

Figure2 设计层次(仅供参考)

4. IFU: 内部包括 PC、IM(指令存储器)及相关逻辑。
 - a) PC: 用寄存器实现, 宽度为 30 位。PC 应具有复位功能。
 - b) IM: 容量为 32bit×32 字, 用 ROM 实现。
 - c) 说明: 由于 IM 地址仅为 5 位, 因此请用 2 个对接的 Splitter 实现将 PC 低位地址与 IM 地址连接。
5. GPR: 以 32 个 32 位具有写使能的寄存器为基础, 辅以多路选择器。
6. ALU: 实现加法及减法时, 不允许使用 logisim 内置的 Adder 及 Subtractor! 加法、减法实现必须以门电路为基础。
7. EXT: 可以使用 logisim 内置的 Bit Extender。
8. DM: 容量为 32bit×32 字, 用 RAM 实现。
 - a) DM 应采用双端口模式, 即设置 RAM 的“Data Interface”属性为“Separate load and store ports”。
9. 必须有时钟源, 即如 Figure1 中绿圈所示。
 - a) 只有设置了时钟源, 系统才能自动运行, 从而让程序连续运行。

三、 模块定义【WORD】

10. 仿照下面给出的 IFU 模块定义, 给出所有功能部件的模块定义。

- a) IFU、GPR、ALU、EXT、DM、Controller。

- a) IFU

模块接口

信号名	方向	描述
IfBeq	I	当前指令是否为 beq 指令标志 1: 当前指令为 beq 0: 当前指令非 beq

Zero	I	ALU 计算结果为 0 标志 1: 计算结果为 0 0: 计算结果非 0
Clk	I	时钟信号
Reset	I	复位信号 1: 复位 0: 无效
Instr[31:0]	0	32 位 MIPS 指令

功能定义

序号	功能名称	描述
1	复位	当复位信号有效时, PC 被设置为 0x00000000
2	取指令	根据 PC 从 IM 中取出指令
3	计算下一条指令地址	如果当前指令不是 beq 指令, 则 $PC \leftarrow PC+1$ 如果当前指令是 beq 指令, 并且 zero 为 1, 则, $PC \leftarrow PC+sign_ext$

b) GPR

模块接口

信号名	方向	描述
Wd[31:0]	I	写入数据的输入
Regwrite	I	读写控制信号 1: 写操作 0: 读操作
Clk	I	时钟信号
Reset	I	复位信号 1: 复位 0: 无效
A1[4:0]	I	读寄存器地址 1
A2[4:0]	I	读寄存器地址 2
A3[4:0]	I	写寄存器地址
Rd1[31:0]	0	32 位数据输出 1
Rd2[31:0]	0	32 位数据输出 2

功能定义

序号	功能名称	描述
1	复位	当复位信号有效时, 所有寄存器被设置为 0x00000000
2	读寄存器	根据输入的寄存器地址读出数据
3	写寄存器	根据输入的地址, 把输入的数据写进选中的寄存器

c) ALU

模块接口

信号名	方向	描述
A[31:0]	I	32 位输入数据 1
B[31:0]	I	32 位输入数据 2
F[1:0]	I	控制信号 01: 或运算 10: 减法

		11: 加法
C[31:0]	0	32 位数据输出

功能定义

序号	功能名称	描述
1	或	A B
2	减	A-B
3	加	A+B

d) EXT

模块接口

信号名	方向	描述
A[15:0]	I	16 位数据输入
Extop	I	控制信号 0: 高位补 0 1: 低位补 0
B[31:0]	0	32 位数据输出

功能定义

序号	功能名称	描述
1	高位补 0	高 16 位补 0
2	低位补 0	低 16 位补 0

e) DM

模块接口

信号名	方向	描述
D[31:0]	I	写入数据的输入
Memtoreg	I	读写控制信号 1: 读操作
Memwrite	I	读写控制信号 1: 写操作
Clk	I	时钟信号
Reset	I	复位信号 1: 复位 0: 无效
A[4:0]	I	操作寄存器地址
out[31:0]	0	32 位数据输出

