



第9章 STC单片机

CCP/PCA/PWM模块原理及实现

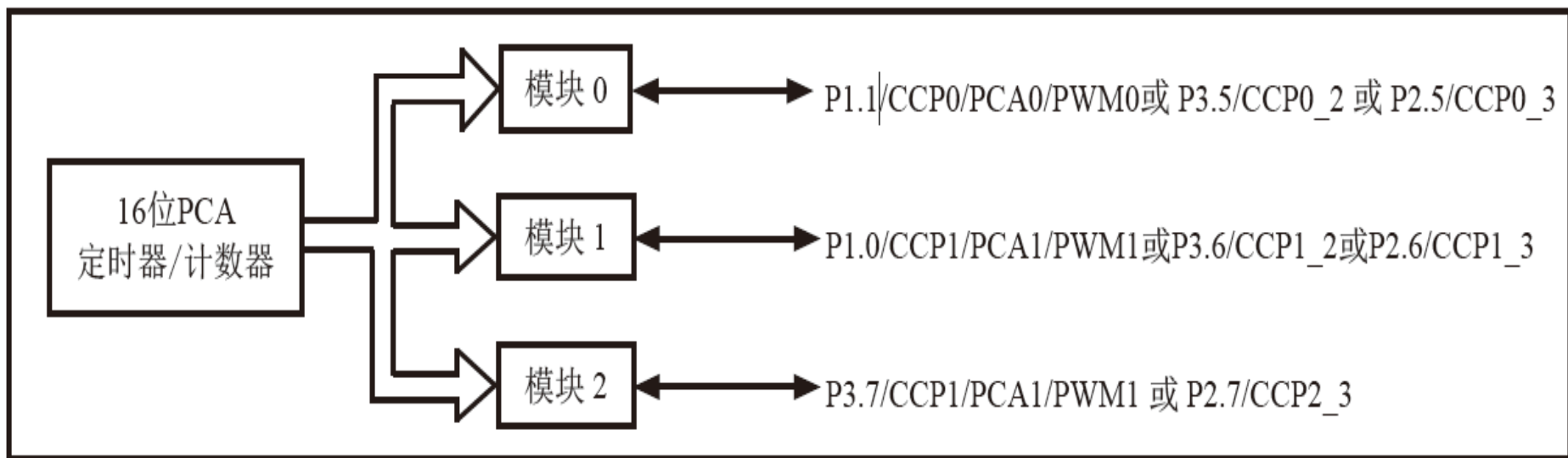
何宾
2015.02

本章主要内容

- CCP/PCA/PWM结构
- CCP/PCA/PWM寄存器组
- CCP/PCA/PWM工作模式

CCP/PCA/PWM结构

STC15系列部分单片机内部集成了3路CCP/PCA/PWM模块



CCP/PCA/PWM结构

PCA模块包含一个特殊的16位定时器，有3个16位的捕获/比较模块和该定时器/计数器模块相连。通过软件程序，每个模块可以设置工作在下面四种模式中的一种：

- **上升/下降沿捕获**
- **软件定时器**
- **高速脉冲输出**
- **可调脉冲输出**

CCP/PCA/PWM结构

通过AUXR1 (P_SW1) 寄存器可以控制这三路CCP/PCA/PWM输出所使用的引脚号。

■ CCP/PCA/PWM的输出

- 对于CCP/PCA/PWM模块0的输出，可以选择使用P1.1、P3.5或者P2.5引脚。
- 对于CCP/PCA/PWM模块1的输出，可以选择使用P1.0、P3.6、P2.6引脚。
- 对于CCP/PCA/PWM模块2的输出，可以选择使用P3.7或者P2.7引脚。

CCP/PCA/PWM结构

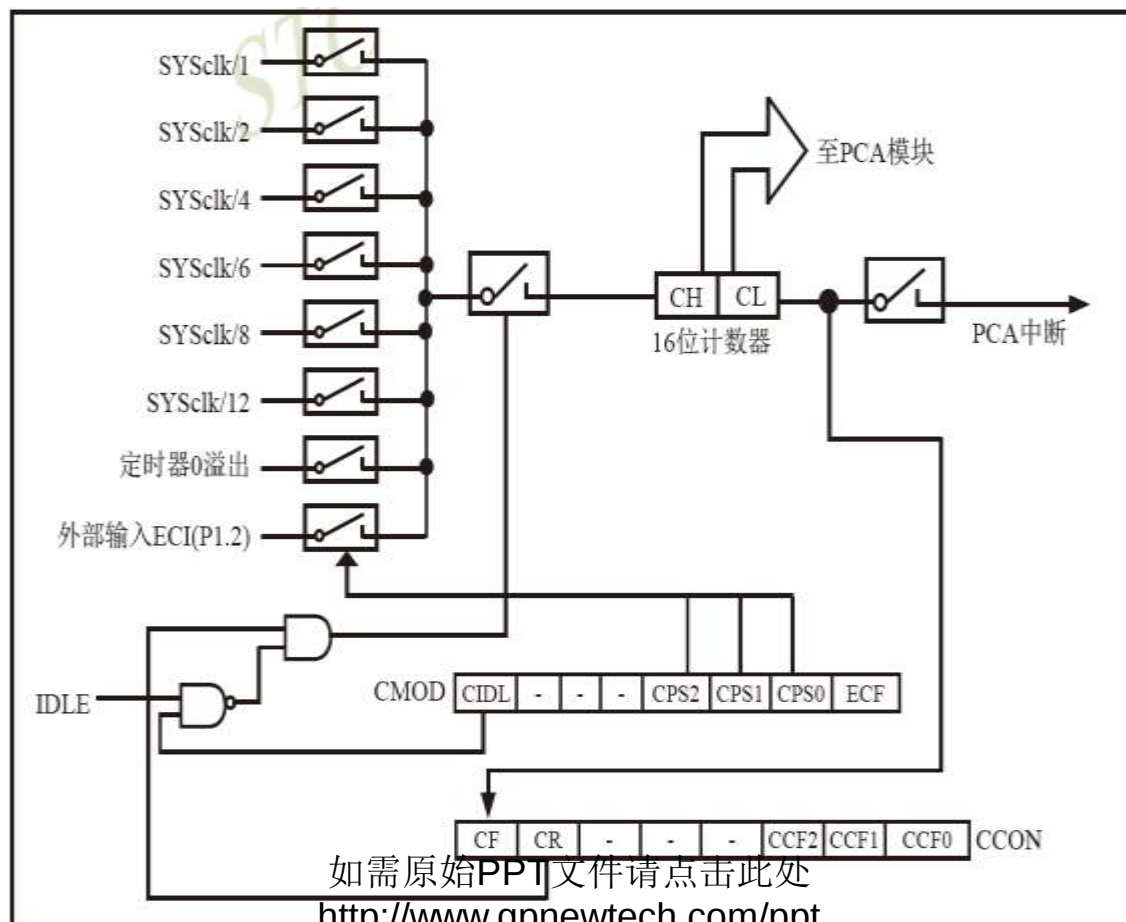
■ CCP/PCA/PWM的外部脉冲输入

- 对于CCP/PCA/PWM不同模块使用一个外部脉冲输入（ External Control Input , CPI ） , 该输入信号可以选择使用P1.2、 P3.4或者P2.4引脚，也就是后面所说的输入可以在P1、 P2和P3口之间进行切换。

CCP/PCA/PWM结构

对16位PCA计数器/定时器的结构。

- 计数器CH和CL的内容是正在自由递增计数16位PCA定时器的值。



CCP/PCA/PWM结构

- PCA定时器是三个模块的公共时间基准。
- CMOD寄存器还有两位与PCA有关，即：
 - CIDL，空闲模式下允许停止控制位；
 - ECF，PCA 中断使能位。即：当PCA定时器溢出时，将CCON寄存器的PCA计数溢出标志位CF置位。
- 通过CMOD寄存器CPS2、CPS1和CPS0位，选择16位PCA定时器/计数器的时钟源，包括：
 - SYSclk/1、SYSclk/2、SYSclk/4、SYSclk/6、SYSclk/8、SYSclk/12、定时器0溢出和外部脉冲输入。

CCP/PCA/PWM结构

- CCON寄存器包含PCA的运行控制位CR和PCA定时器标志CF以及各个模块的标志CCF2、CCF1和CCF0。
 - 通过软件置位CR为1（CCON.6），使能运行PCA；当软件置位CR为0时，禁止运行PCA。
 - 当PCA计数器溢出时，置位CF。如果CMOD寄存器的ECF位为1，则产生PCA中断。

CCP/PCA/PWM结构

CCON寄存器的第2位~第1位是PCA各个模块的标志位。

- 第0位对应与模块0；
- 第1位对应模块1；
- 第2位对应模块2。

注：当发生匹配或者比较时，由硬件置位这些比特位。

CCP/PCA/PWM结构

PCA的每个模块都分别对应一个寄存器。

- 模块0对应于CCAPM0 ;
- 模块1对应于CCAPM1 ;
- 模块2对应于CCAPM2 ;

