EE4363 / CSci 4203 – Computer Architecture Machine Problem 2

ALEX LEMA

Lemac001

We want to support hazard detection; therefore, we must add stall as bellow: ECMEMStall & IFIDStall new pipelines stall.

```
            //NEW STALL**************************
            EXMEMStall, IFIDStall; // pipeline latches
```

Feeding NOP instructions to the execution stage:

```
  //new added*****************
    IFIDStall = nop;
    EXMEMStall = nop;
      //ExMemStall: ExMem_STALL <= '0';
      //IfIdStall: IfId_STALL <= '0';
```

Iqual to zero:

```
  //new added*****************
    IFIDStall = 0;
    EXMEMStall = 0;
      //ExMemStall: ExMem_STALL <= '0';
      //IfIdStall: IfId_STALL <= '0';
```

Tests for MIPS pipeline Verilog module

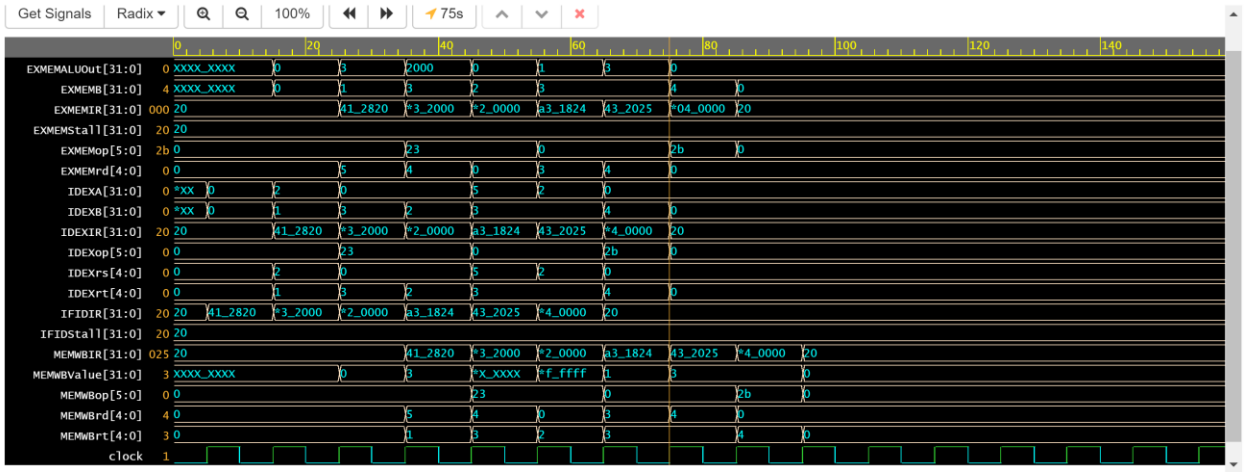
```
    // Initialize pipeline data
    // test some instructions
    for (i=0;i<=31;i=i+1) Regs[i] = i; // initialize registers

    // IMemory[0] = 32'h8c210003;
    // IMemory[1] = 32'hac020000;
    // IMemory[2] = 32'h00642820;

    for (j=0;j<=1023;j=j+1) IMemory[j] = nop;

    // DMemory[0] = 32'h00000000;
    // DMemory[1] = 32'hffffffff;

    for (k=0;k<=1023;k=k+1) DMemory[k] = 0;

    //new define*****************8
    `define RTYPE(Op, RD, RS, RT,funct)\(Op << (26))|(RS << (21))|(RT<<(16))|(RD<<(11))|(funct)
    `define IMRTYPE(Op, RD, IM)\(IM << (11))|(RD << (16))|(Op << (26))
     IMemory[0] = `RTYPE(ALUop, 5, 2, 1, 32);
     IMemory[1] = `IMRTYPE(LW, 3, 4);
     IMemory[2] = `IMRTYPE(LW, 2, 0);
     IMemory[3] = `RTYPE(ALUop, 3, 5, 3, 36);
     IMemory[4] = `RTYPE(ALUop, 4, 2, 3, 37);
     IMemory[5] = `IMRTYPE(SW, 4, 0);

    DMemory[0] = 32'hffffffff;
    DMemory[4] = 32'hffffffff;
```

With these we complete the Verilog code for your MIPS processor. I have modify the code to support ADD, OR, & AND instructions.

```
        //AND , OR//
           36: EXMEMALUOut <= Ain & Bin; // and operation
           37: EXMEMALUOut <= Ain | Bin; // or operation
```

In the output we can see the instructions don’t depend on completion of data access by a previous instruction compare with the incomplete code given in canvas for this exercise.

With nop:



Without nop: