51单片机轻松入门

—基于STC15W4K系列

（C语言版）

李友全　编著

2016年3月编辑整理（第8章）

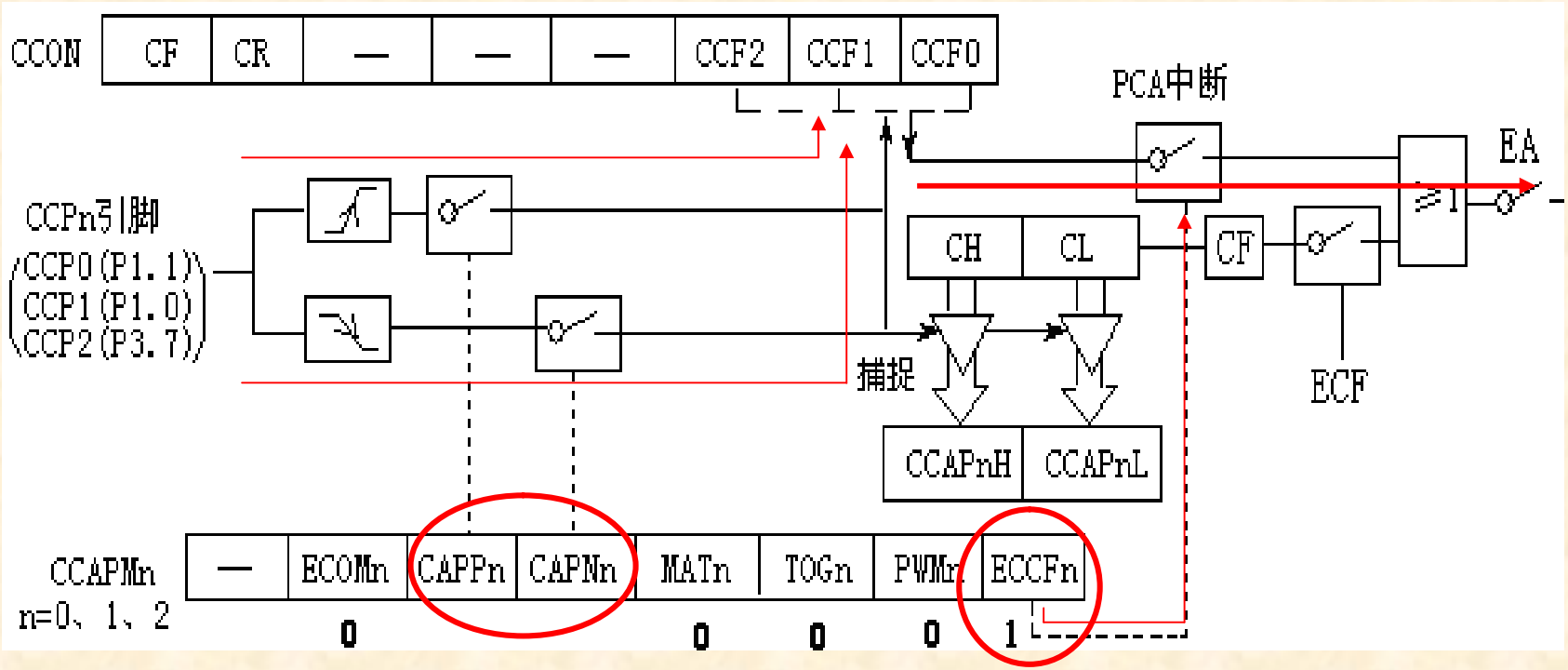
第8章　CCP/PCA/PWM模块(可用作DAC)

1 捕获模式(用于扩展外中断)
2 16位定时器模式
3 高速输出模式
4 脉宽调节模式

CCP/PCA/PWM模块其实是1个模块，通过软件不同的设置可实现4种不同的功能：外中断（3个）、定时器（1个）、高速时钟输出（3个）、 PWM脉宽调制输出（3个）。

1 捕获模式(用于扩展外中断)

