Chase Jones

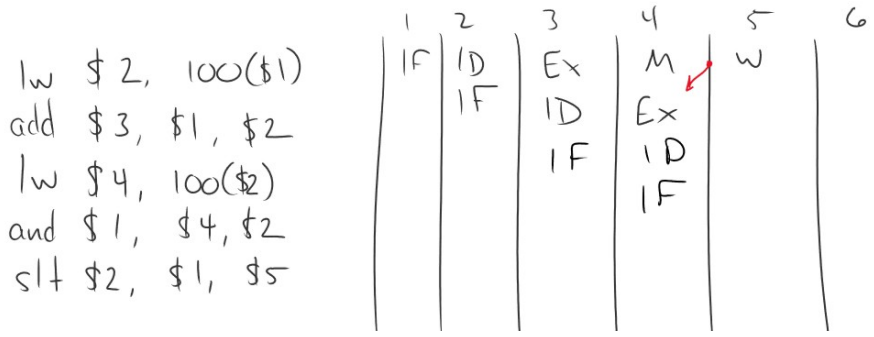
Arad

CSC 142

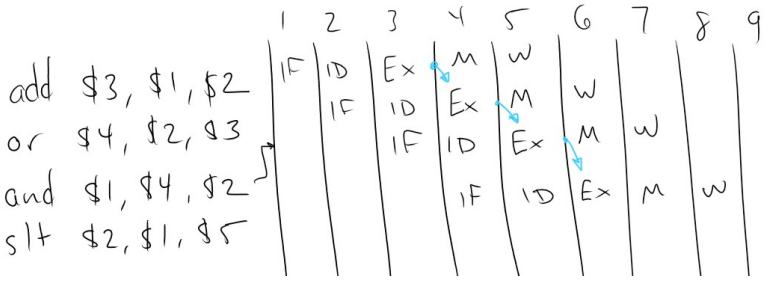
29 October 2018

Assignment 5

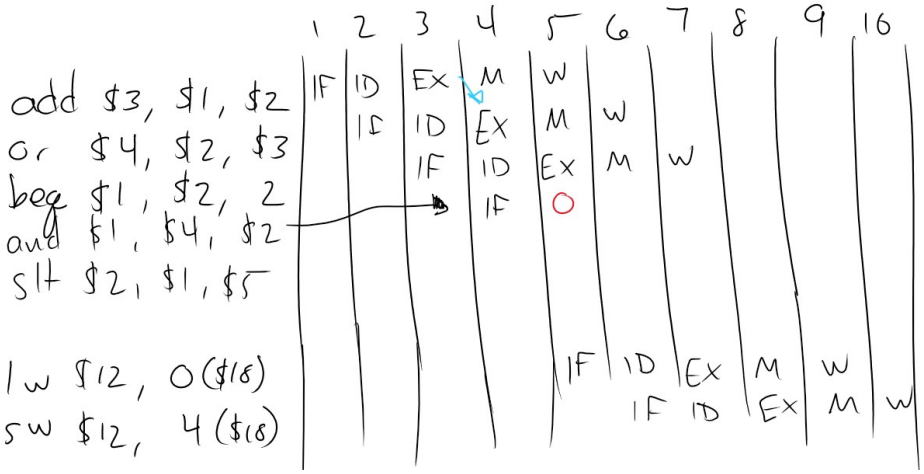
1. Based on the diagram provided in Figure 4.60, this data path has no way to execute the program correctly. The result of memory access is not fed into the forwarding unit, so the execution of ADD is missing the correct result of the load in $2.



