第七章 定时器/计数器的工作原理和应用

学习目的及要求

- 熟悉89S51片内两个16位定时器/计数器T0和T1的硬件结构及其与CPU的关系;
- 掌握T0和T1的两种工作方式，即计数方式与定时方式，四种工作模式(即计数器长度);
- 牢记TMOD和TCON各位的含意,学会定时器控制及应用方法;掌握定时器的四种模式的应用。

两个可编程的定时器/计数器T1、T0。

2种工作模式：

（1）计数器工作模式

（2）定时器工作模式

4种工作方式(方式0-方式3)。

7.1 定时器/计数器的结构

TMOD：选择定时器/计数器T0、T1的工作模式和工作方式。

TCON：控制T0、T1的启动和停止计数，同时包含了T0、T1的状态。

7．1 定时器/计数器的结构

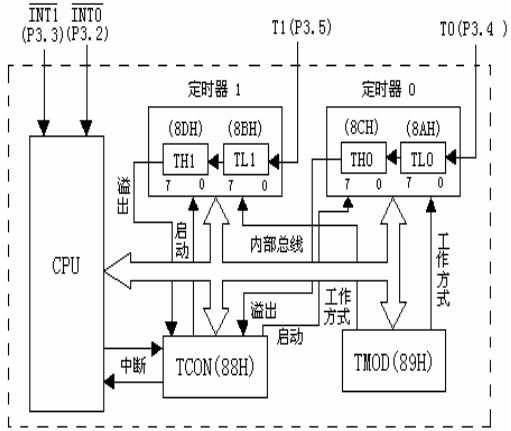
Timer/ Counter

89S51单片机内部有二个16位的可编程的定时器/计数器：

定时器/计数器0（T/C0）

定时器/计数器1（T/C1）

它们都有定时和对外部事件计数的功能，可用于定时控制、延时、对外部事件检测和计数等场合.



T/C0和T/C1的结构及与CPU的关系如图7-1所示。

定时器/计数器（T/C0和T/C1）的组成

T/C0和T/C1都是16位的加1计数器

结构组成：

- T/C0由两个8位的TH0（8CH）和TL0（8AH）组成
- T/C1由TH1（8DH）和TL1（8BH）组成

软件设置：

- 计数初值： 对TH1、TL1 、TH0、和TL0的初始化编程
- 工作方式： 通过TMOD和 TCON对每个T/C设置定时或计数

特殊功能寄存器 TMOD(Timer Mode Register)
TCON(Timer Control Register)
THx 存放计数初值的高8位
TLx 存放计数初值的低8位

两个可编程的定时器/计数器T1、T0。

2种工作模式：

（1）计数器工作模式

（2）定时器工作模式

4种工作方式(方式0-方式3)。

7.1 定时器/计数器的结构

TMOD：选择定时器/计数器T0、T1的工作模式和工作方式。

TCON：控制T0、T1的启动和停止计数，同时包含了T0、T1的状态。

7.1 定时器/计数器的结构与基本概念

1. 计数概念的引入

从选票的统计谈起：画“正”。这就是计数。
生活中计数的例子处处可见。例：录音机上的计数器、家里用的电度表、汽车上的里程表等等.

7.1 定时器/计数器的结构与基本概念

2. 计数器的容量

- 从一个生活中的例子看起：一个水盆在水龙头下，水龙没关紧，水一滴滴地滴入盆中。水滴不断落下，盆的容量是有限的，过一段时间之后，水就会逐渐变满。
- 电表上的计数器最多只计到9999….那么单片机中的计数器有多大的容量呢？
- 8031单片机中有两个计数器，分别称之为T0和T1，这两个计数器分别是由两个8位的RAM单元组成的，即每个计数器都是16位的计数器，最大的计数量是65536。

7.1 定时器/计数器的结构与基本概念

3. 定时的概念-1

8031中的计数器除了可以作为计数之用外，还可以用作时钟，时钟的用途当然很大，如打铃器，电视机定时关机，空调定时开关等等，那么计数器是如何作为定时器来用的呢？

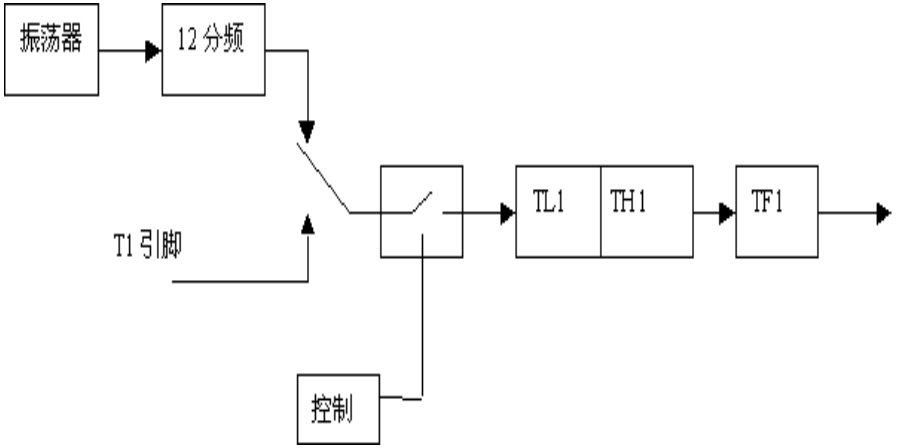
一个闹钟，我将它定时在1个小时后闹响，换言之，也可以说是秒针走了（3600）次，所以时间就转化为秒针走的次数的，也就是计数的次数了，可见，计数的次数和时间之间的确十分相关。那么它们的关系是什么呢？那就是秒针每一次走动的时间正好是1秒。

3. 定时的概念-2

- 只要计数脉冲的间隔相等，则计数值就代表了时间的流逝。
- 由此，单片机中的定时器和计数器是同一个事物，只不过计数器是记录的外界发生的事情，而定时器则是由单片机提供一个非常稳定的计数源。
- 那么提供组定时器的是计数源是什么呢？看图1，原来就是由单片机的晶振经过12分频后获得的一个脉冲源。晶振的频率当然很准，所以这个计数脉冲的时间间隔也很准。
- 问题：一个12M的晶振，它提供给计数器的脉冲时间间隔是多少呢？当然这很容易，就是12M/12等于1M，也就是1个微秒。
- 结论：计数脉冲的间隔与晶振有关，12M的晶振，计数脉冲的间隔是1微秒。

