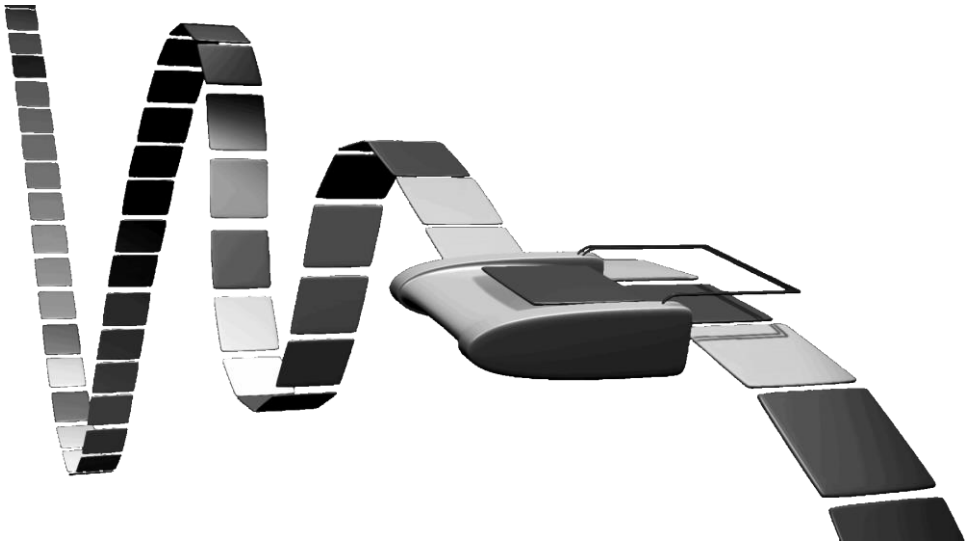
计算机的指令系统

2022年秋

内容概要

- 计算机程序及分类
- 指令系统基本知识
- RISC-V指令系统简介
- THINPAD RV32指令系统
- THINPAD指令模拟器

图灵和图灵机



Turing Machine
1937

Alan Turing

计算机程序

- **Computer programs**(also **software programs**, or just **programs**) are instructions for a computer. A computer requires programs to function, and a computer program does nothing unless its instructions are executed by a central processor. Computer programs are either executable programs or the source code from which executable programs are derived (e.g., compiled).

- 程序员和计算机硬件之间交互的语言
- 高级语言
- 汇编语言
- 机器语言

程序举例(sum.s)

```
int sum()
{
    int sum = 0;
    for (int i = 1; i <= 10; i++)
        sum += i;
    return sum
}
```

-Og版本

```
sum:
    li   a5,1
    li   a0,0
.L2:
    li   a4,10
    bgt  a5,a4,.L4
    add  a0,a0,a5
    addi    a5,a5,1
    j    .L2
.L4:
    ret
```

-O2版本

```
sum:
    li   a0,55
    ret
```

程序举例(fib.s)

```
int fibo[10];
void fib()
{
    int i;
    fibo[0]=1; fibo[1] =1;
    for ( i=2 ; i<10; i++)
     fibo[i]= fibo[i-1] + fibo[i-2];
}
```

```
    .align  2
    .globl  fib
    .type   fib, @function
fib:
    lui a5,%hi(fibo)          # high value of fibo in a5
    addi    a4,a5,%lo(fibo)   # set a4 = address of fibo
    li  a3,1                  # a3 = 1
    addi    a5,a5,%lo(fibo)   # set a5 = address of fibo
    sw  a3,0(a4)              # fibo[0] = 1
    sw  a3,4(a4)              # fibo[1] = 1
    addi    a2,a5,32          # a2 = address of fibo[8]
    li  a4,1                  # a4 = 1
    j   .L3                   # goto L3
.L5:
    lw  a4,4(a5)              # a4 = a5[1]
    lw  a3,0(a5)              # a3 = a5[0]
.L3:
    add a4,a4,a3              # a4 = a4 + a3
    sw  a4,8(a5)              # a5[2] = a4
    addi    a5,a5,4           # a5 += 4 (points to next )
    bne a5,a2,.L5             # not finished? goto L5
    ret
```

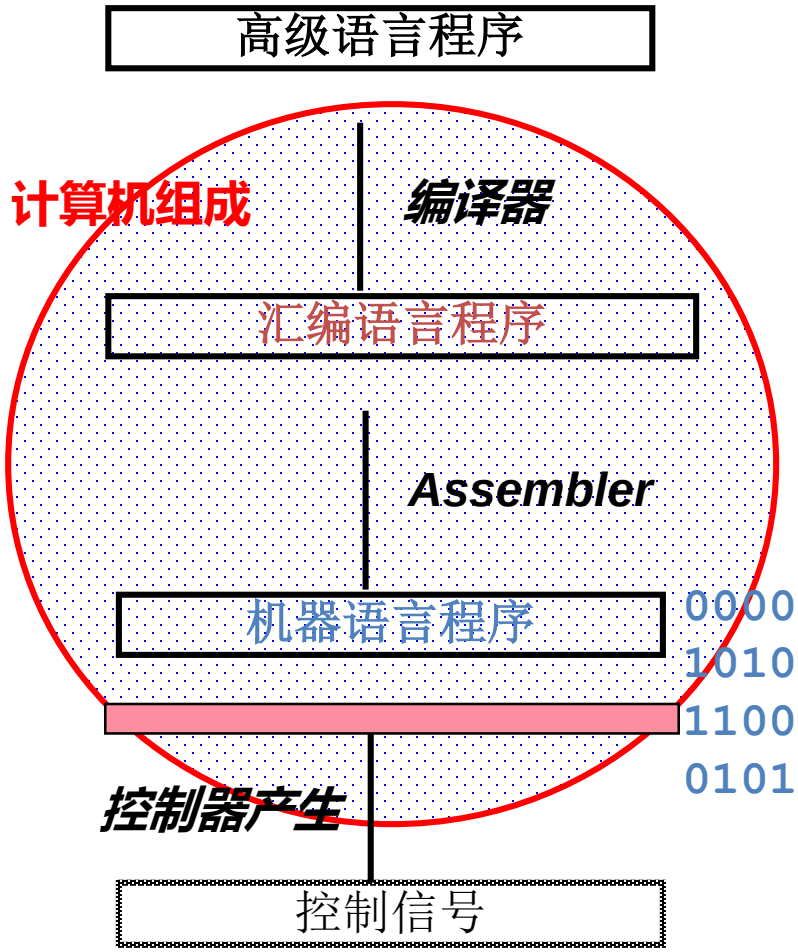
高级语言

- 高级语言又称算法语言，它的实现思路，不是过分地“靠拢”计算机硬件的指令系统，而是着重面向解决实际问题所用的算法，瞄准的是如何使程序设计人员能够方便地写出处理问题和解题过程的程序，力争使程序设计工作的效率更高。
- 用高级语言语言设计出来的程序，需要经过编译程序先翻译成机器语言程序，才能在计算机的硬件系统上予以执行，个别的选用解释执行方案。
- 高级语言的程序通用性强，在不同型号的计算机之间更容易移植。对高级语言进行编译、汇编后得到机器语言在计算机上运行。

