



北京航空航天大学  
BEIHANG UNIVERSITY



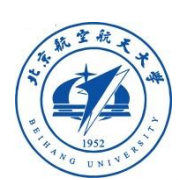
# 微机原理与接口技术

---

林 新

Lx@buaa.edu.cn

北京航空航天大学 自动化学院

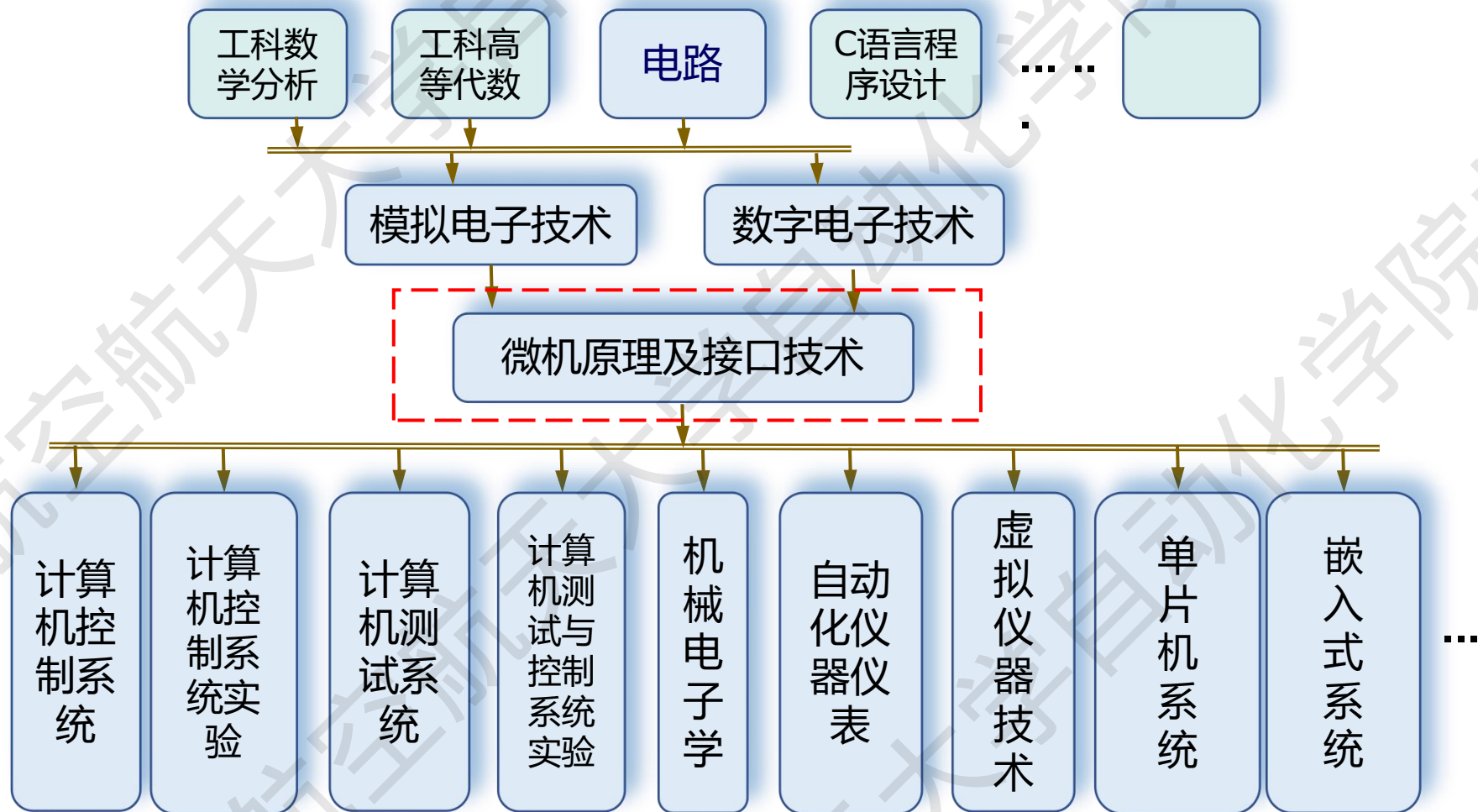


# 自动化专业课程体系（部分）

基础课

专业基础课

专业课





# 课程目标

微机原理是自动控制专业重要的专业基础课之一，通过学习本课程，掌握以下知识：

## ■ 计算机系统基本概念

- 数制、编码、数值与逻辑运算规则
- CPU、微处理器、主机、计算机系统
- 寄存器、存储器、存储器组织结构、总线
- 输入输出接口（IO接口）、端口
- CPU与外界进行数据传输的基本方法：程序法、中断、DMA
- ... ..

这些概念适用于  
所有计算机系统

其它课程（如单片机、  
嵌入式系统等等）

- 部分概念没有涉及
- 相关概念不成体系



# 课程目标

---

微机原理是自动控制专业重要的专业基础课之一，通过学习本课程，掌握以下知识：

- 计算机系统基本概念
- 微机系统的基本组成和运行机制
  - 以8086为例，讲解CPU内部结构、计算机系统架构
  - CPU如何与外围芯片连接
  - 通过学习和使用汇编语言
    - 理解CPU如何与存储器/IO接口芯片交互数据
    - 理解CPU如何精确控制外围芯片



# 课程目标

微机原理是自动控制专业重要的专业基础课之一，通过学习本课程，掌握以下知识：

- 计算机系统基本概念
- 微机系统的基本组成和运行机制
- 典型IO接口芯片原理与编程
  - 定时器芯片8253
  - 并行接口芯片8255A
  - 数模转换芯片DAC0832
  - 模数转化芯片ADC0809
  - 中断控制器8259A

工作原理与现在的  
新型器件类似

也与单片机内部  
的相关模块类似



# 课程目标

微机原理是自动控制专业重要的专业基础课之一，通过学习本课程，掌握以下知识：

- 计算机系统基本概念
- 微机系统的基本组成和运行机制
- 典型IO接口芯片原理与编程
- 使用EDA软件设计计算机应用系统
  - 学习EDA软件的基本操作方法
  - 在EDA软件中配置计算机应用系统所需的芯片、元器件，并进行设置和连接
  - 编写汇编程序，实现特定的控制功能
  - 在EDA软件中运行程序，对系统进行仿真
  - 使用EDA软件提供的调试工具（如数字示波器等），调试系统



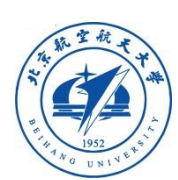
# 课程目标

微机原理是自动控制专业重要的专业基础课之一，通过学习本课程，掌握以下知识：

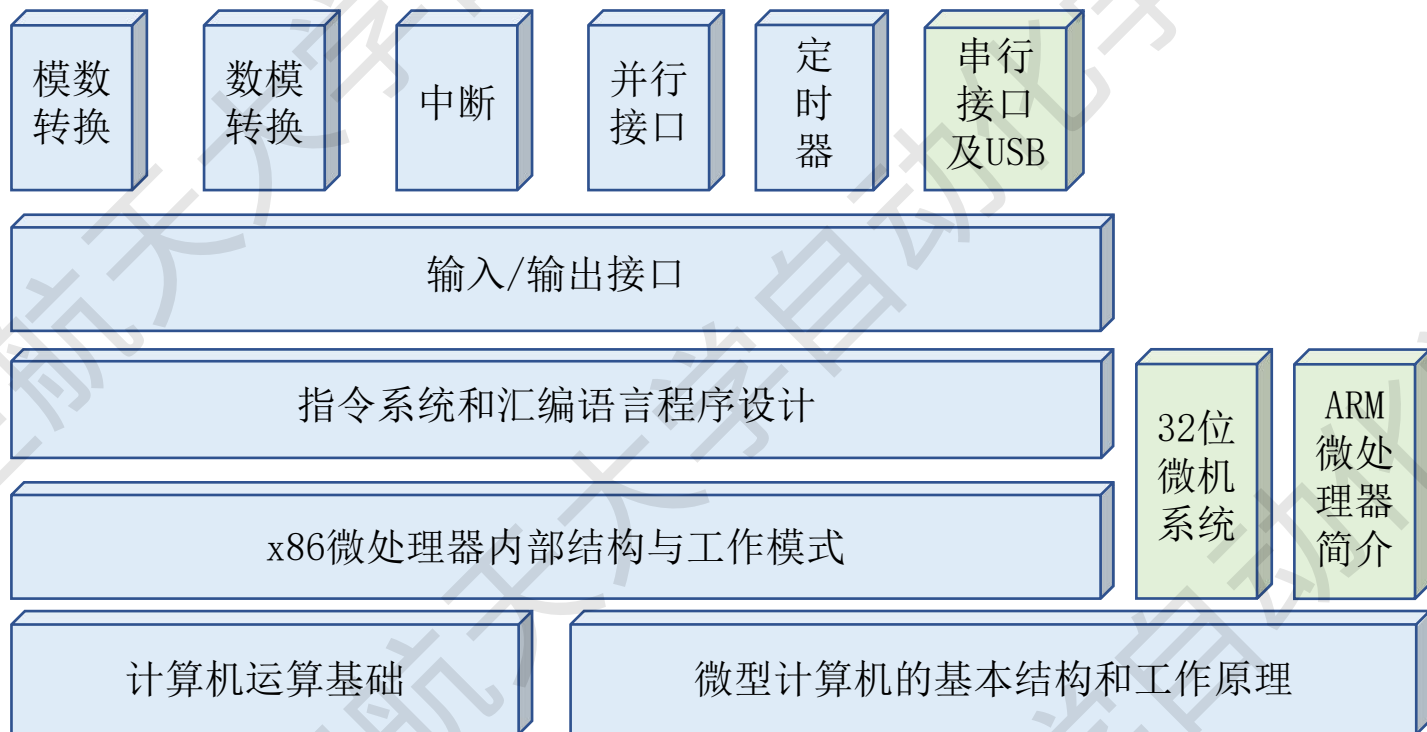
- 计算机系统基本概念
- 微机系统的基本组成和运行机制
- 典型IO接口芯片原理与编程
- 使用EDA软件设计计算机应用系统

本课程完成以下能力的培养：

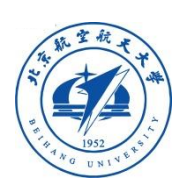
- 进一步学习或自学单片机系统、嵌入式系统、x86微机系统的能力
- 计算机应用系统、计算机控制系统设计能力



# 课程内容



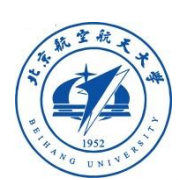




# 参考书

## ■ 课程PPT

- 微型计算机原理与接口技术（第5版），吴宁、乔亚男主编，清华大学出版社
- 微机原理与接口技术-基于8086和Proteus仿真（第三版），顾晖，陈越等主编，电子工业出版社
- 微型计算机原理与接口技术（第5版），周荷琴、吴秀清编著，中国科学技术大学出版社



# 课程要求

---

- 记课堂笔记
- 及时交作业
- 实验课要预先准备好程序
- 有问题及时提问



# 考核考试

---

- 平时40%
  - 出勤、课堂练习、章节作业、大作业、研究报告
- 期末考试 60%
  - 预计第18周之后



# 如何学习

## ■ 课程特点

- 概念多，知识点多



**难点：**理解、掌握和融会贯通众多概念和知识点，形成知识体系

- 偏重工程实践

## ■ 学习方法：多练习、勤实践

- 有疑问一定要**查阅参考书或提问**
- 一定要**自己动手**完成编程、搭建计算机硬件电路等任务



# 学习方法：多练习、勤实践

## 卖油翁

【作者】欧阳修 【朝代】宋

陈康肃公善射，当世无双，公亦以此自矜。尝射于家圃，有卖油翁释担而立，睨之久而不去。见其发矢十中八九，但微颌之。

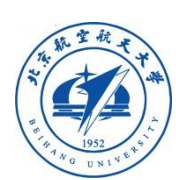
康肃问曰：“汝亦知射乎？吾射不亦精乎？”翁曰：“无他，但手熟尔。”康肃忿然曰：“尔安敢轻吾射！”翁曰：“以我酌油知之。”乃取一葫芦置于地，以钱覆其口，徐以杓酌油沥之，自钱孔入，而钱不湿。因曰：“我亦无他，**惟手熟尔。**”康肃笑而遣之。