功能定义

序号	功能名称	描述
1	复位	当复位信号有效时, 所有数据被设置为 0x00000000
2	读	根据输入的寄存器地址读出数据
3	写	根据输入的地址, 把输入的数据写入

f) controller

模块接口

信号名	方向	描述
op[5:0]	I	六位 op
func[5:0]	I	六位 function

Regdst	0	写地址控制
Alusrc	0	cpu 第二操作数选择控制
Memtoreg	0	DM 读控制
Regwrite	0	GPR 读写控制
Memwrite	0	DM 写控制，写入 GPR 数据选择
npc_sel	0	Beq 指令标志
extop	0	控制 ext 扩展方式
Aluop[1:0]	0	控制 cpu 进行相应运算

11. IFU 模块定义(参考样例)

(1) 基本描述

IFU 主要功能是完成取指令功能。IFU 内部包括 PC、IM(指令存储器)以及其他相关逻辑。IFU 除了能执行顺序取值令外，还能根据 BEQ 指令的执行情况决定顺序取值令还是转移取值令。

(2) 模块接口

信号名	方向	描述
IfBeq	I	当前指令是否为 beq 指令标志。 1: 当前指令为 beq 0: 当前指令非 beq
Zero	I	ALU 计算结果为 0 标志。 1: 计算结果为 0 0: 计算结果非 0
clk	I	时钟信号
Reset	I	复位信号。 1: 复位 0: 无效
Instr[31:0]	O	32 位 MIPS 指令

(3) 功能定义

序号	功能名称	功能描述
1	复位	当复位信号有效时，PC 被设置为 0x00000000。
2	取指令	根据 PC 从 IM 中取出指令。
3	计算下一条指令地址	如果当前指令不是 beq 指令，则 $PC \leftarrow PC+1$ 如果当前指令是 beq 指令，并且 zero 为 0，则 $PC \leftarrow PC+1$ 如果当前指令是 beq 指令，并且 zero 为 1，则 $PC \leftarrow PC+sign_ext(当前指令 15..0)$ [注]PC 取地址为 4 字节，固低 2 位地址可以去除。

四、 控制器设计

12. 请仿照下图给出 MIPS-Lite2 指令集的单周期控制器真值表。【WORD】

See MIPS Green Sheet

	func	10 0000	10 0010	n/a		
	op	00 0000	00 0000	00 1101	10 0011	10 1011 00 0100
		add	sub	ori	lw	sw beq
Control Signals	RegDst	1	1	0	0	X X
	ALUSrc	0	0	1	1	1 0
	MemtoReg	0	0	0	1	X X
	RegWrite	1	1	1	1	0 0
	MemWrite	0	0	0	0	1 0
	nPC_sel	0	0	0	0	0 1
	ExtOp	X	X	0	1	1 X
	ALUctr<2:0>	Add	Subtract	Or	Add	Add Subtract

All Supported Instructions

Figure3 控制信号真值表

a) 结合真值表，请给出数据通路每个功能部件的每个控制信号的布尔表达式。

Funcnt[5:0]	100001	100011						
OpCode[5:0]	000000		001101	000100	101011	100011	000110	001111
	ADDU	SUBU	ORI	BEQ	SW	LW	BLEZ	LUI
RegDst	1	1	0	X	X	0	X	0
RegWrite	1	1	1	0	0	1	0	1
ALUSrc[1:0]	0	0	1	0	1	1	2	X
MemtoReg[1:0]	0	0	0	X	X	1	X	2
MemWrite	0	0	0	0	1	0	0	0
nPC_sel	0	0	0	1	0	0	0	0
ALUctr[1:0]	2	3	1	3	2	2	3	X
IfBlez	0	0	0	0	0	0	1	0