PCA模块工作于捕获模式的结构如图所示。



要使PCA 模块工作在捕获模式，寄存器CCAPMn（n = 0、1、2）的两位CAPPn和CAPNn中至少有一位必须置1，PCA模块工作于捕获模式时，对外部输入CCPn引脚的跳变进行采样，当采样到有效跳变时，置位CCON中的CCFn，如果CCAPMn中的ECCFn位为1，将产生中断，可在中断服务程序中根据标志CCF2、CCF1、CCF0判断是哪一个模块产生了中断，并注意中断标志位的软件清0问题。

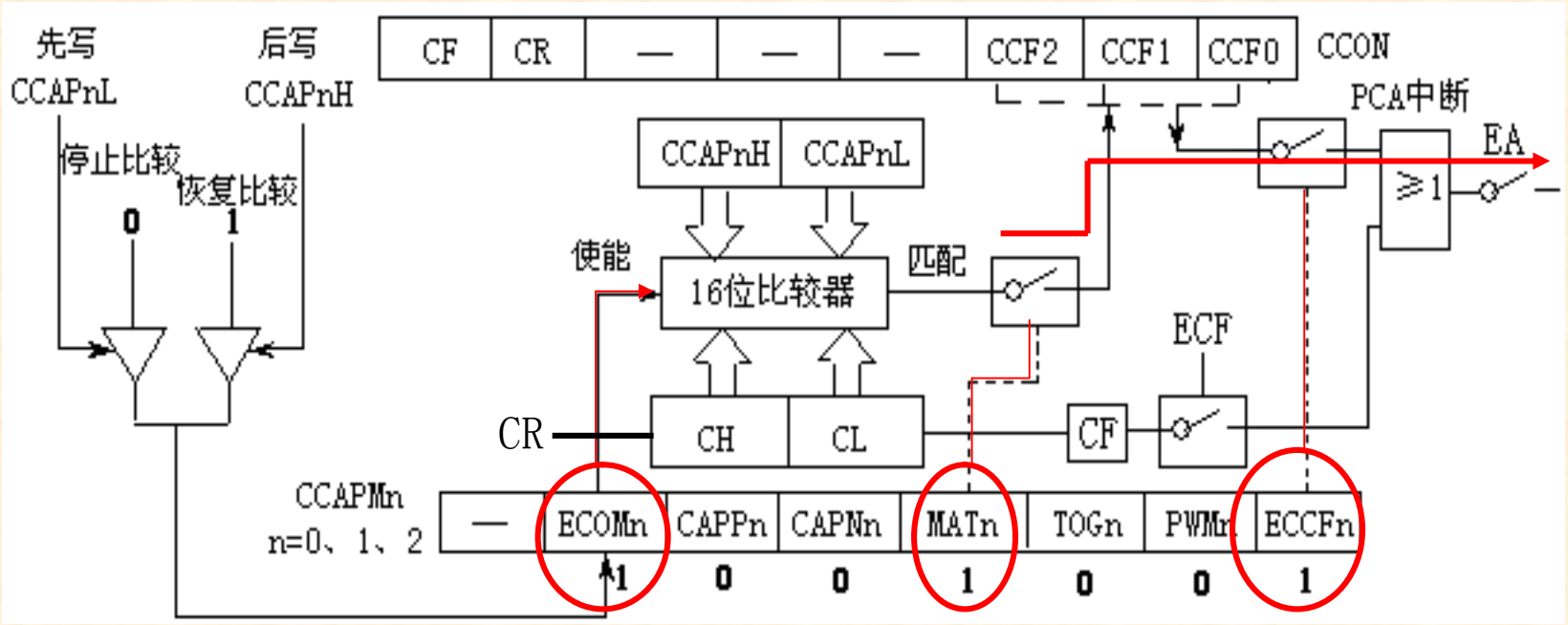
例8.1　利用PCA模块扩展2路外部中断。

说明：将P1.0（PCA通道1的外部输入）用作上升沿/下降沿都可触发的外部中断，当中断产生时对P0.0取反，将P1.1（PCA通道0的外部输入）用作下降沿触发的外部中断，当中断产生时对P0.1 取反，P0.0 、P0.1连接LED灯指示状态。

```
#include "STC15W4K.H"   // 包含STC15W4K寄存器定义文件
sbit LED_PCA0=P0^1;     // PCA0对应P1.1脚
sbit LED_PCA1=P0^0;     // PCA1对应P1.0脚
void main (void)
{
      CCAPM0=0x11;      // 设置PCA模块0下降沿触发捕捉功能
      CCAPM1=0x31;      // 设置PCA模块1上升/下降沿均可触发捕捉功能
      EA=1;             // 开整个单片机所有中断共享的总中断控制位
      while(1);         // 等待中断
}
void PCA(void) interrupt 7      // PCA中断服务程序
{
    if(CCF0)                    // PCA模块0中断服务程序
    {
