SI100B EE Part Project • Tutorial 3

R Type & I Type Datapath

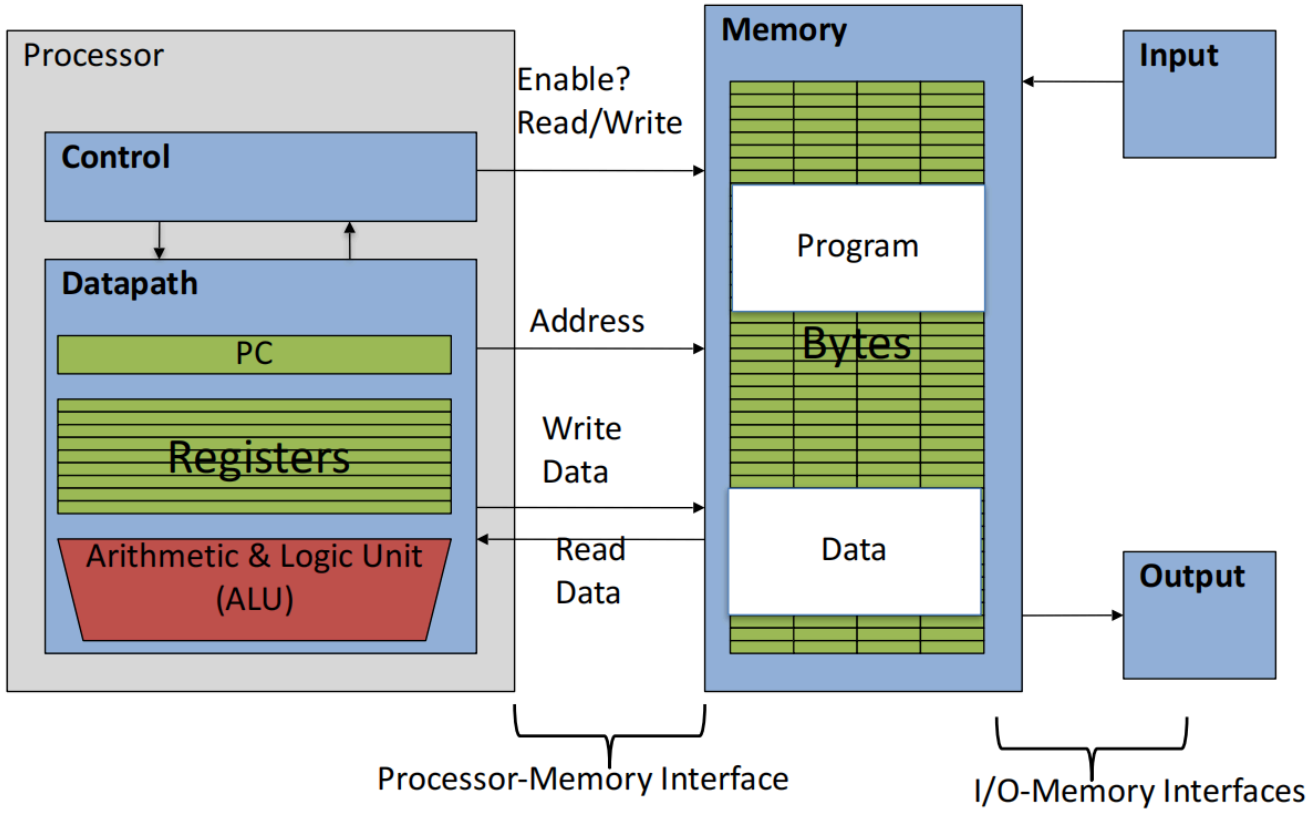
@Yunxiang He, heyx1@shanghaitech.edu.cn
@Quanyu Chen, chenqy2022@shanghaitech.edu.cn
@Yongzhi Zhang, zhangyz2022@shanghaitech.edu.cn
@Chaofan Li, lichf1@shanghaitech.edu.cn
@Qihan Ding, dingqh@shanghaitech.edu.cn

Contents

1. Datapath Introduction ->
2. Start with basic components ->
3. R-Type Instruction Datapath ->
4. I-Type Instruction Datapath ->
5. Venus Guide ->
6. Todo This week ->

1. Datapath Introduction

Components of a Computer



Processor（处理器）

- **Control Unit:**（控制单元，Controller）
  - Decodes instructions.
  - Controls the operations.
- **Datapath:**（数据通路）
  - PC: Program Counter.
  - Registers: also called register files.
  - ALU: Arithmetic & Logic Unit .

Memory

- Program area: for storing instructions.
- Data area: for storing data.

1. Datapath Introduction

Controller

- 对指令进行译码，生成指令对应的控制信号，控制数据通路的动作。
- 能对执行部件发出控制信号，是**指令的控制部件**。

Datapath

- 指令执行过程中，数据所经过的路径，包括路径中的部件。
- 它是**指令的执行部件**。

1. Datapath Introduction

- 数据通路由两类部件组成
  - 组合逻辑元件（也称操作元件）
  - 存储元件 （也称状态元件）
- 数据通路如何构成?
  - 由“操作元件”和“存储元件”连接而成
- 数据通路的功能是什么?
  - 进行数据存储、处理、传送

