Lecture 5: Data Representation 2

数据的机器级表示 2

十进制数的表示

- 数值数据（numerical data）的两种表示

  Binary (二进制数)

  - 定点整数：Fixed-point number (integer)
    - Unsigned and signed int
  - 浮点数：Floating-point number (real number)

  Decimal (十进制数)

  - 用ASCII码表示
  - 用BCD（Binary coded Decimal）码表示

- 计算机中为什么要用十进制数表示数值？
  - 日常使用的都是十进制数，所以，计算机外部都使用十进制数。在一些有大量数据输入/出的系统中，为减少二进制数和十进制数之间的转换，在计算机内部直接用十进制数表示数值。

用ASCII码表示十进制数

- 前分隔数字串
  - 符号位单独用一个字节表示，位于数字串之前。
  - 正号用“+”的ASCII码(2BH)表示；负号用“-”的ASCII码(2DH)表示
  - 例：十进制数+236表示为: 2B 32 33 36H

    0010 1011 0011 0010 0011 0011 0011 0110B

    十进制数-2369表示为: 2D 32 33 36 39H

    0010 1101 0011 0010 0011 0011 0011 0110 0011 1001B
- 后嵌入数字串
  - 符号位嵌入最低位数字的ASCII码高4位中。比前分隔方式省一个字节。
  - 正数不变；负数高4位变为0111.
  - 例：十进制数+236表示为: 32 33 36H

    0011 0010 0011 0011 0011 0110B

    十进制数-2369表示为: 32 33 36 79H

    0011 0010 0011 0011 0011 0110 0111 1001B

缺点：占空间大，且需转换成二进制数或BCD码才能计算。

用BCD码表示十进制数

- 编码思想： 每个十进数位至少有4位二进制表示。而4位二进制位可组合成16种状态，去掉10种状态后还有6种冗余状态。
- 编码方案
  1. 十进制有权码
     - 每个十进制数位的4个二进制位（称为基2码）都有一个确定的权。8421码是最常用的十进制有权码。也称自然BCD（NBCD）码。
  2. 十进制无权码
     - 每个十进制数位的4个基2码没有确定的权。在无权码方案中，用的较多的是余3码和格雷码。
  3. 其他编码方案 （5中取2码、独热码等）
- 符号位的表示：
  - “+”：1100 ； “-”：1101
  - 例：+236=$(1100\ 0010\ 0011\ 0110)_{8421}$ (占2个字节)

    - 2369=$(1101\ 0000\ 0010\ 0011\ 0110\ 1001)_{8421}$ (占3个字节)

    补0以使数占满一个字节

逻辑数据的编码表示

- 表示
  - 用一位表示 真：1 / 假：0
  - N位二进制数可表示N个逻辑数据，或一个位串
- 运算
  - 按位进行
  - 如:按位与 / 按位或 / 逻辑左移 / 逻辑右移 等
- 识别
  - 逻辑数据和数值数据在形式上并无差别，也是一串0/1序列，机器靠指令来识别。

西文字符的编码表示

- 特点
  - 是一种拼音文字，用有限几个字母可拼写出所有单词
  - 只对有限个字母和数学符号、标点符号等辅助字符编码
  - 所有字符总数不超过256个，使用7或8个二进位可表示
- 表示（常用编码为7位ASCII码）

  要求必须熟悉数字、字母和空格(SP)的表示
  - 十进制数字：0/1/2…/9
  - 英文字母：A/B/…/Z/a/b/…/z
  - 专用符号：+/-/%/*/&/……
  - 控制字符（不可打印或显示）
- 操作
  - 字符串操作，如:传送/比较　等

汉字及国际字符的编码表示

- 特点
  - 汉字是表意文字，一个字就是一个方块图形。
  - 汉字数量巨大，总数超过6万字，给汉字在计算机内部的表示、汉字的传输与交换、汉字的输入和输出等带来了一系列问题。
- 编码形式
  - 有以下几种汉字代码：

    输入码：对汉字用相应按键进行编码表示，用于输入

    内码：用于在系统中进行存储、查找、传送等处理

    字模点阵或轮廓描述: 描述汉字字模点阵或轮廓，用于显示/打印

问题：西文字符有没有输入码？有没有内码？
有没有字模点阵或轮廓描述？

汉字的输入码

向计算机输入汉字的方式：

① 手写汉字联机识别输入，或者是印刷汉字扫描输入后自动识别，这两种方法现均已达到实用水平。

② 用语音输入汉字，虽然简单易操作，但离实用阶段还相差很远。

③ 利用英文键盘输入汉字：每个汉字用一个或几个键表示，这种对每个汉字用相应按键进行的编码称为汉字“输入码”，又称外码。输入码的码元为按键。是最简便、最广泛的汉字输入方法。