       LED_PCA0=! LED_PCA0;     // LED_PCA0取反,表示PCA模块0发生了中断
       CCF0=0;                  // 清PCA模块0中断标志
    }
    else if(CCF1)               // PCA模块1中断服务程序
    {
        LED_PCA1=!LED_PCA1;     // LED_PCA1取反, 表示PCA模块1发生了中断
        CCF1=0;                // 清PCA模块1中断标志
    }
}
```

2 16位定时器模式

16位定时器模式的结构如下图所示，定时精度与16位自动重装的通用定时器相同，与通用定时器不同的是：通用定时器中断函数中只需要编写定时时间到的代码（16位自动重装方式），PCA定时器中断函数中除了编写定时时间到的代码，还要修改CCAPnH、CCAPnL的值（类似于重装定时器初值）



PCA模块用作定时器使用时，需要将寄存器CCAPMn（n = 0、1、2）的ECOMn、MATn和ECCFn位置1。

PCA计数器[CH,CL]每隔一定时间自动加1，时间间隔取决于选择的时钟源。默认时钟源为SYSclk/12时，每12个时钟周期[CH,CL]加1，当[CH,CL]增加到等于捕捉/比较寄存器[CCAPnH,CCAPnL] 的值时，CCFn=1，产生中断请求，如果每次PCA模块中断后，在中断程序中给[CCAPnH,CCAPnL]增加一个相同的数值，那么下一次中断来临的时间间隔T也是相同的，从而实现了定时功能。

PCA计数器步长值 = 定时时间 * 计数脉冲频率

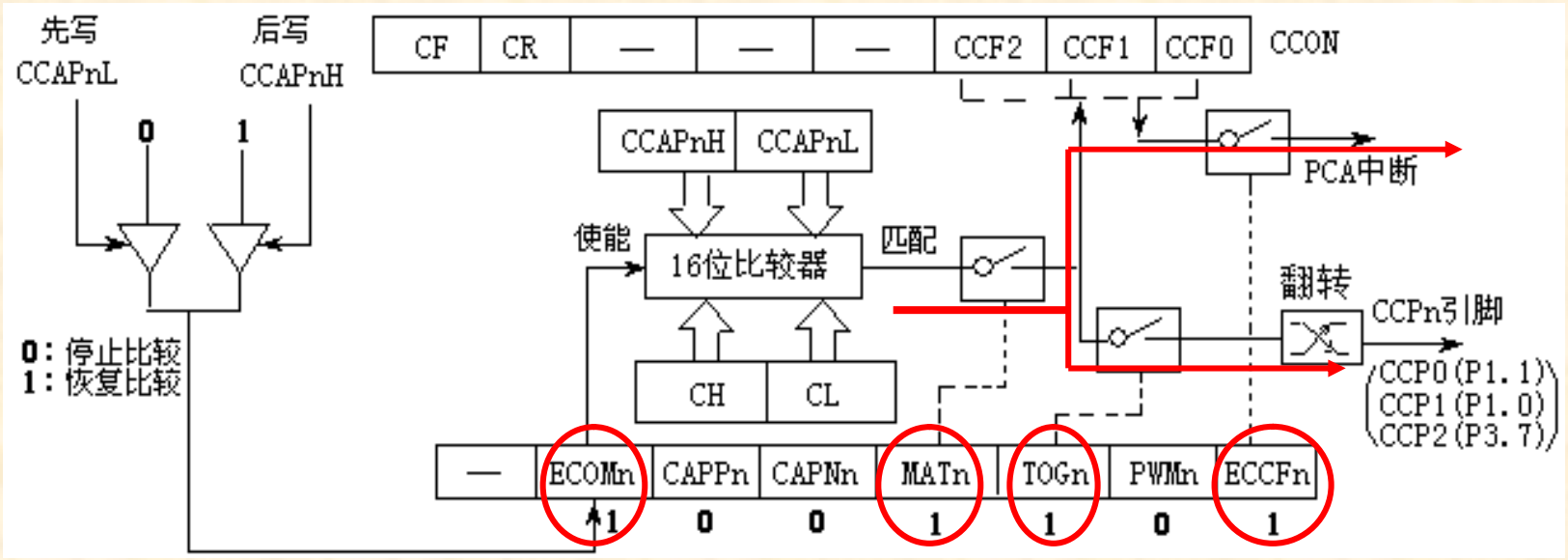
假设统时钟频率SYSclk = 22.1184MHz，使用默认时钟源SYSclk/12，定时时间T要求为5ms，则PCA计数器步长值为：T*（SYSclk/12）= 0.005*22118400/12=9216 = 2400H。

