2021 计算机组成原理考试要求和范围

# 第一、试卷满分及考试时间

本试卷满分为 100 分，考试时间为 120 分钟。

# 第二、答题方式

答题方式为闭卷、笔试。

# 第三、试卷题型结构

选择题共 10 分（共 10 小题，每题 1 分，共计 10 分）

简答题共 40 分（共 8 小题，每题 5 分，共计 40 分）

综合题共 50 分（共 5 小题，每题 10 分，共计 50 分）

# 第四 考查内容

【基本要求】

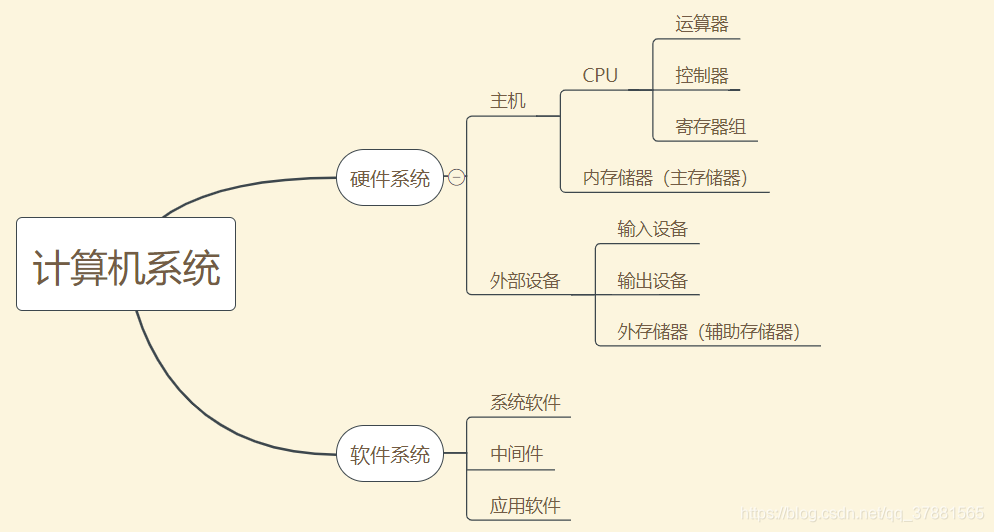
1. **理解单处理器计算机系统中各部件的内部工作原理、组成结构以及相互连接方式，具有完整的计算机系统的整机概念。**
2. **理解计算机系统层次化结构概念，熟悉硬件与软件之间的界面，掌握指令集体系结构的基本知识和基本实现方法。**
3. **能够综合运用计算机组成的基本原理和基本方法，对有关计算机硬件系统中的理论和实际问题进行计算、分析，对一些基本部件进行简单设计。**

【具体要求】

一、计算机系统概述

(一)计算机系统层次结构

1.计算机系统的基本组成



1. **计算机硬件的基本组成**

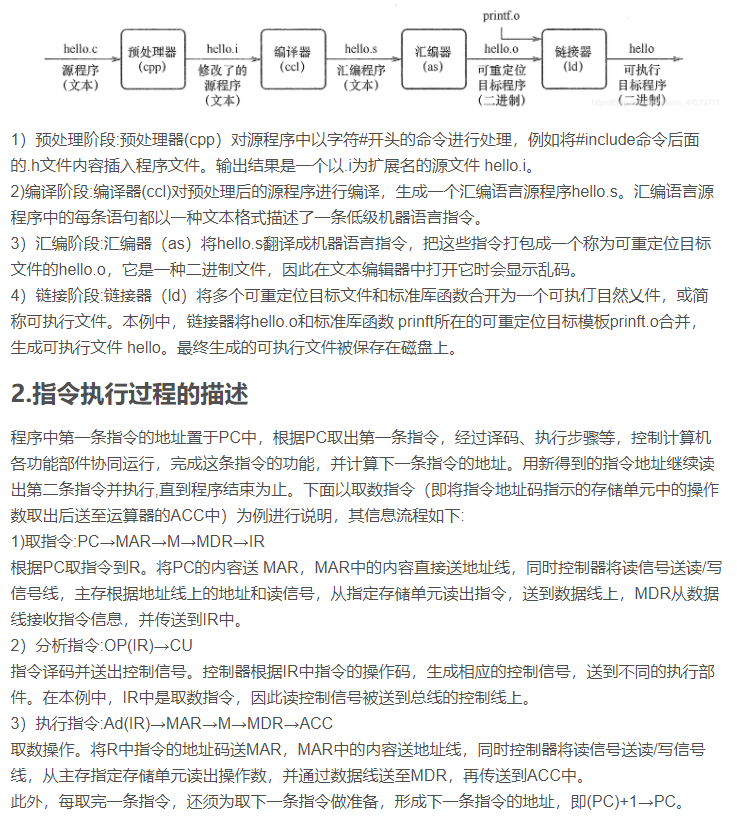
**基本的计算机硬件系统主要由运算器、控制器、存储器、输入设备和输出设备五个部分组成。**

1. **计算机软件和硬件的关系**

**计算机软件和硬件是组成完整计算机系统的两个部分。**

1. **软件和硬件相互依存 （2） 软件硬件没有明显的界限 （3） 软件和硬件协同发展**
2. **计算机的工作过程**

**1.把数据和程序都装入主存储器中。2.将源程序转变成可执行文件 3. 可执行文件的首地址开始逐条执行指令。**



**(二)计算机性能指标**

吞吐量、响应时间;CPU 时钟周期、主频;MIPS、MFLOPS

吞吐量：计算机系统在单位时间内处理请求的数量。

响应时间：指计算机系统对请求所做出的的相应时间，响应时间为CPU时间和等待时间的和。

Cpu时钟周期：CPU主频的倒数，是CPU中最小的时间元素。每一个动作至少要一个时钟周期。

主频：CPU内数字脉冲信号震荡的速度。

MIPS：每秒执行多少百万条指令。

MFLOPS：每秒执行多少百万次浮点运算。

二、数据的表示和运算

(一)数制与编码

**1.进位计数制及其相互转换**

**B（Binary）-二进制数**

**O（Octal）或Q-八进制数**

**D（Decimal）-十进制数**

**H（Hexadecimal）-十六进制数**

**https://blog.csdn.net/qq\_44486437/article/details/104137038?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425136116780262512276%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425136116780262512276&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-104137038.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E8%BF%9B%E4%BD%8D%E8%AE%A1%E6%95%B0%E5%88%B6%E5%8F%8A%E5%85%B6%E7%9B%B8%E4%BA%92%E8%BD%AC%E6%8D%A2&spm=1018.2226.3001.4187**

1. **真值和机器数**

真值**是指在数值前面用“+”号表示正数，用“-”号表示负数的带符号二进制数**

机器数**（又称为机器码）是指在数字系统中用“0”表示符号“+”，用“1”表示符号“-”，即把符号“数值化”后的带符号二进制数。常用的机器数有原码、反码和补码、**

