CENTEC

INSTITUTO CENTRO DE ENSINO TECNOLÓGICO

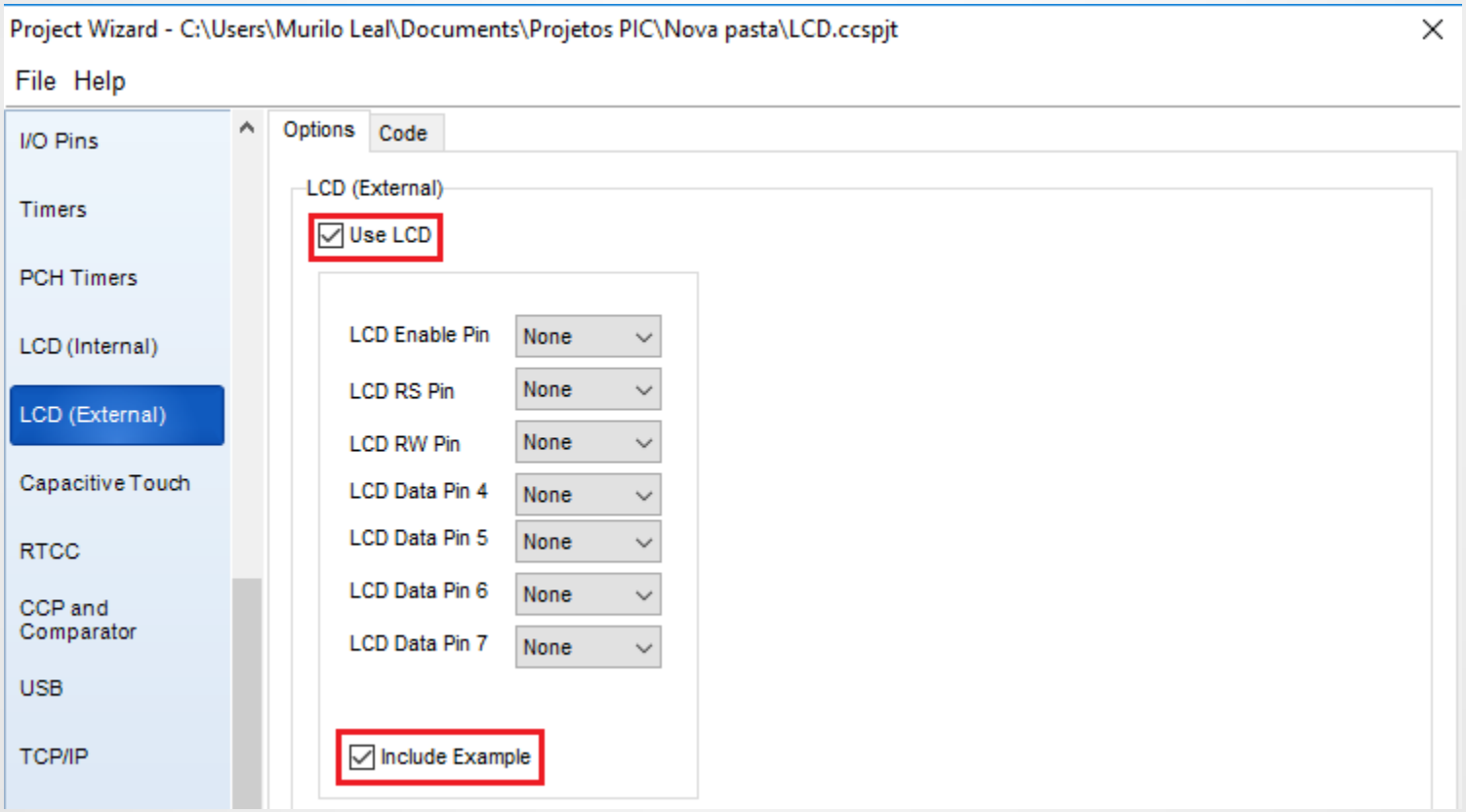
# Funções dos Microcontroladores

## Microcontroladores e Microprocessadores

## Especialização em Automação Industrial

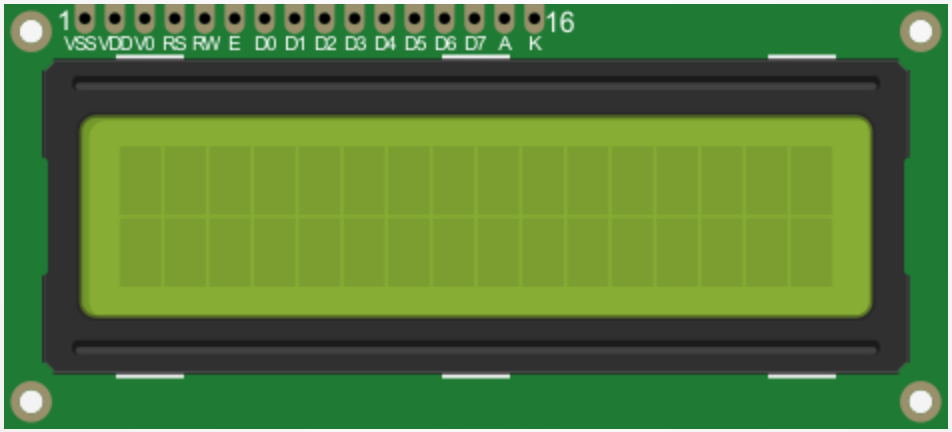
# LCD - PIC

•**Configure um projeto normalmente e na aba “LCD (External)” marque as opções “Use LCD” para inclusão da biblioteca do mesmo e “Include Example” caso queira iniciar com um código base.**



# Display de LCD - PIC

- O LCD alfanumérico é dividido em linhas e colunas. No caso do LCD de 16x2, temos 16 colunas e 2 linhas.



