8-bit CPU的Verilog设计

# 一、概况

本8-bit CPU采用流水线设计，对指令同步进行“取指”“译码”“执行”“写回”四步操作。在指令集的设计上，功能简单，易于电路实现，并通过多重操作完成复杂的功能。具有一定的分支预测能力。

内部有两个数据寄存器，分别为寄存器0和寄存器1。其中，寄存器1又称为地址寄存器，它的数据不但可以直接被指令赋值，并且可以帮助内存寻址。

不过，由于该指令集的指令允许算术逻辑单元

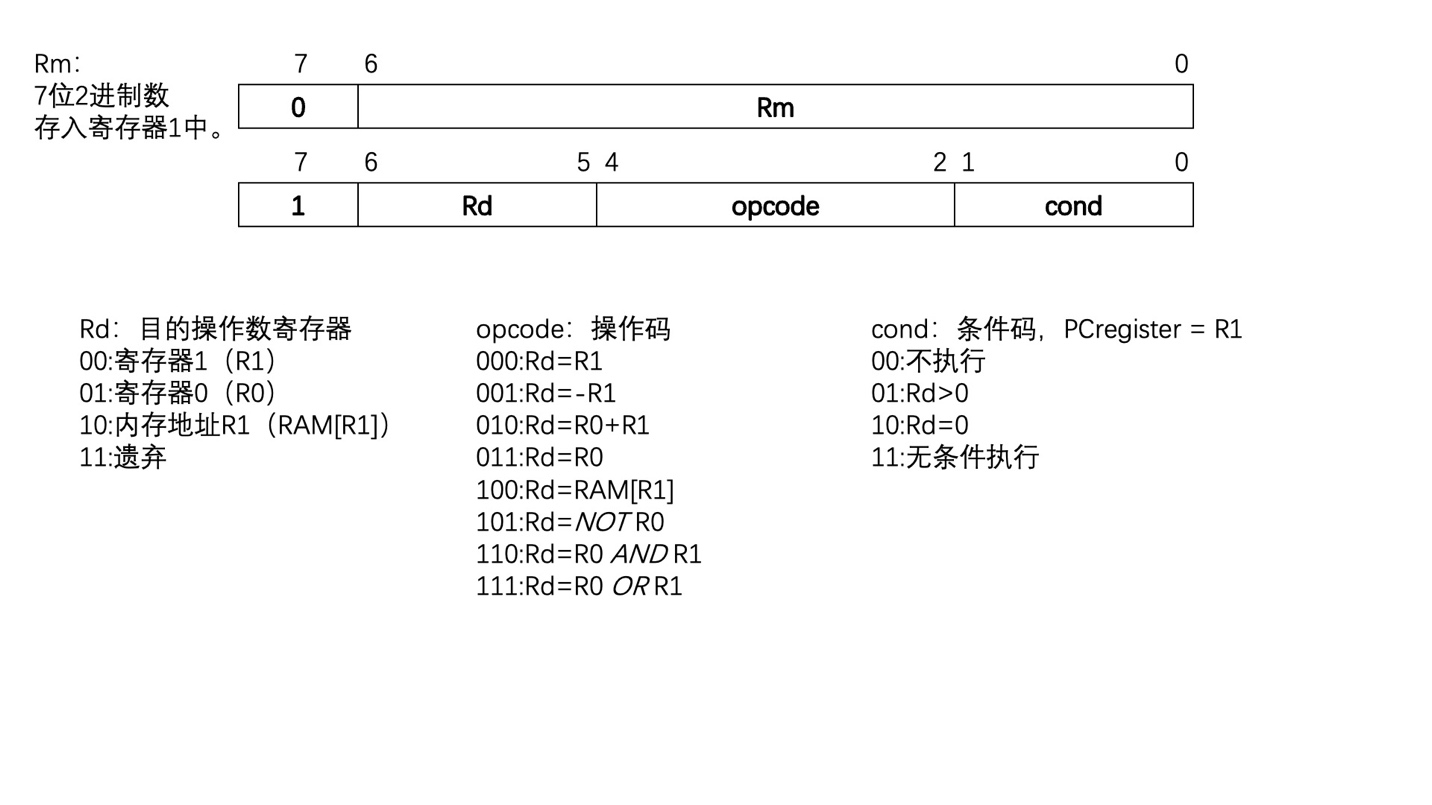
# 二、指令集结构

指令集由8位构成，分为“数据指令”和“控制指令”。

数据指令：当第7位（最左侧的一位）为0时，将整个指令作为数据存入寄存器1中。

控制指令：当第7位为1时，将指令进行正常的译码以控制整个CPU。

指令集的具体规则如下图所示。

  
图1

# 三、CPU结构

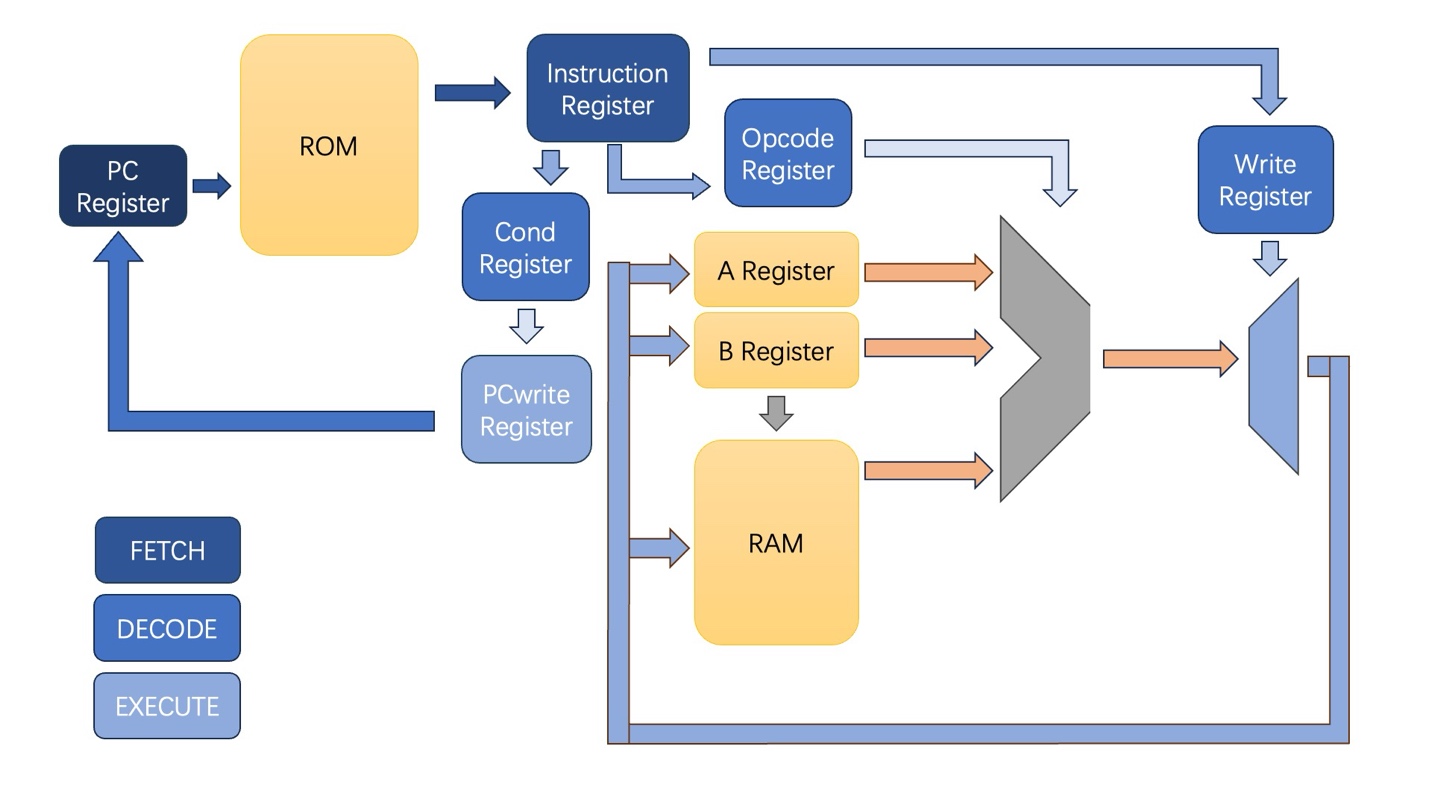
PC寄存器（Program Counter，程序计数器）是CPU中最基本的控制单元，在默认情况下，它是一个计数器，在每个时钟周期数值加1。

ALU单元（Arithmetic Logic Unit，算术逻辑单元）是CPU的核心单元，是一个非时序的电路，它接受3位操作码，将寄存器0、寄存器1或者内存的数据进行特定的计算并输出。

A Register和B Register 分别为寄存器0和寄存器1。