例：PCA通道0用作5mS定时器控制P0.0连接的LED闪烁

```
#include "STC15W4K.H"   // 包含STC15W4K寄存器定义文件
sbit LED=P0^0;
void main (void)
{
    CCAP0L=0;           // 给PCA模块0的CCAP0L置初值
    CCAP0H=0x24;        // 给PCA模块0的CCAP0H 置初值
    CCAPM0=0x49;        // 设置PCA模块0为16位定时器
    EA=1;               // 开总中断
    CR=1;               // 启动定时器
    while(1);                       // 等待中断
}
void PCA(void) interrupt 7          // 每5ms进入一次PCA中断
{
    unsigned int temp;
    temp=(CCAP0H<<8)+CCAP0L+0x2400; // 运算符“+”的优先级大于“<<”
    CCAP0L=temp;            // 取计算结果的低8位
    CCAP0H=temp>>8;         // 取计算结果的高8位
    CCF0=0;                 // 清 PCA 模块0 中断标志
    LED =!LED;              // 在P0.0输出脉冲宽度为5ms的方波
}
```

3　高速输出模式

高速输出模式的结构如图8-5所示，原理与16位定时器模式几乎完全相同，与通用时钟（前面第3章）输出不同的是：通用时钟输出不占用中断，PCA时钟输出需要CPU反复中断。



PCA模块用作时钟输出时，需要将寄存器CCAPMn（n = 0、1、2）的ECOMn、MATn、TOGn位置1。当PCA计数器CH、CL的值与模块捕捉/比较寄存器CCAPnH、CCAPnL的值相等时，PCA模块的输出引脚CCPn将发生翻转，为了得到需要的输出频率，需要在中断函数中修改CCAPnH、CCAPnL递增步长值。

PCA计数器步长值= 计数脉冲频率/(2×Fout)

其中Fout表示PCA模块通道n输出时钟频率。比如系统时钟频率SYSclk = 22.1184MHz，使用默认时钟源SYSclk/12时，要求在CCPn引脚输出10KHz的方波，步长值 = (22118400/12)/(2×10000)=92.16，四舍五入取整得92，即十六进制005C。

