单周期 MIPS CPU 设计

2023 年 6 月 8 日

1 实验目的

掌握典型 MIPS 指令的数据通路和控制信号，掌握控制器设计的基本原理，能利用硬布线控制器的设计原理，在 Logisim 平台中设计实现 MIPS 单周期 CPU。

2 实验内容

利用 Logisim 平台构建的运算器、寄存器文件、存储系统等部件，以及其它功能部件，构建一个 32 位 MIPS CPU 单周期处理器。要求支持 9 条 MIPS 核心指令，包括运算类指令 ADD、SUBI、AND、ORI、SLT，访存指令 LW、SW，分支指令 BEQ、J。

3 MIPS 指令介绍

MIPS 指令集有三种指令格式：I 型指令，J 型指令，R 型指令。其中，I 型指令用于有立即数的指令，如访存指令 LW 和 SW、分支指令等；J 型指令用于指令 j 和 jal；R 型指令用于所有其他的指令。

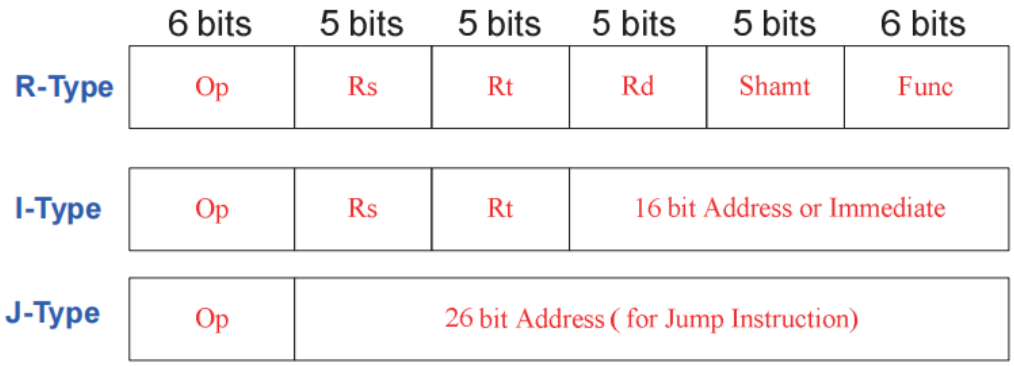

图 1: MIPS 指令格式

在 R 型指令中，各指令段的含义如下：

- Op: 指令操作码
- Rs: 第一个源操作数寄存器号，参与运算使用

- Rt: 第二个源操作数寄存器号，参与运算使用
- Rd: 目的寄存器号，存放运算结果
- Shamt: 位偏移量，仅在位移指令使用，在此直接置 0
- Func: 指令函数码，用于选择 Op 操作中的具体函数

比如加法运算，在指令操作码中，指出它是算术运算；在指令函数码中，指出它是算术运算中的加法运算。

在 I 型指令中，各指令段的含义如下：

- Op: 指令操作码
- Rs: 第一个源操作数寄存器号，参与运算使用
- Rt: 第二个源操作数寄存器号，参与运算使用
- 16 位立即数: 作为数据，参与运算使用

在 I 型指令中，各指令段的含义如下：

- Op: 指令操作码
- 26 位地址数: 作为地址，参与寻址使用。通常用于指令的跳转使用，后面的数据用于提供跳转地址

4 MIPS 数据通路设计

对于 MIPS 数据通路的设计，要求每位同学建立数据通路的思路，理清每类指令执行过程中数据的来源、参与的运算与数据的去处。本节以 R 型指令为例，简要介绍数据通路的设计。其他两类指令的数据通路设计与此类似，请同学们自行设计，最后将所有数据通路合到一起，构建出完整的数据通路。在这个过程中，充分体会自底向上，从部分到整体的设计思想。

R 型指令执行过程中数据的流动主要涉及四个阶段：

- 从指令存储器中取指令
- 根据源操作数寄存器号从寄存器堆中取两个操作数
- 根据 op 和 func 对源操作数进行运算
- 根据目的寄存器号，将运算结果写回寄存器堆