2. Based on the diagram in 4.60, with forwarding, the program executes in 8 cycles.



3. Based on the diagram in 4.62, with branch taken, the program executes in 10 cycles



4.

```
module register(data_in, data_out, clk, rst, w_Enable);
parameter SIZE=32;

input [SIZE-1:0]data_in;
input clk, rst, w_Enable;
output reg [SIZE-1:0]data_out;

reg [SIZE-1:0]value;

always@(posedge clk, negedge rst)
begin
        if(!rst)
                value = 0;
        else if(w_Enable == 1)
                value <= data_in;
        else    value <= value;
end

always@(*)
        data_out = value;

endmodule
```

```
`include "register.v"
module register_fixture();

reg [31:0] a;
reg [63:0] c;
reg [15:0] e;

wire [31:0] b;
wire [63:0] d;
wire [15:0] f;

reg clk, rst, w;

register #(.SIZE(32)) reg32(.data_in(a), .data_out(b), .clk(clk), .rst(rst),
.w_Enable(w));
```

```
register #(.SIZE(64)) reg64(.data_in(c), .data_out(d), .clk(clk), .rst(rst),
.w_Enable(w));
register #(.SIZE(16)) reg16(.data_in(e), .data_out(f), .clk(clk), .rst(rst),
.w_Enable(w));

initial
begin
        a=32'b0;
        c=64'b0;
        e=16'b0;
end

initial
        forever #1 clk = ~clk;

initial
begin
        clk = 0;
        rst = 1;
        w = 0;
        $vcdpluson;
end

initial $monitor("Time = %3d\n 16 bit reg = %b = %4d\n 32 bit reg = %b = %4d\n 64
bit reg = %b = %4d\n w= %b rst = %b\n", $time,f,f,b,b,d,d,w,rst);

initial
begin
        $display("No write.");
        #10
         a=32'd6100;
         c=64'd1500;
         e=16'b10;
        $display("Time = %3d    16 bit in = %4d, 32 bit in = %4d, 64 bit in =
%4d\nw = %b, rst= %b\n\n", $time,e,a,c,w,rst);

        #10
        $display("Write asserted.");
        w = 1;
```

```
        #10
        $display("No write.");
        $display("Time = %3d    16 bit in = %4d, 32 bit in = %4d, 64 bit in =
%4d\nw = %b, rst= %b\n\n", $time,e,a,c,w,rst);
        w=0;
        a=32'd12;
        c=64'd15;
        e=16'd9;

        #10
        $display("Write asserted.");
        w=1;

        #10
        $display("Reset asserted.");
        rst = 0;
        w=0;
        #5;
        rst = 1;

        #10
        $display("Write during reset.");
        $display("Time = %3d    16 bit in = %4d, 32 bit in = %4d, 64 bit in =
%4d\nw = %b, rst= %b\n\n", $time,e,a,c,w,rst);
        w=1;
        rst = 0;
        a = 32'd100;
        c = 64'd200;
        e = 16'd300;
        #10
        w=0;
        #5;
        rst = 1;
        #10;

        $finish;
end

endmodule
```

```
No write.
Time =   0
 16 bit reg = xxxxxxxxxxxxxxxx =     x
 32 bit reg = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx =          x
 64 bit reg = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx =
x
 w= 0 rst = 1

Time =  10      16 bit in =    2, 32 bit in = 6100, 64 bit in = 1500
w = 0, rst= 1


Write asserted.
Time =  20
 16 bit reg = xxxxxxxxxxxxxxxx =     x
 32 bit reg = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx =          x
 64 bit reg = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx =
x
 w= 1 rst = 1

Time =  21
 16 bit reg = 0000000000000010 =    2
 32 bit reg = 00000000000000000001011111010100 = 6100
 64 bit reg = 0000000000000000000000000000000000000000000000000010111011100 =
1500
 w= 1 rst = 1

No write.
Time =  30      16 bit in =    2, 32 bit in = 6100, 64 bit in = 1500
w = 1, rst= 1


Time =  30
 16 bit reg = 0000000000000010 =    2
 32 bit reg = 00000000000000000001011111010100 = 6100
 64 bit reg = 0000000000000000000000000000000000000000000000000010111011100 =
1500
 w= 0 rst = 1

Write asserted.
```

```
Time =  40
 16 bit reg = 0000000000000010 =    2
 32 bit reg = 00000000000000000001011111010100 = 6100
 64 bit reg = 0000000000000000000000000000000000000000000000000000010111011100 =
1500
 w= 1 rst = 1

Time =  41
 16 bit reg = 0000000000001001 =    9
 32 bit reg = 00000000000000000000000000001100 =   12
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000001111 =
15
 w= 1 rst = 1

Reset asserted.
Time =  50
 16 bit reg = 0000000000000000 =    0
 32 bit reg = 00000000000000000000000000000000 =    0
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000000000 =
0
 w= 0 rst = 0

Time =  55
 16 bit reg = 0000000000000000 =    0
 32 bit reg = 00000000000000000000000000000000 =    0
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000000000 =
0
 w= 0 rst = 1

Write during reset.
Time =  65      16 bit in =    9, 32 bit in =   12, 64 bit in =   15
w = 0, rst= 1



Time =  65
 16 bit reg = 0000000000000000 =    0
 32 bit reg = 00000000000000000000000000000000 =    0
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000000000 =
0
 w= 1 rst = 0
```

```
Time =  75
 16 bit reg = 0000000000000000 =    0
 32 bit reg = 00000000000000000000000000000000 =    0
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000000000 =
0
 w= 0 rst = 0

Time =  80
 16 bit reg = 0000000000000000 =    0
 32 bit reg = 00000000000000000000000000000000 =    0
 64 bit reg = 0000000000000000000000000000000000000000000000000000000000000000 =
0
 w= 0 rst = 1

$finish called from file "register_fixture.v", line 84.
$finish at simulation time                   90
           V C S   S i m u l a t i o n   R e p o r t
Time: 90
CPU Time:      0.340 seconds;       Data structure size:   0.0Mb
Mon Oct 29 13:08:16 2018
```