只读存储器ROM是存放二进制指令的地方，随机存取存储器RAM是保存数据的地方。

CPU的结构框图如下图所示。

  
图2

虽然直接存储数据的寄存器只有两个，但是实际的CPU设计中还包含了众多寄存器暂存指令。

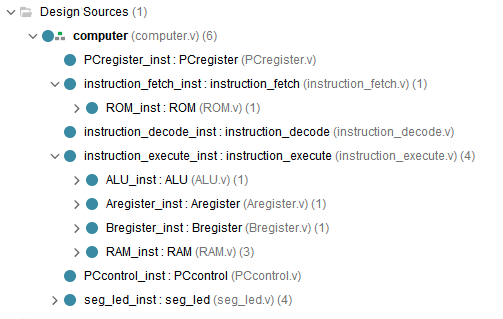
PC寄存器控制着CPU的“取指”操作，从对应的ROM地址中取出二进制指令，取出的指令将会暂存在Instruction Register中。

之后，指令会被“译码”操作。如果是控制指令，那么条件码会进入Cond Register中，操作码会进入Opcode Register中，ALU输出的目的地会写入Write Register中。

最后，指令进行“执行”操作。Write Register相当于控制了一个数据分配器，决定了ALU输出结果的最后走向。

此外，还有一个PCwrite 寄存器，它会基于条件码和ALU输出结果判断是否要进行跳转。如果需要跳转，在“执行”操作的下一个时钟周期，PC寄存器的数据将会被更新为寄存器1的数据。但是在流水线中，跳转指令从取指到执行需要经过3个周期，将会导致不需要的运行的指令也进入了流水线作业。所以，PCwrite还需要拦截下个周期的“取指”“译码”和“执行”操作。

# 四、Verilog实现

  
图3

<https://box.nju.edu.cn/f/274d7195d3554b9aac42/>