常用的方法有：搜狗拼音、五笔字型、智能ABC、微软拼音等

使用汉字输入码的原因：

① 键盘面向西文设计，一个或两个西文字符对应一个按键，非常方便。

② 汉字是大字符集，专门的汉字输入键盘由于键多、查找不便、成本高等原因而几乎无法采用。

字符集与汉字的内码

问题：西文字符常用的内码是什么？其内码就是ASCII码。

对于汉字内码的选择，必须考虑以下几个因素：

① 不能有二义性，即不能和ASCII码有相同的编码。

② 尽量与汉字在字库中的位置有关，便于汉字查找和处理。

③ 编码应尽量短。

国标码（国标交换码）

1981年我国颁布了《信息交换用汉字编码字符集·基本集》(GB2312—80)。该标准选出6763个常用汉字，为每个汉字规定了标准代码，以供汉字信息在不同计算机系统间交换使用

可在汉字国标码的基础上产生汉字机内码

GB2312-80字符集

- 由三部分组成：
  - ① 字母、数字和各种符号，包括英文、俄文、日文平假名与片假名、罗马字母、汉语拼音等共687个
  - ② 一级常用汉字，共3755个，按汉语拼音排列
  - ③ 二级常用汉字，共3008个 ，不太常用，按偏旁部首排列
- 汉字的区位码
  - 码表由94行、94列组成，行号为区号，列号为位号，各占7位
  - 指出汉字在码表中的位置，共14位，区号在左、位号在右
- 汉字的国标码
  - 每个汉字的区号和位号各自加上32（20H），得到其“国标码”
  - 国标码中区号和位号各占7位。在计算机内部，为方便处理与存储，前面添一个0，构成一个字节

汉字内码

- 至少需2个字节才能表示一个汉字内码。为什么？
  - 由汉字的总数决定！
- 可在GB2312国标码的基础上产生汉字内码
  - 为与ASCII码区别，将国标码的两个字节的第一位置“1”后得到一种汉字内码

例如，汉字“大”在码表中位于第20行、第83列。因此区位码为0010100 1010011，国标码为00110100 01110011，即3473H。前面的34H和字符“4”的ACSII码相同，后面的73H和字符“s”的ACSII码相同，将每个字节的最高位各设为“1”后，就得到其内码：B4F3H (1011 0100 1111 0011B)，因而不会和ASCII码混淆。

国际字符集

国际字符集的必要性

- 不同地区使用不同字符集内码，如中文GB2312 / Big5、日文Shift-JIS / EUC-JP等。在安装中文系统的计算机中打开日文文件，会出现乱码。
- 为使所有国际字符都能互换，必须创建一种涵盖全部字符的多字符集。

国际多字符集

- 通过对各种地区性字符集规定使用范围来唯一定义各字符的编码。
- 国际标准ISO/IEC 10646提出了一种包括全世界现代书面语言文字所使用的所有字符的标准编码，有4个字节编码(UCS-4)和2字节编码(UCS-2) 。
- 我国（包括香港、台湾地区）与日本、韩国联合制订了一个统一的汉字字符集（CJK编码），共收集了上述不同国家和地区共约2万多汉字及符号，采用2字节编码 (即：UCS-2) ，已被批准为国家标准(GB13000)。
- Windows操作系统(中文版)已采用中西文统一编码，收集了中、日、韩三国常用的约2万汉字，称为“Unicode”，采用2字节编码，与UCS-2一致 。

汉字的字模点阵码和轮廓描述

- 为便于打印、显示汉字，汉字字形必须预先存在机内
  - 字库 (font)：所有汉字形状的描述信息集合
  - 不同字体 (如宋体、仿宋、楷体、黑体等) 对应不同字库
  - 从字库中找到字形描述信息，然后送设备输出
- 字形主要有两种描述方法：
  - 字模点阵描述（图像方式）
  - 轮廓描述（图形方式）
    - 直线向量轮廓
    - 曲线轮廓（True Type字形）

