STC单片机C语言运算符

主 讲：何宾

Email：hebin@mail.buct.edu.cn

2016.03

STC单片机C语言运算符

- 在C语言中，提供了丰富的运算符用于实现复杂的运算功能，包括：算术运算、逻辑运算及逻辑判断等。本章将详细的介绍C语言中所提供的赋值运算符、算术运算符、递增和递减运算符、关系运算符、逻辑运算符、位运算符、符合赋值运算符、逗号运算符、条件运算符、强制类型转换符以及sizeof运算符。
- 通过本章内容的学习，掌握C语言运算符的表示方法及所实现的功能，灵活高效地运用这些运算符实现对数据进行处理。

C语言运算符分类

- 在C语言中，提供了丰富的运算符用于对数据的处理。通过运算符和数据的有效合理搭配，就构成了可以完成所要求运算功能的表达式。对于C语言的运算符来说，可以从下面的角度进行分类：
- 根据所实现的功能，C语言中的运算符可以分为赋值运算符、算术运算符、递增和递减运算符、关系运算符、逻辑运算符、位运算符、符合赋值运算符、逗号运算符、条件运算符、指针和地址运算符、强制类型转换运算符和sizeof等运算符。

C语言运算符分类

- 根据参与运算数据的个数，C语言中的运算符可以分为单目运算符、双目运算符和三目运算符。
  - 对于单目运算符，只有一个操作数。典型地，包括：递增递减运算符、取地址运算符、指针运算符、取正数运算符，取负数运算符、按位取反运算符和sizeof运算符。
  - 对于双目操作符，有两个操作数。典型地，包括加法、减法、乘法和除法运算符等。
  - 对于三目操作符，有三个操作数。典型地，条件运算符。

C语言运算符优先级

- 所谓的运算符优先级是指，在一个运算表达式中，执行运算的先后顺序。在C语言中，共分了15级优先级，最高优先级称为第1级，最低优先级称为第15级，也就是所数字越高优先级越低。

注：表中的一些其它运算符在后续章节中将详细介绍。

GPNEWTECH 汇众新特

赋值运算符

在C语言中，赋值操作使用“=”号实现，“=”称为赋值运算符，赋值语句的格式为：

变量=表达式;

先计算由表达式所得到的值，然后在将该值分配给变量。

【例10-1】不同类型之间赋值操作的例子

代码清单10-1 main.c文件

```
int main()
{
volatile unsigned int a;      //定义无符号的整型变量a，带有volatile属性
volatile int b=-2345;        //定义带符号的整型变量，初值为-2345
a=b;                         //将b的值赋给a，
return 0;
}
```

GPNEWTECH 汇众新特

赋值运算符

- 读者可以进入到本书所提供资料的stc_program_example\例子10-1\目录下，在Keil μVision5集成开发环境下打开该设计，并进入调试器模式，使用单步运行。根据前面说提到的，由于b是有符号的变量，而a是无符号的变量，当把有符号变量的数赋值给无符号变量a的时候，会有什么结果？下面对该段代码进行分析。
- 在Disassembly窗口中，查看该段C语言代码的反汇编代码。

GPNEWTECH 汇众新特

赋值运算符

代码清单10-2 main.c的反汇编代码

```
 2: {
 3:        volatile unsigned int a;
 4:        volatile int b=-2345;
C:0x0003   750AF6   MOV     0x0A,#0xF6
C:0x0006   750BD7   MOV     0x0B,#0xD7
 5:        a=b;
C:0x0009   850A08   MOV     0x08,0x0A
C:0x000C   850B09   MOV     0x09,0x0B
 6:        return 0;
```

GPNEWTECH 汇众新特

赋值运算符

```
C:0x000F    E4      CLR     A
C:0x0010    FE      MOV     R6,A
C:0x0011    FF      MOV     R7,A
    7: }
```

- **从代码很清楚的可以知道下面的事实，即：**
  - **将立即数0xF6D7保存到8051单片机片内基本数据RAM地址为0x0A和0x0B的空间，该数字是变量b的值-2345的二进制补码表示。**
  - **b=a的操作就是，将8051单片机片内基本数据RAM地址为0x0A和0x0B的内容保存到8051片内基本数据RAM地址为0x08和0x09的位置，也就是变量a的内容。**

