu_control__in__funct;
        wire [1:0]         alu_control__in__ALUop;

        assign  pc__in__next_32=mux_pc_in_2__out_32;

        assign         IfBr = ((if_equal__out__if_zero)&&(control__out__branch==1'b1))
||(((if_equal__out__if_zero==1'b0)&&(control__out__bne==1'b1))?1:0);

    assign         PcNext = pc__out__address_32+ 32'h4;

    ////////////////////////////////////////////?////////////////////

    assign         PCBranch = if_id__out__PCNext_32 + (sign_extend__out__data_32<<2);

    MUX221 #(32) mux_pc_in_1(.sel(IfBr),
    .a(PcNext),
```

```
        .b(PCBranch),
        .out(mux_pc_in_1__out_32)
        );


//        MUX221 #(32) mux_pc_in_2(.sel(1'b1),
        MUX221 #(32) mux_pc_in_2(.sel(control__out__jump),
        .a(mux_pc_in_1__out_32),
        .b(jump_address__out__data_32),
        .out(mux_pc_in_2__out_32)
        );


        PC pc(.clk(clock),
                .hold(pc__in__hold),
                .next(pc__in__next_32),
                .address(pc__out__address_32)
        );
            assign pc_out=pc__out__address_32;


        wire [25:0] jump_address__in__jump_where;
        assign jump_address__in__jump_where = if_id__out__ins_32[25:0];
        wire [3:0] jump_address__in__PcNext;
        assign jump_address__in__PcNext = PcNext;

        Jump_Address jump_address(
                .JumpWhere26(if_id__out__ins_32[25:0]),
                .PCNext_4(PcNext[31:28]),
                .JumpAddress(jump_address__out__data_32)
        );


        Instruction_Mem ins_mem(.addr(pc__out__address_32),
                                .out_Instr(ins_mem__out__ins_32)
        );

        assign if_id__in__addr_32=PcNext;

        IF_ID if_id(.clk(clock),
                .hold(if_id__in__hold),
                .Flush(if_id__in__flush),
                .Instr(ins_mem__out__ins_32),
                .Addr(if_id__in__addr_32),
                .out_Instr(if_id__out__ins_32),
                .out_Addr(if_id__out__PCNext_32)
        );

         Control control (.opcode(if_id__out__ins_32[31:26]),
                 // output
                .jump(control__out__jump),
                .Branch(control__out__branch),
                .Bne(control__out__bne),
                //
```

```
        .MemRead(control__out__MemRead),
        .MemtoReg(control__out__MemtoReg) ,
        .MemWrite(control__out__MemWrite),
        .ALUSrc(control__out__ALUSrc),
        .RegWrite(control__out__RegWrite),
        .RegDst(control__out__RegDst),
        .ALUOp(control__out__ALUOp)
);


Hazard hazard(
                .ID_EX_MemRead(id_ex__out__MemRead),
                .EX_MEM_MemRead(ex_mem__out__MemRead),
                .ID_EX_RegWrite(id_ex__out__RegWrite),
                .ID_EX_RegDst(id_ex__out__RegDst),
                .jump(control__out__jump),
                .bne(control__out__bne),
                .beq(control__out__branch),
                .IfEqual(if_equal__out__if_zero),
                .ID_EX_RegisterRt(id_ex__out__ReadRegister2_5),
                .ID_EX_RegisterRd(id_ex__out__Rd_5),
                .IF_ID_RegisterRs(id_ex__in__ReadRegister1_5),
                .IF_ID_RegisterRt(id_ex__in__ReadRegister2_5),
                .EX_MEM_RegisterRd(ex_mem__out__RegisterDst),
                //                output
                .PC_Hold(hazard__out__pc_hold),
                .IF_ID_Hold(hazard__out__if_id_hold),
                .ID_EX_Flush(hazard__out__id_ex_flush),
                .IF_Flush(hazard__out__if_flush)
);

assign pc__in__hold = hazard__out__pc_hold;
assign if_id__in__hold = hazard__out__if_id_hold;
assign if_id__in__flush = hazard__out__if_flush;

assign control__out__combined = {
                control__out__MemRead,
                control__out__MemtoReg,
                control__out__MemWrite,
                control__out__ALUSrc,
                control__out__RegWrite,
                control__out__RegDst,
                control__out__ALUOp};

assign mux_control_out__in__id_ex_flush  = hazard__out__id_ex_flush;