数据的基本宽度

- 比特（bit）是计算机中处理、存储、传输信息的最小单位
- 二进制信息的计量单位是“字节”(Byte)，也称“位组”
  - 现代计算机中，存储器按字节编址
  - 字节是最小可寻址单位 *(addressable* unit *)*
- 除比特和字节外，还经常使用“字”(word)作为单位
- “字”和 “字长”的概念不同

数据的基本宽度

- “字”和“字长”的概念不同
  - “字长”指数据通路的宽度。

    （数据通路指CPU内部数据流经的路径以及路径上的部件，主要是CPU内部进行数据运算、存储和传送的部件，这些部件的宽度基本上要一致，才能相互匹配。因此，“字长”等于CPU内部总线的宽度、运算器的位数、通用寄存器的宽度等。）
  - “字”表示被处理信息的单位，用来度量数据类型的宽度。
  - 字和字长的宽度可以一样，也可不同。

    例如，x86体系结构定义“字”的宽度为16位，但从386开始字长就是32位了。

数据量的度量单位

- 存储二进制信息时的度量单位要比字节或字大得多
- 容量经常使用的单位有：
  - “千字节”(KB)，1KB=2^{10}字节=1024B
  - “兆字节”(MB)，1MB=2^{20}字节=1024KB
  - “千兆字节”(GB)，1GB=2^{30}字节=1024MB
  - “兆兆字节”(TB)，1TB=2^{40}字节=1024GB
- 通信中的带宽使用的单位有：
  - “千比特/秒”(kb/s)，1kbps=10^3 b/s=1000 bps
  - “兆比特/秒”(Mb/s)，1Mbps=10^6 b/s =1000 kbps
  - “千兆比特/秒”(Gb/s)，1Gbps=10^9 b/s =1000 Mbps
  - “兆兆比特/秒”(Tb/s)，1Tbps=10^{12} b/s =1000 Gbps

如果把b换成B，则表示字节而不是比特（位）

例如，10MBps表示 10兆字节/秒

程序中数据类型的宽度

- 高级语言支持多种类型、多种长度的数据
  - 例如，C语言中Char类型的宽度为1个字节，可表示一个字符（非数值数据），也可表示一个8位的整数（数值数据）
  - 不同机器上表示的同一种类型的数据可能宽度不同
- 必须确定相应的机器级数据表示方式和相应的处理指令
  (在第五章指令系统介绍具体指令)

C语言中数值数据类型的宽度 (单位：字节)

C声明	典型32位机器	Compaq Alpha机器
char	1	1
short int	2	2
int	4	4
long int	4	8
char*	4	8
float	4	4
double	8	8

从表中看出：同类型数据并不是所有机器都采用相同的宽度，分配的字节数随机器字长和编译器的不同而不同。

Compaq Alpha是一个针对高端应用的64位机器，即字长为64位

数据的存储和排列顺序

- 80年代开始，几乎所有机器都用字节编址
- ISA设计时要考虑的两个问题：
  - 如何从一个字节地址中取到一个32位的字？- 字的存放问题
  - 一个字能否存放在任何字节边界？- 字的边界对齐问题

例如，若 int i = 0x01234567，存放在内存100号单元，则用“取数”指令访问100号单元取出 i 时，必须清楚i的4个字节是如何存放的。

Word:

