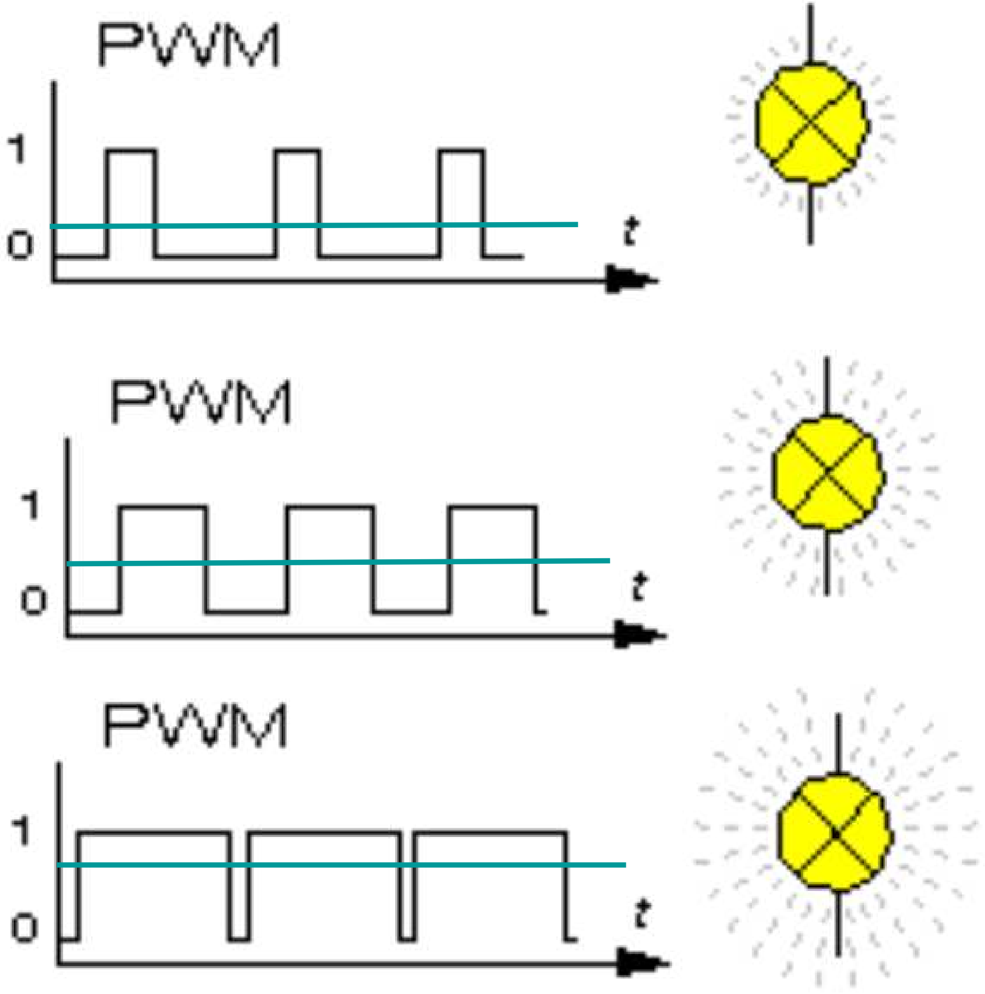
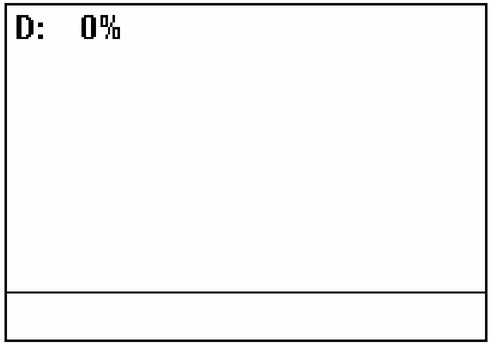
DA-conversion, usually PWM



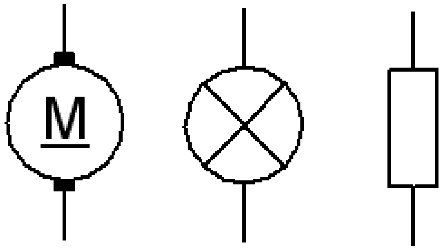
A DA converter takes too much space on the processor chip. The most common DA solution is instead a pulse width modulator. Many components "notice" no difference between a stable analog value or the mean value of fast pulses.

