图灵机停机问题是不可计算的。

冯诺依曼机组成部分：

运算器、控制器、存储器、输入和输出设备。

冯诺依曼机特征：

指令和数据“以同等地位”存放于存储器内，分别按地址访问；

指令和数据均用二进制码表示；

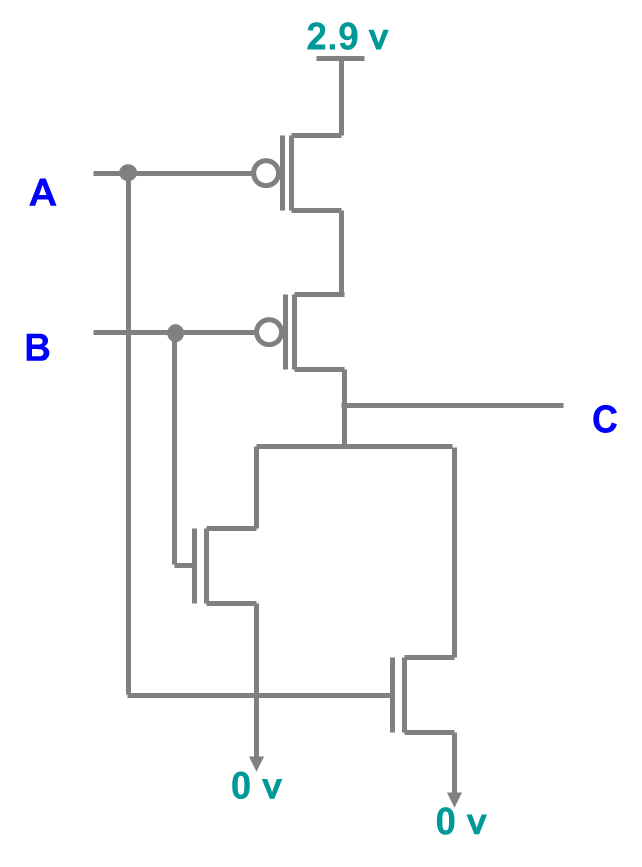
指令由操作码和地址码构成；

指令按顺序存放，顺序执行。

冯诺依曼机以运算器为核心，现代计算机以存储器为核心。

指令的执行过程：取指、译码、取操作数、执行、写回。

NOR：或非门，只要输入有1，输出就为0。



RS锁存器：

S=0，R=1，锁存器清0；S=1，R=0，锁存器置1；S=1，R=1禁止。

[X]补求[-X]补：每一位取反后再在最低位+1。

8位补码最小值：1000 0000。

正数的原码，反码，补码表示均相同。

补码加法：[A]补+[B]补=[A+B]补

补码减法：[A-B]补=[A]补+[-B]补

双符号含义：

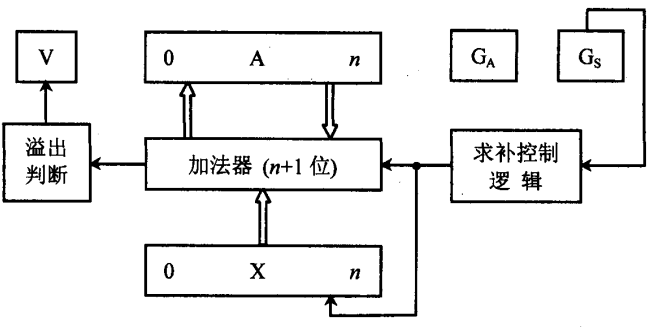
00表示运算结果为正数；

01表示运算结果正溢出；

10表示运算结果负溢出；

11表示运算结果为负数。

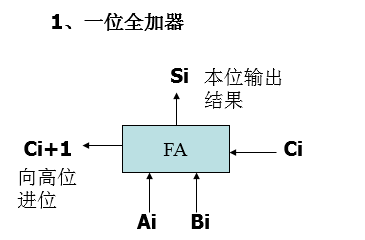
图要会画：





XOR异或门：两个输入相同时为0，不同时为1。

一位全加器：



**Si＝Ai⊕Bi⊕Ci**

**Ci＋1＝AiBi＋BiCi＋CiAi**

**上面两个算式中，⊕是异或，AiBi是与，+是或。**

补码一位乘：BOOTH算法

Booth算法中，操作的方式取决于表达式(ynyn+1)的值。

:yn+1yn：

10：部分积加**[-x]补**

01：部分积加**[x]补**

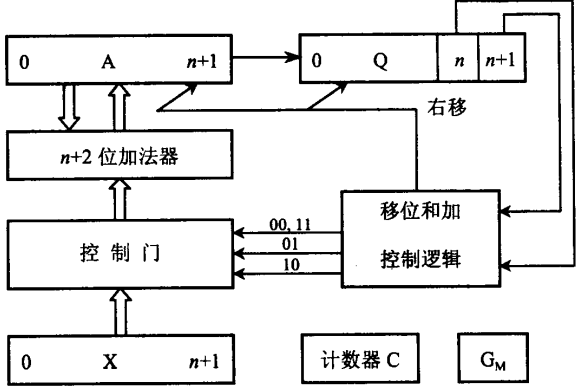
00或11：部分积右移一位

第一步不移位。

最后一步算完不再移位。

例子：[x]补=0.1101，[y]补=0.1011，[x·y]补=0.10001111

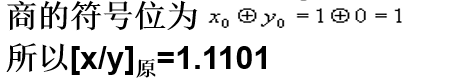
|  |  |  |  |
| --- | --- | --- | --- |
| **部分积** | **乘数yn** | **附加位yn+1** | **说明** |
| **00.0000**  **+ 11.0011** | **01011** | **0** | **初值[z0]补=0**  **Ynyn+1=10，部分积加[-x]补** |
| **11.0011**  **11.1001**  **11.1100**  **+ 00.1101** | **10101**  **11010** | **1**  **1** | **→1位，得[z1]补**  **Ynyn+1=11，部分积→1位得[z2]补**  **Ynyn+1=01，部分积加[x]补** |