3. 定时的概念-3

图1 由单片机的晶振经过12分频后获得的一个脉冲源。



7.1 定时器/计数器的结构与基本概念

4. 溢出的概念

让我们再来看水滴的例子，当水不断落下，盆中的水不断变满，最终有一滴水使得盆中的水满了。这时如果再有一滴水落下，就会发生什么现象？水会漫出来，用个术语来讲就是“溢出”。

水溢出是流到地上，而计数器溢出后将使得TF0变为“1”。至于TF0是什么我们稍后再谈。一旦TF0由0变成1，就是产生了变化，产生了变化就会引发事件，就象定时的时间一到，闹钟就会响一样。现在我们来研究另一个问题：要有多少个计数脉冲才会使TF0由0变为1。

5. 任意定时及计数的方法

刚才已讲过，计数器的容量是16位，也就是最大的计数值到65536，因此计数计到65536就会产生溢出。这个没有问题，问题是我们现实生活中，经常会有少于65536个计数值的要求，如包装线上，一打为12瓶，一瓶药片为100粒，怎么样来满足这个要求呢？

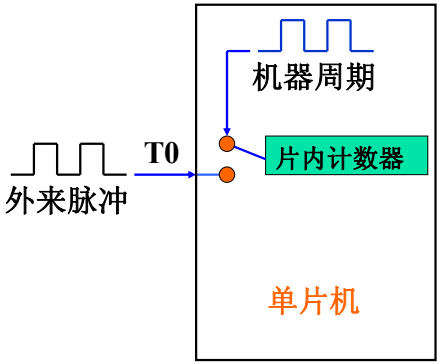
提示：如果是一个空的盆要1万滴水滴进去才会满，我在开始滴水之前就先放入一勺水，还需要10000滴嘛？
我们采用预置数的方法，我要计100，那我就先放进65436，再来100个脉冲，不就到了65536了吗。
定时也是如此，每个脉冲是1微秒，则计满65536个脉冲需时65.536毫秒，但现在我只要10毫秒就可以了，怎么办？
……
10个毫秒为10000个微秒，所以，只要在计数器里面放进55536就可以了。

8051内部设有两个16位的定时器/计数器，可用软件控制。

定时器:对机器周期计数，每过一个机器周期，计数器内容加1；

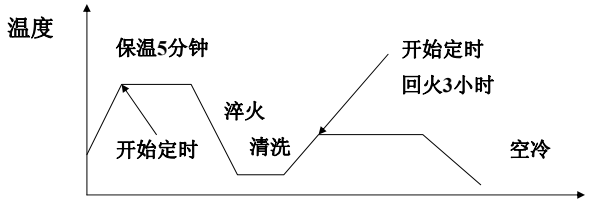
计数器：对外来脉冲进行计数，

T0、T1引脚上从高电平到低电平跳变时，计数器内容加1



定时的应用示例

例如某机械零件的热处理工艺曲线为：

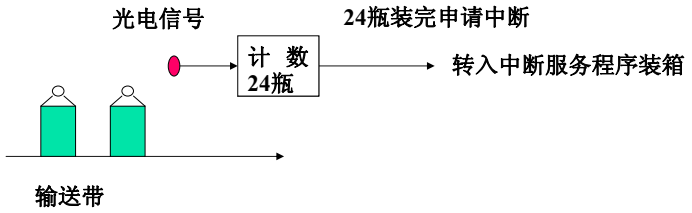


实际控制可以由单片机定时发出信号控制自动完成整个工艺过程。

计数的应用示例

计数功能：对外界发生的事件计数，当达到程序规定的计数值时，输出一脉冲信号，申请中断。

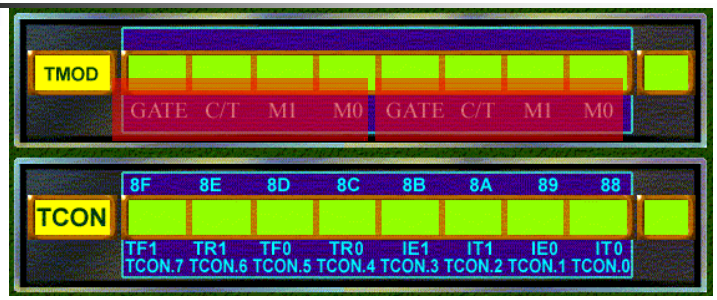
例如一啤酒生产线，如下图所示



6. 定时/计数器的方式控制字-1

单片机中的定时/计数器都可以有多种用途，那么我怎样才能让它们工作于我所需要的用途呢？这就要通过定时/计数器的方式控制字来设置。

在单片机中有两个特殊功能寄存器与定时/计数有关，这就是TMOD和TCON。TMOD和TCON是名称，我们在写程序时就可以直接用这个名称来指定它们，当然也可以直接用它们的地址89H和88H来指定它们。



TMOD：选择定时器/计数器T0、T1的工作模式和工作方式。

TCON：控制T0、T1的启动和停止计数，同时包含了T0、T1的状态。

单片机复位时，两个寄存器的所有位都被清0。

7.1.1 工作方式控制寄存器TMOD

D7	D6	D5	D4	D3	D2	D1	D0	
GATE	C/T	M1	M0	GATE	C/T	M1	M0	(89H)
T1 方式字段				T0 方式字段				

8位分为两组，高4位控制T1，低4位控制T0。

（1）GATE——门控位

0：以TRX（X=0,1）来启动定时器/计数器运行。

1：用外中断引脚(INT0*或INT1*)上的高电平和TRX来启动定时器/计数器运行。

7.1.1 工作方式控制寄存器TMOD

D7	D6	D5	D4	D3	D2	D1	D0	
GATE	C/T	M1	M0	GATE	C/T	M1	M0	(89H)
T1 方式字段				T0 方式字段				

8位分为两组，高4位控制T1，低4位控制T0。

（2）M1、M0——工作方式选择位

M1	M0	工 作 方 式
0	0	方式0，13位定时器/计数器。
0	1	方式1，16位定时器/计数器。
1	0	方式2，8位常数自动重新装载
1	1	方式3，仅适用于T0，T0分成两个8位计数器，T1停止计数。

7.1.1 工作方式控制寄存器TMOD

D7	D6	D5	D4	D3	D2	D1	D0	
GATE	C/T	M1	M0	GATE	C/T	M1	M0	(89H)
T1 方式字段				T0 方式字段				

8位分为两组，高4位控制T1，低4位控制T0。

(3) C/T*——计数器模式和定时器模式选择位

0：定时器模式，对系统时钟12分频后的脉冲进行计数。

1：计数器模式，计数器对外部输入引脚T0（P3.4）或T1（P3.5）的外部脉冲（负跳变）计数。

7.1.2 定时器/计数器控制寄存器TCON

	D7	D6	D5	D4	D3	D2	D1	D0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
位地址	8F	8E	8D	8C	8B	8A	89	88
	T1		T0		外中断			