**1.原码**

**求取方法：符号位用0表示正，用1表示负；数值位不变，即与真值的数值位相同。**

**例如，若x＝0.1011、y＝–10110，则**

**[x]原码＝0.1011（由于x为正数，所以符号位为0，数值位与真值相同）**

**[y]原码＝110110（由于y为负数，所以符号位为1，数值位与真值相同）**

**特点：求取方便，运算不方便，“0”有“+0”和“-0”两种不同形式。**

**2.反码**

**求取方法：符号位用0表示正，用1表示负。数值位与符号位相关，当为正数时，数值位不变，即与真值的数值位相同；当为负数时，数值位是真值的数值位按位取反（0变为1，1变为0）。**

**例如，若x＝-0.1010、y＝+10101,则**

**[x]反码＝1. 0101（由于x为负数，所以符号位为1，数值位为真值的数值位按位变反）**

**[y]原码＝010101（由于y为正数，所以有符号位为0，数值位与真值相同）**

**特点：求取较方便，运算较方便（可以将减法运算转化为加法运算），“0”有“+0”和“-0”两种不同形式。**

**3.补码**

**求取方法：符号位用0表示正，用1表示负。数值位与符号位相关，当为正数时，数值位不变，即与真值的数值位相同；当为负数时，数值位是真值的数值位按位取反，并在末位加1。**

**例如，若x＝-11010、y＝+0.1010，则**

**[x]补码＝100110（由于x为负数，所以符号位为1，数值位为真值的数值位按位变反，末位加1）**

**[y]补码＝0.1010（由于y为正数，所以有符号位为0，数值位与真值相同）**

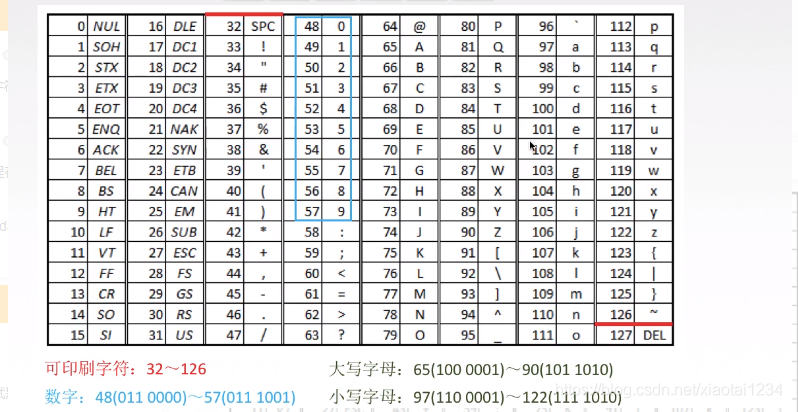
**特点：运算方便（可以将减法运算转化为加法运算），“0”只有“+0”一种形式。**

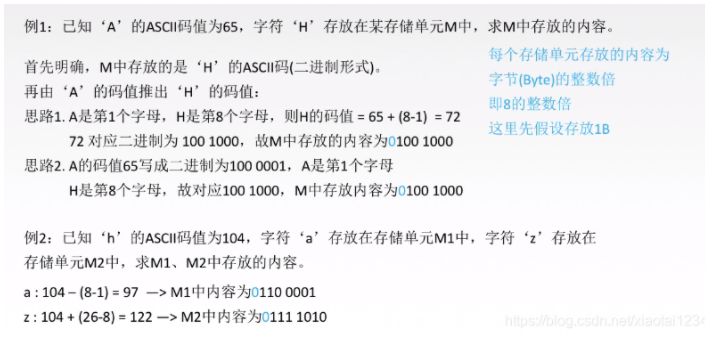




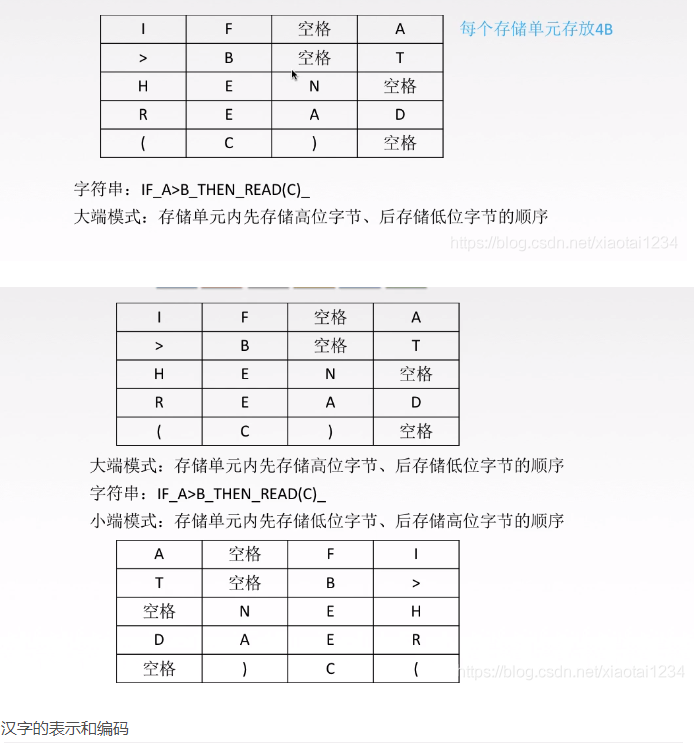
1. **字符与字符串**

**ascII码**

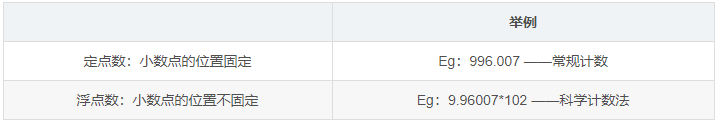




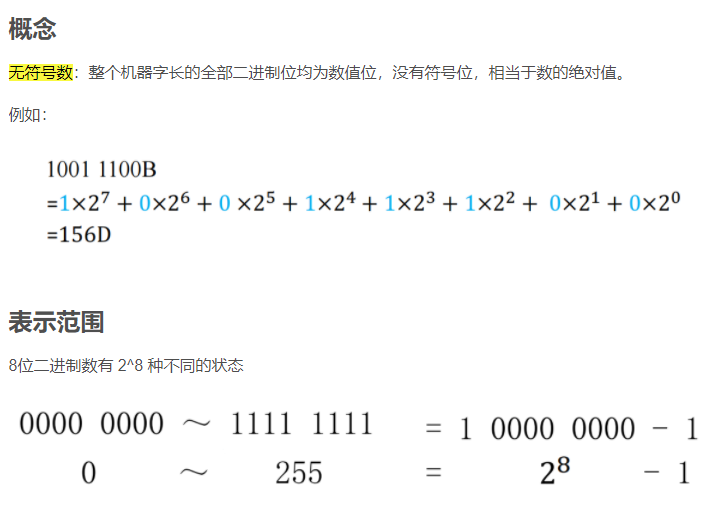
字符串存储原理



1. 定点数的表示和运算1.定点数的表示



无符号数的表示;



有符号整数的表示。

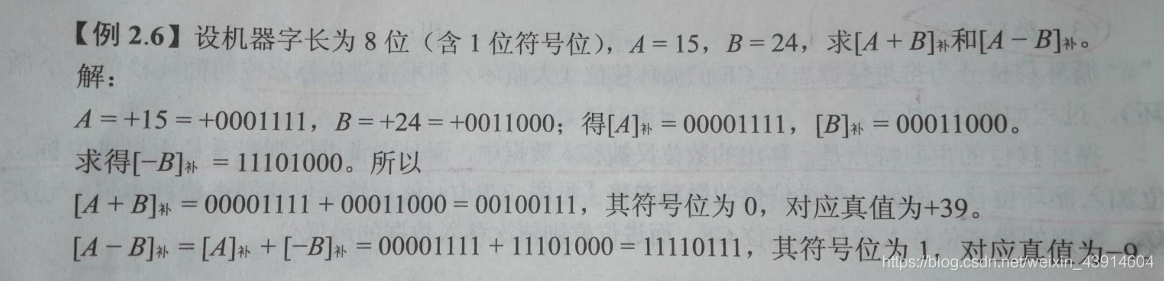
https://blog.csdn.net/weixin\_45525272/article/details/116739831?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425215816780264097832%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425215816780264097832&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-2-116739831.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E5%AE%9A%E7%82%B9%E6%95%B0%E7%9A%84%E8%A1%A8%E7%A4%BA&spm=1018.2226.3001.4187

