**实验八 单周期的CPU设计**

一、实验目的及环境

1、深入了解单周期CPU中指令控制器的结构和工作原理。

2、学习使用Verilog HDL设计实现单周期CPU中指令控制器。

3、装有vivado的计算机 1台

4、EGO1开发板 1块

二、实验目标及任务

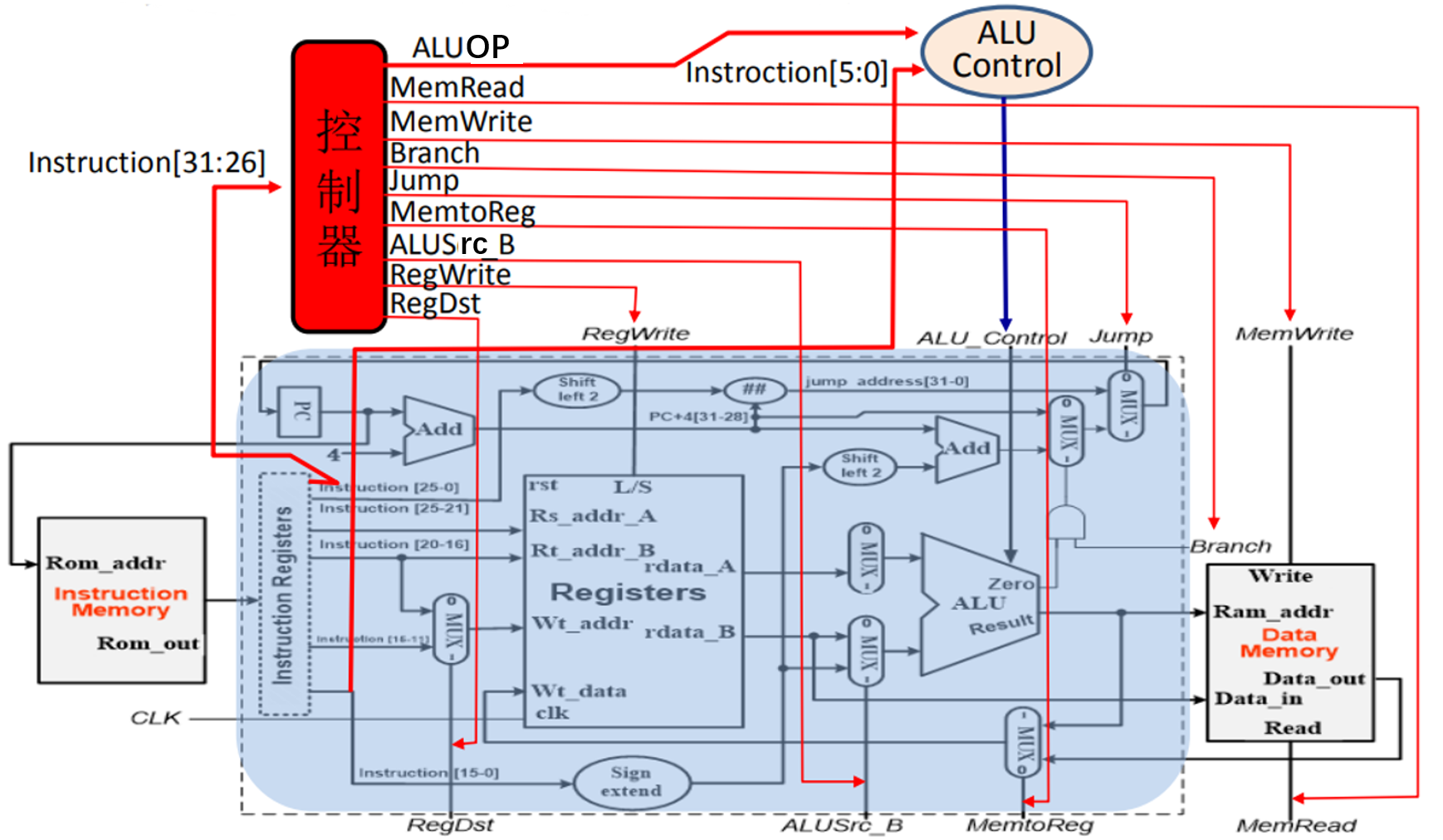
1、设计和实现单周期CPU中指令控制器的结构并且进行功能仿真。

三、实验过程及记录



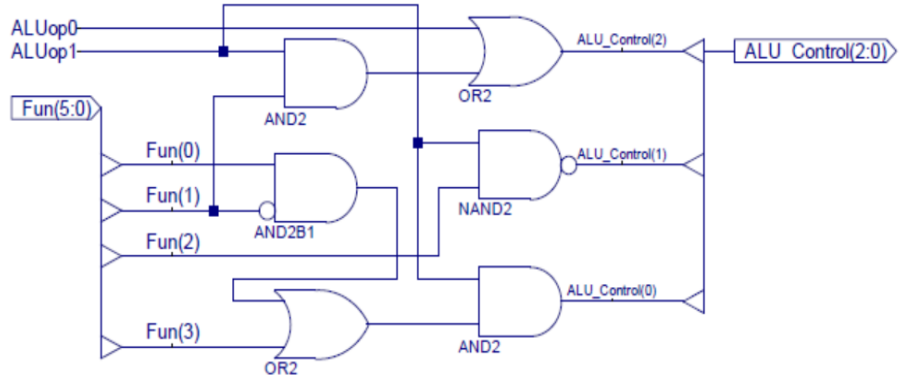