因此，数据通路是**由操作元件和存储元件连接而成**的**进行数据存储、处理、传**送的路径。

2. Start with basic components

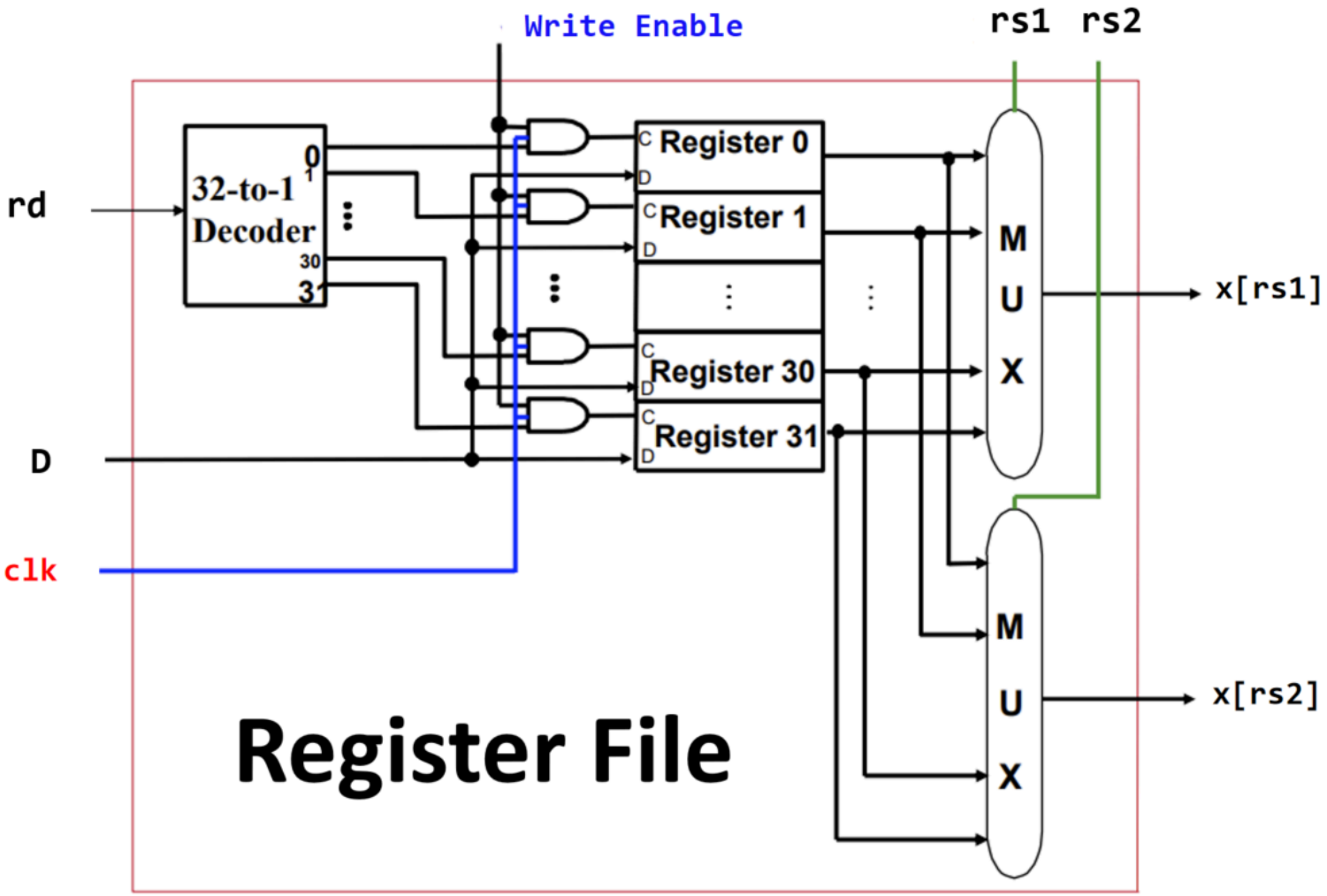
2. Start with basic components

回顾：我们完成了哪些工作?

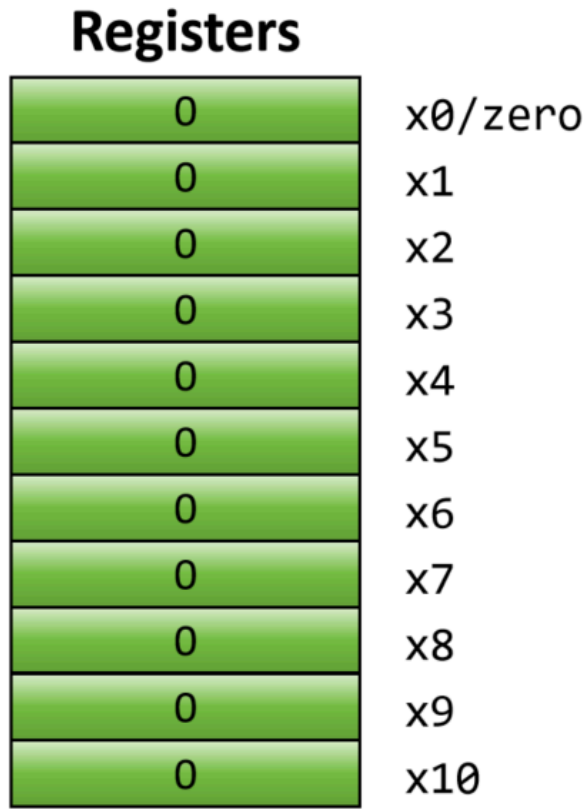
- 一个 Register File 模块
- 一个 ALU 模块
- 一个 PC （程序计数器）

☒ 接下来，我们利用这些模块，开始构建我们的 miniCPU!

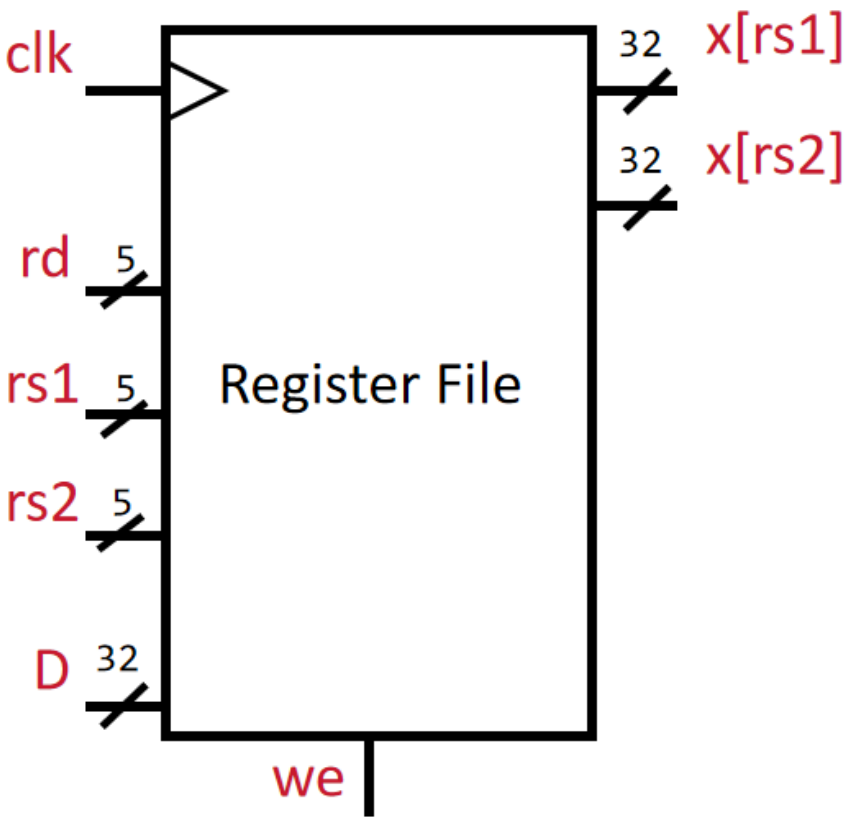
2.1. Full Register File



As we mentioned last week:



