```
1   //Aaron Hong (ahong02)
2   //Stephen Macris (smacris)
3   //5/3/23
4   //EE469 Lab3
5
6   //This module creates an asynchronous conditional unit that manages the flags register
7   //and determines conditional execution.
8
9   module cond_unit (cond, flags, ALUFlags, flag_write, flags_out, cond_ex);
10
11     input  logic [3:0] cond, flags, ALUFlags;
12     input  logic [1:0] flag_write;
13     output logic cond_ex;
14     output logic [3:0] flags_out;
15
16     always_comb begin
17        case (cond)
18           4'b0000: cond_ex = flags[2];
19           4'b0001: cond_ex = !flags[2];
20           4'b0010: cond_ex = flags[1];
21           4'b0011: cond_ex = !flags[1];
22           4'b0100: cond_ex = flags[3];
23           4'b0101: cond_ex = !flags[3];
24           4'b0110: cond_ex = flags[0];
25           4'b0111: cond_ex = !flags[0];
26           4'b1000: cond_ex = !flags[2] && flags[1];
27           4'b1001: cond_ex = flags[2] || !flags[1];
28           4'b1010: cond_ex = !(flags[3] ^ flags[0]);
29           4'b1011: cond_ex = flags[3] ^ flags[0];
30           4'b1100: cond_ex = !flags[2] && !(flags[3] ^ flags[0]);
31           4'b1101: cond_ex = flags[2] || (flags[3] ^ flags[0]);
32           4'b1110: cond_ex = 1;
33           default: cond_ex = 0;
34        endcase
35
36        case (flag_write)
37           2'b11: flags_out = ALUFlags;
38           2'b10: flags_out = {ALUFlags[3:2], 2'b00};
39           2'b01: flags_out = {2'b00, ALUFlags[1:0]};
40           2'b00: flags_out = 4'b0000;
41           default: flags_out = 4'b0000;
42        endcase
43     end
44
45  endmodule
46
47
```