本节重点介绍实验的具体过程，包括：代码设计层次结构图及说明、源代码（包括注释）、PC机上进行的关键步骤截图及说明、调试过程等，这部分的内容应当与实际操作过程类似即可（简单明了）。

①电路原理图:

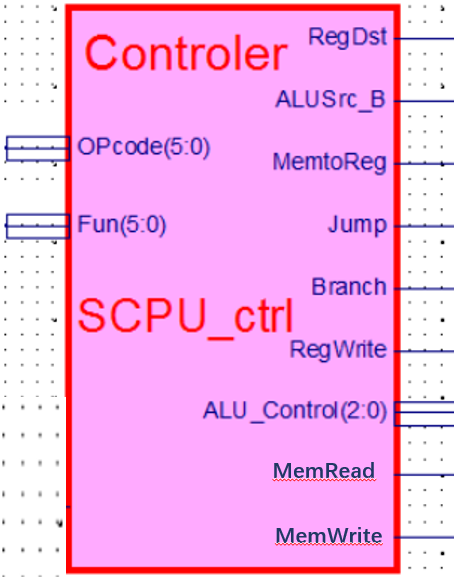


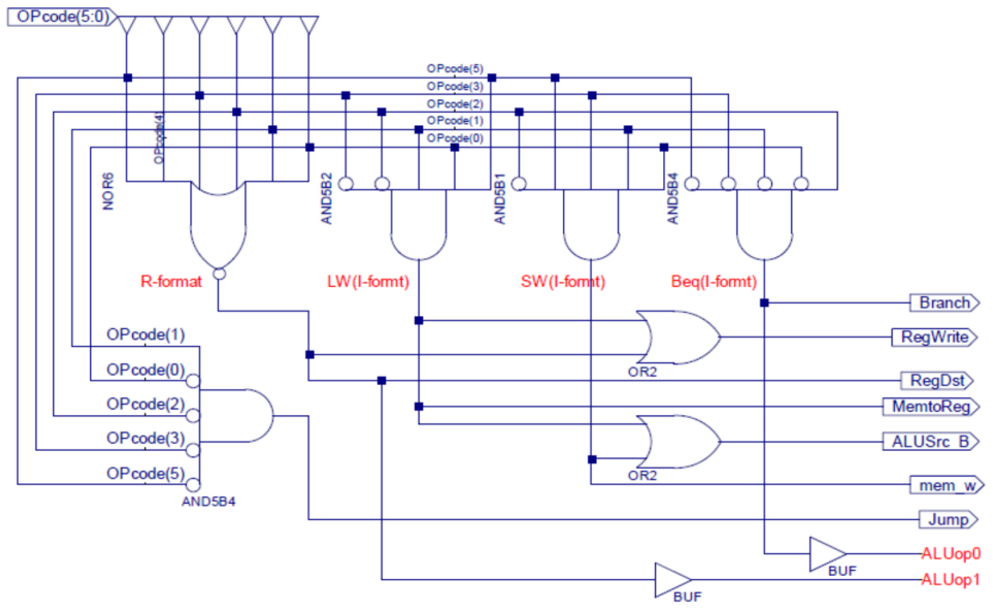


由实验原理图以及上述控制输出的表格可以得知，我们首先需要实现ALU\_Control模块，该模块通过其中的控制信号ALUOP与指令机器码的低6位instruction[5:0]联合起来，生成运算器运算类型相对应的控制信号ALU\_Control。