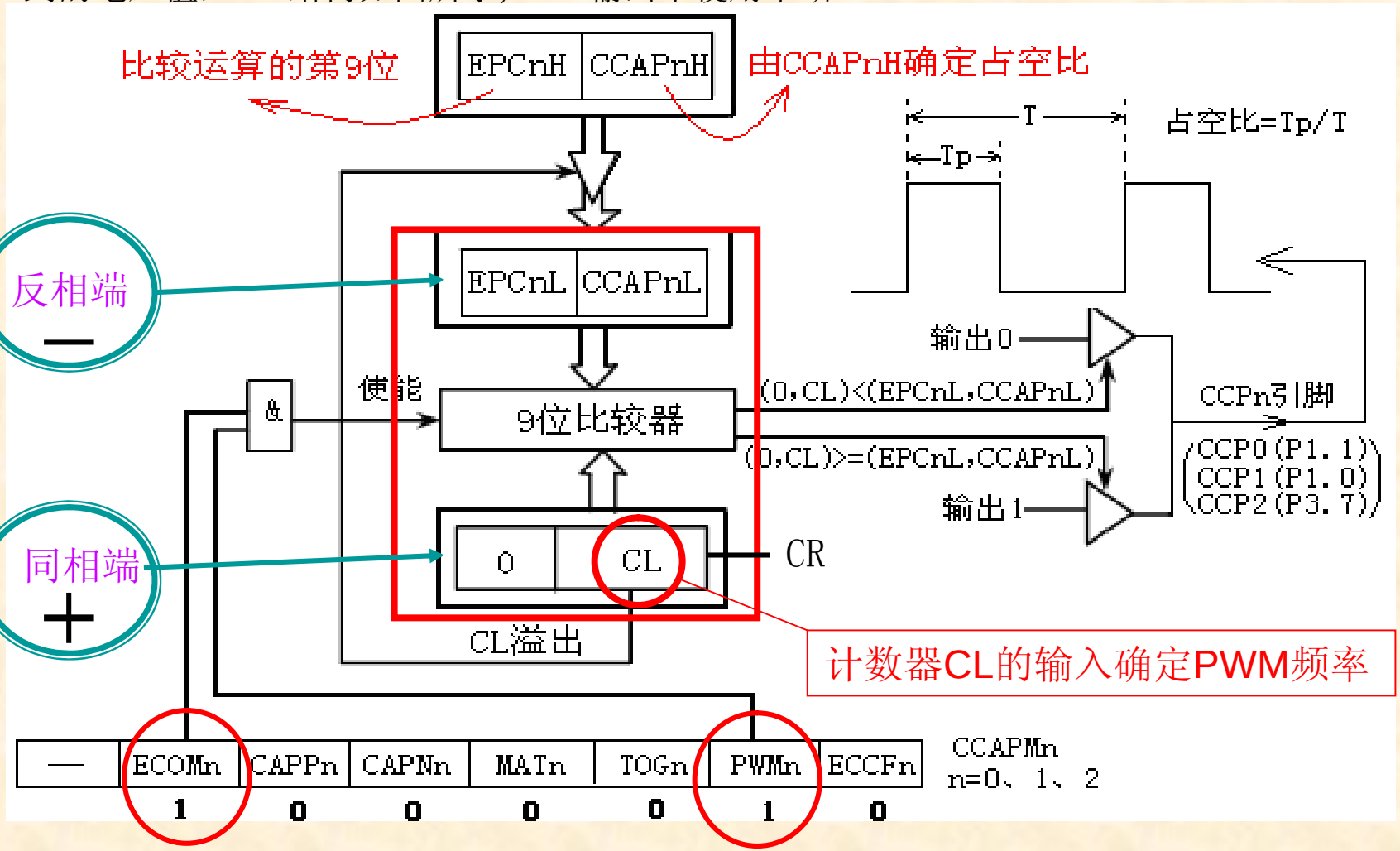
例：PCA通道0用作10KHz时钟输出（输出口P1.1）

```
#include "STC15W4K.H"          // 包含STC15W4K寄存器定义文件
void main()
{
        CCAP0L=0x5C;           // 给PCA模块0的CCAP0L置初值
        CCAP0H=0;              // 给PCA模块0的CCAP0H 置初值
        CCAPM0=0x4D;           // 设置PCA模块通道0为时钟输出模式。
        EA=1;                  // 开总中断
        CR=1;                  // 启动定时器
        while(1);              // 等待中断
}

void PCA() interrupt 7
{
    unsigned int temp;                  // 临时变量
    temp=(CCAP0H<<8)+CCAP0L+0x5C; // 运算符“+”的优先级大于“<<”
    CCAP0L=temp;                        // 取计算结果的低8位
    CCAP0H=temp>>8;                     // 取计算结果的高8位
    CCF0 = 0;                           // 清PCA模块0中断标志
}
```