CCP/PCA/PWM结构

每个模块还对应于另外两个寄存器，CCAPnH和CCAPnL。

- 当出现捕获或者比较时，它们可以用来保存16位的计数值。
- 当PCA模块用于PWM模式时，用来控制输出的占空比。

CCP/PCA/PWM寄存器组

--PCA工作模式寄存器

PCA工作模式寄存器CMOD。

- 该寄存器位于STC单片机特殊功能寄存器地址为0xD9的位置。
- 当复位后，该寄存器的值为0xxx0000。

PCA工作模式寄存器CMOD各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	CIDL	--	--	--	CPS2	CPS1	CPS0	ECF

■ CIDL

- 空闲模式下是否停止PCA计数的控制位。当该位为0时，空闲模式下PCA计数器继续工作；当该位为1时，空闲模式下PCA计数器停止工作。

CCP/PCA/PWM寄存器组

--PCA工作模式寄存器

■ CPS2~CPS0

□ PCA计数脉冲源选择控制位。

CPS2	CPS1	CPS0	选择
1	1	1	SYSc1k/8
1	1	0	SYSc1k/6
1	0	1	SYSc1k/4
1	0	0	SYSc1k
0	1	1	外部控制脉冲输入SYSc1k/2
0	1	0	定时器溢出脉冲
0	0	1	SYSc1k/2
0	0	0	SYSc1k/12

■ ECF

□ PCA计数溢出中断使能位。当该位为0时，禁止寄存器CCON中CF位的中断；当该位为1时，允许寄存器CCON中CF位的中断。

CCP/PCA/PWM寄存器组

--PCA控制寄存器



PCA控制寄存器CCON

- 该寄存器位于STC单片机特殊功能寄存器地址为0xD8的位置。
- 当复位后，该寄存器的值为“00xxx000”。

PCA控制寄存器CCON各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	CF	CR	--	--	--	CCF2	CCF1	CCF0

- CF
 - PCA计数器阵列溢出标志位。当PCA计数器溢出时，硬件将该位置1。此时，如果CMOD寄存器的ECF位为1，则CF标志位可用于产生中断。可以通过软件或者硬件给CF位置1，但是只能通过软件将CF位清0。

CCP/PCA/PWM寄存器组

--PCA控制寄存器

■ CR

- PCA计数器阵列运行控制位。当该位为1时，启动PCA计数器阵列；当该位为0时，关闭PCA计数器阵列。

■ CCF2

- PCA模块2中断标志。当出现匹配或者捕获时，由硬件将该位置1。该位必须通过软件清0。

■ CCF1

- PCA模块1中断标志。当出现匹配或者捕获时，由硬件将该位置1。该位必须通过软件清0。

■ CCF0

- PCA模块0中断标志。当出现匹配或者捕获时，由硬件将该位置1。该位必须通过软件清0。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器

PCA比较捕获寄存器CCAPM，包括3个寄存器CCAPM0、CCAPM1、CCAPM2。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAPM0

PCA比较捕获寄存器CCAPM0

- 该寄存器位于STC单片机特殊功能寄存器地址为0xDA的位置。
- 当复位后，该寄存器的值为“x0000000”。

比较捕获寄存器CCAPM0各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	--	ECOM0	CAPP0	CAPN0	MAT0	TOG0	PWM0	EECF0

■ ECOM0

- 允许比较器功能控制位。当该位为1时，允许比较器功能；当该位为0时禁止比较器功能。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAPM0

■ CAPP0

□ 上升沿控制位。当该位为1时，允许上升沿捕获；当该位为0时，禁止上升沿捕获。

■ CAPN0

□ 下降沿控制位。当该位为1时，允许下降沿捕获；当该位为0时，禁止下降沿捕获。

■ MAT0

□ 匹配控制位。当该位为1时，PCA计数值与模块的比较/捕获寄存器值的匹配将置位CCON寄存器的中断标志CCF0。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAPM0

■ TOG0

- 翻转控制位。当该位为1时，工作在PCA高速脉冲输出模式，PCA计数器的值与模块的比较/捕获寄存器值的匹配将使CCP0引脚翻转。

■ PWM0

- 脉冲宽度调节模式。当该位为1时，允许CCP0用于PWM输出；当该位为0时，禁止CCP0用于PWM输出。

■ ECCF0

- 使能CCF0中断。当该位为1时，使能寄存器CCON的比较/捕获标志CCF0产生中断。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP1

PCA比较捕获寄存器CCAPM1

- 该寄存器位于STC单片机特殊功能寄存器地址为0xDB的位置。
- 当复位后，该寄存器的值为“x0000000”。

PCA比较捕获寄存器CCAPM1 各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	--	ECOM1	CAPP1	CAPN1	MAT1	TOG1	PWM1	EECF1

■ ECOM1

- 允许比较器功能控制位。当该位为1时，允许比较器功能；当该位为0时，禁止比较器功能。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP1

■ CAPP1

- 上升沿控制位。当该位为1时，允许上升沿捕获；当该位为0时，禁止上升沿捕获。

■ CAPN1

- 下降沿控制位。当该位为1时，允许下降沿捕获；当该位为0时，禁止下降沿捕获。

■ MAT1

- 匹配控制位。当该位为1时，PCA计数值与模块的比较/捕获寄存器值的匹配将置位CCON寄存器的中断标志CCF1。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP1

■ TOG1

- 翻转控制位。当该位为1时，工作在PCA高速脉冲输出模式，PCA计数器的值与模块的比较/捕获寄存器值得匹配将使CCP1引脚翻转。

■ PWM1

- 脉冲宽度调节模式。当该位为1时，允许CCP1用于PWM输出；当该位为0时，禁止CCP1用于PWM输出。

■ ECCF1

- 使能CCF1中断。当该位为1时，使能寄存器CCON的比较/捕获标志CCF1产生中断。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP2

PCA比较捕获寄存器CCAPM2

- 该寄存器位于STC单片机特殊功能寄存器地址为0xDA的位置。
- 当复位后，该寄存器的值为“x0000000”。

PCA比较捕获寄存器CCAPM2各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	--	ECOM2	CAPP2	CAPN2	MAT2	TOG2	PWM2	EECF2

■ ECOM2

- 允许比较器功能控制位。当该位为1时，允许比较器功能；当该位为0时，禁止比较器功能。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP2

■ CAPP2

□ 上升沿控制位。当该位为1时，允许上升沿捕获；当该位为0时，禁止上升沿捕获。

■ CAPN2

□ 下降沿控制位。当该位为1时，允许下降沿捕获；当该位为0时，禁止下降沿捕获。

■ MAT2

□ 匹配控制位。当该位为1时，PCA计数值与模块的比较/捕获寄存器值的匹配将置位CCON寄存器的中断标志CCF2。

CCP/PCA/PWM寄存器组

--PCA比较捕获寄存器CCAMP2

■ TOG2

- 翻转控制位。当该位为1时，工作在PCA高速脉冲输出模式，PCA计数器的值与模块的比较/捕获寄存器值匹配将使CCP2引脚翻转。

■ PWM2

- 脉冲宽度调节模式。当该位为1时，允许CCP2用于PWM输出；当该位为0时，禁止CCP2用于PWM输出。

■ ECCF2

- 使能CCF2中断。当该位为1时，使能寄存器CCON的比较/捕获标志CCF2产生中断。

CCP/PCA/PWM寄存器组

--PCA的16位计数器

PCA的16位计数器

- 分别用CL寄存器和CH寄存器表示。
- CL和CH寄存器分别位于STC单片机特殊功能寄存器地址为0xE9和0xF9的位置。
- 当复位后，CL和CH寄存器的值均为“00000000”。

PCA的寄存器CL各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	16位计数值的低8位							

PCA的寄存器CH各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	16位计数值的高8位							

9.2 CCP/PCA/PWM寄存器组

PCA捕获/比较寄存器

本节介绍PCA捕获/比较寄存器，分别对应于每个模块，即：CCAP0、CCAP1和CCAP2。当PCA模块用于捕获或者比较时，它们用于保存各个模块的16位捕捉计数值；当PCA模块用于PWM时，它们用来控制输出的占空比。

表1 PCA的寄存器CL各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	16位计数值的低8位							

表2 PCA的寄存器CH各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	16位计数值的高8位							

CCP/PCA/PWM寄存器组

--PCA捕获/比较寄存器CCAP0

PCA捕获/比较寄存器CCAP0

- 分别用CCAP0L寄存器和CCAP0H寄存器表示。
- CCAP0L和CCAP0H寄存器分别位于STC单片机特殊功能寄存器地址为0xEA和0xFA的位置。
- 当复位后，CCAP0L和CCAP0H寄存器的值均为“00000000”。