低4位与外部中断有关，已介绍。高4位的功能如下：

(1) TF1、TF0——计数溢出标志位

当定时器溢出时，硬件电路置TFx为“1”，响应中断时硬件自动清零TFx。

(2) TR1、TR0——计数运行控制位

1：启动定时器/计数器工作

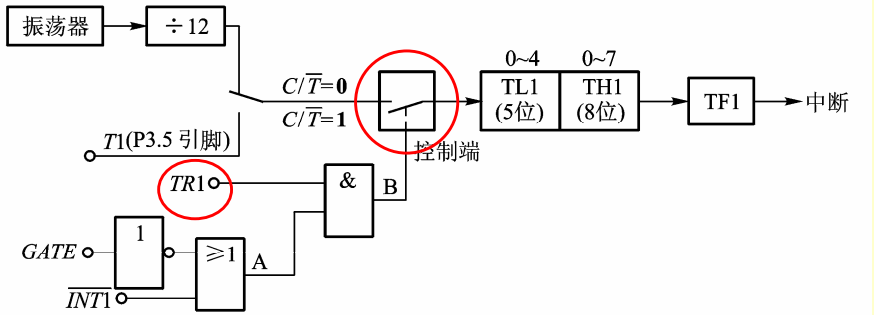
0：停止定时器/计数器工作

7.2 定时器/计数器的4种工作方式

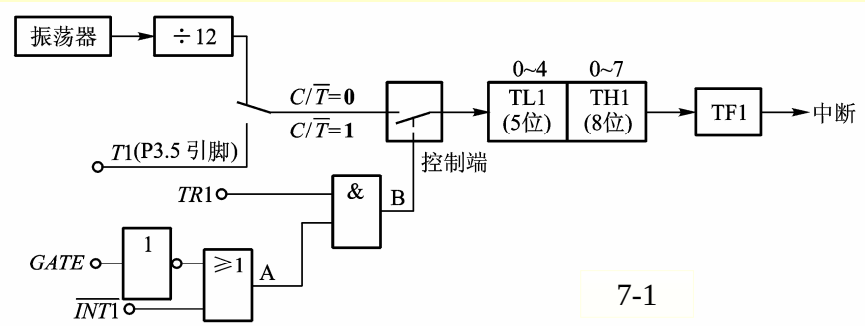
7.2.1 方式0

M1、M0为00 ，方式0，13位计数器。

D12	D5 D4	D0
THx 高8位	TLx 低5位	



7-1

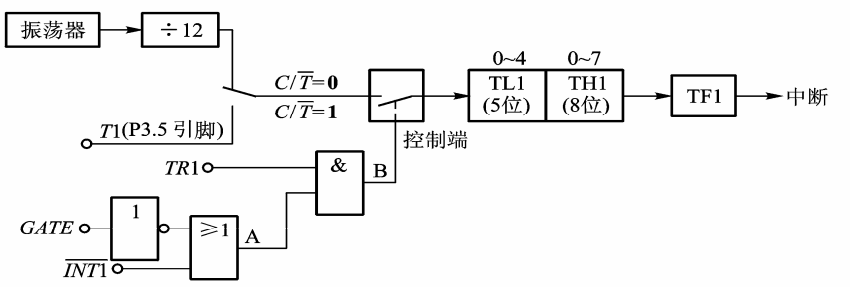


7-1

C/T* 位决定工作模式：

0：开关打在上面，为定时器工作模式；

1：开关打在下面，为计数器工作模式，计数脉冲为P3.4、P3.5引脚上的外部输入脉冲，当引脚上发生负跳变时，计数器加1。



7-1

GATE位：决定定时器/计数器的运行取决于TRx一个条件还是TRx和INTx*引脚两个条件。

（1）0：A点（见图6-2）是否计数，仅取决于TRx的状态。

（2）1：B点电位由INTX*的输入电平和TRX的状态这两个条件来确定。是否计数是由TRx和INTx*二个条件来控制的。

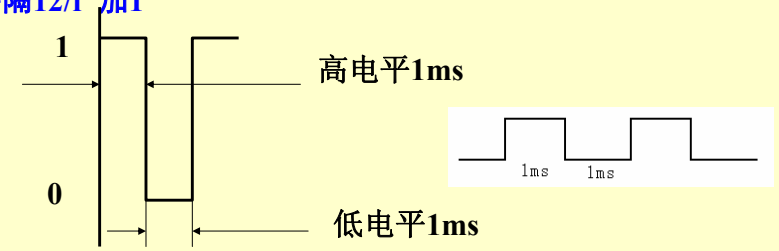
时间常数计算公式为：定时：　$(2^{13}-X)\times 12/f$ =定时间隔

计数：　$2^{13}-X$ = 计数次数

例如：要求从P1.0脚发出周期为2ms的方波只要选T0定时间隔1ms求反P1.0即可满足。

定时工作方式中令计数器加1的周期是每个机器周期加1，

即每隔12/f 加1



设定时器时间常数为X

则有

$(2^{13}-X)\times(12/f)=1ms$

设f=6MHZ

$(2^{13}-X)\times(12/6\times10^{6})=1\times10^{-3}$

X=8192 - 500=7692

7692=1E0CH　　00011110 00001100B

11110000　　00001100

TH0=F0H　　TL0=0CH

即共加500次，每次耗费2μs，共耗费时间1ms

②讨论计数方式：

例如前述的啤酒生产线，计数24瓶中断转入装箱程序。

选T1方式0计数，TMOD的高4位为：0 1 0 0

初始常数X的计算:

$2^{13}-X=24$

X=8192 - 24=8168

8168=1FE8H　　00011111 11101000B

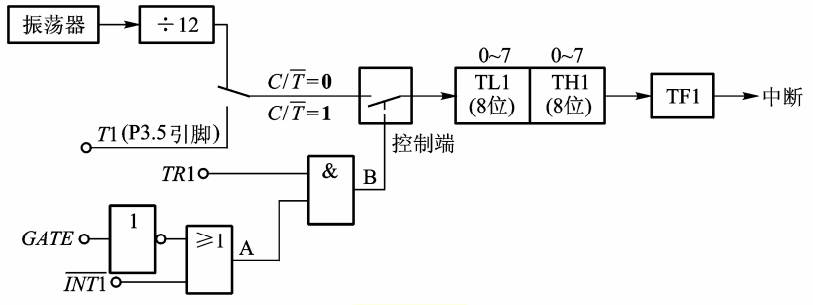
11111111　　00001000

TH1=FFH　　TL1=08H

加24次即溢出中断。

7.2.2 方式1

M1 M0=0 1，方式1, 16位定时器/计数器，其它的同方式0。方式1和方式0的差别仅仅在于计数器的位数不同.