01	23	45	67
103	102	101	100
msb			lsb
100	101	102	103

little endian word 100

big endian word 100

大端方式（Big Endian）：MSB所在的地址是数的地址

e.g. IBM 360/370, Motorola 68k, MIPS, Sparc, HP PA

小端方式（ Little Endian）：LSB所在的地址是数的地址

e.g. Intel 80x86, DEC VAX

有些机器两种方式都支持，可以通过特定的控制位来设定采用哪种方式。

BIG Endian versus Little Endian

Ex1: Memory layout of a number ABCDH located in 1000

In Big Endian: →

CD	1001
AB	1000

In Little Endian: →

AB	1001
CD	1000

Ex2: Memory layout of a number 00ABCDEFH located in 1000

In Big Endian: →

00	1000
AB	1001
CD	1002
EF	1003

In Little Endian: →

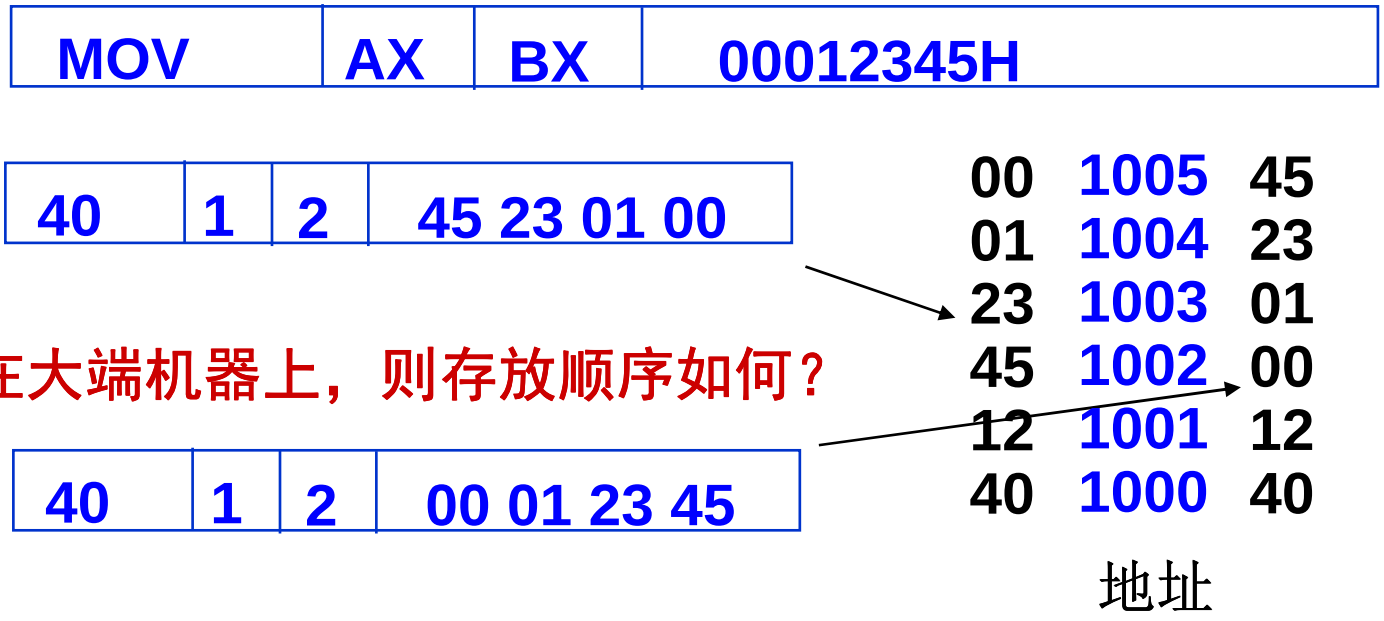
00	1003
AB	1002
CD	1001
EF	1000

BIG Endian versus Little Endian

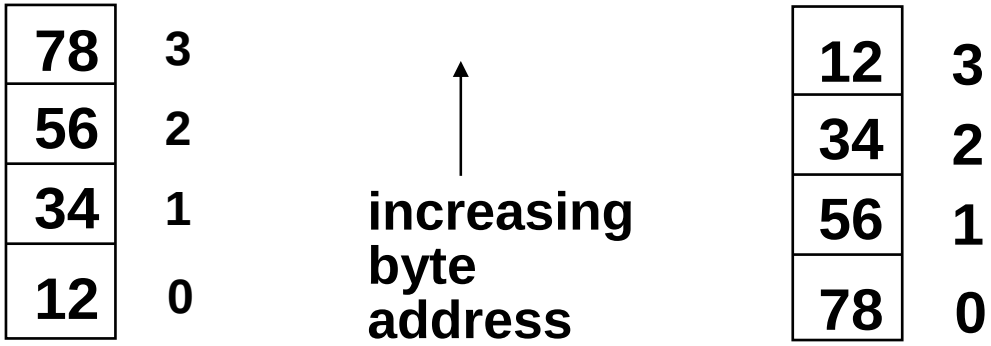
Ex3: Memory layout of a instruction located in 1000

假定小端机器中指令：**mov AX, 0x12345(BX)**

其中操作码**mov**为**40H**，寄存器**AX**和**BX**分别为**0001B**和**0010B**，立即数占**32**位，则存放顺序为：



Byte Swap Problem（字节交换问题）



上述存放在0号单元的数据（字）是什么？12345678H？ 78563412H？

存放方式不同的机器间程序移植或数据通信时，会发生什么问题？

- 每个系统内部是一致的，但在系统间通信时可能会发生问题！
- 因为顺序不同，需要进行顺序转换

音、视频和图像等文件格式或处理程序都涉及到字节顺序问题

ex. Little endian: GIF, PC Paintbrush, Microsoft RTF,etc

Big endian: Adobe Photoshop, JPEG, MacPaint, etc

Alignment(对齐)