根据上面的实验原理图我们可以实现ALU\_Control模块。之后我们需要设计CPUControl模块实现Cpu的控制。





由上述两个电路原理图可以很容易知道CpuControl模块的编写。最后通过顶层Top文件完成CPU的设计。

②源代码:

ALU\_Control:

`timescale 1ns / 1ps

*// ALU 控制模块*

module ALU\_Ctrl (

    input      [1:0] ALUOp,

    input      [5:0] Func,

    output reg [2:0] ALU\_Control

);

    and and1 (temp1, ALUOp[1], Func[1]);

    and and2 (temp2, Func[0], ~Func[1]);

    or or1 (temp3, Func[3], temp2);

    or or2 (temp4, ALUOp[0], temp1);

    nand nand1 (temp5, ALUOp[1], Func[2]);

    and and3 (temp6, ALUOp[1], temp3);

*// ALU 控制信号输出*

    always @(\*) begin

        ALU\_Control <= {temp4, temp5, temp6};

    end

endmodule

SCPU\_Control:

`timescale 1ns / 1ps

*// CPU 控制模块*

module SCPU\_Ctrl (

    input      [5:0] OPcode,       *// OPcode*

    input      [5:0] Func,         *// Function*

    output reg       RegDst,

    output reg       ALUSrc\_B,

    output reg       MemtoReg,

    output reg       Jump,

    output reg       Branch,

    output reg       RegWrite,

    output reg [2:0] ALU\_Control,

    output reg       MemRead,

    output reg       MemWrite

);

    wire [2:0] ALU\_Control\_t;

*// 控制信号生成*

    nor nor6 (RFormat, OPcode[5], OPcode[4], OPcode[3], OPcode[2], OPcode[1], OPcode[0]);

    and and5b1 (LFormat\_LW, OPcode[5], ~OPcode[3], ~OPcode[2], OPcode[1], OPcode[0]);

    and and5b2 (LFormat\_SW, OPcode[5], OPcode[3], ~OPcode[2], OPcode[1], OPcode[0]);

    and and5b3 (LFormat\_Beq, ~OPcode[5], ~OPcode[3], OPcode[2], ~OPcode[1], ~OPcode[0]);

    and and5b4 (JFormat, ~OPcode[5], ~OPcode[3], ~OPcode[2], OPcode[1], ~OPcode[0]);

*// ALU 控制*

    ALU\_Ctrl u\_ALU\_Ctrl (

        .ALUOp      ({RFormat, LFormat\_Beq}),

        .Func       (Func),

        .ALU\_Control(ALU\_Control\_t)

    );

*// 控制信号输出*

    always @(\*) begin

        RegDst <= RFormat;

        ALUSrc\_B <= LFormat\_LW | LFormat\_SW;

        MemtoReg <= LFormat\_LW;

        Jump <= JFormat;

        Branch <= LFormat\_Beq;

        RegWrite <= RFormat | LFormat\_LW;

        MemRead <= LFormat\_LW;

        MemWrite <= LFormat\_SW;

        ALU\_Control <= ALU\_Control\_t;

    end

*// 初始化*

    initial begin

        RegDst <= 0;

        ALUSrc\_B <= 0;

        MemtoReg <= 0;

        Jump <= 0;

        Branch <= 0;

        RegWrite <= 0;

        ALU\_Control <= 0;

        MemRead <= 0;

        MemWrite <= 0;

    end

endmodule

Top顶层模块:

`timescale 1ns / 1ps

module top (

    input  [15:0] Instrument\_Select,

    output        RegDst,

    output        ALUSrc\_B,

    output        MemtoReg,

    output        Jump,

    output        Branch,

    output        RegWrite,

    output [ 2:0] ALU\_Control,

    output        MemRead,

    output        MemWrite

);