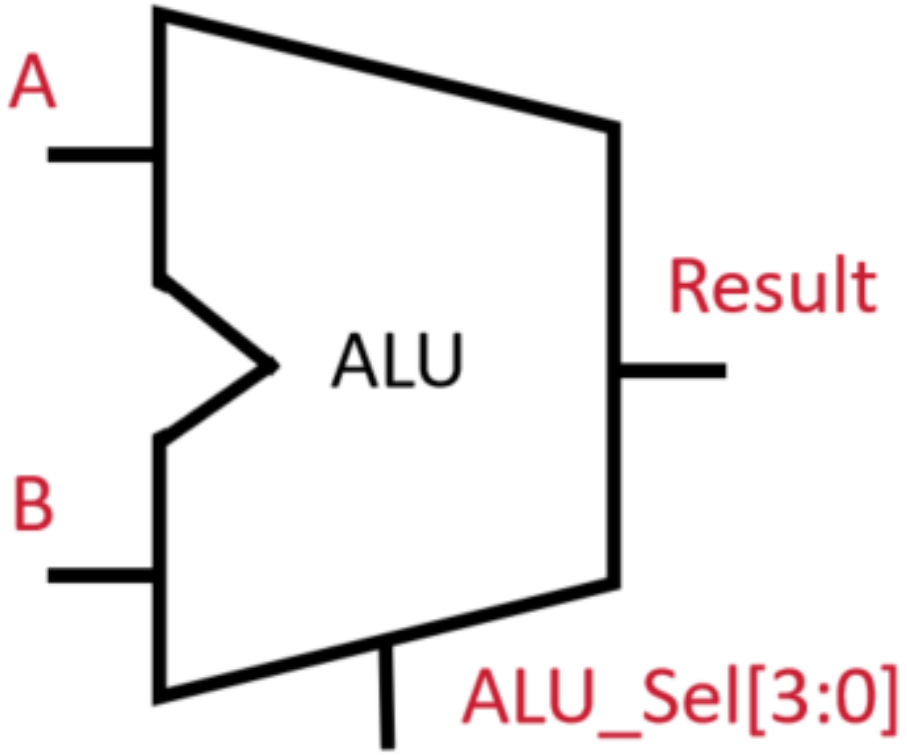
2.1. Full Register File



Signals involved:

1. `clk` (Clock)
2. `rd` (Register Destination)
3. `we` (Write Enable)
4. `D` (Write Data)
5. `rs1` (Register 1)
6. `rs2` (Register 2)
7. `x[rs1]` (Register 1 Output)
8. `x[rs2]` (Register 2 Output)

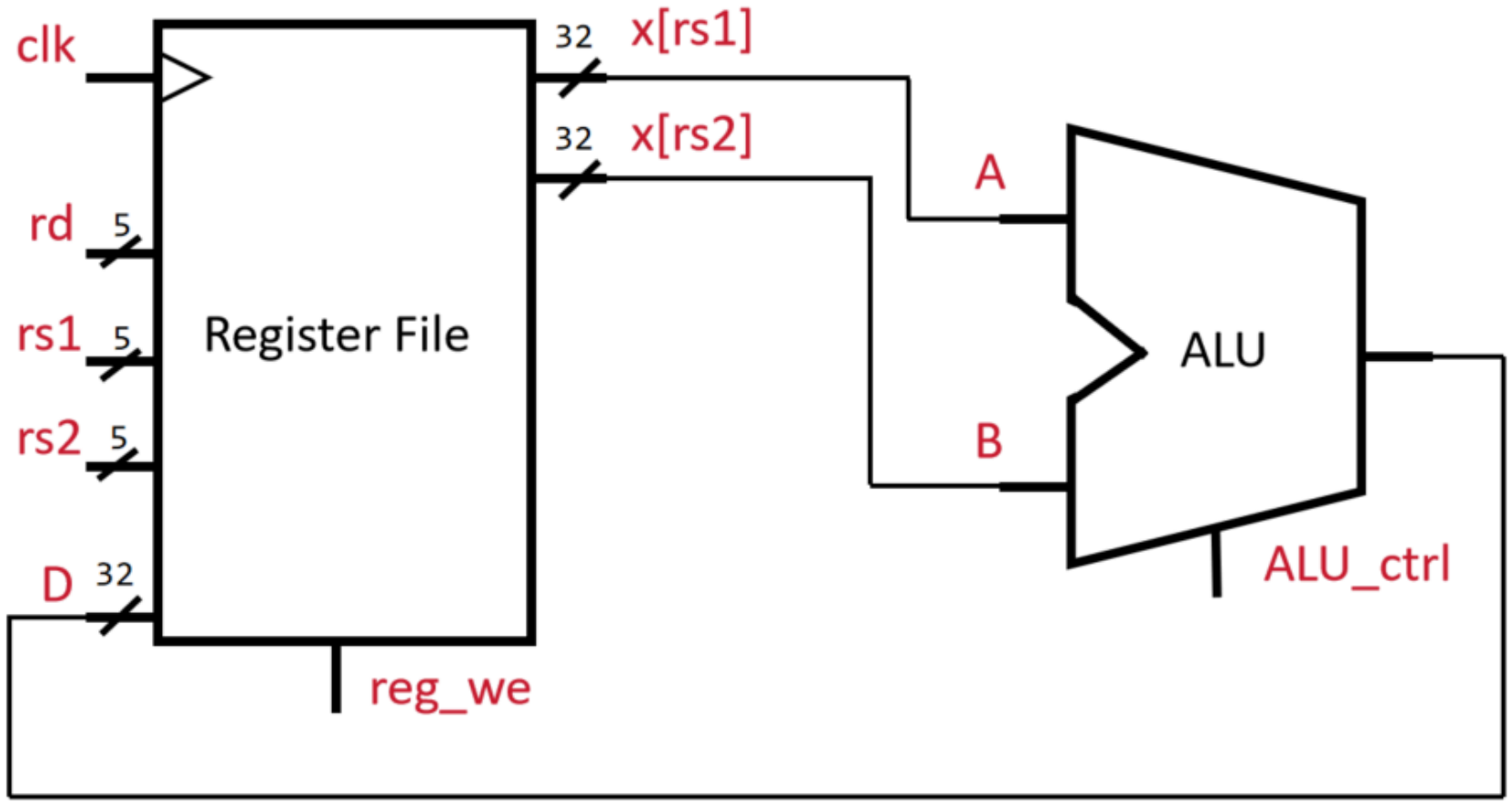
2.2. ALU



Signals involved

1. `A` (Input A)
2. `B` (Input B)
3. `Result` (Output)
4. `ALU_Sel` (Operation Select)

