

中国科学院大学计算机组成原理实验课

实 验 报 告

学号: 2015K8009929014 姓名: 李云志 专业: 计算机科学与技术

实验序号: 2 实验名称: 单周期 CPU

注 1: 本实验报告请以 PDF 格式提交。文件命名规则: [学号]-PRJ[实验序号]-RPT.pdf, 其中文件名字母大写, 后缀名小写。例如: [2014K8009959088]-PRJ[1]-RPT.pdf
注 2: 实验报告模板以下部分的内容供参考, 可包含但不限定如下条目内容。

一、 逻辑电路结构与仿真波形的截图及说明 (比如关键 RTL 代码段{包含注释})

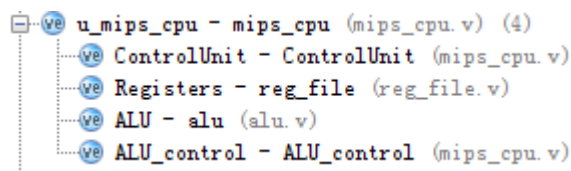
及其对应的逻辑电路结构、相应信号的仿真波形和信号变化的说明等)

1、 单周期 CPU 设计

CPU 模块构成: 根据单周期 CPU 的数据通路以及控制通路等概念,

将 CPU 规划为 5 部分, 分别为 mips_cpu, ControlUnit,

Registers, ALU, ALU_control (如图)



2、 核心代码段及逻辑电路图:

```
PC :
always@(posedge clk)//PC
begin
  if(rst)
    PC_reg<=0;
  else if(Branch&(~Zero))
    PC_reg<=PC_reg+4+Shift_left2;
  else
```

```

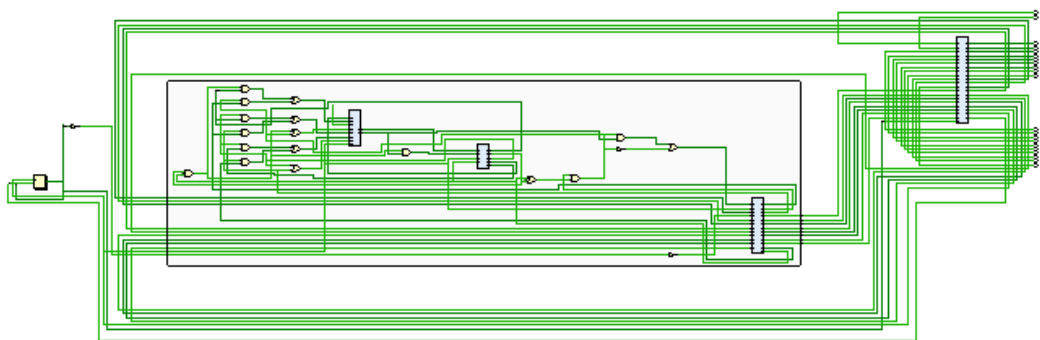
ALU_control :
module ALU_control(
    input [5:0] funct,
    input [1:0] ALUOp,
    output [2:0] ALUop
);
    assign ALUop[0]=(funct[0]|funct[3])&ALUOp[1];
    assign ALUop[1]=(~funct[2])|(~ALUOp[1]);
    assign ALUop[2]=ALUOp[0]|(funct[1]&ALUOp[1]);
endmodule