赋值运算符

- 但是声明a是无符号的整数，因此就将8051片内基本数据RAM地址为0x08和0x09位置的内容当作无符号处理即0xF6D7的无符号数表示为63191。
- 因此a的值最终表示为63191，b的值最终表示为-2345。
- 注：虽然这两个变量a和b的值在8051片内基本数据RAM内饰完全一样的，但是由于在声明变量a和b的时候，其类型是不同的，因此最终导致对0xF6D7的理解完全不同。
- 通过这个分析过程清楚地知道，要想学好C语言是需要了解一些CPU的细节的，否则C语言的这些语法是无法彻底理解清楚的。

赋值运算符

- 从上面的过程可以清楚地看出来赋值的过程：
  - 赋值过程是从右到左，即先等到“=”号右侧表达式的值；然后，在将右侧表达式的值赋值给“=”号左侧的变量。
  - 在赋值过程中，如果“=”号左右两侧的数据类型不相同的时候，将右侧表达式的值自动的转换到与“=”左侧的数据类型相一致。

赋值运算符

- 在调试器模式下，在keil μVision集成开发环境右下角的Watch 1窗口中，分别输入两个变量a和b的名字，在右侧可以清楚地显示这两个变量的值，一目了然。
- 退出调试器模式，并关闭该设计。

Watch 1

Name	Value	Type
a	63191	uint
b	-2345	int
<Enter expression>		

GPNEWTECH 汇众新特

赋值运算符

【例10-2】正确赋值操作的例子

代码清单10-3 main.c文件

```
int main()
{
int a;
char b;
float c;
a=10000;
b=200;
c=0.5;
        return 0;
}
```

GPNEWTECH 汇众新特

赋值运算符

总结：

当赋值运算符两边的运算对象不同时，将要发生类型转换，转换的规则是：把赋值运算符右侧表达式的类型转换为左侧变量的类型，即：

- **浮点型float与整型int。将浮点数转换为整数时，将舍弃浮点数的小数部分，只保留整数部分。将整数值赋值给浮点变量时，数值保持不变，只是将形式改为浮点形式。**
- **字符型char与整型int。当整型int数值赋值给字符型char变量时只保留其最低8位，高位部分舍弃。当字符型char数值赋值给int型变量时，如果原来的char型数据取正值，转换后仍为正值；如果原来char型值可正可负，则转换后保持原值，只是数据的内部表示形式有所不同，比如对于下面代码：**

赋值运算符

```
volatile char a=-127;
  volatile int c;
  c=a;
```

- 当a赋值给c后，c的值也是-127。只不过a和c在8051单片机内表示的形式不同而已，a=-127用一个字节表示为0x81，而c=-127用两个字节表示为0xFF81，从实质上来说，也是符号扩展。当变量a值的符号位为1，表示负数时，使用1扩展，即用1将高位填满得到整型变量的值；当变量a的符号位为0，表示正数时，使用0扩展，即用0将高位填满得到整型变量的值。

赋值运算符

- 整型int与长整型long。当把长整型long的数值赋值给整型int变量时，将低16位值送给int型变量，而将高16位截断舍弃；将整型int的数值赋值给长整型long变量时，其外部值保持不变，而内部形式有所改变，比如对于下面的代码：

```
volatile int a=-32768;
volatile long int c;
c=a;
```

GPNEWTECH 汇众新特

赋值运算符

- 当a赋值給c后，c的值也是-32768。只不过a和c在8051单片机内表示的形式不同而已，a=-32768用两个字节表示为0x8000，而c=-127用四个字节表示为0xFFFF8000，从实质上来说，也是符号扩展。当变量a值的符号位为1，表示负数时，使用1扩展，即用1将高位填满得到长整型变量的值；而当变量a的符号位为0，表示正数时，使用0扩展，即用0将高位填满得到长整型变量的值。
- 无符号整数。将一个无符号型unsigned数据赋值给一个占用相同存储单元的整型变量时，比如 unsigned int到int，unsigned long到long，unsigned short到short，原值照赋，内部存储方式不变，但是外部的值可能变化，对于下面的代码：

```
volatile unsigned int a=40001;
volatile  int c;
c=a;
```

