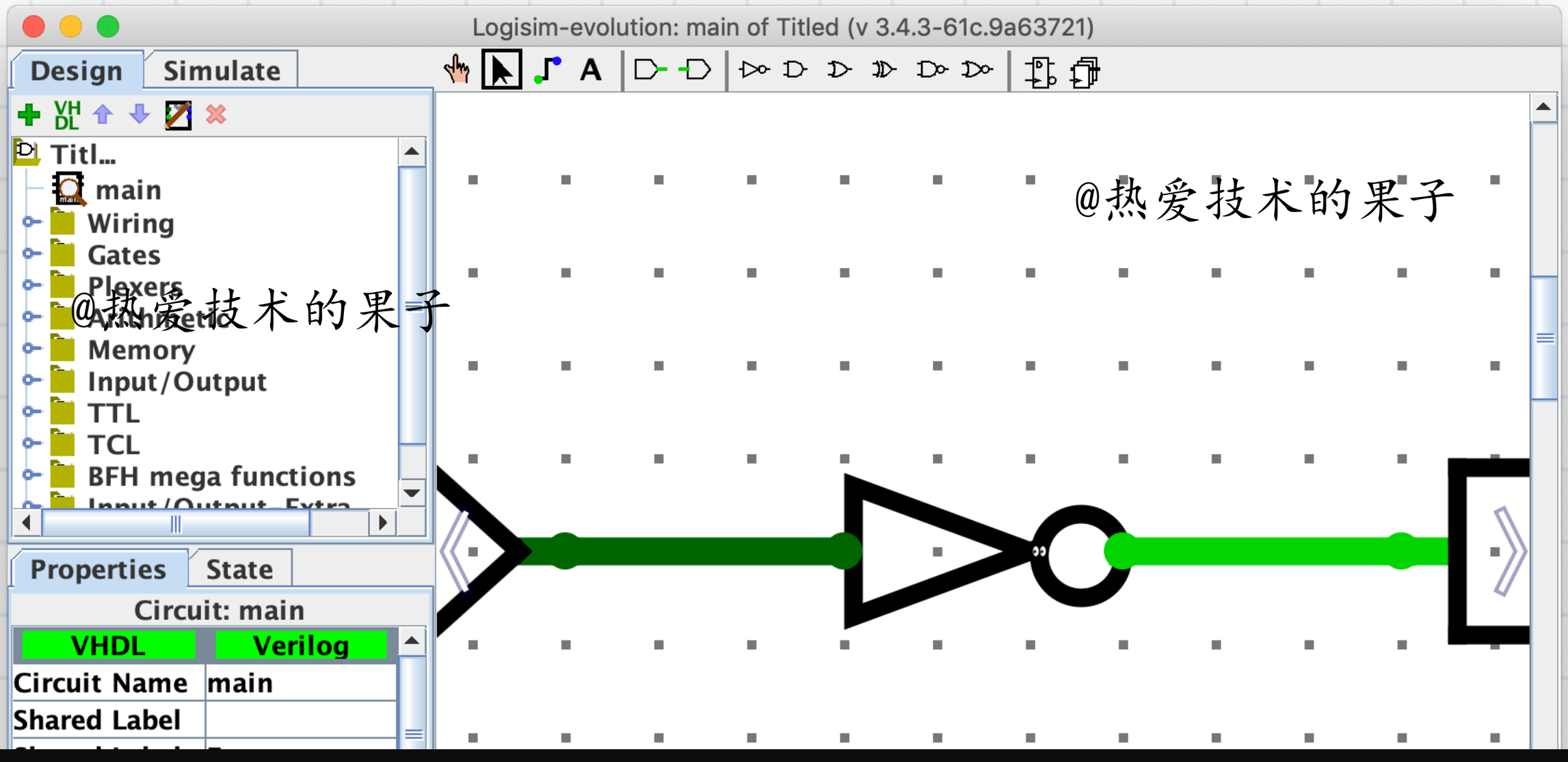
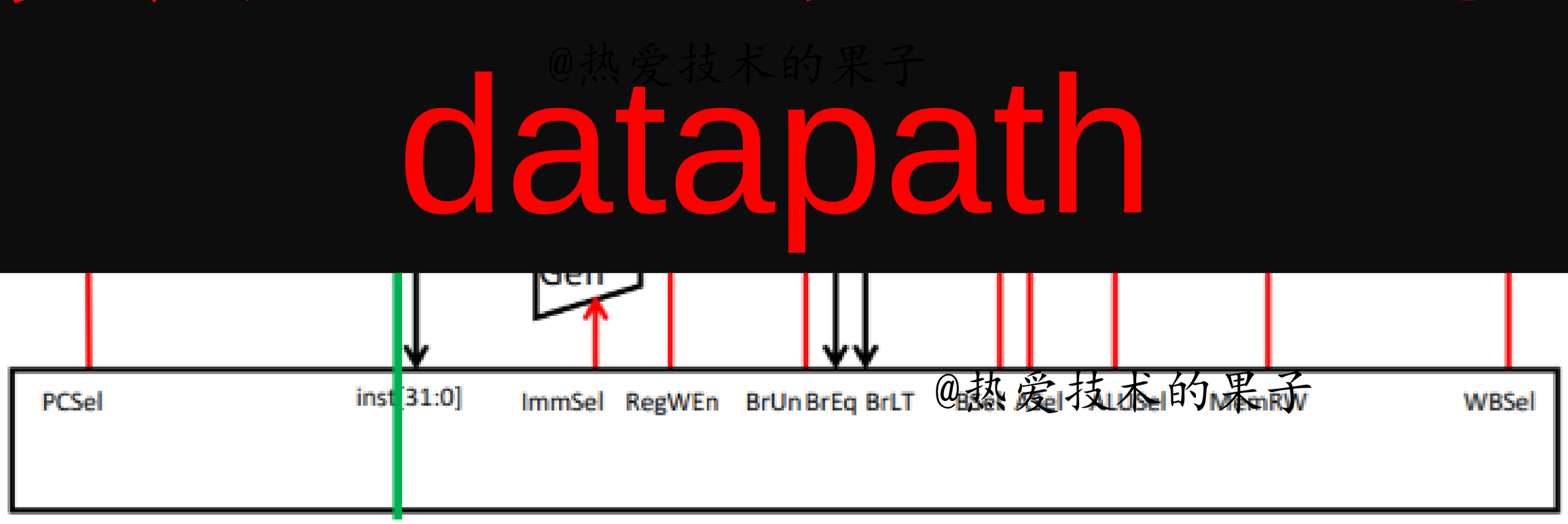


