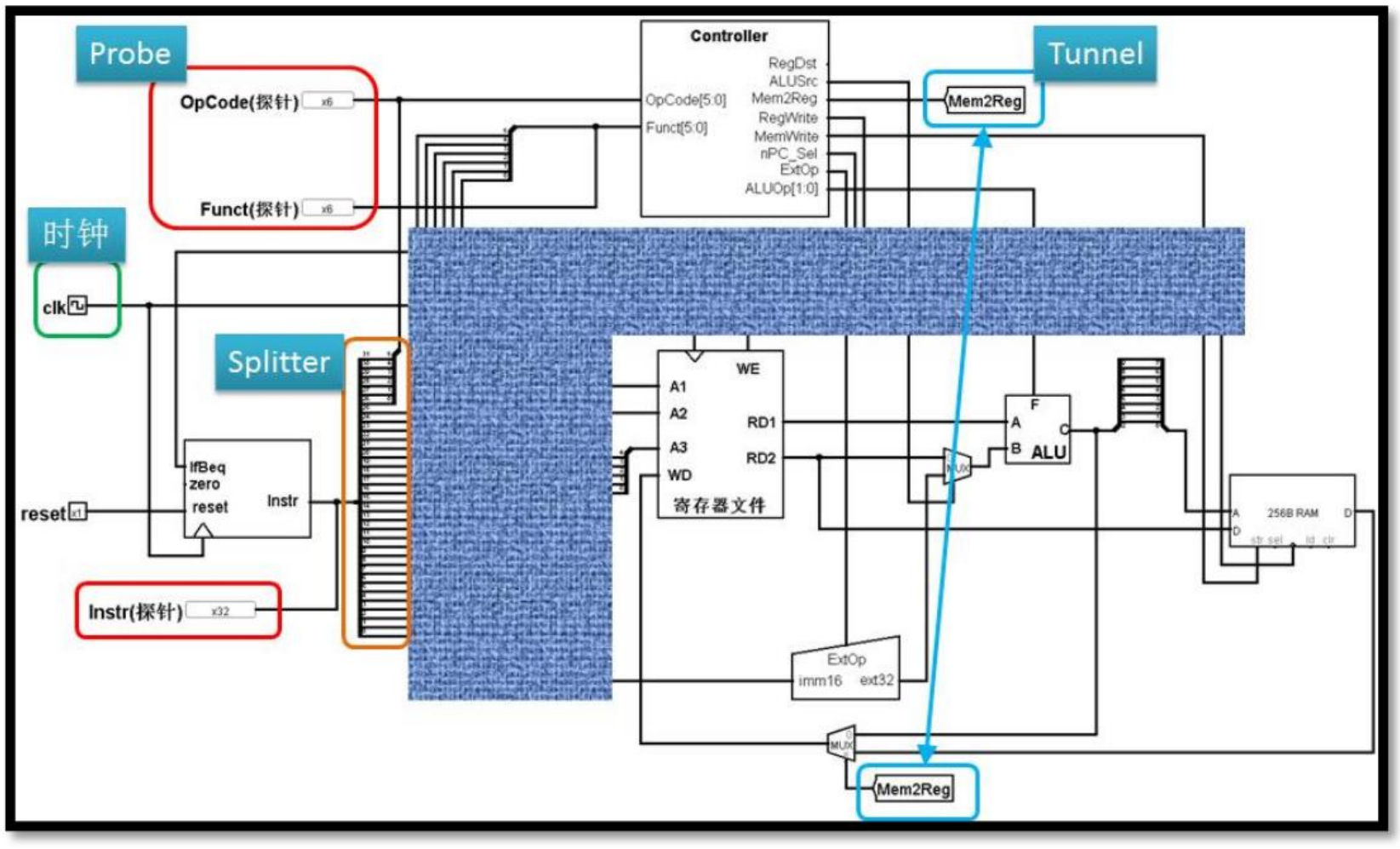
概述

通过 P0、P1、P2 的学习，想必你已经基本掌握了本课程工具集的使用，是时候小试牛刀了。在本节中，你将使用 Logisim 开发一个简单的 MIPS 单周期处理器，并使用 Mars 自行编写测试程序，验证 CPU 设计的正确性。本节的实验将带你走进 CPU 的世界，一窥 CPU 内部的构造，为今后实验的学习打下基础。在这个过程中，我们希望同学能够自行阅读理论课本的有关章节作为参考。 在这个过程中，我们也希望能让同学从头开始编写好自己的 CPU 设计文档，它是我们进行电路搭建的指南。**值得一提的是，我们将会在在课下以及课上测试环节对文档中的内容（设计文档、思考题）加以检查，请做好准备！**