$$RegDst = \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot \overline{o1} \cdot \overline{o0} \cdot f5 \cdot \overline{f4} \cdot \overline{f3} \cdot \overline{f2} \cdot f0$$

$$RegWrite = \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot \overline{o1} \cdot \overline{o0} \cdot f5 \cdot \overline{f4} \cdot \overline{f3} \cdot \overline{f2} \cdot f0 + \overline{o5} \cdot \overline{o4} \cdot o3 \cdot o2 \cdot o0 \\ + o5 \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot o1 \cdot o0$$

$$ALUsrc = \overline{o5} \cdot \overline{o4} \cdot o3 \cdot o2 \cdot o0 + o5 \cdot \overline{o4} \cdot \overline{o2} \cdot o1 \cdot o0$$

$$PCsrc = \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot o2 \cdot \overline{o1} \cdot \overline{o0}$$

$$MemWrite = o5 \cdot \overline{o4} \cdot o3 \cdot \overline{o2} \cdot o1 \cdot o0$$

$$MemRead = o5 \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot o1 \cdot o0$$

$$MemtoReg = o5 \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot o1 \cdot o0$$

$$ExtOp = o5 \cdot \overline{o4} \cdot \overline{o2} \cdot o1 \cdot o0$$

$$ALUctr[0] = \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o2} \cdot \overline{o1} \cdot \overline{o0} \cdot f5 \cdot \overline{f4} \cdot \overline{f3} \cdot \overline{f2} \cdot f1 \cdot f0 + \overline{o5} \cdot \overline{o4} \cdot o3 \cdot o2 \cdot o1 \cdot o0 + \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot o2 \cdot \overline{o1} \cdot \overline{o0}$$

$$ALUctr[1] = \overline{o5} \cdot \overline{o4} \cdot o3 \cdot o2 \cdot o0$$

- b) 表达式中只能使用“与、或、非”3种基本逻辑运算。
- c) 每个控制信号的表达式应该是指令 opcode 域与 funct 域的函数。
- d) 对于多位的控制信号(如 ALUctr)，应诸位给出其逻辑表达式。