# 第一章 绪论

---

1. 数制与信息编码
2. 机器数：数字在计算机中的表示形式
3. 计算机基本结构
4. 微机系统结构
5. 微机系统发展概况

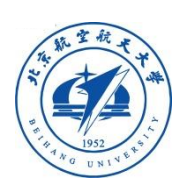


# 1.1 数制与信息编码

- 数制、机器数、二进制数运算法则、定点与浮点数、信息编码

## 1.1.1 进位计数制

1. 2,10,16进制的概念、表达方式
2. 不同数制之间的转换方法
  - 2、16  $\rightarrow$  10进制
  - 10  $\rightarrow$  2、16进制（整数转换、小数转换）
3. 二进制数运算法则：加减乘除、与或非、异或
4. Byte（8位）/Word（16位）无符号数、有符号数的表示范围



## 1.1.1 进位计数制 -- 数制

常用的数制有以下几种：

- 二进制——后缀B

例如1101.101B

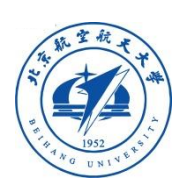
- 十进制——后缀D或者省略

例如231.1123

- 十六进制——后缀H

例如37CF.56H





## 1.1.1 进位计数制 -- 十进制

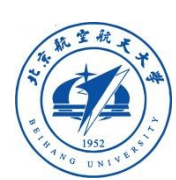
- 十进制的**基数为10**，包含有0、1、2、3、4、5、6、7、8和9十个符号数字。任何十进制数X可以用以下公式来表示：

$$(X)_{10} = X_n X_{n-1} \dots X_1 X_0 X_{-1} \dots X_{-m} = \sum_{-m}^n X_i \times 10^i$$

式中：

$X_i$ ：第*i*位数字

$10^i$ ：第*i*位数码的十进制权



## 1.1.1 进位计数制 -- 二进制

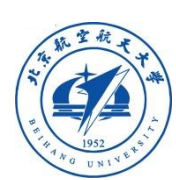
- 二进制数的**基数为2**，包含0,1两个符号数字。  
任何二进制数X可以表示为：

$$(X)_2 = X_n X_{n-1} \dots X_1 X_0 X_{-1} \dots X_{-m} = \sum_{-m}^n X_i \times 2^i$$

式中：

$X_i$ ：第*i*位数字

$2^i$ ：第*i*位数码的二进制权



## 1.1.1 进位计数制 -- 十六进制

- 十六进制数的**基数为16**，包含0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F十六个符号数字。任何十六进制数X可以表示为：

$$\begin{aligned}(X)_{16} &= X_n X_{n-1} \dots X_1 X_0 X_{-1} \dots X_{-m} \\ &= \sum_{-m}^n X_i \times 16^i\end{aligned}$$

式中：

$X_i$ ：第*i*位数字

$16^i$ ：第*i*位数码的十六进制权



## 1.1.1 进位计数制-- 不同数制之间的转换

---

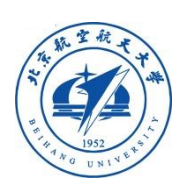
- 其它数制转换为十进制
- 十进制转换为其它数制
- 二进制编码的十六进制



## 1.1.1 进位计数制-- 不同数制之间的转换

### ■ 其它数制**转换为十进制**

- 任何基数的数制转换为十进制时，按相应的**权表达式展开**，即该数每位上的数字与其对应的权值的乘积之和，就是其数制对应的十进制。



## 1.1.1 进位计数制-- 不同数制之间的转换

### ■ 其它数制**转换为十进制**

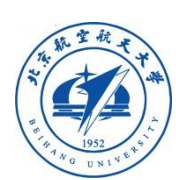
- 任何基数的数制转换为十进制时，按相应的**权表达式展开**，即该数每位上的数字与其对应的权值的乘积之和，就是其数制对应的十进制。  
转换十进制**整数**部分的算法：

### ■ 十进制转 十进制 计算。 时需要

1. 用其它数制的基数**除**十进制整数部分
2. 保存余数（最先得到的余数为最低有效位）
3. 重复1)和2)，直到商为0。

转换十进制**小数**部分的算法

1. 用其它数制的基数**乘**十进制小数部分
2. 保存结果的整数部分（最先得到的整数结果为最高位小数）
3. 重复1)和2)，直到小数部分为0。



## 1.1.1 进位计数制-- 不同数制之间的转换

### ■ 其它数制**转换为十进制**

- 任何基数的数制转换为十进制时，按相应的**权表达式展开**，即该数每位上的数字与其对应的权值的乘积之和，就是其数制对应的十进制。

### ■ 十进制**转换为其它数制**

- 十进制转换为其它数制时，需要**分为整数和小数两部分**计算。转换整数部分时需

例如：

0011 1010 1100 0110 B

=> 3AC6H

### ■ 二进制编码的**十六进制**

- 每位十六进制数分别对应4位二进制数码

二进制编码的十六进制数对应关系表

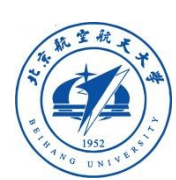
十进制	二进制	16进制		十进制	二进制	16进制
0	0000	0		8	1000	8
1	0001	1		9	1001	9
2	0010	2		10	1010	A
3	0011	3		11	1011	B
4	0100	4		12	1100	C
5	0101	5		13	1101	D
6	0110	6		14	1110	E
7	0111	7		15	1111	F





## ■ 二进制编码的十六进制

- 二进制→十六进制：从小数点开始向左（即整数部分）每4位一组进行转换，最左侧不足4位则补0；小数点右侧（即小数部分）也是每4位一组进行转换，最右侧不足4位则补0。
- 十六进制→二进制：每个十六进制数用4位二进制数代替



## 1.1.1 进位计数制-- 二进制运算基本规则

- 数学运算：加、减、乘、除
- 逻辑运算：与、或、非、异或

\*注：

1. 关于数学运算：

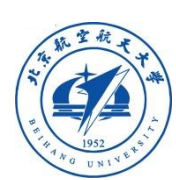
1) 乘法可转换为左移和加法；

2) 除法可转换为右移和减法；

3) 减法可转换为加法：减数转换为补码，与被减数相加

因此，计算机中可使用加法器配合移位等功能实现二进制数的四则运算。

2. 计算机中的二进制运算受**字长**限制



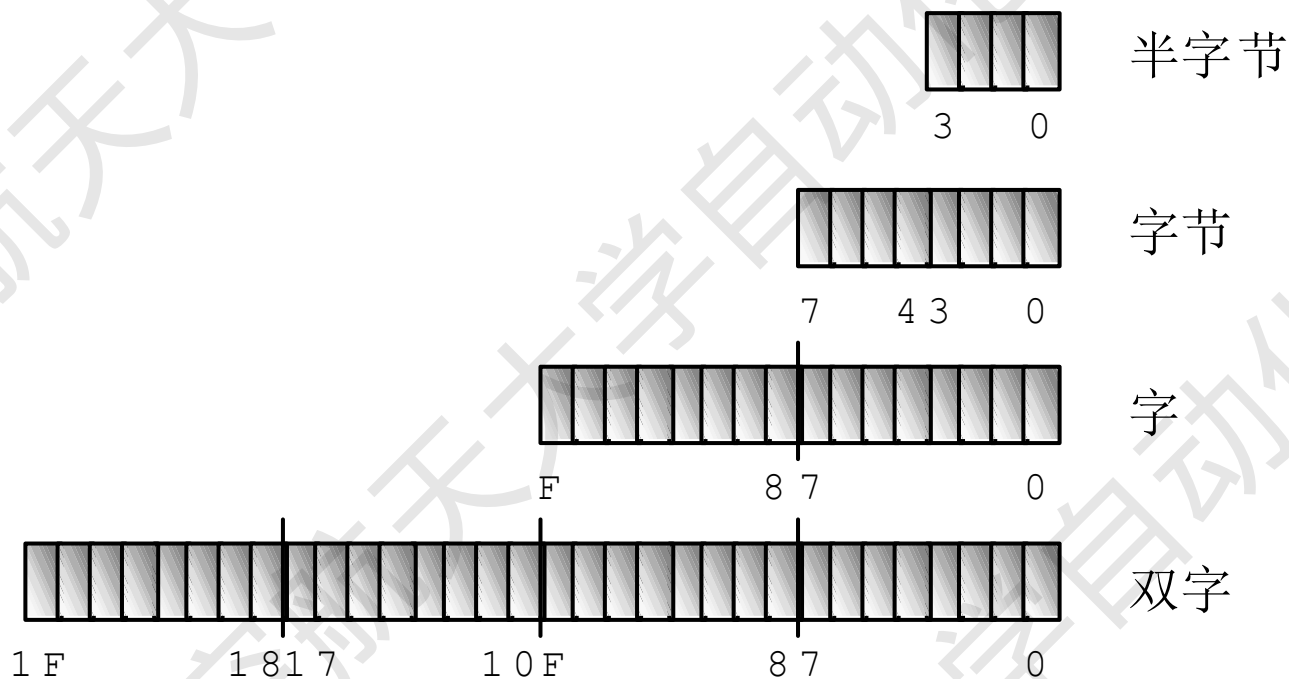
## 1.1.1 进位计数制 -- 基本概念

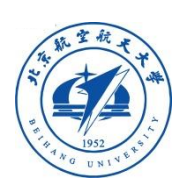
### 位、字节、字和字长

- **位 (bit)** 是计算机所能表示的最小最基本的数据单位, 它指的是取值只能为0或1的一个二进制数值位。
- **字节 (byte)** 由**8个二进制位**组成, 通常用作计算存储容量的单位。通常字节作为单位时记作B,  $1 \text{ Byte} = 8 \text{ bit}$
- **字 (word)** 由**16个二进制位**组成, 相当于两个字节, 其中D15~D8为高字节, D7~D0为低字节。作为单位时记作W,  $1 \text{ Word} = 2 \text{ byte} = 16 \text{ bit}$



# 1.1.1 进位计数制 -- 基本概念





## 1.1.1 进位计数制 -- 基本概念

- **字长**是微处理器一次可以**直接处理**的二进制数码的位数。
  - 字长通常取决于微处理器**内部通用寄存器的位数**和**数据总线的宽度**。字长越长运算精度越高。微处理器的字长有4位、8位、16位、32位和64位等等。
  - 8051是8位微处理器，8086是16位微处理器，intel 酷睿 i5/i7是64位微处理器

\*计算机中的二进制运算受**字长**限制



## 1.1.2 二进制编码

---

二进制编码：在计算机中使用二进制数来表示十进制数字、英文字母、运算符号、中文文字等等。本课程主要用到以下两种：

- BCD码
- ASCII码



## 1.2.2 二进制编码 -- BCD码

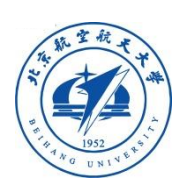
将十进制的每一位数用4位二进制编码表示，即：十进制的0 ~ 9分别用BCD数的0000到1001表示，而不是将整个十进制数转换为二进制形式。分为两种：

- 压缩BCD码（每字节表示2个数字）
- 非压缩BCD码（每字节表示1个数字）

例：

十进制	12	623
压缩BCD	0001 0010	0000 0110 0010 0011
非压缩BCD	00000001 00000010	00000110 00000010 00000011
二进制	1100	0010 0110 1111

\*注：又称为8421BCD码，8421代表了4位二进制数每个位的权值



## 1.2.2 二进制编码 -- ASCII码

**ASCII** (American Standard Code for Information Interchange) 码，即美国标准信息交换码，是使用8位的二进制数对字符进行编码，其中最高位为0，余下的7位可以用来定义**128个字符**

- 有**95个字符**对应着计算机终端（键盘）能够输入、并且能够显示和打印的字符
- 还有**33个字符**，编码0~31和127，对应的是控制码，用于控制计算机外设工作特性或计算机软件的运行，如**LF**、**CR**
- **数字0~9**的编码0110000B~01101001B，高3位均是011，后4位正好与其对应的二进制代码相符（即**30H~39H**）。
- 英文**字母A~Z**的ASCII码从1000001 (**41H**) 开始顺序递增，**字母a~z**的ASCII码从1100001 (**61H**) 开始顺序递增，这样的排列对信息检索十分有利。