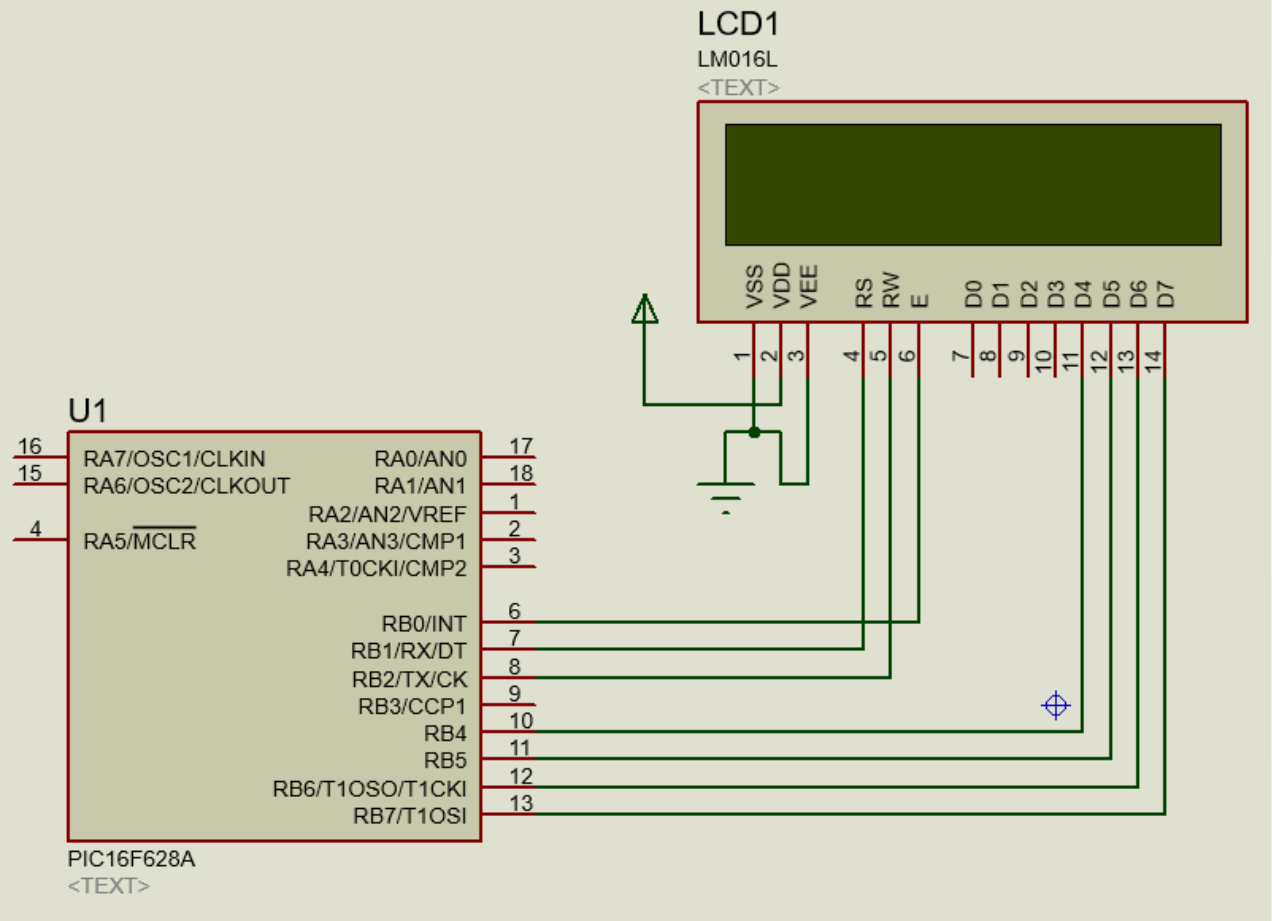
# LCD – Formatadores de Dados - PIC

•Para a correta exibição do texto que é exibido no display, se utilizam os seguintes códigos de formatação.

\n - Pulo de linha
\t - Tab
\b - Retrocesso
\" - Aspas
\\ - Barra invertida
\f - Limpa a tela
\0 - Nulo

%c - caractere simples
%d - decimal
%e - notação científica
%f - ponto flutuante
%o - octal
%s - cadeia de caracteres
%u - decimal sem sinal
%x - hexadecimal