2.3. Connect the Components



3. R-Type Instruction Datapath

3. R-Type Instruction Datapath

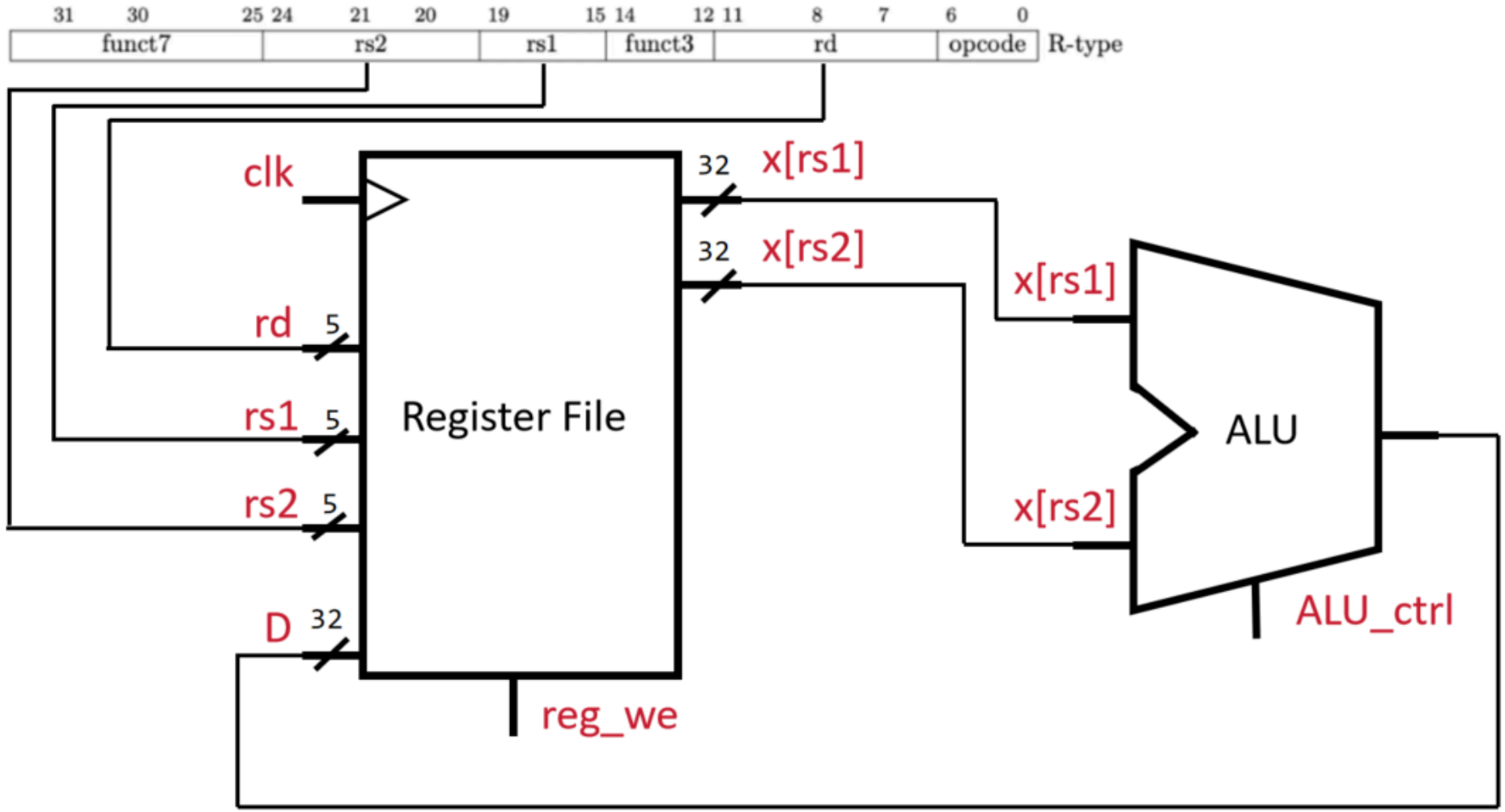
R-Type 指令是寄存器之间的运算指令。

在电路中实现 R-Type 指令的数据通路，需要完成以下几个步骤：

1. 从指令中提取出 rs1, rs2, rd 三个寄存器的编号。
2. 从寄存器文件中读取 rs1, rs2 两个寄存器的值。
3. 将 rs1, rs2 两个寄存器的值输入到 ALU 中进行运算。
4. 将 ALU 的运算结果写回到 rd 寄存器中。

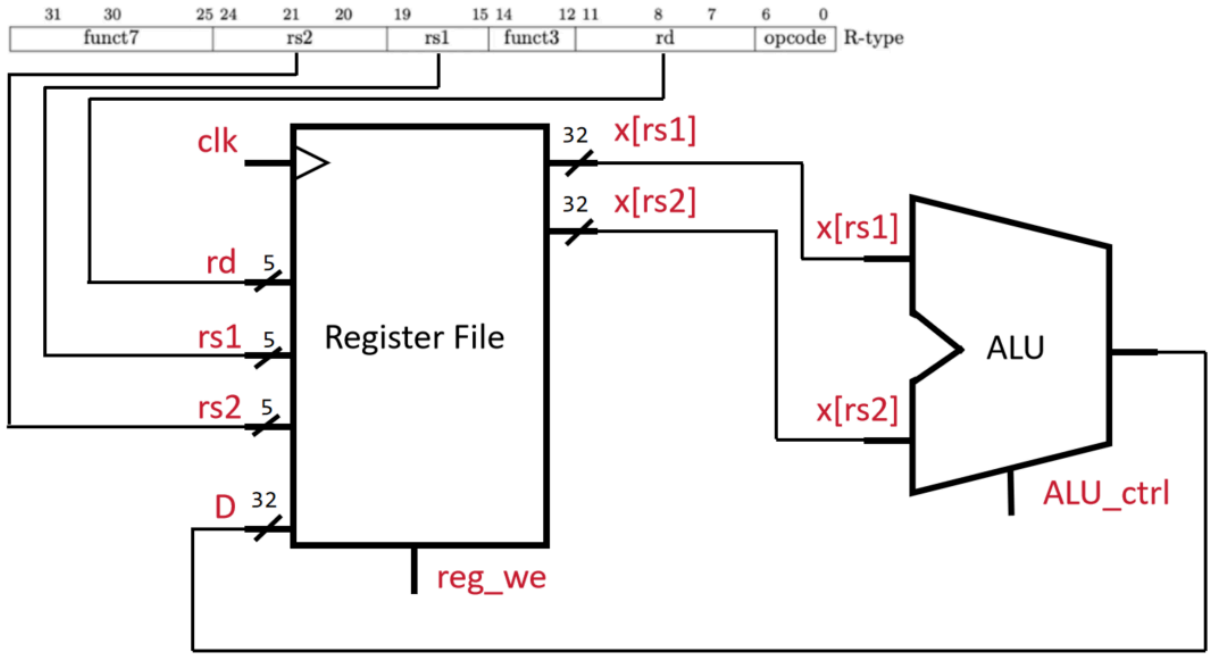
接下来，让我们看看 R-Type 指令的 Datapath 的模块图吧！

3. R-Type Instruction Datapath



3. R-Type Instruction Datapath

> “ Recall : Processor = PC + Register + ALU



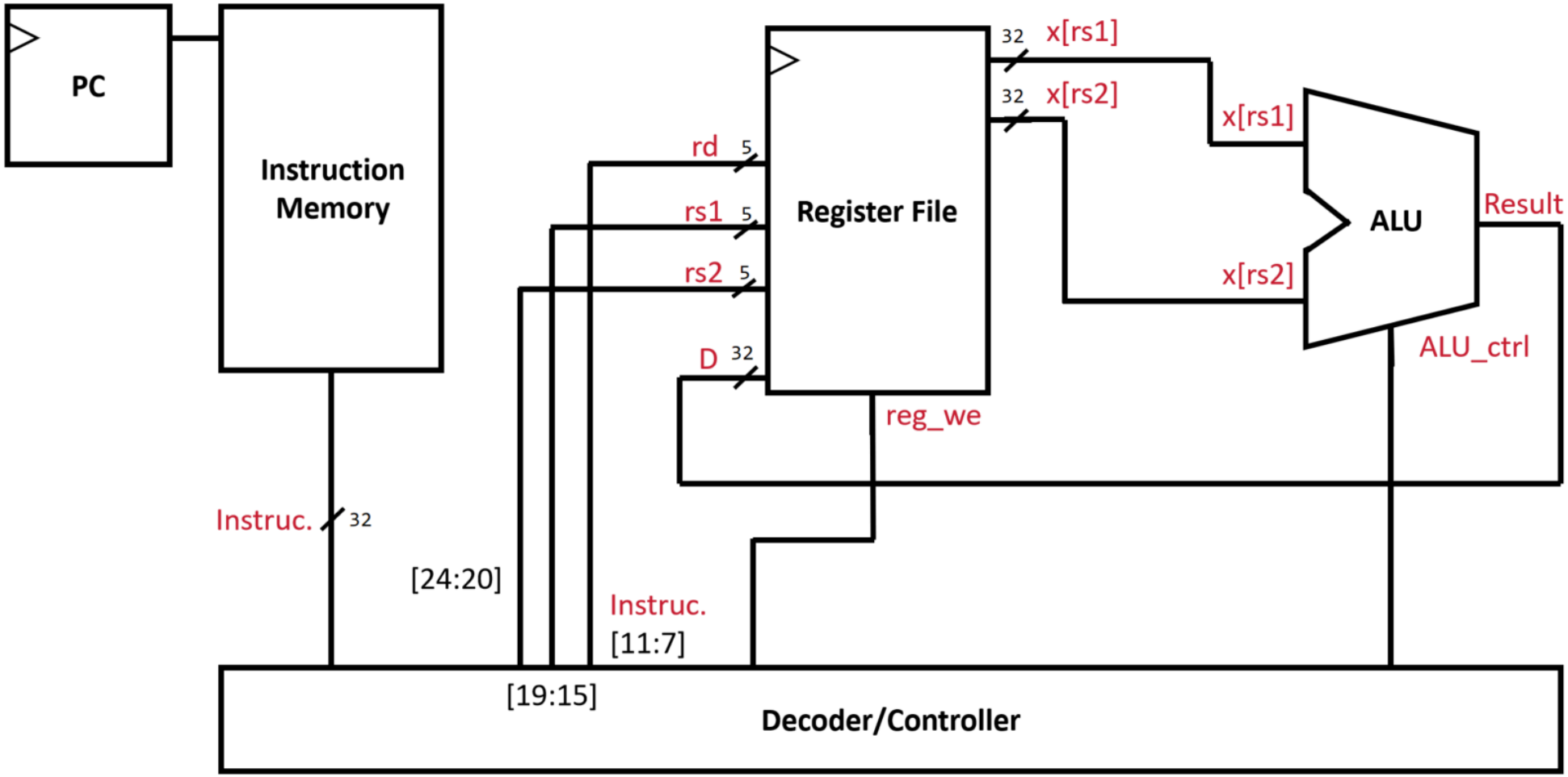
添加到 PC 中的 Decoder 用于解码指令，并向 Datapath 提供操作数、控制信号等，具体如下：

31-25	24-20	19-15	14-12	11-7	6-0
`funct7`	`rs2`	`rs1`	`funct3`	`rd`	`opcode`
7 bits	5 bits	5 bits	3 bits	5 bits	7 bits

3. R-Type Instruction Datapath

- 上图展示了直接从指令里提取出的 rs1, rs2, rd 三个寄存器的编号，并传递给寄存器文件的过程。
- 实际上，这个流程在电路里的完成模式是：
  1. 我们根据 PC 的值从 Instruction Memory 中读取出指令。
  2. 我们通过 Instruction Decoder 模块将指令的不同部分提取出来
- 因此，我们将上图等价地表示为下图，其中从指令中提取出的 rs1, rs2, rd 三个寄存器的编号的过程用实际的电路模块表示。

3. R-Type Instruction Datapath



4. I-Type Instruction Datapath

4. I-Type Instruction Datapath

I-Type 指令是立即数和寄存器之间的运算指令。

在电路中实现 I-Type 指令的数据通路，需要完成以下几个步骤：

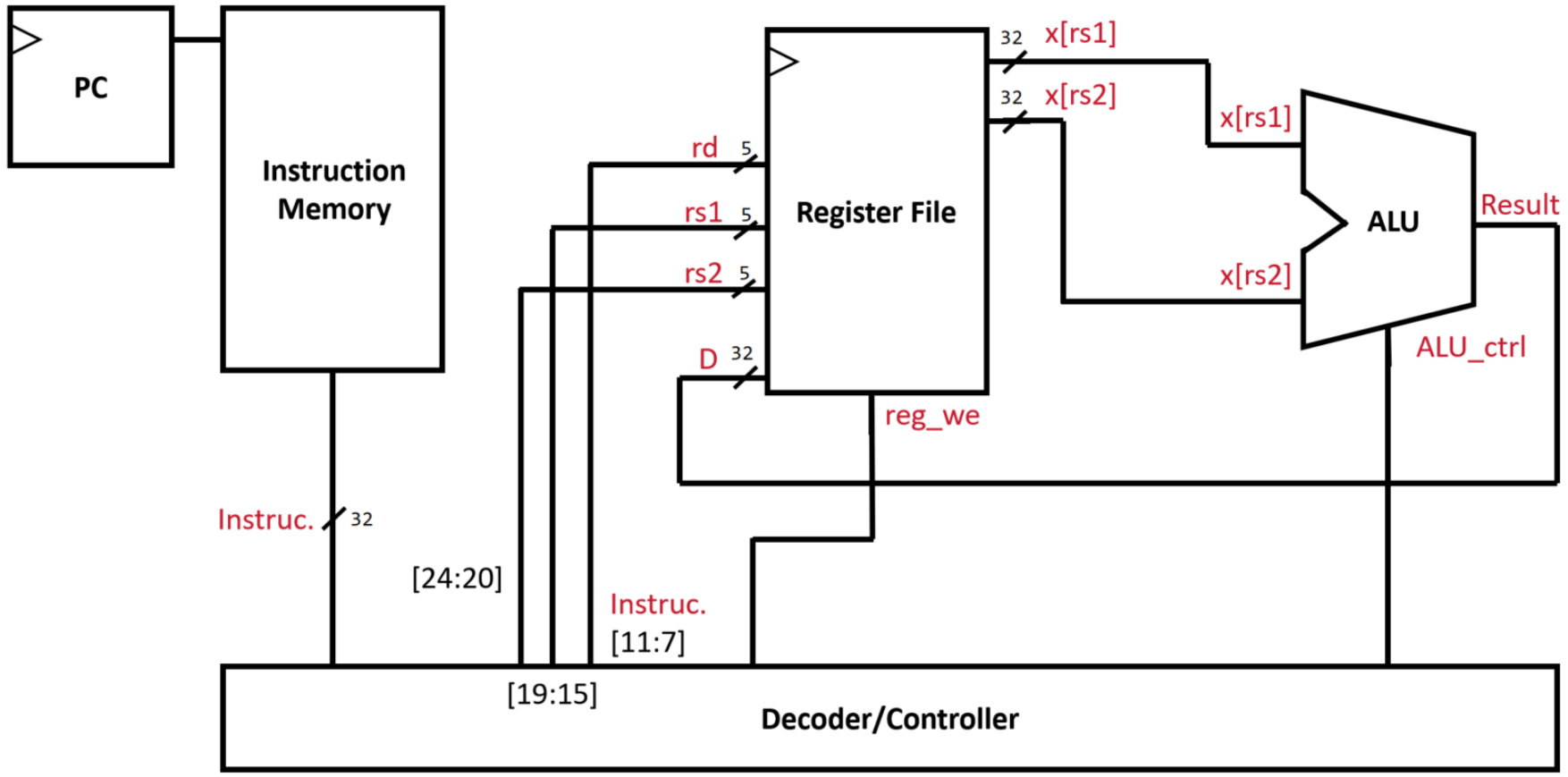
1. 从指令中提取出 rs1, rd 两个寄存器的编号，以及立即数。
2. 从寄存器文件中读取 rs1 寄存器的值。
3. 将 rs1 寄存器的值和立即数输入到 ALU 中进行运算。
4. 将 ALU 的运算结果写回到 rd 寄存器中。

