

第四章 处理器体系结构--速记清单

4-1 节—指令集体系结构

1. 程序寄存器：15 个寄存器（除了 %r15），每个 64 位。其中特别注意，0xF 代表无寄存器。对于仅使用一个寄存器的指令，简单有效的处理方法是用特定编码 0xF 表示操作数不是寄存器。

2. Y86-64 指令集的指令长度：1 到 10 个字节（具体来说，长度分别是 1、2、9、10 个字节的指令长度）。

3. halt、exit 和 break 的区别。

halt：退出程序。

exit：退出过程、函数。如果在主程序，则效果和 halt 一样。

break：跳出循环。

4-2 节—逻辑设计

1. 组合逻辑和时序逻辑的区别。

(1) 组合逻辑—输出仅取决于当前的输入。

(2) 时序逻辑—输出与当前和之前的输入有关（在时钟上升沿到来时才更新输出）。

2. 寄存器文件。

写：只在时钟上升沿更新。（时序逻辑）

读：类似组合逻辑，根据输入地址产生输出数据（但也有延迟）。

4-3 节—顺序执行的处理器

1. 顺序执行分成 6 个阶段。

(1) 取指：

①从指令存储器读取指令

②ValC=ISA 的 V/D/Dest

③ValP=PC+指令长度

(2) 译码：读程序寄存器 rA rB %rsp

(3) 执行：计算数值或地址 valE CC

(4) 访存：读或写数据 valM

(5) 写回：写程序寄存器 valE valM

(6) 更新 PC：更新程序计数器 PC

2. 指令的格式，如图 3-1 所示。

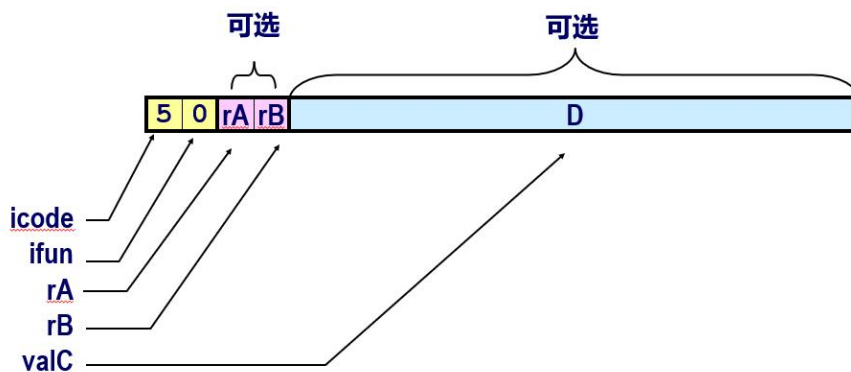


图 3-1 指令通用格式（以 `mrmovq D(rB), rA` 为例）

icode:ifun（第一字节固定不变）

可选的寄存器（共 1 字节） rA:rB

可选的常数（8 字节） valC

3. 计算的数值

(1) 取指

icode	指令码
ifun	功能码
rA	源寄存器 A 的编号
rB	源寄存器 B 的编号
valC	指令中的常数
valP	增加后的 PC

(2) 译码

srcA	源寄存器 A 的编号
srcB	源寄存器 B 的编号
valA	值 A（大部分时候对应寄存器 srcA 中的值）
valB	值 B（大部分时候对应寄存器 srcB 中的值）
dstE	valE 的写寄存器编号
dstM	valM 的写寄存器编号

(3) 执行

valE	ALU 运算结果，将被写入到 dstE 寄存器中
Cnd	分支或转移标识

(4) 访存

valM	内存中的数值，将被写入到 dstM 寄存器中
------	------------------------

(5) 写回

写回阶段更新寄存器，无计算。

(6) 更新 PC

PC

4. 通用的 Y86-64 的微指令。所有指令遵循同样的一般格式，区别在于每一步计算的不同。具体每条指令的计算过程，见附表“[Y86-64 微指令卡片.pdf](#)”。

4-4 节 流水线的实现基础

1. 流水线分为 5 个阶段。(注意把顺序执行的修改 PC 阶段, 添加到了取指阶段里)

(1) 取指:

- ①选择当前 PC
- ②读取指令
- ③计算 PC 的值

(2) 译码: 读取程序寄存器

(3) 执行: 操作 ALU

(4) 访存: 读或写存储器

(5) 写回: 更新寄存器文件

2. 参照 Y86-64 流水线 CPU 的实现, 说明流水线如何工作。

答: 流水线化的系统, 待执行的任务被划分成若干个独立的阶段, 将处理器的硬件也组织成若干个单元, 让各个独立的任务阶段在不同的硬件单元上一次执行, 从而使多个任务并行操作。

结合案例: 如 Y86-64 将指令执行分为取指、译码、执行、访存、写回 5 个阶段, 通过在每个阶段插入流水线寄存器, 利用时钟信号控制流水线的时序和操作, 理想情况下可实现 5 条指令的同时运行。

4-5 节 流水线实现高级技术

1. Y86-64 流水线 CPU 中的冒险的种类与处理方法。

答: (1) 数据冒险: 指令使用寄存器 R 为目的, 瞬时之后使用 R 寄存器为源。

处理方法有:

①暂停: 通过在执行阶段插入气泡 (bubble/nop), 使得当前指令执行暂停在译码阶段;

②数据转发: 增加 valM/valE 的旁路路径, 直接送到译码阶段;

(2) 加载使用冒险: 指令暂停在取指和译码阶段, 在执行阶段插入气泡 (bubble/nop)

(3) 控制冒险: 分支预测错误: 在条件为真的地址 target 处的两条指令分别插入 1 个 bubble。ret: 在 ret 后插入 3 个 bubble。

4-6 节 处理器的性能

CPI: Cycle Per Instruction。

1. CPI 的计算:

C: 时钟周期

I: 执行完成的指令数

B: 插入的气泡个数 ($C = I + B$)

$$CPI = C/I = (I+B)/I = 1.0 + B/I$$

其中针对我们前面设计的流水线, 平均插入的气泡个数计算如图 6-1 所示:

$B/I = LP + MP + RP$	
<ul style="list-style-type: none"> ■ LP: 由加载/使用冒险停顿产生的处罚 <ul style="list-style-type: none"> ▪ 加载指令的比例 ▪ 加载指令需要停顿的比例 ▪ 每次插入气泡的数量 	Typical Values
$\Rightarrow LP = 0.25 * 0.20 * 1 = 0.05$	0.25 0.20 1
<ul style="list-style-type: none"> ■ MP: 由错误的分支预测产生的处罚 <ul style="list-style-type: none"> ▪ 条件转移指令的比例 ▪ 条件转移预测错误的比例 ▪ 每次插入气泡的数量 	
$\Rightarrow MP = 0.20 * 0.40 * 2 = 0.16$	0.20 0.40 2
<ul style="list-style-type: none"> ■ RP: 由ret指令产生的处罚 <ul style="list-style-type: none"> ▪ 返回指令站的比例 ▪ 每次插入的气泡数量 	
$\Rightarrow RP = 0.02 * 3 = 0.06$	0.02 3

图 6-1 平均插入气泡个数的计算

因此我们设计的流水线处罚造成的影响（三种处罚的总和）为：

$$0.05 + 0.16 + 0.06 = 0.27$$

$$\rightarrow CPI = 1.27$$

2. 补充：超标量：CPI < 1。