7-2

7.2.2 方式1

方式1与方式0唯一的区别是计数器是16位,即TL 8位,TH 8位,因此,计算时间常数的公式中 2^{13} 应改为2^{16},上述程序如改用方式1,则常数计算为:

定时：$(2^{16}-X)\times(12/f)=1\times10^{-3}$

解之X =FE0CH 即TH0=FEH ,TL0=0CH

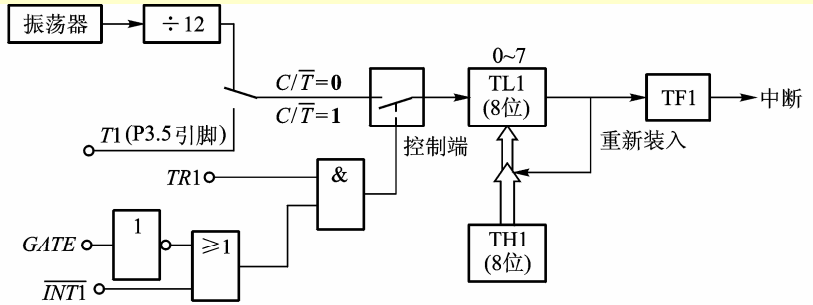
计数：$2^{16}-X=24$

解之 X=FFE8H 即TH1=FFH,TL0=E8H

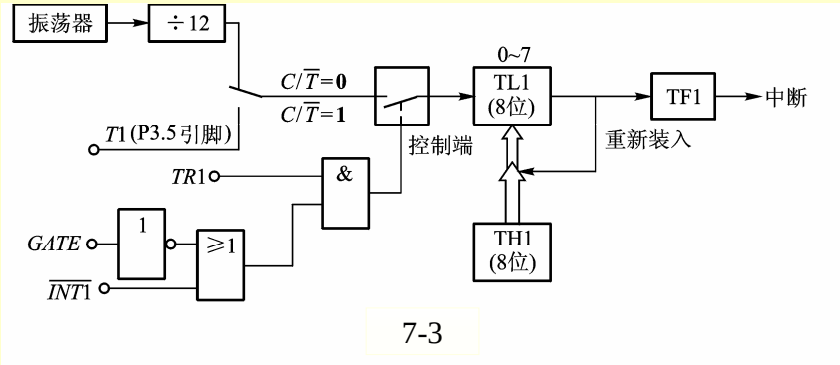
程序中的TMOD赋值相应改为方式1

7.2.3 方式2

M1、M0为10，方式2，8位计数器,计数满后自动装入计数初值,定时准确。

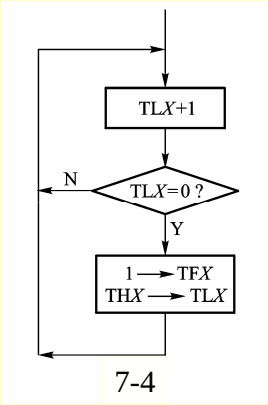


7-3



7-3

TLX作为常数缓冲器，当TLX计数溢出时，在置“1”溢出标志TFX的同时，还自动的将THX中的初值送至TLX，使TLX从初值开始重新计数。定时器/计数器的方式2工作过程如图7-4(X=0,1)。



7-4

省去用户软件中重装初值的程序，精确的定时。

7.2.4 方式3

增加一个附加的8位定时器/计数器，从而具有3个定时器/计数器。

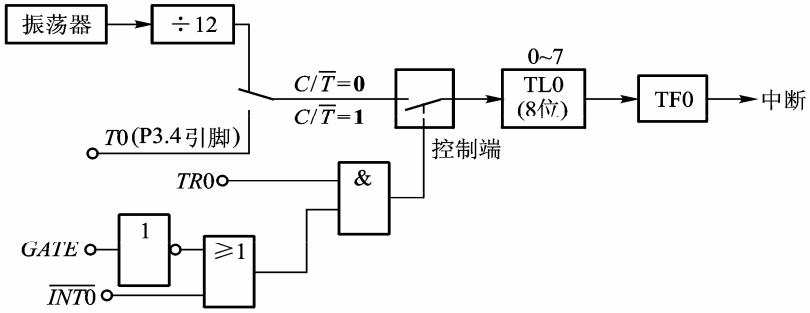
只适用于定时器/计数器T0。T1不能工作在方式3，

T1方式3时相当于TR1=0，停止计数（此时T1可用来作串行口波特率产生器）。

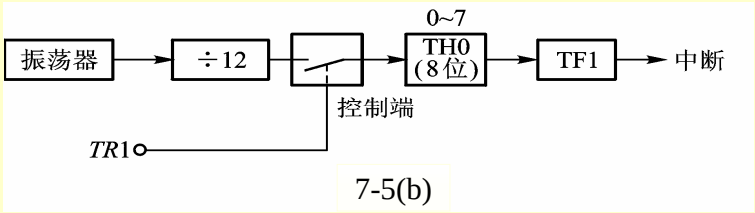
1．工作方式3下的T0

T0分为两个独立的8位计数器:TL0和TH0 。TL0使用T0的状态控制位C/T*、GATE、TR0、，而TH0被固定为一个8位定时器（不能作外部计数模式），并使用定时器T1的状态控制位TR1和TF1，同时占用定时器T1的中断请求源TF1。

各引脚与T0的逻辑关系如图所示：



7-5(a)

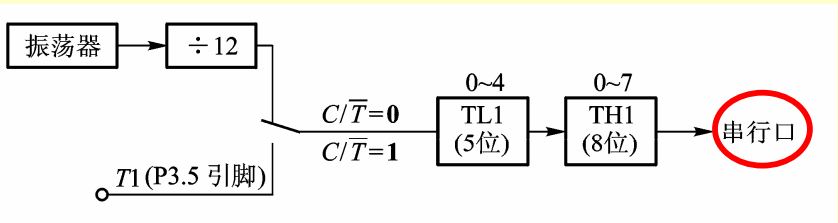


7-5(b)