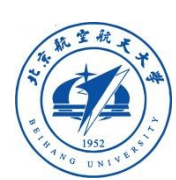


## 1.2.2 二进制编码 -- ASCII码

ASCII 字符表

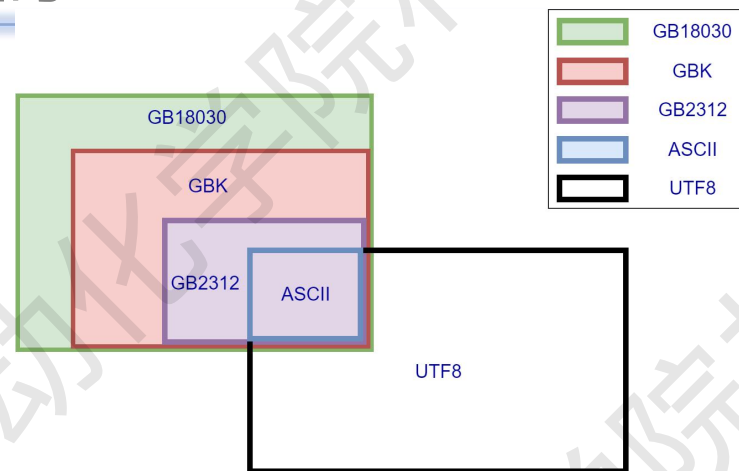
$\begin{array}{c} \text{H} \\ \diagdown \\ \text{L} \end{array}$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENG	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	~
1111	SI	US	/	?	O	←	o	DEL

注：H 表示高 3 位，L 表示低 4 位。



## 1.2.2 二进制编码 -- 中文编码

- GB2312简体中文编码，包含了生活中常用的所有汉字。每个汉字或符号占用2个字节，共有6763个汉字以及682个特殊符号



- GBK编码。是对GB2312的扩展，加入了繁体字的支持。每个字2字节，共有20902个汉字，另有984个汉语标点符号、部首等。注意这里的繁体字编码和台湾BIG5不同。
- GB18030编码。对GBK再次扩展，支持更多的繁体字，并编入了少数民族文字、朝鲜文、日文等，共有7万多文字符号编码。每个文字符号占用1、2、4字节不等。
- Unicode及UTF8（略）

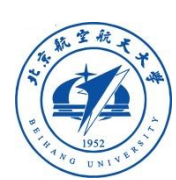
例：  
汉字“中”的 **GB18030** 编码：**D6D0H**；  
汉字“国”的编码：**B9FAH**



## 1.1.3 机器数

日常所用到的数大多数为**有符号数**，也就是所说的正数、负数和零。数的符号在计算机中也用二进制数表示，通常用二进制数的**最高位**表示数的符号。

- 一个数及其符号在机器中的表示形式称为**机器数**，数及其符号本身称为**真值**。
- 常用的机器数有三种：原码、补码、反码。
- 在计算机系统中，机器数(**有符号数**)默认为**补码**。



## 1.1.3 机器数 -- 二进制有符号数

### ■ 原码

- 通常用最高位表示符号：对于正数，该位取0，对于负数，该位取1。而数值部分保持数的原有形式

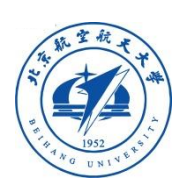
### ■ 反码

- 对于正数，其反码与原码相同，也与补码相同；
- 对于负数，其反码等于对应的正数按位（包括符号位）求反。

### ■ 补码

- 正数的补码与其原码、反码相同；
- 负数的补码等于它的反码加1，即：负数的补码等于对应的正数按位求反（包括符号位）再加1

\*计算机内部的定点数字运算都是补码



## 1.1.3 机器数 -- 补码运算

### ① 十进制有符号数与补码的相互转换

- 正数

直接使用二进制 $\longleftrightarrow$ 十进制转换方法

- 负数

需要先求出对应的正数，然后再转换，最后再求负

$$[X]_{\text{补}} \xrightarrow{\text{按位求反加1}} [-X]_{\text{补}} \xrightarrow{\text{按位求反加1}} [X]_{\text{补}}$$

### ② 补码的算术运算规则

- 补码的加法规则

$$[X + Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}}$$

- 补码的减法规则

$$[X - Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$$



## 1.1.3 机器数 -- 补码运算

### ③ 位宽、表示范围和溢出

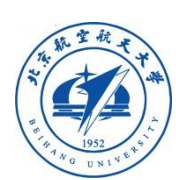
- 由于二进制数字的位数（位宽）的限制，所表达的大小也被限制在一定范围之内

表 二进制数的表达范围

	无符号数	补码
字节(8位)	0~255	-128~127
字(16位)	0~65535	-32768~32767

- 当补码运算的结果超出该数据类型所表示的范围时，机器就不能正确表示，这时会产生溢出。
- 在溢出的情况下，机器计算的结果是错误的。

\*注：无符号数运算结果超出范围时，不称为溢出。原因是这种情况下有专用的硬件标志位保存所产生的进位/借位



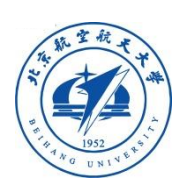
## 1.1.3 机器数 -- 补码运算

$$[X]_{\text{补}} = [-65D]_{\text{补}} = 1011\ 1111$$

$$[Y]_{\text{补}} = [-96D]_{\text{补}} = 1010\ 0000$$

$$\text{则 } [X]_{\text{补}} + [Y]_{\text{补}} = 0101\ 1111$$

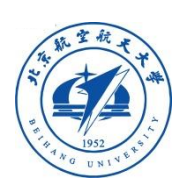
符号位为0，结果为一个正数，显然是错的。其原因是这两个数的和是-161，已经超出了8位数据补码的表示范围，出现了溢出。



## 判断补码运算是否溢出的方法

- ① 将补码转换为十进制数进行运算，查看运算结果是否超出补码位宽的表示范围
- ② 设最高位 $D_n$ 产生的进位为 $C_n$ ，次高位 $D_{n-1}$ 产生的进位为 $C_{n-1}$ 。若 $C_n \oplus C_{n-1} = 1$ ，则结果溢出
- ③ 根据补码运算结果直接判断为溢出：
  - 正数+正数（或正数-负数）结果为负数
  - 负数+负数（或负数-正数）结果为正数





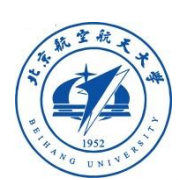
## 1.1.4 定点数与浮点数

常用的数据一般有三种：

- 纯整数，如二进制的1011
- 纯小数，如二进制的0.1101
- 既有小数又有整数，如二进制的10.0011

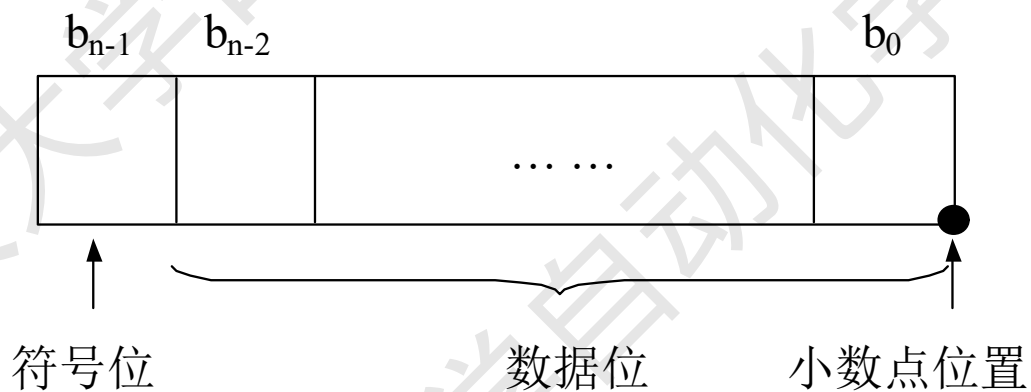
在计算机中，表示这三种数据的方法有两种：定点表示法和浮点表示法。

- 计算机中数的小数点位置固定的表示法称为定点表示法，用定点表示法表示的数据称为**定点数**；
- 计算机中数的小数点位置不固定的表示法称为浮点表示法，用浮点表示法表示的数据称为**浮点数**。



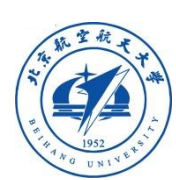
## 1.1.4 定点数与浮点数

### ① 定点纯整数



### ② 定点纯小数





## 1.1.4 定点数与浮点数

### ③浮点数

根据IEEE754[1985]标准，规格化的**单精度浮点数**（32位，4字节）的数学公式定义如下：

$$N = (-1)^S 2^{E-127} (1.M)$$

其中：

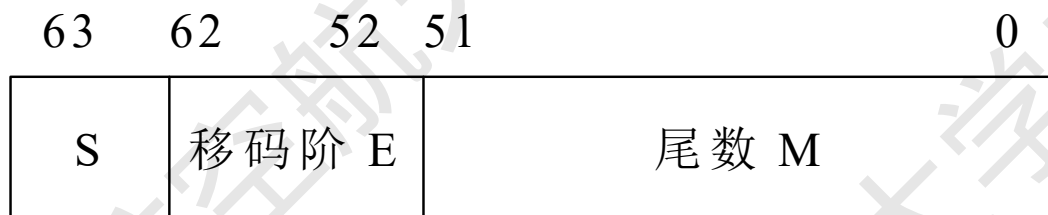
- **S**是浮点数的符号（1位）
- **E**是浮点数的移码阶（8位）， $E \in (0, 255)$
- “**127**” 是阶码的偏移量
- **M**是**规格化**后的尾数（23位）



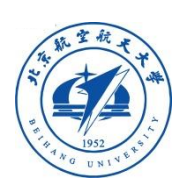
a) 浮点数



b) 单精度浮点数       $N = (-1)^S 2^{E-127} (1.M)$



c) 双精度浮点数       $N = (-1)^S 2^{E-1023} (1.M)$



## 1.1.4 定点数与浮点数

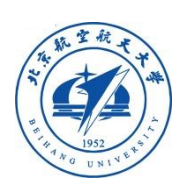
- 所谓规格化，是指通过对阶码的调整，使得尾数处于 $[1,2]$ 之间。

例如： $1100B = 1.1 \times 2^3$

$$0.01101B = 1.101 \times 2^{-2}$$

注意：

1. 规格化的浮点数中尾数前面的1，在存储器中存储时是隐含的，没有实际存储。实际存储的只是小数点后面的数值。
2. 不同计算机系统、不同的数值处理，“规格化”的含义是不同的。前面公式中的规格化仅仅是IEEE754中的规定。



## 1.1.4 定点数与浮点数

---

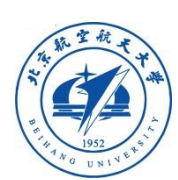
- 当总位数不变时，阶码的位数越多，表示的数值范围就越大，同时由于尾数的位数减少，精度降低。



