计算机导论与程序设计 [CS006001]

段江涛

机电工程学院

西安电子科技大学 XIDIAN UNIVERSITY

2020 年 9 月

lecture-1 主要内容

课程内容

- 计算机导论:了解计算机的基本知识;掌握计算机操作基本技能。
- 程序设计:掌握结构化程序设计方法,训练程序逻辑思维能力。会读、会编、会调试 C 语言程序。
- 学习方法:线上、线下相结合。课堂笔记,认真完成上机练习作业,鼓励大量编程练习。
- 教材
  - 大学计算机,龚尚福,贾澎涛,西安电子科技大学出版社
  - C 程序设计第五版,谭浩强,清华大学出版社
- 智慧教育平台 (使用 Chrome 浏览器): https://cvnis.xidian.edu.cn/
- 线上参考课程资源链接: online resource.pdf

线上导论部分学习内容

1. 计算机历史、现状、发展趋势与前沿技术概述
2. 计算机体系结构及其编码方式
3. 计算机组成与软件系统
4. 计算机应用实践

考核

1. 平时成绩: 10% 根据上机练习作业成绩考核。
2. 导论部分: 20% 结合线上资源, 自学字处理软件。总结知识点, 撰写课程学习报告。
3. 期中考试: 30% 根据机试系统给出的题目编写程序, 通过调试得到正确结果并通过**机试系统提交**。
4. 期末考试: 40% 根据机试系统给出的题目编写程序, 通过调试得到正确结果并通过**机试系统提交**。

计算机导论主要内容

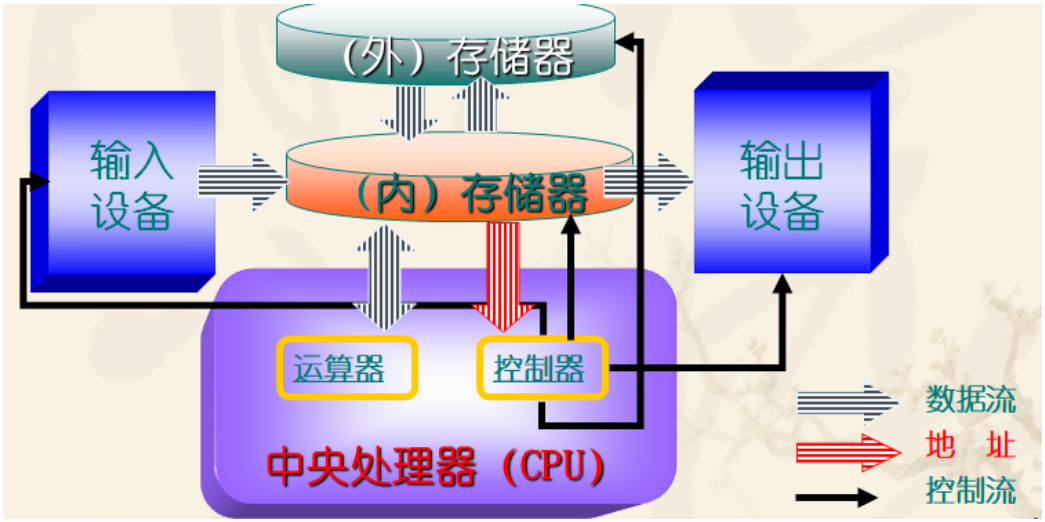
总体要求：了解计算机的基本知识；掌握计算机操作基本技能。

- 计算机系统组成
- 计算机工作原理
- 操作系统
- 字处理：Microsoft Word
- 电子表格：Microsoft Excel
- 演示文稿：Microsoft PowerPoint

计算机工作原理

工作原理: “存储程序” + “程序控制”

1. 以**二进制**形式表示数据和指令
2. 将程序存入存储器中, 由控制器自动读取并执行
3. 外部存储器存储的程序和所需数据 $\Longrightarrow$ 计算机内存 $\Longrightarrow$ 在程序控制下由 CPU 周而复始地取出指令、分析指令、执行指令 $\Longrightarrow$ 操作完成。



十进制与二进制

十进制: 以 10 为底的幂展开式:

$(123)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$;

自低到高各位数 (除 10 取余至商为 0): $3 = 123\%10$, $2 = 123/10\%10 = 12\%10$,

$1 = 123/10/10\%10 = 1\%10$

二进制: 以 2 为底的幂展开式:

$$(77)_{10} = (0100 \quad 1101)_2 = 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4$$
$$+ 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