基本思路

本节中，我们设计的 CPU 将包含 Controller（控制器）、IFU（取指令单元）、GRF（通用寄存器组，也称为寄存器文件、寄存器堆）、ALU（算术逻辑单元）、DM（数据存储器）、 EXT（位扩展器）等基本部件，通过 MUX、 Splitter 等内置器件组合连接成数据通路。一个可能的顶层设计参考图示为：



设计与测试说明

- **处理器为 32 位处理器。**
- **处理器应支持的指令集为：{addu, subu, ori, lw, sw, beq, lui, nop}。**
- **nop 机器码为 0x00000000， 即空指令，不进行任何有效行为（修改寄存器等）**
- addu,subu 可以不支持溢出。
- 处理器为**单周期**设计。
- 需要采用**模块化**和**层次化**设计。顶层有效的驱动信号要求包括且仅包括：**reset (clk 请使用内置时钟模块).**
- **需自行构造测试集**，验证设计的正确性。（通过课下自动测试并**不意味着**你的设计**完全不存在问题**）
- **之后每节说明性内容后都附有文档撰写建议与相关思考题，请务必完成相关思考题并附在 CPU 设计文档后！**
- 最后需要提交的内容： CPU 设计文档（含思考题）、 Logisim 电路源文件。
- 友情提示：请通读本 Lab 所有内容后再进行设计避免无谓的修改！

模块规格

模块规格是设计文档的重要部分，它包含了 CPU 各个模块的**端口说明**与**功能定义。**一份好的模块规格可以让其他人快速理解该模块的功能并加以实现。这就相当于一份 CPU 重要部件的“说明书”。

在模块规格这个部分我们不直接给出详细的端口说明与功能规定，仅给出简略的要求，希望同学们在使用 Logisim 搭建 CPU 过程中，自行完成相应的设计，**构建较为完整的端口与功能说明并上传。**在课上测试中，我们将会检查此部分，**请保证你一定能够让其他人理解你的说明！**

下面我们就各个模块给出几点要求：

1. IFU（取指令单元）：内部包括 PC（程序计数器）、IM(指令存储器)及相关逻辑。

- PC 用寄存器实现，应具有复位功能。
- 起始地址：0x00000000。
- IM 用 ROM 实现，容量为 32bit * 32 字。
- 因 IM 实际地址宽度仅为 5 位，故需要使用恰当的方法将 PC 中储存的地址同 IM 联系起来。

2. GRF（通用寄存器组，也称为寄存器文件、寄存器堆）

- 用具有写使能的寄存器实现，寄存器总数为 32 个。
- **0 号寄存器**的值保持为 0。

3. ALU（算术逻辑单元）

- 提供 32 位加、减、或运算及大小比较功能。
- 可以不支持溢出（不检测溢出）。

4. DM（数据存储器）

- 使用 RAM 实现，容量为 32bit * 32 字。
- 起始地址：0x00000000。
- RAM 应使用双端口模式，即设置 RAM 的 Data Interface 属性为 Separate load and store ports。