MUX221 #(8)        mux_control_out(
        .sel(mux_control_out__in__id_ex_flush),
        .a(control__out__combined),
        .b(8'h0),
        .out(mux_control_out__out__combined)
);

Sign_Extend sign_extend(.small_In(if_id__out__ins_32[15:0]),
```

```
            .big_Out(sign_extend__out__data_32)
    );

    assign id_ex__in__ExtendedIm_32 = sign_extend__out__data_32;

assign reg_file__in__RegWrite=mem_wb__out__RegWriteWB;
    assign reg_file__in__write_data_32 = mux_mem_wb_out__out__data_32;
    assign reg_file__in__write_addr_2_5 = mem_wb__out__Rd;
    Reg_File reg_file(.clk(clock),
    .Reg_switch(register_switch),
            .RegWrite(reg_file__in__RegWrite),
            .ReadRegister1(reg_file__in__read_addr_1_5),
            .ReadRegister2( reg_file__in__read_addr_2_5),
            .WriteReg(reg_file__in__write_addr_2_5),
            .WriteData(reg_file__in__write_data_32),
            .read_data1(reg_file__out__read_data_1_32),
            .read_data2(reg_file__out__read_data_2_32),
            .Reg_Read(register_out)
    );

    assign reg_file__in__read_addr_1_5 = if_id__out__ins_32[25:21];
        assign reg_file__in__read_addr_2_5 = if_id__out__ins_32[20:16];

    MUX221 #(32) mux_regfile_out_1(
            .sel(Fw1),
            .a(reg_file__out__read_data_1_32),
            .b(ex_mem__out__ALUResult_32),
            .out(mux_regfile_out_1__out__data_32)
    );


    MUX221 #(32) mux_regfile_out_2(
            .sel(Fw2),
            .a(reg_file__out__read_data_2_32),
            .b(ex_mem__out__ALUResult_32),
            .out(mux_regfile_out_2__out__data_32)
    );

    //wire         if_equal__out__if_zero;
    If_Equal if_equal(
            .a(mux_regfile_out_1__out__data_32),
            .b(mux_regfile_out_2__out__data_32),
            .IfEqual(if_equal__out__if_zero)
    );


    assign id_ex__in__RegDst = mux_control_out__out__combined[2];
    assign id_ex__in__MemRead = mux_control_out__out__combined[7];
    assign id_ex__in__MemtoReg = mux_control_out__out__combined[6];
    assign id_ex__in__MemWrite = mux_control_out__out__combined[5];
    assign id_ex__in__ALUSrc = mux_control_out__out__combined[4];
    assign id_ex__in__RegWrite = mux_control_out__out__combined[3];
    assign id_ex__in__ALUOp_2=mux_control_out__out__combined[1:0];
```

```
assign id_ex__in__ReadRegister1_5=reg_file__in__read_addr_1_5;
assign id_ex__in__ReadRegister2_5=reg_file__in__read_addr_2_5;
assign id_ex__in__Rt_5=reg_file__in__write_addr_2_5;
assign id_ex__in__Rd_5=if_id__out__ins_32[15:11];



ID_EX id_ex(.clk(clock),
        .RegDst(id_ex__in__RegDst),
        .MemRead(id_ex__in__MemRead),
        .MemtoReg(id_ex__in__MemtoReg),
        .MemWrite(id_ex__in__MemWrite),
        .ALUSrc(id_ex__in__ALUSrc),
        .RegWrite(id_ex__in__RegWrite),
        // input 1:0
        .ALUOp(id_ex__in__ALUOp_2),
        // input 31:0
        .ReadData1(id_ex__in__ReadData1_32),
        .ReadData2(id_ex__in__ReadData2_32),
        .ExtendedIm(id_ex__in__ExtendedIm_32),
        // input 4:0
        .ReadRegister1(id_ex__in__ReadRegister1_5),
        .ReadRegister2(id_ex__in__ReadRegister2_5),
        .Rt(id_ex__in__Rt_5),
        .Rd(id_ex__in__Rd_5),
        // ouput
        .RegDstEX(id_ex__out__RegDst),
        .MemReadEX(id_ex__out__MemRead),
        .MemtoRegEX(id_ex__out__MemtoReg),
        .MemWriteEX(id_ex__out__MemWrite),
        .ALUSrcEX(id_ex__out__ALUSrc),
        .RegWriteEX(id_ex__out__RegWrite),
        // ouput 31:0
        .ALUOpEX(id_ex__out__ALUOp_2),
        .ReadData1EX(id_ex__out__ReadData1_32),
        .ReadData2EX(id_ex__out__ReadData2_32),
        .ExtendedImEX(id_ex__out__ExtendedIm_32),
        // 4:0
        .ReadRegister1EX(id_ex__out__ReadRegister1_5),
        .ReadRegister2EX(id_ex__out__ReadRegister2_5),
        .RtEX(id_ex__out__Rt_5),
        .RdEX(id_ex__out__Rd_5)
);