## 1.2 计算机基本结构

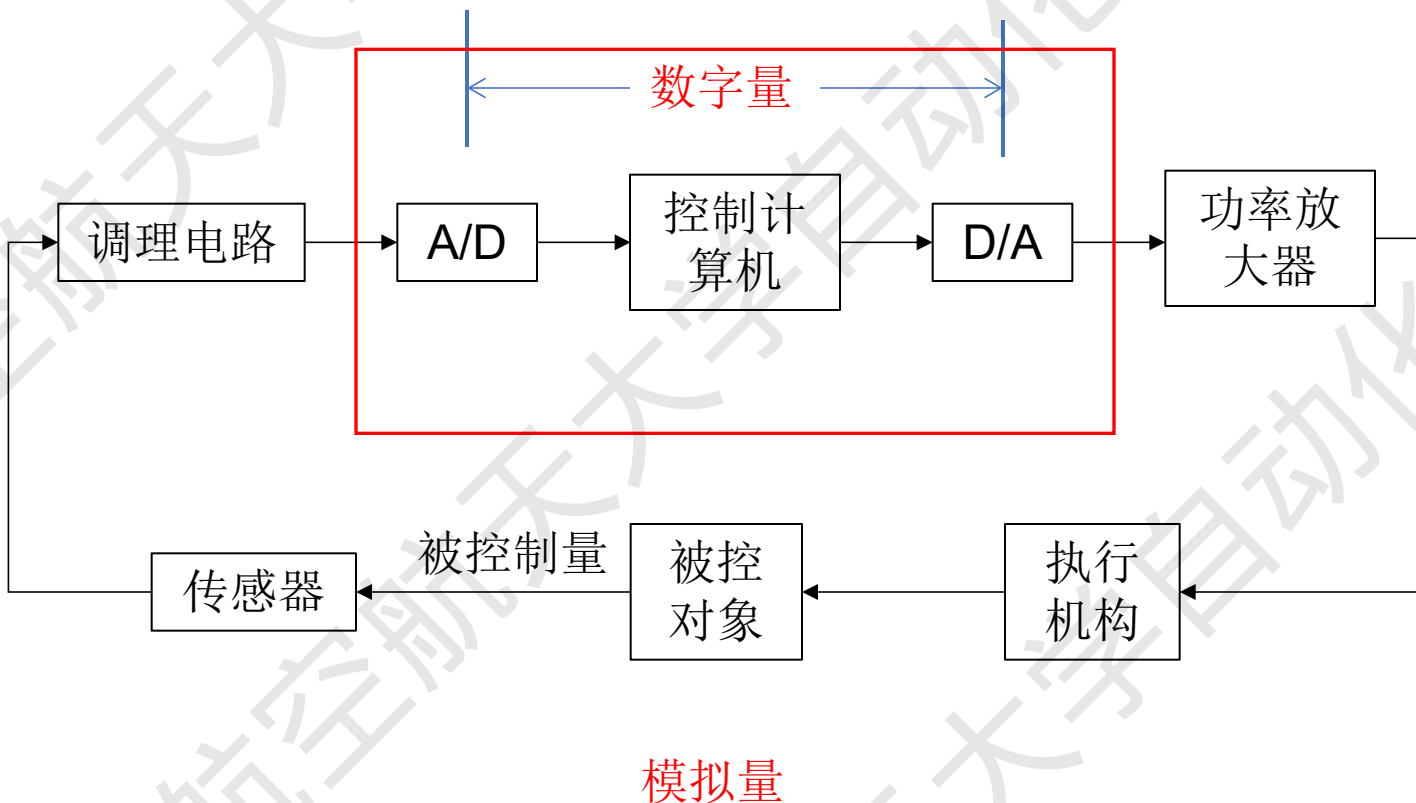
---

- 计算机硬件
  - 结构与组成：冯·诺依曼结构和哈佛结构
  - 工作过程
- 计算机软件
  - 指令和程序
  - 机器码
  - 汇编语言
  - 高级编程语言
  - 操作系统



## 1.2 计算机基本结构

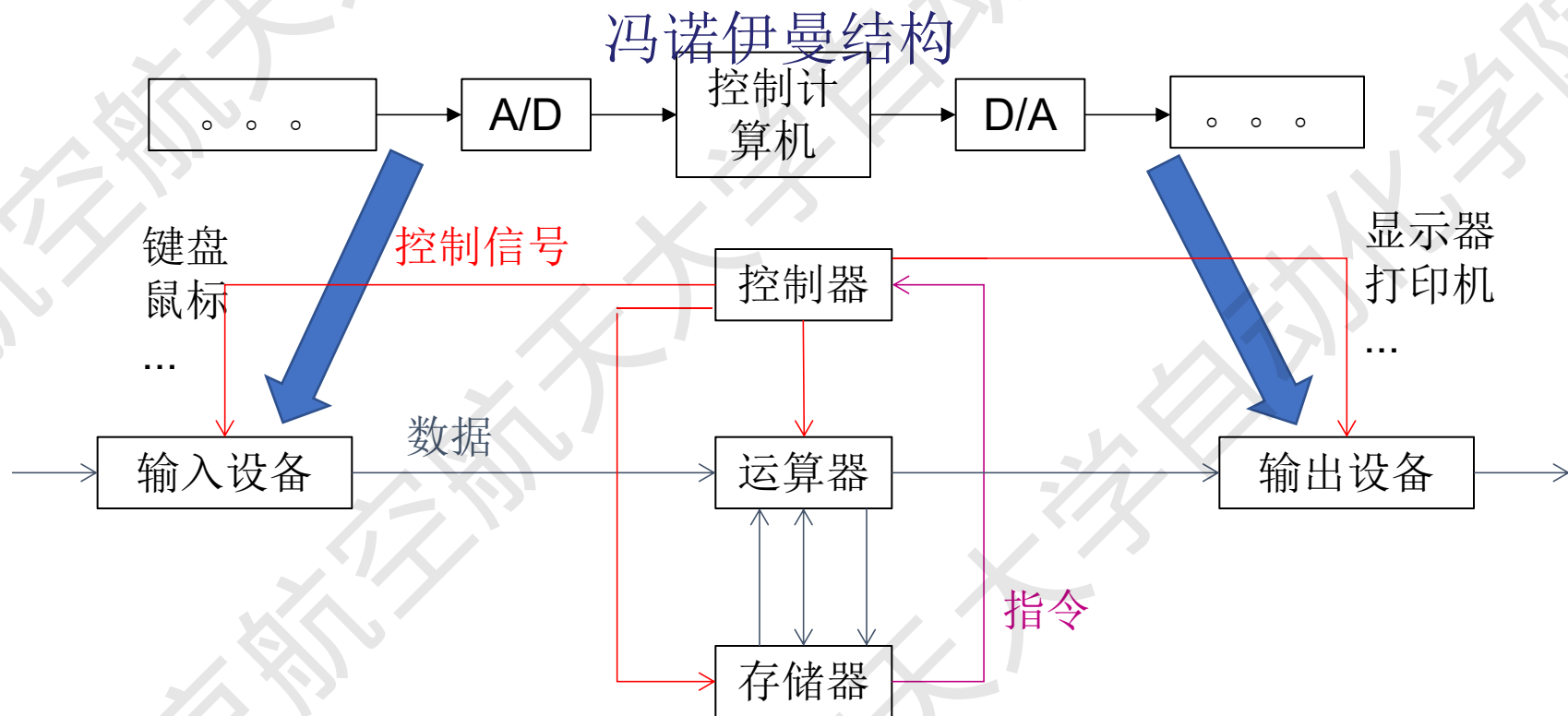
### 典型的计算机控制系统

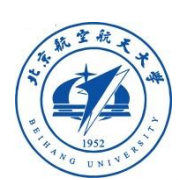






## 1.2 计算机基本结构





## 1.2.1 冯诺伊曼结构

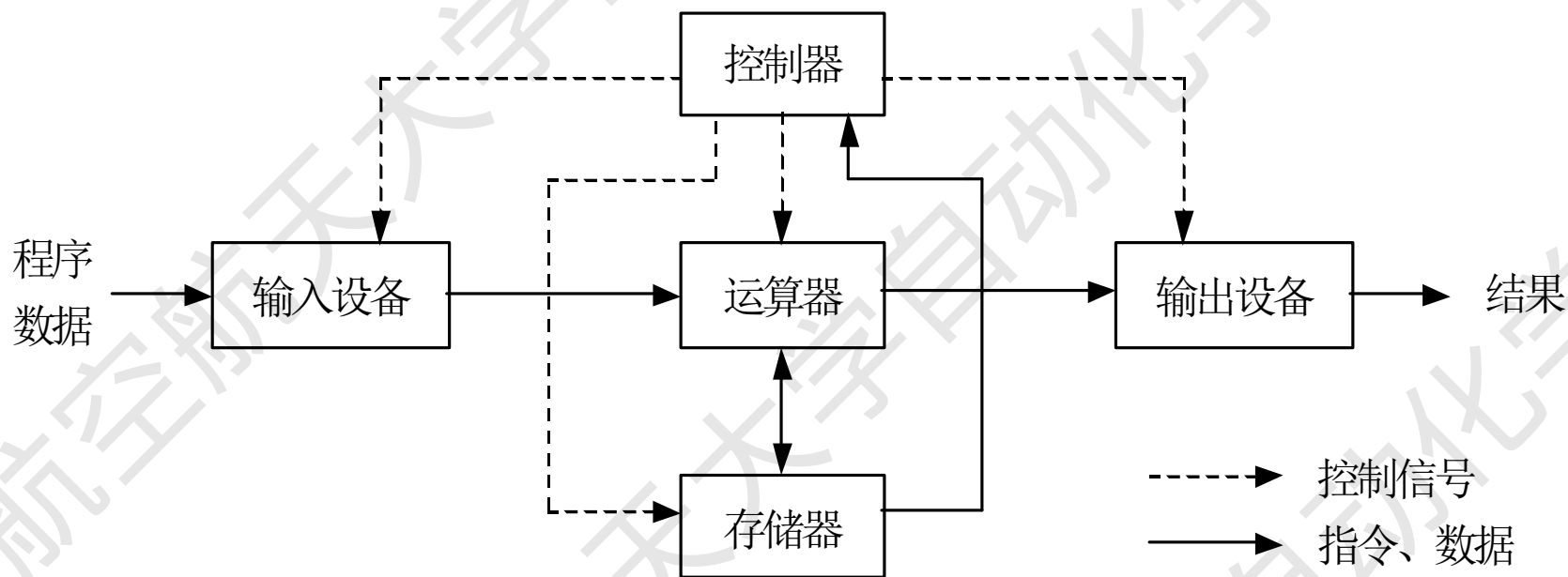


图 冯诺伊曼结构



## 1.2.1 冯诺伊曼结构

- **运算器**：用来执行加法运算、逻辑运算和移位等操作
- **存储器**：以二进制方式存放程序、数据、中间结果
- **控制器**：根据程序中的指令控制各个部件协调工作
- **输入设备**：输入原始数据和程序
- **输出设备**：将运算结果输出到其它设备