*// mem用于读取指令*

    reg [11:0] memory [0:1];

*// SCPU\_Ctrl 模块输入*

    reg [ 5:0] OPcode;

    reg [ 5:0] Func;

*// SCPU\_Ctrl 模块实例化*

    SCPU\_Ctrl u\_SCPU\_Ctrl (

        .OPcode     (OPcode),

        .Func       (Func),

        .RegDst     (RegDst),

        .ALUSrc\_B   (ALUSrc\_B),

        .MemtoReg   (MemtoReg),

        .Jump       (Jump),

        .Branch     (Branch),

        .RegWrite   (RegWrite),

        .ALU\_Control(ALU\_Control),

        .MemRead    (MemRead),

        .MemWrite   (MemWrite)

    );

*// 读取指令到 memory*

    initial begin

        $readmemb("Is.txt", memory);

    end

*// 指定执行第几条指令*

    reg [4:0] n = 0;

    always @(Instrument\_Select) begin

        case (Instrument\_Select)

            1 << 0: n <= 0;

            1 << 1: n <= 1;

        endcase

        OPcode <= memory[n][11:6];

        Func   <= memory[n][5:0];

    end

endmodule

四、实验结果分析

1.这里应给出相应的实验结果。分析应有条理，要求采用规范的书面语。

2.每个实验都需要做模拟仿真，需要对仿真波形进行简单的文字说明。

3.对下载到开发板上的图片结果做分析说明。

4.原则上要求使用图片与文字结合的形式说明，因为word和PDF文档不支持视频，所以请不要使用视频文件。

5.图片请在垂直方向，不要横向。不要用很大的图片（不要超过1M），请先做裁剪操作。

①激励文件:

`timescale 1ns / 1ps

module tb\_top;

    parameter XH = 2022217587;

*// top 模块输入*

    reg  [15:0] Instrument\_Select;

*// top 模块输出*

    wire        RegDst;

    wire        ALUSrc\_B;

    wire        MemtoReg;

    wire        Jump;

    wire        Branch;

    wire        RegWrite;

    wire [ 2:0] ALU\_Control;

    wire        MemRead;

    wire        MemWrite;

*// top 模块实例化*

    top u\_top (

        .Instrument\_Select(Instrument\_Select),

        .RegDst           (RegDst),

        .ALUSrc\_B         (ALUSrc\_B),

        .MemtoReg         (MemtoReg),

        .Jump             (Jump),

        .Branch           (Branch),

        .RegWrite         (RegWrite),

        .ALU\_Control      (ALU\_Control),

        .MemRead          (MemRead),

        .MemWrite         (MemWrite)

    );

*// 依次送入第 1 ~ 16 条指令*

    initial begin

        Instrument\_Select <= 16'b0000000000000001;

        #XH;

        Instrument\_Select <= 16'b0000000000000010;

        #XH;

        Instrument\_Select <= 16'b0000000000000100;

        #XH;

        Instrument\_Select <= 16'b0000000000001000;

        #XH;

        Instrument\_Select <= 16'b0000000000010000;

        #XH;

        Instrument\_Select <= 16'b0000000000100000;

        #XH;

        Instrument\_Select <= 16'b0000000001000000;

        #XH;

        Instrument\_Select <= 16'b0000000010000000;

        #XH;

        Instrument\_Select <= 16'b0000000100000000;

        #XH;

        Instrument\_Select <= 16'b0000001000000000;

        #XH;

        Instrument\_Select <= 16'b0000010000000000;

        #XH;

        Instrument\_Select <= 16'b0000100000000000;

        #XH;

        Instrument\_Select <= 16'b0001000000000000;

        #XH;

        Instrument\_Select <= 16'b0010000000000000;

        #XH;

        Instrument\_Select <= 16'b0100000000000000;

        #XH;

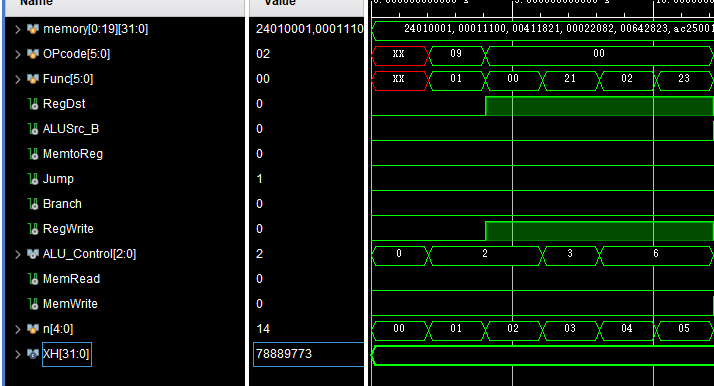
        Instrument\_Select <= 16'b1000000000000000;

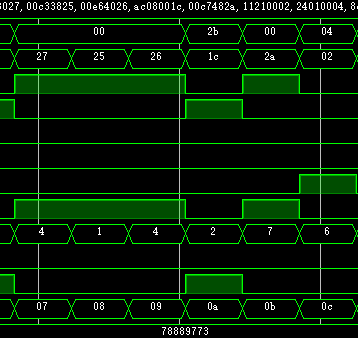
        #XH;

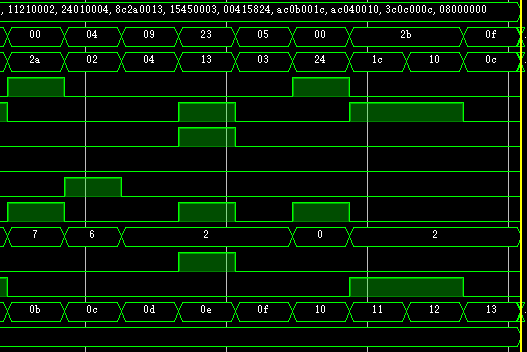
    end

endmodule

②仿真波形:

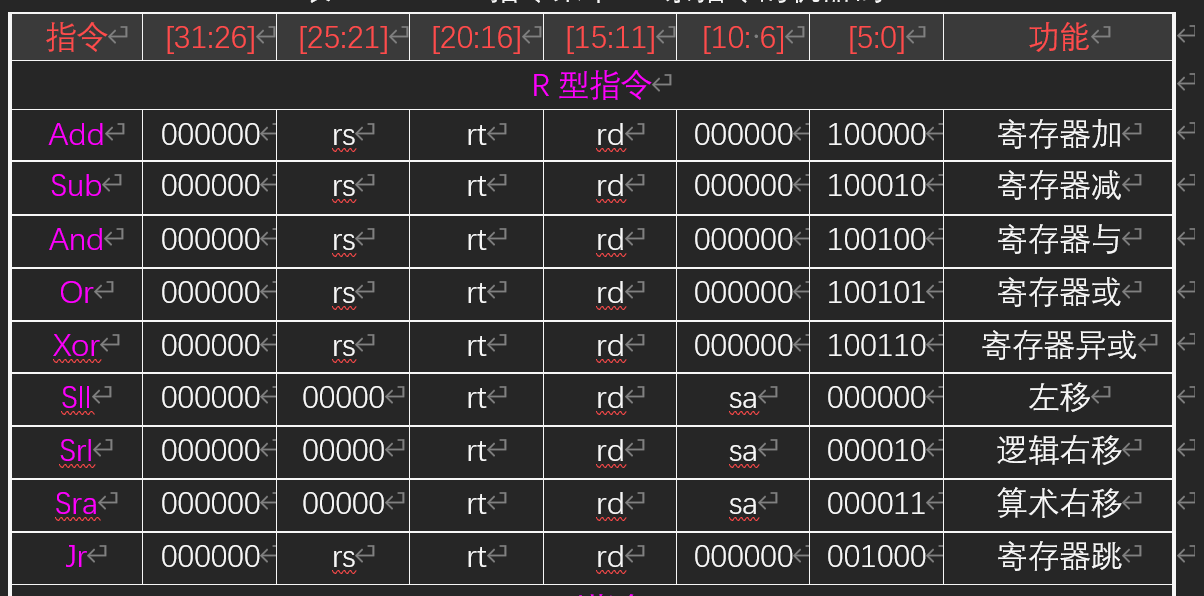






③结果解释:

在这个激励文件的仿真波形中，使用了学号作为XH变量，依次送入指令内容，ALU\_Control的输出结果与下面的表格输出内容一致。



④Instruction文件的编写:

000000100000

000000100010

000000100100

000000100101

000000100110

000000000000

000000000010

000000000011

000000001000

该文件中的部分使用上述图片中的指令形式，通过Top文件进行依次读取并在开发板中进行显示。

⑤约束文件:

*# /////////////////////////////系统时钟和复位////////////////////////////////////*

*# set\_property -dict {PACKAGE\_PIN P17 IOSTANDARD LVCMOS33} [get\_ports sys\_clk\_in ]*

*# set\_property -dict {PACKAGE\_PIN P15 IOSTANDARD LVCMOS33} [get\_ports sys\_rst\_n  ]*

*# /////////////////////////////////////5个按键//////////////////////////////////////*

*# set\_property -dict {PACKAGE\_PIN R11 IOSTANDARD LVCMOS33} [get\_ports {btn\_pin[0]}]*

*# set\_property -dict {PACKAGE\_PIN R17 IOSTANDARD LVCMOS33} [get\_ports {btn\_pin[1]}]*

*# set\_property -dict {PACKAGE\_PIN R15 IOSTANDARD LVCMOS33} [get\_ports {btn\_pin[2]}]*

*# set\_property -dict {PACKAGE\_PIN V1  IOSTANDARD LVCMOS33} [get\_ports {btn\_pin[3]}]*

*# set\_property -dict {PACKAGE\_PIN U4  IOSTANDARD LVCMOS33} [get\_ports {btn\_pin[4]}]*

*# ///////////////////////////////////拨码开关sw0~sw7////////////////////////////////*

set\_property -dict {PACKAGE\_PIN P5 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[15]}]

set\_property -dict {PACKAGE\_PIN P4 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[14]}]

set\_property -dict {PACKAGE\_PIN P3 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[13]}]

set\_property -dict {PACKAGE\_PIN P2 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[12]}]

set\_property -dict {PACKAGE\_PIN R2 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[11]}]

set\_property -dict {PACKAGE\_PIN M4 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[10]}]

set\_property -dict {PACKAGE\_PIN N4 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[9]}]

set\_property -dict {PACKAGE\_PIN R1 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[8]}]

*# //////////////////////////////////拨码开关sw8~sw15/////////////////////////////*

set\_property -dict {PACKAGE\_PIN U3 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[7]}]

set\_property -dict {PACKAGE\_PIN U2 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[6]}]

set\_property -dict {PACKAGE\_PIN V2 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[5]}]

set\_property -dict {PACKAGE\_PIN V5 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[4]}]

set\_property -dict {PACKAGE\_PIN V4 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[3]}]

set\_property -dict {PACKAGE\_PIN R3 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[2]}]

set\_property -dict {PACKAGE\_PIN T3 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[1]}]

set\_property -dict {PACKAGE\_PIN T5 IOSTANDARD LVCMOS33} [get\_ports {Instrument\_Select[0]}]

*# //////////////////////////////////////LED0~LED15////////////////////////////////*

set\_property -dict {PACKAGE\_PIN F6 IOSTANDARD LVCMOS33} [get\_ports {RegDst}]

set\_property -dict {PACKAGE\_PIN G4 IOSTANDARD LVCMOS33} [get\_ports {ALUSrc\_B}]

set\_property -dict {PACKAGE\_PIN G3 IOSTANDARD LVCMOS33} [get\_ports {MemtoReg}]

set\_property -dict {PACKAGE\_PIN J4 IOSTANDARD LVCMOS33} [get\_ports {Jump}]

set\_property -dict {PACKAGE\_PIN H4 IOSTANDARD LVCMOS33} [get\_ports {Branch}]

set\_property -dict {PACKAGE\_PIN J3 IOSTANDARD LVCMOS33} [get\_ports {RegWrite}]

set\_property -dict {PACKAGE\_PIN J2 IOSTANDARD LVCMOS33} [get\_ports {MemRead}]

set\_property -dict {PACKAGE\_PIN K2 IOSTANDARD LVCMOS33} [get\_ports {MemWrite}]

set\_property -dict {PACKAGE\_PIN K1 IOSTANDARD LVCMOS33} [get\_ports {ALU\_Control[2]}]

set\_property -dict {PACKAGE\_PIN H6 IOSTANDARD LVCMOS33} [get\_ports {ALU\_Control[1]}]

set\_property -dict {PACKAGE\_PIN H5 IOSTANDARD LVCMOS33} [get\_ports {ALU\_Control[0]}]