PCA捕获/比较寄存器CCAP0L各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP0低8位							

PCA捕获/比较寄存器CCAP0H各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP0高8位							

CCP/PCA/PWM寄存器组

--PCA捕获/比较寄存器CCAP1

PCA捕获/比较寄存器CCAP1

- 分别用CCAP1L寄存器和CCAP1H寄存器表示。
- CCAP1L和CCAP1H寄存器分别位于STC单片机特殊功能寄存器地址为0xEB和0xFB的位置。
- 当复位后，CCAP1L和CCAP1H寄存器的值均为“00000000”。

PCA捕获/比较寄存器CCAP1L各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP1低8位							

PCA捕获/比较寄存器CCAP1H各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP1高8位							

CCP/PCA/PWM寄存器组

--PCA捕获/比较寄存器CCAP2

PCA捕获/比较寄存器CCAP2

- 分别用CCAP2L寄存器和CCAP2H寄存器表示。
- CCAP2L和CCAP2H寄存器分别位于STC单片机特殊功能寄存器地址为0xEC和0xFC的位置。
- 当复位后，CCAP2L和CCAP2H寄存器的值均为“00000000”。

PCA捕获/比较寄存器CCAP2L各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP2低8位							

PCA捕获/比较寄存器CCAP2H各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	捕获/比较寄存器CCAP2高8位							

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器

PCA模块PWM寄存器PCA_PWM

- 它包含PCA_PWM0、PCA_PWM1和PCA_PWM2，这些寄存器分别对应一个模块。

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM0

PCA模块PWM寄存器PCA_PWM0

- 该寄存器位于STC单片机特殊功能寄存器地址为0xF2的位置。
- 当复位后，该寄存器的值为“00xxxx00”。

PCA模块PWM寄存器PCA_PWM0

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	EBS0_1	EBS0_0	--	--	--	--	EPC0H	EPC0L

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM0

■ EBS0_1和EBS0_0

□ 当PCA模块工作在PWM模式时的功能选择位。

ESB0_1	ESB0_0	功能
0	0	PCA模块0工作于8位PWM功能
0	1	PCA模块0工作于7位PWM功能
1	0	PCA模块0工作于6位PWM功能
1	1	无效，PCA模块0仍工作于8位PWM功能

■ EPC0H

□ 在PWM模式下，与CCAP0H组成9位数。

■ EPC0L

□ 在PWM模式下，与CCAP0L组成9位数。

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM1

PCA模块PWM寄存器PCA_PWM1

- 该寄存器位于STC单片机特殊功能寄存器地址为0xF3的位置。
- 当复位后，该寄存器的值为“00xxxx00”。

PCA模块PWM寄存器PCA_PWM1各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	EBS1_1	EBS1_0	--	--	--	--	EPC1H	EPC1L

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM1

■ EBS1_1和EBS1_0

□ 当PCA模块工作在PWM模式时的功能选择位。

ESB1_1	ESB1_0	功能
0	0	PCA模块1工作于8位PWM功能
0	1	PCA模块1工作于7位PWM功能
1	0	PCA模块1工作于6位PWM功能
1	1	无效，PCA模块1仍工作于8位PWM功能

■ EPC1H

□ 在PWM模式下，与CCAP1H组成9位数。

■ EPC1L

□ 在PWM模式下，与CCAP1L组成9位数。

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM2

PCA模块PWM寄存器PCA_PWM2

- 该寄存器位于STC单片机特殊功能寄存器地址为0xF4的位置。
- 当复位后，该寄存器的值为“00xxxx00”。

PCA模块PWM寄存器PCA_PWM2各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	EBS2_1	EBS2_0	--	--	--	--	EPC2H	EPC2L

CCP/PCA/PWM寄存器组

--PCA模块PWM寄存器PCA_PWM2

■ EBS2_1和EBS2_0

□ 当PCA模块工作在PWM模式时的功能选择位。

ESB2_1	ESB2_0	功能
0	0	PCA模块2工作于8位PWM功能
0	1	PCA模块2工作于7位PWM功能
1	0	PCA模块2工作于6位PWM功能
1	1	无效，PCA模块2仍工作于8位PWM功能

■ EPC2H

□ 在PWM模式下，与CCAP2H组成9位数。

■ EPC2L

□ 在PWM模式下，与CCAP2L组成9位数。

CCP/PCA/PWM寄存器组

--PCA工作模式寄存器AUXR1

PCA模块引脚切换寄存器AUXR1 (P_SW1)

- 用于选择CCP输出、SPI接口和串口所使用的引脚在单片机上的位置。
- 该寄存器位于STC单片机特殊功能寄存器地址为0xA2的位置
- 当复位后，该寄存器的值为“00000000”。

PCA模块引脚切换寄存器AUXR1 (P_SW1) 各位的含义

比特	B7	B6	B5	B4	B3	B2	B1	B0
名字	S1_S1	S1_S0	CCP_S1	CCP_S0	SPI_S1	SPI_S0	0	DPS

CCP/PCA/PWM寄存器组

--PCA工作模式寄存器AUXR1

■ S1_S1和S1_S0确定串口1在单片机上引脚的位置。

S1_S1和S1_S0各位的含义

S1_S1	S1_S0	功能
0	0	选择串口1分别对应于单片机P3. 0/RxD和P3. 1/TxD引脚
0	1	选择串口1分别对应于单片机P3. 6/RxD_2和P3. 7/TxD_2引脚
1	0	选择串口1分别对应于单片机P1. 6/RxD_3和P1. 7/TxD_3引脚
1	1	无效

CCP/PCA/PWM寄存器组

--PCA工作模式寄存器AUXR1

■ CCP_S1和CCP_S0确定CCP输出在单片机上引脚的位置

CCP_S1和CCP_S0各位的含义

CCP_S1	CCP_S0	功能
0	0	选择CCP输入对应于单片机P1. 2/ECI引脚，同时选择CCP三个输出分别对应于单片机的P1. 1/CCP0、P1. 0/CCP1和P3. 7/CCP2引脚
0	1	选择CCP输入对应于单片机P3. 4/ECI_2引脚，同时选择CCP三个输出分别对应于单片机的P3. 5/CCP0_2、P3. 6/CCP1_2和P3. 7/CCP2_2引脚
1	0	选择CCP输入对应于单片机P1. 2/ECI引脚，同时选择CCP三个输出分别对应于单片机的P2. 4/CCP0_3、P2. 6/CCP1_3和P2. 7/CCP2_3引脚
1	1	无效

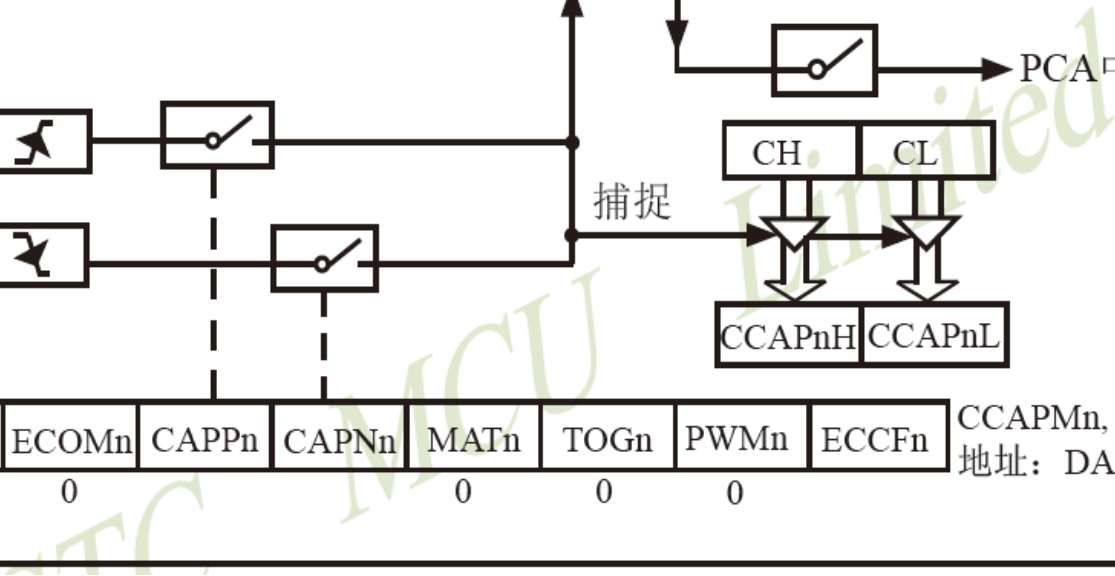
CCP/PCA/PWM寄存器组

--PCA工作模式寄存器AUXR1

■ SPI_S1和SPI_S0确定SPI接口在单片机上引脚的位置。

SPI_S1和SPI_S0各位的含义

SPI_S1	SPI_S0	功能
0	0	选择SPI接口分别对应于单片机P1. 2/SS、 P1. 3/MOSI、 P1. 4/MISO、 P1. 5/SCLK引脚
0	1	选择SPI接口分别对应于单片机P2. 4/SS_2、 P2. 3/MOSI_2、 P2. 2/MISO_2、 P2. 1/SCLK_2引脚
1	0	选择SPI接口分别对应于单片机P5. 4/SS_3、 P4. 0/MOSI_3、 P4. 1/MISO_3、 P4. 3/SCLK_3引脚
1	1	无效