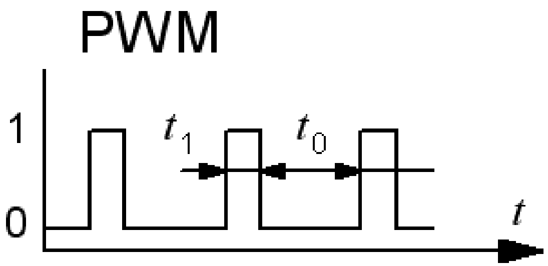
William Sandqvist william@kth.se

DA-conversion, usually PWM



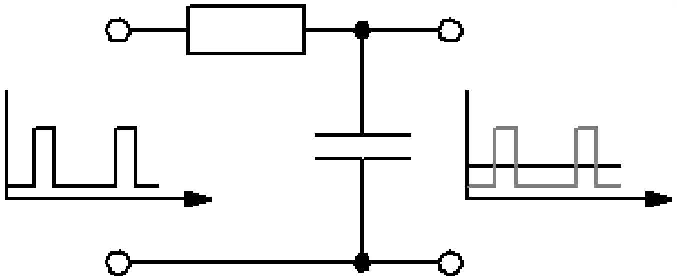
Components with inertia
= mean value



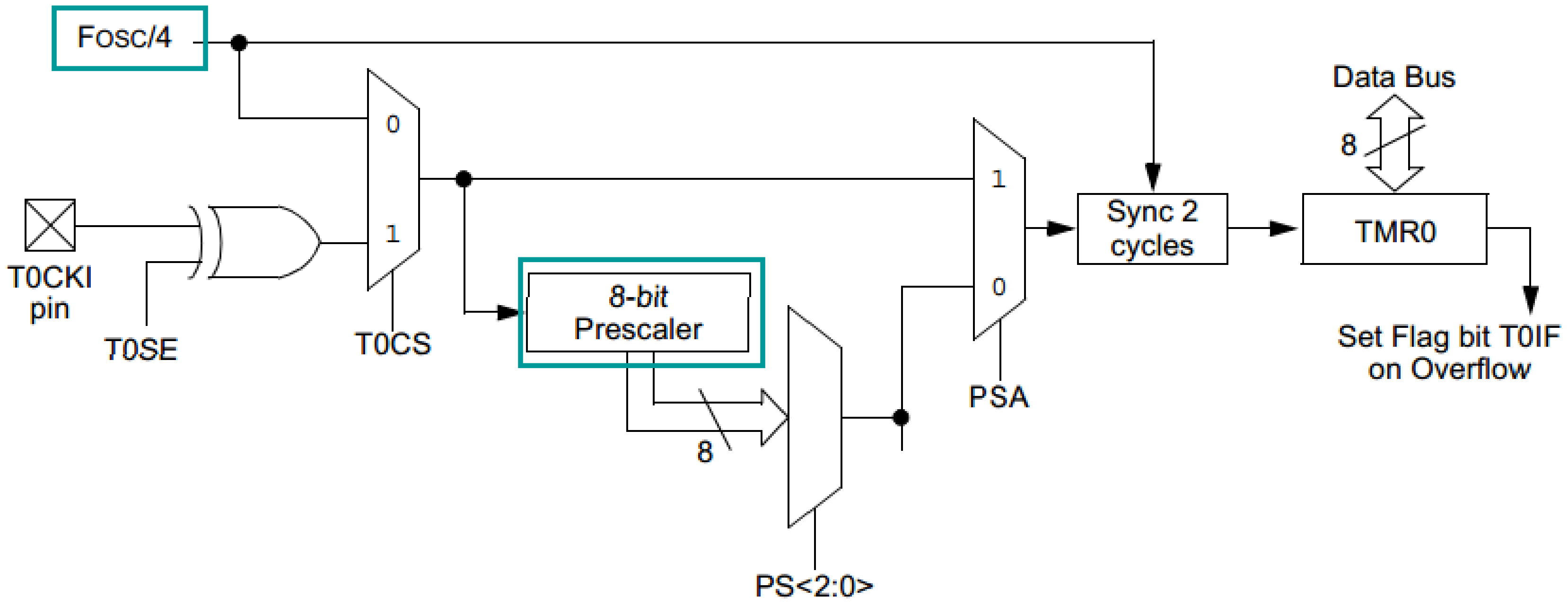


$$\text{DutyCycle} = \frac{t_1}{t_1 + t_0}$$

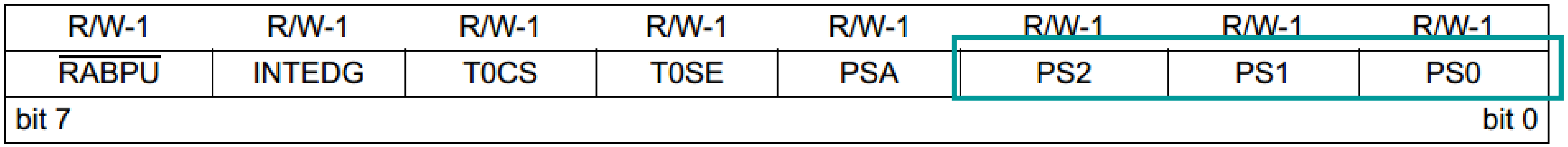
Low pass filter
= DC voltage



TIMER0



REGISTER 5-1: OPTION_REG: OPTION REGISTER



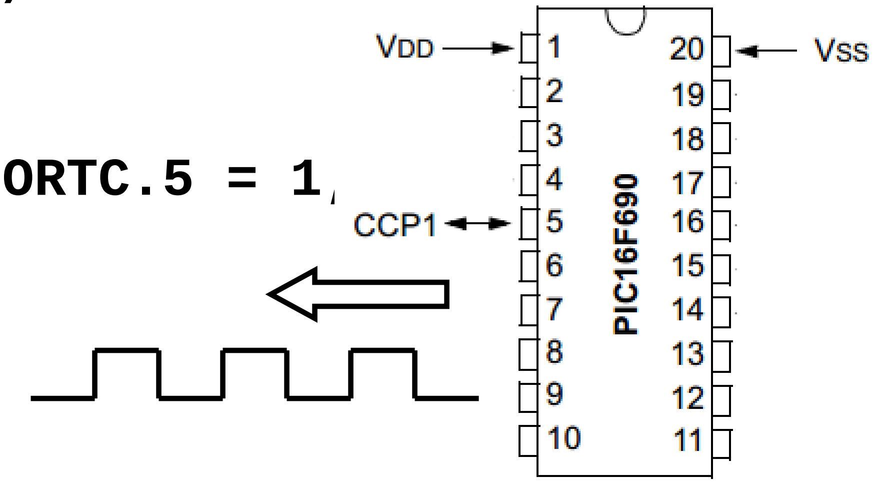
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RABPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

PWM-program

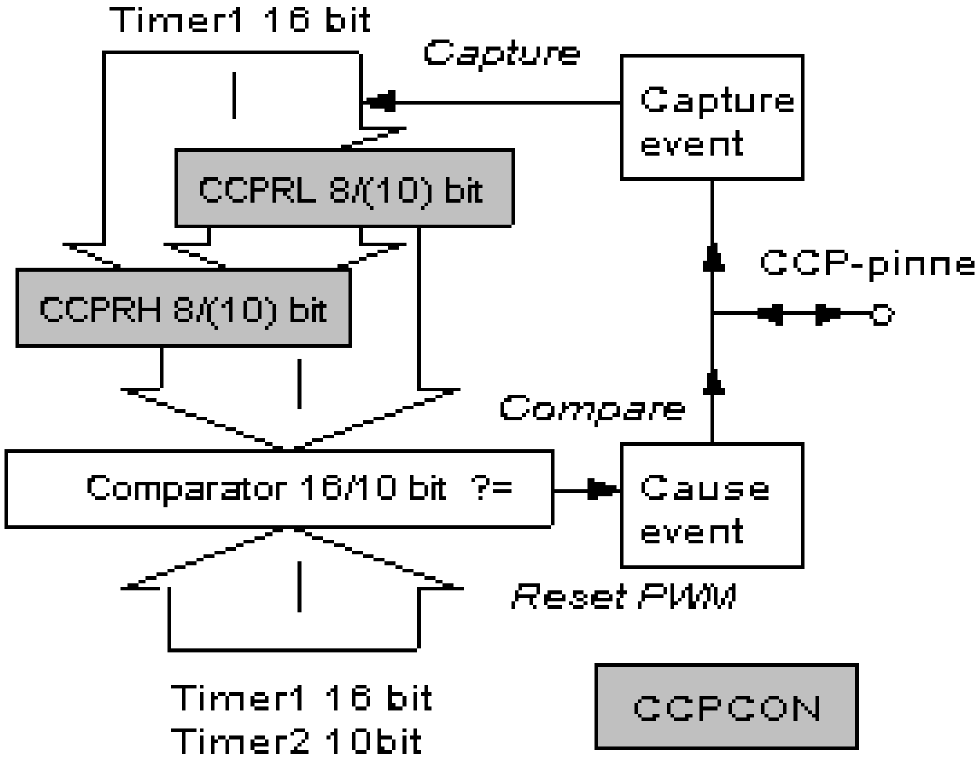
Problem. If the program is to do anyting more then it has to be done in between the TIMER0 tick'!

```
#define DUTY 128

void main( void)
{
  TRISC.5 = 0;          /* PORTC.5 is output */
  OPTION = 0b10000.111; /* 256 prescale      */
  while (1) /* forever */
   {
     char i;
     if (TMR0 < DUTY ) PORTC.5 = 1;
     else PORTC.5 = 0;
   }
}
```



CCP-unit



CCP

- **Capture**
- **Compare**
- **PWM**

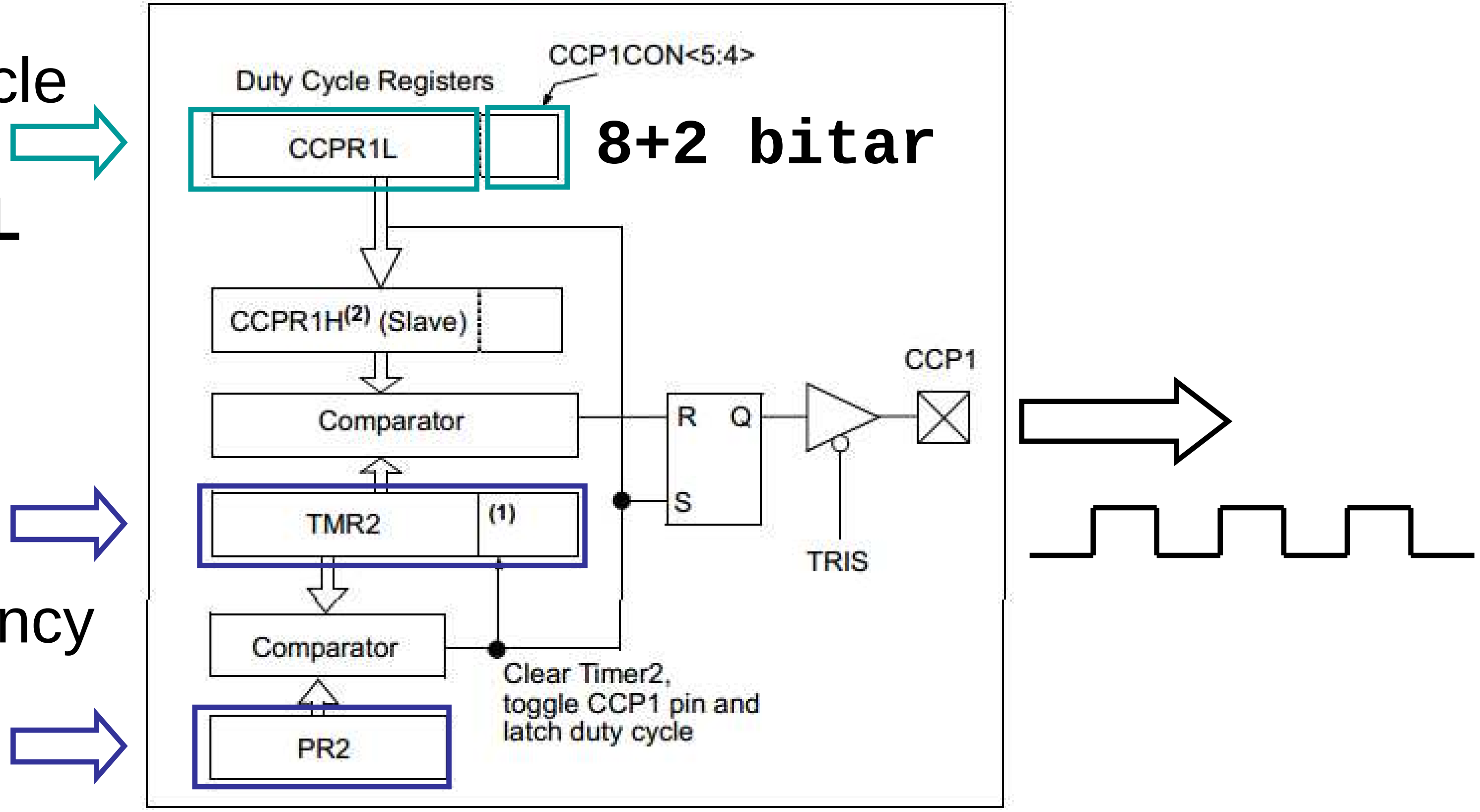
There is an obvious need for a stand alone unit for generating PWM. CCP unit can be programmed to this!