13. 请在 logisim 中完成控制器设计。

- a) 控制器整体结构需要仿照 Figure 4 实现。

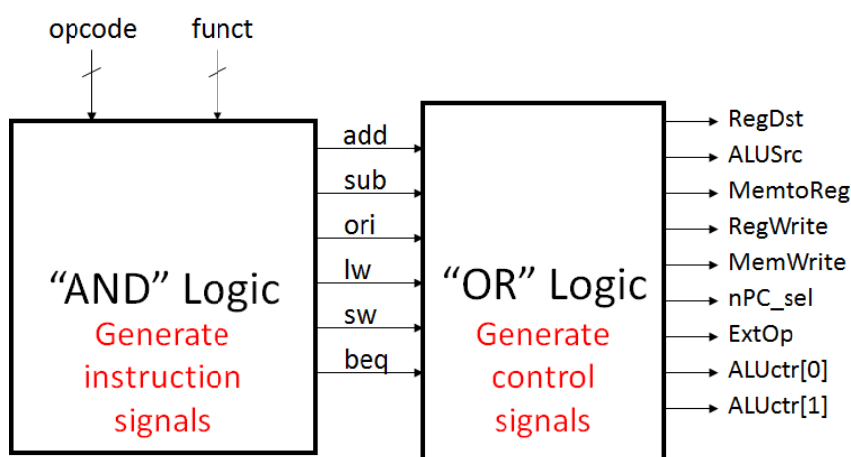


Figure 4 控制器基本结构

- b) 控制信号必须仿照下图方式实现。

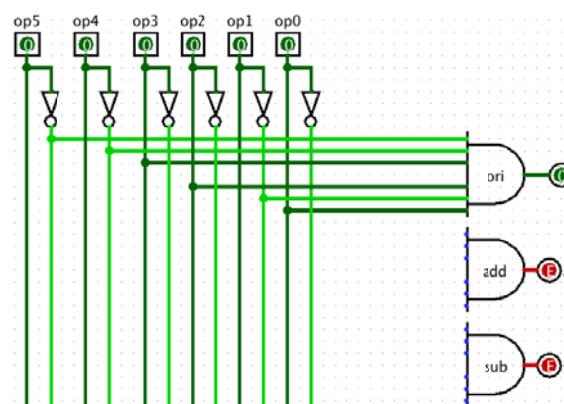


Figure5 与阵列：译码产生指令标识

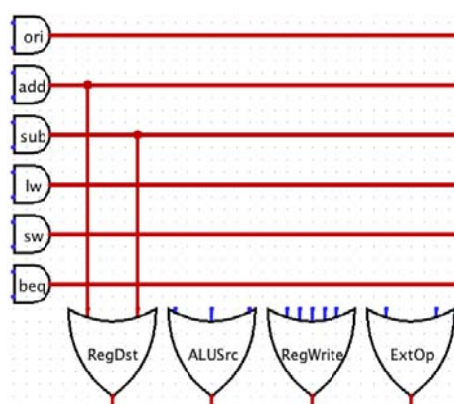


Figure6 或阵列：产生最终的控制信号

五、 测设要求

14. 所有指令都应被测试充分。
15. 构造 1 个至少 20 条以上指令的测试程序，并加载至 IFU 中运行通过。
 - a) MIPS-Lite2 定义的每条指令至少出现 1 次以上。
 - b) 演示时，测试程序必须已经通过 IFU 中的 IM 的“Load Image”加载完毕。
16. 详细说明你的测试程序原理。**【WORD】**
 - a) 应明确说明测试程序的测试期望，即应该得到怎样的运行结果。

blez 测试程序要实现这样的功能：依次向 DM 内写入 0xAABB_0000 到 0xAABB_000f16 个数。
 - b) 每条汇编指令都应该有注释。

\$0 值为 0，辅助赋值及恒等跳转

\$1 值为 1，用来参加循环中\$17 的自减和\$19 的自增；

\$2 值为 8，代表每次循环中向 DM 内存数 2 次（每次偏移量为 4）；

\$16 初始值为 0x0000_0000，每循环一次值加 8，是每次循环的基址；

\$17 初始值为 8，每循环一次值减 1，表示循环将执行 8 次；

\$18 初始值为 0xAABB_0000，在循环开始时将当前\$19 的值存入\$16，并在循环结束时保存\$19 的地址；

\$19 无初始值，在每次循环中由基址自增，并保存至基址的下一个偏移量(\$16+4)，然后再自增。

六、 问答 **【WORD】**

17. 请充分利用 Figure3 中的 X 可以将控制信号化简为最简单的表达式。

$$RegDst = \overline{o5} \cdot \overline{o4} \cdot \overline{o3} \cdot \overline{o1} \cdot \overline{o0} \cdot f5 \cdot \overline{f4} \cdot \overline{f3} \cdot \overline{f2} \cdot f0$$

$$MemtoReg = o5 \cdot \overline{o4} \cdot \overline{o2} \cdot o1 \cdot o0$$

18. 对于 Figure5、Figure6 中的与或阵列来说，1 个 3 输入与门最终转化为 2 个 2 输入与门，1 个 4 输入与门最终转化为 3 个 2 输入与门，依次类推。或阵列也类似计算。那么

a) 请给出采用 Figure5、Figure6 中的方法设计的每个控制信号所对应的 2 输入与门、2 输入或门、非门的数量。

19.	2 输入与门	2 输入或门	非门
RegDst	17	1	10
RegWrite	32	4	10
ALUsrc	20	3	5
PCsrc	5	0	5
MemWrite	5	0	2
MemRead	5	0	3
MemtoReg	5	0	3
ExtOp	15	2	4
ALUctr[1]	10	1	3
ALUctr[0]	21	2	9

a) 请与第 17 项对比，你更喜欢哪种设计方法。为什么

第一种的控制信号都需要对其分配单独的与门、或门，因为它是直接对 op、func 的 12 位或 6 位信号的逻辑表达式，所以没有针对性并且浪费元件。

而第二种是先把 op、func 变成相应的指令信号，再由指令信号生成控制信号。当一种指令对应多种控制信号为 1 时，不必再对每个信号再单独为这个指令分配与门，而可以共用这个指令的信号，再添加或门就可以了。

