Lab 4

Control Unit

```
`timescale 1ns / 1ps
// fpga4student.com
// FPGA projects, VHDL projects, Verilog projects
// Verilog code for RISC Processor
// Verilog code for Control Unit
module Control_Unit(
      input[3:0] opcode,
      output reg[1:0] alu_op,
      output
reg  jump,  beq,  bne,  mem_read,  mem_write,  alu_src,
reg_dst, mem_to_reg, reg_write
    );


always @(*)
begin
 case(opcode)
 4'b0000:  // LW
   begin
    reg_dst = 1'b0;
    alu_src = 1'b1;
    mem_to_reg = 1'b1;
    reg_write = 1'b1;
    mem_read = 1'b1;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b10;
    jump = 1'b0;
   end
 4'b0001:  // SW
   begin
    reg_dst = 1'b0;
    alu_src = 1'b1;
    mem_to_reg = 1'b0;
```

```
    reg_write = 1'b0;
    mem_read = 1'b0;
    mem_write = 1'b1;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b10;
    jump = 1'b0;
  end
4'b0010:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b0011:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b0100:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
```

```
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b0101:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b0110:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b0111:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
```

```
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b1000:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b1001:  // data_processing
  begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b0;
  end
4'b1011:  // BEQ
  begin
    reg_dst = 1'b0;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b0;
    mem_read = 1'b0;
    mem_write = 1'b0;
```

```
    beq = 1'b1;
    bne = 1'b0;
    alu_op = 2'b01;
    jump = 1'b0;
  end
4'b1100:  // BNE
  begin
    reg_dst = 1'b0;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b0;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b1;
    alu_op = 2'b01;
    jump = 1'b0;
  end
4'b1101:  // J
  begin
    reg_dst = 1'b0;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b0;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
    alu_op = 2'b00;
    jump = 1'b1;
  end
default: begin
    reg_dst = 1'b1;
    alu_src = 1'b0;
    mem_to_reg = 1'b0;
    reg_write = 1'b1;
    mem_read = 1'b0;
    mem_write = 1'b0;
    beq = 1'b0;
    bne = 1'b0;
```

```
    alu_op = 2'b00;
    jump = 1'b0;
   end
 endcase
 end

endmodule
```

Register

```
`timescale 1ns / 1ps
// fpga4student.com
// FPGA projects, VHDL projects, Verilog projects
// Verilog code for RISC Processor
// Verilog code for register file
module GPRs(
 input    clk,
 // write port
 input    reg_write_en,
 input  [2:0] reg_write_dest,
 input  [15:0] reg_write_data,
 //read port 1
 input  [2:0] reg_read_addr_1,
 output  [15:0] reg_read_data_1,
 //read port 2
 input  [2:0] reg_read_addr_2,
 output  [15:0] reg_read_data_2
);
 reg [15:0] reg_array [7:0];
 integer i;
 // write port
 //reg [2:0] i;
 initial begin
  for(i=0;i<8;i=i+1)
   reg_array[i] <= 16'd0;
 end
```

```
 always @ (posedge clk ) begin
   if(reg_write_en) begin
    reg_array[reg_write_dest] <= reg_write_data;
   end
 end


 assign reg_read_data_1 = reg_array[reg_read_addr_1];
 assign reg_read_data_2 = reg_array[reg_read_addr_2];


endmodule
```

ALU Control

```
`timescale 1ns / 1ps
//fpga4student.com: FPGA projects, Verilog projects,
VHDL projects
// Verilog code for 16-bit RISC processor
// ALU_Control Verilog code
module alu_control( ALU_Cnt, ALUOp, Opcode);
 output reg[2:0] ALU_Cnt;
 input [1:0] ALUOp;
 input [3:0] Opcode;
 wire [5:0] ALUControlIn;
 assign ALUControlIn = {ALUOp,Opcode};
 always @(ALUControlIn)
 casex (ALUControlIn)
   6'b10xxxx: ALU_Cnt=3'b000;
   6'b01xxxx: ALU_Cnt=3'b001;
   6'b000010: ALU_Cnt=3'b000;
   6'b000011: ALU_Cnt=3'b001;
   6'b000100: ALU_Cnt=3'b010;
   6'b000101: ALU_Cnt=3'b011;
   6'b000110: ALU_Cnt=3'b100;
   6'b000111: ALU_Cnt=3'b101;
   6'b001000: ALU_Cnt=3'b110;
   6'b001001: ALU_Cnt=3'b111;
  default: ALU_Cnt=3'b000;
  endcase
endmodule
```