可以观察到，I-Type 指令的数据通路和 R-Type 指令的数据通路有很多相似之处。

我们尝试扩展 R-Type 的 Datapath，来得到一个能够兼容两种 Datapath 的电路！

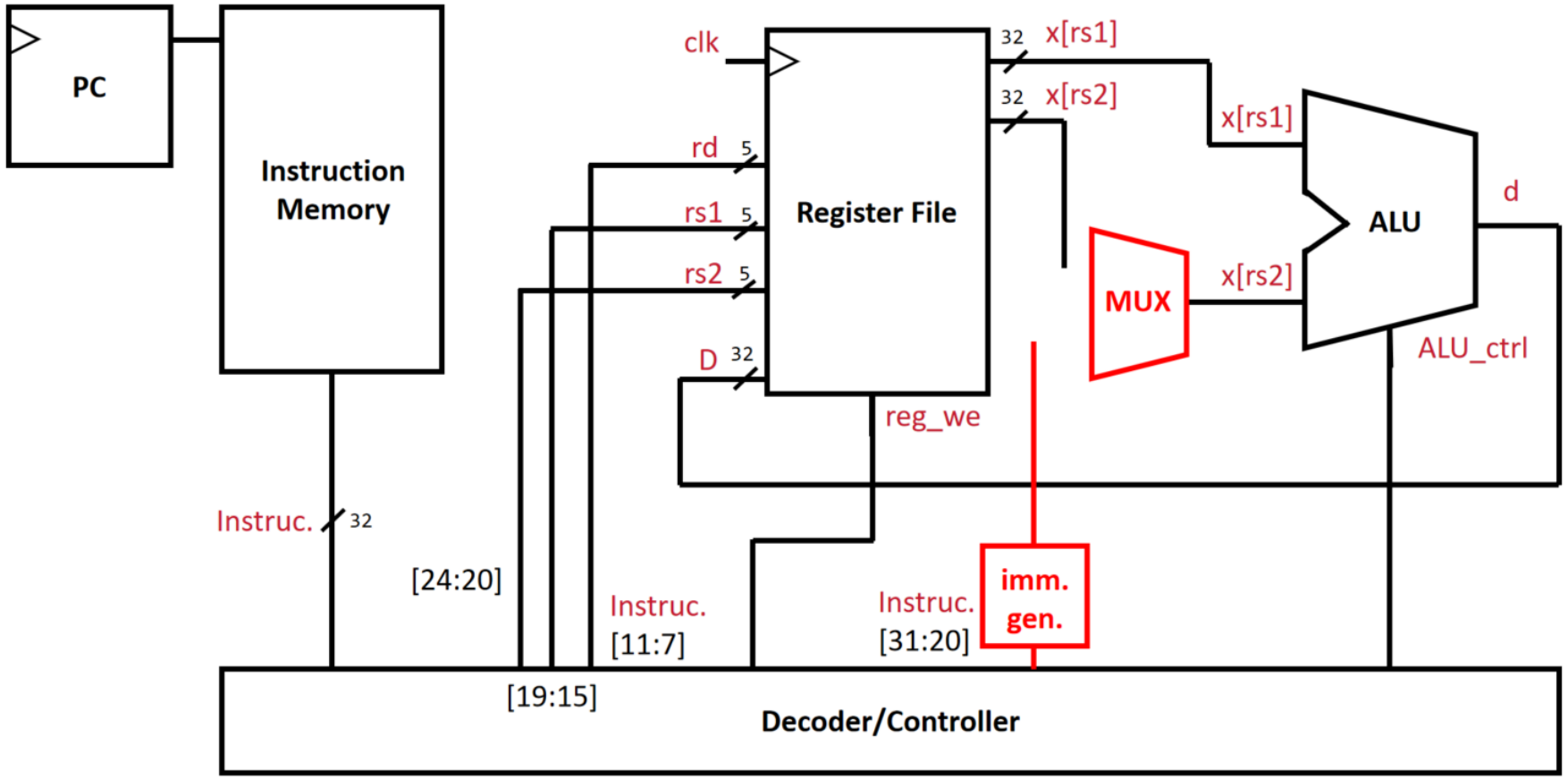
4. I-Type Instruction Datapath

原始的 R-Type 电路模块



4. I-Type

I-Type 的新增模块



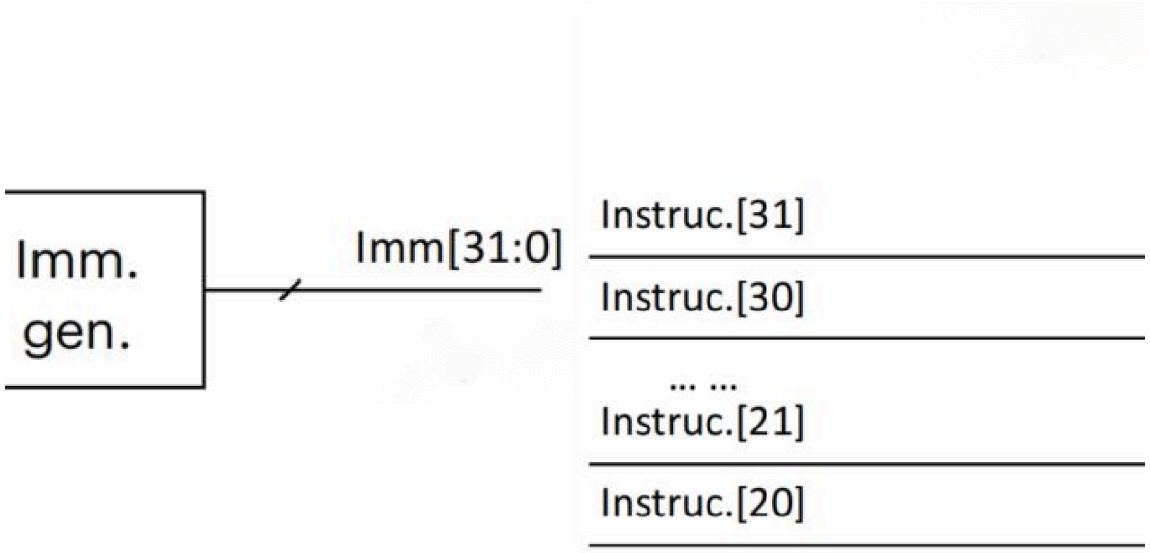
4. I-Type

Imm-Gen 的实现

Imm-Gen 是提取指令中的立即数部分的模块。

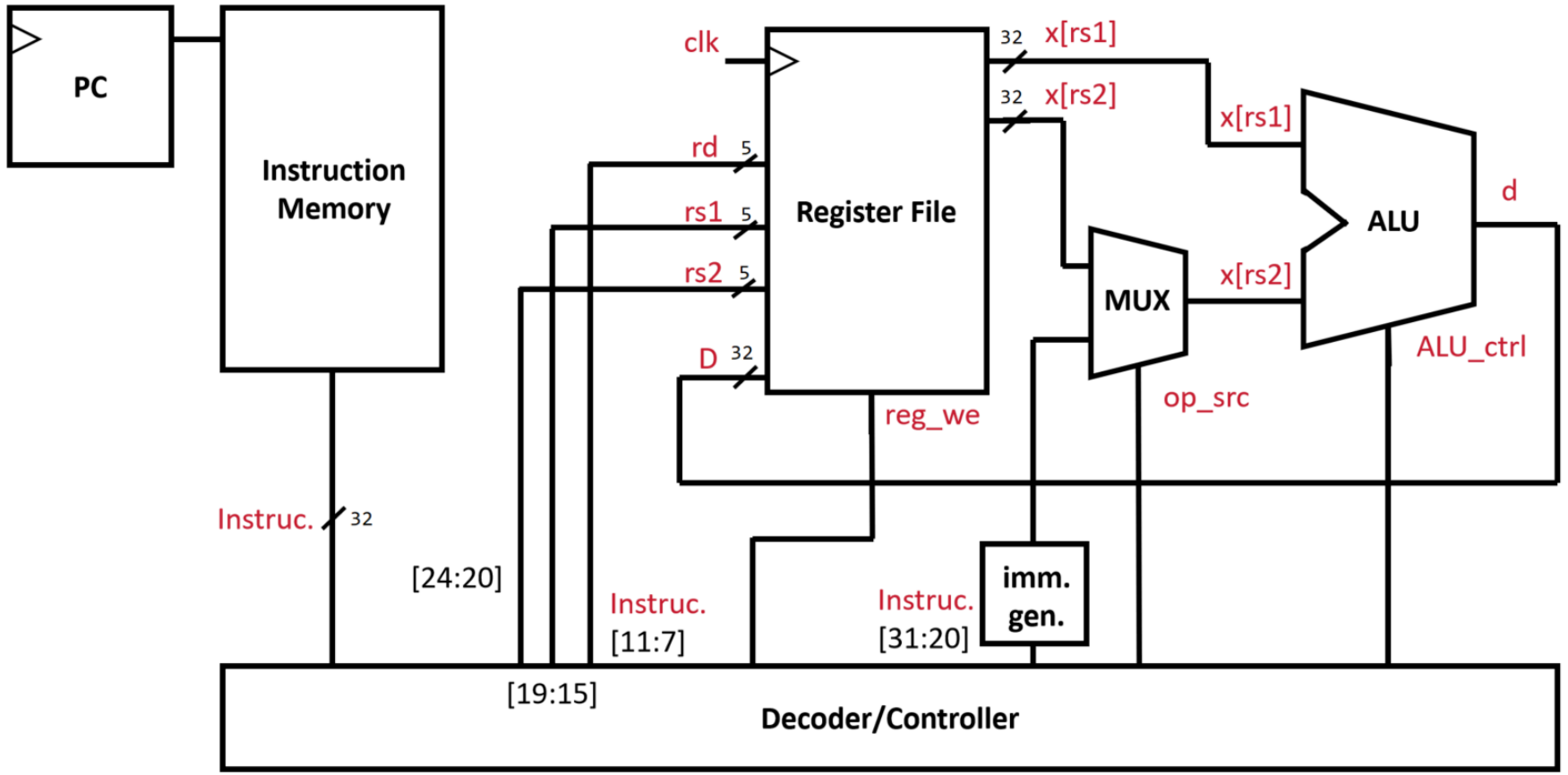
我们在这里以 I-Type 的立即数提取模块为例（还有 S-Type 也需要提取立即数）：