GPNEWTECH 汇众新特

赋值运算符

- 因为a为无符号的整型变量，它在8051单片机中占用两个字节表示为0x9c41。当赋值给c的时候，将0x9c41赋值给c，但是由于c为带符号的整型变量，此时将0x9c41理解为负数，也就是补码等效的十进制数为-25535。
- 当把一个非unsigned整型数据赋值给长度相同的unsigned型变量时，内部存储实型不变，但是外部表示时总是无符号的，对于下面的代码：

```
volatile  int a=-30000;
volatile unsigned int c;
c=a;
```

赋值运算符

- 因为a是有符号的整型变量，它在8051单片机内占用两个字节表示为0x8ad0。当赋值给c的时候，将0x8ad0赋值给c，但是由于c为无符号的整型变量，此时将0x8ad0理解为正数，即十进制数的35536。
- C语言这种赋值时的类型转换形式可能会使程序员感到不严谨，因为不管表达式的值怎样，系统都自动将其转换为赋值运算符左边变量的类型。

❖ 注：在进行赋值操作的时候，一定要注意变量和表达式值的数据类型。否则，会产生意想不到的后果，这可能是C语言的一个缺点，但是不要忘记当初C语言出现的时候，就是为了替换汇编语言而设计的，所以类型变换比较随意。

算术运算符

- 在C语言中，所提供的算术运算符包括：+（加法运算或者取正数运算）、 -（减法运算或者取负数运算）、*（乘法运算）、/（除法运算）、%（取余运算）。在这些算术运算中，除取正和取负运算是单目运算外，其他都是双目运算。
- 在求取表达式的值时，按照运算符的优先级进行。按照下面优先级：

- 单目运算的优先级要高于双目运算。
- 在双目运算中，优先级按照*、/、%、+、 - 从高到低排列。程序员可以通过使用（ ）修改运算的优先级顺序。

GPNEWTECH 汇众新特

算术运算符

- 对于整数除法运算来说，遵循以下的规则：
  - 整数相除且不能整除时，如果除数和被除数其中只有一个为负数运算符“/”的运算结果就为负数；如果除数和被除数两个均为负数，运算符“/”的运算结果就为正数，这与代数运算的结果一致。
  - 整数相除且不能整除时，如果除数和被除数中出现负数，则运算符“%”的运算结果由表达式(a/b)*b+a%b=a确定，即a%b=a-(a/b)*b。
  - 对于运算符“/”和“%”来说，除数（第二个操作数）不能为0。

GPNEWTECH 汇众新特

算术运算符

【例10-3】有符号整数相除的例子。

代码清单10-4 main.c文件

```
int main()
{
int a1=-20,b1=3,c1,d1;
int a2=20,b2=-3,c2,d2;
int a3=-20,b3=-3,c3,d3;
c1=a1/b1;
d1=a1%b1;
c2=a2/b2;
d2=a2%b2;
c3=a3/b3;
d3=a3%b3;
return 0;
}
```

GPNEWTECH 汇众新特

算术运算符

【例10-4】不同数据类型运算和赋值的例子。

代码清单10-5 main.c文件

```
int main()
{
int a=1000,b=30,c=15;
float d,e;
d=a/b*c;
e=a/(b*c);
return 0;
}
```

GPNEWTECH 汇众新特

算术运算符

下面对该程序代码进行分析。

- 对于d=a/b*c的运算来说：
  - “=”号右侧都是整型变量，因此按照整数运算的规则。先计算a/b的结果，然后该结果再与c相乘。
  - 由于等号左侧变量d的数据类型为浮点数，所以将等号右侧的整型运算结果转换成浮点型，然后赋值给变量d。

算术运算符

- 对于e=a/(b*c)运算来说：
  - “=”号右侧都是整型变量，因此按照整数运算的规则。在该运算中，通过符号“()”改变了运算顺序，即先计算b*c的结果，然后a再与该结果相除。
  - 由于等号左侧变量e的数据类型为浮点数，所以将等号右侧的整型运算结果转换成浮点型，然后赋值给变量e。

GPNEWTECH 汇众新特

算术运算符

【例10-5】算术运算操作的例子

代码清单10-6 main.c文件

```
int main()
{
int a=1000,b=33,c,d,h,i;
long int e,j;
float m=13414e-3,n=0.6547;
  float f,g,k,l;
c=a/b;              //整数除法，求商
d=a%b;              //整数除法，求余
e=a*b;              //乘法求积
f=a/b;              //整数除法，然后类型转换
```