R 型指令的数据通路如下图所示：

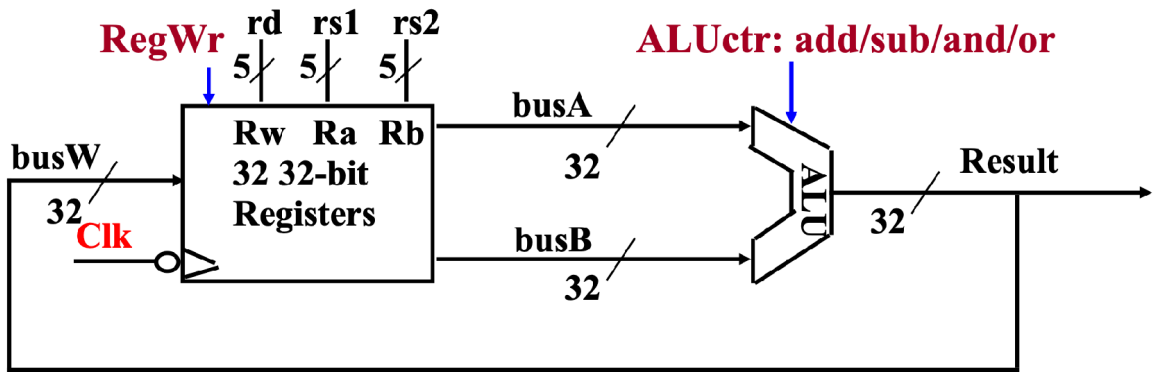

图 2: R 型指令数据通路

Ra, Rb, Rw 分别对应指令的 rs1, rs2, rd

ALUctr、RegWr：指令译码后产生的控制信号，它们的取值分别是：ALUctr=add，RegWr=1。

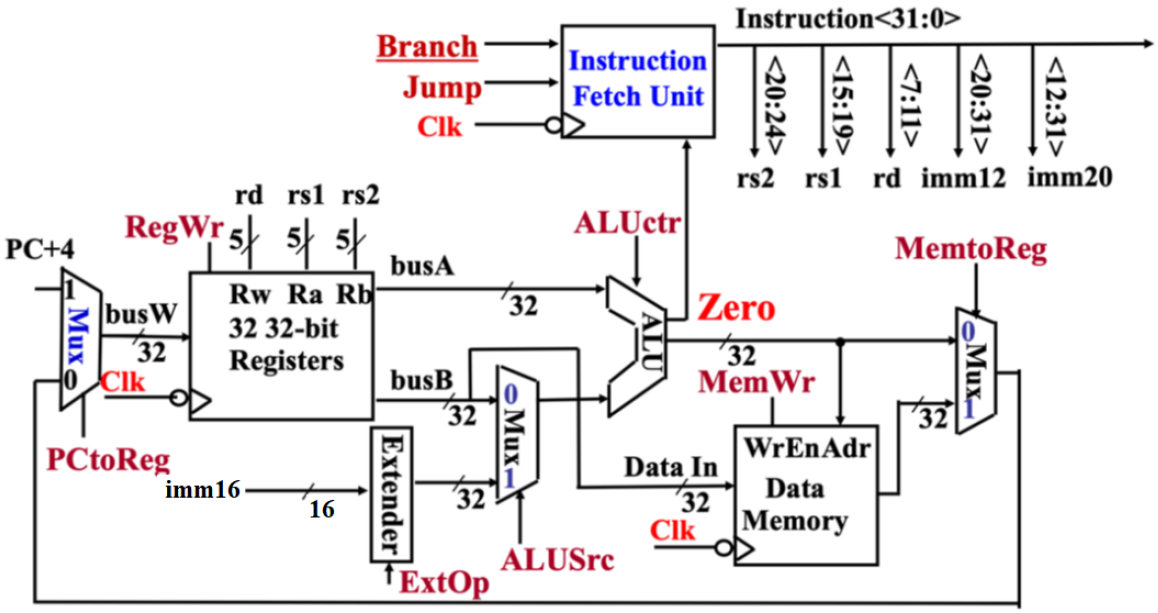

图 3: 整体数据通路设计参考

5 MIPS 控制器设计

MIPS 控制器用于产生每条指令执行所需的全部控制信号，例如寄存器读写信号，存储器读写信号，alu 运算类型信号、跳转信号等，主要涉及两项技术：固定字段译码和立即数扩展操作。在 MIPS 指令格式中，操作码字段以及 rs、rt 字段都是在固定的位置，这种技术称为固定字段译码技术。