assign id_ex__in__ReadData1_32 =  mux_regfile_out_1__out__data_32;
assign id_ex__in__ReadData2_32 =  mux_regfile_out_2__out__data_32;


MUX321 #(32) mux_ex_1(
        .sel(mux_ex_1__sel__ForwadA),
        .a(id_ex__out__ReadData1_32),
        .b(mux_mem_wb_out__out__data_32),
        .c(ex_mem__out__ALUResult_32),
        .out(mux_ex_1__out__data_32)
```

```
);

MUX321 #(32) mux_ex_2(
        .sel(mux_ex_2__sel__ForwadB),
        .a(id_ex__out__ReadData2_32),
        .b(mux_mem_wb_out__out__data_32),
        .c(ex_mem__out__ALUResult_32),
        .out(mux_ex_2__out__data_32)
);


MUX221 #(32) mux_ex_3(
        .sel(id_ex__out__ALUSrc),
        .a(mux_ex_2__out__data_32),
        .b(id_ex__out__ExtendedIm_32),
        .out(mux_ex_3__out__data_32)
);

assign mux_ex_4__in__RegDst = id_ex__out__RegDst;

MUX221 #(5) mux_ex_4(
        .sel(mux_ex_4__in__RegDst),
        .a(id_ex__out__ReadRegister2_5),
        .b(id_ex__out__Rd_5),
        .out(ex_mem__in__RegisterDst)
);



ALU alu(
        .control(alu__in__alu_control),
        .a(mux_ex_1__out__data_32),
        .b(mux_ex_3__out__data_32),
        .result(alu__out__data_32)
);


assign alu_control__in__ALUop = id_ex__out__ALUOp_2;
assign alu_control__in__funct = id_ex__out__ExtendedIm_32[5:0];

ALU_Control alu_control(
        .funct(alu_control__in__funct),
        .ALUop(alu_control__in__ALUop),
        .control(alu__in__alu_control)
);


Forwarding forwarding(
        .MEM_WB_RegWrite(mem_wb__out__RegWriteWB) ,
        .EX_MEM_RegWrite(ex_mem__out__RegWrite),
        .bne(control__out__bne),
        .beq(control__out__branch),
//              input [4:0]
        .MEM_WB_RegisterRd(mem_wb__out__Rd),
```

```
                .ID_EX_RegisterRs(id_ex__out__ReadRegister1_5),
                .EX_MEM_RegisterRd(ex_mem__out__RegisterDst),
                .ID_EX_RegisterRt(id_ex__out__ReadRegister2_5),
                .IF_ID_RegisterRs(reg_file__in__read_addr_1_5),
                .IF_ID_RegisterRt(reg_file__in__read_addr_2_5),
//                  output reg [1:0]
                .ForwardA(mux_ex_1__sel__ForwadA),
                .ForwardB(mux_ex_2__sel__ForwadB),
//                  output reg
                .Fw1(Fw1),
                .Fw2(Fw2)
        );


        assign ex_mem__in__MemRead = id_ex__out__MemRead;
        assign ex_mem__in__MemtoReg = id_ex__out__MemtoReg;
        assign ex_mem__in__MemWrite = id_ex__out__MemWrite;
        assign ex_mem__in__RegWrite = id_ex__out__RegWrite;
        assign ex_mem__in__ReadData_32 = mux_ex_2__out__data_32;
        assign ex_mem__in__ALUResult_32 = alu__out__data_32;