# LCD – Esquemático - PIC

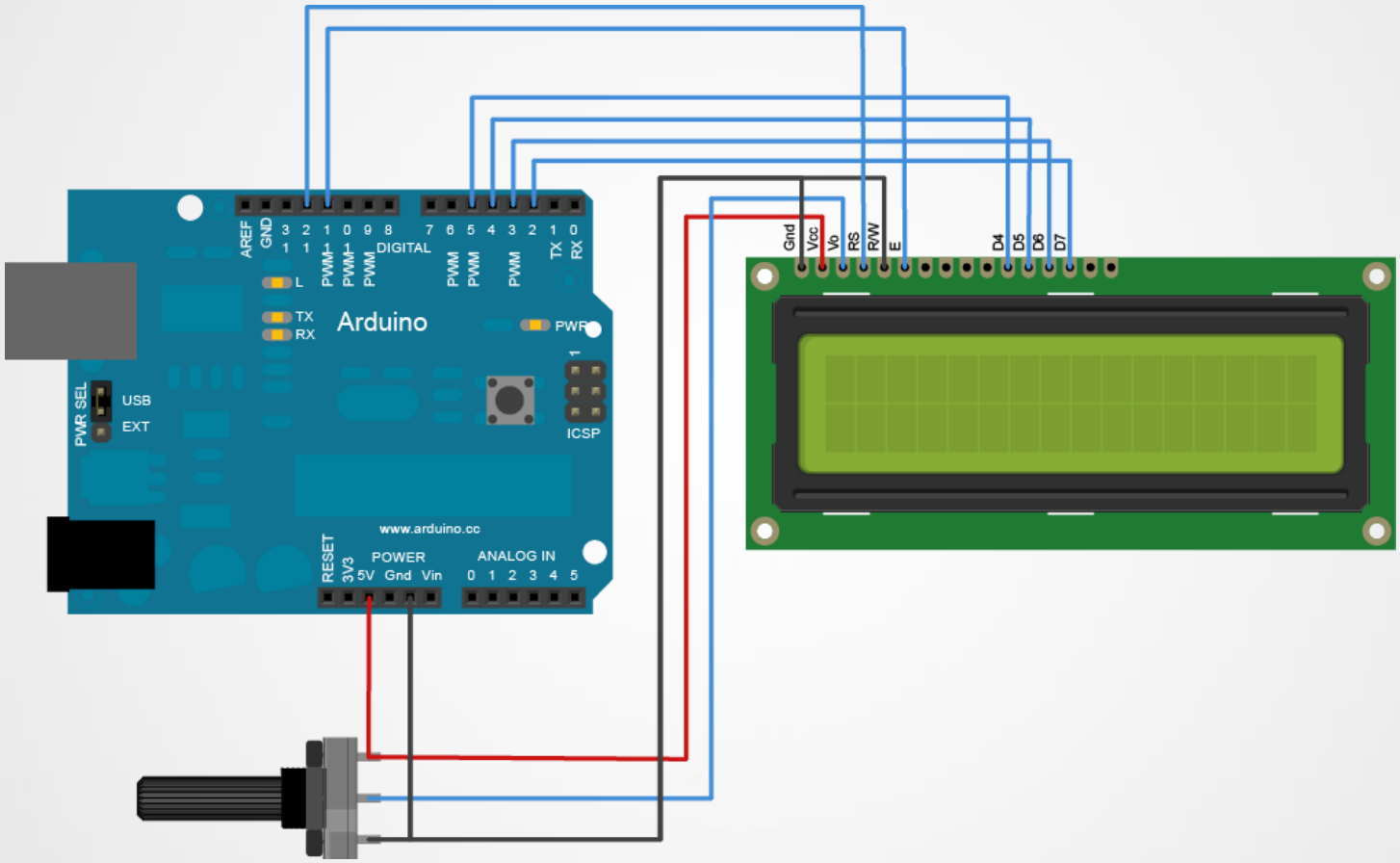


# LCD - Código – PIC

•Para a correta exibição do texto que é exibido no display, se utilizam os seguintes códigos de formatação.

```
#include <LCD.h> //Inclusão das bibliotecas do LCD
#include <lcd.c> //Inclusão das bibliotecas do LCD
void main()
{
   char k;
   lcd_init(); //Inicialização do LCD
   lcd_putc("\fReady...\n"); //Imprime “Ready...” no display
   while(TRUE)
   {
      k = kbd_getc();
      if(k!=0)
         if(k=='*')
            lcd_putc('\f');
         else
            lcd_putc(k);
      //TODO: User Code
   }
}
```

# LCD – Esquemático - Arduino



# LCD - Código – Arduino

**•Para a correta exibição do texto que é exibido no display, se utilizam os seguintes códigos de formatação.**

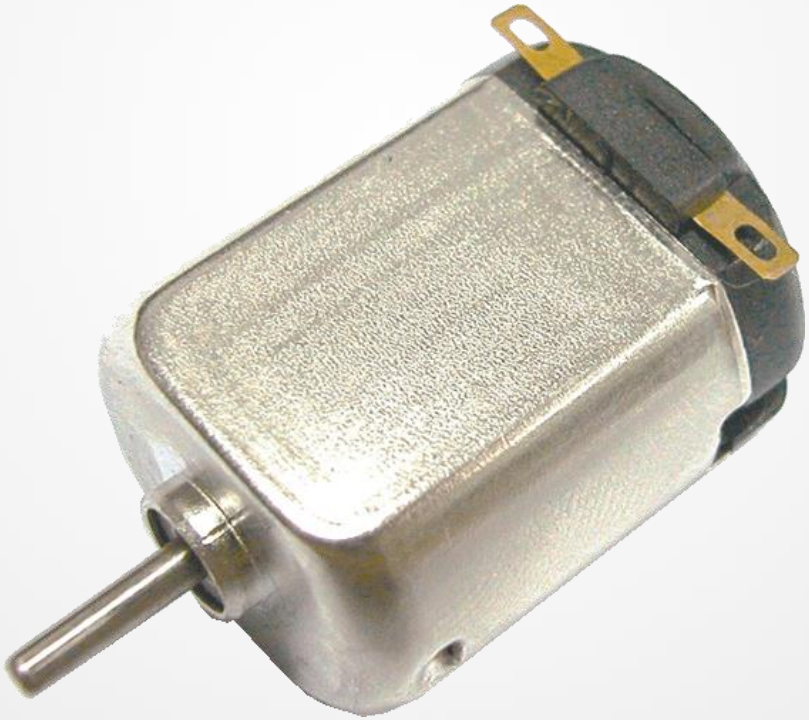
```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Define os pinos do Arduino a serem utilizados

void setup() {
  //Inicializa o display configurando o número de linhas e de colunas
  lcd.begin(16, 2);
  //Imprime um texto no display
  lcd.print("Ola, mundo!");
}

void loop() {
  //Posiciona o cursor na coluna 0, linha 1
  // (OBS: linha 1 é a segunda linha, a contagem inicia com 0):
  lcd.setCursor(0, 1);
  lcd.print(millis() / 1000);
}
```