汇编语言以及机器语言

- 汇编语言是对计算机机器语言进行符号化处理的结果,再增加一些为方便程序设计而实现的扩展功能。
- 在汇编语言中，可以用英文单词或其缩写替代二进制的指令代码，更容易记忆和理解；还可以选用英文单词来表示程序中的数据(常量、变量和语句标号)，使程序员不必亲自为这些数据分配存储单元，而是留给汇编程序去处理, 达到基本可用标准。
- 若在此基础上，能够在支持程序的不同结构特性（如循环和重复执行结构，子程序所用哑变元替换为真实参数）等方面提供必要的支持，使该汇编语言的实用程度更高。
- 汇编程序要经过汇编器翻译成机器语言后方可运行
- 机器语言是计算机硬件能直接识别和运行的指令的集合,是二进制码组成的指令，用机器语言设计程序基本不可行。
- 程序的最小单元是指令，同时，指令也是计算机硬件执行程序的最小单位。

不同层次的程序




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw t0, t2, 0
lw t1, t2, 4
sw t1, t2, 0
sw t0, t2, 4
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

Von Neumann结构计算机

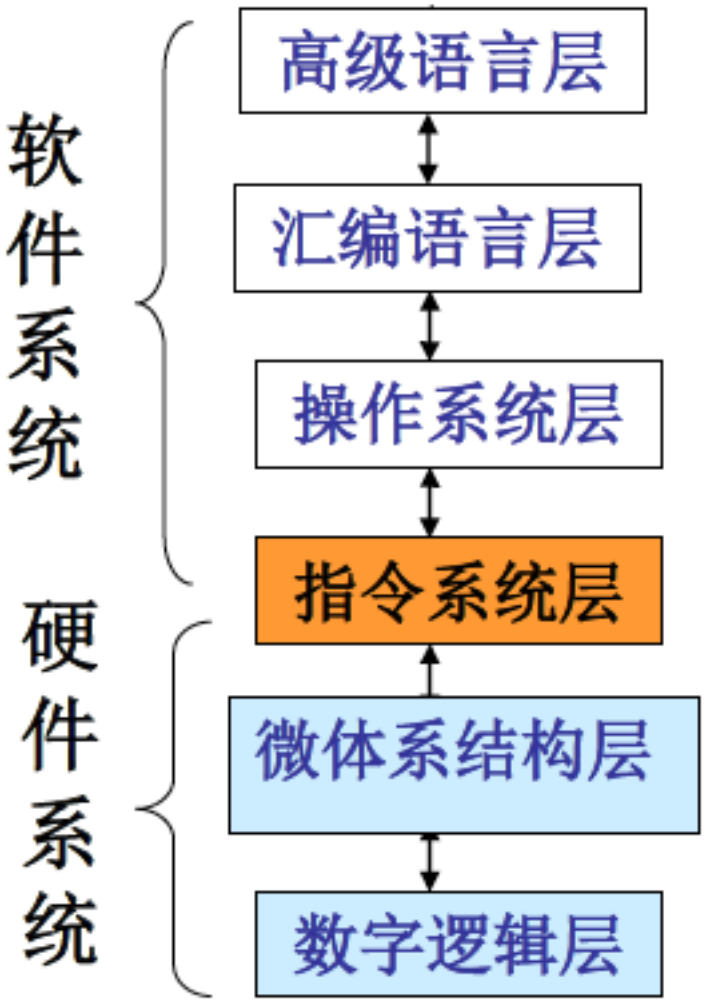
- 存储程序计算机
  - 程序由指令构成
  - 程序功能通过指令序列描述
  - 指令序列在存储器中顺序存放
- 顺序执行指令
  - 用PC指示当前被执行的指令
  - 从存储器中读出指令执行
  - PC指向下一条指令

指令和指令系统

- 计算机系统由硬件和软件两大部分组成。硬件指由中央处理器、存储器以及外围设备等组成的实际装置。软件是为了使用计算机而编写的各种系统的和用户的程序，程序由一个序列的计算机指令组成。
- 指令是计算机运行的最小的功能单元，是指挥计算机硬件运行的命令，是由多个二进制位组成的位串，是计算机硬件可以直接识别和执行的信息体。指令中应指明指令所完成的操作，并明确操作对象。
- 一台计算机提供的全部指令构成该计算机的指令系统。指令用于程序设计人员告知计算机执行一个最基本运算、处理功能，多条指令可以组成一个程序，完成一项预期的任务。

指令系统地位

- 可以从6 个层次分析和看待计算机系统的基本组成。
- 指令系统层处在硬件系统和软件系统之间，是硬、软件之间的接口部分，对两部分都有重要影响。
- 硬件系统用于实现每条指令的功能，解决指令之间的衔接关系。
- 软件由按一定规则组织起来的许多条指令组成，完成一定的数据运算或者事务处理功能。
- 指令系统优劣是一个计算机系统是否成功的关键因素。



计算机系统的层次结构

指令的功能分类

- 数据运算指令
  - 算术运算，逻辑运算
- 数据传输指令
  - 内存/寄存器，寄存器/寄存器
- 控制指令
  - 无条件跳转，条件跳转，子程序的支持（调用和返回）
- 输入输出指令
  - 与输入输出端口的数据传输（输入输出模型如何）
- 其它指令
  - 停机、开/关中断、空操作、特权指令、设置条件码

指令格式

- 指令格式：操作码，操作数地址的二进制分配方案

  - 操作码：指令的操作功能
  - 操作数地址：操作数存放的地址，或者操作数本身
- 指令字：完整的一条指令的二进制表示
- 指令字长：指令字中二进制代码的位数
  - 机器字长：计算机能够直接处理的二进制数据的位数
  - 指令字长（字节倍数）：0.5，1，2，……个机器字长
  - 定长指令字结构 变长指令字结构
  - 定长操作码 扩展操作码

寻址方式

- 寻址方式（又称编址方式）指的是确定本条指令的操作数地址及下一条要执行的指令地址的方法。
- 不同的计算机系统,使用数目和功能不同的寻址方式，其实现的复杂程度和运行性能各不相同。有的计算机寻址方式较少，而有些计算机采用多种寻址方式。
- 通常需要在指令中为每一个操作数专设一个地址字段，用来表示数据的来源或去向的地址。在指令中给出的操作数（或指令）的地址被称为形式地址，使用形式地址信息并按一定规则计算出来或读操作得到的一个数值才是数据（或指令）的实际地址。在指令的操作数地址字段，可能要指出：
- ①运算器中的累加器的编号或专用寄存器名称（编号）
- ②输入/输出指令中用到的I/O 设备的入出端口地址
- ③内存储器的一个存储单元（或一I/O设备）的地址

寻址方式

- 如何找寄存器?
- 如何找立即数?
- 如何找内存地址?
  - 直接给出内存地址
  - 通过寄存器进行偏移找出地址
- 如何找输入输出的端口地址?

评价计算机性能的指标

- 吞吐率
  - 单位时间内完成的任务数量
- 响应时间
  - 完成任务的时间
- 衡量性能的指标
  - MIPS
  - CPI
  - CPU Time
  - CPU Clock
- 综合测试程序

指令系统分类

- 复杂指令集（CISC：Complex Instruction Set Computing）
  - 特点是指令数目多而复杂，每条指令字长并不相等，计算机必须加以判断，并为此付出了性能的代价。
  - 例子：x86指令集
- 精简指令集（RISC：Reduced Instruction Set Computing）
  - 特点是对指令数目和寻址方式都做了精简，指令长度规整，长度相同（压缩指令除外），使其实现更容易，指令并行执行程度更好，编译器的效率更高。
  - 例子：MIPS指令集，RISC-V指令集，ARM指令集
- 超长指令字（VLIW：Very Long Instruction Word）
  - 将简短而长度统一的精简指令组合出超长指令，每次运行一条超长指令，等于并发运行多条短指令。
  - 例子：Intel安腾指令集

RISC-V 指令系统

- RV32I - 基础整数指令
- 特权指令
- RV32F – 单精度浮点指令
- RV32D – 双精度浮点指令
- 扩展指令

- RV32
  - 面向32位处理器的指令系统
- RV64
  - 64位字长的RISC-V指令集

ThinPAD RISC-V指令系统

- 采用与RV32I兼容的指令格式
  - 32位固定字长
  - 操作码位置及长度固定
  - 寻址方式简单
- 共设计有19+6+1条指令
  - 19条是基础指令，可以用于运行基本的监控程序（不包括异常与中断，不包括虚拟地址）
  - 6条是支持优先级，支持中断与异常的指令
  - 1条是支持虚拟地址的指令（没有TLB的话可实现为Nop）
  - 可根据需要进行扩展
- 作为本课程教学实验的指令系统

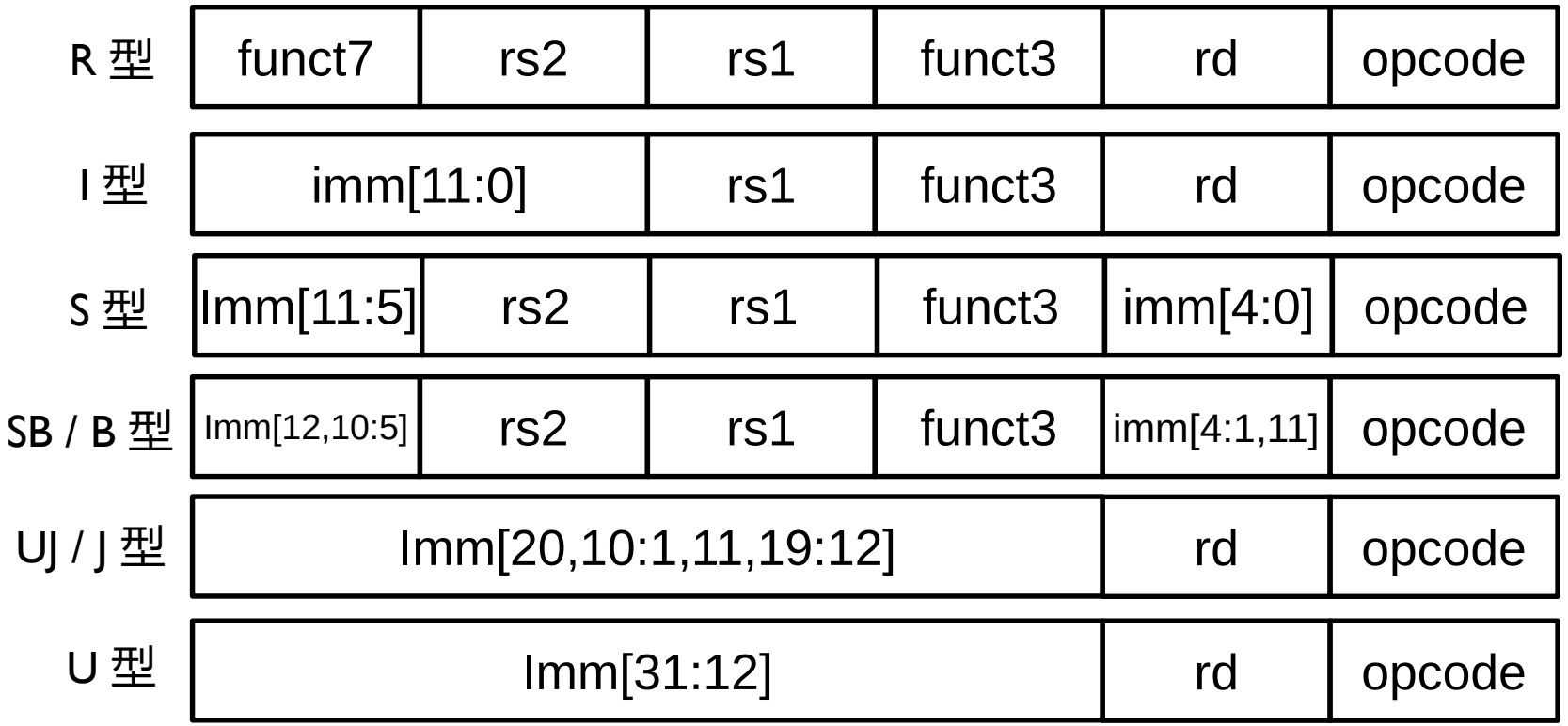
ThinPAD RV32指令系统概况

所需要实现的基础指令19条

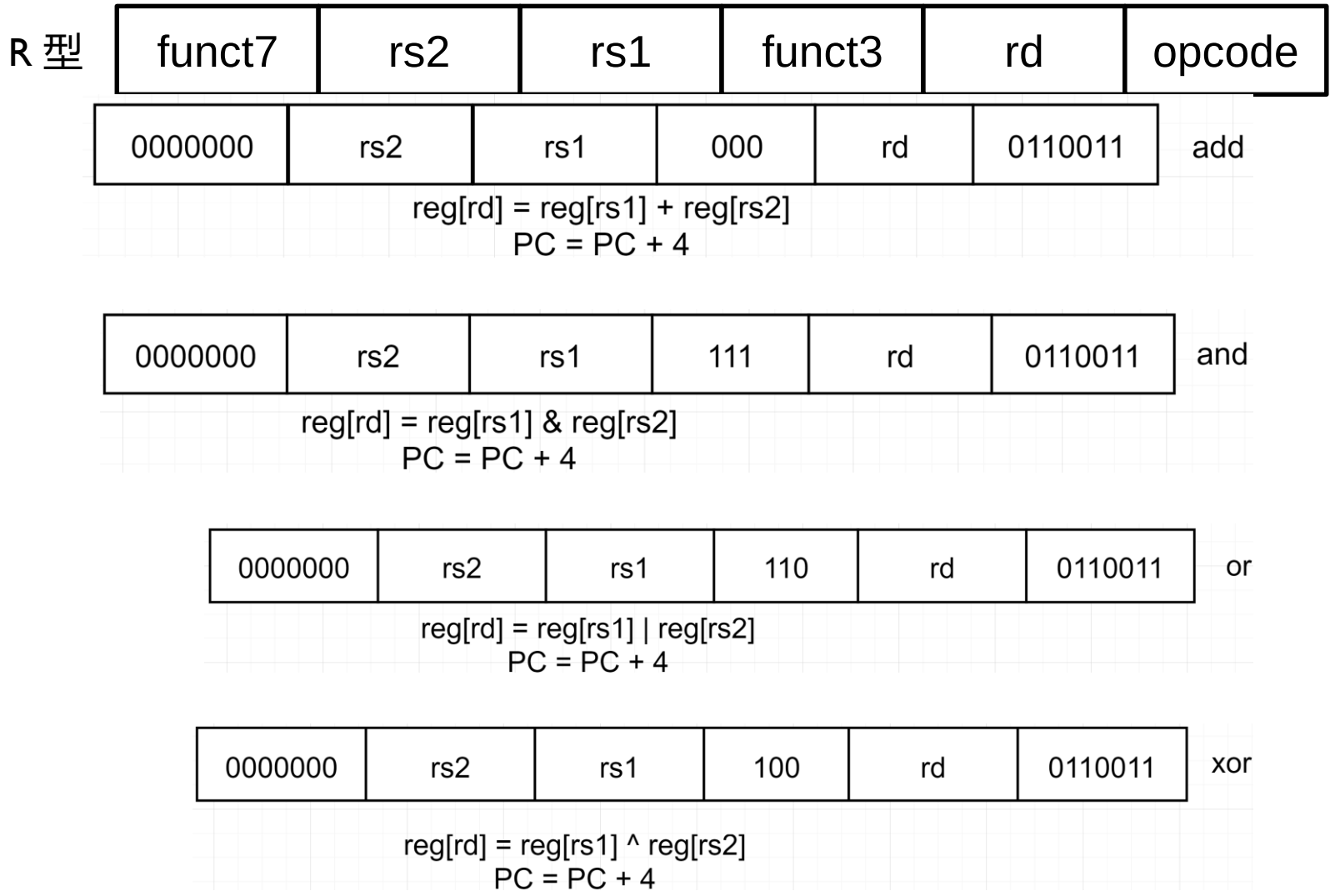
监控程序基础版本	ADD, ADDI, AND, ANDI, AUIPC, BEQ, BNE, JAL, JALR, LB, LUI, LW, OR, ORI, SB, SLLI, SRLI, SW, XOR
监控程序支持中断版本	基础版本所有指令+CSRRC, CSRRS, CSRRW, EBREAK, ECALL, MRET
监控程序支持TLB版本	上一行所有指令+SFENCE.VMA

RISC-V 指令格式（32位）

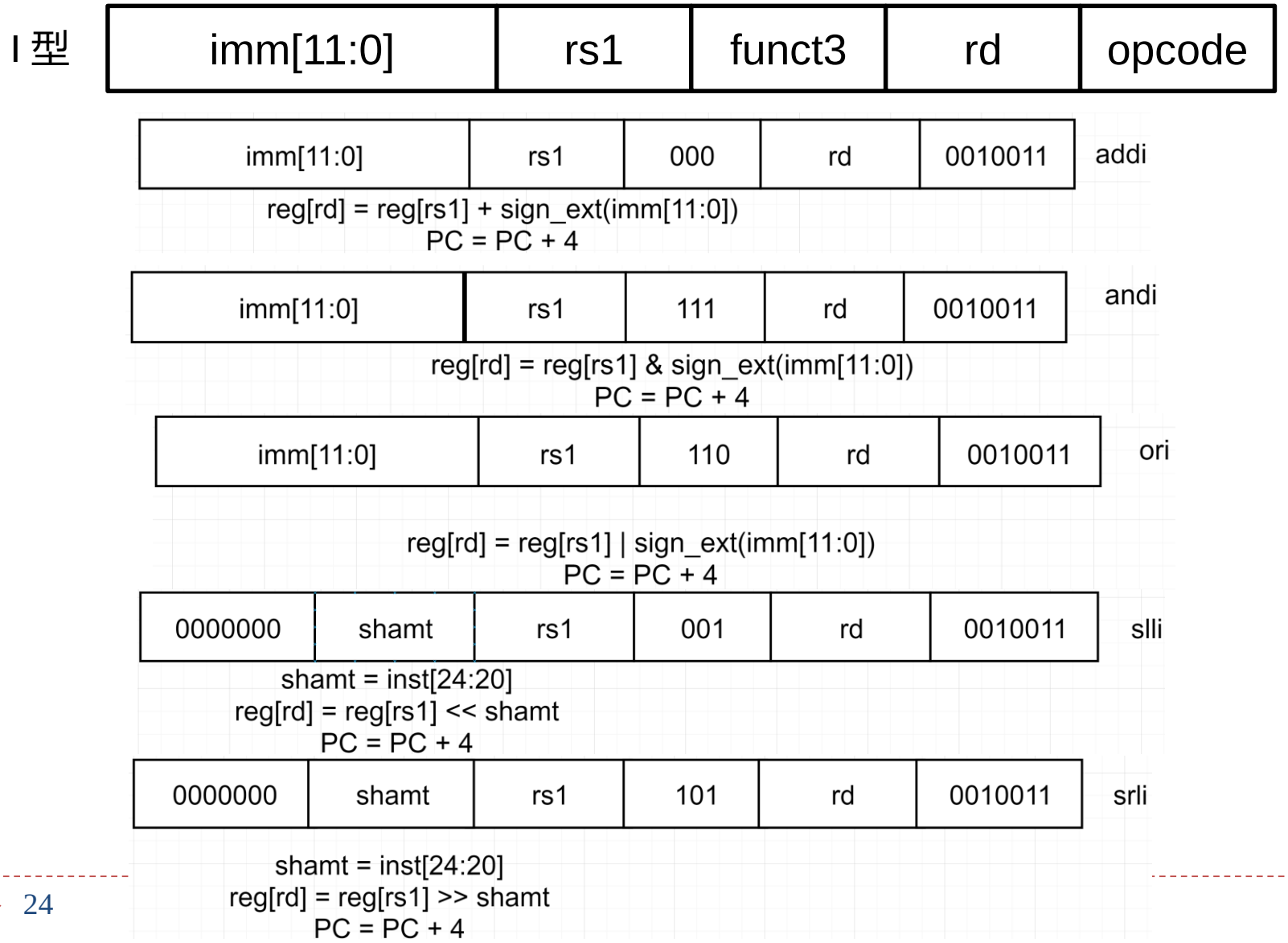
- 所有的指令都是32位字长，有 6 种指令格式：寄存器型，立即数型，存储型，分支指令、跳转指令和大立即数