*# set\_property -dict {PACKAGE\_PIN J5 IOSTANDARD LVCMOS33} [get\_ports {led\_pin[11]}]*

*# set\_property -dict {PACKAGE\_PIN K6 IOSTANDARD LVCMOS33} [get\_ports {led\_pin[12]}]*

*# set\_property -dict {PACKAGE\_PIN L1 IOSTANDARD LVCMOS33} [get\_ports {led\_pin[13]}]*

*# set\_property -dict {PACKAGE\_PIN M1 IOSTANDARD LVCMOS33} [get\_ports {led\_pin[14]}]*

*# set\_property -dict {PACKAGE\_PIN K3 IOSTANDARD LVCMOS33} [get\_ports {led\_pin[15]}]*

*# ///////////////////////////////8个数码管位选信号/////////////////////////////////*

*# set\_property -dict {PACKAGE\_PIN G2 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[0]}]*

*# set\_property -dict {PACKAGE\_PIN C2 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[1]}]*

*# set\_property -dict {PACKAGE\_PIN C1 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[2]}]*

*# set\_property -dict {PACKAGE\_PIN H1 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[3]}]*

*# set\_property -dict {PACKAGE\_PIN G1 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[4]}]*

*# set\_property -dict {PACKAGE\_PIN F1 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[5]}]*

*# set\_property -dict {PACKAGE\_PIN E1 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[6]}]*

*# set\_property -dict {PACKAGE\_PIN G6 IOSTANDARD LVCMOS33} [get\_ports {seg\_cs\_pin[7]}]*

*# ///////////////////////////////////数码管段选信号//////////////////////////////////*

*# set\_property -dict {PACKAGE\_PIN B4 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[0]}]*

*# set\_property -dict {PACKAGE\_PIN A4 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[1]}]*

*# set\_property -dict {PACKAGE\_PIN A3 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[2]}]*

*# set\_property -dict {PACKAGE\_PIN B1 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[3]}]*

*# set\_property -dict {PACKAGE\_PIN A1 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[4]}]*

*# set\_property -dict {PACKAGE\_PIN B3 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[5]}]*

*# set\_property -dict {PACKAGE\_PIN B2 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[6]}]*

*# set\_property -dict {PACKAGE\_PIN D5 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_0\_pin[7]}]*

*# set\_property -dict {PACKAGE\_PIN D4 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[0]}]*

*# set\_property -dict {PACKAGE\_PIN E3 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[1]}]*

*# set\_property -dict {PACKAGE\_PIN D3 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[2]}]*

*# set\_property -dict {PACKAGE\_PIN F4 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[3]}]*