CCP/PCA/PWM工作模式

--捕获模式

- 要想工作在该模式，寄存器CCAPMn的两位（CAPNn和CAPPn）或者其中一位必须置为1。
 - 当该模块工作于捕获模式时，对模块外部CCPn输入（可选择：CCP0/P1.1、CCP1/P1.0或CCP2/P3.7）的跳变进行采样。
 - 当采样到有效跳变时，PCA硬件就将PCA计数器阵列寄存器（CH和CL）的值加载到模块的捕获寄存器CCAPnL和CCAPnH中。
- 如果CCON寄存器中位CCF_n和CCAPMn寄存器中的ECCF_n置位为1，则将产生中断。可在中断服务程序中，判断产生中断的模块，并注意中断标志的清零问题。

CCP/PCA/PWM工作模式

--捕获模式

【例】 捕获模式C语言描述的例子

```
#include "reg51.h"
```

```
sfr P_SW1=0xA2;           //声明P_SW1寄存器的地址0xA2
```

```
sfr CCON=0xD8;            //声明CCON寄存器的地址0xD8
```

```
sfr CMOD=0xD9;            //声明CMOD寄存器的地址0xD9
```

```
sfr CL=0xE9;              //声明CL寄存器的地址0xE9
```

```
sfr CH =0xF9;             //声明CH寄存器的地址0xF9
```

```
sfr CCAPM0=0xDA;          //声明CCAPM0寄存器的地址0xDA
```

```
sbit CCF0=CCON^0;         //声明CCF0标志位
```

```
sbit CCF1 =CCON^1;        //声明CCF1标志位
```

```
sbit CR=CCON^6;           //声明CR标志位
```

```
sbit CF =CCON^7;          //声明CF标志位
```

如需原始PPT文件请点击此处
<http://www.gpnewtech.com/ppt>

CCP/PCA/PWM工作模式

--捕获模式

```
void PCA_int() interrupt 7           //声明PCA中断服务程序
{
    CCF0=0;                          //CCF0标志清零
    P46=!P46;                        //P4.6端口取反
    P47=!P47;                        //P4.7端口取反
}

void main()
{
    P46=0;                          //P4.6端口置0，灯亮
    P47=0;                          //P4.7端口置0，灯亮
    P_SW1=0x00;                     //CCP_S0=0,CCP_S1=0
    CCON=0;                         //停止PCA定时器，清除CF和CCF0标志
    CL=0;                           //CL寄存器清零
}
```

CCP/PCA/PWM工作模式

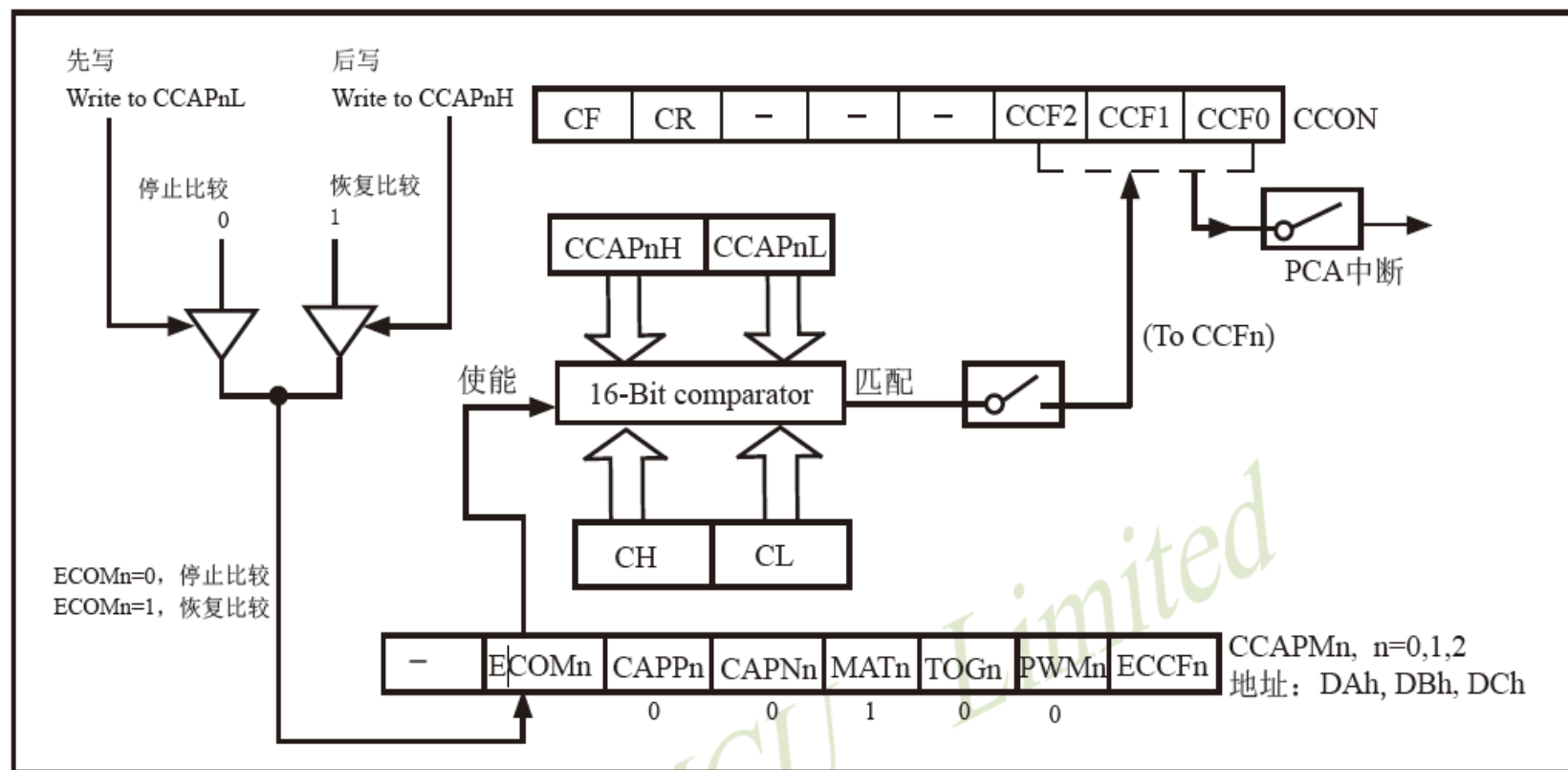
--捕获模式

```
CH=0;           //CH寄存器清零
CMOD=0x00;      //设置时钟源，禁止CF溢出中断
CCAPM0=0x11;    //PCA模块下降沿触发
CR=1;           //启动PCA定时器
EA=1;           //CPU允许响应中断请求
while(1);
}
```

CCP/PCA/PWM工作模式

--16位软件定时器模式

16位软件定时器模式结构



CCP/PCA/PWM工作模式

--16位软件定时器模式

- 通过设置CCAPMn寄存器中的ECOM和MAT位，使得PCA模块工作在16位软件定时器模式。
 - PCA定时器的值与模块捕获寄存器的值进行比较，当它们相等时，如果CCON寄存器的CCFn位和CCAPMn寄存器的ECCFn位都置位，则将产生中断。

CCP/PCA/PWM工作模式

--16位软件定时器模式

- 在16位软件定时器模式下，每个时钟节拍（由所选择时钟源确定）到来时，自动加1。
 - 当[CH, CL]增加到等于[CCAPnH, CCAPnL]时，CCFn=1，产生中断请求。
 - 如果每次PCA模块中断后，在中断服务程序给CCAPnH, CCAPnL增加相同的值时，下次中断来临的间隔时间也是相同的，从而实现了定时功能。
 - 定时时间的长短取决于时钟源的选择，以及PCA计数器计数值的设置。

CCP/PCA/PWM工作模式

--16位软件定时器模式

【例】16位软件定时器模式C语言描述的例子

```
#include "reg51.h"

#define value 3906           //定义value为3906

sfr P_SW1=0xA2;             //声明P_SW1寄存器的地址0xA2
sfr CCON=0xD8;              //声明CCON寄存器的地址0xD8
sfr CMOD=0xD9;              //声明CMOD寄存器的地址0xD9
sfr CL=0xE9;                //声明CL寄存器的地址0xE9
sfr CH =0xF9;               //声明CH寄存器的地址0xF9
sfr CCAPM0=0xDA;            //声明CCAPM0寄存器的地址0xDA
sfr CCAP0L  =0xEA;          //声明CCAP0L寄存器的地址为0xEA
sfr CCAP0H  =0xFA;          //声明CCAP0H寄存器的地址为0xFA
sfr CLK_DIV  =0x97;          //声明CLK_DIV寄存器的地址为0x97
```