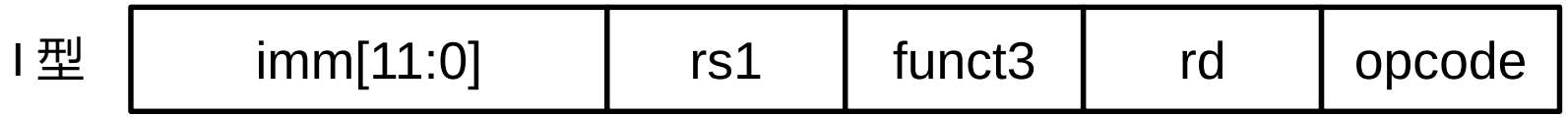
ThinPAD RV32指令（R型）

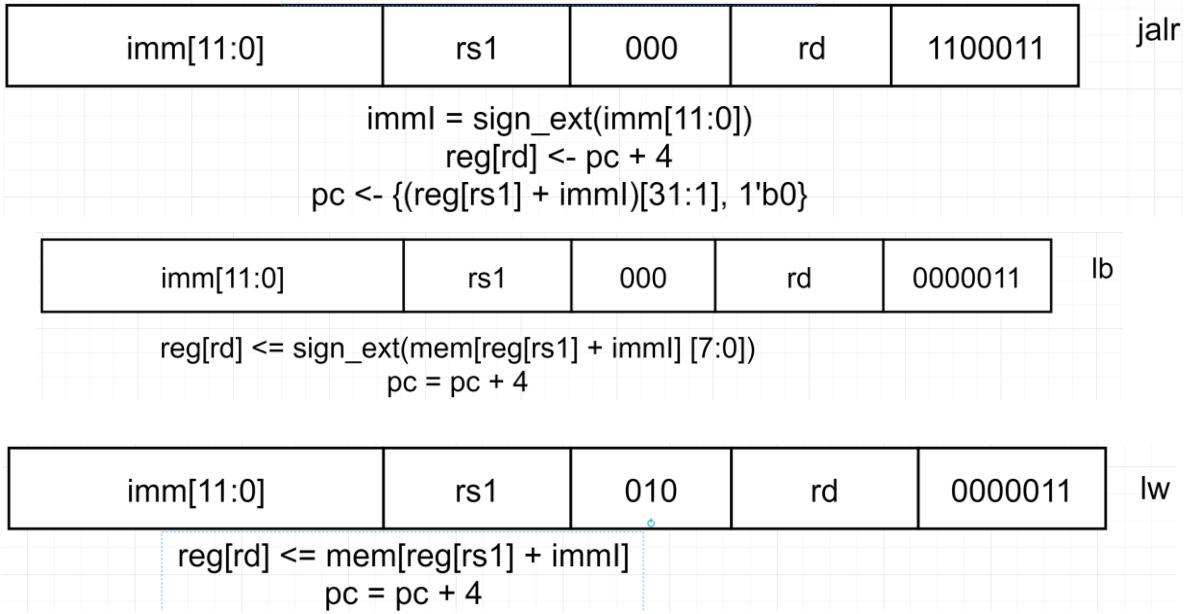


ThinPAD RV32指令（I型-I）



ThinPAD RV32指令（I型-2）





ThinPAD RV32指令（S型）

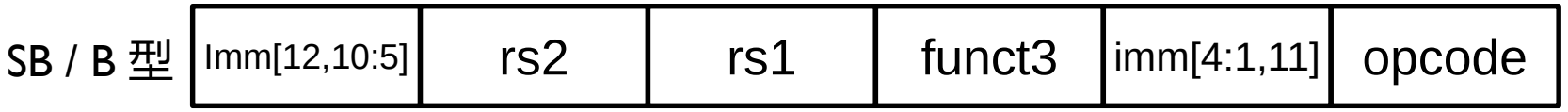
imm[11:5]	rs2	rs1	000	imm[4:0]	0100011	sb

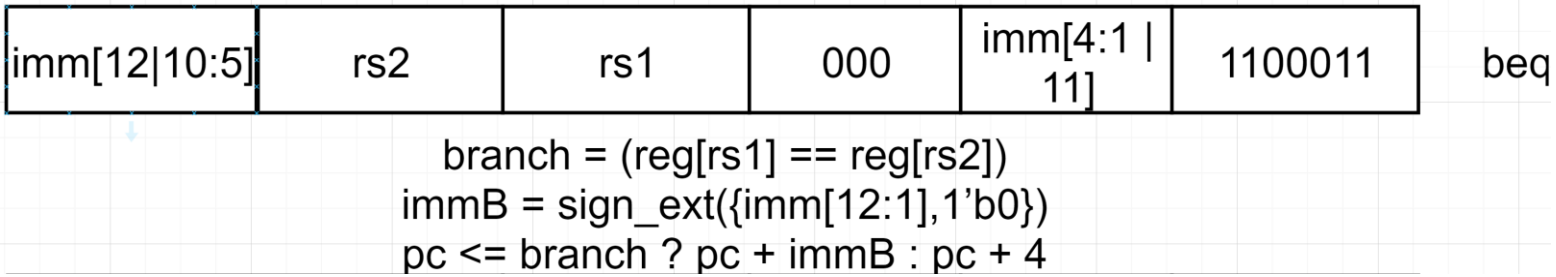
mem[reg[rs1] + sign_ext(imm[11:0]) <= reg[rs2] [7:0] byte store
pc = pc + 4

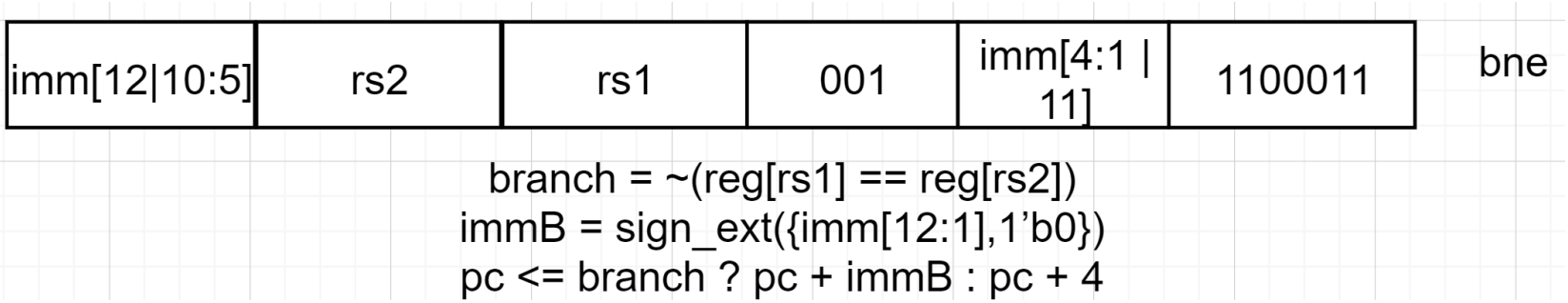
imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	sw