自低到高各位数 (除 2 取余至商为 0): $1 = 77\%2$, $0 = 77/2\%2 = 38\%2$

$1 = 77/2/2\%2 = 38\%2$, $1 = 77/2/2/2\%2 = 38/2\%2 = 19\%2, \cdots$,

$0 = 77/2/2/2/2/2/2\%2 = 1$, $0 = 77/2/2/2/2/2/2/2\%2 = 0$

10 进制、2 进制、16 进制的幂展开式

$$(D)_{10} = D_{n-1} \times 10^{n-1} + D_{n-2} \times 10^{n-2} + \cdots + D_1 \times 10^1 + D_0 \times 10^0 + D_{-1} \times 10^{-1} + D_{-2} \times 10^{-2} + \cdots + D_{-m+1} \times 10^{-m+1} + D_{-m} \times 10^{-m}$$

$$(B)_2 = B_{n-1} \times 2^{n-1} + B_{n-2} \times 2^{n-2} + \cdots + B_1 \times 2^1 + B_0 \times 2^0 + B_{-1} \times 2^{-1} + B_{-2} \times 2^{-2} + \cdots + B_{-m+1} \times 2^{-m+1} + B_{-m} \times 2^{-m}$$

$$(H)_{16} = H_{n-1} \times 16^{n-1} + H_{n-2} \times 16^{n-2} + \cdots + H_1 \times 16^1 + H_0 \times 16^0 + H_{-1} \times 16^{-1} + H_{-2} \times 16^{-2} + \cdots + 16_{-m+1} \times 16^{-m+1} + H_{-m} \times 16^{-m}$$

进制对照表

十进制	二进制	十六进制	十进制	二进制	十六进制
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

十进制、二进制与十六进制举例

$$(123)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0;$$

自低到高各位数: $3 = 123\%10,\ 2 = 123/10\%10,\ 1 = 123/10/10\%10$

$$(77)_{10} = (0100 \quad 1101)_2 = 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4$$
$$+ 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$(77)_{10} = (4D)_{16} = 4 \times 16^1 + 13 \times 16^0$$

实型十进制数转换为二进制数

分别转换整数部分和小数部分。整数部分: 除 2 取余, 至商为 0, 逆序排列余数, 得到整数部分的二进制位。小数部分, 乘 2 取整, 至小数部分为 0 或指定精度, 正序排列, 即得小数部分的二进制位。

$$
\begin{aligned}
(11.625)_{10} &= 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3} \\
&= (1011.101)_2
\end{aligned}
$$

$$
\begin{array}{rrr}
0.625 & 0.25 & 0.5 \\
\times\quad 2 & \times\quad 2 & \times\quad 2 \\
\hline
1.250 & 0.50 & 1.0
\end{array}
$$

正序排列各整数得到小数部分的二进制位 $(101)_2$

$(0.101)_2 = 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3} = 0.625$

例: 把 0.5773 转换成二进制 (保留到小数点后 7 位)。

注: 用二进制表示小数部分有精度问题。

	积的整数部分
$0.5773 \times 2 = 1.1546$	1
$0.1546 \times 2 = 0.3092$	0
$0.3092 \times 2 = 0.6184$	0
$0.6184 \times 2 = 1.2368$	1
$0.2368 \times 2 = 0.4736$	0
$0.4736 \times 2 = 0.9472$	0
$0.9472 \times 2 = 1.8944$	1

$$(0.1001001)_2 = 1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} + 0 \times 2^{-5} + 0 \times 2^{-6} + 1 \times 2^{-7}$$
$$= 0.5703125 \neq 0.5773$$

数值在计算机中的表示 (以 8bit 编码为例)

- 原码:正数的符号为 0,负数的符号为 1,其它位按一般的方法表示数的绝对值。

$$x = (+103)_{10} \qquad [x]_{原} = (01100111)_2$$

$$x = (-103)_{10} \qquad [x]_{原} = (11100111)_2$$

- 反码: 正数的反码与原码相同;负数的反码是符号位不变,其他位按位取反
- 补码: 正数的补码与其原码相同;负数的补码为其反码最末位加 1. 即,

负数补码 = 反码 +1 = 2^n − 该数的绝对值, n 是编码二进制位数.

$$(77)_{10} = (0100\quad 1101)_2, \qquad (-77)_{10} = (1100\quad 1101)_2$$

$$(-77)_{补} = 2^8 - 77 = 1111\quad 1111 + 0000\quad 0001 - 0100\quad 1101$$

$$= \underbrace{1111\quad 1111 - 0100\quad 1101}_{(-77)_{反}} + 0000\quad 0001$$

$$= \underbrace{1011\quad 0010}_{(-77)_{反}} + 0000\quad 0001 = 1011\quad 0011$$

数值表示示例


+77
原码
0 1 0 0 1 1 0 1
反码
0 1 0 0 1 1 0 1
补码
0 1 0 0 1 1 0 1
−77
原码
1 1 0 0 1 1 0 1
反码
1 0 1 1 0 0 1 0
补码
1 0 1 1 0 0 1 1

