计算机组成与结构研讨课
实验报告

2023K8009970008　徐昕妍

实验序号：6　　实验名称：单周期 MIPS CPU 设计

1　实验电路图

1.1　单周期硬布线控制器

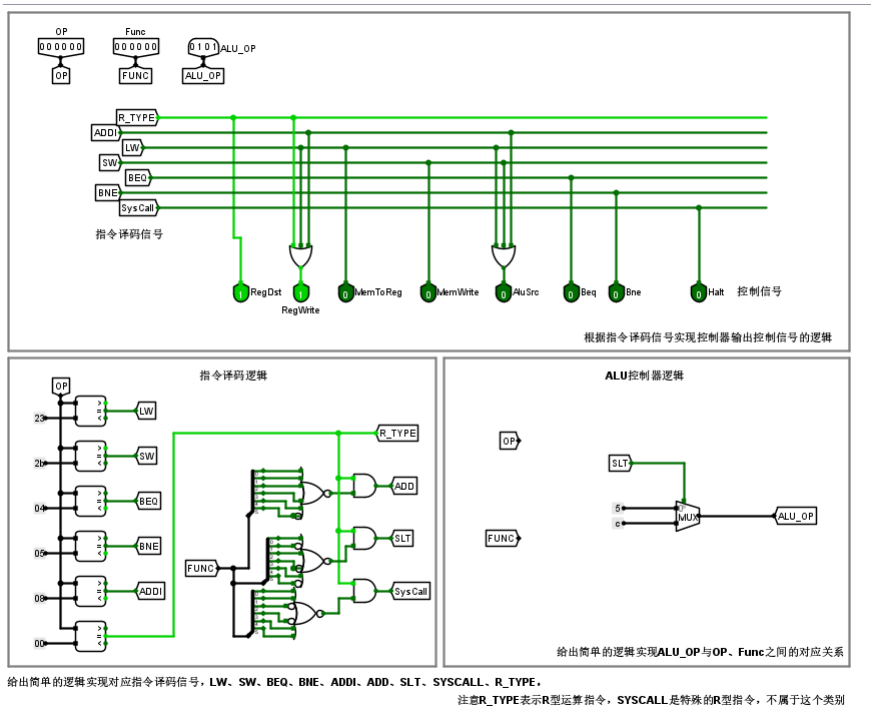

图 1: 单周期硬布线控制器

设计思路：

【指令译码】

该部分主要使用比较器，将 op 字段或者 Func 字段进行比较。其中 00 代表 R 型指令，由 Func 决定具体功能。OP 与 Func 的设定如下表所示：

表 5.8 常用 R 型指令及其 funct 编码

funct	指令助记符	指令功能描述	备注
00	sll rd,rt,shamt	R[rd]=R[rt]<<shamt	逻辑左移，注意 rs 字段未使用
02	srl rd,rt,shamt	R[rd]=R[rt]>>shamt	逻辑右移，注意 rs 字段未使用
03	sra rd,rt,shamt	R[rd]=R[rt]>>shamt	算术右移，注意 rs 字段未使用
04	sllv rd,rt,rs	R[rd]=R[rt]<<R[rs]	可变左移
08	jr rs	PC=R[rs]	R[rs] 值应是 4 的倍数，字对齐
09	jalr rs	R[31]=PC+8 PC=R[rs]	子程序调用
12	syscall	系统调用	无操作数
16	mfhi rd	R[rd]=HI	取 HI 寄存器的值，mflo 取 LO
17	mthi rs	HI=R[rs]	存 HI 寄存器的值，mtlo 存 LO
24	mult rs,rt	{HI,LO}=R[rs]*R[rt]	有符号乘，64 位结果送入 HI、LO 寄存器
32	add rd,rs,rt	R[rd]=R[rs]+R[rt]	溢出时发生异常，且不修改 R[rd]
34	sub rd,rs,rt	R[rd]=R[rs]-R[rt]	溢出时发生异常，且不修改 R[rd]
36	and rd,rs,rt	R[rd]=R[rs]&R[rt]	逻辑与
37	or rd,rs,rt	R[rd]=R[rs]|R[rt]	逻辑或
42	slt rd,rs,rt	R[rd]=(R[rs]<R[rt])?1:0	小于置位指令，有符号比较