2．T0工作在方式3下T1的各种工作方式

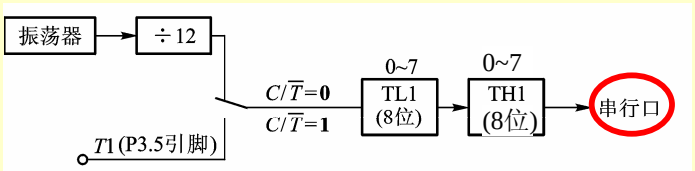
当T1用作串行口的波特率发生器时， T0才工作在方式3。 T0处于方式3时， T1可定为方式0、方式1和方式2, 用来作为串行口的波特率发生器，或不需要中断的场合。

（1）T1工作在方式0



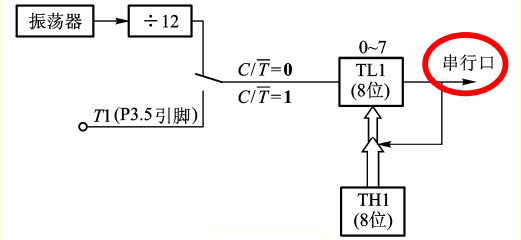
7-6

（2） T1工作在方式1



7-7

（3） T1工作在方式2



7-8

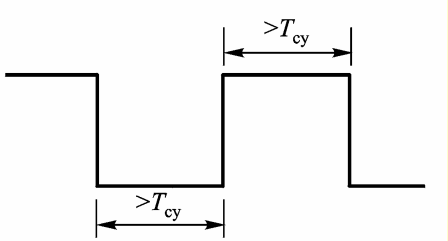
7.3 定时器/计数器对输入信号的要求

外部计数脉冲的最高频率为系统振荡器频率的1/24，

例如选用12MHz频率的晶体，则可输入500KHz的外部脉冲。

输入信号的高、低电平至少要保持一个机器周期。

如图7-9所示，图中T_{cy}为机器周期



7-9

7.4 定时器/计数器的编程和应用

4种工作方式中，方式0与方式1基本相同，由于方式0是为兼容MCS-48而设，初值计算复杂，在实际应用中，一般不用方式0，而采用方式1。

7.4.1 P1口控制8只LED每0.5s闪亮一次

例7-1 在AT89S51的P1口上接有8只LED，原理电路见图7-10。采用T0方式1的定时中断方式，使P1口外接的8只LED每0.5s闪亮一次。

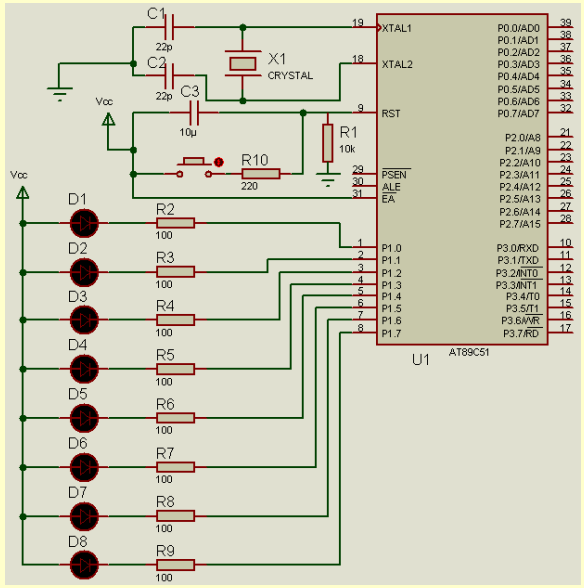


图7-10 方式1定时中断控制LED闪亮

（1）设置TMOD寄存器

T0工作在方式1，应使TMOD寄存器的M1、M0=01；应设置C/T*=0，为定时器模式；对T0的运行控制仅由TR0来控制，应使相应的GATE位为0。定时器T1不使用，各相关位均设为0。所以，TMOD寄存器应初始化为0x01。

（2）计算定时器T0的计数初值

设定时时间5ms（即5 000μs），设T0计数初值为X，假设晶振的频率为11.059 2MHz，则定时时间为：

定时时间=(2^{16}–X)×12/晶振频率

则 5 000=(2^{16} –X) ×12/11.059 2

得 X = 60 928

转换成十六进制：0xee00，其中0xee装入TH0，0x00装入TL0。

（3）设置IE寄存器

本例由于采用定时器T0中断，因此需将IE寄存器中的EA、ET0位置1。

（4）启动和停止定时器T0

将定时器控制寄存器TCON中的TR0=1，则启动定时器T0；TR0=0，则停止定时器T0定时。

参考程序：

```
#include<reg51.h>
char  i=100;
void main ()
{
        TMOD=0x01;                  //定时器T0为方式1
        TH0=0xee;                   //设置定时器初值
        TL0=0x00;
    P1=0x00;                  //P1口8个LED点亮
        EA=1;                       //总中断开
        ET0=1;                      //开T0中断
        TR0=1;                      //启动T0
        while(1);                   //循环等待
        {;}
```



```
}
void timer0() interrupt 1           //T0中断程序
{
    TH0=0xee;                       //重新赋初值
    TL0=0x00;
    i--;                            //循环次数减1
    if(i<=0)
    {
        P1=~P1;                     //P1口按位取反
        i=100;                      //重置循环次数
    }
}
```

7.4.2 计数器的应用

【例7-2】如图7-11，T1采用计数模式，方式1中断，计数输入引脚T1（P3.5）上外接按钮开关，作为计数信号输入。按4次按钮开关后，P1口的8只LED闪烁不停。

（1）设置TMOD寄存器

T1工作在方式1，应使TMOD的M1、M0=01；设置C/T*=1，为计数器模式；对T0运行控制仅由TR0来控制，应使GATE0=0。定时器T0不使用，各相关位均设为0。所以，TMOD寄存器应初始化为0x50。