在进行 MIPS 控制器设计时，首先要列出 9 条核心指令执行所需的全部控制信号，并给出每个控制信号的产生条件，例如通过真值表的方式得到每个信号的逻辑表达式。

本节以 RegWrite 信号为例介绍 MIPS 控制器的设计。首先，Regwrite 信号可以仅通过 op 字段产生，如下图所示。

	信号	R型	LW	SW	BEQ
输入	Op5	0	1	1	0
	Op4	0	0	0	0
	Op3	0	0	1	0
	Op2	0	0	0	1
	Op1	0	1	1	0
	Op0	0	1	1	0
输出	RegDst	1	0	X	X
	AluSrc	0	1	1	0
	MemtoReg	0	1	X	X
	RegWrite	1	1	0	0
	MemRead	0	1	0	0
	MemWrite	0	0	1	0
	Branch	0	0	0	1
	ALUOp1	1	0	0	0
	ALUOp2	0	0	0	1

图 4: 控制器 control 真值表（标红为 regWrite 信号）

由此得到 Regwrite 信号的逻辑表达式：

RegWr = R-type + LW + SW + BEQ

= !Op<5>&!Op<4>&！Op<3>&!Op<2>&！Op<1>&！Op<0>　　(R-type)

+ Op<5>&!Op<4>&!Op<3>&!Op<2>&Op<1>&Op<0>　　(LW)

+ Op<5>&！Op<4>&Op<3>&!Op<2>&Op<1>&Op<0>　　(SW)

+ ！Op<5>&！Op<4>&!Op<3>&Op<2>&!Op<1>&！Op<0>　　(BEQ)

最后，在 Logisim 中对 Regwrite 信号进行硬件布线。

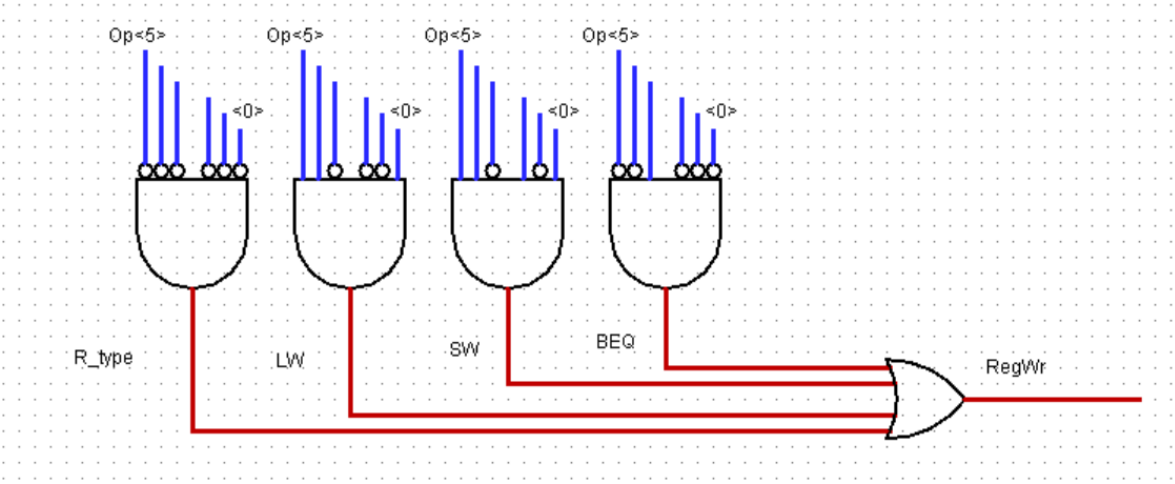

图 5: regWrite 信号硬件布线

6 Logisim 介绍与使用

logisim 是一个用于设计和模拟数字逻辑电路的教育工具，具有简单易学的界面、分层电路、线束和大型元件库。作为一个 Java 应用程序，它可以在许多平台上运行。该软甲具有如下特点：(i) 使用一个直观的图形界面设计电路（ii）在绘制电路的过程中，观察电路的模拟情况（iii）可在 Linux、MacOS X 和 Windows 下运行（iv) 支持组合分析模块从真值表建立电路。