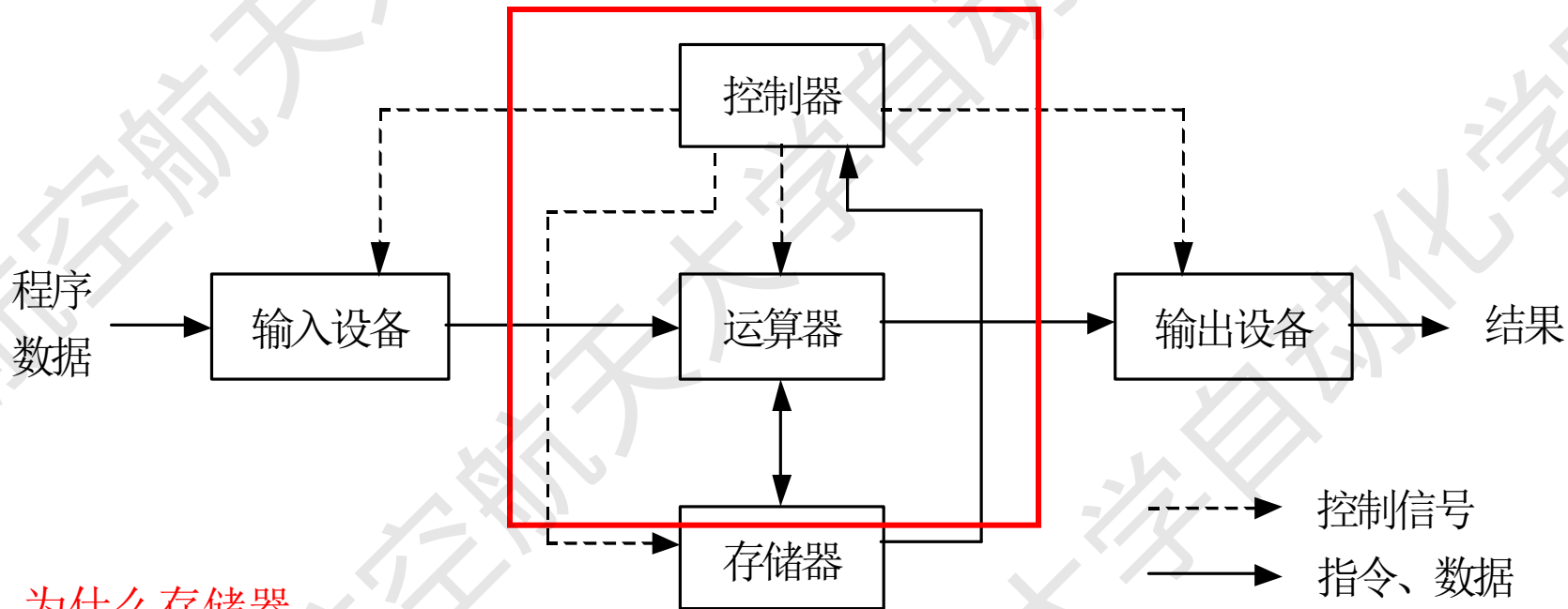


## 1.2.2 微型计算机结构

CPU在哪里？

什么是CPU？

CPU: Central Processing Unit, 用于执行指令操作, 进行算术和逻辑运算, 完成信息处理、控制程序执行等功能



为什么存储器只划入一部分？

运行速度与制造成本的折中

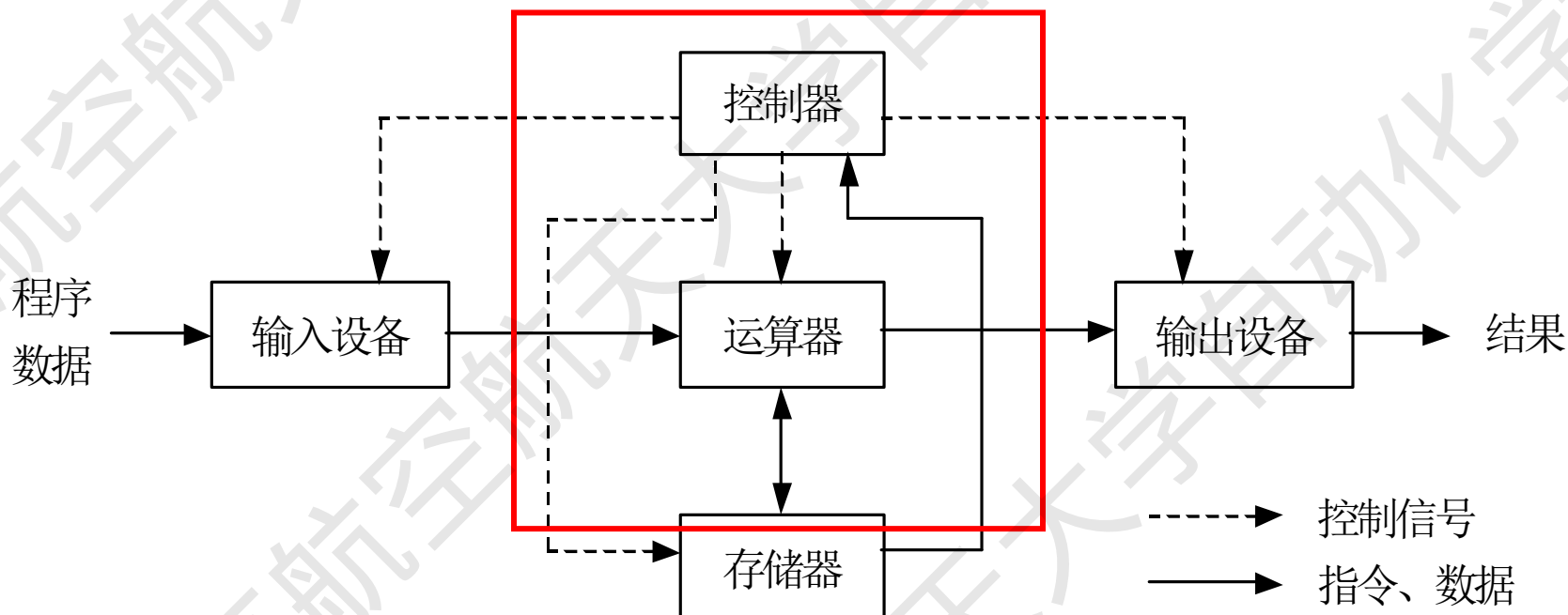


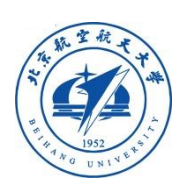
CPU是计算机系统的核心，主要包括：

- 运算器：也称为算术逻辑单元ALU
- 控制器：一般由指令寄存器、指令译码器、以及控制电路组成
- 寄存器：用来存放经常使用的数据。

CPU内部的寄存器，速度快、数量少；

CPU外部的存储器，一般称为内存、主存储器，速度一般比CPU内部的寄存器慢，用于存储大容量的程序和数据。





## 1.2.2 微型计算机结构 -- 微处理器/CPU

**微处理器**和**CPU**的概念略有不同，但是现在通常将这两者混称。

- **CPU**指的是计算机中执行运算和控制功能的区域，由算术逻辑部件(ALU)、控制部件、以及一组寄存器组成。
- 早期的CPU都是由**分立元件**搭建而成，当集成电路出现后，CPU被**集成在一片芯片**上，称之为**微处理器**。
- 现在的**微处理器**除了含有CPU外，有时还集成有FLASH、RAM/ROM、输入/输出接口电路等模块。



## 程序执行过程?

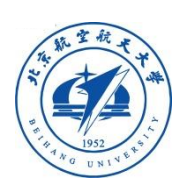
## CPU如何取指令?

每个存储单元的  
位宽为一个字节



## 什么是总线？

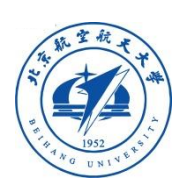
## -- 总线是数据传输的公共通道



## 1.2.2 微型计算机结构--基本概念

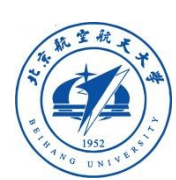
- **操作**：CPU读写存储器、进行逻辑运算、输入输出等等；
- **指令**：以命令的形式记录计算机所执行的操作称为指令；
  - 指令集：计算机所有指令的集合
  - 指令和数据在计算机内部以二进制数方式存放
- **程序**：将算法或任务分解为不同步骤，每个步骤由若干指令序列组成，以便按照一定的顺序控制计算机完成工作。





## 1.2.2 微型计算机结构--基本概念

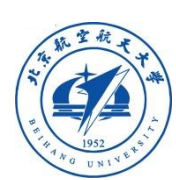
- **机器码**：由二进制数表示的计算机指令；
  - 机器码通常由**操作码**和**操作数**两部分组成。
- **汇编语言**：代替二进制机器码的助记符，以便于记忆。
  - 用指令功能的英文缩写替代操作码
  - 符号或寄存器来替代或存放地址
  - 汇编语言与CPU相对应。
- **汇编程序 (Assembler)**：将汇编语言编写的程序转换为二进制机器码的软件



## 1.2.2 微型计算机结构--基本概念

### ■ 高级编程语言

- 与具体的计算机系统、CPU类型无关或者相关性很小，程序便于移植；
- 易于理解、学习和掌握
- C/C++, BASIC, FORTRAN, JAVA等等



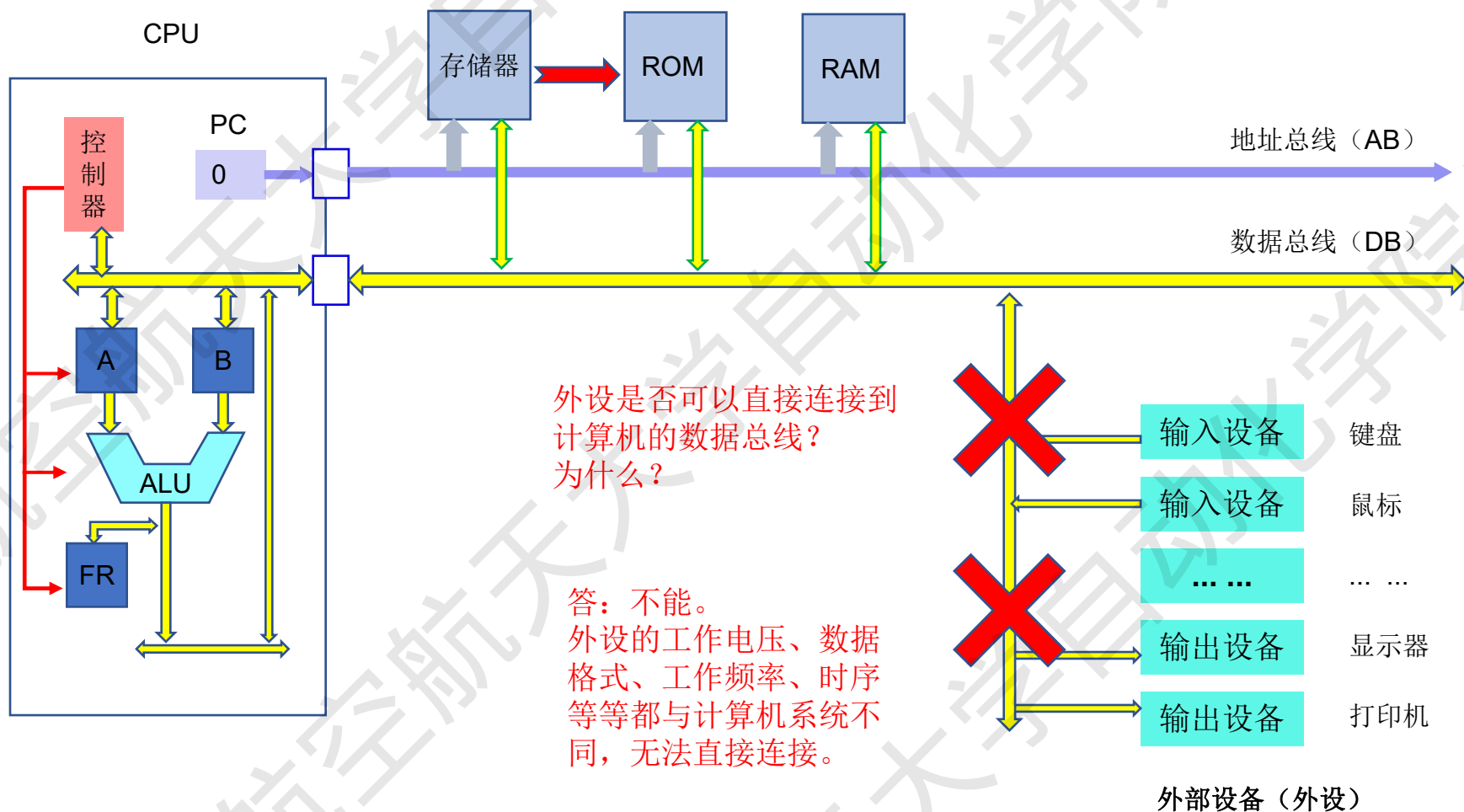
## 1.2.2 微型计算机结构--基本概念

- **存储器**：用于存放指令和数据。
  - 通常划分为一个个的**存储单元**，每个单元可存放**1字节**（8bit）的二进制信息。
  - 对每个存储单元进行编号，以便相互区分。此编号称为存储单元的**地址**。
  - 存储器的**容量**：以**字节（Byte）**为单位计量存储器中单元的数目，如B，KB，MB，GB，TB
  - 存储单元中存放的信息称为**存储单元的内容**，记作：  
(地址) = 内容，如：(0003H) = 3CH

