单周期 CPU 实验报告

1. 控制信号设计（ControllerUnit.v）

- RegDst：控制寄存器写入端地址。

  设计 Mux 进行选择：

  00：rd，R 型指令

  01：rt，lw、ori、lui 等

  10：jal 函数调用

- RegWrite：是否将数据写入寄存器堆

  bool 类型：

  0：失效

  1：lw、lui 等

- ALUSrc：选择 ALU B 端的输入

  设计 Mux 进行选择：

  00：rt，R、beq 等

  01：imm，lw、sw、ori 等

  10：lui 指令

- PCSrc：PC 的输入源

  设计 Mux 进行选择：

  00：jr 寄存器跳转

  01：beq 分支指令

  10：jal 函数调用

  11：不需要跳转

- PCJumpEnable：根据 PCSrc 进行赋值，除 11 情况赋 0，其余均赋 1
- MemRead：数据存储器 DM 是否读数据

  bool 类型：

  0：失效

  1：lw 指令

- MemWrite：数据存储器 DM 是否写数据

  bool 类型：

  0：失效

  1：sw 指令

- MemtoReg：寄存器写入数据来自 ALU 输出还是 DM 输出

  设计 Mux 进行选择：

  00：ALU

  01：DM

  10：PC+4

2. 具体实现

```
// RegDst
// 00:Rt;01:Rd;10:$ra
if(Op == 6'b000000 && Func != 6'b001100 && Func != 6'b001000) // no syscall or jr
    RegDst <= 2'b01; // R
else if(Op == 6'b100011 || Op == 6'b001101 || Op == 6'b001111)
    RegDst <= 2'b00; // lw、ori、lui
```

```
        else if(Op == 6'b000011) // jal
            RegDst <= 2'b10;
        else
            RegDst <= 2'b11;
        // RegWrite: lw、ori、jal、lui、R
        if(Op == 6'b000000 || Op == 6'b100011 || Op == 6'b001101 || Op == 6'b000011 || Op ==
6'b001111)
            RegWrite <= 1;
        else
            RegWrite <= 0;
        // ALUSrc
        // 00:GR[rt];01:SignExtended imm;10:lui;11:0
        if(Op == 6'b000000 || Op == 6'b000100)
            ALUSrc <= 2'b00; // R、beq
        else if(Op == 6'b100011 || Op == 6'b101011 || Op == 6'b001101)
            ALUSrc <= 2'b01; // lw、sw、ori
        else if(Op == 6'b001111)
            ALUSrc <= 2'b10; // lui
        else // if(Op == 6'b000011)
            ALUSrc <= 2'b11; // jal and others
        // PCSrc & PCJumpEnabled
        // 00:jr;01:beq;10:jal
        if(Op == 6'b000000 && Func == 6'b001000) // jr
        begin
            PCJumpEnabled <= 1;
            PCSrc <= 2'b00;
        end
        else if(Op == 6'b000100) // beq
        begin
            PCJumpEnabled <= 1; // need to & ALUResult == zero
            PCSrc <= 2'b01;
        end
        else if(Op == 6'b000011) // jal
        begin
            PCJumpEnabled <= 1;
            PCSrc <= 2'b10;
        end
        else
        begin
            PCJumpEnabled <= 0;
            PCSrc <= 2'b11;
        end
        // MemRead
        if(Op == 6'b100011)// lw
```

```
        MemRead <= 1;
else
        MemRead <= 0;
// MemWrite
if(Op == 6'b101011) // sw
        MemWrite <= 1;
else
        MemWrite <= 0;
// MemToReg
// Reg data source
// 00:ALU;01:Mem;10:PC + 4;11 other(ALU into Regs)
if(Op == 6'b000000)
        MemToReg <= 2'b00; // R 型指令
else if(Op == 6'b100011)
        MemToReg <= 2'b01; // lw
else if(Op == 6'b000011)
        MemToReg <= 2'b10;// jal
else
        MemToReg <= 2'b00;
```