PWM

FIGURE 11-3: SIMPLIFIED PWM BLOCK DIAGRAM

DutyCycle

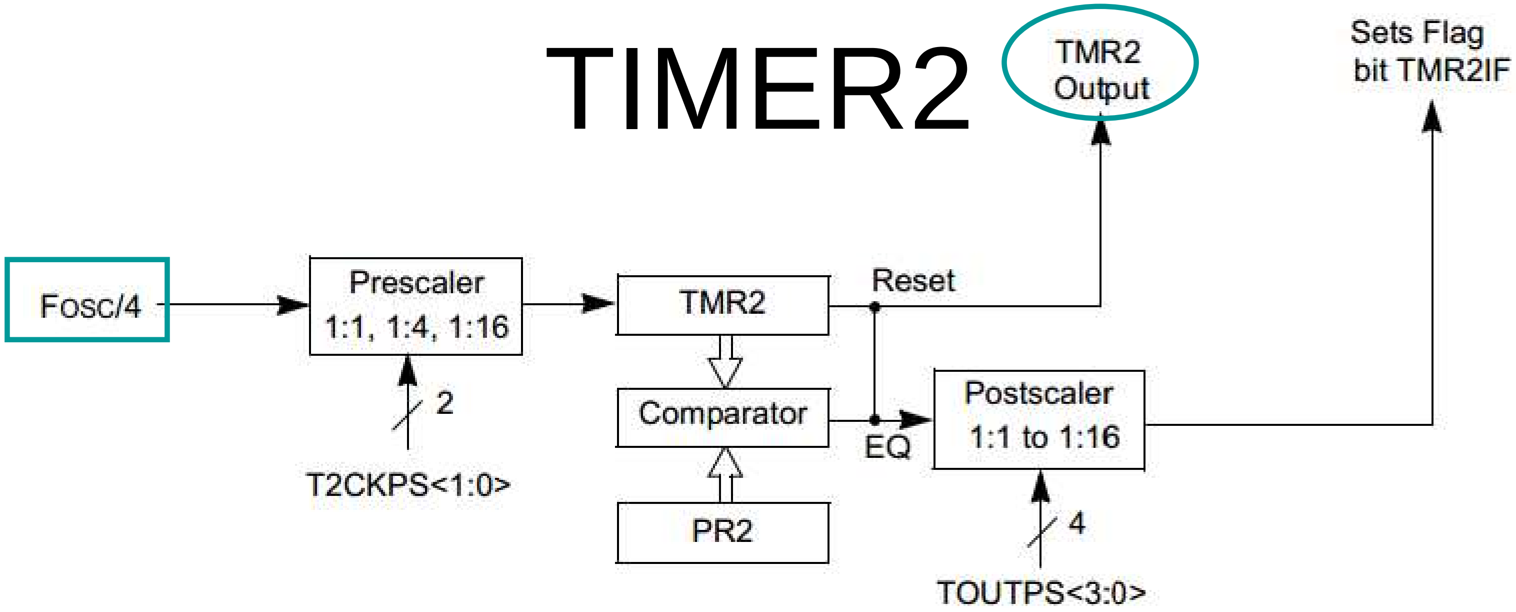
CCPR1L



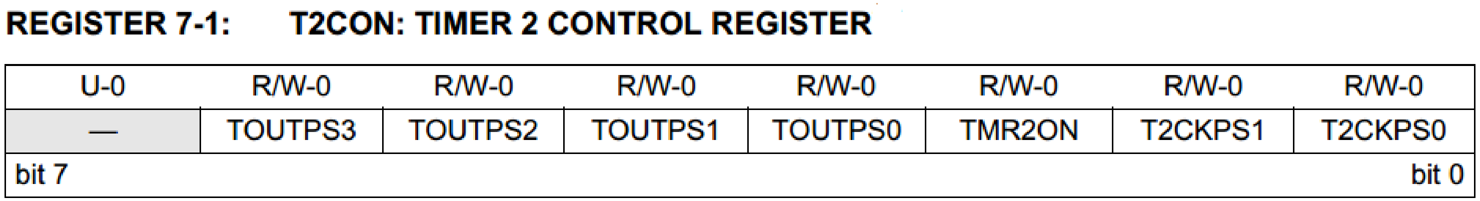
8+2 bitar

Frequency
PR2

TIMER2



TIMER2 is a 8-bit counter (up to modulo 256). It has a **prescaler** from the processor clock, and a register **PR2** that can ”shorten” the count cycle – it will count ”modulo PR2”. This provides many opportunities to set TIMER2 output frequency.