七、 其他要求

20. 打包文件：Logisim 工程文件、测试程序二进制文件、项目报告。

21. 时间要求：各班实验指导教师指定。

22. 本实验要求文档中凡是出现了【WORD】字样，就意味着该条目需要在实验报告中清晰表达。
23. 实验报告请按照《计算机组成原理实验报告撰写规则.doc》要求排版。

八、 实验测试要求

24. 实验成绩由下列部分组成：回答问题、MIPS-Lite2 处理器正确性、增加新指令后的处理器正确性等。
25. 实验测试时，你需要展示你的设计并证明其正确性。
26. 实验指导教师会临时增加 1~2 条指令，你需要在规定的时间内完成对原有设计的修改，并通过实验指导教师提供的测试程序。

九、 开发与调试技巧

27. 对于每条指令，请认真阅读《MIPS32® Architecture For Programmers Volume II: The MIPS32® Instruction Set》！
 - a) 如果测试时，你无法清楚的解释所要求的指令，测试成绩将减一档！
28. Figure1 中 Tunnel 的用途是将具有相同 name 的 tunnel 连接在一起。Tunnel 可以避免将图画得很乱。
29. Figure1 中 Probe 的用途是显示被 probed 信号的值，便于调试。
30. Figure1 中 Splitter 的用途是从某组信号中提取其中部分信号。例如，IFU 输出 32 位指令，需要提取高 6 位(OpCode)和低 6 位(Funct)分别输入 controller。
 - a) splitter 是有位序的！但字号太小，需要放大设计图(界面左下有比例设置)。
 - b) 建议高位永远在上，低位永远在下
31. 如果你对于 logisim 内置的某个部件的端口不明白，请：
 - a) 仔细阅读 Help→Library Refrence 关于该部件的描述。
 - b) 放大 logisim 显示比例直至能清晰看到代表部件的各个端口的圆点，然后将鼠标停留相应的圆点上，就可以读取端口具体信息。
32. 建议先在 MARS 中编写测试程序并调试通过。
 - a) 注意 MARS 中的“Settings→Memory Configuration”只能配置指令存储器起始地址为 0 地址，而不能将指令存储器和数据存储器的起始地址均配置为 0 地址！

- b) 由于 logisim 设计中的 DM 起始地址为 0, 因此请仔细观察所用到的指令, 在把 MARS 中调试通过的二进制码导出后, 你可能需要手工修改指令码中的数据偏移。
 - c) 提示: 事实上, 在现代主流计算机中, 数据存储器 and 指令存储器的起始地址不应该重叠。但在本设计中, 由于采用分离存储器设计方案, 因此可以暂时忽略这一点。
33. 当然, 如果你能再自学一点点存储器译码的知识, 那么只需再增加一个 DM 片选信号, 一切都搞定了(就不需要再考虑第 32.b)了)。
- a) 片选信号就是对指令发出的数据存储器地址的高位分析。
 - b) 假设 DM 有 256MB 容量, 并且映射在 0x3000_0000~0x3FFF_FFFF 区间。那么只需要把高 4 位地址与 0x3 进行比较, 比较结果就是 DM 的片选信号。
 - c) Logisim 内置的 RAM 有片选信号!
34. 提示: 你可以考虑增加 7 段数码管等输入输出来让你的测试结果更加直观。
- a) 本条非必做要求。
 - b) 7 段数码管也需要类似片选等信号, 其工作原理与第 33 项类似。