CCP/PCA/PWM工作模式

--16位软件定时器模式



```
sbit CCF0=CCON^0;           //声明CCF0标志位

sbit CCF1 =CCON^1;          //声明CCF1标志位

sbit CR=CCON^6;              //声明CR标志位

sbit CF =CCON^7;             //声明CF标志位

void PCA_int() interrupt 7    //声明PCA中断服务程序
{

    CCF0=0;                   //CCF0标志清零

    CL=0;                     //CL寄存器清零

    CH=0;                     //CH寄存器清零

    P46=!P46;                 //P4.6端口清零

    P47=!P47;                 //P4.7端口清零

}
```

CCP/PCA/PWM工作模式

--16位软件定时器模式

```
void main()
```

```
{
```

```
P46=0;           //P4.6端口置0 , 灯亮
P47=0;           //P4.7端口置0 , 灯亮
CLK_DIV=0x07;    //设置SYSclk频率=主时钟频率/128
P_SW1=0x00;      //CCP_S0=0 , CCP_S1=0
CCON=0;          //停止PCA计数器 , 清除CF和CCF0标志
CL=0;            //CL寄存器清零
CH=0;            //CH寄存器清零
CMOD=0x00;       //设置PCA时钟源SYSclk/12
```

CCP/PCA/PWM工作模式

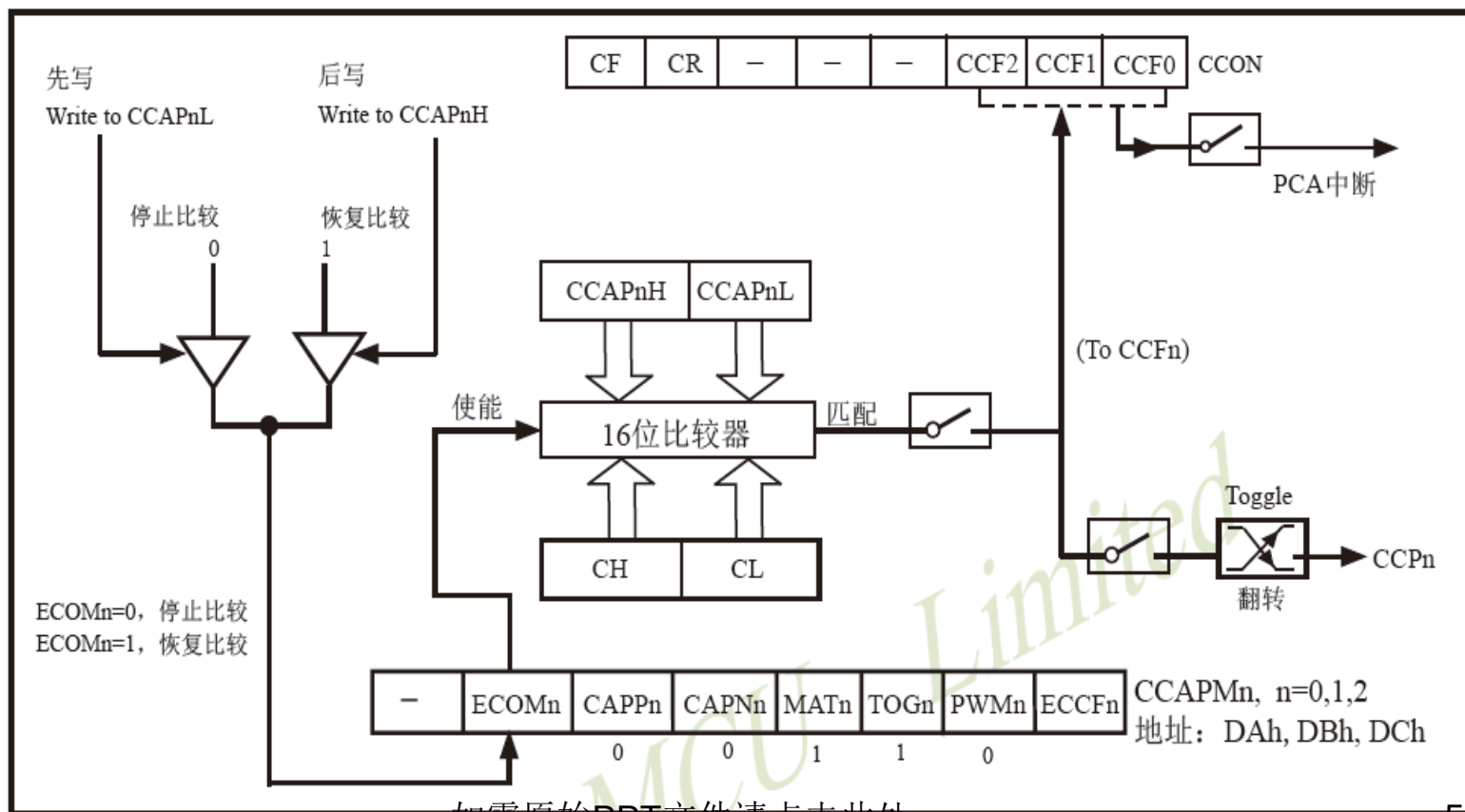
--16位软件定时器模式

```
CCAP0L=value;           //value低8位赋值给CCAP0L寄存器
CCAP0H=value>>8;        //value高8位赋值给CCAP0H寄存器
CCAPM0=0x49;             //打开比较器，使能匹配控制，使能CCF0中断
CR=1;                    //启动PCA定时器/计数器
EA=1;                    //CPU允许响应中断请求
while(1);
}
```

CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

高速脉冲输出模式结构



CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

- 当PCA计数器的计数值与模块捕获寄存器的值匹配时，PCA模块的CCPn输出将发生翻转。
- 当CCAPMn寄存器的TOGn位、MATn位和ECOMn位都置为1时，PCA模块工作在高速脉冲模式。
- CCAPnL的值决定了PCA模块n的输出脉冲频率。
 - 当PCA时钟源是SYSclk/2时，输出脉冲的频率为：
$$f = \text{SYSclk} / (4 \times \text{CCAPnL})$$
 - 因此，就可以得到对应的CCAPnL寄存器的值。

CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

【例】高速脉冲输出模式C语言描述的例子

```
#include "reg51.h"
```

```
#define value 3906
```

```
sfr P_SW1    =0xA2;    //声明P_SW1寄存器的地址0xA2
```

```
sfr CCON     =0xD8;    //声明CCON寄存器的地址0xD8
```

```
sfr CMOD     =0xD9;    //声明CMOD寄存器的地址0xD9
```

```
sfr CL       =0xE9;    //声明CL寄存器的地址0xE9
```

```
sfr CH       =0xF9;    //声明CH寄存器的地址0xF9
```

```
sfr CCAPM0   =0xDA;    //声明CAPM0寄存器的地址0xDA
```

```
sfr CCAP0L   =0xEA;    //声明CCAP0L寄存器的地址0xEA
```

```
sfr CCAP0H   =0xFA;    //声明CCAP0H寄存器的地址0xFA
```

```
sfr CLK_DIV  =0x97;    //声明CLK_DIV寄存器的地址0x97
```

如需原始PPT文件请点击此处
<http://www.gpnewtech.com/ppt>

CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

```
sbit CCF0  =CCON^0;           //声明CCON寄存器的CCF0位
sbit CCF1  =CCON^1;           //声明CCON寄存器的CCF1位
sbit CR    =CCON^6;           //声明CCON寄存器的CR位
sbit CF    =CCON^7;           //声明CCON寄存器的CF位
void PCA_int() interrupt 7     //声明中断服务程序
{
    CCF0=0;                    //CCF0标志清零
    CL=0;                      //CL寄存器清零
    CH=0;                      //CH寄存器清零
    P46=!P46;                  //P4.6端口取反
    P47=!P47;                  //P4.7端口取反
}
```

CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

```
void main()
```

```
{
```

```
P46=0;           //P4.6端口置0，灯亮
P47=0;           //P4.7端口置0，灯亮
CLK_DIV=0x07;    //SYSclk频率=主时钟频率/128
P_SW1=0x00;      //CCP_S0=0，CCP_S1=0
CCON=0;          //停止PCA定时器，清除CF和CCF0位
CL=0;            //CL寄存器清空
CH=0;            //CH寄存器清空
CMOD=0x00;       //设置时钟源，SYSclk/12
```

