* opcodes.svh

`define LOAD 3'b000

`define STORE 3'b001

`define ADD 3'b010

`define SUB 3'b011

`define BNE 3'b100

`define ADDI 3'b101

`define XOR 3'b110 // New instruction

`define LOADR 3'b111

* New control signal ALU\_XOR

module **cpu3** #(parameter WORD\_W = 8, OP\_W = 3)

             (input logic clock, n\_reset,

              input logic [WORD\_W-1:0] switches,

              output logic [6:0] disp0, disp1);

logic load\_REG,  load\_IR, ALU\_REG, ALU\_add, ALU\_sub, ALU\_xor, INC\_PC, load\_PC, WE, IMM, z\_flag;

module **alu** #(parameter WORD\_W = 8)

            (input logic clock, n\_reset, load\_REG, ALU\_REG, ALU\_add, ALU\_sub, ALU\_xor,

             input logic [WORD\_W-1:0] Adata,

             output logic [WORD\_W-1:0] Wdata,

             output logic z\_flag);

module **decoder** #(parameter OP\_W = 3)

                  (input logic clock, n\_reset, z\_flag,

                   input logic [OP\_W-1:0] op,

                   output logic load\_REG, load\_PC, load\_IR,  ALU\_REG,

                                ALU\_add, ALU\_sub, ALU\_xor, INC\_PC, WE, IMM, IND);

`XOR:   begin

              load\_REG = 1'b1;

              ALU\_REG = 1'b1;

              ALU\_xor = 1'b1;

              end

always\_comb

  begin: **com**

*// reset all the control signals to default 0*

  load\_REG = 1'b0;

  load\_IR = 1'b0;

  ALU\_REG = 1'b0;

  ALU\_add = 1'b0;

  ALU\_sub = 1'b0;

  ALU\_xor = 1'b0;

  INC\_PC = 1'b0;

  load\_PC = 1'b0;

  WE = 1'b0;

  IMM = 1'b0;

Case for XOR: ALU\_XOR sets to high

ALU\_REG and load\_REG set high for ALU

Reset control signals

* Program using XOR

always\_comb

  begin

  Idata = 0;

  case (Iaddress)

    0: Idata = {`STORE, 5'd30}; *//Store the contents of data register*

    1: Idata = {`LOAD, 5'd31};

    2: Idata = {`XOR, 5'd3};

    3: Idata = {`STORE, 5'd30}; *//Store the contents of data register*

    default: Idata = 0;         *//rest of ROM is 0*

  endcase

  end