\*书中有时省略16进制中的H，但是在书写程序时不能省略



## 1.2.2 微型计算机结构--外部设备





# 1.2.2 微型计算机结构

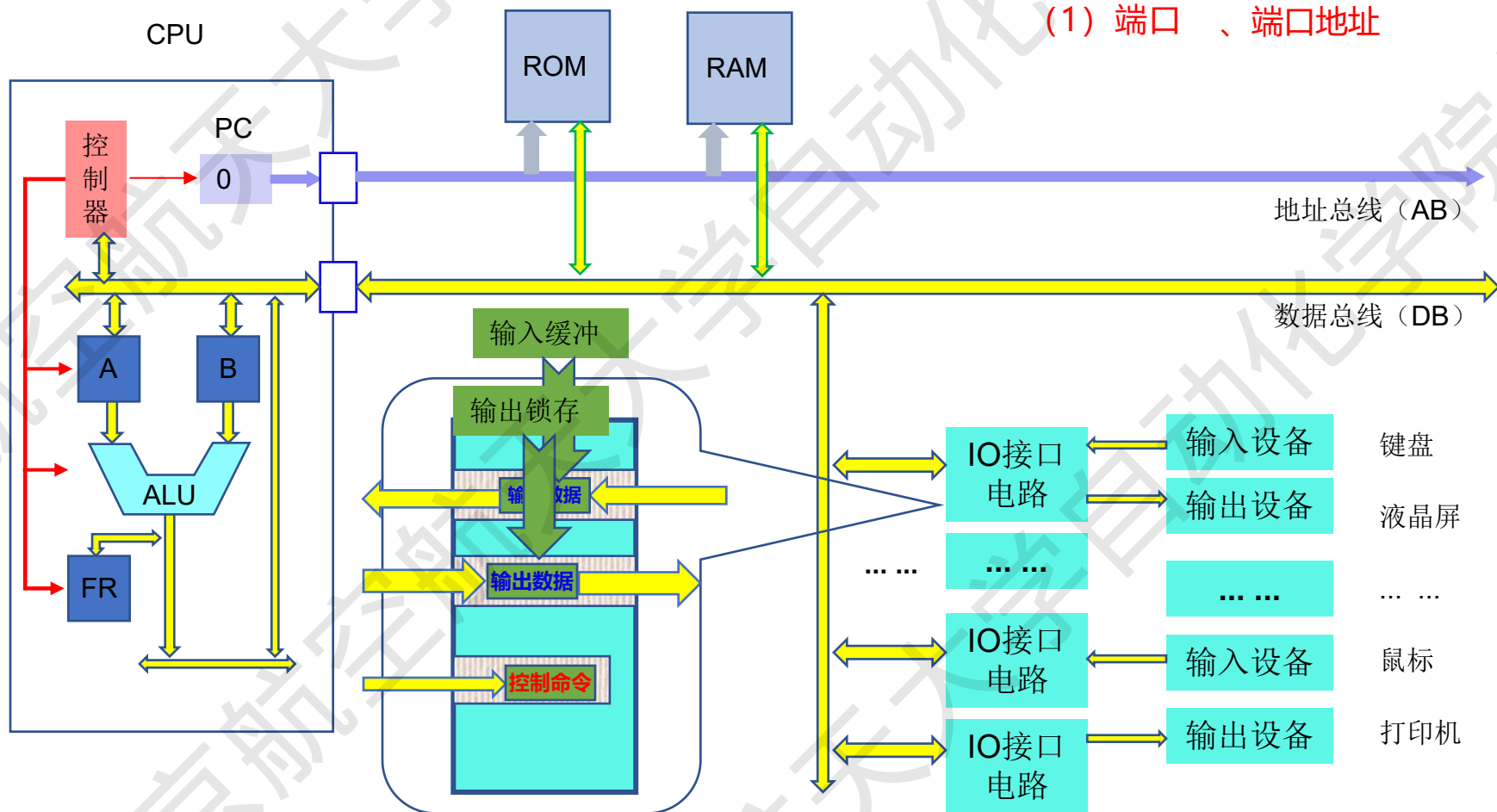
## -- 端口

IO接口电路的作用:

- ✓ 输入缓冲、输出锁存;
- ✓ 电平转换;
- ✓ 时序匹配;
- ✓ 数据格式转换;
- ... ..

IO接口电路如何实现CPU与外设之间的数据传输?

(1) 端口 、端口地址



端口：数据传输通道，起到输入缓冲、输出锁存的功能



# 1.2.2 微型计算机结构

## -- 端口地址

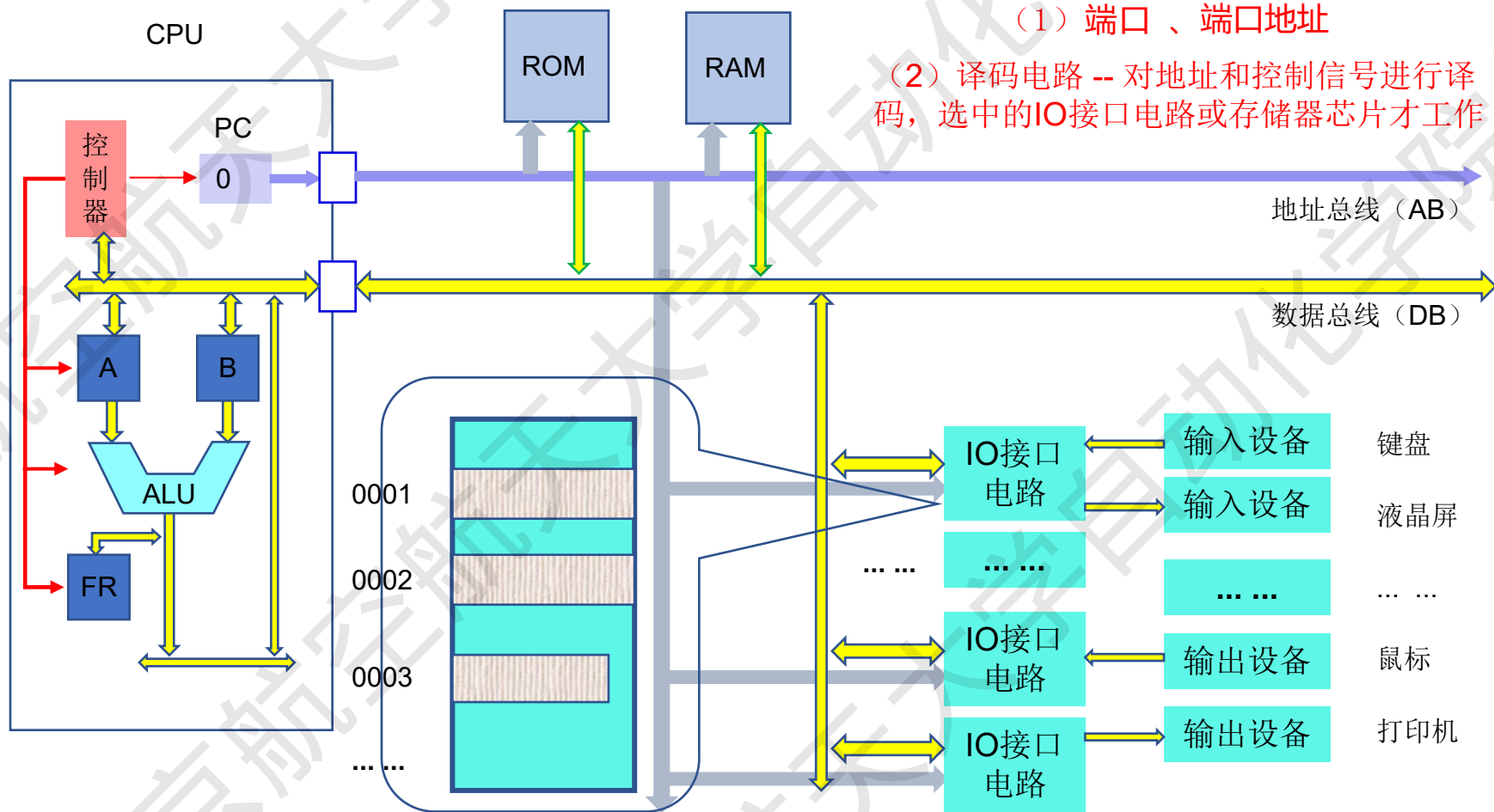
- IO接口电路的作用：
- ✓ 输入缓冲、输出锁存；
  - ✓ 电平转换；
  - ✓ 时序匹配；
  - ✓ 数据格式转换；

... ..