4　脉宽调节模式

脉冲宽度调制简称PWM ，可用于调整输出直流平均电压，对于矩形波而言，输出平均压等于峰值电压×占空比，占空比是一个脉冲周期内高电平时间与周期的比值，例如，峰值电压等于5V，占空比等于50%的方波信号平均电压等于2.5V，也就是万用表直流档测量得到的电压值，PWM结构如图所示，PWM输出不使用中断。



要使能PWM模式，模块CCAPMn寄存器的ECOMn和PWMn位必须置位。
{0,CL[7:0]}与[EPCnL,CCAPnL[7:0]]进行比较，当{0,CL[7:0]}中的值小于{EPCnL,CCAPnL[7:0]}时，输出为低，当{0,CL[7:0]}中的值等于或大于{EPCnL,CCAPnL[7:0]}时，输出为高，当EPCnL=0且CCAPnL=00H时，PWM固定输出高，当EPCnL=1且CCAPnL=FFH 时，PWM固定输出低。

PWM默认输出频率=系统时钟/12/256　//系统时钟典型值22.1184MHz

PWM输出占空比 =（1- CCAPnH/256）×100%，进一步推导可得出：
CCAPnH =（1-占空比）×256
比如要求输出占空比为87.5%的信号， CCAPnH=（1-0.875）×256=32，即0x20。