```

```

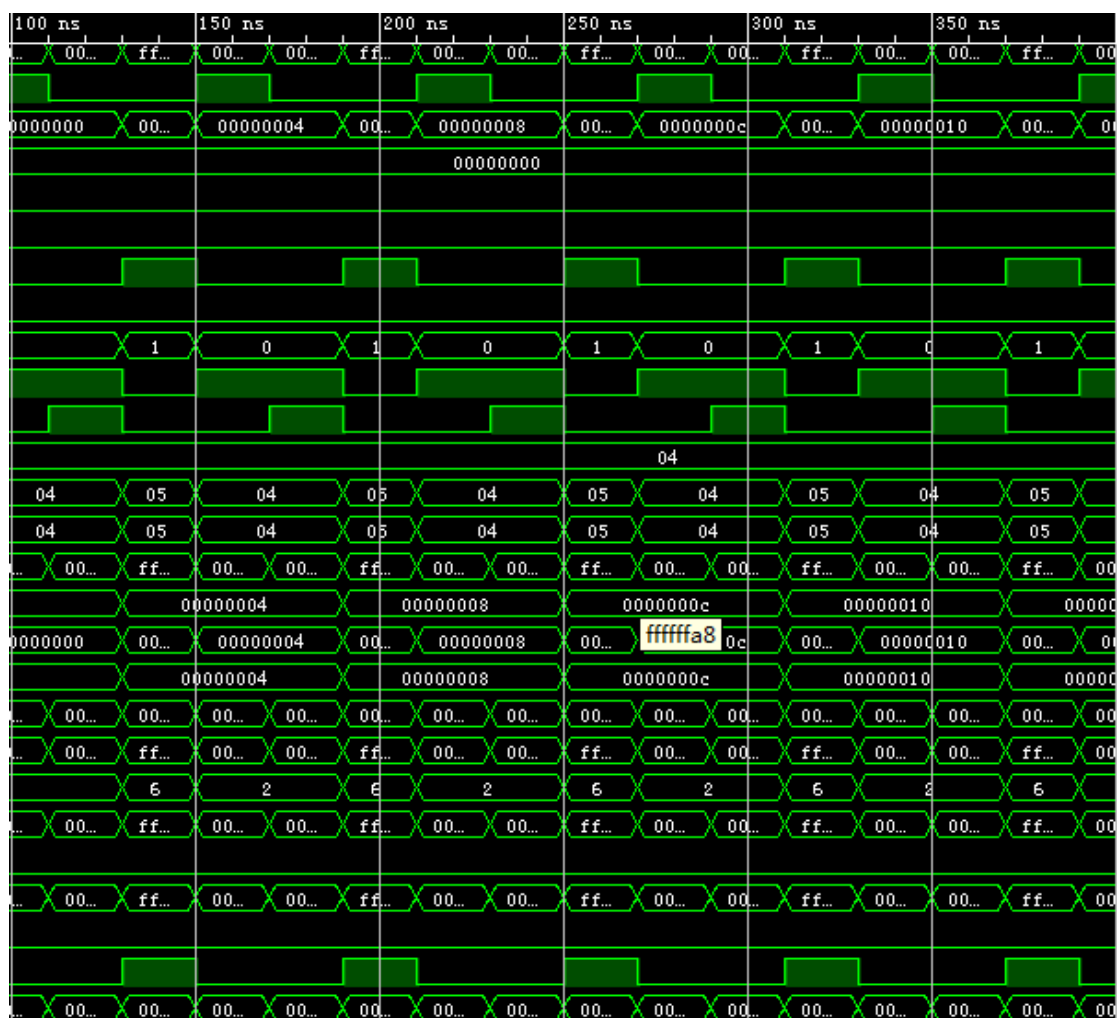
ControlUnit （部分） :
module ControlUnit(
    input [5:0] opcode,
    output reg RegDst,
    output reg Branch,
    output reg MemRead,
    output reg MemtoReg,
    output reg [1:0] ALUOp,
    output reg MemWrite,
    output reg ALUSrc,
    output reg RegWrite
);
    always@(opcode)//控制逻辑
    case(opcode)
        6'b001001:
        begin
            RegDst=0;
            Branch=0;
            MemRead=0;
            MemtoReg=0;
            ALUOp=2'b00;
            MemWrite=0;
            ALUSrc=1;
            RegWrite=1;
        end
    endcase
endmodule

```



3、Memcpy 行为仿真波形及结果

波形:



Ideal_Mem (部分):

[95][31:0]	00000050	Array
[94][31:0]	0000004c	Array
[93][31:0]	00000048	Array
[92][31:0]	00000044	Array
[91][31:0]	00000040	Array
[90][31:0]	0000003c	Array
[89][31:0]	00000038	Array
[88][31:0]	00000034	Array
[87][31:0]	00000030	Array
[86][31:0]	0000002c	Array
[85][31:0]	00000028	Array
[84][31:0]	00000024	Array
[83][31:0]	00000020	Array
[82][31:0]	0000001c	Array
[81][31:0]	00000018	Array
[80][31:0]	00000014	Array
[79][31:0]	00000010	Array
[78][31:0]	0000000c	Array

4、 后期调试（频率）

布局布线后：频率提示至 85M: **Worst Negative Slack (NWS):** 0.273_ns

经过多次试验，发现布局布线策略选择

Performance_NetDelay_Low 所获得的性能较好

5、 思考与总结

- (1) 频率提升方面：分析初始代码的 Timing Report 发现原寄存器堆中的 rst 造成的影响很大，又考虑到实际寄存器堆中并不需要 rst 这一行为，所以将其删除，频率相比于原来的 75M 提升了 10M
- (2) 完成了单周期 CPU 这一项目后，对 CPU 的内部结构、数据通路、控制逻辑等概念均有了更加深入的理解。同时，经历了时序分析这一环节，对于 vivado 工具以及时序分析的方法有了初步认识与体验，为以后的项目完成打下了一定的基础。
- (3) 对 mips 指令有了比较深入的认识，虽然实验只完成了其中

的几条指令，但是对于 mips 指令以及 RSIC 规整、简洁的特点有了更好地认识