2.定点数的运算

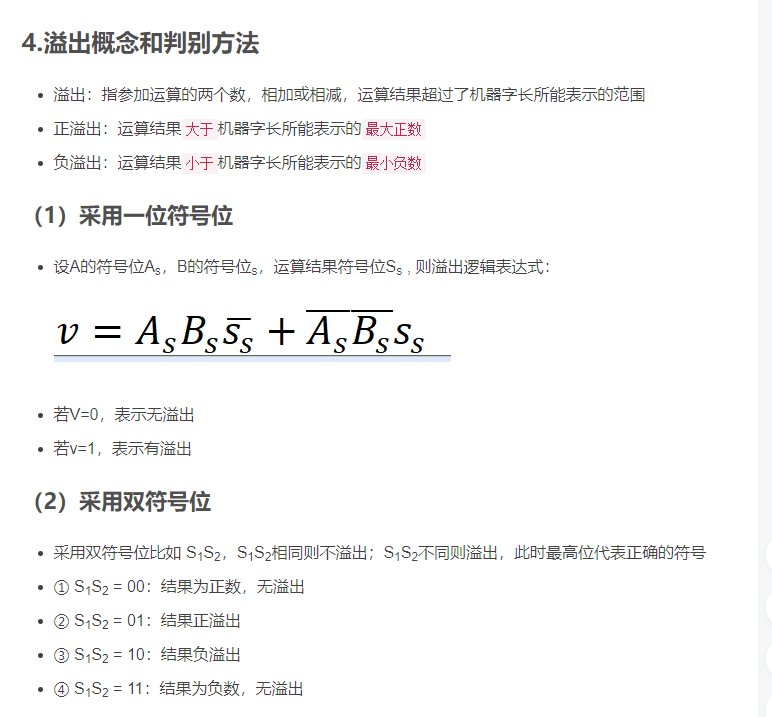
补码定点数的加/减运算;定点数的乘运算;溢出概念和判别方法。

https://blog.csdn.net/weixin\_43914604/article/details/104229538?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425259116780255217948%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request\_id=162425259116780255217948&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_v2~rank\_v29-1-104229538.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E8%A1%A5%E7%A0%81%E5%AE%9A%E7%82%B9%E6%95%B0%E7%9A%84%E5%8A%A0%2F%E5%87%8F%E8%BF%90%E7%AE%97%3B%E5%AE%9A%E7%82%B9%E6%95%B0%E7%9A%84%E4%B9%98%E8%BF%90%E7%AE%97%3B%E6%BA%A2%E5%87%BA%E6%A6%82%E5%BF%B5%E5%92%8C%E5%88%A4%E5%88%AB%E6%96%B9%E6%B3%95%E3%80%82&spm=1018.2226.3001.4187