mem[reg[rs1] + sign_ext(imm[11:0])] <= reg[rs2]
pc = pc + 4

ThinPAD RV32指令（SB/B型）

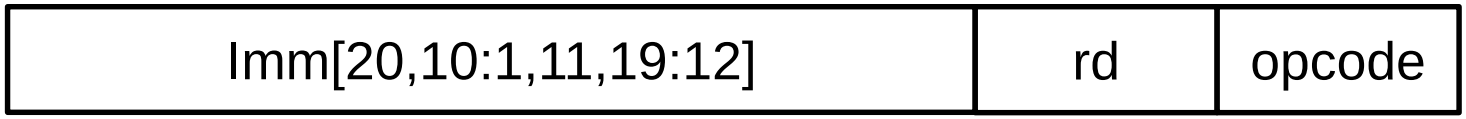


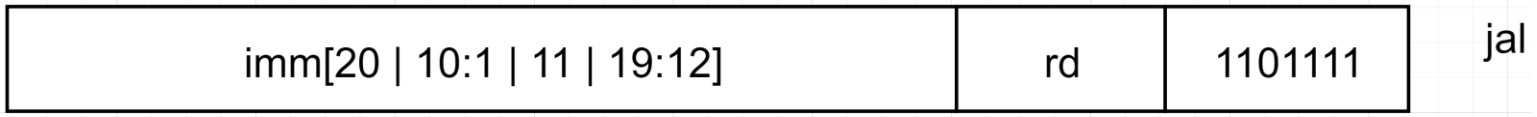




ThinPAD RV32指令（UJ/J型）

UJ / J 型



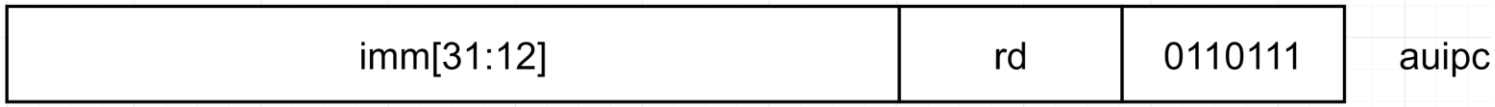


jal

```
immJ = sign_ext(imm[20:1], 1'b0)
reg[rd] <- pc + 4
pc <- pc+immJ
```

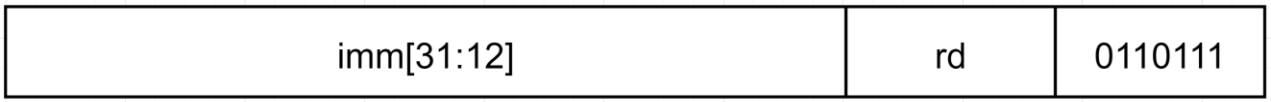
ThinPAD RV32指令（U型）

U 型



auipc

immU = inst[31:12]
reg[rd] <= {immU, 000000000000} + pc