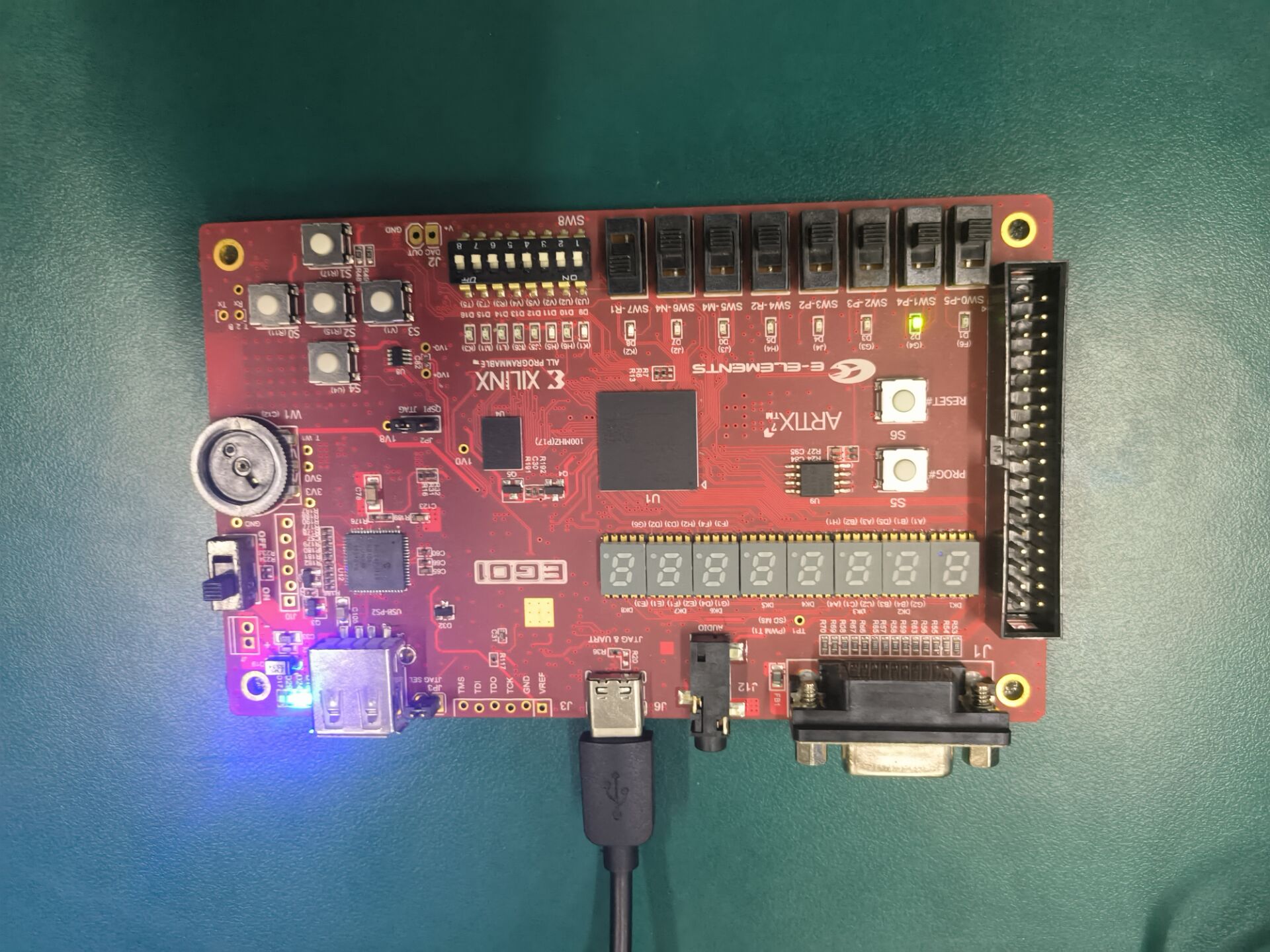
*# set\_property -dict {PACKAGE\_PIN F3 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[4]}]*

*# set\_property -dict {PACKAGE\_PIN E2 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[5]}]*

*# set\_property -dict {PACKAGE\_PIN D2 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[6]}]*

*# set\_property -dict {PACKAGE\_PIN H2 IOSTANDARD LVCMOS33} [get\_ports {seg\_data\_1\_pin[7]}]*

⑥开发板运行结果:



根据图片不难发现实验开发板与实验所编写代码运行结果一致。

五、遇到的问题与实验心得体会

简要地叙述一下实验过程中的感受，以及其他的问题描述和自己的感想。特别是实验中遇到的困难，最后如何解决的。在用verilog代码写程序时遇到语法或其他错误，如何修改解决的。

在计算机组成实验中设计单周期CPU顶层模块是一项重要的任务。在本次实验中，我完成了这两个工作。

在本次实验中，我首先实现了ALU——Control模块，这个模块用于输出运算类型相对应的控制信号，之后进行CPU模块主体部分的编写。 在实验中，花费更多时间的是对整个CPU结构的理解，在CPU中，需要有指令寄存器、寄存器堆、译码和控制单元、ALU，还有指令的执行过程：取指、析指、执行、访存。对于CPU和指令的执行都有了更加深入的理解。在理解了CPU的结构过后，在最后的实现上，也没有花费太多的时间，只需要将其中的模块一一实现，然后使用数据线或控制线连接各个模块，就可以得到一个相对简单的CPU。