图 2: 常用 R 型指令及 Func 编码

按照上表中的编码再次使用比较器与 Func 进行比较即可。其中 R_TYPE 代表 R 型运算指令的集合，不含 SysCall(R 型特殊指令)，因此在本实验里 R_TYPE 就是 $ADD + SLT$。

【ALU 控制逻辑】

除了 SLT 指令，都一律用加法操作 (操作码为 5)，否则使用有符号比较 (操作码为 c)

【根据指令输出对应的控制信号】

相应的控制信号如下图所示：

表 6.9 单周期 MIPS 处理器的控制信号及功能

控制信号	信号分类	功能说明
Jump	指令译码信号	j、jal 等无条件分支指令时生成
Branch	指令译码信号	分支指令译码信号，beq、bne 指令需要产生类似的译码信号
RegDst	多路选择控制	写入目的寄存器的选择，为 1 时写入 rd 寄存器，为 0 时写入 rt 寄存器
AluSrc	多路选择控制	控制 ALU 的第二输入，为 0 时输入 rt 寄存器，为 1 时输入扩展后的立即数
MemToReg	多路选择控制	为 1 时将从数据存储器读出的数据写回寄存器堆，否则将运算结果写回运算器
RegWrite	功能部件控制	控制寄存器堆写操作，为 1 时数据需要写入指定寄存器，受时钟驱动
AluOp	功能部件控制	控制 ALU 进行不同运算，具体取值和位宽与 ALU 的设计有关
MemWrite	功能部件控制	控制数据存储器写操作，为 1 时进行写操作，为 0 时进行读操作，写入受时钟驱动