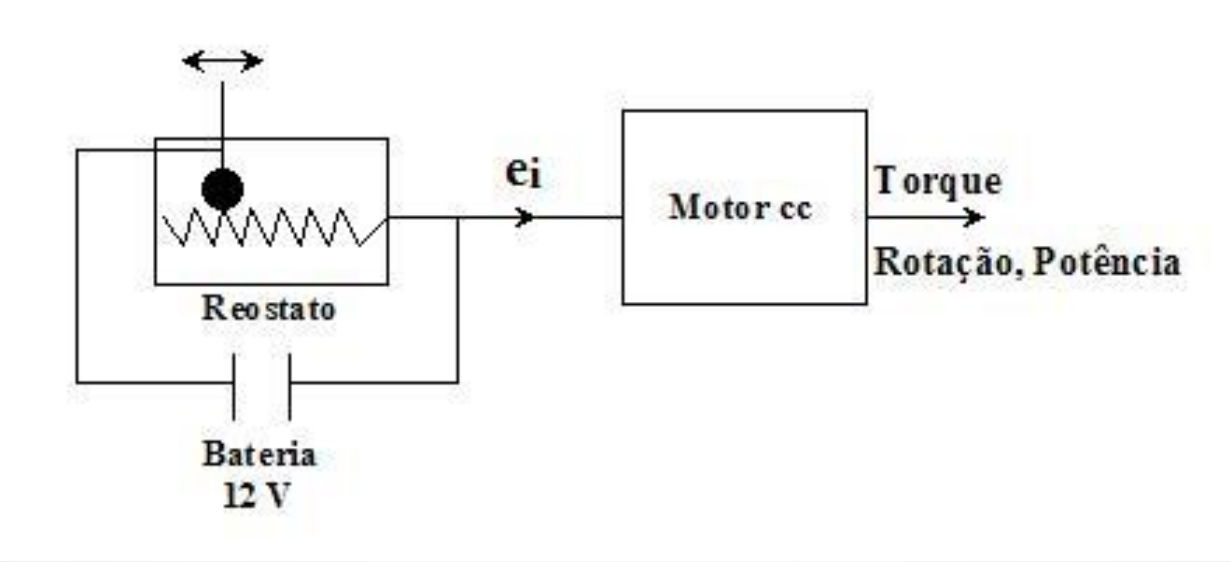
# PWM – Modulação de Largura de Pulso

**•PWM é a sigla para Pulse Width Modulation (Modulação por Largura de Pulso). É uma técnica utilizada para controle de intensidade de funcionamento de cargas de corrente contínua. Exemplo: Controle de velocidade de um motor DC.**



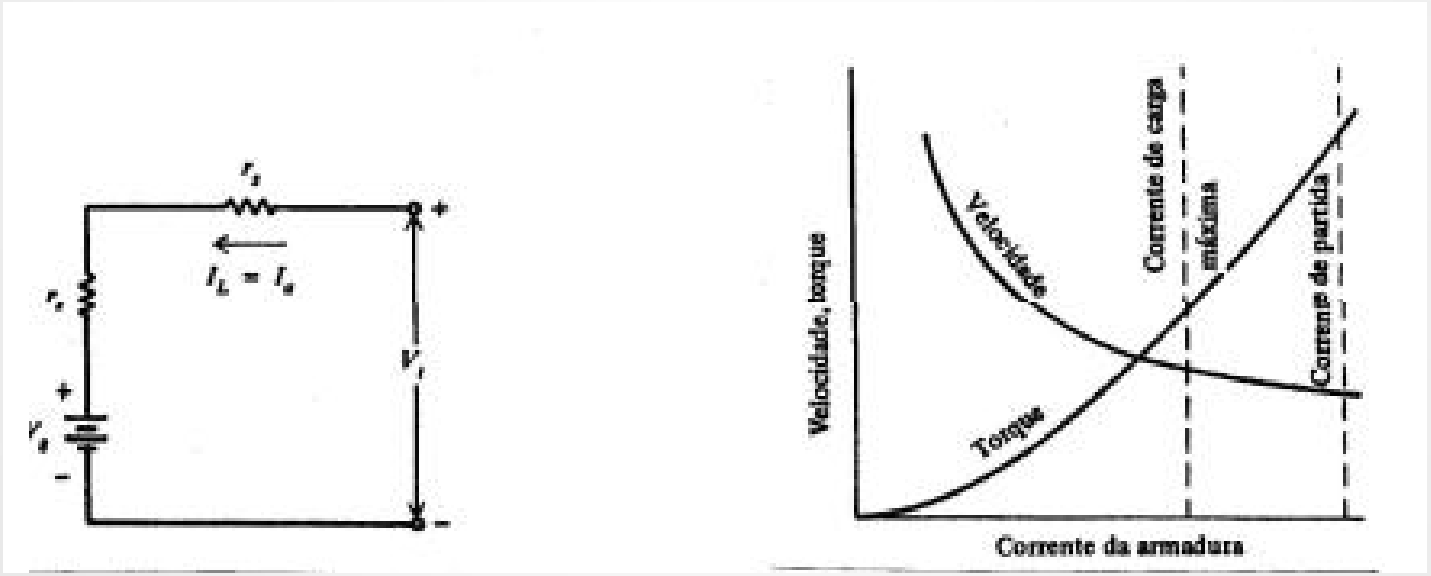
# PWM – Modulação de Largura de Pulso

- **Uma das formas de controlar a velocidade de um motor DC seria usar um reostato para limitar a corrente aplicada no mesmo.**



# PWM – Modulação de Largura de Pulso

•Por exemplo, se um motor é ligado a um reostato regulado no mínimo da sua capacidade, o motor gira no máximo de velocidade, à medida que se for aumentada a resistência, a velocidade cai inversamente proporcional.