        EX_MEM ex_mem(
        //                  input
                        .clk(clock),
                        .MemRead(ex_mem__in__MemRead),
                        .MemtoReg(ex_mem__in__MemtoReg),
                        .MemWrite(ex_mem__in__MemWrite),
                        .RegWrite(ex_mem__in__RegWrite),
        //              input [31:0]
                        .ALUResultAddr(ex_mem__in__ALUResult_32),
                        .DataWriteIn(ex_mem__in__ReadData_32),
        //              input[4:0]
                        .RegisterDst(ex_mem__in__RegisterDst),
        //              output reg
                        .MemReadM(ex_mem__out__MemRead),
                        .MemtoRegM(ex_mem__out__MemtoReg),
                        .MemWriteM(ex_mem__out__MemWrite),
                        .RegWriteM(ex_mem__out__RegWrite),
        //              output reg [31:0]
                        .ALUResultAddrM(ex_mem__out__ALUResult_32),
                        .DataWriteInM(ex_mem__out__ReadData_32),
        //              output reg [4:0]
                        .RegisterDstM(ex_mem__out__RegisterDst)
        );



        Data_Mem data_mem(
        //        input [31:0]
                .address(ex_mem__out__ALUResult_32),
                .Write_data(ex_mem__out__ReadData_32),
        //        input
                .MemWrite(ex_mem__out__MemWrite),
```

```
                .MemRead(ex_mem__out__MemRead),
                .clk(clock),
        //        output [31:0]
                .Read_data(data_mem__out__Data_32)
        );


        assign mem_wb__in__RegWrite = ex_mem__out__RegWrite;
        assign mem_wb__in__MemtoReg = ex_mem__out__MemtoReg;
        MEM_WB mem_wb(.clk(clock),
        //              input
                        .MemtoReg(mem_wb__in__MemtoReg),
                        .RegWrite(mem_wb__in__RegWrite),
        //              input [31:0]
                        .Data2Write(data_mem__out__Data_32),
                        .ALUResult(ex_mem__out__ALUResult_32),
        //              input [4:0]
                        .RegisterDst(ex_mem__out__RegisterDst),
        //              output reg
                        .MemtoRegWB(mem_wb__out__MemToReg),
                        .RegWriteWB(mem_wb__out__RegWriteWB),
        //              output reg [31:0]
                        .Data2WriteWB(mem_wb__out__ReadFromMemory_32),
                        .ALUResultWB(mem_wb__out__ALUResult_32),
        //              output reg [4:0]
                        .RegisterDstWB(mem_wb__out__Rd)
        );



        //wire [31:0]        mux_mem_wb_out__out__data_32;
        MUX221 #(32) mux_mem_wb_out(.sel(mem_wb__out__MemToReg),
                .a(mem_wb__out__ReadFromMemory_32),
                .b(mem_wb__out__ALUResult_32),
                .out(mux_mem_wb_out__out__data_32)
        );


endmodule
```

**pipeline__tb.v**

```
`timescale 1ns / 1ps

`include "pipeline.v"

module pipeline_tb;

    integer i = 0;

        reg clk;

        Pipeline uut (
```

```
                .clock(clk)
        );

        initial begin
                // Initialize Inputs
                clk = 0;
        $display("Texual result of pipeline:");
        $display("============================================================");
        #630;
        $stop;
        end

        //wire [31:0]         pc__out__address_32;
    always #10 begin
        $display("Time: %d, CLK = %d, PC = 0x%H", i, clk, uut.pc__out__address_32);
        $display("[$s0] = 0x%H, [$s1] = 0x%H, [$s2] = 0x%H", uut.reg_file.registers[16],uut.reg_file.reg
        $display("[$s3] = 0x%H, [$s4] = 0x%H, [$s5] = 0x%H", uut.reg_file.registers[19], uut.reg_file.re
        $display("[$s6] = 0x%H, [$s7] = 0x%H, [$t0] = 0x%H", uut.reg_file.registers[22], uut.reg_file.re
        $display("[$t1] = 0x%H, [$t2] = 0x%H, [$t3] = 0x%H", uut.reg_file.registers[9], uut.reg_file.reg
        $display("[$t4] = 0x%H, [$t5] = 0x%H, [$t6] = 0x%H", uut.reg_file.registers[12], uut.reg_file.re
        $display("[$t7] = 0x%H, [$t8] = 0x%H, [$t9] = 0x%H", uut.reg_file.registers[15], uut.reg_file.re
        $display("================================================");

        clk = ~clk;
        if (~clk) i = i + 1;
    end



endmodule
```