机内以补码形式存储有符号数

1. 对于正数，原码 = 反码 = 补码
2. 对于负数，补码 = 反码 + 1

   反码 = 符号位不变，其他位按位取反
3. 补码是可逆的，即再对补码求补得到原码。
4. 引入补码后，使减法统一为加法。

   $(+77)_{补} + (-77)_{补} = 0100\ 1101 + 1011\ 0011 = 0000\ 0000$

补码运算实例 (以 8bit 编码为例)

补码可逆:

$$[-25]_{原} = (1001\ 1001)_2 \quad [-25]_{反} = (1110\ 0110)_2$$

$$[-25]_{补} = [-25]_{反} + 1 = (1110\ 0110 + 1)_2 = (1110\ 0111)_2$$

$$[-25]_{原} = ([-25]_{补})_{补} = (1001\ 1000 + 1)_2 = (1001\ 1001)_2$$

减法统一为加法: $[a - b]_{补} = a_{补} + [-b]_{补}$

$$[102 - 25]_{补} = [77]_{补} = (0100\ 1101)_2 = 77$$

$$[102]_{补} + [-25]_{补} = (0110\ 0110)_2 + (1110\ 0111)_2 = (0100\ 1101)_2 = 77$$

所以, $[102 - 25]_{补} = [102]_{补} + [-25]_{补}$

同样有, $[25 - 102]_{补} = [25]_{补} + [-102]_{补}$

ASCII 编码表 $B_6B_5B_4B_3B_2B_1B_0$

$B_6B_5B_4$ / $B_3B_2B_1B_0$	000	001	010	011	100	101	110	111
0000	NUL	DLE	空格	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	ENQ	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

- ASCII 码连续排列

  ‘0’～‘9’, ‘A’～‘Z’, ‘a’～‘z’

- 数字 = 编码值 - ‘0’

  9=‘9’-‘0’

- 大小字符间隔:

  ‘a’ - ‘A’ = 32

  ‘a’=0110 0001=61H=0X61=97

  ’A’=0100 0001=41H=0X41=65

计算机程序

指令

可以被计算机理解并执行的基本操作命令。

程序

一组计算机能识别和执行的指令。
一个特定的指令序列用来完成一定的功能。

软件

与计算机系统操作有关的计算机程序、规程、规则，以及可能有的文件、文档及数据。

计算机语言

机器语言

计算机能直接识别和接受的二进制代码称为**机器指令**。机器指令的集合就是该计算机的**机器语言**。

特点：难学，难记，难检查，难修改，难以推广使用。依赖具体机器难以移植。

```
B8  7F 01
BB  21 02
03   D8
B8  1F 04
2B  C3
```

汇编语言

机器语言的符号化。用英文字母和数字表示指令的**符号语言**。

特点：相比机器语言简单好记，但仍然难以普及。

汇编指令需通过**汇编程序**转换为机器指令才能被计算机执行。依赖具体机器难以移植。

```
MOV  AX  383
MOV  BX  545
ADD   BX  AX
MOV  AX  1055
SUB    AX  BX
```

高级语言

高级语言更接近于人们习惯使用的自然语言和数学语言。

特点：功能强大，不依赖于具体机器。用高级语言编写的**源程序**需要通过**编译程序**转换为机器指令的**目标程序**。

```
int  x =1055, y = 383, z = 545
int S;
S = x-(y+z);
S=1055-(383+545)
```

高级语言的发展

规定：

程序必须由具有良好特性的基本结构(**顺序结构、选择结构、循环结构**)构成，程序中的流程不允许随意跳转，程序总是由上而下顺序执行各个基本结构。

特点：

程序结构清晰，易于编写、阅读和维护。

C 语言的特点

1. 语言简洁、紧凑,使用方便、灵活
2. 运算符丰富
3. 数据类型丰富
4. C 语言是完全模块化和结构化的语言

   具有结构化的控制语句 (顺序、选择、循环结构)

