1 Project 6: Verilog Code Development – MIPS ALU Controller Design (CS 7381 only)

10 points

Students enrolled in CS 7381 (graduate version of the course) have an additional Verilog programming assignment.

For this exercise, you will write a Verilog code program to implement the MIPS ALU Control Unit. You will test your Control Unit module using the testbench provided in the assignment.

In the previous Verilog assignment, you modified the code for a MIPS ALU. One of the inputs to the ALU was the ALU Control signal vector **ALUctl**. The **ALUctl** vector is output from the ALU Control Unit, and its value is based on the **ALUOp** signal vector from the MIPS main Control Unit and **funct** field of the MIPS instruction. The following table shows how these signals are mapped:

opcode	ALUOp	Operation	funct	ALU function	ALUctl
lw	00	load word	XXXXXX	add	0010
beq	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
		subtract	100010	subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111
		NOR	100111	NOR	1100
		XOR	100110	Exclusive-Or	1110

1. Please download the following Verilog file from the assignment page:
   a. **S23_ALU_control_tb.v** - the testbench for testing your ALU Control Unit
2. Using the function mapping table above, design your ALU Control Unit so that it produces the correct ALU Control signal vector given the inputs **ALUOp** and **funct**.
   a. Save the program as a *.v file – use the first initial of your first name and the first 4 letters of your last name, then the number 2 (to distinguish from your code for Project 5). For example, my file submission name would be **tmani2.v.**
   b. Test your ALU Control Unit using the testbench provided in Step 1.
3. **Please include the following for your homework submission:**
   a. Your ALU Control Unit Verilog file – **submit the actual *.v file so that the grader can run them.**
   b. Your testbench results – this can be a copy of the results on a Word document.
   c. PLEASE MAKE SURE THAT YOUR NAME APPEARS ON ALL SUBMITTED ITEMS FOR PROPER CREDIT

2 SOLUTION

2.1 Verilog code: ALU Control Unit

An example of the Verilog code solution is the following:

```
//============================================================
// tmani2.v
// T. Manikas    2022 Dec 29
//
// ALU Control Unit for MIPS processor datapath
//
// translates ALUOP and funct into ALUctl signal
// for corresponding ALU operation
//
//============================================================

`timescale 1ns / 1ps

module ALU_control(ALUctl, ALUOp, funct);
   input [1:0] ALUOp;          // ALUOp (from MIPS Control Unit
   input [5:0] funct;           // funct (last 6 bits of instruction)
   output [3:0] ALUctl;         // ALUctl signal for ALU
   reg [3:0]     ALUctl;


   always @(ALUOp or funct)
     begin
      case (ALUOp)
        2'b00:  ALUctl = 4'b0010;         // lw
        2'b01:  ALUctl = 4'b0110;            // beq
        2'b10: case (funct)                  // R-type
             6'b100000: ALUctl = 4'b0010;       // add
             6'b100010: ALUctl = 4'b0110;       // sub
             6'b100100: ALUctl = 4'b0000;       // and
             6'b100101: ALUctl = 4'b0001;       // or
             6'b101010: ALUctl = 4'b0111;       // slt
             6'b100111: ALUctl = 4'b1100;       // nor
             6'b100110: ALUctl = 4'b1110;       // xor
             default: ALUctl = 4'b0000;
           endcase // case (funct)

        default:  ALUctl = 4'b0000;
      endcase // case (ALUOp)
     end

endmodule // ALU_control
```

2.2 Testbench results

```
 0 ALUOp=00 funct=000000 ALUctl=0010
 4 ALUOp=01 funct=000000 ALUctl=0110
 6 ALUOp=10 funct=100000 ALUctl=0010
 8 ALUOp=10 funct=100010 ALUctl=0110
10 ALUOp=10 funct=100100 ALUctl=0000
12 ALUOp=10 funct=100101 ALUctl=0001
14 ALUOp=10 funct=101010 ALUctl=0111
16 ALUOp=10 funct=100111 ALUctl=1100
18 ALUOp=10 funct=100110 ALUctl=1110
```