#### 补码加减：



溢出概念和判断方法

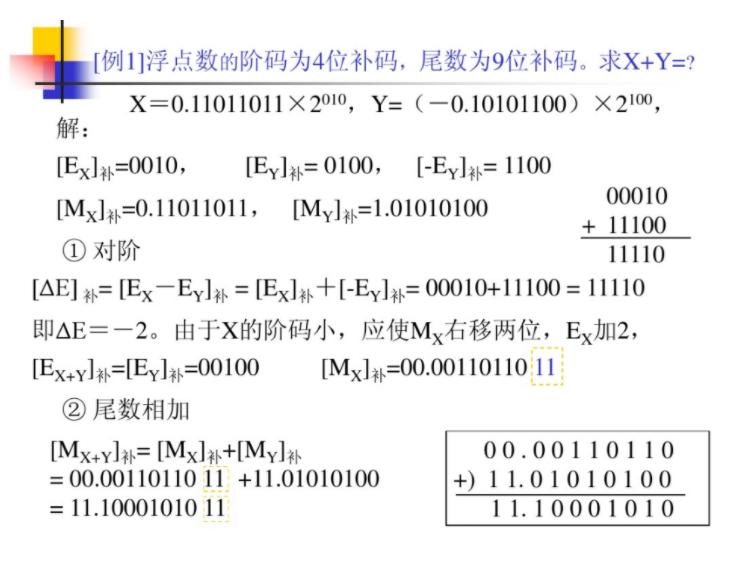


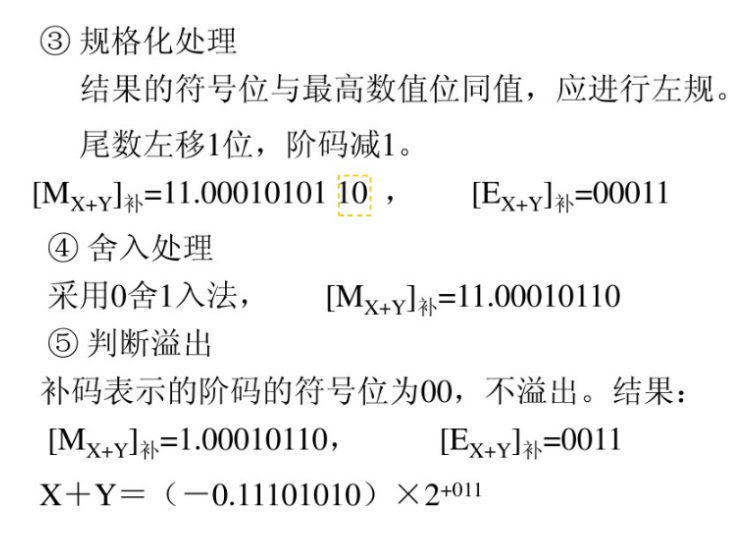
(三)浮点数的表示和运算

1. **浮点数的表示IEEE 754 标 准**



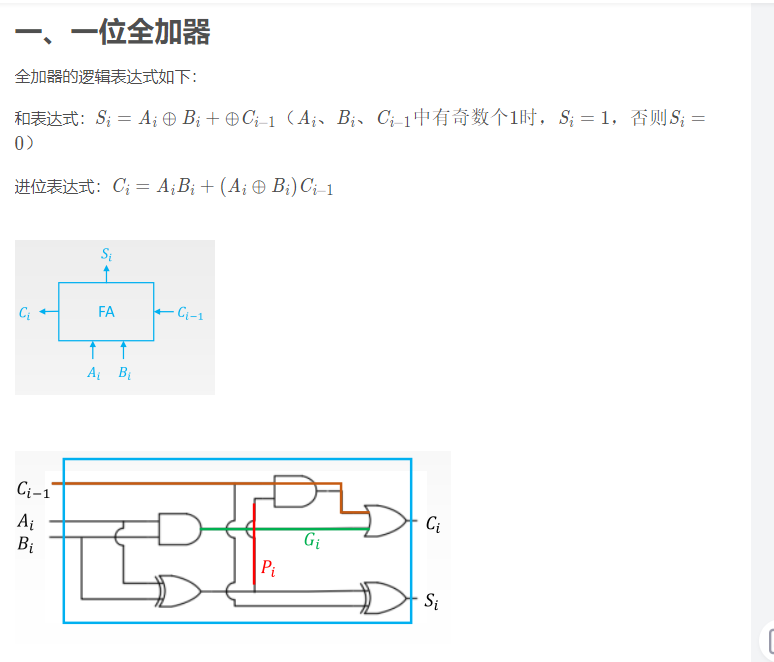
运算例题：

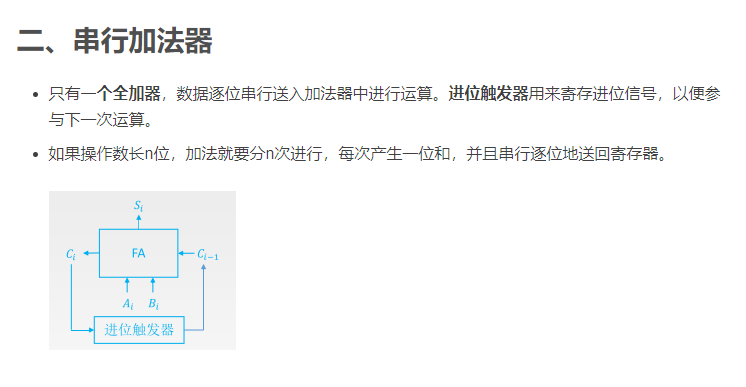




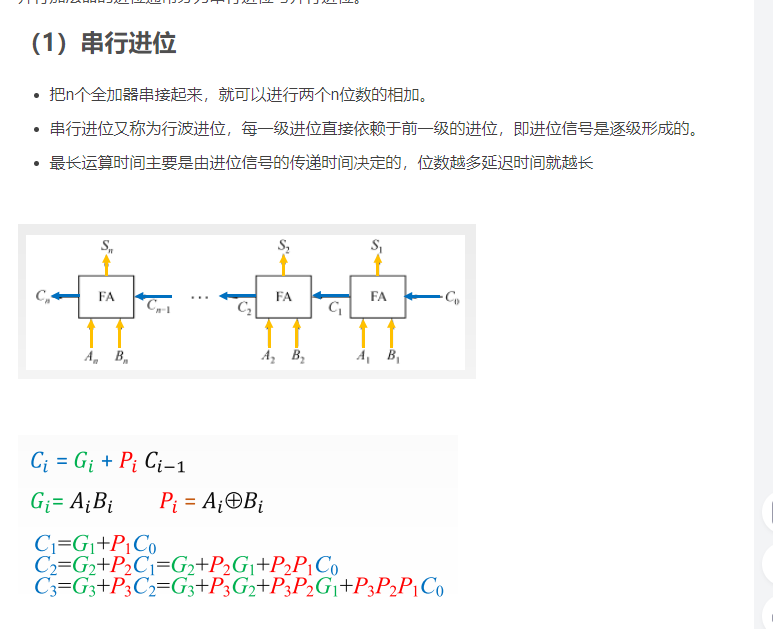
(四)算术逻辑单元 ALU

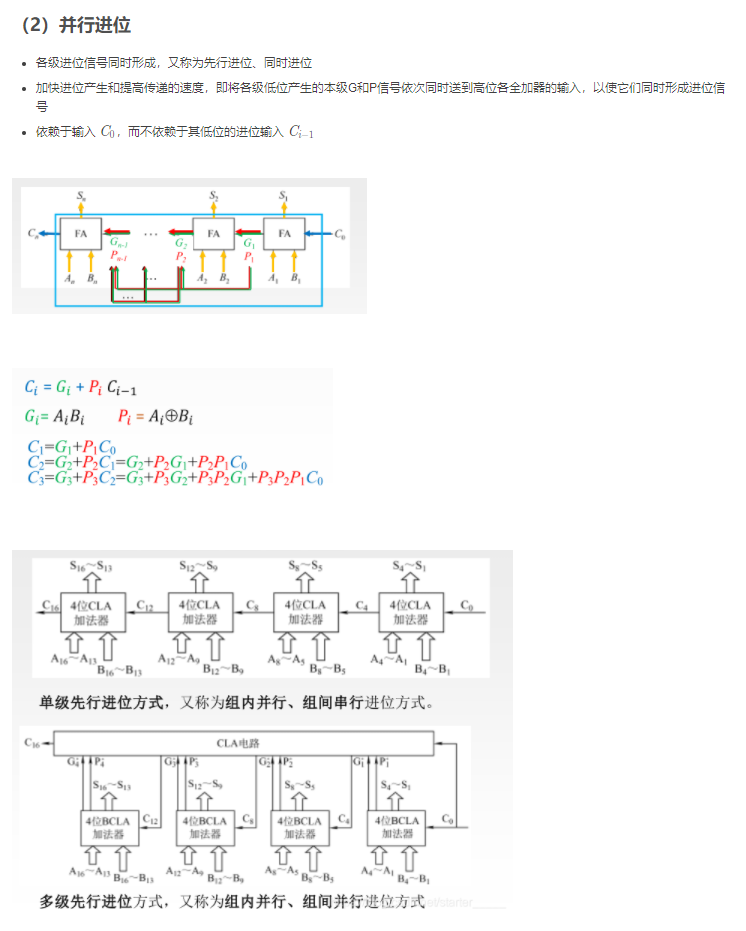
* 1. **串行加法器和并行加法器**



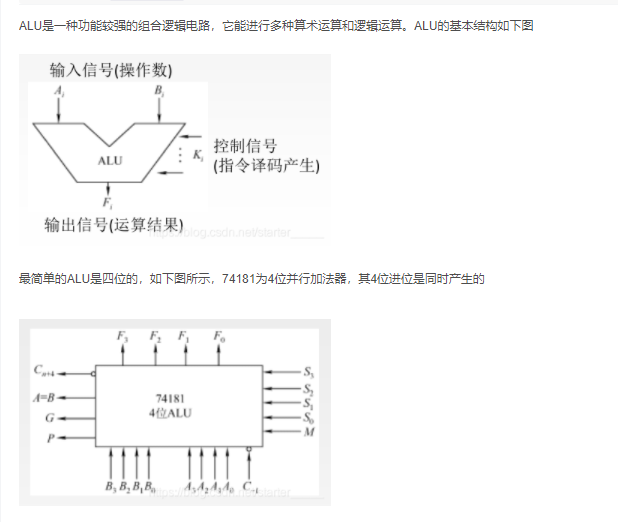


1. 并行全加器



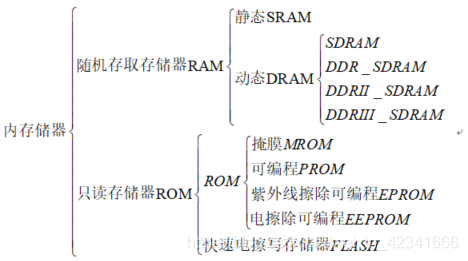


* 1. **算术逻辑单元 ALU 的功能和结构**

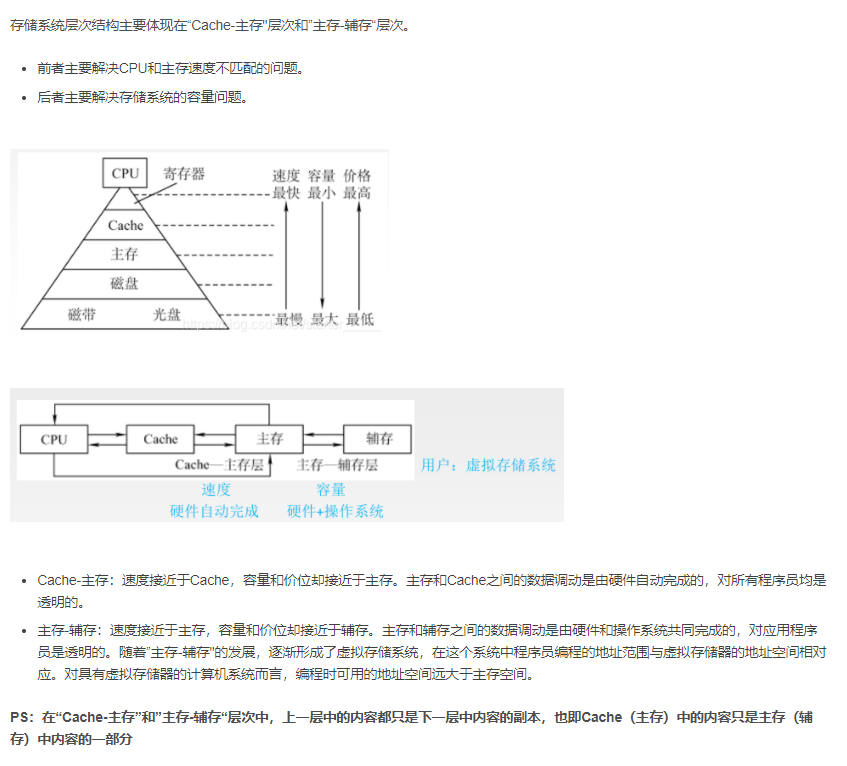


# 三、存储器层次结构

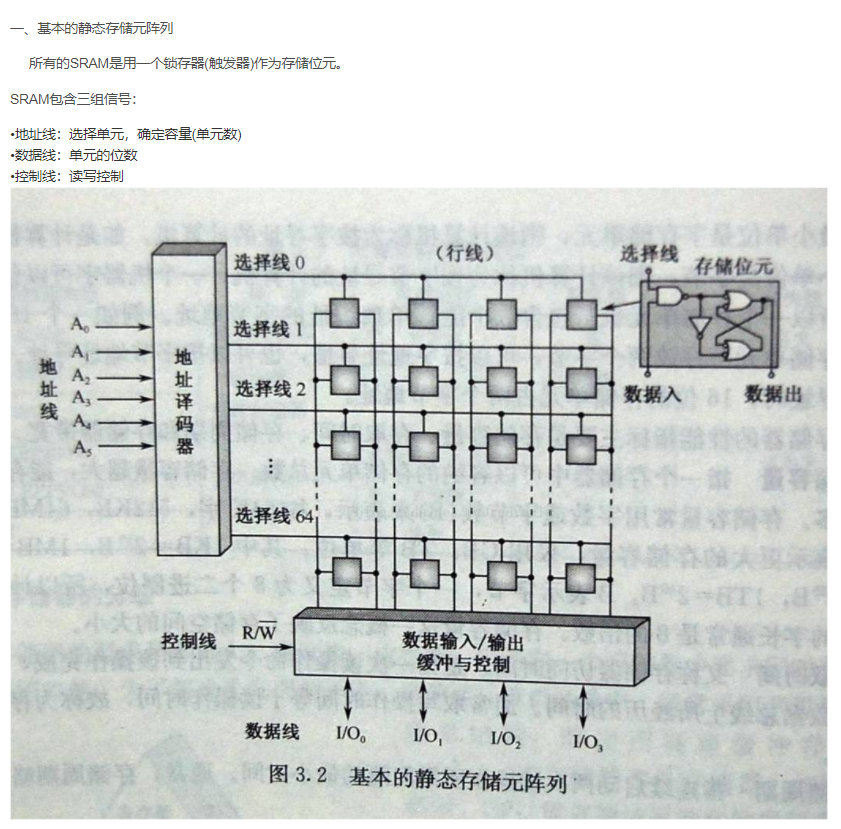
# (一)存储器的分类



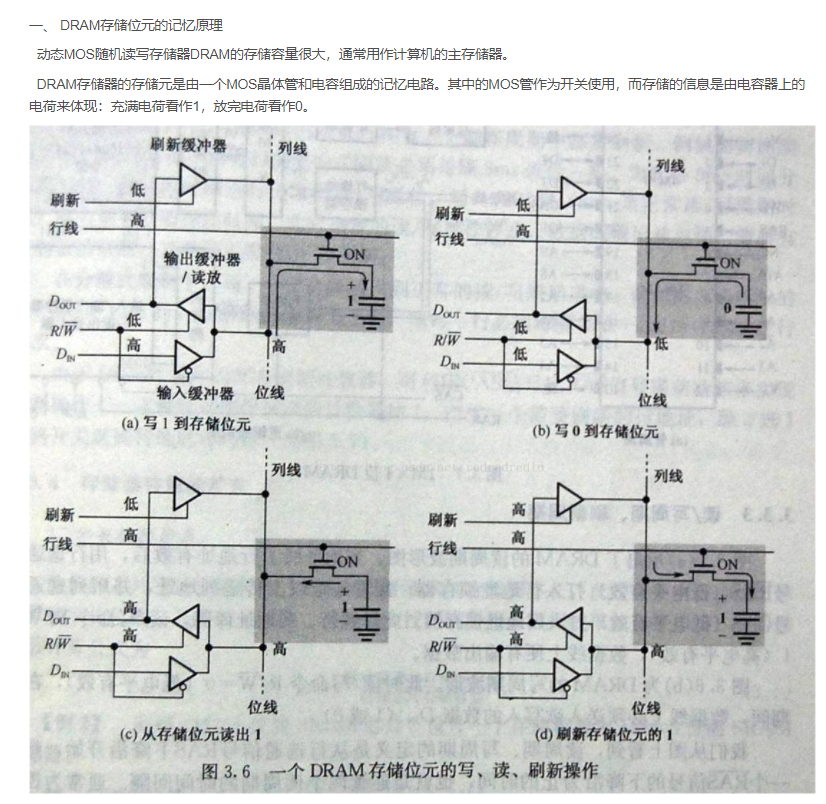
## (二)存储器的层次化结构



1. 半导体随机存取存储器的基本概念1.SRAM 存储器



1. **DRAM 存储器**



(四)主存储器的逻辑设计

https://blog.csdn.net/hpdlzu80100/article/details/115907115?