



Lab 3: Simple ALU

Chien-Ming Wu

TSRI, NARL, Taiwan, R.O.C.

Lan-Da Van

Department of Computer Science

National Yang Ming Chiao Tung University

Taiwan, R.O.C.

Fall, 2021



Lab 3 Goal: Simple ALU

Lab 3

- ◆ In this lab, you will practice Verilog to design one simple ALU design.
- ◆ The lab file submission deadline is on 10/18 by 6:00pm.

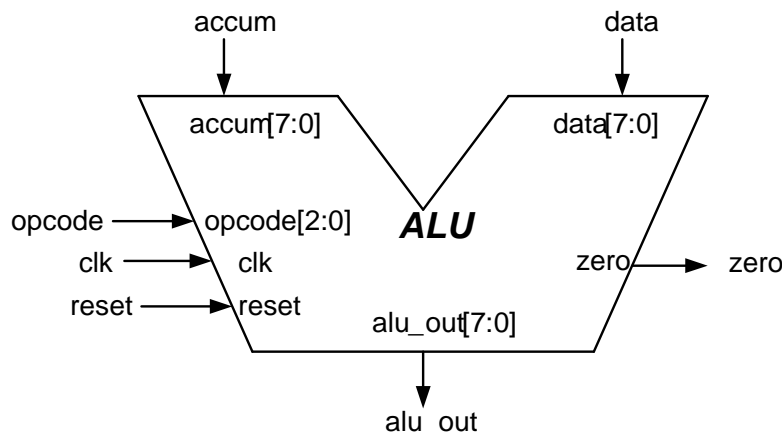




Simple ALU

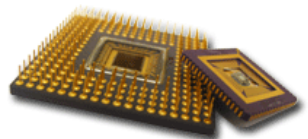
Lab 3

- ◆ 所有輸入及輸出（除了「**zero**」訊號以外）均需同步於clock的正緣（rising edge）。
- ◆ 同步reset架構。當reset為1時表示reset啟動，此時alu_out訊號輸出為0。
- ◆ accum、data及alu_out訊號的數值使用2補數表示。
- ◆ 當accum輸入為0時，zero訊號輸出為1；反之當accum輸入不為0時，zero訊號輸出為0。並且zero訊號不需理會reset訊號的動作。
- ◆ 當opcode輸入為X(**unknown**)時，其alu_out訊號輸出為**0**。



opcode	ALU operation	
000	Pass accum	
001	accum + data	(add)
010	accum - data	(subtraction)
011	accum AND data	(bit-wise AND)
100	accum XOR data	(bit-wise XOR)
101	ABS(accum)	(absolute value)
110	MUL	(multiplication)
111	Pass data	

1. absolute value時，使用accum[7]當作signed bit
2. MUL is for sign multiplication





The Testbench of the ALU Module

Lab 3

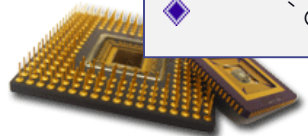
```

◆ wire [7:0] alu_out;
◆ reg [7:0] data, accum;
◆ reg [2:0] opcode;
◆ wire [7:0] mask;
◆ reg clk, reset;

◆
◆ parameter ranseed = 8; // Seed for the random function
◆                          // Modify the seed for different inputs
◆
◆ // Instantiate the ALU. Named mapping allows the designer to have
◆ freedom with the order of port declarations
◆ alu    alu1 (.alu_out(alu_out), .zero(zero), //outputs from ALU
◆             .opcode(opcode), .data(data & mask), //inputs to ALU
◆             .accum(accum & mask), .clk(clk), .reset(reset));

◆ // Define mnemonics to represent opcodes
◆ `define PASSA 3'b000
◆ `define ADD   3'b001
◆ ...
◆ `define PASSD 3'b111

```





The Testbench of the ALU Module

Lab 3

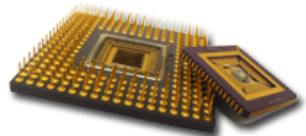
```
◆ // Define a safe delay between each strobing of the ALU
  inputs/outputs

◆ `define strobe          20

◆ // To perform a 4-bit multiplication, set the first 4 bits of the
  input to 4'b0000 when opcode is 3'b110 (Multiplication)

◆ assign mask = (opcode == 3'b110)? 8'h0f: 8'hff;

◆ // Clock generate
◆ initial    clk = 0;
◆ always #(`strobe/2) clk = ~clk;
```





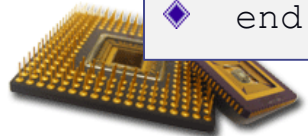
The Testbench of the ALU Module

Lab 3

```

◆ // pattern generate
◆ initial begin
◆     // SET UP THE OUTPUT FORMAT FOR THE TEXT DISPLAY
◆     $display("\t\t\t\t\t INPUTS\t\t\t\t\t REAL\t\t\t\t\t OUTPUT\t\t\t\t\t \n");
◆     ...
◆     reset = 0;
◆     # `strobe;
◆     accum = 8'h37;
◆     data = 8'hD6;
◆     reset = 1; //reset the ALU
◆     # `strobe;
◆     reset = 0;
◆     #(`strobe/4) opcode = 3'b001; // Set operation code
◆
◆     // APPLY STIMULUS TO THE INPUT PINS
◆     accum = $random % ranseed; //Set inputs to the ALU
◆     data = $random % ranseed;
◆     //Wait for ALU to process inputs
◆     #(`strobe/2) check_outputs; //call a task to verify outputs
◆ end

```





The Testbench of the ALU Module

Lab 3

```

// SUBROUTINES TO DISPLAY THE ALU OUTPUTS
task check_outputs;
    casez (opcode)
        `PASSA : begin
            $display("PASS ACCUM OPERATION:",
                    "      %b      %b  %b  |  %b      %b",
                    opcode, data, accum, alu_out, zero);
        end
        `ADD : begin
            $display("ADD OPERATION      :",
                    "      %b      %b  %b  |  %b      %b",
                    opcode, data, accum, alu_out, zero);
        end
        ...
        default : begin
            $display("UNKNOWN OPERATION  :",
                    "      %b      %b  %b  |  %b      %b",
                    opcode, data, accum, alu_out, zero);
        end
    endcase
endtask

```



Lab 3 Demo Guide

Lab 3

- ◆ You can download the sample testbench file `alu_test.v` from E3, and create a Vivado project for it.
- ◆ You should upload your lab3 solution to E3 before the deadline.
- ◆ During the demo time, TA will ask you to modify the testbench to show different results.
 - You can download your code from E3 during demo.