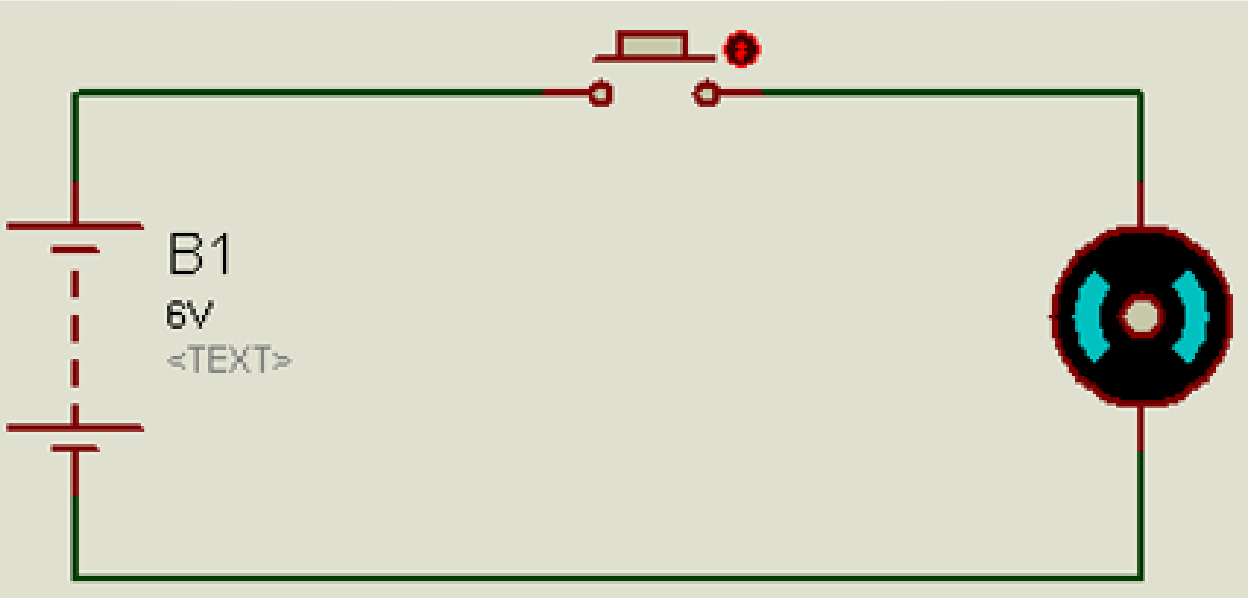
# PWM – Modulação por Largura de Pulso

•O controle PWM permite que as cargas tenham a intensidade de funcionamento variada através do aumento ou diminuição no tempo de ciclo ativo. No exemplo do motor, pode-se observar as seguintes vantagens:

- ✓ Variação na velocidade de giro do motor sem perca do torque, já que a tensão aplicada continua a mesma, porém a ciclos variados. Se ao invés do PWM fosse usado um potenciômetro para limitar a tensão do motor, a velocidade diminuiria, mas com perda de torque.
- ✓ Essa variação na velocidade permite partidas suaves quando os motores tiverem que trabalhar com uma carga elevada.
- ✓ Se o ciclo ativo for de 50%, há uma economia de 50% no consumo de energia e a velocidade de giro do motor vai ser a metade. Exemplo: Motor ligado o tempo todo = 1000rpm e motor 50% ligado e 50% desligado = 500rpm.

# PWM – Modulação de Largura de Pulso

•No circuito abaixo enquanto o botão não estiver pressionado, o motor estará obviamente desligado. Enquanto o botão estiver pressionado, o motor estará girando.

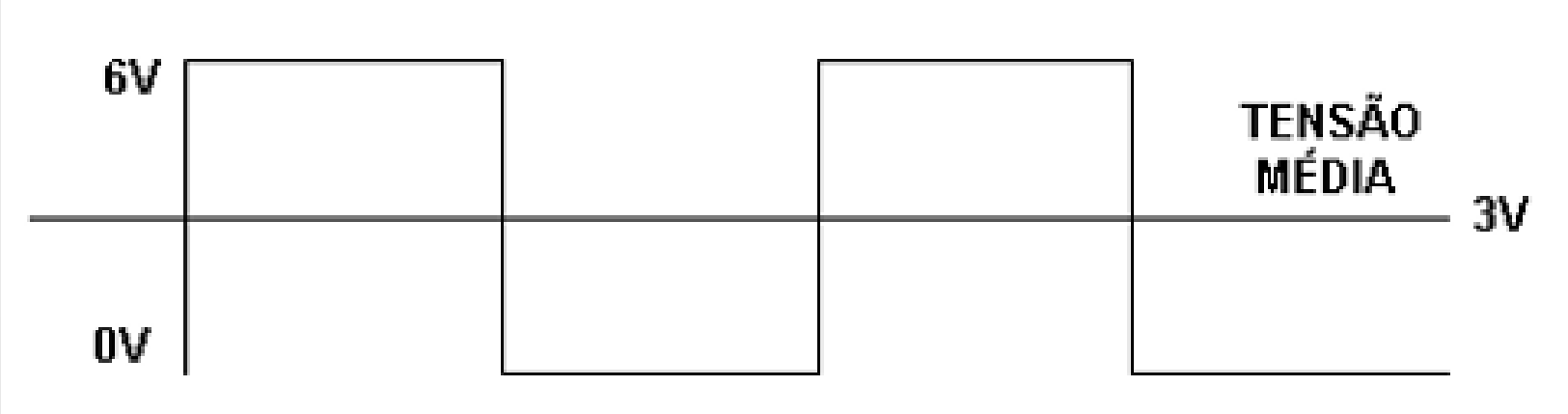


# PWM – Modulação de Largura de Pulso

•Suponha que você consiga pressionar o botão e soltar uma quantidade elevada de vezes por segundo, passando metade do tempo ligado e metade desligado. O sinal aplicado ao motor está descrito no gráfico abaixo:

# PWM – Modulação de Largura de Pulso

•**Embora seja praticamente impossível, se o tempo ligado for exatamente o mesmo do tempo desligado, então teremos uma tensão média de 3V no caso do motor, já que o tempo ligado é o mesmo desligado.**



# PWM – Modulação de Largura de Pulso

•**Para diminuir a velocidade do motor, diminuímos a largura do pulso, fazendo com que o motor passe menos tempo ligado. Da mesma forma, para aumentar a velocidade do motor, aumentamos a largura do pulso.**