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425346616780262545924%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425346616780262545924&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-115907115.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E4%B8%BB%E5%AD%98%E5%82%A8%E5%99%A8%E7%9A%84%E9%80%BB%E8%BE%91%E8%AE%BE%E8%AE%A1&spm=1018.2226.3001.4187

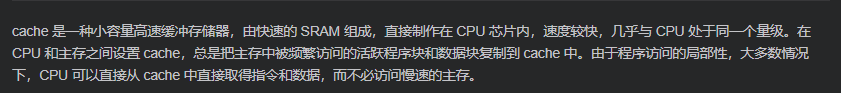
(基本原则和容量拓展)

https://blog.csdn.net/hpdlzu80100/article/details/115907146?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425346616780262545924%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425346616780262545924&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-2-115907146.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E4%B8%BB%E5%AD%98%E5%82%A8%E5%99%A8%E7%9A%84%E9%80%BB%E8%BE%91%E8%AE%BE%E8%AE%A1&spm=1018.2226.3001.4187

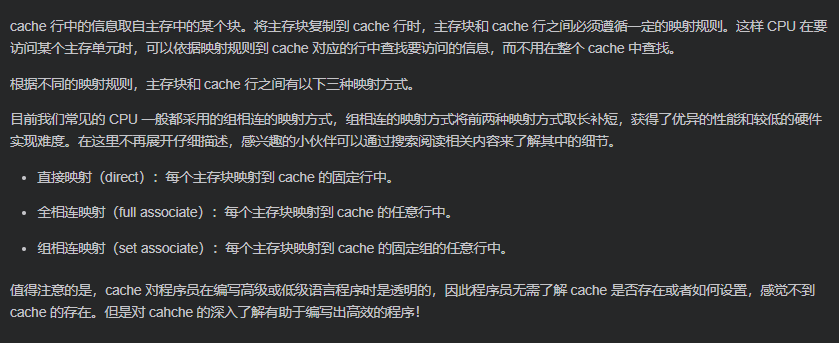
（负载分析和速度估算）

(五)高速缓冲存储器(Cache)

