第十七节：加法运算的 5 种常用格式。

【17.1　单片机本身具备基础的数学算术能力。】

单片机本身是一个成品，本身就具备了基础的加减乘除能力，把单片机当做一个大人，我们需要做的只是沟通而已，叫他做加法他就做加法，叫他做减法就他就做减法，至于他是如何计算出来的不用管，“他”本身内部的电路结构就具备了这种基础运算的能力。人机沟通依然是用 C 语言，本节讲的加法运算，用的 C 语言符号跟我们日常用的数学加法符号是一样的，都是符号“+”。多说一句，单片机这种内置的基础运算能力并不是无限大的，而是数值不能超过某个范围，如果在加数或者运算结果的数值范围超过 4294967295 的情况下，要继续实现这类加法运算，这个就需要我们在单片机本身基础的运算能力上专门去编写一套大数据算法的程序才能实现，这个大家暂时不用深入理解，先学好当前基础再说。

【17.2　加法语法格式。】

加法语法格式：

“保存变量”=“加数 1”+“加数 2”+...+“加数 N”；

含义：右边的“加数”与“加数”相加（这里统一把平时所说的被加数也归类为加数），并且把最终的运算结果赋值给左边的“保存变量”。注意，这里的符号“=”不是等于号的意思，而是赋值的意思。左边的“保存变量”必须是变量，不能是常量，否则编译时会报错。而右边的“加数”既可以是变量，也可以是常量，也可以是“保存变量”本身自己。多说一句，什么是变量和什么是常量？变量就是可以在程序中被更改的，是分配的一个 RAM 空间。而常量往往就是常数值，或者是被分配在 ROM 空间的一个具体数值。下面根据右边“加数”与“加数”的不同组合，列出了加法运算的 5 种常用格式。

第 1 种：“加数 1”是常量，“加数 2”是常量。比如：

```
unsigned char a;
a=3+15;
```

分析：数字“3”和“15”都是常量。执行上述语句后，保存变量 a 变成了 18。

第 2 种：“加数 1”是变量，“加数 2”是常量。比如：

```
unsigned char b;
unsigned char x=10;
b=x+15;
```

分析：x 是变量，“15”是常量。由于原来 x 变量里面的数值是 10，执行上述语句后，保存变量 b 变成了 25。而变量 x 则保持不变，执行完所有语句后 x 还是 10。

第 3 种：“加数 1”是变量，“加数 2”是变量。比如：

```
unsigned char c;
unsigned char x=10;
unsigned char y=6;
c=x+y;
```

分析：x 是变量，y 也是变量。由于原来 x 变量里面的数值是 10，y 变量里面的数值是 6，执行上述语句后，保存变量 c 变成了 16。而变量 x 和 y 则保持不变，x 还是 10，y 还是 6。

第 4 种：“加数 1”是保存变量本身，“加数 2”是常量。比如：

```
unsigned char d=2;
```

```
d=d+18;
d=d+7;
```

分析：d 是保存变量本身，“18”是常量。这类语句有一个特点，具备了自加功能，可以更改自己本身的数值。比如原来保存变量 d 的数值是 2，执行“d=d+18;”语句后，d 变成了 20，接着再执行完“d=d+7;”语句后，d 最后变成了 27。

第 5 种：“加数 1”是保存变量本身，“加数 2”是变量。比如：

```
unsigned char e=2;
unsigned char x=10;
unsigned char y=6;
e=e+x;
e=e+y;
```

分析：e 是保存变量，x 与 y 都是变量。这类语句有一个特点，具备了自加功能，可以更改自己本身的数值。比如原来保存变量 e 的数值是 2，x 的数值是 10，执行“e=e+x;”语句后，e 变成了 12。由于 y 的数值是 6，接着再执行完“e=e+y;”语句后，所以 e 最后变成了 18。

【17.3　例程练习和分析。】

现在我们编写一个程序来验证上面讲到的 5 个加法例子：

程序代码如下：

```
/*---C 语言学习区域的开始。-----------------------------------------------*/

void main() //主函数
{
    unsigned char a;      //定义一个变量 a，并且分配了 1 个字节的 RAM 空间。
    unsigned char b;      //定义一个变量 b，并且分配了 1 个字节的 RAM 空间。
    unsigned char c;      //定义一个变量 c，并且分配了 1 个字节的 RAM 空间。
    unsigned char d=2;    //定义一个变量 d，并且分配了 1 个字节的 RAM 空间。初始化默认为 2.
    unsigned char e=2;    //定义一个变量 e，并且分配了 1 个字节的 RAM 空间。初始化默认为 2.

    unsigned char x=10;   //定义一个变量 x，并且分配了 1 个字节的 RAM 空间。初始化默认为 10.
    unsigned char y=6;    //定义一个变量 y，并且分配了 1 个字节的 RAM 空间。初始化默认为 6.

    //第 1 种：“加数 1”是常量，“加数 2”是常量。
    a=3+15;

    //第 2 种：“加数 1”是变量，“加数 2”是常量。
    b=x+15;

    //第 3 种：“加数 1”是变量，“加数 2”是变量。
    c=x+y;
```

```
        //第 4 种：“加数 1”是保存变量本身，“加数 2”是常量。
        d=d+18;
        d=d+7;

        //第 5 种：“加数 1”是保存变量本身，“加数 2”是变量。
        e=e+x;
        e=e+y;

        View(a);   //把第 1 个数 a 发送到电脑端的串口助手软件上观察。
        View(b);   //把第 2 个数 b 发送到电脑端的串口助手软件上观察。
        View(c);   //把第 3 个数 c 发送到电脑端的串口助手软件上观察。
        View(d);   //把第 4 个数 d 发送到电脑端的串口助手软件上观察。
        View(e);   //把第 5 个数 e 发送到电脑端的串口助手软件上观察。

        while(1)
        {
        }
    }

    /*---C 语言学习区域的结束。-----------------------------------------------*/
```

在电脑串口助手软件上观察到的程序执行现象如下：

```
开始...

第 1 个数
十进制:18
十六进制:12
二进制:10010

第 2 个数
十进制:25
十六进制:19
二进制:11001

第 3 个数
十进制:16
十六进制:10
二进制:10000

第 4 个数
十进制:27
十六进制:1B
二进制:11011
```

```
第 5 个数
十进制:18
十六进制:12
二进制:10010
```

分析：

通过实验结果，发现在单片机上的计算结果和我们的分析是一致的。

【17.4　如何在单片机上练习本章节 C 语言程序？】

直接复制前面章节中第十一节的模板程序，练习代码时只需要更改“C 语言学习区域”的代码就可以了，其它部分的代码不要动。编译后，把程序下载进带串口的 51 学习板，通过电脑端的串口助手软件就可以观察到不同的变量数值，详细方法请看第十一节内容。