| **00.1001**  **00.0100**  **+ 11.0011** | **11**  **11101** | **0** | **→1位，得[z3]补**  **Ynyn+1=10，部分积加[-x]补** |
| **11.0111**  **11.1011**  **+ 00.1101** | **11110** | **1** | **→1位，得[z4]补**  **Ynyn+1=01，部分积加[x]补** |
| **00.1000** | **1111** |  | **最后一步不移位，得[x·y]补** |



原码除法：加减交替法

已知x=-0.1011,y=0.1101,求：[x/y]原

|  |  |  |
| --- | --- | --- |
| 被除数(余数) | 商 | 说     明 |
| 0.1011 + 1. 0011 | 0.0000 | +[-y\*]补（减除数） |
| 1.1110  1.1100 + 0.1101 | 0      0 | 余数为负，上商0 ← 1位 +[y\*]补 (加除数) |
| 0.1001  1.0010 + 1.0011 | 0 1  0 1 | 余数为正，上商1 ← 1位 +[-y\*]补（减除数） |
| 0.0101  0.1010 + 1.0011 | 0 1 1  0 1 1 | 余数为正，上商1 ← 1位 +[-y\*]补（减除数） |
| 1.1101  1.1010 + 0.1101 | 0 1 1 0   0 1 1 0 | 余数为负，上商0 ← 1位 +[y\*]补 (加除数) |
| 0.0111 | 0 1 1 0 1 | 余数为正，上商1 |



**补码除法：加减交替法：**

**x=-0.1001, y=+0.1101 求： [x/y]补。[x／y]补=1.0101**

|  |  |  |
| --- | --- | --- |
| **被除数（余数）** | **商  上商** | **说    明** |
| **1.0111  +  0.1101** | **0.0000** | **[x]补与[y]补异号，+[y]补** |
| **0.0100    0.1000  +  1.0011** | **1      1** | **[R]补与[y]补同号，上商1 ← 1位 +[-y]补** |
| **1.1011    1.0110  +  0.1101** | **1 0 1 0** | **[R]补与[y]补异号，上商0 ← 1位 +[y]补** |
| **0.0011    0.0110  +  1.0011** | **1 0 1 1 0 1** | **[R]补与[y]补同号，上商1 ← 1位 +[-y]补** |
| **1.1001    1.0010** | **1 0 1 0**  **1 0 1 0 1** | **[R]补与[y]补异号，上商0 ← 1位,末位商恒置“1”** |