Alignment: 要求数据的地址是相应的边界地址

- 目前机器字长一般为32位或64位，而存储器地址按字节编址
- 指令系统支持对字节、半字、字及双字的运算，也有位处理指令
- 各种不同长度的数据存放时，有两种处理方式：
  - 按边界对齐 （假定字的宽度为32位，按字节编址）
    - 字地址：4的倍数(低两位为0)
    - 半字地址：2的倍数(低位为0)
    - 字节地址：任意
  - 不按边界对齐

    坏处：可能会增加访存次数！

    （学了第四章存储器组织后会更明白！）

Alignment(对齐)

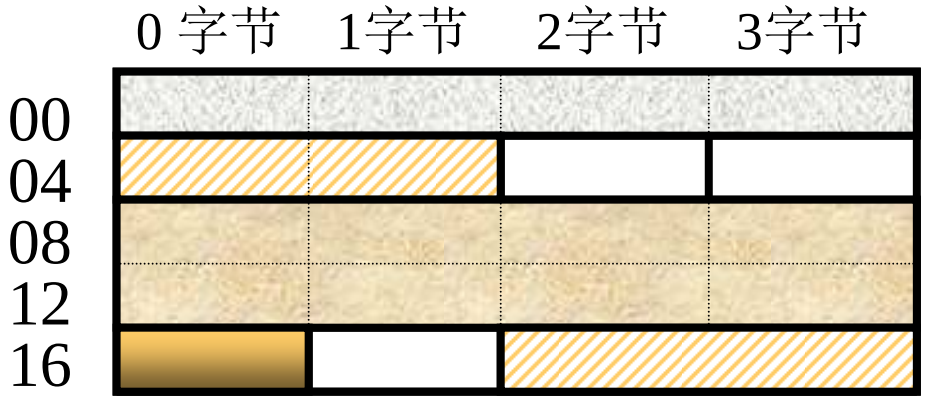
示例　假设数据顺序：字-半字-双字-字节-半字-……

如：int i, short k, double x, char c, short j,……

按边界对齐

x：2个周期

j：1个周期



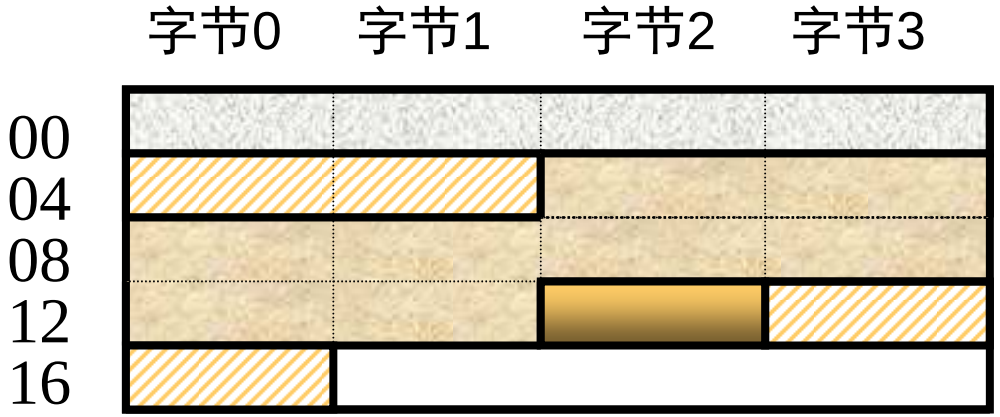
则：&i=0; &k=4; &x=8; &c=16; &j=18;……

边界不对齐

x：3个周期

j：2个周期

增加了访存次数！



则：　&i=0; &k=4; &x=6; &c=14; &j=15;……

数据的检/纠错（Error Detect/Correct）

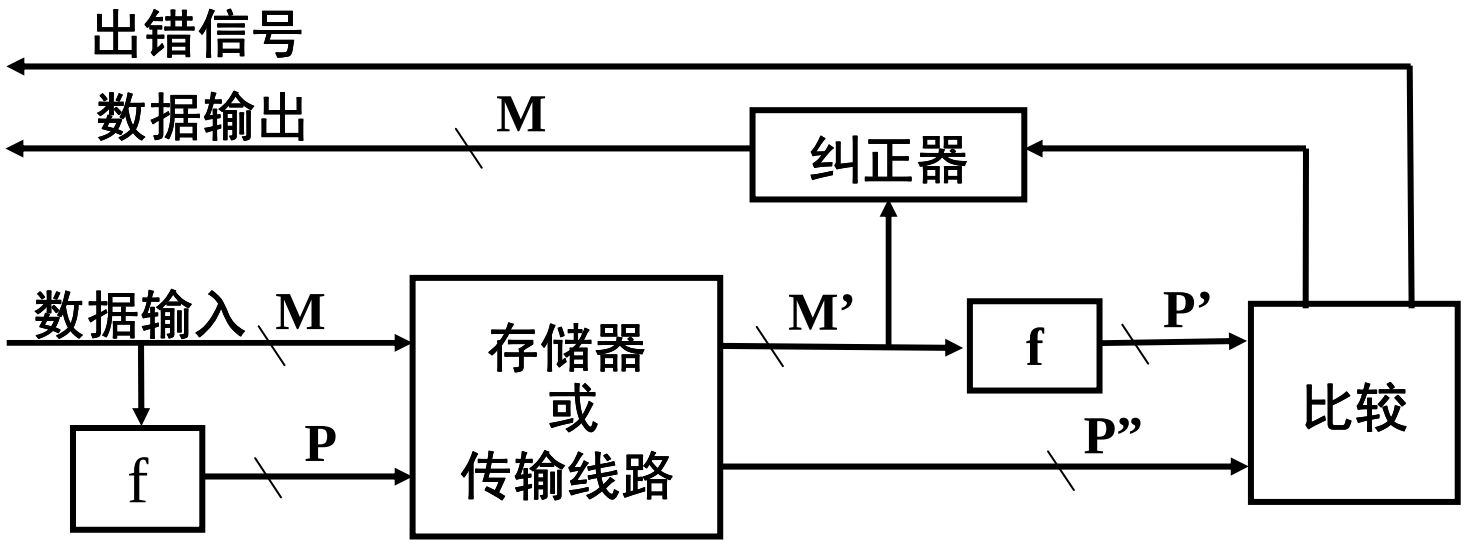
- 为什么要进行数据的错误检测与校正？

  存取和传送时，由于元器件故障或噪音干扰等原因会出现差错。措施：

  (1) 从计算机硬件本身的可靠性入手，在电路、电源、布线等各方面采取必要的措施，提高计算机的抗干扰能力；