pc = pc + 4

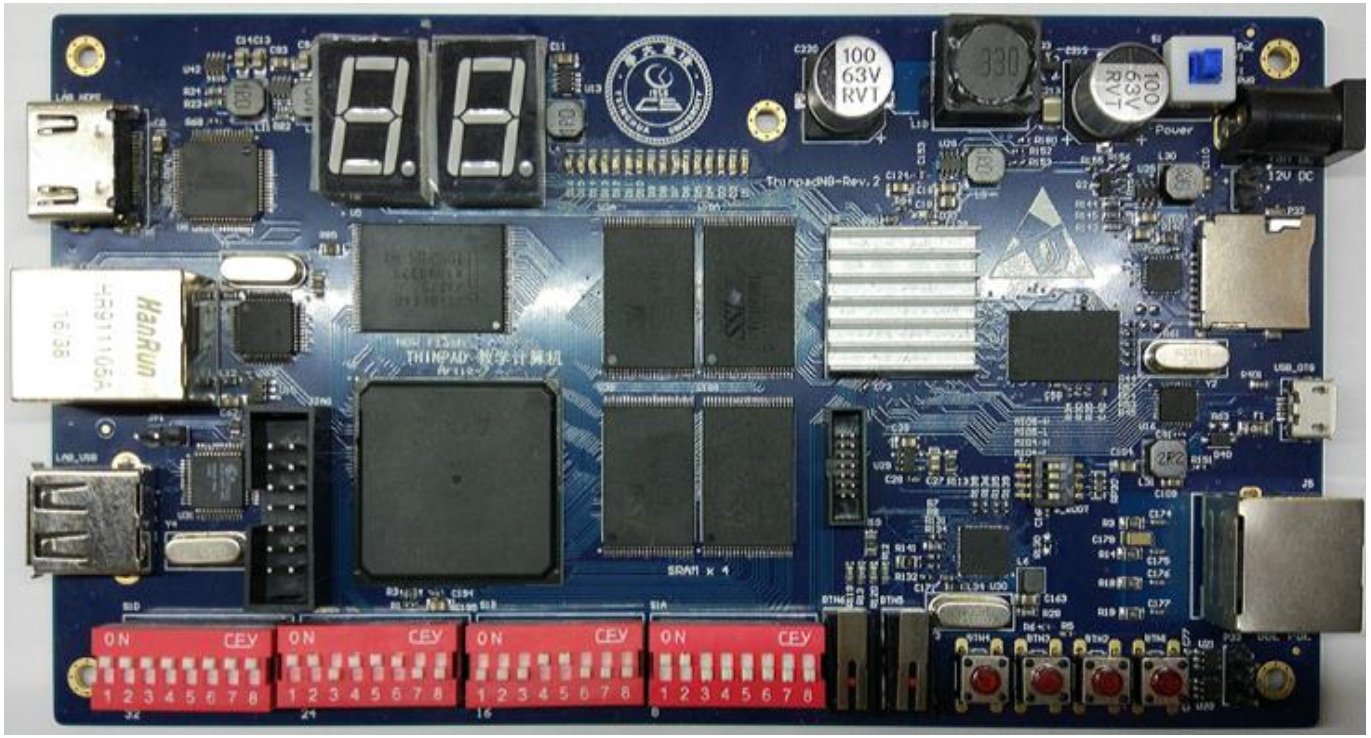


lui

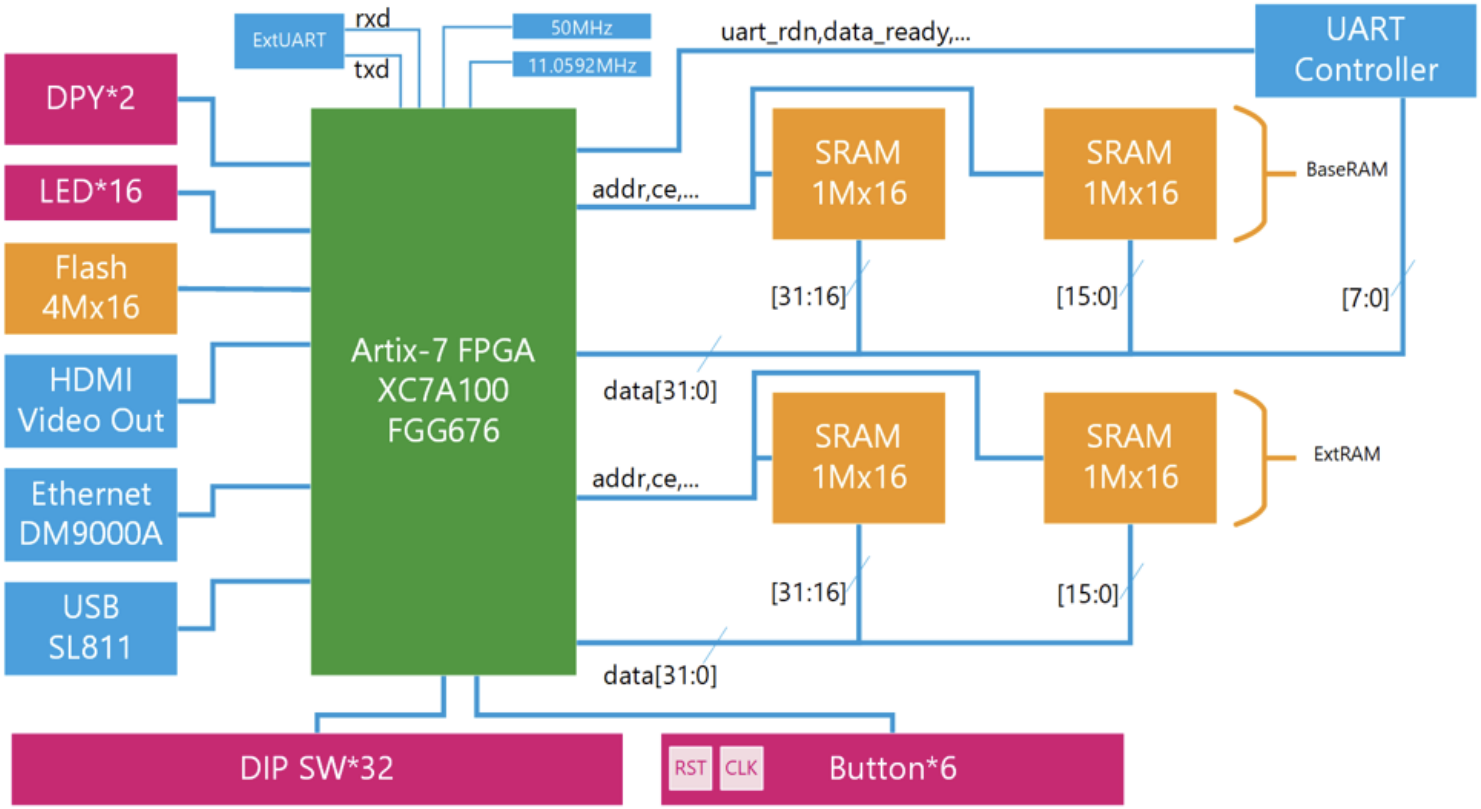
immU = inst[31:12]
reg[rd] <= {immU, 000000000000}
pc = pc + 4

ThinPAD的硬件组成

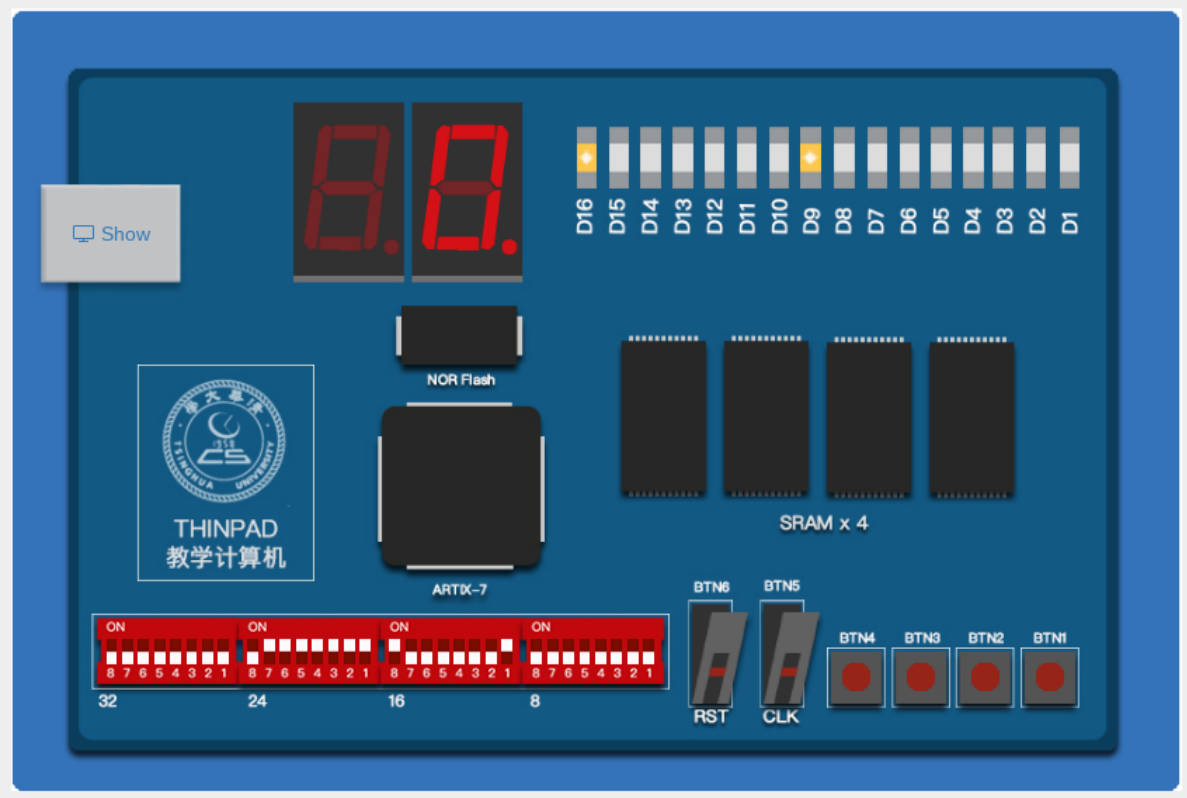
- 主要由一块FPGA，四块SRAM，串口以及其它的一些外围接口电路组成



硬件平台的内部电路构成



网络控制界面


Show
D16 D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1
NOR Flash
SRAM x 4
THINPAD
教学计算机
ARTIX-7
ON
8 7 6 5 4 3 2 1
32 24 16 8
BTN6 BTN5 BTN4 BTN3 BTN2 BTN1
RST CLK