1.Cache 的基本工作原理

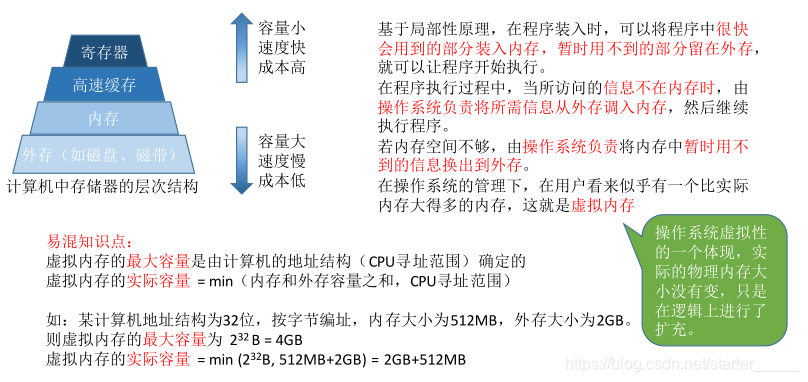


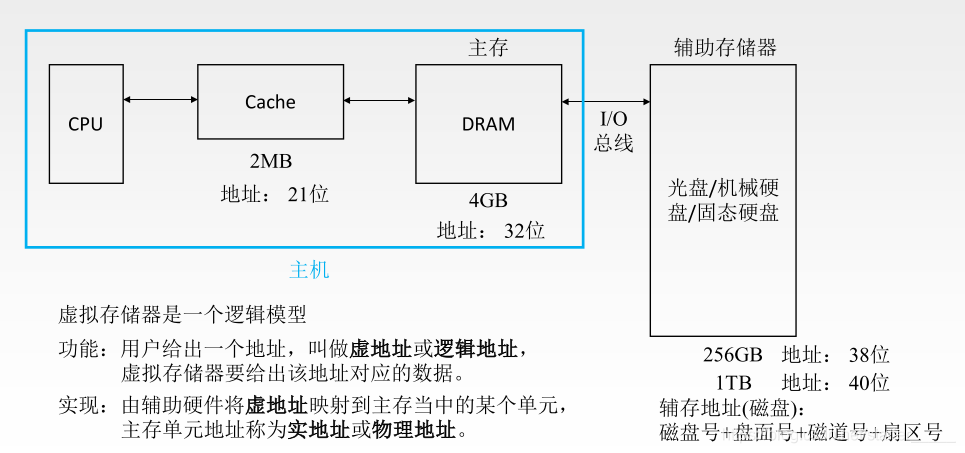
2.Cache 和主存之间的映射方式



(六)虚拟存储器

1.虚拟存储器的基本概念

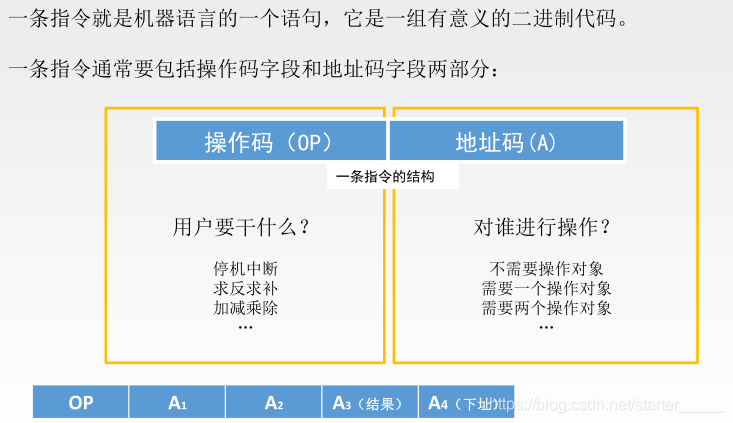




四、指令系统

(一)指令格式

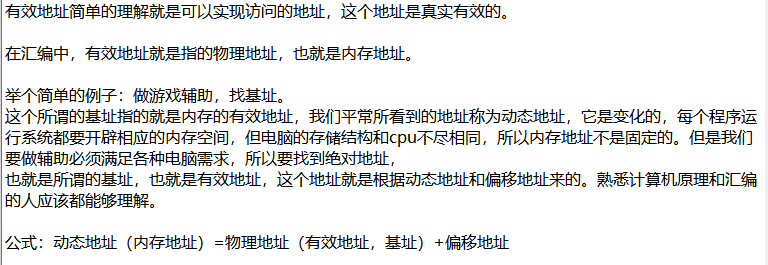
1. **指令的基本格式**



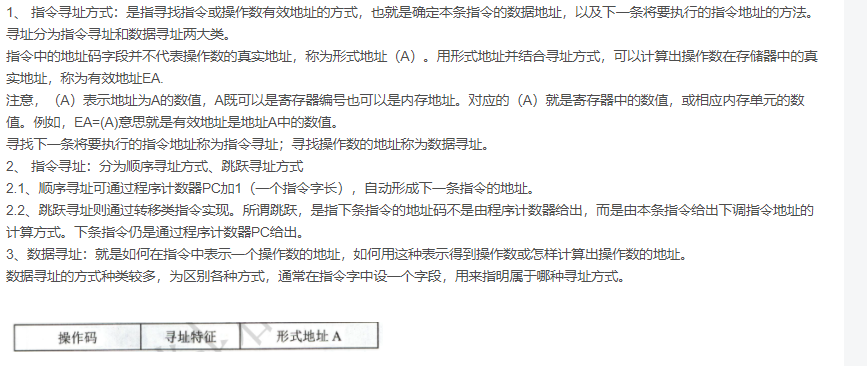
https://blog.csdn.net/starter\_\_\_\_\_/article/details/97755168?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425383616780255291935%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425383616780255291935&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-4-97755168.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E6%8C%87%E4%BB%A4%E7%9A%84%E5%9F%BA%E6%9C%AC%E6%A0%BC%E5%BC%8F&spm=1018.2226.3001.4187

**2，指令的寻址方式**

**1.有效地址的概念**



1. **数据寻址和指令寻址**



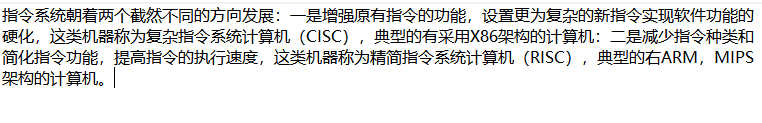
1. **常见寻址方式**

https://blog.csdn.net/duan\_qiao925/article/details/106166866?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425446716780264035663%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request\_id=162425446716780264035663&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_v2~rank\_v29-1-106166866.first\_rank\_v2\_pc\_rank\_v29&utm\_term=3.%E5%B8%B8%E8%A7%81%E5%AF%BB%E5%9D%80%E6%96%B9%E5%BC%8F&spm=1018.2226.3001.4187

（十种）

(三)CISC 和 RISC 的基本概念

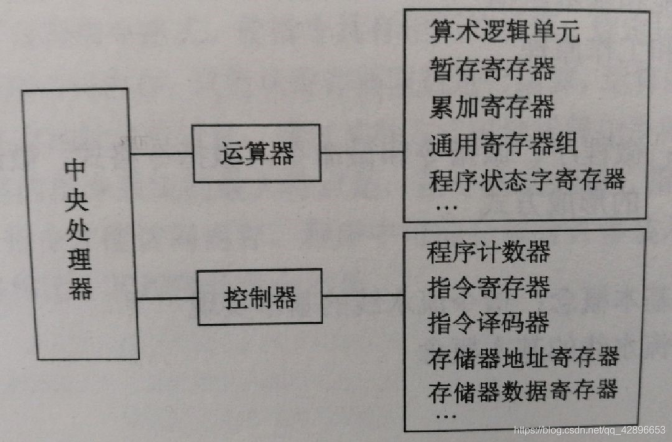




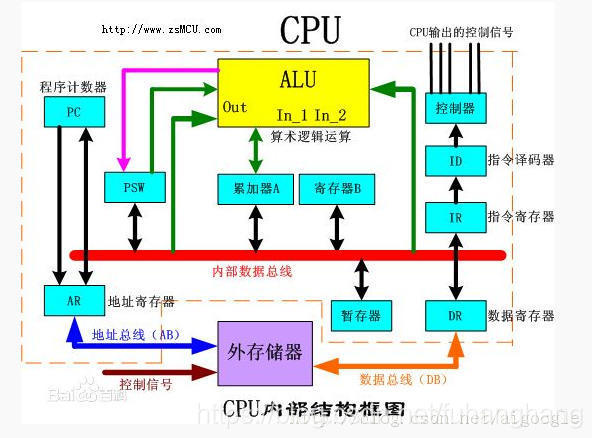
五、中央处理器(CPU)