CCP/PCA/PWM工作模式

-- 高速脉冲输出模式

```
CCAP0L=value;      //value的低8位赋值给CCAP0L
CCAP0H=value>>8;   //value的高8位赋值给CCAP0H
CCAPM0=0x4d;        //PCA模块为16位定时器模式，同时翻转CCP0
CR=1;               //启动PCA定时器
EA=1;               //CPU允许响应中断请求
while(1);

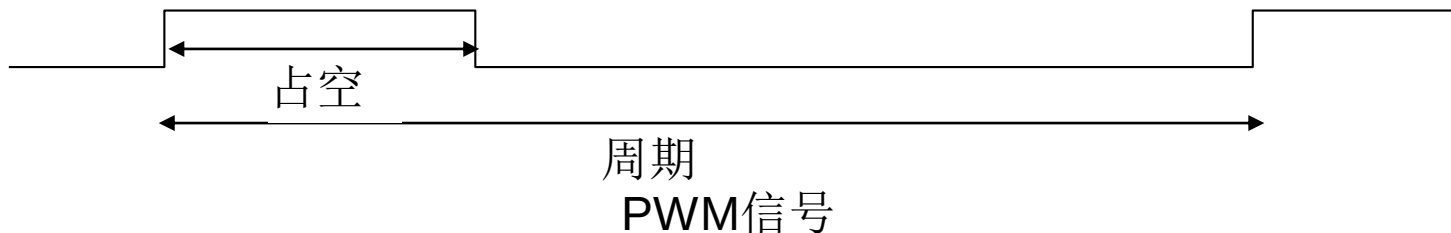
}
```

CCP/PCA/PWM工作模式

--脉冲宽度调制模式

脉冲宽度调试原理

- 使用数字电路来控制电机的速度，通常使用脉冲宽度调制PWM信号波形。
- 在PWM中，脉冲周期是恒定的。
- 在一个脉冲周期内，将高电平的时间称为占空，占空是可变的。



□ 占空比表示为:

$$\text{占空比} = \frac{\text{占空}}{\text{脉冲周期}} \times 100\%$$

如需原始PPT文件请点击此处
<http://www.gpnewtech.com/ppt>

CCP/PCA/PWM工作模式

--脉冲宽度调制模式

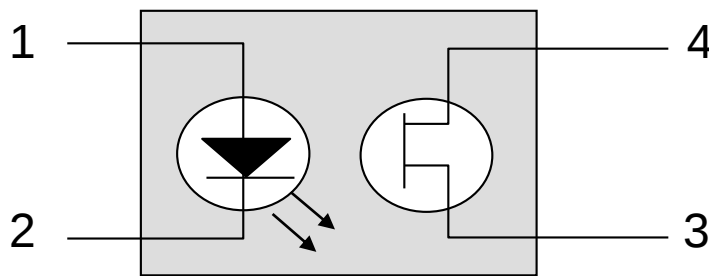
- PWM信号的直流平均值与占空是成比例的。
 - 一个占空比为50%的PWM信号，其直流值为PWM信号幅度最大值的1/2。
- 如果通过电机的电压与PWM成正比，简单的改变脉冲占空比就可以改变电机的速度。

CCP/PCA/PWM工作模式

--脉冲宽度调制模式

- 当连接电机或其他负载时，可能向数字电路（CPLD，FPGA和微处理器）流入很大的电流，因此最安全和最容易的方法是使用一些类型的固态继电器（solid-state relay, SSR）

- 数字电路提供小的电流（5-10mA）到输入引脚1和2，将开启固态继电器内的LED，来自LED的光将打开MOSFET，这样将允许引脚3和4之间流经很大的电流。
- 这种光电耦合电路将数字电路隔离开，这样可以降低电路的噪声和防止对数字电路造成的破坏。



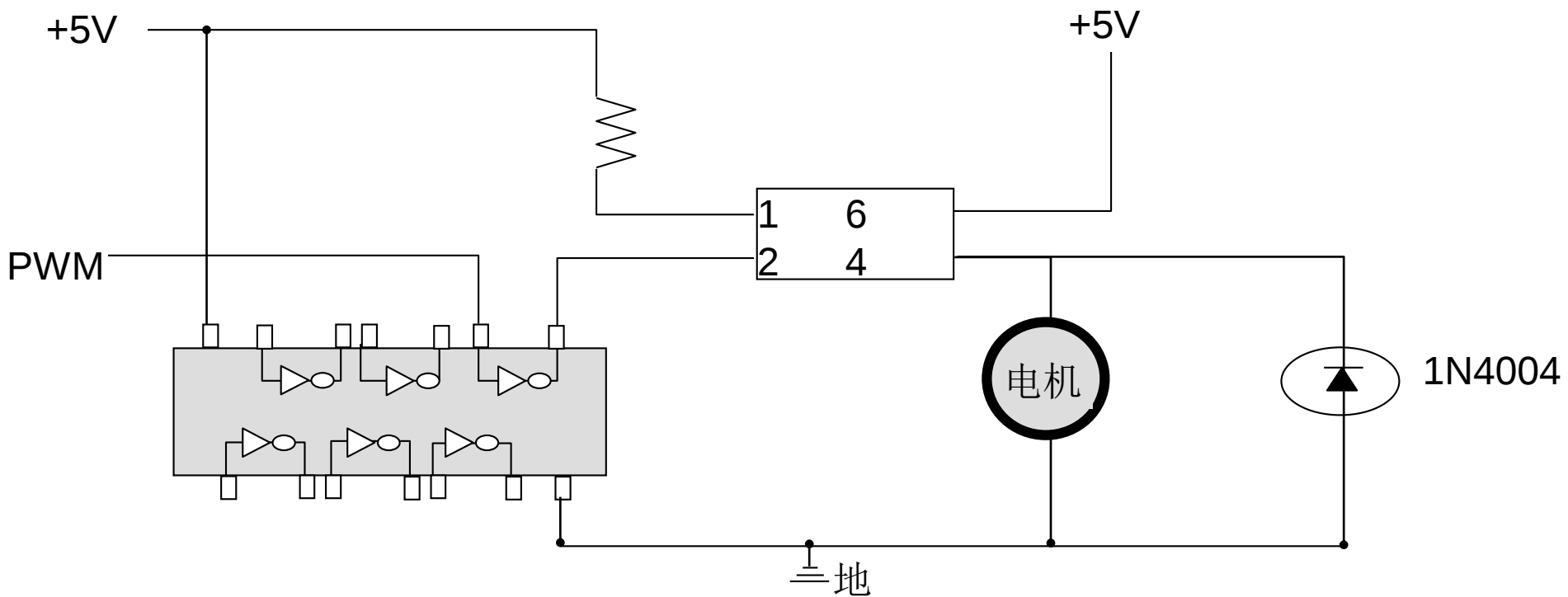
CCP/PCA/PWM工作模式

--脉冲宽度调制模式

- SSR适用于控制直流负载。然而一些SSR有两个MOSFET和背对背的二极管控制交流负载。当使用直流或交流SSR时，需要确认SSR能处理所使用的电压和电流负载。通常需要为电机提供独立的电源，将两个地连在一起。
 - 一个例子是使用G3VM-61B1/E1固态继电器SSR，该SSR是欧姆龙公司的一个6脚的MOSFET继电器，能用作直流或交流SSR。最大的交流负载电压是60V，最大负载电流是500mA（将两个MOSFET并联后可为直流负载提供1A电流）。

CCP/PCA/PWM工作模式

--脉冲宽度调制模式



PWM控制直流电机电路

CCP/PCA/PWM工作模式

--脉冲宽度调制模式

- 直流电机的速度取决于电机的电压，电压越高，电机转动的越快。
 - 如果需要电机以恒定的速度旋转，将4脚和电源连接起来，将电机连在3脚和地之间（也可以将3脚连接到地，或者连接电机到第四引脚和电源之间）。
 - 连接到电机的电源极性决定了电机的转动方向。如果转动方向错误，只需要将连接电机的两端对调。

CCP/PCA/PWM工作模式

--脉冲宽度调制实现

通过设置PCA各个模块CCAPM_n寄存器的PWM_n和ECOM_n比特位，使得PCA模块工作在PWM模式。

- 此外，通过设置PCA模块各自PCA_PWM_n (n=0,1,2) 寄存器中的EBS_n_1以及EBS_n_0比特位，使得PCA模块工作在8位、7位或者6位PWM模式。