（2）计算定时器T1的计数初值

由于每按1次按钮开关，计数1次，按4次后，P1口的8只LED闪烁不停。因此计数器初值为65 536-4=65 532，将其转换成十六进制后为0xfffc，所以，

TH0=0xff，TL0=0xfc。

（3）设置IE寄存器

本例由于采用T1中断，因此需将IE寄存器的EA、ET1位置1。

（4）启动和停止定时器T1

将寄存器TCON中TR1=1，则启动T1计数；TR1=0，则停止T1计数。

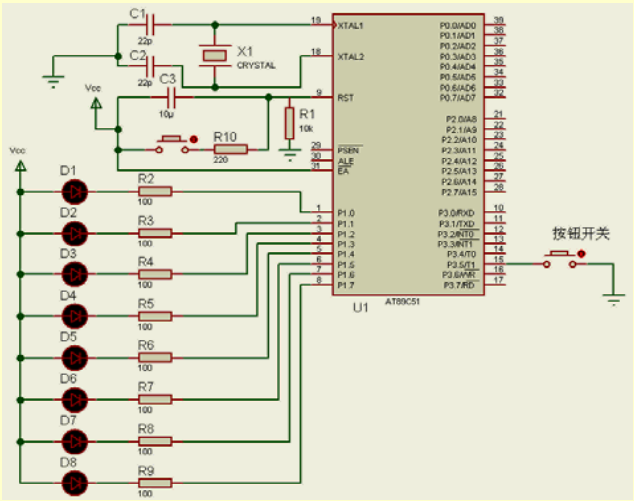


图7-11 由外部计数输入信号控制LED的闪烁

参考程序如下：

```
#include <reg51.h>
  void Delay(unsigned int i)          //定义延时函数Delay( )，i是
                                      形式参数，不赋初值
{
    unsigned int j;
    for(;i>0;i--)                     //变量i由实际参数传入一个值
                                      //因此i不能赋初值
    for(j=0;j<125;j++)
    {;}                               //空函数
}
```

```
void  main( )            //主函数
{
    TMOD=0x50;           //设置定时器T1为方式1计数
    TH1=0xff;            //向TH1写入初值的高8位
    TL1=0xfc;            //向TL1写入初值的低8位
    EA=1;                //总中断允许
     ET1=1;              //定时器T1中断允许
    TR1=1;               //启动定时器T1
    while(1) ;           //无穷循环，等待计数中断
}
```



```
void T1_int(void)  interrupt 3   //T1中断函数
{
    for(;;)                  //无限循环
    {
          P1=0xff;           //8位LED全灭
          Delay(500) ;       //延时500ms
          P1=0;              //8位LED全亮
          Delay(500);        //延时500ms
    }
}
```


7.4.3 控制P1.0产生周期为2ms的方波

【例7-3】假设系统时钟为12MHz，设计电路并编写程序实现从P1.0引脚上输出一个周期为2ms的方波，见图7-12。

要在P1.0上产生周期为2ms的方波，定时器应产生1ms的定时中断，定时时间到则在中断服务程序中对P1.0求反。使用定时器T0，方式1定时中断，GATE不起作用。

本例的原理电路见图7-13。其中在P1.0引脚接有虚拟示波器，用来观察产生的周期2ms的方波。

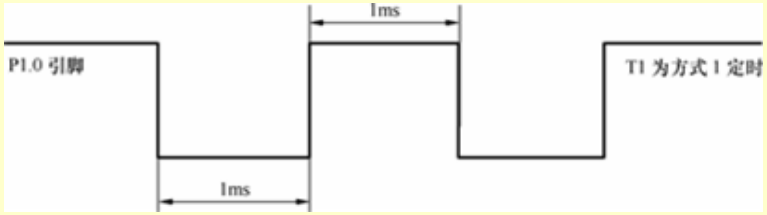


图7-12 定时器控制P1.0输出一个周期2ms方波

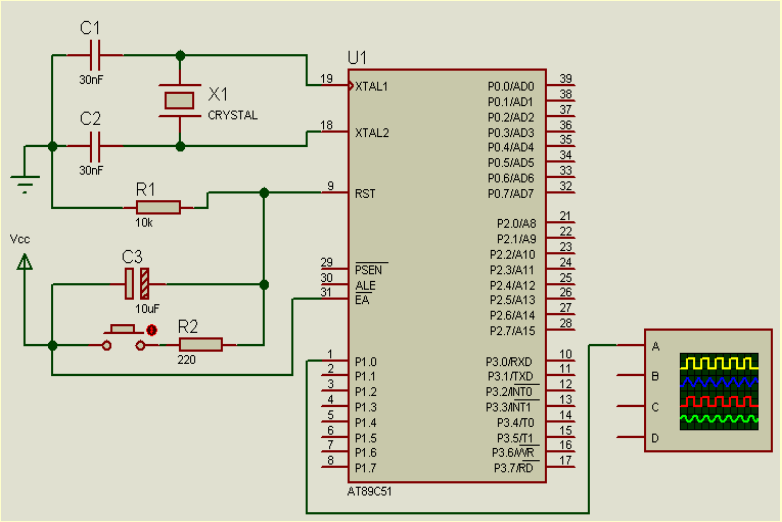


图7-13 定时器控制P1.0输出周期2ms的方波的原理电路

下面来计算T0初值X：

设T0的初值为X，有

$$(2^{16}-X)\times 1\times 10^{-6}=1\times 10^{-3}$$

即 $65536-X=1000$

得X=64536，化为16进制数就是0xfc18。将高8位0xfc装入TH0，低8位0x18装入TL0。

参考程序如下：

```
#include <reg51.h>          //头文件reg51.h
sbit P1_0=P1^0;             //定义特殊功能寄存器P1的位变量
void main(void)             //主程序
{
    TMOD=0x01;              //设置T0为方式1
    TR0=1;                  //接通T0
    while(1)                //无限循环
```



```
{
  TH0=0xfc;          //置T0高8位初值
    TL0=0x18;          //置T0低8位初值
      do{}while(!TF0);     //TF0为0原地循环，为1则T0溢出，往下执行
    P1_0=!P1_0;        // P1.0状态求反
    TF0=0;             //TF0标志清零
    }
}
```

仿真时，右键单击虚拟数字示波器，出现下拉菜单，点击“Digital oscilloscope”选项，就会在数字示波器上显示P1.0引脚输出周期为2ms方波，如图7-14所示。