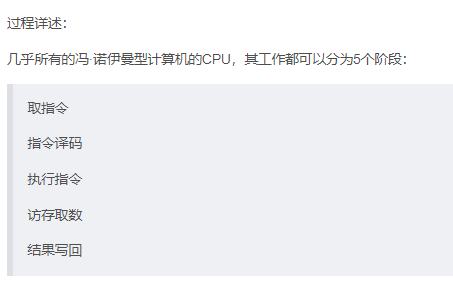
# CPU 的功能和基本结构





指令执行过程





## 数据通路的功能和基本结构

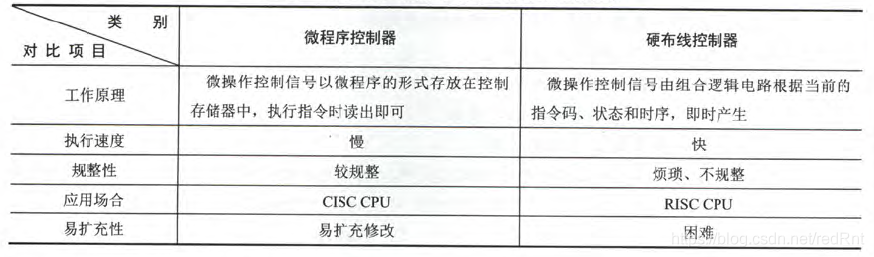
(三)控制器的功能和工作原理

1、硬布线控制器

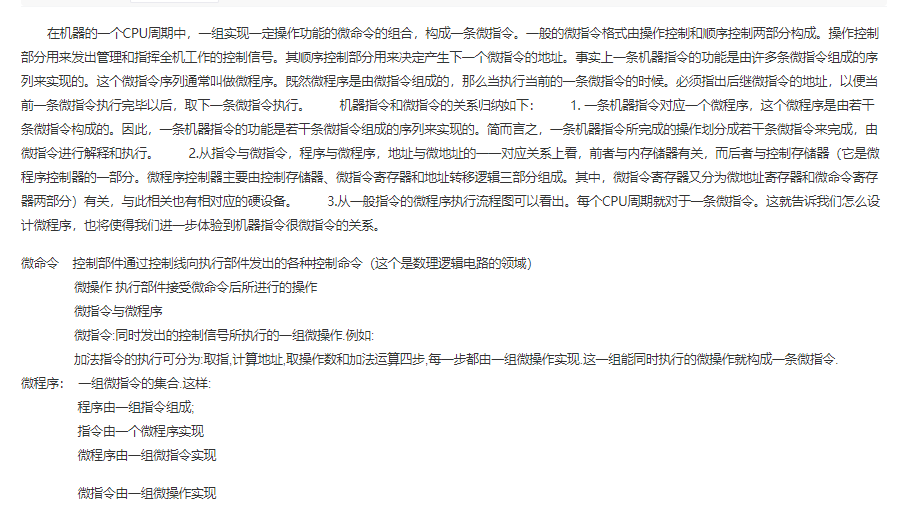
https://blog.csdn.net/redredredlu/article/details/45477285?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425513716780269828284%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425513716780269828284&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-45477285.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E7%A1%AC%E5%B8%83%E7%BA%BF%E6%8E%A7%E5%88%B6%E5%99%A8&spm=1018.2226.3001.4187

2、微程序控制器

https://blog.csdn.net/WWIandMC/article/details/105523503?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425516116780269879789%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425516116780269879789&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~top\_click~default-2-105523503.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E5%BE%AE%E7%A8%8B%E5%BA%8F%E6%8E%A7%E5%88%B6%E5%99%A8&spm=1018.2226.3001.4187



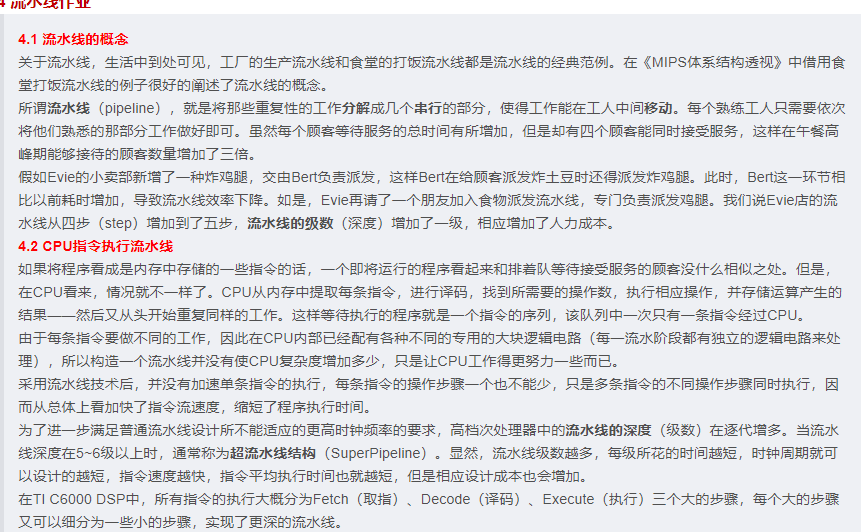
微程序、微指令和微命令;微指令的编码方式;微地址的形式方式。



https://blog.csdn.net/weixin\_51340156/article/details/112277552?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425522016780261948703%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425522016780261948703&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-112277552.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E5%BE%AE%E6%8C%87%E4%BB%A4%E7%9A%84%E7%BC%96%E7%A0%81%E6%96%B9%E5%BC%8F&spm=1018.2226.3001.4187 （微指令的编码方式）

(四)指令流水线

1. 指令流水线的基本概念



六、总线

(一)总线概述

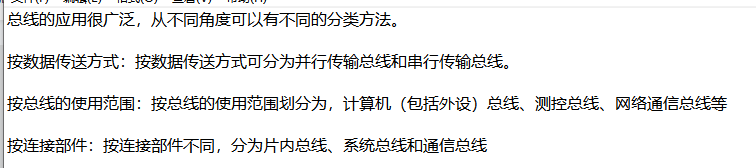
1、总线的基本概念

总线是连接多个部件的信息传输线，是各部件共享的传输介质。当多个部件与总线相连时，如果出现两个或两个以上部件同时向总线发送信息，势必导致信号冲突，传输无效。因此，在某一时刻，只允许有一个部件向总线发送信息，而多个部件可以同时从总线上接受相同的信息。

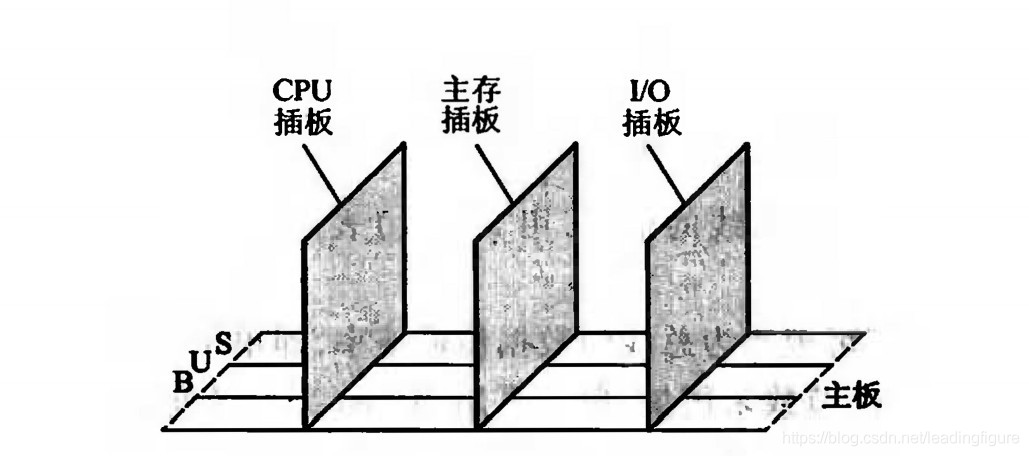
存储总线：连接 CPU 和主存，称为存储总线（M总线）

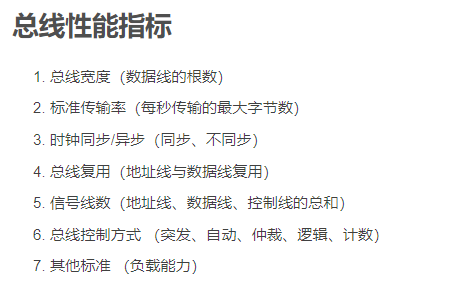
输入/输出总线：用来建立 CPU 和各 I/O 设备之间交换信息的通道，称为输入/输出总线（I/O 总线）。各种 I/O 设备通过 I/O 接口挂到 I/O 总线上，更便于增删设备。