   用函数作为程序的模块单位,便于实现程序的模块化
5. 兼具高级语言和低级语言的功能

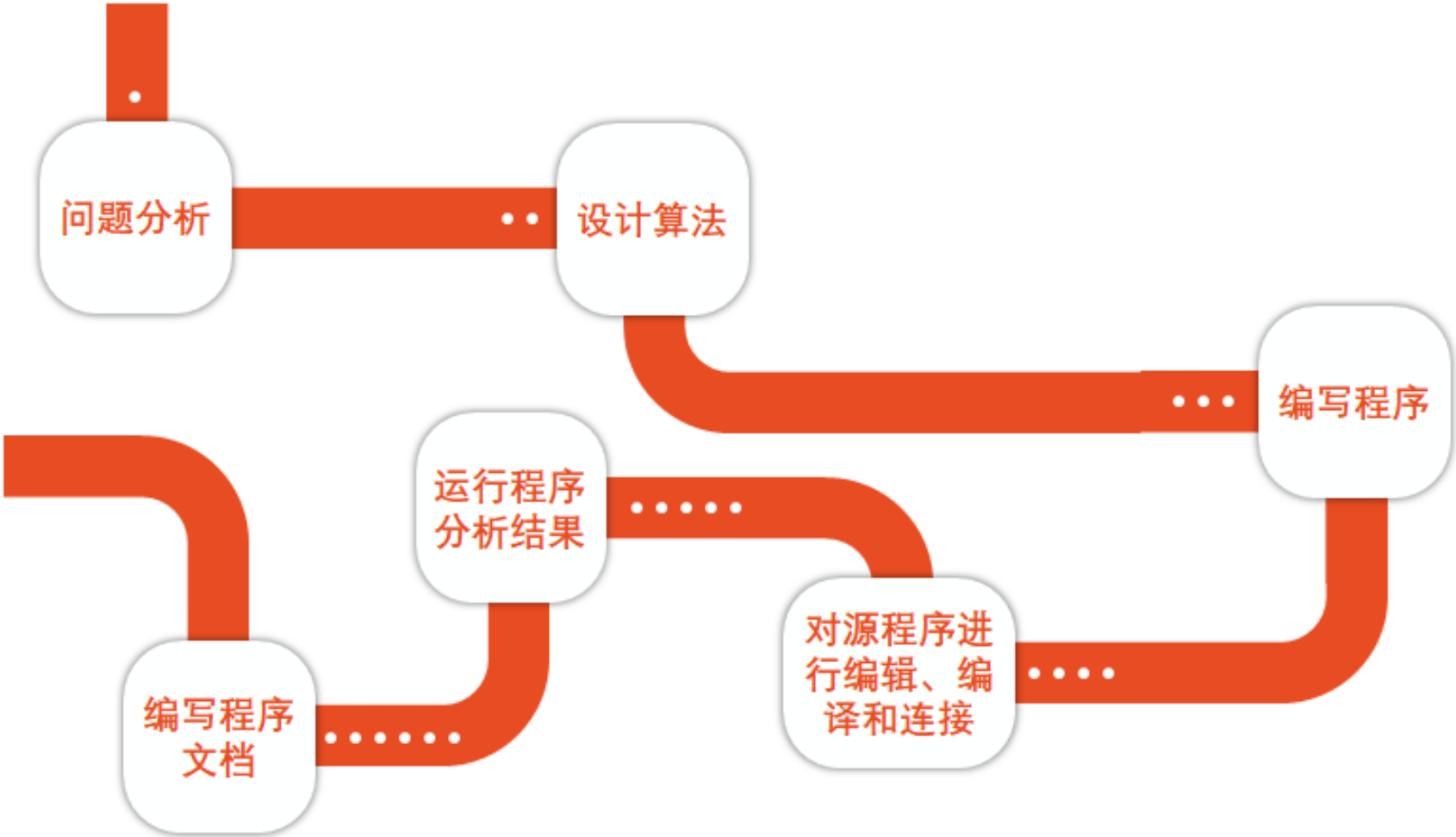
   允许直接访问物理地址

   能进行位 (bit) 操作

   能实现汇编语言的大部分功能

   可以直接对硬件进行操作

程序设计的任务



第一个 C 语言程序

```c
#include<stdio.h>                    // standard input/output编译预处理指令
int main()                           // 主函数
{                                    // 函数开始标志
    printf("Hello World!");          // 输出一行信息
    return 0;                        // 函数执行完毕返回函数值0
}                                    // 函数结束标志
```

求两个整数之和

```c
#include<stdio.h>           // standard input/output编译预处理指令
int main()                  // 主函数
{                           // 函数开始标志
    int a,b,sum;            // 定义a,b,sum为整型变量
    a=123;                  // 对a,b赋值
    b=456;
    sum=a+b;                // 计算a+b, 并把结果存放在变量sum中
    printf("sum␣is␣%d\n",sum);// 输出结果
    return 0;               // 函数执行完毕返回函数值0
}                           // 函数结束标志
```

欢迎批评指正！