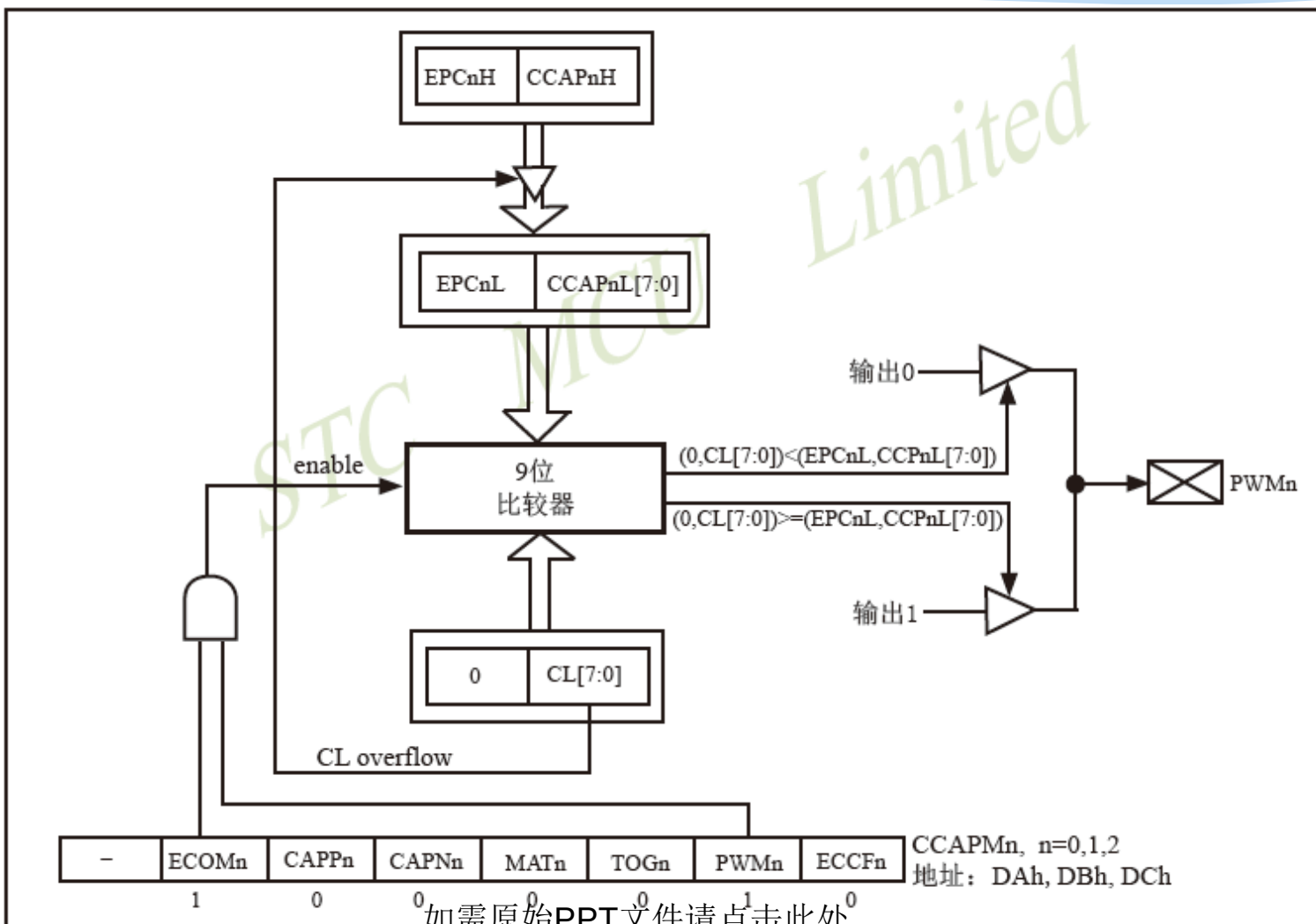
脉冲宽度调制实现

--8位脉冲宽度调节方式

- 当设置[EBSn_1, EBSn_0]=[0, 0]或者[1, 1]时, PCA模块工作在8位PWM模式。
 - {0, CL[7:0]}与捕获寄存器{EPCnL, CCAPnL[7:0]}进行比较。

脉冲宽度调制实现

--8位脉冲宽度调节方式



如需原始PPT文件请点击此处

<http://www.gpnewtech.com/ppt>

脉冲宽度调制实现

--8位脉冲宽度调节方式

当PCA模块工作于8位模式时，由于所有模块共用仅有的PCA定时器，因此它们的输出频率相同。

■ 每个模块的占空比各自独立，只与该模块的捕获寄存器{EPCnL, CCAPnL[7:0]}有关，即：

□ 当{0, CL[7:0]}的值<{EPCnL, CCAPnL[7:0]}时，输出为低；

□ 当{0, CL[7:0]}的值≥{EPCnL, CCAPnL[7:0]}时，输出为高；

■ 当CL的值由FF变成00溢出时，将{EPCnH, CCAPnH[7:0]}的内容加载到{EPCnL, CCAPnL[7:0]}中。因此，就可以实现无干扰更新PWM。

□ 在8位模式下，PWM的频率由下式确定：

$$f_{\text{PWM}} = \frac{\text{PCA时钟输入源频率}}{256}$$

如需原始PPT文件请点击此处
<http://www.gpnewtech.com/ppt>

脉冲宽度调制实现

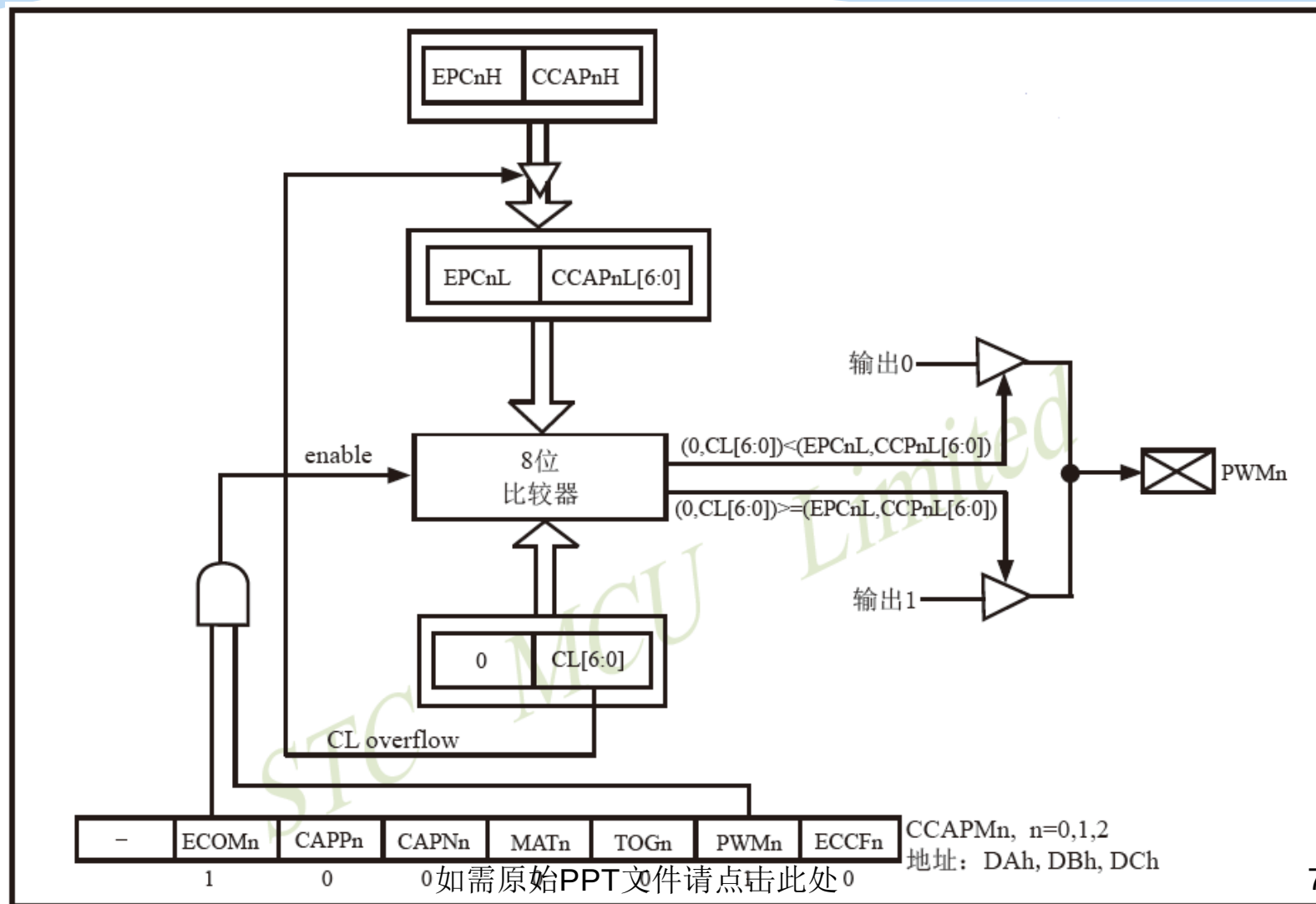
--7位脉冲宽度调节方式

当设置[EBSn_1 , EBSn_0]= [0 , 1]时 , PCA模块工作在7位PWM模式

□ 此时 , {0,CL[6:0]}与捕获寄存器{EPCnL,CCAPnL[6:0]}进行比较。

脉冲宽度调制实现

--7位脉冲宽度调节方式



脉冲宽度调制实现

--7位脉冲宽度调节方式

当PCA模块工作于7位模式时，由于所有模块共用仅有的PCA定时器，因此它们的输出频率相同。

- 每个模块的占空比各自独立，只与该模块的捕获寄存器{EPCnL, CCAPnL[6:0]}有关，即：
 - 当{0, CL[6:0]}的值<{EPCnL, CCAPnL[6:0]}时，输出为低；
 - 当{0, CL[6:0]}的值≥{EPCnL, CCAPnL[6:0]}时，输出为高；
- 当CL的值由7F变成00溢出时，将{EPCnH, CCAPnH[6:0]}的内容加载到{EPCnL, CCAPnL[6:0]}中。因此，就可以实现无干扰更新PWM。
 - 在7位模式下，PWM的频率由下式确定：