IO接口电路如何实现CPU与外设之间的数据传输？

(1) 端口、端口地址

(2) 译码电路 -- 对地址和控制信号进行译码，选中的IO接口电路或存储器芯片才工作





# 1.2.2 微型计算机结构

## -- 读写控制

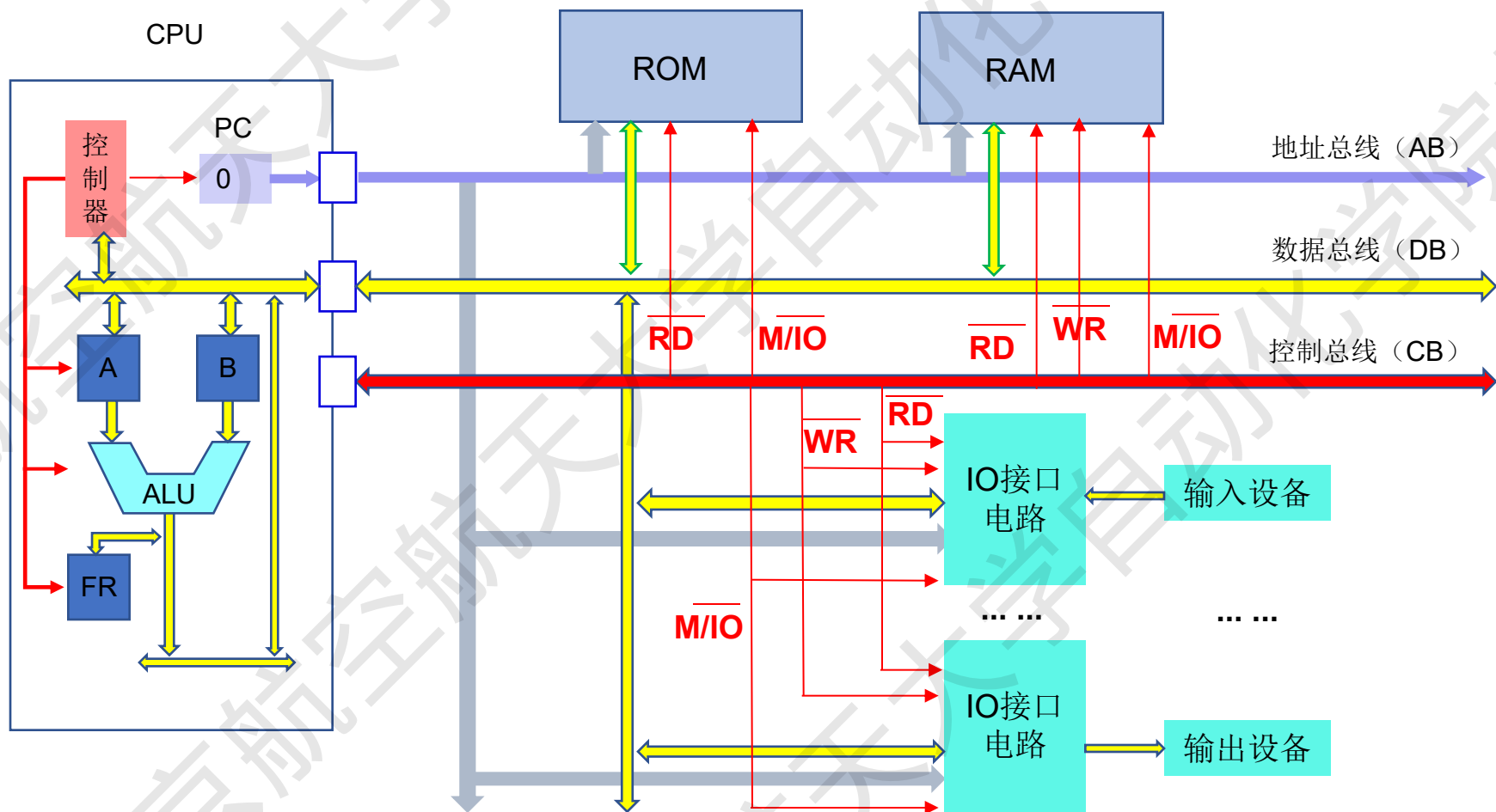
如何区分操作是针对存储器的还是IO接口的？

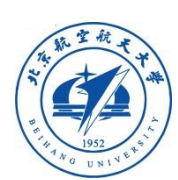
CPU进行读、写操作时，如何通知存储器或IO接口电路？

(1) 读操作  $\overline{RD}$

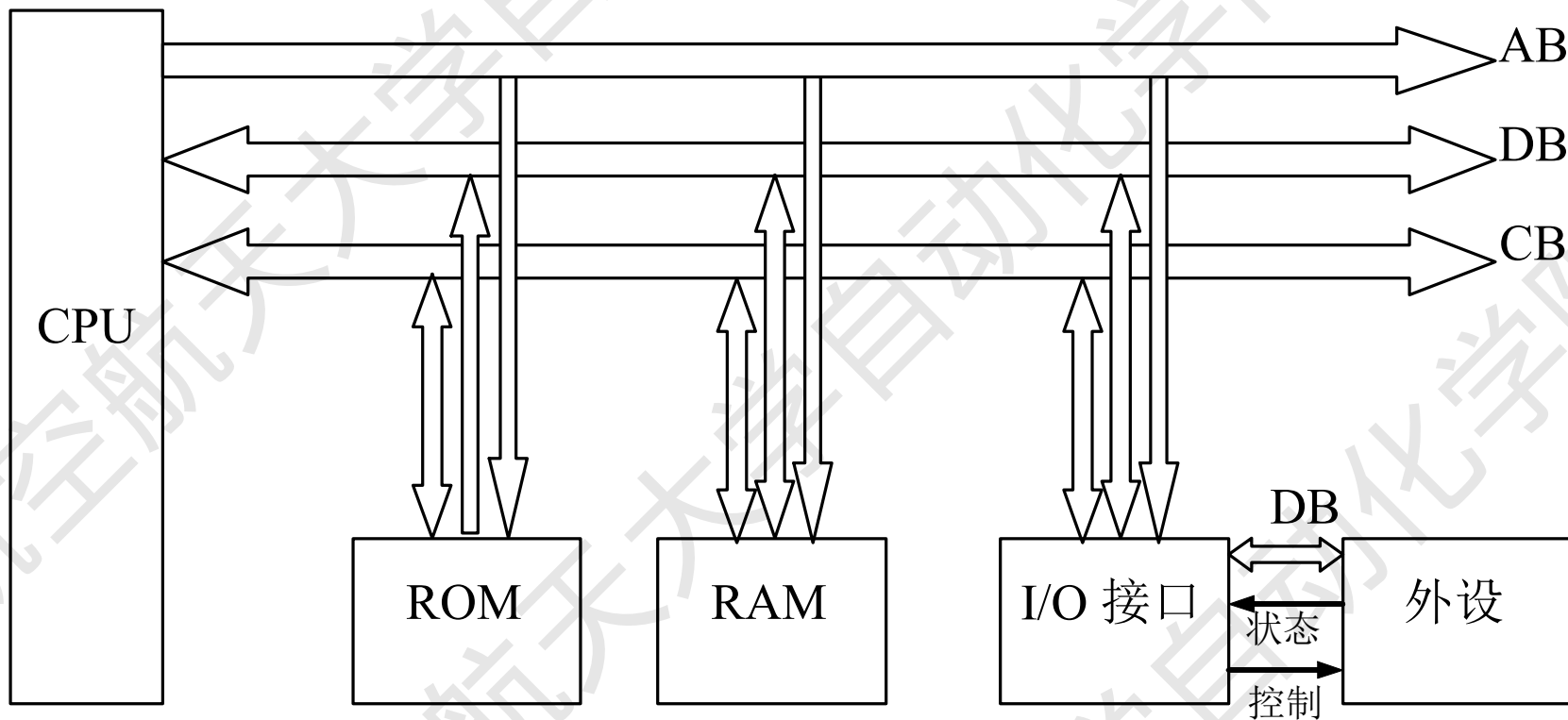
(2) 写操作  $\overline{WR}$

(3) 存储器/IO控制信号  $\overline{M}/\overline{IO}$





## 1.2.2 微型计算机结构



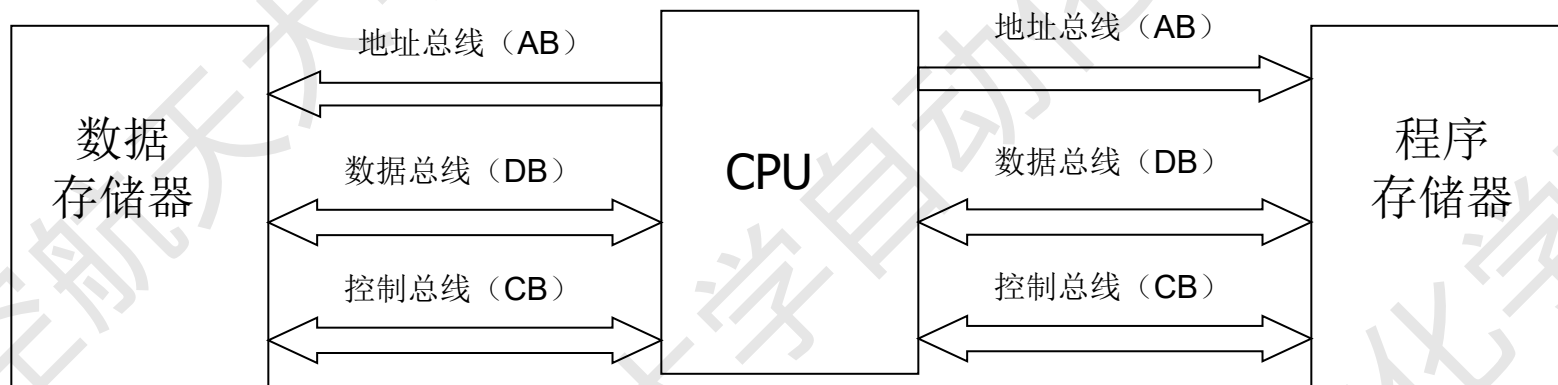
**图 微型计算机基本结构 (简化)**



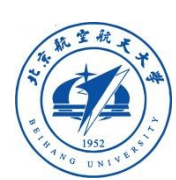


## 1.2.2 微型计算机结构

另外一种常用微机系统结构：哈佛结构



\* MCS-51单片机、ARM Cortex-A/R/M系列芯片采用此结构



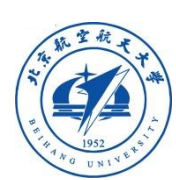
## 1.2.2 微型计算机结构

微机硬件结构由以下部分构成。

- CPU
- 存储器
- 外部设备
- IO接口
- 系统总线

主要特点：

- 运算器、控制器和主寄存器被集成进CPU中；
- 存储器、I/O接口和输入输出设备挂在地址总线、数据总线、以及控制总线上，通过三种总线与CPU相连。

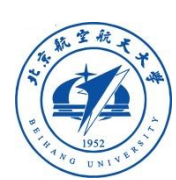


## 1.2.2 微型计算机结构--外部设备

### ■ 外部设备(输入/输出设备、IO设备)

计算机系统的输入 (IN) 和输出 (OUT) 是以相对于主机/CPU而言

- 原始数据和程序传送到计算机中称为**输入**
- 计算机处理好的数据或结果送到外部的过程称为**输出**



## 1.2.2 微型计算机结构--IO接口电路

### ■ 输入/输出接口电路 (IO接口电路)

- 用于CPU/主机与外设之间的信息交换。
- 由于外设的种类繁多，这些设备与CPU之间的工作速度不同、信号电平不同、数据格式不同，因此需要配备不同的I/O接口电路，实现CPU与外设之间的速度匹配、信号电平匹配、信号格式匹配、时序控制、中断控制等等。



## 1.2.2 微型计算机结构---IO接口电路

### ■ 输入/输出接口电路 (IO接口电路)

- 用于CPU/主机与外设之间的信息交换。
- 由于外设的种类繁多，这些设备与CPU之间的工作

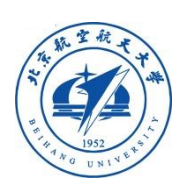
速度不同主要的接口芯片有：

要配备不

速度匹配

制、中

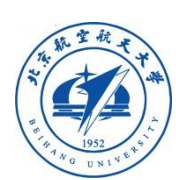
- 可编程计数/定时器8253
- 编程并行接口8255A
- 可编程中断控制器8259A
- 可编程串行接口8251A
- 可编程DMA控制器8237A
- 数模、模数转换芯片(DAC0832,ADC0809)



## 1.2.2 微型计算机结构--总线

### ■ 总线

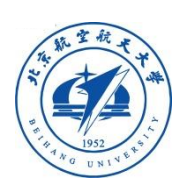
- 总线的一般定义是：传送信息的公共通道，在物理形态上是一组公用的导线和相应的驱动电路。
- 减少各个部件之间的连接线
- 方便系统的构成、扩充或更新



## 1.2.2 微型计算机结构--总线

总线的设计、生产、和使用等方面必须遵守某种规范 (specification) 或标准。总线规范一般包括4个方面：

1. **机械规范**：模板的尺寸、插槽、插头的结构、总线引脚的宽度及间距等等。
2. **逻辑规范**：引脚的定义，信号的描述等等。
3. **电气规范**：信号的电平，传递方向等等。
4. **时序规范**：信号的时序（何时有效、何时无效等等）、相互之间的配合等等。



## 1.2.2 微型计算机结构--总线

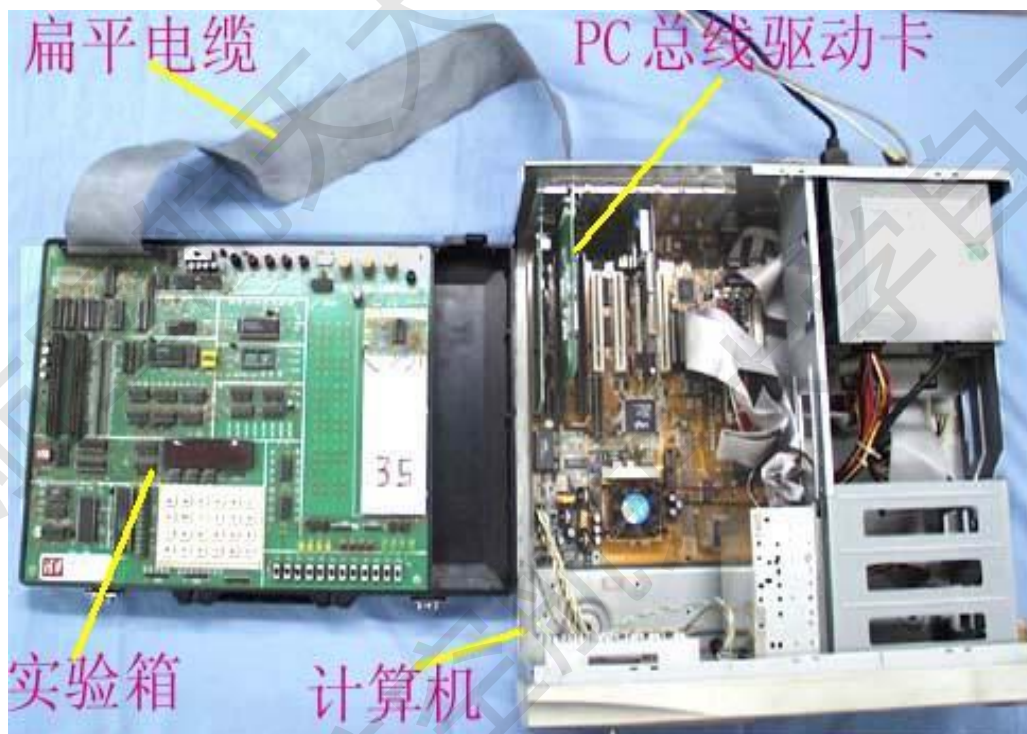
### 总线分类

- 1) **内部总线**: 片内总线
- 2) **元件级总线**: 插件板/电路板内总线
- 3) **系统总线**: 主板/插件板间总线
- 4) **外部总线**:
  - 主板/插件板与外设之间
  - 计算机系统与其它系统之间