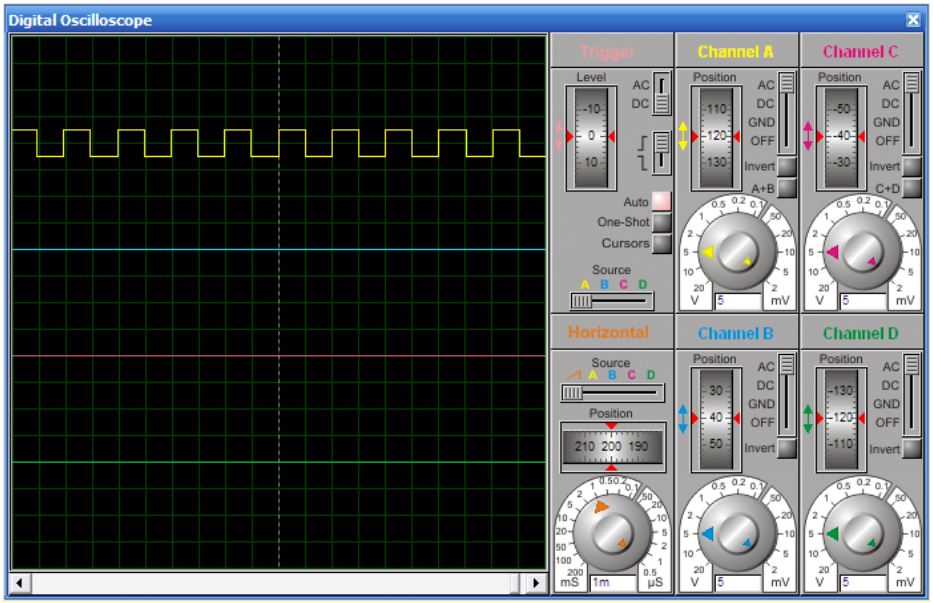


图7-14 虚拟数字示波器显示的2ms的方波波形

【例7-4】用2位数码管显示计时时间，最小计时单位为“百毫秒”，计时范围0.1～9.9s。当第1次按一下计时功能键时，秒表开始计时并显示；第2次按一下计时功能键时，停止计时，将计时的时间值送到数码管显示；如果计时到9.9s，将重新开始从0计时；第3次按一下计时功能键，秒表清0。再次按一下计时功能键，则重复上述计时过程。

本秒表应用定时器模式，计时范围0.1～9.9s。此外还涉及如何编写控制LED数码管显示的程序。原理电路见图7-15。

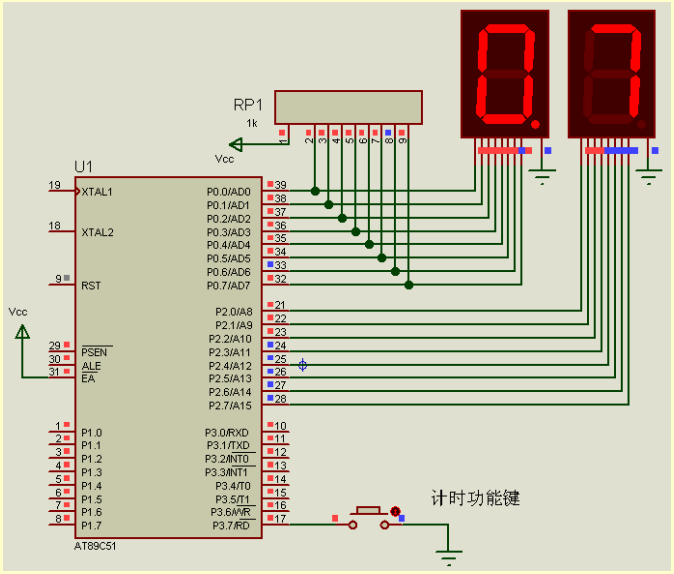


图7-15 LED数码管显示的秒表原理电路及仿真

参考程序如下：

```
#include<reg51.h>                          //头文件
unsigned char code discode1[]=
        {0xbf,0x86,0xdb,0xcf,0xe6,0xed,0xfd,0x87,0xff,0xef};
                //数码管显示0～9的段码表,带小数点
unsigned char code discode2[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};

                //数码管显示0～9的段码表，不带小数点
unsigned char timer=0;              //timer记录中断次数
unsigned char second;               //second储存秒
unsigned char key=0;                //key记录按键次数
main()                              //主函数
{
        TMOD=0x01;                  //定时器T0方式1定时
        ET0=1;                      //允许定时器T0中断
```

```
EA=1;                          //总中断允许
second=0;                      //设初始值
P0=discode1[second/10];        //显示秒位0
P2=discode2[second%10];        //显示0.1s位0
while(1)                        //循环
{
        if((P3&0x80)==0x00)    //当按键被按下时
        {
                key++;         //按键次数加1
                switch(key)    //根据按键次数分三种情况
                {
                case 1:        //第一次按下为启动秒表计时
                TH0=0xee;      //向TH0写入初值的高8位
```



```
                TL0=0x00;      //向TL0写入初值的低8位，定时5ms
                TR0=1;         //启动定时器T0
                break;
                case 2:        //按下两次暂定秒表
                TR0=0;         //关闭定时器T0
                break;
                case 3:        //按下3次秒表清0
                key=0;         //按键次数清
                second=0;      //秒表清0
                P0=discode1[second/10];  //显示秒位0
                P2=discode2[second%10];  //显示0.1s位0
                break;
        }
while((P3&0x80)==0x00);        //如果按键时间过长在此循环
}
```



```
        }
}
void int_T0() interrupt 1  using 0      //定时器T0中断函数
{
        TR0=0;                  //停止计时，执行以下操作（会带来计时误差）
        TH0=0xee;               //向TH0写入初值的高8位
        TL0=0x00;               //向TL0写入初值的低8位，定时5ms
        timer++;                //记录中断次数
        if (timer==20)          //中断20次，共计时20*5ms=100ms=0.1s
        {
                timer=0;        //中断次数清0
                second++;       //加0.1s
                P0=discode1[second/10];    //根据计时，即时显示秒位
        P2=discode2[second%10]; //根据计时，即时显示0.1s位          }
```



```
if(second==99)                  //当计时到9.9s时
{
        TR0=0;                  //停止计时
        second=0;               //秒数清0
        key=2;                  //按键数置2，当再次按下按键时，
                                //key++,即key=3，秒表清0复原
}
else                            //计时不到9.9s时
{
        TR0=1;                  //启动定时器继续计时
}
}
```

7.4.4 门控制位GATE的应用—测量脉冲宽度

GATE1可使定时器/计数器T1的启动计数受INT1*的控制，可测量引脚INT1*（P3.3）上正脉冲的宽度（机器周期数）。

【例7-6】门控位GATE1可使T1启动计数受INT1*控制，当GATE1=1，TR1=1时，只有INT1*引脚输入高电平时，T1才被允许计数。利用该功能，可测量INT1*脚正脉冲宽度，方法见图7-16。