GPNEWTECH 汇众新特

算术运算符

```
g=a+b-c;          //整数加法和减法运算，然后类型转换
h=(a+b)*c;        //带()的优先级加法和乘法运算
i=b-a*c;          //整数乘法和减法运算
j=(long)a*b;      //转换为长整型，然后长整型乘法
k=m*n;            //浮点数相乘
l=m/n;            //浮点数相除
return 0;
}
```

GPNEWTECH 汇众新特

算术运算符

- 在C语言算数表达式的计算过程中，每一步计算所得结果的数据类型由参与运算的运算对象决定，相同数据类型的两个对象运算，结果数据类型不变，不同数据类型的运算对象进行运算，结果的数据类型由高精度的运算对象确定。如果一个运算符两边的运算类型不同，先要将其转换为相同的类型，即较低类型转换为较高类型，然后再参与运算，转换规则如下：

char，short->int->unsigned->long->float

GPNEWTECH 汇众新特

算术运算符

【例10-6】不同类型数据进行算术运算操作的例子

代码清单10-7 main.c文件

```
int main()
{
volatile  int a=-30000;
volatile unsigned int b=20000;
int c;
c=a+b;
return 0;
}
```

GPNEWTECH 汇众新特

算术运算符

- 读者可以进入到本书所提供资料的stc_program_example\例子10-6\目录下，在Keil μVision5集成开发环境下打开该设计，并进入调试器模式，使用单步运行。根据前面说提到的，由于b是有符号的变量，而a是无符号的变量，当把有符号变量的数赋值给无符号变量a的时候，会有什么结果？下面对该段代码进行分析。
- 在Disassembly窗口中，查看该段C语言代码的反汇编代码。

代码清单10-8 main.c的反汇编代码

```
2: {
3:        volatile  int a=-30000;
C:0x0003    75088A   MOV      0x08,#TL0(0x8A)
C:0x0006    7509D0   MOV      0x09,#PSW(0xD0)
```

GPNEWTECH 汇众新特

算术运算符

```
4:        volatile unsigned int b=20000;
  5:        int c;
C:0x0009    750A4E   MOV      0x0A,#0x4E
C:0x000C    750B20   MOV      0x0B,#0x20
  6:        c=a+b;
C:0x000F    E509     MOV      A,0x09
C:0x0011    250B     ADD      A,0x0B
C:0x0013    F50D     MOV      0x0D,A
C:0x0015    E508     MOV      A,0x08
C:0x0017    350A     ADDC     A,0x0A
C:0x0019    F50C     MOV      0x0C,A
```

GPNEWTECH 汇众新特

算术运算符

```
 7:        return 0;
C:0x001B   E4     CLR    A
C:0x001C   FE     MOV    R6,A
C:0x001D   FF     MOV    R7,A
    8: }
```

- **从代码很清楚的可以知道下面的事实，即：**
  - **变量a是有符号的整型数，在8051中占用两个字节，用补码表示为0x8AD0。**
  - **变量b是无符号的整型数，在8051中占用两个字节，用补码表示为0x4E20。**

GPNEWTECH 汇众新特

算术运算符

- 虽然a是有符号的整型数，但在执行a+b的运算时，并不在乎0x8AD0原本表示的数是有符号还是无符号的，直接就是0x8AD0+0x4E20的运算操作。当然，从C语言的角度是说，当在运算的时候，因为变量a和变量b的类型不同，将其转换成无符号的类型。因此，C语言的这些规则，其实是真实运算工程的文字表述而已。
- 在运算完成后，由于赋值符号左侧为有符号的整型，所以将0x8AD0+0x4E20的计算结果0xd8f0作为有符号数，等效于十进制数的-10000。

■ 退出调试器模式，并关闭该设计

递增和递减运算符

- 在C语言中，提供了递增运算符++和递减运算符－－，它们的作用是为了对运算的数据进行加一和减一的操作。典型的,++i,i++,－－i,i－－。
- ++i和i++的含义是不一样的：
  - ++i是先执行i+1操作，然后再使用i的值，即：先执行i+1=>i的操作，然后使用递增后的i值。
  - i++是先使用i的值，然后再执行i+1的操作，即先使用i当前的值，然后再执行i+1=>i的操作。
- 类似的，－－i和i－－是不一样的：
  - －－i是先执行i－1操作，然后再使用i的值，即：先执行i-1=>i的操作，然后再使用i的值。
  - i－－是先使用i的值，然后再执行i－1的操作，即先使用i当前的值，然后再执行i-1=>i的操作。