  (2) 采取相应的数据检错和校正措施，自动地发现并纠正错误。

- 如何进行错误检测与校正？
  - 大多采用“冗余校验”思想，即除原数据信息外，还增加若干位编码，这些新增的代码被称为校验位。



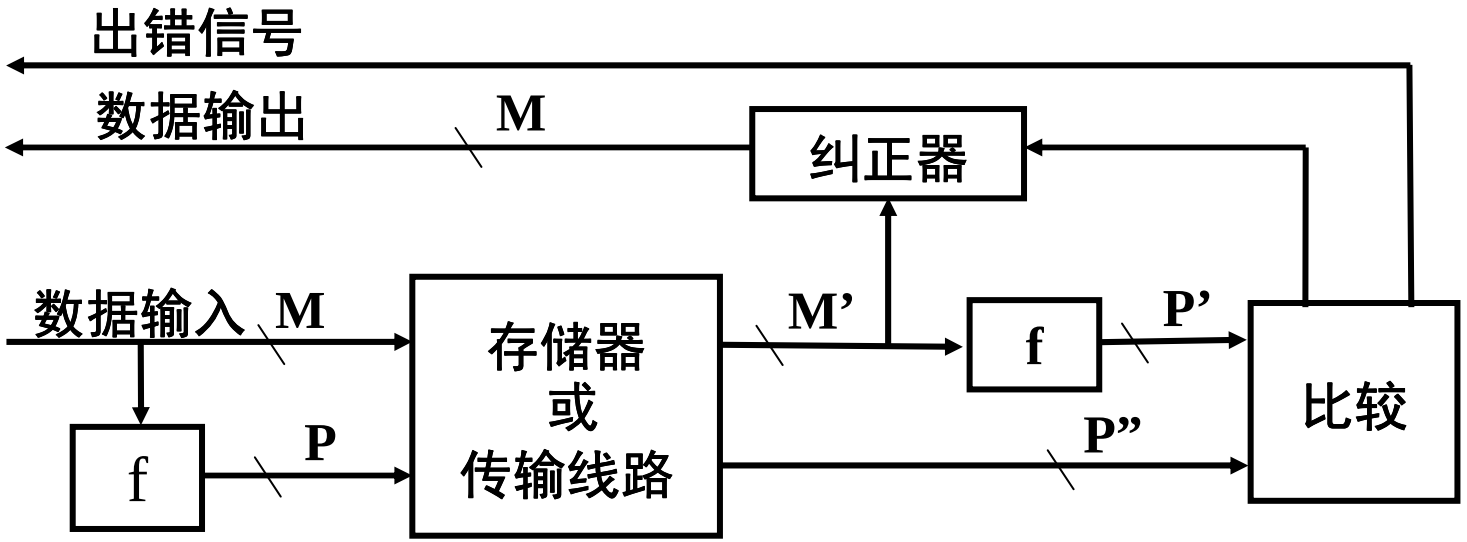
数据的检/纠错

比较的结果为以下三种情况之一：

① 没有检测到错误，得到的数据位直接传送出去。

② 检测到差错，并可以纠错。数据位和比较结果一起送入纠错器，将正确数据位传送出去。

③ 检测到错误，但无法确认哪位出错，因而不能进行纠错处理，此时，报告出错情况。



BACK

码字和码距

- 什么叫码距？
  - 由若干位代码组成的一个字叫“码字”
  - 两个码字中具有不同代码的位的个数叫这两个码字间的“距离”
  - 码制中各码字间最小距离为“码距”，它就是这个码制的距离。

  问题：“8421”码的码距是几？

  2（0010）和3（0011）间距离为1，“8421”码制的码距为1。
- 数据校验中“码字”指数据位和校验位按某种规律排列得到的代码
- 码距与检错、纠错能力的关系（当d≤4）

  ① 若码距d为奇数，则能发现d-1位错，或能纠正(d-1)/2位错。

  ② 若码距d为偶数，则能发现d/2位错，并能纠正(d/2-1)位错。
- 常用的数据校验码有：

  奇偶校验码、海明校验码、循环冗余校验码。

奇偶校验码

基本思想：增加一位奇（偶）校验位并一起存储或传送，根据终部件得到的相应数据和校验位，再求出新校验位，最后根据新校验位确定是否发生了错误。

实现原理：假设数据$B=b_{n-1}b_{n-2}...b_1b_0$从源部件传送至终部件。在终部件接收到的数据为$B'=b_{n-1}'b_{n-2}'...b_1'b_0'$。