监控程序的地址空间划分

虚地址区间	说明
0x80000000-0x800FFFFF	Kernel代码空间
0x80100000-0x803FFFFF	用户代码空间
0x80400000-0x807EFFFF	用户数据空间
0x807F0000-0x807FFFFF	Kernel数据空间
0x10000000-0x10000008	串口数据及状态

串口寄存器位定义

地址	位	说明
0x10000000（数据寄存器）	[7:0]	串口数据，读、写地址分别表示串口接收、发送一个字节
0x10000005（状态寄存器）	[5]	状态位，只读，为1时表示串口空闲，可发送数据
0x10000005（状态寄存器）	[0]	状态位，只读，为1时表示串口收到数据

模拟器

- 模拟器可以模拟ThinPAD硬件的执行，通过QEMU来模拟RISCV处理器
- 监控程序可以直接运行在模拟器上
- 模拟器可以运行在Windows，Linux和Mac三个平台
- Linux是模拟器天然支持的环境
- Windows需要小调整
- 不建议在Mac下面运行，因为后续的实验环境Vivado只能在Linux和Windows下面执行
  - 在Mac下面建议安装虚拟机，使用VirtualBox就行

实验提示

- 监控程序在kern下，命令行程序在term下
- 务必仔细阅读监控程序下的README.md，这个文件非常重要，务必仔细阅读，包含了监控程序的指示，以及需要实现的指令和格式
- 按照实验指导材料运行监控程序，编写汇编代码，熟悉监控程序

请仔细阅读今年的实验手册

https://lab.cs.tsinghua.edu.cn/cod-lab-docs-2022/

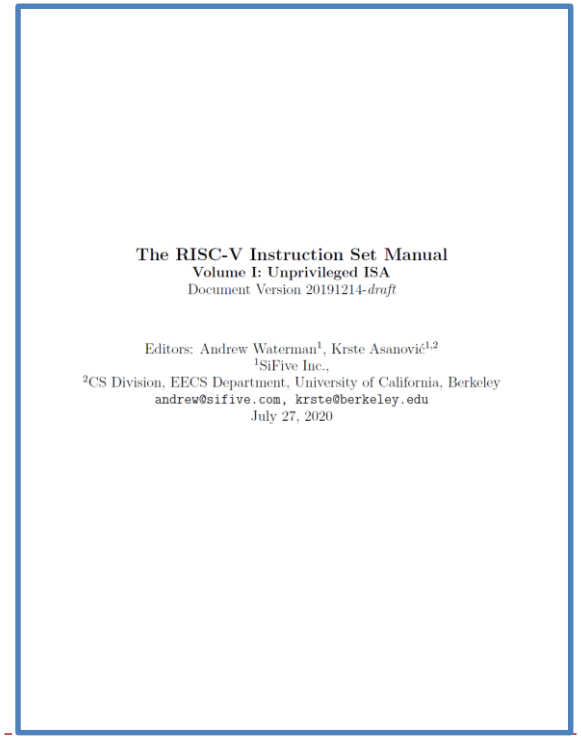
- 实验手册详细给出了
  - 实验环境
  - 模拟器安装和使用
  - 指令概况
  - Vivado使用入门
  - 云平台使用指南
  - 顶层项目简介
  - 各个实验的要求和实验步骤，提示
  - 其它一些相关内容

实验环境说明

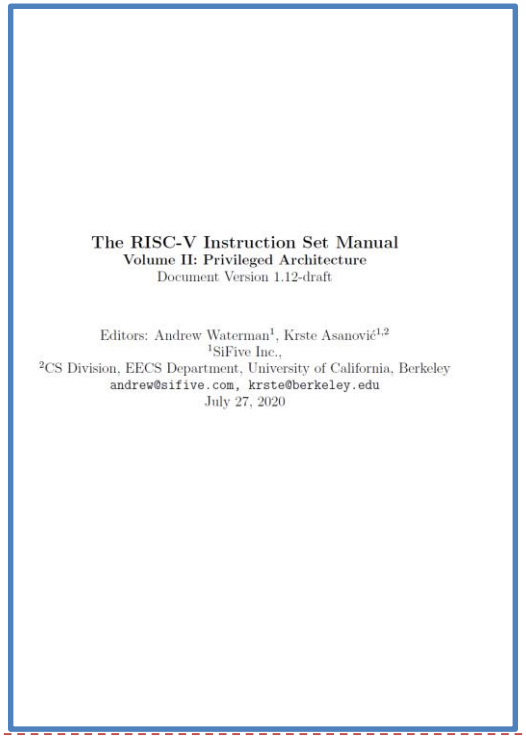
- 所有的实验都可以云上进行，而不需要直接使用实验板子，本年度的实验不发实验板子
- 网络实验能够支持运行监控程序，ucore等，可以完成基本实验
- 后期做扩展实验的小组，有需要的提出申请，每一个小组可以借出一套实验装置，期末之前归还

实验参考书

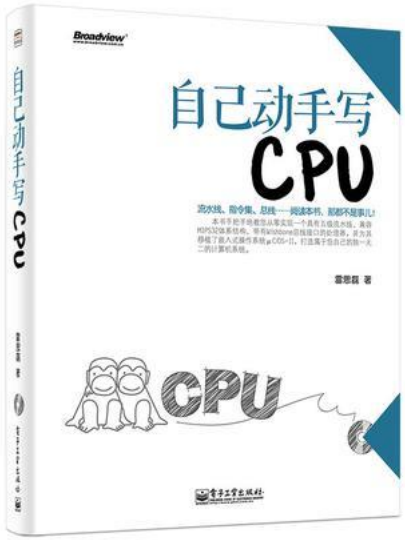
- 《自己动手写CPU》 详细给出了如何使用verilog写一个MIPS CPU的例子，可以作为实验的基础
- RISCV指令手册，用户层编程手册，系统编程手册


The RISC-V Instruction Set Manual
Volume I: Unprivileged ISA
Document Version 20191214-*draft*

Editors: Andrew Waterman[1], Krste Asanović[1,2]
[1]SiFive Inc.,
[2]CS Division, EECS Department, University of California, Berkeley
andrew@sifive.com, krste@berkeley.edu
July 27, 2020


The RISC-V Instruction Set Manual
Volume II: Privileged Architecture
Document Version 1.12-draft

Editors: Andrew Waterman[1], Krste Asanović[1,2]
[1]SiFive Inc.,
[2]CS Division, EECS Department, University of California, Berkeley
andrew@sifive.com, krste@berkeley.edu
July 27, 2020



小结

- 计算机程序语言
  - 高级语言
  - 汇编语言
  - 机器语言
- 指令系统
  - 硬件/软件接口
  - 指令功能/指令格式
- ThinPAD RISC-V指令系统
- 实验环境简介

阅读和思考

- 阅读
  - 教材：第2章 指令系统
- 思考
  - 指令系统的作用和地位?
  - 为实现ThinPAD指令系统，ALU应该具备哪些功能?
  - 数据在计算机内部如何表示？应表示哪几类数据?
- 练习
  - 分析ThinPAD RISCV指令系统的特点
  - 完成实验一

谢谢