GPNEWTECH 汇众新特

递增和递减运算符

【例10-7】递增和递减运算符的例子

代码清单 10-9 main.c文件

```
void main()
{
int a=40,b,c,d,e,f,g;        //整型变量a的初始值为40
b=a--;                       //先执行a=>b操作,然后再执行a-1=>a操作
c=a;                         //执行a=>c操作
d=--a;                       //先执行a-1=>a操作，然后再执行a=>d操作
e=a++;                       //先执行a=>e操作，然后再执行a+1=>a操作
f=a;                         //执行a=>f操作
g=++a;                       //先执行a+1=>a操作，然后再执行a=g操作
}
```

GPNEWTECH 汇众新特

关系运算符

在C语言中，提供了下面的关系运算符：>（大于）、<（小于）、>=（大于等于）、<=（小于等于）、==（等于）和!=（不等于）。

其中：前四个关系运算符的优先级相同，后两个关系运算符优先级相同。前一组关系运算符的优先级高于后一组的关系运算符。

由这些运算符构成的关系表达式用于后面所介绍的条件判断语句中，用于确定判断的条件是否成立。关系表达式的格式为：

表达式1 关系运算符 表达式2

关系运算的结果只有“1”和“0”两个值，即：当满足比较的条件时，关系运算的结果为“1”；否则为“0”。比如：100>99，返回值为1；10>(2+10)，返回值为0。

GPNEWTECH 汇众新特

关系运算符

【例10-8】关系运算符的例子

代码清单10-10 main.c文件

```
int main()
{
int a=40,b=10;        //初始化整型变量a和b的值分别为40和10
bit c,d,e,f,g,h;      //声明bit类型的变量c、d、e、f、g和h
c=a<b;            //a<b判断条件不成立，返回0
d=a<=b;            //a<=b判断条件不成立，返回0
e=a>b;            //a>b判断条件成立，返回1
f=a>=b;           //a>=b判断条件成立，返回1
g=a!=b;           //a!=b判断条件成立，返回1
h=a==b;           //a==b判断条件不成立，返回0
return 0；
}
```

GPNEWTECH 汇众新特

逻辑运算符

在C语言中，提供了逻辑运算符，包括：&&（逻辑与）、||（逻辑或）和!（逻辑非）。

逻辑运算符用在多个关系表达式的条件判断语句中，有以下三种格式。

- 条件表达式1 && 条件表达式2 && ……

用于确定这些条件表达式条件是否同时成立。如果条件表达式1到条件表达式n都成立，则返回1；否则返回0。

- 条件表达式1 || 条件表达式2|| …..

用于确定这些条件表达式中是否存在有条件表达式成立的情况。如果在条件表达式1到条件表达式n中只要有一个条件表达式条件成立则返回1；否则返回0。

GPNEWTECH 汇众新特

逻辑运算符

- ！条件表达式

用于对条件表达式的返回值进行取反操作。如果条件表达式的值为1，则!操作将对条件表达式的值取反，变成0；如果条件表达式的值为0，则!操作将对条件表达式的值取反，变成1。逻辑关系符的优先级按照：!、&&、||依次降低。

【例10-9】逻辑运算符的例子

代码清单 10-11 main.c文件

```
int main()
{
int a=40,b=10;        //声明两个整型变量a和b的值 分别为40和10
bit c,d,e,f,g,h;      //声明bit型变量c、d、e、f、g、h和i
```

GPNEWTECH 汇众新特

逻辑运算符

```
c=a<b && a>b;       //条件a<b和a>b并不同时成立，所以&&操作返回0
d=a<=b || a>b;      //条件a<=b和a>b有一个条件成立，所以||操作返回1
e=!(a>b);           //a>b条件成立，取！后返回0
f=a!=b && a==b;     //a!=b和a==b并不同时成立，所以&&操作返回0
g=a!=b || a==b;     //a!=b和a==b有一个条件成立，所以||操作返回0
h=!(a==b);          //a==b条件不成立返回0，!取反后为1
    return 0；
}
```