5. EXT：

- 可以使用 logisim 内置的 Bit Extender。

6. Controller（控制器）

- 使用与或门阵列构造控制信号，
- 具体方法见后文叙述。

文档撰写建议

模块规格部分中，由于控制器较为特殊以及其在 CPU 中的核心地位，建议单独开一章节描述控制器的设计。

本节思考题

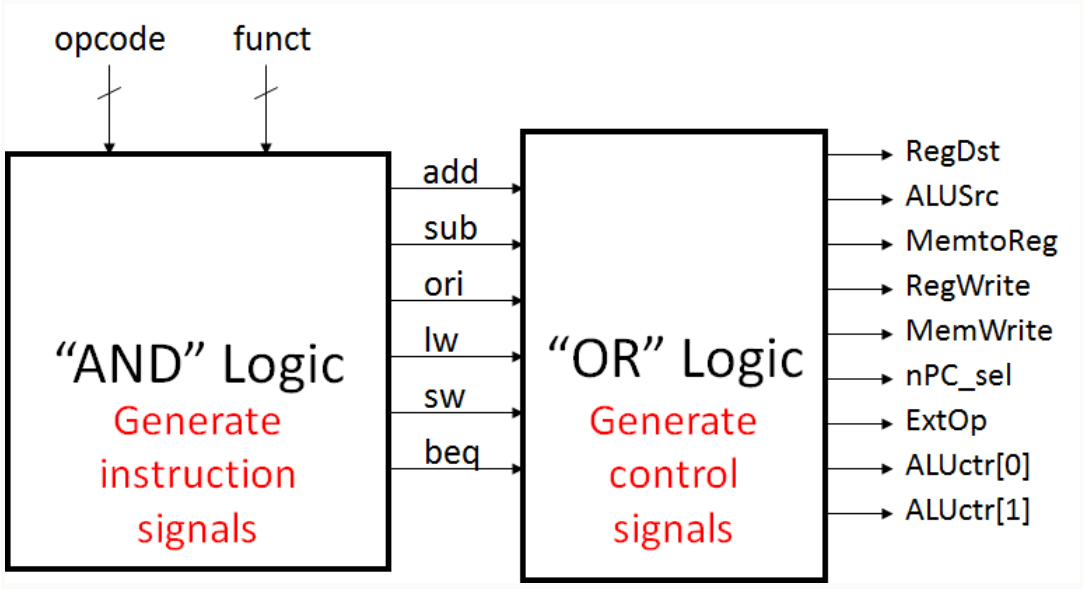
1. 在上个学年的计组课程中，PC（程序计数器）位数被规定为 30 位，试分析其与 32 位 PC 的优劣。

2. 现在我们的模块中 IM 使用 ROM， DM 使用 RAM， GRF 使用寄存器，这种做法合理吗？ 请给出分析，若有改进意见也请一并给出。

控制器设计

控制器的设计，从最基本的层面来说，是一个**译码**的过程，将每一条机器指令中包含的信息，转化为给 CPU 各部分的控制信号（RegDst， ALUSrc 等等），相信同学们在理论课上也了解了该部分的内容。而到了实践层面，如何让解码过程变的工程上可实现，是一个重要的问题。最为暴力的办法就是强行生成真值表，而这样在实际操作中不具有可扩展性与易调试性，在指令数一多时就非常容易犯错。 聪明的前辈们给我们创造了如下的方法。

我们把解码逻辑分解为**和逻辑**和**或逻辑**两部分：和逻辑的功能是**识别**，将输入的机器码识别为相应的指令；或逻辑的功能是**生成**，根据输入的指令的不同，产生不同的控制信号。这种拆分使得两部分逻辑目的明确，这是一种朴素的**抽象**与**模块化**，希望同学们能够在其他的领域也加以借鉴。（这种思想的集大成者是 UNIX， 感兴趣的同学可以搜索"UNIX 哲学"加深了解）



