```
1   `timescale 1ns / 1ps
2   /**********************************************************************************
3    *
4    * Author:   Jesus Luciano & Rosswell Tiongco
5    * Filename: IDP.v
6    * Date:     2/25/2019
7    * Version:  1.2
8    *
9    * Notes:1.0 Module connects register file and ALU, and ties additional control
10   *           signals. 2 multiplexers are used to select ALU_OUT and the second
11   *           input to the RT register.
12   *
13   *           Two of the new resgisters, HI and LO, are used to store the output
14   *           of the ALU module.
15   *
16   *           RS and RT register load the data output form the register file
17   *           after every clock cycle and feed it to the ALU.
18   *
19   *           The ALU_Out register loads the data output from the ALU after
20   *           every clock cycle.
21   *
22   *           The D_in register loads the DY input after every clock cycle
23   *
24   *       1.1 Added DA_mux to select between D_Addr and T_Addr for the write
25   *           address of the regfile, controlled by new DA_sel signal
26   *
27   *       1.2 Increased size of DA_sel to select between more inputs to the
28   *           DA_mux
29   *
30   *       1.3 Added mux to D_in input controlled by io_rd signal that selects
31   *           between io_out and dmem_out. Expanded T_MUX to now select between
32   *           the flags, SE_16, T output of the Regfile, and the PC
33   *
34   *       1.4 Added flags input and flags output. S_MUX added that selects between
35   *           the regfile output and the ALU_OUT output
36   *
37   **********************************************************************************/
38  module IDP(clk, reset, S_Addr, FS, HILO_ld, D_En, D_Addr, T_Addr,
39                         DT, T_Sel, C, V, N, Z, DY, PC_in, Y_Sel, ALU_OUT, D_OUT,
40                         DA_Sel, shamt, io_rd, io_out, stack, flags_in, flags_out);
41
42      input         clk, reset, HILO_ld, D_En, io_rd, stack;
43      input [ 1:0]  DA_Sel;
44      input [ 2:0]  Y_Sel, T_Sel;
45      input [ 4:0]  S_Addr, FS, D_Addr, T_Addr, shamt, flags_in;
46      input [31:0]  DT, DY, PC_in, io_out;
47
48      output             C, V, N, Z;
49      output      [ 4:0] flags_out;
50      output      [31:0] D_OUT;
51      output wire [31:0] ALU_OUT;
52
53      wire [ 4:0] DA_mux, stack_mux;
54
55      wire [31:0] REG_FILE_S, REG_FILE_T, T_MUX, S_MUX,
56                  Y_hi, Y_lo, HI_out, LO_out,
57                  RS_out, RT_out, ALU_reg_out, D_in_out;
```

```
58
59      wire [31:0] mem_in;
60
61      //DA-mux for Regfile Write Address
62      //Selects between rd and rt as source for write address
63      assign DA_mux = (DA_Sel == 2'b11) ? 5'h1D ://Reg 29
64                      (DA_Sel == 2'b10) ? 5'h1F ://Reg 31
65                      (DA_Sel == 2'b01) ? T_Addr:
66                                          D_Addr;//Default
67
68      //S-MUX
69      assign S_MUX = (T_Sel == 3'b100) ? ALU_OUT   ://Y_Mux output
70                                         REG_FILE_S;//Regfile
71
72      //T-MUX
73      assign T_MUX = (T_Sel == 3'b011) ? {26'b0, flags_in}://flags_in
74                     (T_Sel == 3'b010) ? PC_in            ://PC
75                     (T_Sel == 3'b001) ? DT               ://SE_16
76                                         REG_FILE_T       ;//Regfile
77
78      //Y-MUX
79      assign ALU_OUT = (Y_Sel == 3'b100) ? HI_out      ://HI register
80                       (Y_Sel == 3'b011) ? LO_out      ://Lo register
81                       (Y_Sel == 3'b010) ? ALU_reg_out ://ALU_OUT register
82                       (Y_Sel == 3'b001) ? D_in_out    ://D_in register
83                                           PC_in       ;//PC
84
85      //IO-Mux - selects between io memory out (1) and data memory (0)
86      assign mem_in = (io_rd) ? io_out : DY;
87
88      //stack mux selects between S_Addr input and the stack pointer as the
89      //output to the RS register
90      assign stack_mux = (stack) ? 5'h1D : S_Addr;
91
92      //strip lower 5 bits out Y-Mux output for flags
93      assign flags_out = ALU_OUT[4:0];
94
95      //module reg32(clk, reset, ld, D, Q);
96      reg32      HI(.clk(clk), .reset(reset), .ld(HILO_ld),
97                    .D(Y_hi), .Q(HI_out) );
98      reg32      LO(.clk(clk), .reset(reset), .ld(HILO_ld),
99                    .D(Y_lo), .Q(LO_out) );
100
101     //ALU_OUT, D_in, RS, RT alwayy load value every clock cycle, load is set to 1
102     reg32 ALU_Out(.clk(clk), .reset(reset), .ld(1'b1),
103                   .D(Y_lo), .Q(ALU_reg_out)  );
104     reg32    D_in(.clk(clk), .reset(reset), .ld(1'b1),
105                   .D(mem_in)  , .Q(D_in_out)     );
106     reg32      RS(.clk(clk), .reset(reset), .ld(1'b1),
107                   .D(S_MUX), .Q(RS_out) );
108     reg32      RT(.clk(clk), .reset(reset), .ld(1'b1),
109                   .D(T_MUX), .Q(D_OUT)       );
110
111     //module alu_32(S, T, FS, C, V, N, Z, Y_hi, Y_lo);
112     alu_32 ALU(.S(RS_out), .T(D_OUT), .FS(FS),
113                .C(C), .V(V), .N(N), .Z(Z),
114                .Y_hi(Y_hi), .Y_lo(Y_lo), .shamt(shamt) );
```

```
115
116       //module regfile32(clk, reset, S, T, D, S_Addr, T_Addr, D_Addr, D_En);
117       regfile32 REG_FILE(.clk(clk), .reset(reset), .S(REG_FILE_S),
118                          .T(REG_FILE_T), .D(ALU_OUT), .S_Addr(stack_mux),
119                          .T_Addr(T_Addr),.D_Addr(DA_mux), .D_En(D_En) );
120
121   endmodule
```