**商：1.0101**

**CISC和RISC各自的特点（简答题）：**

指令系统：

RISC指令少而且简单，对不常用的功能，常通过组合指令来完成。RISC上可用流水技术和超标量技术。

CISC 计算机的指令丰富，处理特殊任务效率较高。

存储器操作：

RISC 对存储器操作有限制，控制简单。

CISC 的存储器操作指令多，操作直接。

程序：

RISC 汇编语言程序一般需要较大的内存空间，实现特殊功能时程序复杂，不易设计。

CISC 汇编语言程序编程相对简单，科学计算及复杂操作的程序设计相对容易，效率较高。

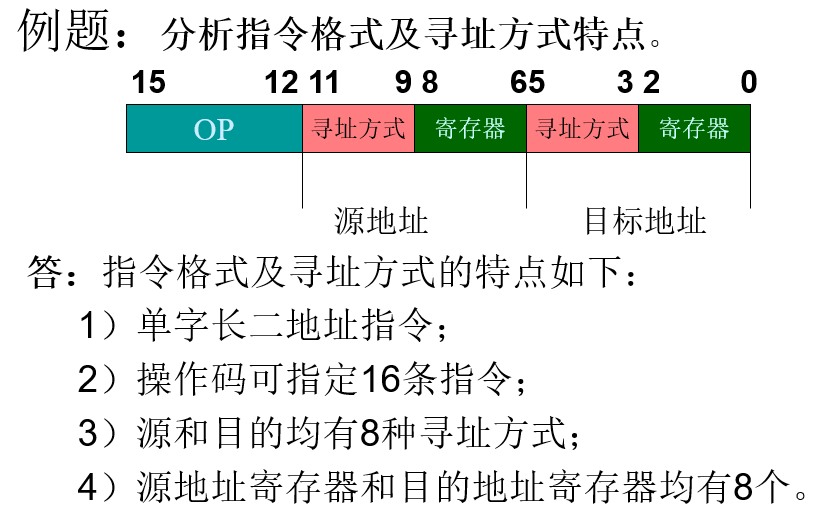
CPU：

RISC CPU 包含有较少的单元电路，因而面积小、功耗低；而CISC CPU 包含有丰富的电路单元，因而功能强、面积大、功耗大。

设计周期：

RISC 微处理器结构简单，设计周期短。

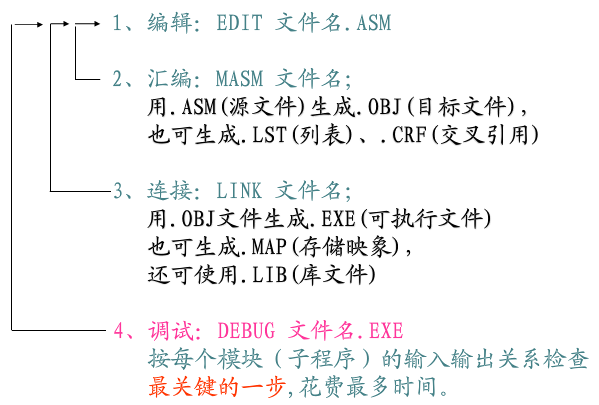
CISC 微处理器结构复杂，设计周期长。



直接寻址、间接寻址。

程序分段结构：

数据段、扩展段、堆栈段、代码段。



一个指令周期包含多个机器周期，一个机器周期又包括多个时钟周期（又叫节拍）。

ADD @X的控制－取指：

PCo、MARi、MDRo、IRi

ADD @X的控制－间址：

MDRo、 MARi 、 MDRo、 IRi

ADD @X的控制－执行：

取数：MDRo、MARi 、MDRo、Yi

计算：ACCo、ALUi

写回：Zo、ACCi

三级存储系统：

Cache（缓存）、主存、辅存

“Cache－主存”层次：弥补主存速度的不足

“主存－辅存” 层次：弥补主存容量的不足

要会画下面的图：