本次实验实现单周期 mips CPU 主要使用 Logisim 进行完成。

6.1 Logisim 安装环境

- 安装：java jdk > 10.0 (推荐 jdk 11)
- 在命令行中使用 java -version 查询 java 开发包版本
- Logisim 下载网址
- 直接打开 logisim，即可使用

6.2 Logisim 软件介绍

Logisim 界面介绍

如图所示6，Logisim 的主界面主要由菜单栏、快捷图表、文件目录、部件部署、部件属性等五部分组成

- 菜单栏：主要是可以新增文件、修改偏好等设置
- 快捷图标：主要包括抓手工具、箭头工具、文字工具、电路封装、以及一些常用的门
- 文件目录中，其中有自行定义的一些电路，以及软件提供的一些常用的电路结构，包括与或非门、加法器之类的电路
- 部件属性：描述页面中选中的元器件的相关属性

a	b	a+b
0	0	0
0	1	1
1	1	0

表 1: 半加器真值表.

- 部件部署页面：主要用于描述电路

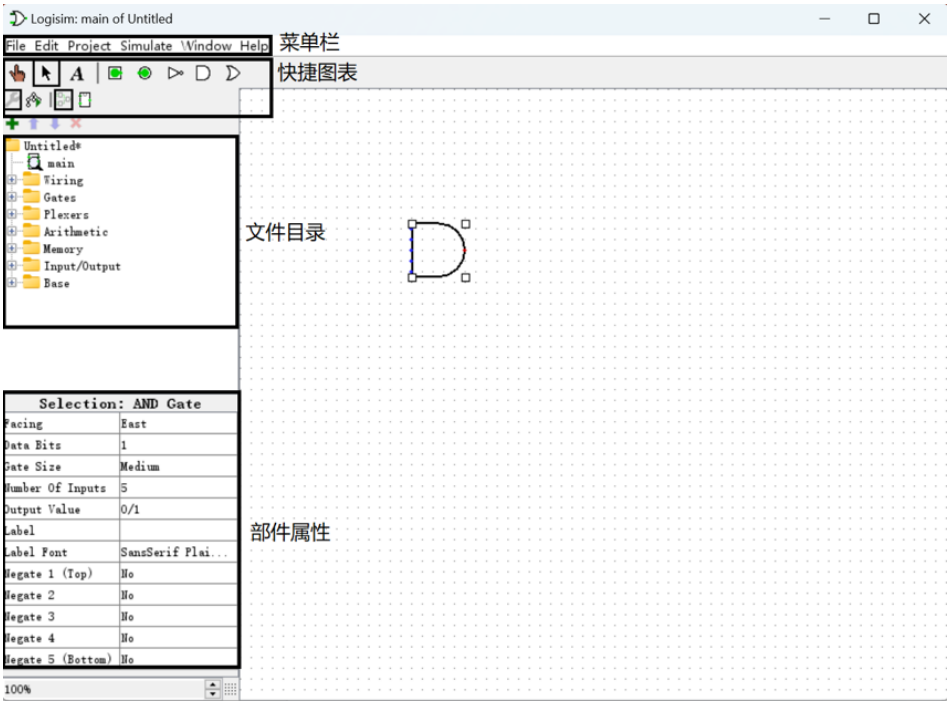

图 6: logisim 主界面

Logisim 实现半加器

半加器的真值表如下表 1所示

通过真值表我们可以知道，半加器在运算上等效于一个异或门。如图7所示，图中使用 Logisim 提供的异或门实现了半加器，并将该半加器于 logisim 提供的标准加法器进行比较。