GPNEWTECH 汇众新特

位运算符

- 在C语言中，提供了对数据进行按位运算的位运算符，包括：~（按位取反）、<<（左移）、>>（右移）、&（按位与）、|（按位或）、^（按位异或）。
- 对于按位取反（~）、按位与（&）、按位或（|）和按位异或（^运算来说，格式为：变量1 按位运算符 变量2；这些运算规则遵守逻辑代数运算规律。在按位的逻辑运算中，数据类型不能是浮点数

逻辑变量x	逻辑变量y	~x	~y	x&y	x | y	x^y
0	0	1	1	0	0	0
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	1	0

GPNEWTECH 汇众新特

位运算符

【例10-10】逻辑按位运算的原理

- **假设两个变量int a=0x1234，int b=0x3456，则对于a&b来说按位与运算过程如下：**

```
  0001,0010,0011,0100
& 0011,0100,0101,0110
  0001,0000,0001,0100
```

结果为0x1014。

- **当按位或运算过程如下：**

```
  0001,0010,0011,0100
| 0011,0100,0101,0110
  0011,0110,0111,0110
```

结果为0x3676。

GPNEWTECH 汇众新特

位运算符

- **当按位异或运算过程如下：**

  0001,0010,0011,0100

  ^ 0011,0100,0101,0110

  0010,0110,0110,0010

结果为0x2662。

- **对于左移和右移运算来说，格式为：**

  变量 移位运算符 移位个数

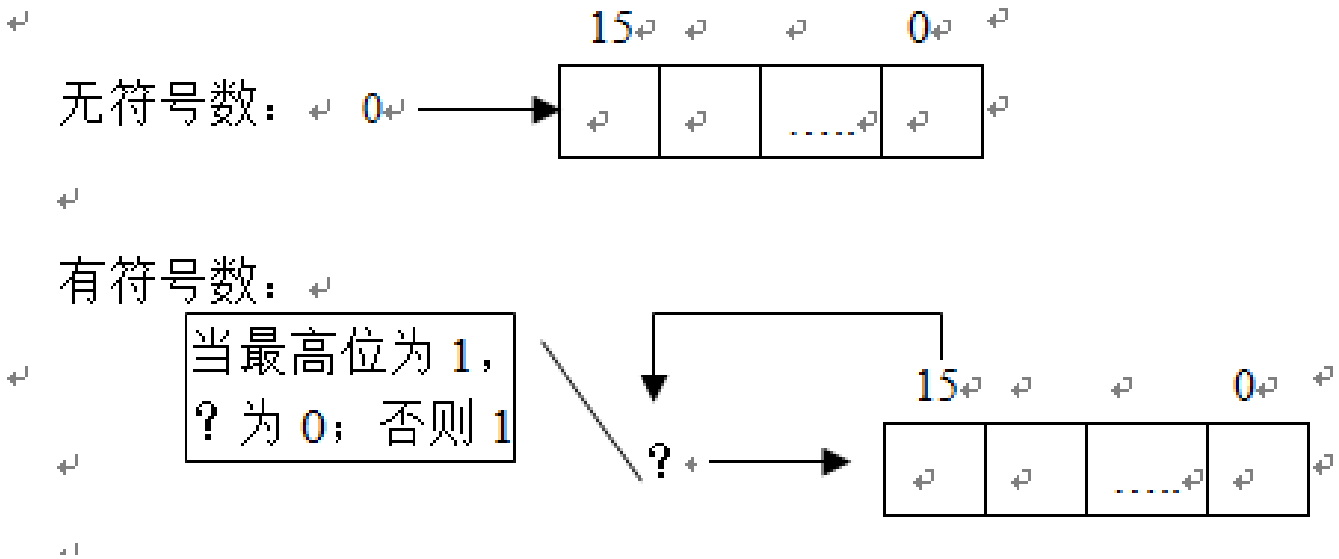
  - **在移位运算中，对于左移操作来说，在最右端（最低位）补0，下面以int型为例，说明左移的方法。**

15			**0** **0**
		…..	