REGISTER 7-1: T2CON: TIMER 2 CONTROL REGISTER

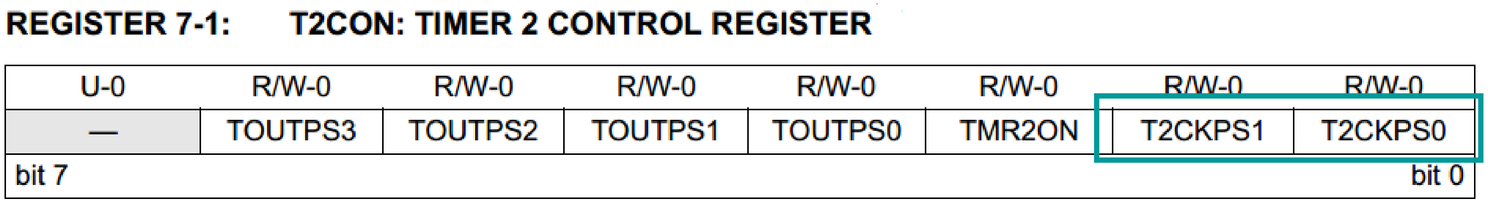
U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

PWM f 1kHz D 50%

Suppose we need to generate a PWM-signal with f **1 kHz** and the dutycycle **50%** (someone that likes nice numbers).

$$\frac{f_{osc}}{4} = \frac{4\cdot 10^6}{4} = 1\cdot 10^6 \qquad \frac{1\cdot 10^6}{\text{prescale}\{1\ \boxed{4}\ 16\}} = 250\cdot 10^3$$

$$\text{PR2} = 249 \quad [0\ldots249] \quad \frac{250\cdot 10^3}{249+1} = \boxed{1000\text{ Hz}}$$

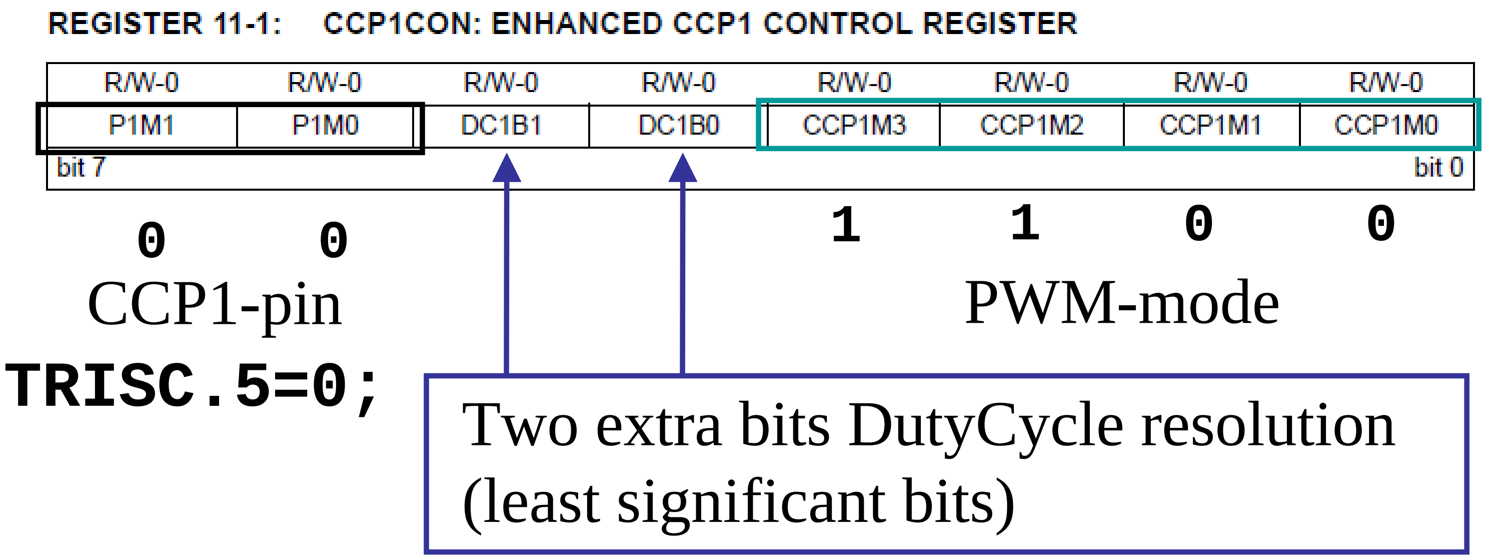


REGISTER 7-1: T2CON: TIMER 2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

- - - - 1 0 1

PWM *f* 1kHz *D* 50%



REGISTER 11-1: CCP1CON: ENHANCED CCP1 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

`0 0`

CCP1-pin

`TRISC.5=0;`

`1 1 0 0`

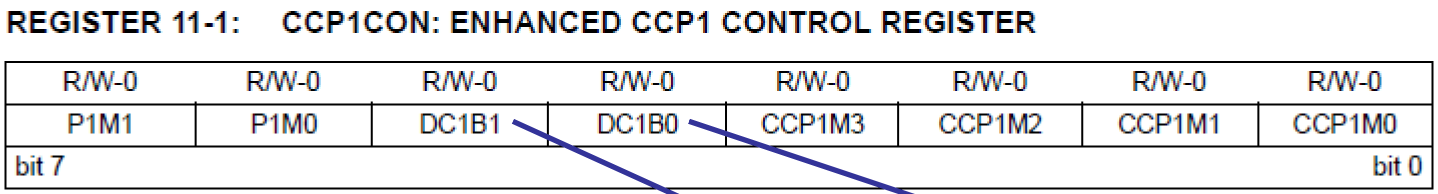
PWM-mode

Two extra bits DutyCycle resolution (least significant bits)