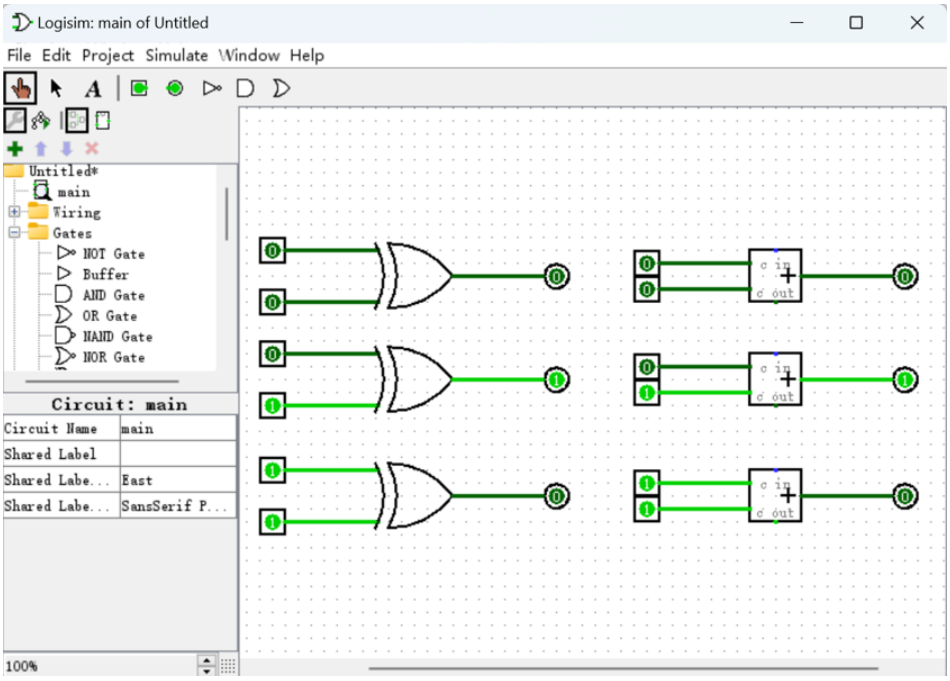

图 7: Logisim 实现半加器

Logisim 电路封装

在一个电路设计好了之后，我们可以使用 Logisim 的封装工具对电路进行封装。这样方便其他电路对该电路的调用。在 Logsim 的快捷按钮中，提供了封装按钮8 。

图 8: Logisim 电路封装快捷按钮

通过 Logisim 对半加器进行封装，如图9所示。可以自行定义输入输出的位置，方便需要进行调用。

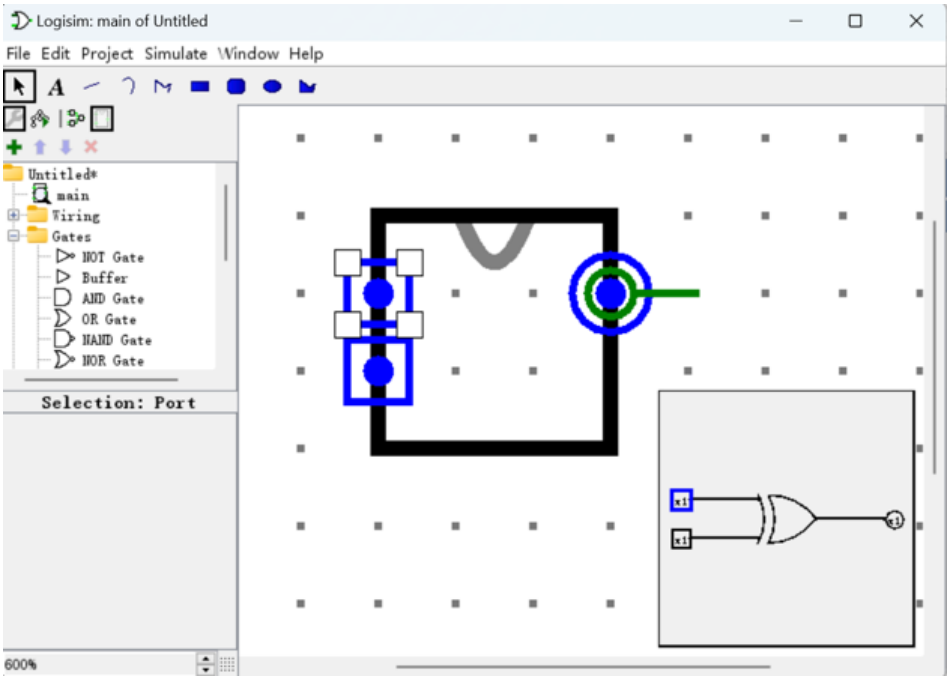

图 9: Logisim 电路封装界面

7 实验测试与验收

自主设计测试程序，验证所设计的单周期 mips CPU 的正确性。
实验结束后，需要进行现场验收，最终形成实验报告。