脉冲宽度调制实现

--6位脉冲宽度调节方式

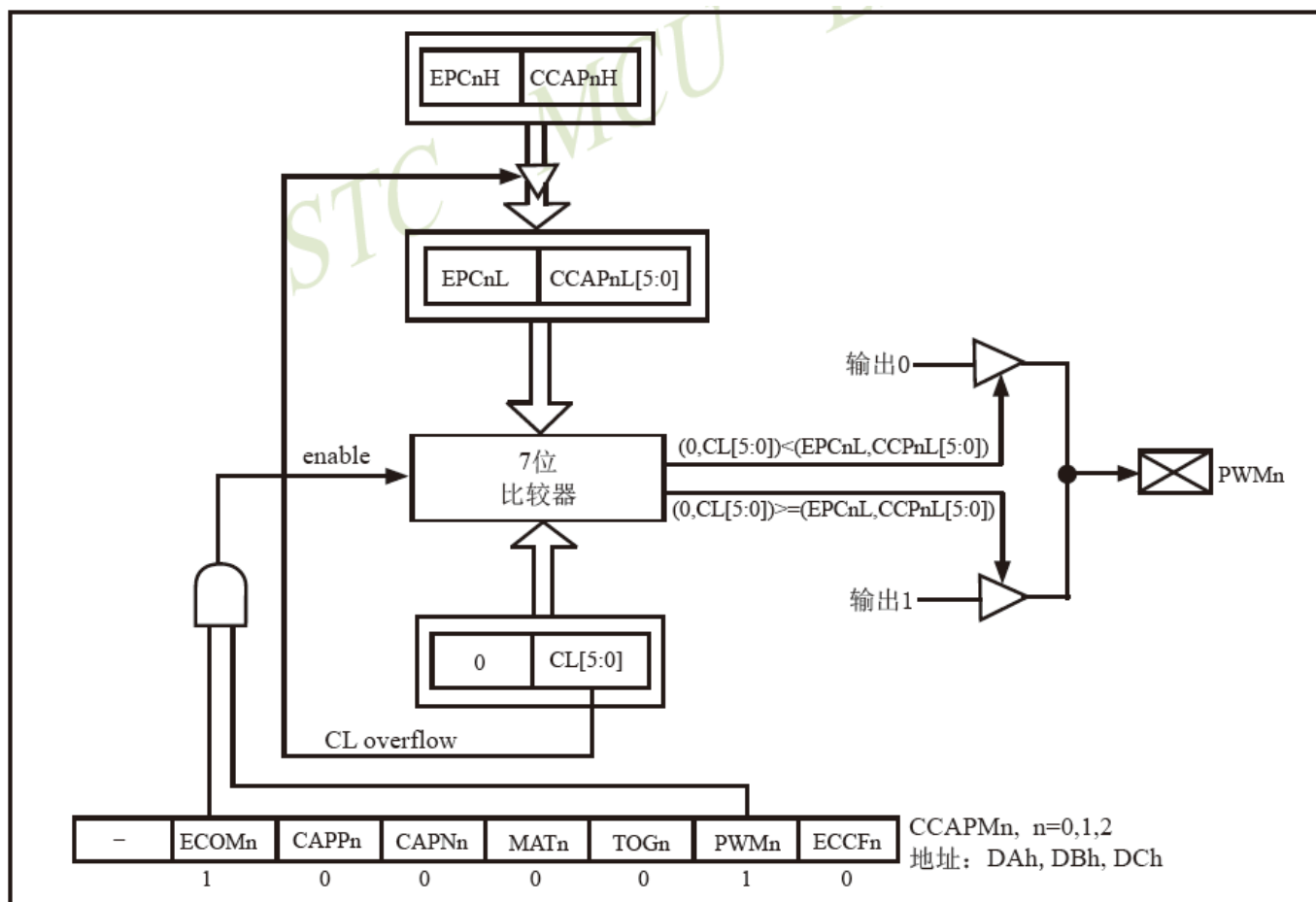


当设置[EBSn_1 , EBSn_0]= [1 , 0]时 , PCA模块工作在6位PWM模式。

- 此时 , {0,CL[5:0]}与捕获寄存器{EPCnL,CCAPnL[5:0]}进行比较。

脉冲宽度调制实现

--6位脉冲宽度调节方式



脉冲宽度调制实现

--6位脉冲宽度调节方式

当PCA模块工作于6位模式时，由于所有模块共用仅有的PCA定时器，因此它们的输出频率相同。

■ 每个模块的占空比各自独立，只与该模块的捕获寄存器{EPCnL, CCAPnL[5:0]}有关，即：

□ 当{0, CL[5:0]}的值<{EPCnL, CCAPnL[5:0]}时，输出为低

□ 当{0, CL[5:0]}的值≥{EPCnL, CCAPnL[5:0]}时，输出为高

■ 当CL的值由3F变成00溢出时，将{EPCnH, CCAPnH[5:0]}的内容加载到{EPCnL, CCAPnL[5:0]}中。因此，就可以实现无干扰更新PWM。

□ 在6位模式下，PWM的频率由下式确定：

$$f_{\text{PWM}} = \text{PCA时钟输入源频率} / 64$$

如需原始PPT文件请点击此处
<http://www.gpnewtech.com/ppt>

CCP/PCA/PWM工作模式

【例】脉冲宽度调制模式C语言描述的例子

```
#include "reg51.h"
```

```
#define value 3906
```

```
sfr P_SW1    =0xA2;
```

```
//声明P_SW1寄存器的地址0xA2
```

```
sfr CCON     =0xD8;
```

```
//声明CCON寄存器的地址0xD8
```

```
sfr CMOD     =0xD9;
```

```
//声明CMOD寄存器的地址0xD9
```

```
sfr CL       =0xE9;
```

```
//声明CL寄存器的地址0xE9
```

```
sfr CH       =0xF9;
```

```
//声明CH寄存器的地址0xF9
```

```
sfr CCAPM0   =0xDA;
```

```
//声明CCAMP0寄存器的地址0xDA
```

```
sfr CCAP0L   =0xEA;
```

```
//声明CCAP0L寄存器的地址0xEA
```

```
sfr CCAP0H   =0xFA;
```

```
//声明CCAP0H寄存器的地址0xFA
```

CCP/PCA/PWM工作模式

```
sfr PCA_PWM0 =0xF2;           //声明PCA_PWM0寄存器的地址0xF2
sbit CCF0    =CCON^0;         //声明CCON寄存器的CCF0比特位
sbit CCF1    =CCON^1;         //声明CCON寄存器的CCF1比特位
sbit CR      =CCON^6;         //声明CCON寄存器的CR比特位
sbit CF      =CCON^7;         //声明CCON寄存器的CF比特位
unsigned char i=0;             //声明无符号的8位全局变量i
void PCA_int() interrupt 7     //声明PCA中断服务程序
{
    CF=0;                      //CF标志清零
    CCAP0H=i;                  //i赋值给CCAP0H寄存器
    CCAP0L=i;                  //i赋值给CCAP0L寄存器
    if(i<255) i++;             //如果i<255 , 则i递增
    else i=0;                  //否则 , i=0
}
```

CCP/PCA/PWM工作模式

```
void main()
```

```
{
```

```
    P_SW1=0x00;           //CCP_S0=0 , CCP_S1=0
    CCON=0;                //停止PCA定时器 , 清除CF和CCF0标志
    CL=0;                  //CL寄存器清零
    CH=0;                  //CH寄存器清零
    CMOD=0x03;             //时钟源主时钟/2 , 允许CF中断
    PCA_PWM0=0x00;         //PCA模块0 , 工作于8位的PWM模式
    CCAP0L=0x10;           //CCAP0L赋初值0x10
    CCAP0H=0x10;           //CCAP0H赋初值0x10
    CCAPM0=0x42;           //使能比较 , 脉宽调制模式
```

CCP/PCA/PWM工作模式

CR=1;

//启动PCA定时器工作

EA=1;

//CPU允许响应中断请求

while(1);

}