Ten bit resolution when PR2=255. Lower values reduce the resolution:

$$\text{resolution} = \frac{\log\left(4 \cdot (\text{PR}2+1)\right)}{\log(2)} \quad [\text{bits}]$$

PWM f 1kHz D 50%



REGISTER 11-1: CCP1CON: ENHANCED CCP1 CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

$$\mathrm{DutyCycle} = \frac{\boxed{\mathrm{CCPR1L}} \cdot 4 + \mathrm{DC1B1} \cdot 2 + \mathrm{DC1B0} \cdot 1}{(\mathrm{PR2} + 1) \cdot 4}$$

$$50\% = \frac{\mathrm{CCPR1L} \cdot 4 + 0 \cdot 2 + 0 \cdot 1}{(249 + 1) \cdot 4} \quad \Rightarrow \quad \mathrm{CCPR1L} = \frac{0{,}5 \cdot 4 \cdot 250}{4} = \boxed{125}$$

Ex. What DutyCycle?

```
PR2 = 208;
CCPR1L = 137;
DC1B1 = 1;
DC1B0 = 1;
```

?

$$\text{DutyCycle} = \frac{\text{CCPR1L} \cdot 4 + \text{DC1B1} \cdot 2 + \text{DC1B0} \cdot 1}{(\text{PR2} + 1) \cdot 4} =$$

$$= \frac{137 \cdot 4 + 1 \cdot 2 + 1 \cdot 1}{(208 + 1) \cdot 4} = 65{,}9\%$$

Ex. What DutyCycle?

```
PR2 = 208;
CCPR1L = 209;
DC1B1 = 1;
DC1B0 = 1;
```

?

$$\xcancel{\text{DutyCycle} = \frac{\text{CCPR1L} \cdot 4 + \text{DC1B1} \cdot 2 + \text{DC1B0} \cdot 1}{(\text{PR2} + 1) \cdot 4} =}$$

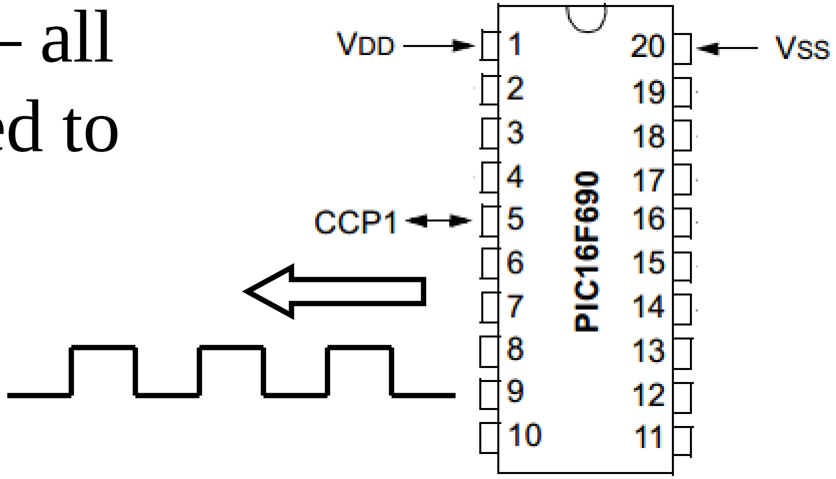
$$\xcancel{= \frac{209 \cdot 4 + 1 \cdot 2 + 1 \cdot 1}{(208 + 1) \cdot 4} = 100{,}4\%}$$

Does not work!
CCPR1L can never be bigger than **PR2**!