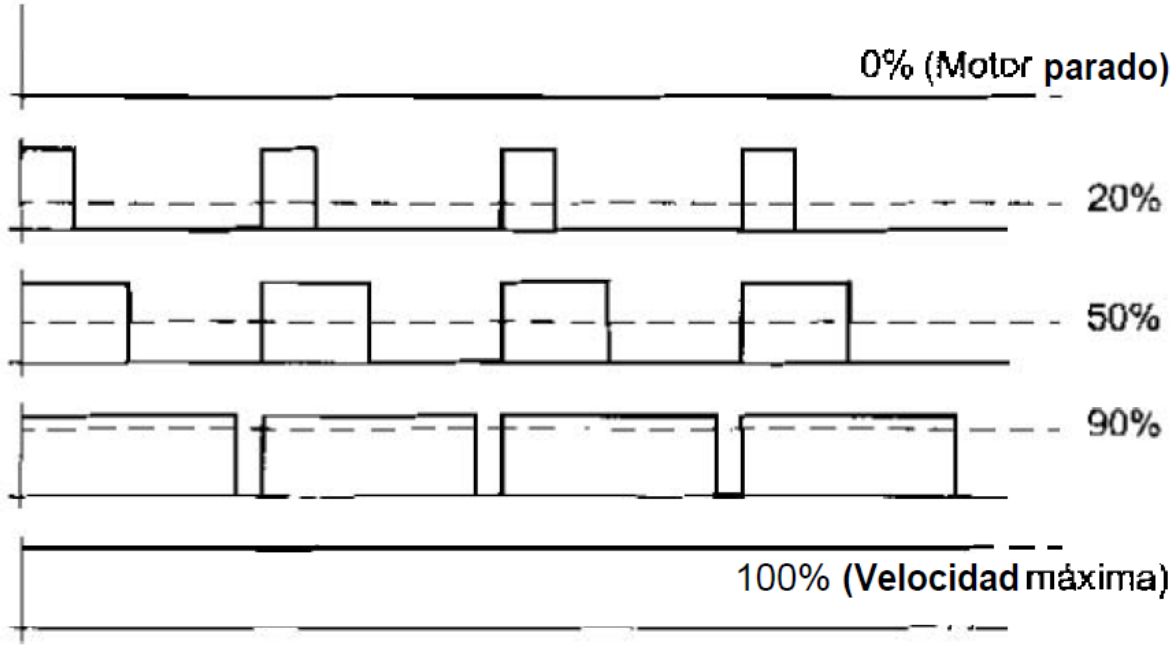



*Figura 29-11 Control* ***velocidad de*** *un motor c.c. mediante* ***PWM***

# PWM – Modulação de Largura de Pulso

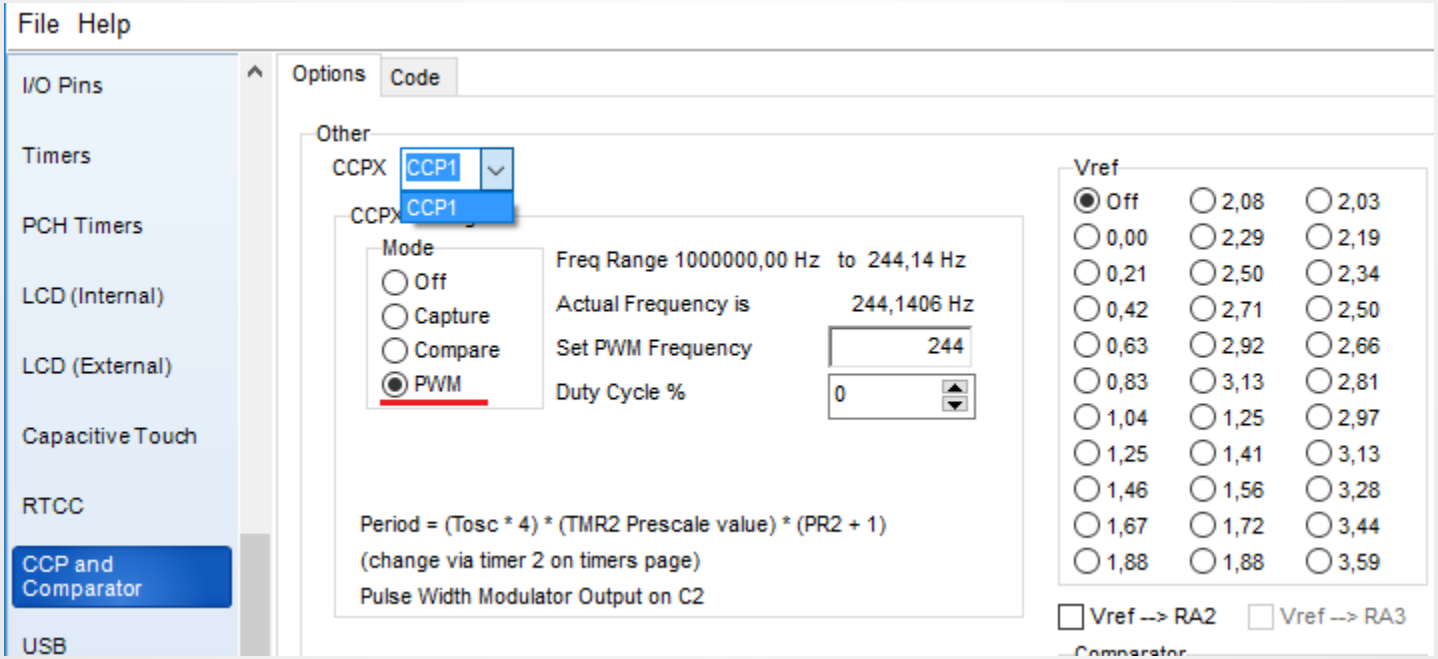
•Podemos generalizar da seguinte forma. Suponha um circuito com um motor que quando ligado 100% do tempo tenha uma rotação de 3100rpm com uma tensão de 12V, potência de 218W e 3,44Nm. Calcule a rotação (rpm), a tensão média (Vm), a potência (W) e o torque (Nm) nos casos a seguir.

- 50% do tempo ligado e 50% do tempo desligado.
- 75% do tempo ligado e 25% do tempo desligado.
- 25% do tempo ligado e 75% do tempo desligado.

- rpm(pwm)=rpm.porcentagem(ligado)
- P=U.I
- I=P/U

# PWM – PIC

•Microcontroladores PIC possuem um módulo denominado CCP (Capture/Compare/PWM) que é útil na modularização de largura de pulso. É possível configurar na aba "CCP and Comparator" do CCS C Compiler selecionando o número do módulo disponível, escolhendo a opção "PWM" e configurando a frequência desejada.