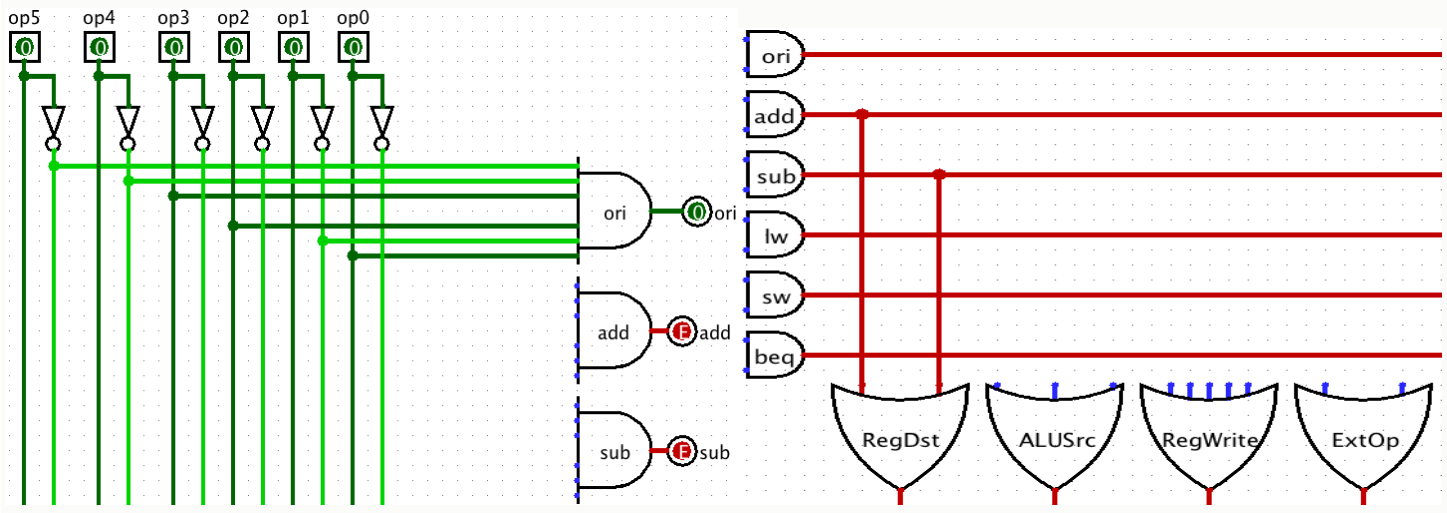
而在设计这两套逻辑的过程中，和逻辑是非常自然的。而或逻辑则需要我们建立从指令到控制信号的映射，为了避免错误的产生，我们希望使用真值表来完成相应的设计任务，并希望通过真值表可以简化相应的逻辑。一个典型的真值表如下图：

See MIPS Green Sheet → func / op

	add	sub	ori	lw	sw	beq
func	10 0000	10 0010	n/a			
op	00 0000	00 0000	00 1101	10 0011	10 1011	00 0100
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

下图是上述逻辑在 Logisim 中的电路具体实现样例——**与或门阵列**（并不完整），当然你也可以有自己的电路设计。



文档撰写建议

- **请务必给出控制信号产生的真值表作为控制器的功能定义。**
- 不妨对各个控制信号的意义也加以说明（图或文字），这样能够更加清楚地表达各个指令的功能。

本节思考题

1. 结合上文给出的样例真值表，给出 RegDst， ALUSrc， MemtoReg，RegWrite, nPC_Sel, ExtOp 与 op 和 func 有关的布尔表达式（表达式中只能使用“与、或、非”3 种基本逻辑运算。）
2. 充分利用真值表中的 X 可以将以上控制信号化简为最简单的表达式， 请给出化简后的形式。
3. 事实上，实现 nop 空指令，我们并不需要将它加入控制信号真值表，为什么？请给出你的理由

测试 CPU

在完成 CPU 搭建后，进行 CPU 的测试是必要的一个环节，它可以检查出我们 CPU 设计和搭建的错误。对于测试程序的设计，我们有如下建议：

- 所有指令都应被测试充分。
- 在 MARS 中编写测试程序并调试通过。
- 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!
- 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。（提示:事实上,在现代主流计算机中,数据存储器和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。）

文档撰写建议

请在文档中附上你的测试程序，并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。

本节思考题

1. 前文提到，“可能需要手工修改指令码中的数据偏移”，但实际上只需再增加一个 DM 片选信号,就可以解决这个问题。请阅读相关资料并设计一个 DM 改造方案使得无需手工修改数据偏移。
2. 除了编写程序进行测试外，还有一种验证 CPU 设计正确性的办法——形式验证。 **形式验证**的含义是根据某个或某些形式规范或属性，使用数学的方法证明其正确性或非正确性。请搜索“形式验证（Formal Verification)"了解相关内容后，简要阐述相比与测试，形式验证的优劣。