果子老师硬核分享搭建自己的CPU

（六）

实现CPU完整电路

datapath



什么是datapath？

本实验所有电路图都是使用 Logisim Evolution v3.8 完成。

CPU datapath（数据通路）是处理器内部用来执行指令、传输和处理数据的核心硬件路径。简单说，它就是 CPU 内部“数据流动的道路”，由一系列硬件部件和连线构成。

@热爱技术的果子

Datapath组成结构 @热爱技术的果子

寄存器文件（Register File）
存放操作数（数据）和计算结果。
提供快速读写能力。

算术逻辑单元 ALU（Arithmetic Logic Unit）
执行加减乘除、与或非、比较等运算。
ALU 的输入来自寄存器或立即数，输出写回寄存器或存储器。

多路选择器 MUX（Multiplexer）
控制数据选择路径，比如 ALU 的输入是来自寄存器还是立即数。

程序计数器 PC（Program Counter）
指向下一条要执行的指令地址。

存储器接口（Memory Access）
与指令存储器和数据存储器交互，完成取指令、读数据和写数据。

立即数生成器 / 移位器（Immediate Generator / Shifter）
从指令中提取立即数，或对数据进行位移操作。

工作流程

数据通路的工作方式执行一条指令时，数据在 datapath 中依次流动：

- 取指令（Instruction Fetch）：PC 把地址送到指令存储器，读出指令。
- 译码（Instruction Decode）：从寄存器文件中读操作数，或生成立即数。
- 执行（Execute）：ALU 进行计算，或生成访问内存的地址。
- 访存（Memory Access）：如需要，读/写数据存储器。
- 写回（Write Back）：结果写回寄存器。