图 3: D 单周期 MIPS 处理器控制信号及功能

1.2 单周期 MIPS（硬布线）

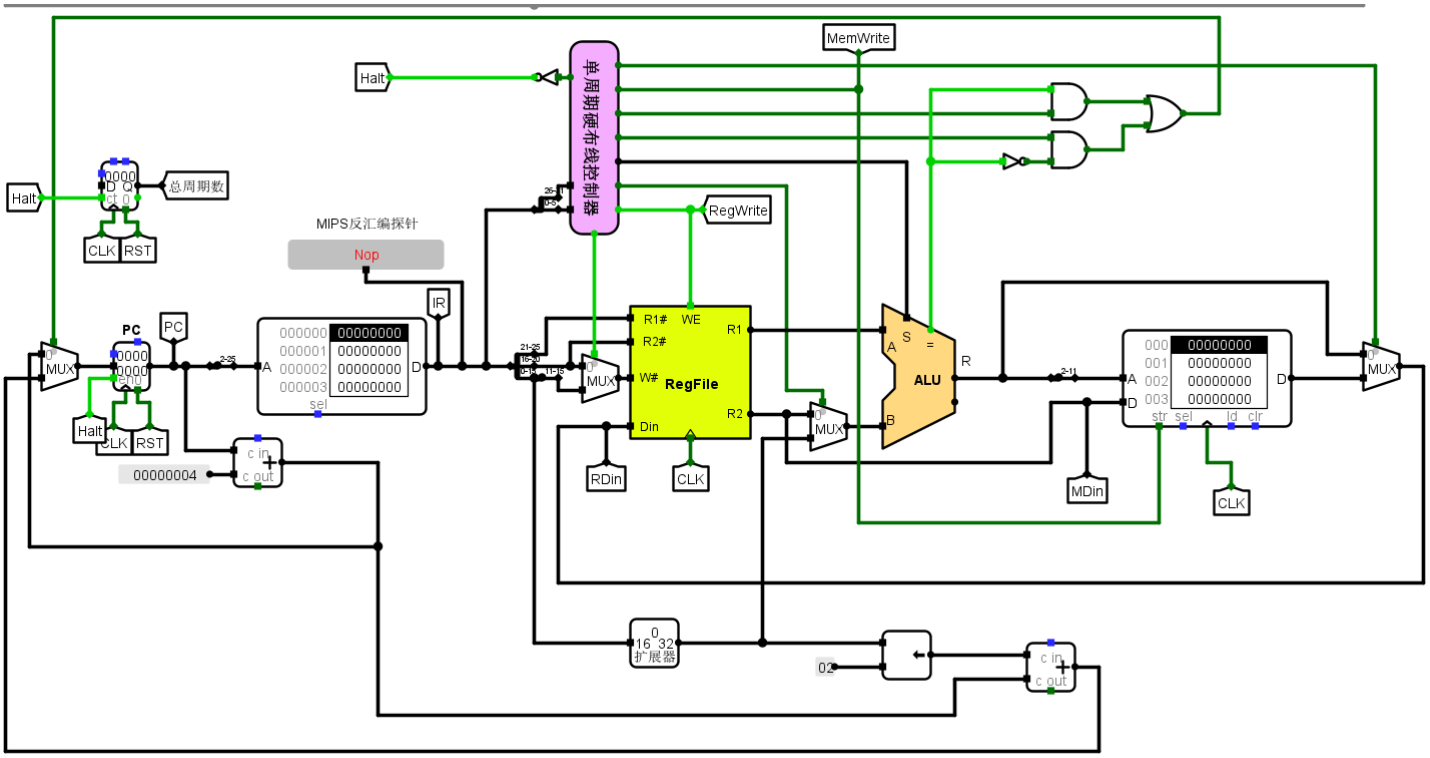

图 4: 单周期 MIPS（硬布线）

设计思路：

单周期 MIPS 数据通路如下图所示：

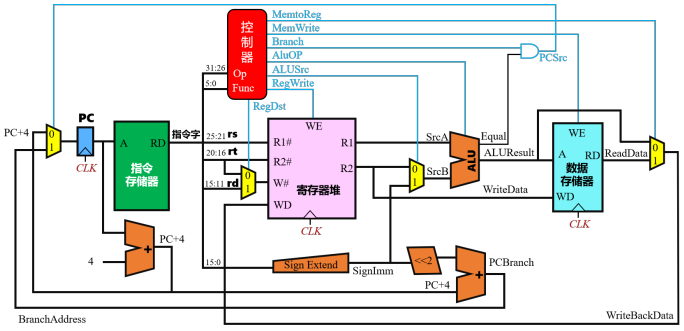

图 5: 单周期 MIPS 数据通路

参考数据通路进行连线。

2 实验中遇到的问题

1. **RegFile 读取指令时对指令的切分存在困难**

   切分困难主要是因为对分线器的使用不够熟练，导致最初不会如何单独切分出指令的某几位。在上网查阅资料后发现可以通过在分线器属性中设置第 x 位对应分线端口的位置及有无来实现对指令片段的切分提取。

2. **对根据指令输出对应的控制信号部分理解不够清晰**

   在设计单周期硬布线控制器的时候由于没有理解“根据指令输出对应的控制信号”的功能导致在第一次连线时忽视了该部分。

3. **混淆了 RAM 和 ROM**

   由于实验材料提供的初始结构将 RAM 放在了指令存储器的位置，因此我直接使用了 RAM 作为指令存储器，这也是因为对指令存储器的功能理解不够准确导致的错误。

   指令存储器使用 ROM 而不是 RAM，是因为指令在程序执行过程中通常是固定不变的，ROM 提供了更高的安全性、可靠性、低成本和低功耗。它的不可修改性确保了指令不会被错误修改或丢失，并且比 RAM 更适合存储程序指令。

   最终经过查阅资料，将指令存储器 RAM 改为 ROM。

3 实验心得

在单周期 MIPS CPU 的设计实践中，我单周期 MIPS CPU 的数据通路有了更具体直观的理解。在连线中切身体会到了数据的传递与存储。这次实验也让我对一些细节有了新的认识，比如指令存储器必须使用 ROM 而非 RAM 等。

同时，在实验中，我对 Logisim 的使用有了更全面的掌握，彻底理解了分线器的工作原理，学会了通过改变分线器属性来实现指令的切分和提取。

CPU 的设计牵扯到各个部件的协同工作，牵一发而动全身，这也要求我在实验时必须更加细致谨慎，否则出错后寻找出错点将异常困难。