第一步：在源部件求出奇（偶）校验位P。

若采用奇校验，则$P=b_{n-1}\oplus b_{n-2}\oplus...\oplus b_1\oplus b_0\oplus 1$。

若采用偶校验，则$P=b_{n-1}\oplus b_{n-2}\oplus...\oplus b_1\oplus b_0$。

第二步：在终部件求出奇（偶）校验位P'。

若采用奇校验，则$P'=b_{n-1}'\oplus b_{n-2}'\oplus...\oplus b_1'\oplus b_0'\oplus 1$。

若采用偶校验，则$P'=b_{n-1}'\oplus b_{n-2}'\oplus...\oplus b_1'\oplus b_0'$。

第三步：计算最终的校验位P*，并根据其值判断有无奇偶错。

假定P在终部件接受到的值为P''，则$P^*=P'\oplus P''$

① 若P*=1，则表示终部件接受的数据有奇数位错。

② 若P*=0，则表示终部件接受的数据正确或有偶数个错。

奇偶校验法的特点

- 问题：奇偶校验码的码距是几？为什么？
  - 码距d=2。

    在奇偶校验码中，若两个数中有奇数位不同，则它们相应的校验位就不同；若有偶数位不同，则虽校验位相同，但至少有两位数据位不同。因而任意两个码字之间至少有两位不同。
- 特点
  - 根据码距和纠/检错能力的关系，它只能发现奇数位出错，不能发现偶数位出错，而且也不能确定发生错误的位置，不具有纠错能力。
  - 开销小，适用于校验一字节长的代码，故常被用于存储器读写检查或按字节传输过程中的数据校验

    因为一字节长的代码发生错误时，1位出错的概率较大，两位以上出错则很少，所以可用奇偶校验。