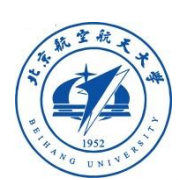


## 1.2.2 微型计算机结构--总线

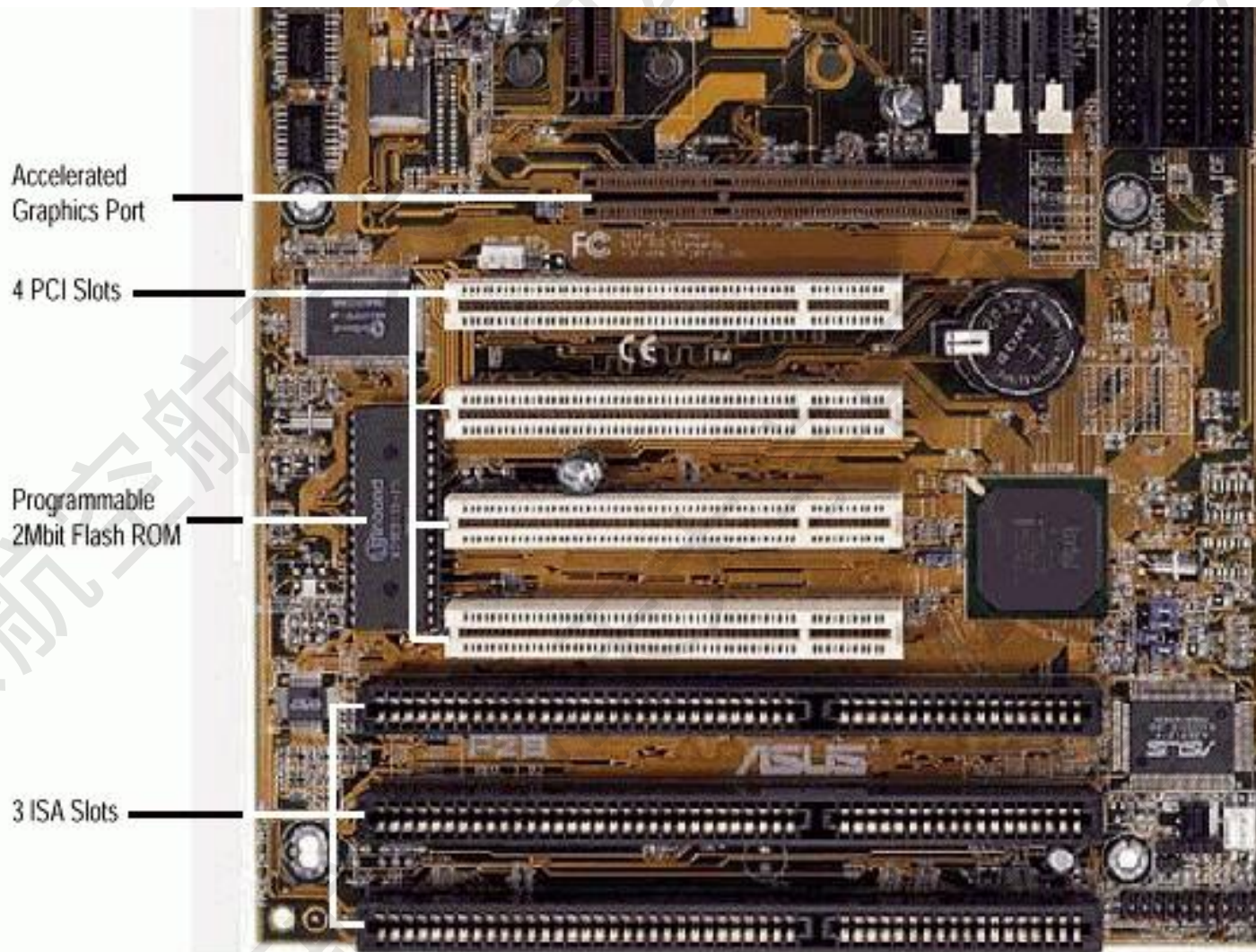


GND	I/O CH CK
RESET	D7
+5V	D6
IRQ2	D5
-5V	D4
DRQ2	D3
-12V	D2
+12V	D1
GND	D0
<u>MEMW</u>	I/O CH RDY
<u>MEMR</u>	AEN
<u>IOW</u>	A19
<u>IOR</u>	A18
<u>DACK3</u>	A17
<u>DRQ3</u>	A16
<u>DACK1</u>	A15
<u>DRQ1</u>	A14
<u>DACK0</u>	A13
CLOCK	A12
IRQ7	A11
IRQ6	A10
IRQ5	A9
IRQ4	A8
IRQ3	A7
<u>DACK2</u>	A6
T/C	A5
ALE	A4
+5V	A3
OSC	A2
GND	A1
	A0

IBM PC/XT总线插槽引脚信号



## 1.2.2 微型计算机结构--总线



PC机主板（部分）



## 1.2.2 微型计算机结构--总线

- 片内总线、元件级总线、系统总线**按功能**进行分类：
  - 地址总线 (AB, Address Bus)
  - 数据总线 (DB, Data Bus)
  - 控制总线 (CB, Control Bus)



## 1.2.3 微机系统组成

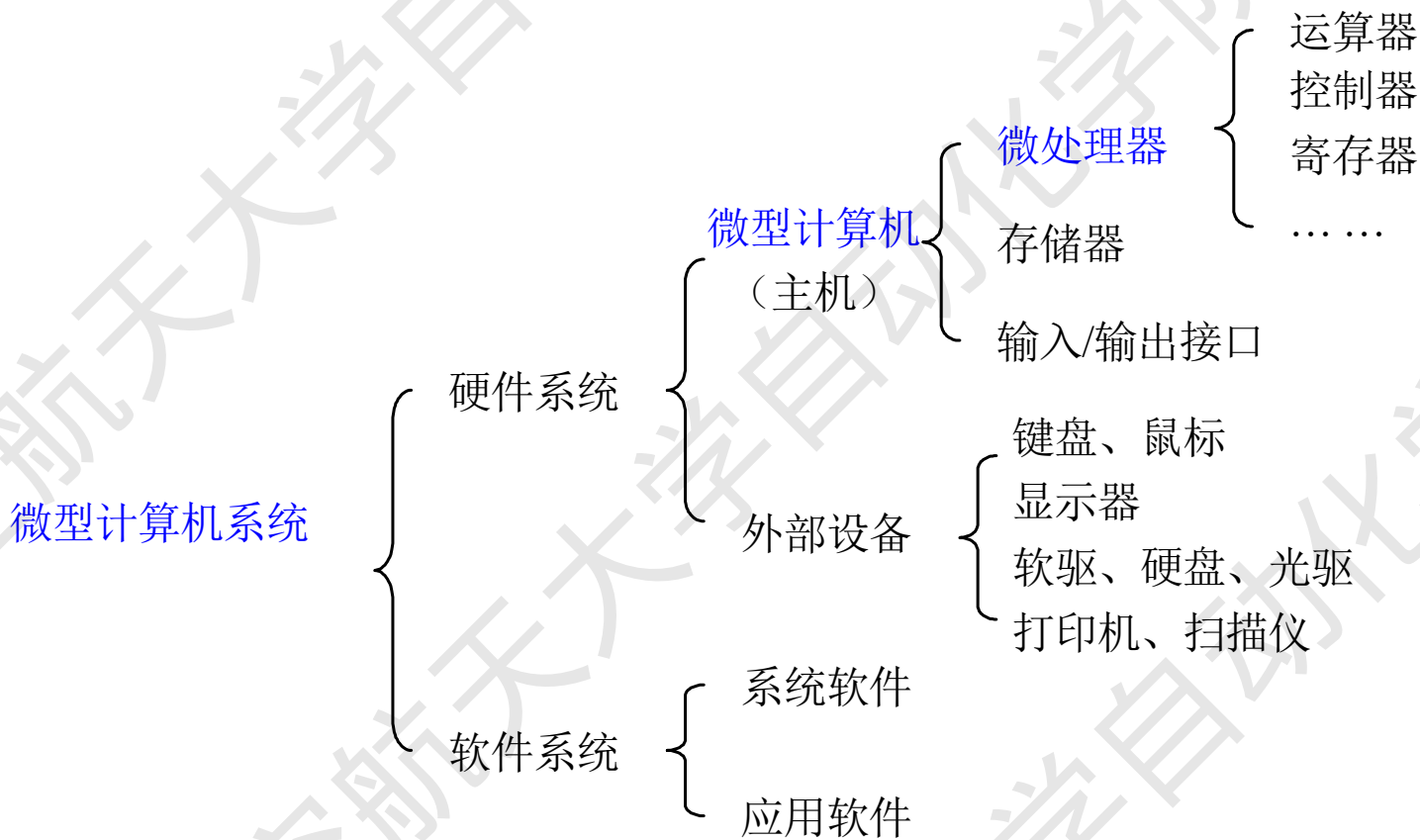
除了前面讲述的硬件系统之外，计算机必须配备软件系统才能工作。软件系统包括系统软件和应用软件。系统软件中比较重要的是操作系统。

- 操作系统：控制计算机中所有的软硬件资源，管理处理器、存储器、外部设备、磁盘文件，对线程/进程进行任务调度和切换。
- 常用的操作系统：
  - DOS, windows
  - UNIX, LINUX, Andoid

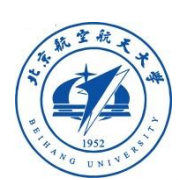




## 1.2.3 微机系统组成



微型计算机与大型计算机、中型计算机、小型计算机在系统结构和工作原理上没有本质的区别，但是由于采用了大规模集成电路器件，使得它具有独特的优点：体积小，重量轻，可靠性高，结构配置灵活，价格低廉

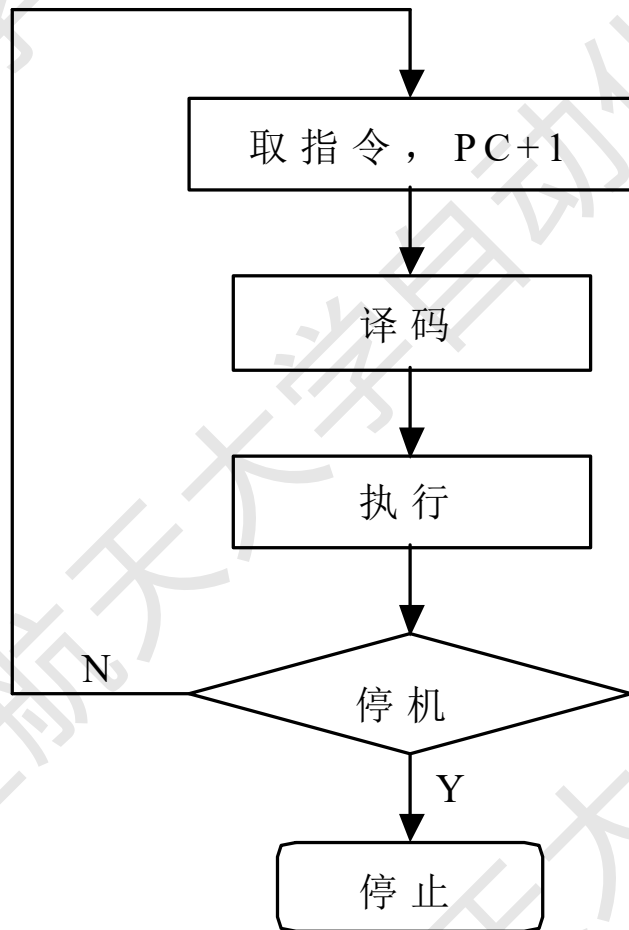


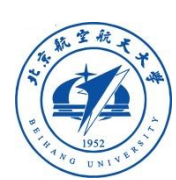
## 1.2.4 计算机系统工作过程

- **计算机的工作过程就是执行程序的过程**，而程序由指令序列组成，因此，执行程序的过程，就是执行指令序列的过程，即逐条地从存储器中取出指令并完成指令所指定的操作。
- 执行每一条指令，都包括**取指、译码和执行**三个基本步骤，所以，微型计算机的工作过程，也就是**不断地取指令、译码和执行的过程**，直到遇到停机指令时才结束机器的运行。



## 1.2.4 计算机系统工作过程





# 第一章小结

- 计数制，计数制之间数的转换：二进制无符号数和十进制的转换。
- 计算机中数的表示方法：计算机中原码、反码、补码的表示法，计算机中定点数的表示方法。二进制补码和十进制之间的转换。
- 信息编码方法：BCD码，ASCII码（换行、回车、英文字符、数字）。
- 算术运算及逻辑运算规则。
- 冯诺依曼结构，微机系统基本结构