当某个I/O 口作为PWM使用时，该口自动切换到强推挽输出模式。因此实验过程中注意IO口不能短路。

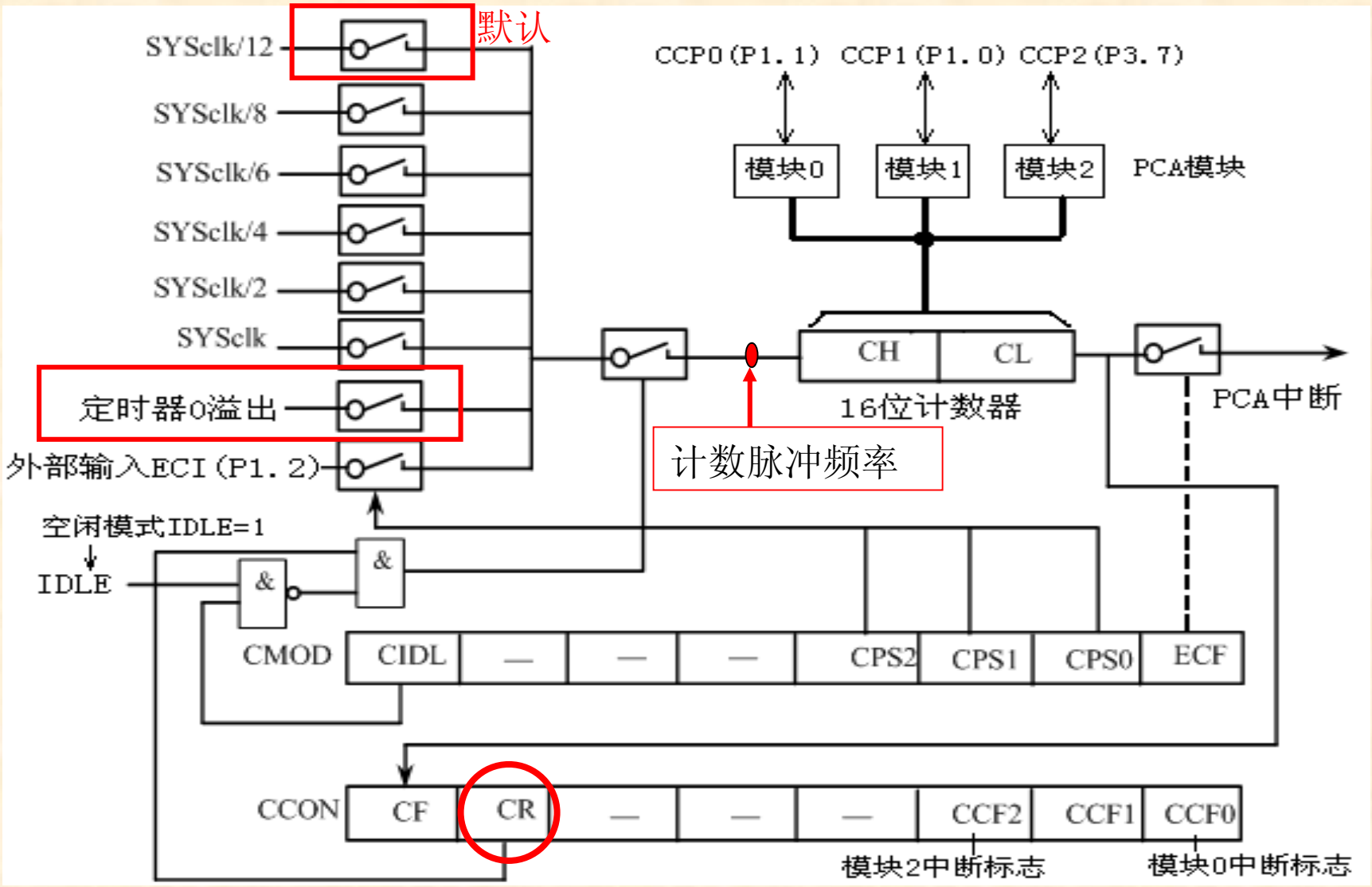
例8.5　利用PCA 模块输出占空比为87.5%的PWM信号。

说明：利用PCA通道0实现在P1.1输出占空比为87.5%的PWM信号，假设R/C时钟频率Fosc = 22.1184MHz。

```
#include "STC15W4K.H"   // 包含STC15W4K寄存器定义文件

void main()
{
        CCAPM0=0x42;        // 设置PCA模块为PWM输出方式。
        CR=1;               // PCA计数器开始运行
        CCAP0H=0x20;        // 脉宽控制
        while(1);           // 让程序停在这里。
}
```

理论计算P1.1频率= 22118400/12/256=7.2 KHz，　实验结果：用万用表测量P1.1输出频率为7.210KHz，占空比为87.5%。可见理论计算与实际结果是一致的。



PWM输出频率 = 计数脉冲频率/256，当选择定时器0溢出时，PWM的频率 =（1/定时时间）/256，反过来，定时时间=1/(256× PWM的频率)

比如：要求 PWM的频率为200Hz，定时时间=19.53125uS,定时器初值=0xFFDC

例：定时器0用作PCA时钟源，通道1输出设定频率（200Hz）的PWM信号。

```
#include "STC15W4K.H"        // 包含 "STC15W4K.H"寄存器定义头文件
void initPCA()
{
    // 初始化定时器T0为16位自动重装方式,其溢出脉冲作PCA计数器的时钟源
    TMOD=0x00;      // 设置T0为16位自动重装方式
    TH0=0xff;
    TL0=0xED;
    TR0=1;          // 启动定时器0
    // 初始化PCA通道1为PWM输出方式
    CMOD=0x84;      // 10000100B ,选择T0为PCA计数器时钟源
    CCAPM1=0x42;    // 设置PCA模块1为8位PWM输出方式。脉冲在P1.0引脚输
                    // 出，PWM无需中断支持。
    CCAP1H=0x20;    // 设置脉冲宽度
    EA=1;           // 开总中断
    CR=1;           // 启动PCA计数器
}

void main (void)
{
    initPCA();
    while(1);       // 这里可以编写其它程序
}
```

网上搜《51单片机轻松入门——基于STC15W4K系列》即可找到对应的视频地址