原理电路见图7-17，图中省略复位电路和时钟电路。利用门控位GATE1来测量INT1*脚上正脉冲宽度，并在6位数码管上以机器周期数显示。对被测量脉冲信号宽度，要求能通过旋转信号源旋钮可调。

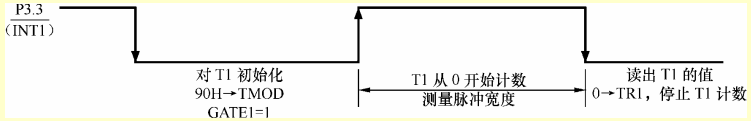


图7-16 利用GATE位测量正脉冲的宽度

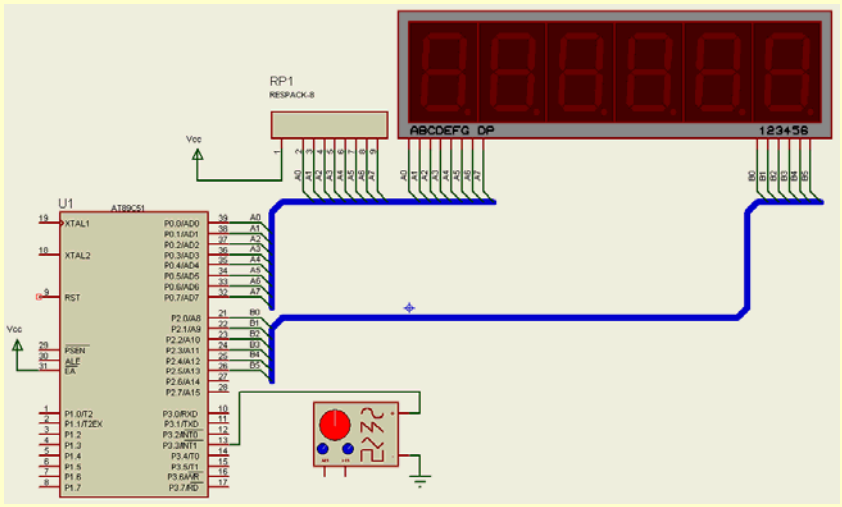


图7-17 利用GATE位测量 INT1*引脚上正脉冲的宽度的原理电路

参考程序如下：

```
#include<reg51.h>
#define uint unsigned int
#define uchar unsigned char
sbit P3_3=P3^3;                    //位变量定义
uchar count_high;                  //定义计数变量，用来读取TH0
uchar count_low;                   //定义计数变量，用来读取TL0
uint num;
uchar shiwan, wan, qian, bai, shi, ge;
uchar flag;
uchar code table[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};
                                   //共阴极数码管段码表
void delay(uint z)                 //延时函数
```

```
{
        uint x,y;
        for(x=z;x>0;x--)
        for(y=110;y>0;y--);
}
void display(uint a,uint b,uint c,uint d,uint e,uint f)
                                        //数码管显示函数
{
        P2=0xfe;
        P0=table[f];
        delay(2);
        P2=0xfd;
        P0=table[e];
        delay(2);
        P2=0xfb;
        P0=table[d];
```



```
        delay(2);
        P2=0xf7;
        P0=table[c];
        delay(2);
        P2=0xef;
        P0=table[b];
        delay(2);
        P2=0xdf;
        P0=table[a];
        delay(2);
}
void read_count()                       //读取计数寄存器的内容
```



```
{
do
        {
                count_high=TH1;          //读高字节
                count_low=TL1;           //读低字节
        }while (count_high!=TH1);
        num=count_high*256+count_low;
                /*可将两字节的机器周期数进行显示处理*/
        }
void main( )
{
        while(1)
        {
                flag=0;
                TMOD=0x90;               //设置定时器T1为方式1定时
```



```
TH1=0;                  //向定时器T1写入计数初值
 TL1=0;
while(P3_3==1);         //等待INT1*变低
TR1=1;                  //如果INT1*为低，启动T1（未真正开始计数）
while(P3_3==0);         //等待INT1* 变高，变高后T1真正开始计数
while(P3_3==1);         //等待INT1*变低，变低后T1停止计数
TR1=0;
read_count();           //读计数寄存器内容的函数
shiwan=num/100000;
wan=num%100000/10000;
qian=num%10000/1000;
bai=num%1000/100;
shi=num%100/10;
```

```
ge=num%10;
while(flag!=100)                    //减小刷新频率
        {
        flag++;
        display(ge,shi,bai,qian,wan,shiwan);
        }
    }
}
```

执行上述程序仿真，把INT1*引脚上出现的正脉冲宽度显示在LED数码管显示器上。晶振频率为12MHz，如果默认信号源输出频率为1kHz的方波，则数码管显示为500。

注意：在仿真时，偶尔显示501是因为信号源的问题，若将信号源换成频率固定的激励源则不会出现此问题。

关于定时器的小结与补充：

1.关于定时计数两种功能:

8051单片机具有T1、T0两个定时器计数器，并分别具有定时和计数两种功能。

C/T = 0选择定时功能，此时令计数器加1的脉冲信号是由机内提供的等间隔的信号，每隔一个机器周期加1，即每隔12/f秒加1。

C/T = 1选择计数功能，此时令计数器加1的脉冲信号由外部输入，即从8031的T0和T1（8031的14脚和15脚）输入，当被计数的外部事件发生一次，则产生一个脉冲信号从T0或T1引脚输入，令计数器加1，该脉冲信号的速率决定与外部事件的发生速率，不一定等间隔。

定时和计数的主要区别是令计数器加1的脉冲信号来源不同。

2.初始常数X的计算公式:

定时：

$(2^n - X) \times (12/f) =$ 定时间隔

计数：

$2^n - X =$ 计数次数

式中的 n,当分别选择工作方式0、1、2、3时，

n =13、16、8、8

3.定时间隔和计数的范围：以6MHZ晶振为例

方式0　13位计数器　最大定时间隔为：当X=0000H时

$2^{13} \times 2 \times 10^{-6} = 16.384$ms

最小定时间隔为：当13位计数器全1，即X = FF1FH或FFFFH时　2μs

计数范围：　1～8192

方式1　16位计数器　最大定时间隔为: 当X=0000H时

$2^{16} \times 2 \times 10^{-6} = 65536 \times 2 \times 10^{-6} = 131.072$ ms

最小定时间隔为: 当X=FFFFH 时　2μs

计数范围：1～65536

方式2和方式3　8位计数器

定时间隔为:　2μs～512μs　计数范围：　1～256