小白怎么从零构建

要想自己从零搭建CPU的data path是非常难的！！！

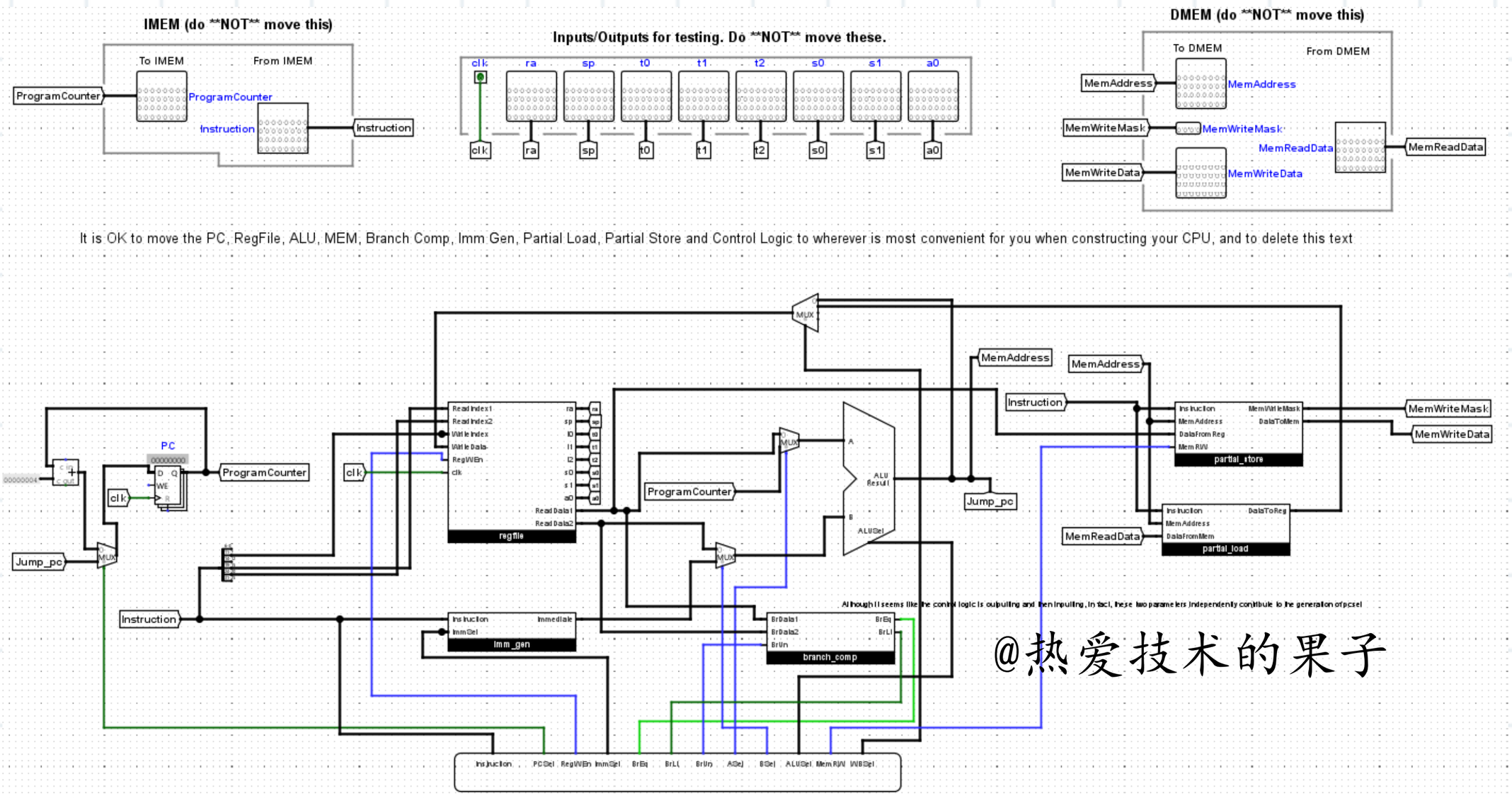
但是分享一些我给学生指导常用的技巧，电脑的CPU负责执行非常多的命令，搭建数据通路的时候不要想着一次性实现所有的命令

同时最重要的，依次搭建每个命令的电路时候应该一次思考下述几个问题，想清楚了再搭建！！！（很多学生按照我的思路一次性就可以成功）

（1）一条32bit的指令由哪些部分组成？每部分取值从哪获取
（2）执行阶段使用ALU的话，应该给ALU传入什么内容？
（3）是否涉及内存相关的操作，如果涉及还需要学习如何使用DMEM模块
（4）结果是否要保存到寄存器？

这些内容我在之前的系列分享都有讲解，左滑给大家讲解~

本次实验最终的Datapath 路线图



步骤一：指令构成

这里以实现CPU执行addi命令的datapath为例，给大家讲解一下按照我之间说的四个思考步骤，怎么实现一个完整的addi命令执行通路：

（1）addi 是哪一类指令？该指令包含哪些不同的字段？每个字段对应哪些比特位？

位段 (bit)	字段名	说明
[31:20]	imm[11:0]	12 位有符号立即数
[19:15]	rs1	源寄存器 1 (source register 1)
[14:12]	funct3	功能码（区分 I-type 指令），addi = 000
[11:7]	rd	目标寄存器 (destination register)
[6:0]	opcode	操作码，addi 属于 I-type，值为 0010011

（2）在 Logisim 中，你会使用哪种工具来把不同的比特组分离出来？

！！！使用分离器（splitter）从指令中提取出 5 个字段。

（3）哪些字段应当连接到寄存器文件？它们应当连接到寄存器文件的哪些输入端口？

1. 从指令中分离出来的 rs1 比特应连接到寄存器文件的 ReadIndex1。
2. 从指令中分离出来的 rd 比特应连接到寄存器文件的 WriteIndex。
3. I 型指令没有 rs2，所以我们现在可以忽略 rs2。
4. 记得把时钟信号连接到寄存器文件！

其余步骤

在这一步，我们将使用已解码的指令字段来实现实际的指令的运算

（1）如何获取我们需要的imm立即数

将指令（Instruction）连接到立即数生成器（immediate generator）。你在上一个任务中制作的立即数生成器应接收指令并输出正确的立即数。

（2）addi指令应向 ALU 输入哪两个数据值（A 和B）

输入 A 应该是来自寄存器文件的 ReadData1。
输入 B 应该是来自立即数生成器的立即数。

（3）指令应向 ALU 输入什么 ALUSel 值?

ALUSel 决定 ALU 执行哪种运算。由于我们现在只关注实现 addi，所以可以将 ALU 硬编码为始终选择加法操作（ALUSel = 0b0000）

（4）addi 指令写入的数据是什么？该指令将这些数据写入哪里?

addi 指令将加法运算的结果（来自 ALU 输出）写入寄存器 rd。将 ALUResult 连接到寄存器文件的 WriteData。由于 addi 指令总是写入寄存器，你现在可以将 RegWEn 硬连为 1，以始终启用寄存器写入。