1. 其模块的输入为指令的 31-20 位，输出为一个 32 位的立即数
2. 我们直接拷贝其低 12 位，但高 20 位需要做额外处理：符号位拓展！



4. I-Type Instruction Datapath

完成的 I-Type 电路模块

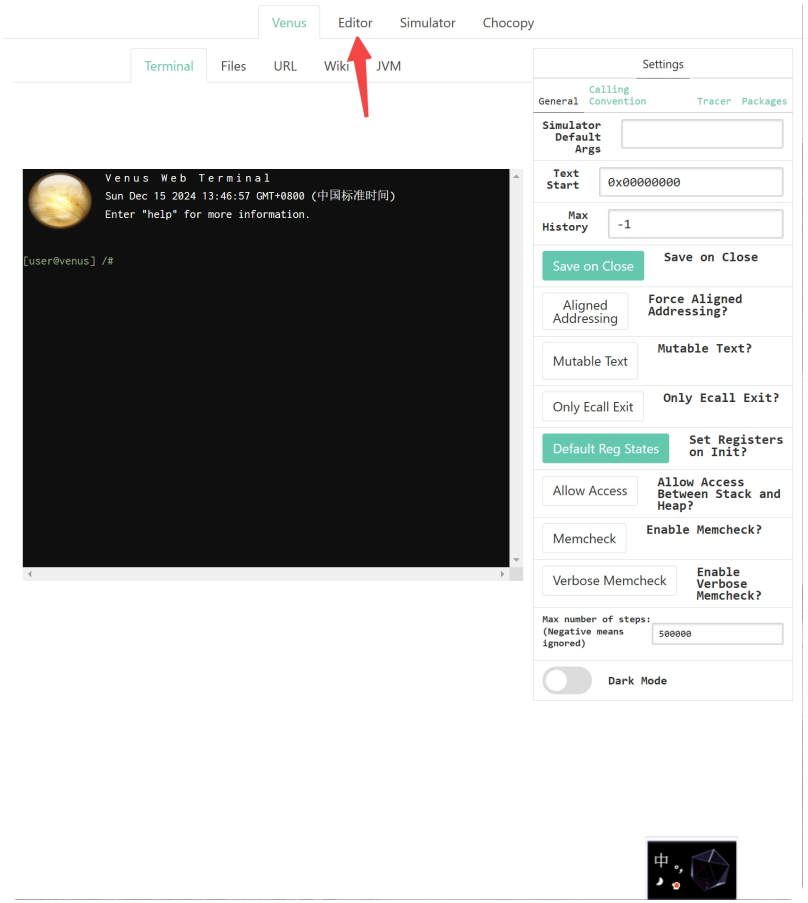


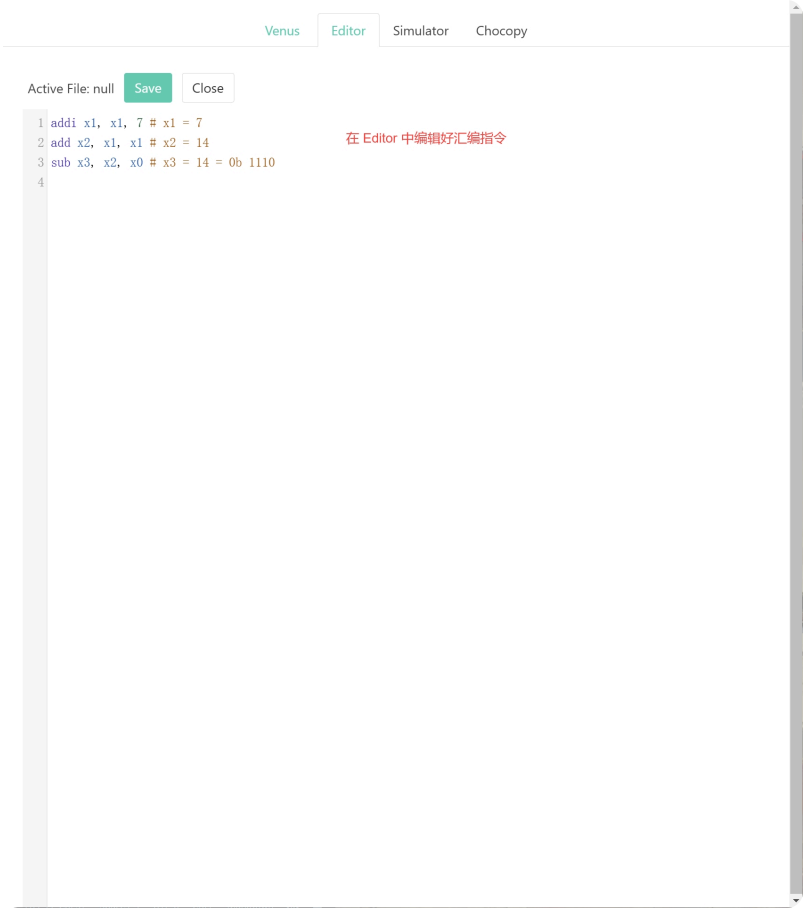
5. Venus Guide

5. Venus Guide

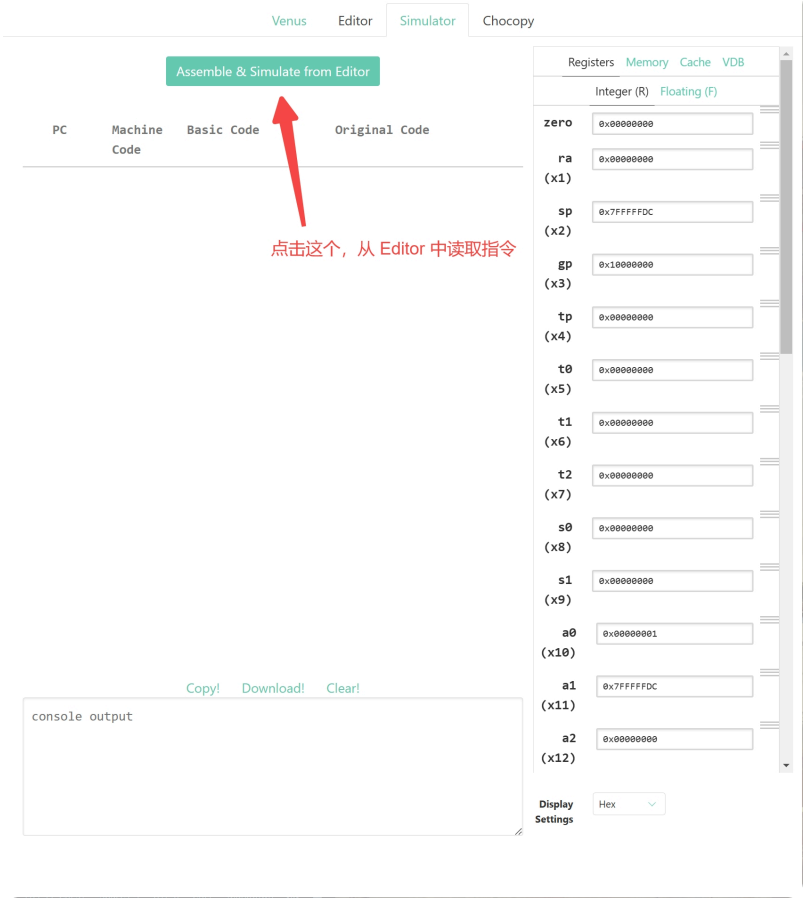
访问 Venus ->

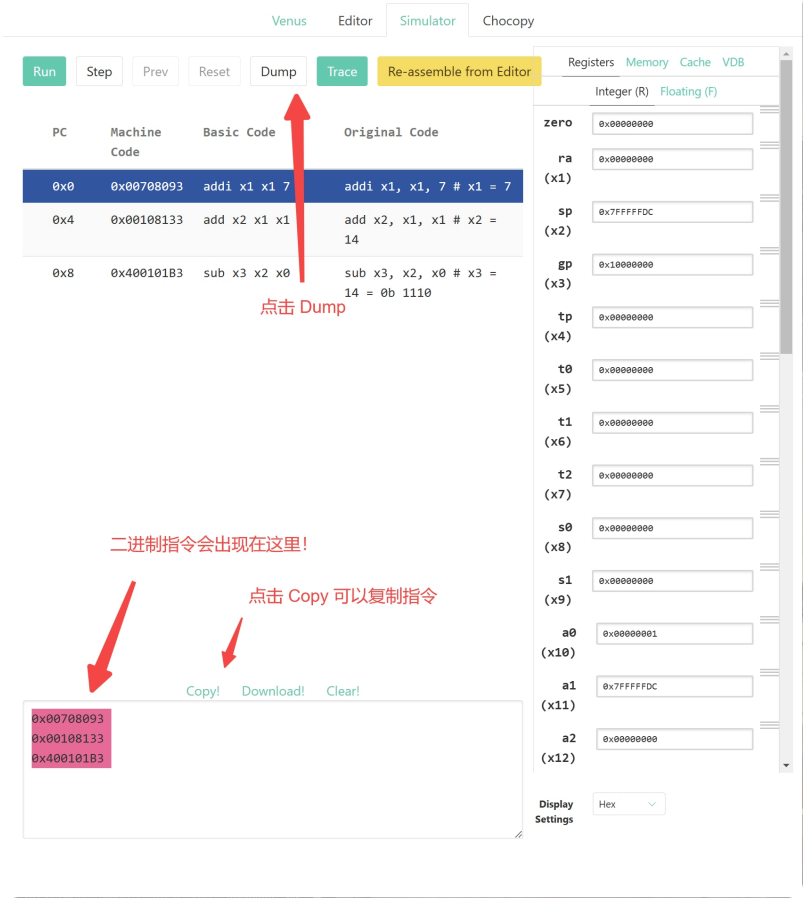
5. Venus Guide





5. Venus Guide





6. ToDO This week

6. ToDO This week

1. 完成基本模块（寄存器堆、ALU）
2. 完成支持 R-Type 和 I-Type 算术指令的电路
3. 尝试编写一组简单的指令，把它 Load 到电路中运行！

SI100B EE Part Project · Tutorial 3

@ Presented by **Yunxiang He**
@ ShanghaiTech University