2、总线的分类



1. 总线的组成及性能指标





(二)总线操作和定时

1、同步定时方式

同步定时方式要求所有的模块由统一的始终脉冲进行操作的控制，各模块的所有动作均在时钟周期的开始产生，并且多数动作在一个时钟周期内完成。

1. 异步定时方式

异步定时方式是一种应答方式或者互锁机制的定时方式。对于异步操作，操作的发生由主设备或从设备的的特定信号来确定。总线上一个事件的发生取决于前一个事件的发生，双方互相提供联络信号。

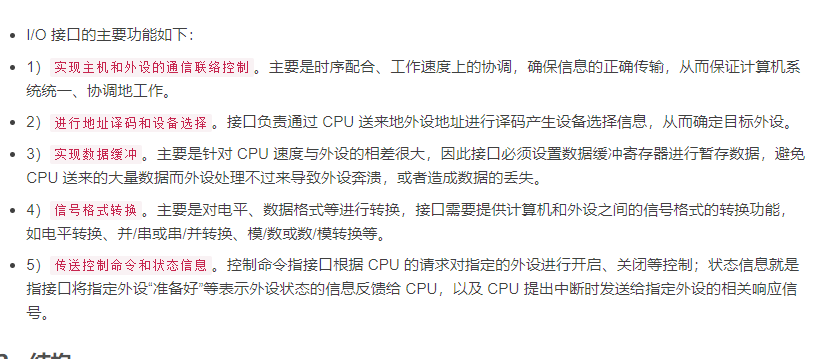
1. (三)总线标准
2. 输入输出(I/O)系统

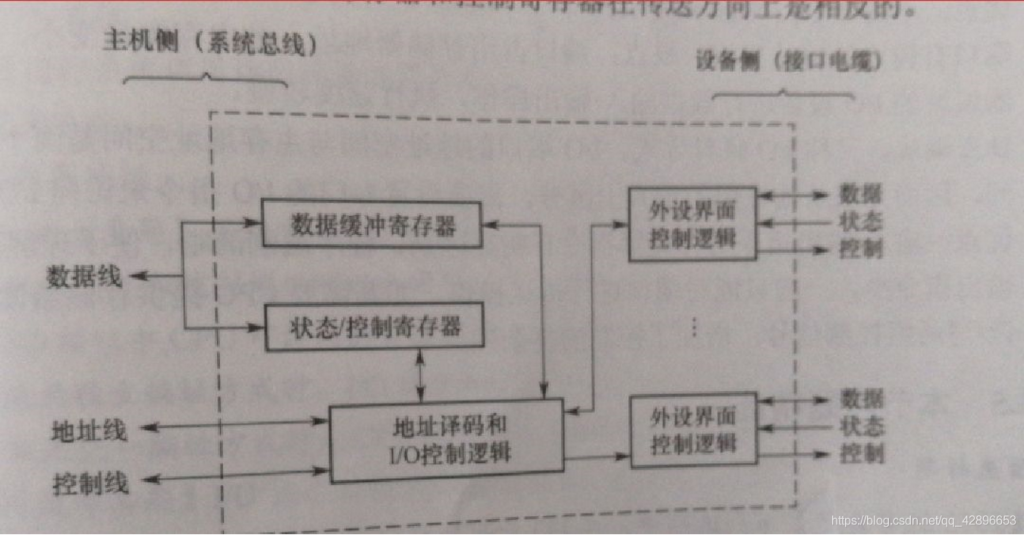
(一)I/O 系统基本概念



(二)I/O 接口(I/O 控制器)

1、I/O 接口的功能和基本结构





1. I/O 端口及其编址

I/O 接口电路也简称接口电路。它是主机和外围设备之间交换信息的连接部件（电路）。

它在主机和外围设备之间的信息交换中起着桥梁和纽带作用。

设置接口电路的必要性：

a) 解决主机CPU 和外围设备之间的时序配合和通信联络问题。

b) 解决CPU 和外围设备之间的数据格式转换和匹配问题。

c) 解决CPU 的负载能力和外围设备端口选择问题。

I/O 接口的编址方式：

① I/O 接口独立编址：

这种编址方式是将存储器地址空间和I/O 接口地址空间分开设置，互不影响。设有专门的输入指令（IN）和输出指令（OUT）来完成I/O 操作。

② I/O 接口与存储器统一编址方式：

这种编址方式不区分存储器地址空间和I/O 接口地址空间，把所有的I/O 接口的端口都当作是存储器的一个单元对待，每个接口芯片都安排一个或几个与存储器统一编号的地址号。也不设专门的输入/输出指令，所有传送和访问存储器的指令都可用来对I/O 接口操作。

③两种编址方式有各自的优缺点。

独立编制方式的主要优点是内存地址空间与I/O 接口地址空间分开，互不影响，译码电路较简单，并设有专门的I/O 指令，所以编程序易于区分，且执行时间短，快速性好。其缺点是只用I/O 指令访问I/O 端口，功能有限且要采用专用I/O周期和专用I/O 控制线，使微处理器复杂化。统一编制方式的主要优点是访问内存的指令都可用于I/O 操作，数据处理功能强；同时I/O 接口可与存储器部分共用译码和控制电路。其缺点一是I/O 接口要占用存储器地址空间的一部分；二是因不用专门的I/O 指令，程序中较难区分I/O 操作。

(三)I/O 方式

1、程序查询方式

https://blog.csdn.net/u014106644/article/details/94539483?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425571616780366532712%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425571616780366532712&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~sobaiduend~default-1-94539483.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E7%A8%8B%E5%BA%8F%E6%9F%A5%E8%AF%A2%E6%96%B9%E5%BC%8F&spm=1018.2226.3001.4187

2、程序中断方式

https://blog.csdn.net/starter\_\_\_\_\_/article/details/100181134?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522162425574616780271533725%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=162425574616780271533725&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~top\_positive~default-1-100181134.first\_rank\_v2\_pc\_rank\_v29&utm\_term=%E7%A8%8B%E5%BA%8F%E4%B8%AD%E6%96%AD%E6%96%B9%E5%BC%8F&spm=1018.2226.3001.4187

中断的基本概念;中断响应过程;中断处理过程;多重中断和中断屏蔽的概念。

3、DMA 方式

DMA 控制器的基本概念。

8.3.1 DMA的基本概念

　　直接内存访问(DMA)是一种完全由硬件执行I/O交换的工作方式。在这种方式中，DMA控制器从CPU完全接管对总线的控制，数据交换不经过CPU，而直接在内存和I/O设备之间进行 。DMA方式一般用于高速传送成组数据。DMA控制器将向内存发出地址和控制信号，修改地址，对传送的字的个数计数，并且以中断方式向CPU报告传送操作的结束。

　　DMA方式的主要优点是速度快。由于CPU根本不参加传送操作，因此就省去了CPU取指令、取数、送数等操作。在数据传送过程中，没有保存现场、恢复现场之类的工作。内存地址修改、传送字个数的计数等等，也不是由软件实现，而是用硬件线路直接实现的。所以DMA方式能满足高速I/O设备的要求，也有利于CPU效率的发挥。