GPNEWTECH 汇众新特

位运算符

- 对于右移操作来说，如果是无符号的数，则总是在最左端（最高位）补0；如果是有符号的数，如果符号位为1，则在最左端（最高位）补1；否则，在最左端（最高位）补0。



GPNEWTECH 汇众新特

位运算符

【例10-11】逻辑移位运算的原理

假设一个变量a=0x8123时，

- 当a<<3，表示执行左移三位的操作时，由于a=0x8123，表示成二进制数为a=1000,0001,0010,0011，当左移三位后，a=0000,1001,0001,1000值变为0x0918。
- 当a>>3，表示执行右移三位的操作时，区分有符号数和无符号数
  - 当a当作无符号数时，由于a=0x8123，表示成二进制数为a=1000,0001,0010,0011，当执行右移操作时，最左端补0，即当右移完三位后，a=0001,0000,0010,0100值变为0x1024。
  - 当a当作符号数时，由于a=0x8123，表示成二进制数为a=1000,0001,0010,0011，当执行右移操作时，最左端补1，即当右移完三位后，a=1111,0000,00010,0100值变为0xF024。

位运算符

【例10-12】位运算符的例子

代码清单 10-12 main.c文件

```
int main()
{
char a=30, b=55;      //定义字符型变量a和b，它们的初值为30和55
char c,d,e,f,g;       //定义字符型变量c、d、e、f和g
int i=-50,j=60;       //定义整型变量a和b，它们的初值为-50和60
int k,h,l,m,n;        //定义整型变量k、h、l和m
c=~a;                 //对变量a按位取反
```

GPNEWTECH 汇众新特

位运算符

```
d=a & b;          //对变量a和变量b按位与
e=a | b;          //对变量a和变量b按位或
f=a ^ b;          //对变量a和变量b按位异或
g=~(a ^ b);       //对于变量a和变量b按位异或非
h=~(a & b);       //对于变量a和变量b按位与非
k=i<<3;           //对于变量i左移3位
n=i>>3;           //对于变量i右移3位
l=j<<4;           //对于变量j左移4位
m=j>>4;           //对于变量j右移4位
    return 0;
}
```

GPNEWTECH 汇众新特

符合赋值运算符

在C语言中，提供了符合赋值运算符。符合赋值运算符是算术运算符、位运算符以及赋值运算符的组合。符合幅值运算符包括：+=（加法赋值）、 - =（减法赋值）、*=（乘法赋值）、/=（除法赋值）、%=（取模赋值）、<<=（左移赋值）、>>=（右移赋值）、&=（逻辑与赋值）、|=（逻辑或赋值）、^=（逻辑异或赋值）和~=（逻辑非赋值）。

符合赋值运算的格式如下：

变量 符合赋值运算符 表达式

在符合赋值表达式中，先进行表达式的运算操作，然后再执行赋值操作的过程。

- 注：在进行完符合赋值运算后，变量的值将发生变化。这点要特别的注意。

GPNEWTECH 汇众新特

符合赋值运算符

【例10-13】符合赋值运算符的例子

代码清单 10-13 main.c文件

```
int main()
{
int a=100,b=45;
int c=900,d=140,e=790,f=9900,g=-90,h=890;
int i=560,j=711;
a+=b;            //a=a+b;
c-=b;            //c=c-b;
d*=b;            //d=d*b
e/=b;            //e=e/b
```

GPNEWTECH 汇众新特

符合赋值运算符

```
f%=b;            //f=f%b
g<<=2;           //g=g<<2
h>>=3;           //h=h>>3
i&=b;            //i=i&b
j|=b;            //j=j|b
    return 0;
}
```

读者可以进入到本书所提供资料的stc_example\例子10-13目录下，在Keil μVision5集成开发环境下打开该设计，并进入调试器模式，使用单步运行。打开Watch 1窗口观察运算得到的结果。

GPNEWTECH 汇众新特

逗号运算符

在C语言中，提供了“，”运算符，将两个表达式连接在一起，称为逗号表达式。逗号表达式的格式为：

表达式1,表达式2,表达式3,….,表达式n

程序运行时，对于表达式的处理是从左到右依次计算出各个表达式的值，而整个逗号表达式的值是最右边的表达式（即表达式n）的值。

【例10-14】逗号运算符的例子1

代码清单10-14 main.c文件

```
int main()
{
int a,b,c,d,e,f;
d=(a=10,b=100,c=1000);
```

GPNEWTECH 汇众新特

逗号运算符

```
e=(a=100,a++,b=a,c=a*b);
f=(a=10,b=a*30,c=a+b);
return 0;
}
```

分析上面的设计代码：

- **对于d=(a=10,b=100,c=1000)来说，由于逗号表达式取最右侧表达式的值，所以d=c=1000。**

GPNEWTECH 汇众新特

逗号运算符

- 对于e=(a=100,a++,b=a,c=a*b)来说，由于逗号表达式是从左到右依次计算各个表达式的值，按下面顺序计算：
  - 得到a=100的值。
  - 计算a++，即a的值变为101。
  - b=a，即b的值为101。
  - c=a*b，即c=101×101，值为10201。
  - 将c的值赋值给e，值为10201。