CPU 设计文档及思考题提交入口

(本题共有 1 分)

注意事项

- 建议将提交内容分为 **CPU 设计文档(必须包含模块规格、控制器设计)**与**思考题**两部分。
- 请尽可能完成所有的思考题
- 由于本内容将会在课上检查，建议使用 pdf 等不依赖软件的格式
- 请按照**计算机组成原理实验报告撰写排版规则**中的格式要求来撰写实验报告。

未答复
检查答案你的回答

CPU 电路源文件提交入口

(本题共有 1 分)

评测说明

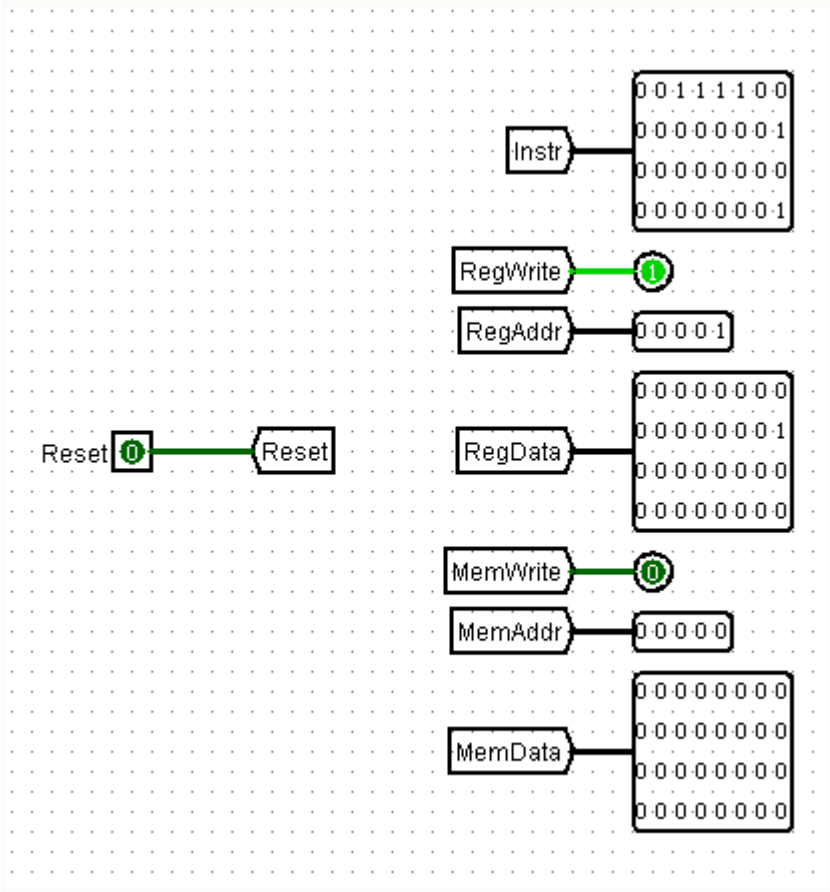
为了鼓励同学们编写自己的测试程序，对于评测机使用的评测程序，我们仅给出机器码来进行本地测试。

测试代码机器码下载地址

评测机将通过如下读取以下信号来对正确性进行判断，因此请保证在顶层视图中有如下信号（作为子电路的端口）

- Instr: 32 位指令信号
- RegWrite: GRF 写入控制信号
- RegAddr: GRF 5 位写入地址
- RegData: GRF 32 位写入数据
- MemWrite: DM 写入控制信号
- MemAddr: DM 5 位写入地址
- MemData: DM 32 位写入数据

我们建议你使用 Tunnel 将相关信号引出，如下图所示。但无论你使用什么方法，请确保输入输出的相对位置、位宽与下图相同。



注意事项

- 顶层模块名称： main
- 请提交单一一个.circ 文件。
- MOOC 返回的错误信息有一定的参考价值，但请不要完全依赖它！