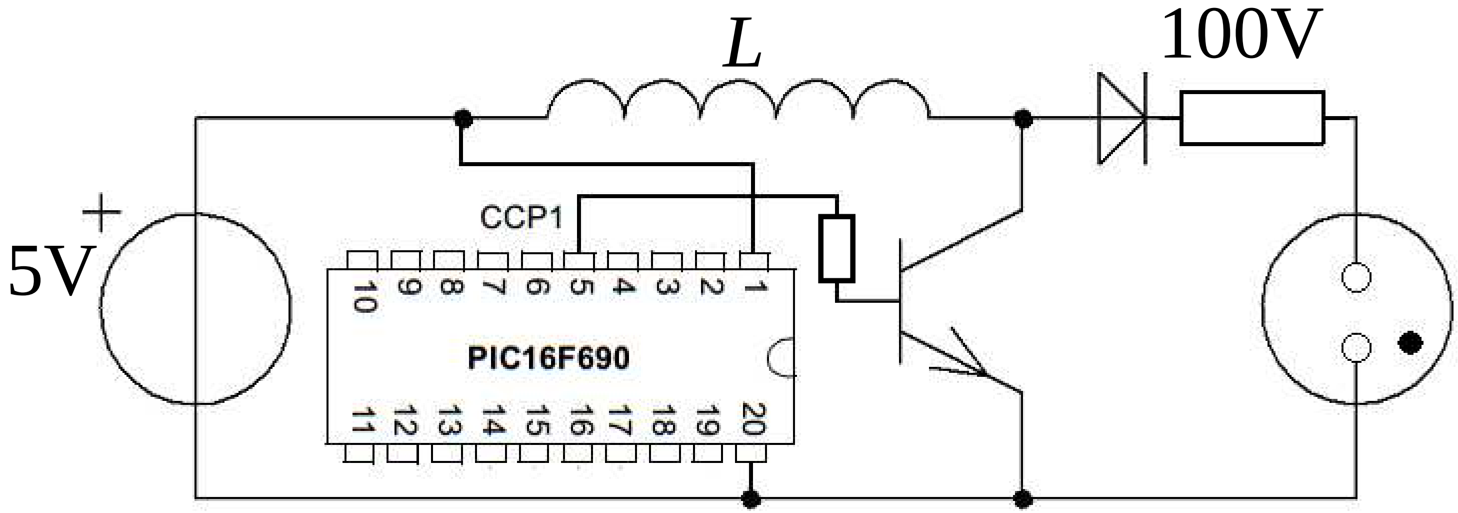
PWM *f* 1kHz *D* 50%

```
TRISC.5=0;                /* CCP1 output    */
T2CON = 0B00000101;       /* prescale 1:4   */
CCP1CON = 0B00.00.1100;   /* PWM-mode       */
PR2 = 249;                /* f_pwm 1000 Hz  */
CCPR1L = 125;             /* Duty 50%       */
```

CCP-unit is stand alone – all processortime can be used to other tasks.

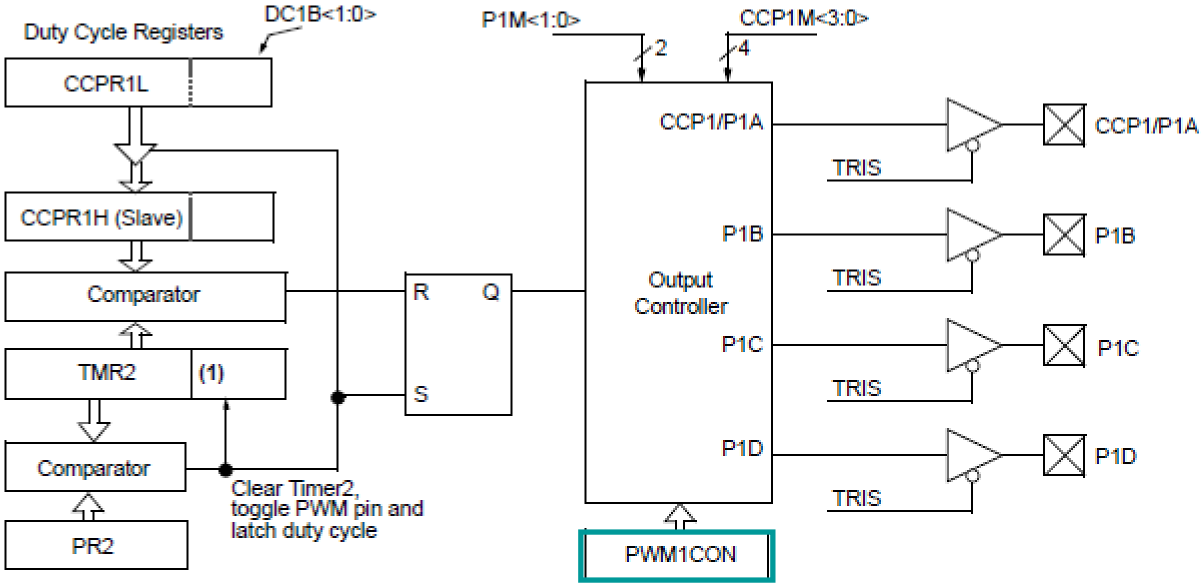


PWM to StepUp



At the lab, you will need 100V to light a neon lamp!

PWM to motors



Another common use for PWM is motor control - we will return to this.