GPNEWTECH 汇众新特

逗号运算符

- f=(a=10,b=a*30,c=a+b)来说，由于逗号表达式是从左到右依次计算各个表达式的值，按下面顺序计算：
  - 得到a=10的值。
  - b=a*30，得到b=10×30，值为300。
  - c=a+b，得到c=10+300=310。
  - 将c的值赋值给f，值为310。
- 读者可以进入到本书所提供资料的stc_program_example\例子10-14目录下，在Keil μVision5集成开发环境下打开该设计，查看运行结果。

GPNEWTECH 汇众新特

逗号运算符

【例10-15】逗号运算符的例子2

代码清单10-15 main.c文件

```
int main()
{
int a=-30,b,c,d,e,f;
d=(a*(b=a*30),c=a+b);
e=(c=50,c++,c++,++c);
f=(c=30,b=c*30,d=c+b);
return 0;
}
```

GPNEWTECH 汇众新特

条件运算符

在C语言中，提供了条件运算符“?:”，该运算符是C语言中唯一的三目运算符，即该运算符要求有三个运算的对象，用于将三个表达式连接在一起构成一个表达式。条件表达式的格式如下：

逻辑表达式 ? 表达式1: 表达式2

首先计算逻辑表达式的值，当逻辑表达式的值为1，即条件为真时，将表达式1的值作为整个条件表达式的值；当逻辑表达式的值为0，即条件为假时，将表达式2的值作为整个条件表达式的值。

GPNEWTECH 汇众新特

条件运算符

【例10-16】条件运算符的例子

代码清单10-16 main.c文件

```
void main()
{
int a=10,b=20,d,e;
d=(a==b) ? a:b;                //a==b条件不成立，因此将b的值赋值给d
e=(a!=b) ? a:b;               //a!=b条件不成立，因此将a的值赋值给e
}
```

GPNEWTECH 汇众新特

强制类型转换符

- 前面在介绍赋值操作和算术运算操作时，反复提到对于赋值操作和算术运算操作时，数据类型不相同的时候，C51编译器会根据转换规则自动的实现数据类型转换。对于一些程序员来说，如果他们不能很清楚知道转换规则，则很容易在数据类型的自动转换过程中出现错误的转换结果。在C语言中，除了允许不同数据类型之间自动转换外，提供了强制类型转换符，用于将一个数据类型强制转换成另一个数据类型，定义格式为：
- （数据类型关键字）表达式/变量
- 当表达式或者变量需要进行数据转换时，在其最前面显示说明需要最终转换的数据类型，并且用()将该显式声明的数据类型括起来。这样，C51编译器就能根据显式声明的数据类型关键字，将变量或者表达式转换为该数据类型。

强制类型转换符

【例10-17】类型转换符的例子

代码清单 10-17 main.c文件

```
void main()
{
int a=1000,b=2000;
  long int c,e;
  float d;
c=(long)a;
d=(float)b;
e=d/100;
}
```

GPNEWTECH 汇众新特

sizeof运算符

在C语言中，提供了求取数据类型、变量以及表达式最终计算结果字节个数的sizeof运算符。其格式为：

sizeof（表达式）

或

sizeof（数据类型）

- 注：sizeof是一种特殊的运算符，不是函数。在C51编译程序的时候，可以通过sizeof计算出字节数。
- 对于8051来说，char类型占用一个字节，int类型占用两个字节，long int占用四个字节，float占用四个字节。

GPNEWTECH 汇众新特

sizeof运算符

- 【例10-18】sizeof运算符的例子
- 代码清单 10-18 main.c文件

```
void main()
{
int a=10;
float b=10.0;
int c,d,e,f,g,h;
c=sizeof(int);
d=sizeof(char);
e=sizeof(unsigned int);
f=sizeof(long int);
g=sizeof(float);
h=sizeof(a+b);
}
```

GPNEWTECH 汇众新特