# PWM – PIC

•O código gerado:

```
#include <pwm.h>

void main()
{
  setup_timer_2(T2_DIV_BY_16,255,1);
  setup_ccp1(CCP_PWM);
  set_pwm1_duty((int16)0);

  while(TRUE)
  {
    //TODO: User Code
  }
}
```

# PWM – Modulação de Largura de Pulso

•O timer específico para o PWM (Timer 2) é um timer de 8 bits e é tomado como base para o cálculo de frequência máxima e frequência mínima do PWM. Tendo 8 bits, a carga do Timer 2 (ou período) pode variar de 0 (00000000) até 255 (11111111) e o prescaler pode ser 1, 4 ou 16. Leva-se em consideração também o clock interno do microcontrolador Ex: 4000000Hz (PIC 16F628A) A fórmula para o cálculo da frequência do PWM é:

• f(PWM) = $\frac{Frequência\ do\ cristal}{(Carga\ do\ timer2 + 1) * (Timer2\ Prescaler) * 4}$

# PWM – Frequência máxima e mínima

•Vamos calcular a frequência máxima e mínima do PWM em um microcontrolador PIC 16F628A)

- Frequência do Cristal - 4000000Hz
- Carga do timer2 (Frequência máxima) – 1;
- Carga do timer2 (Frequência mínima) – 255;
- Prescaler do timer2 (Frequência máxima) – 1;
- Prescaler do timer2 (Frequência mínima) – 16;

Então:

- $fmín(PWM) = \frac{4000000}{(255 + 1) * (16) * 4} = \frac{4000000}{16384} = 244Hz$

- $fmáx(PWM) = \frac{4000000}{(1 + 1) * (1) * 4} = \frac{4000000}{8} = 500000Hz$

# PWM – Duty Cicle

• **Exemplo para 16 bits:**

```
set_pwm1_duty(128L);
```

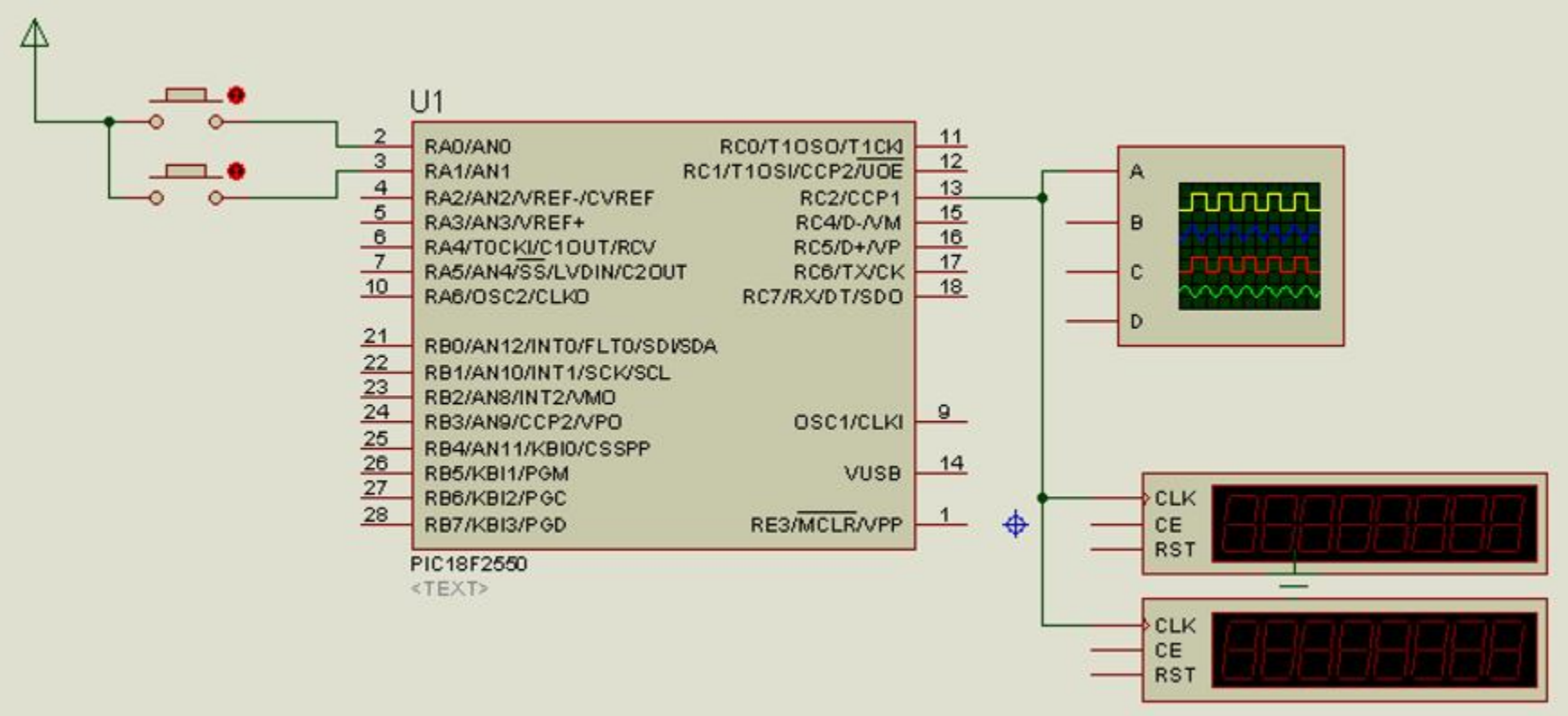
• **Duty Cicle =** $\frac{16\text{-}bit\ Duty\ cycle}{(Carga\ do\ timer2 + 1) * 4} = \frac{128}{(255 + 1) * 4} = 12{,}5\%$

•**É possível também, além do L após o valor, declarar uma variável com nome "valor" com o tipo int8 ou int16 para diferenciar.**

```
int16 valor=128;
...
set_pwm1_duty(valor);
```

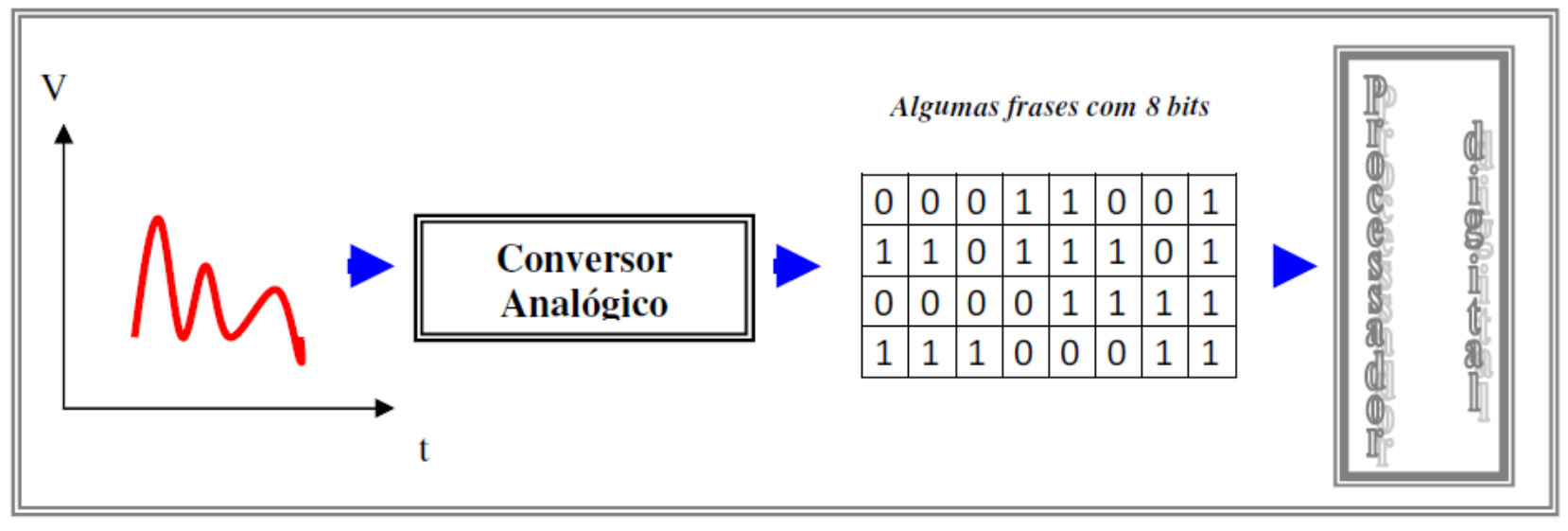
# PWM – Esquemático - PIC

- **Monte o circuito no Proteus usando o PIC 18F2550 e um osciloscópio de "Virtual Instrument Modes", um "Counter Timer" e dois buttons**



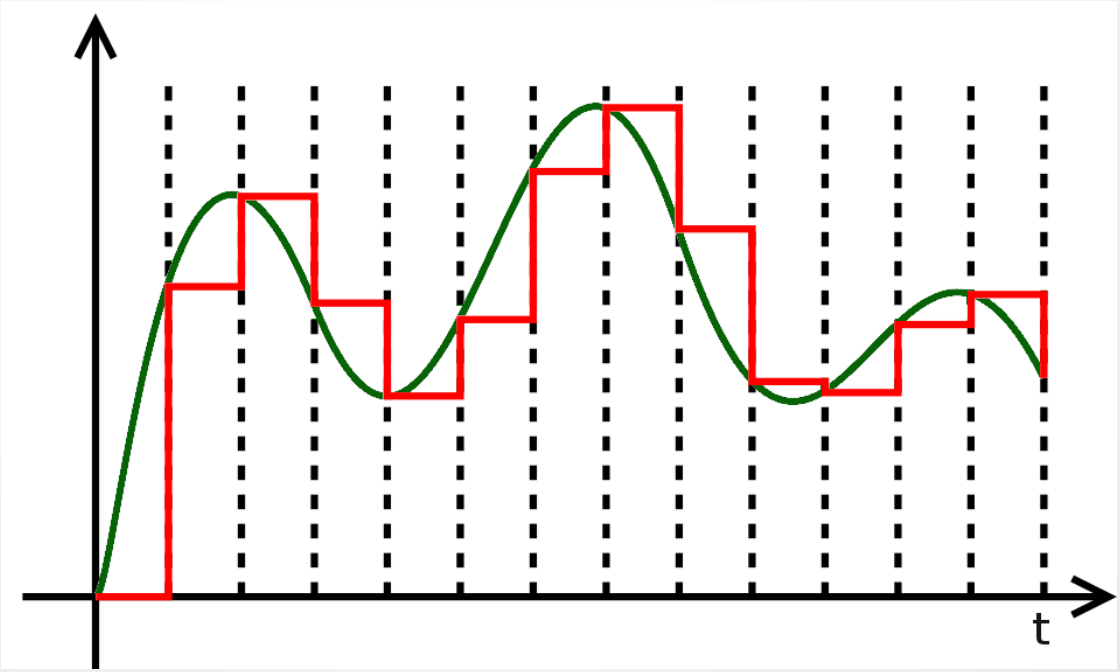
# Conversor Analógico/Digital

•Conversores analógico digitais (conversores A/D ou ADC) convertem um sinal analógico em um equivalente digital, comumente com resoluções de 8 bits variando valores entre 0 (00000000) e 255 (11111111), 10 bits variando valores entre 0 (0000000000) e 1023 (1111111111), etc



# Conversor Analógico/Digital

•Essa conversão é feita por base na leitura de tensão e de corrente numa porta do microcontrolador e dependendo da resolução utilizada, cada nível de tensão tem um valor binário equivalente. Por exemplo: Se for feita a leitura de um sinal analógico que varia de 0 a 5V a uma resolução de 8 bits (variação de valores de 0 a 255) então teremos que a cada (5/255)V teremos um incremento no valor digital.



# Conversor Analógico/Digital

•Sendo assim, 5/255 é igual a aproximadamente 0.0196. Então a cada 0.0196V teremos um incremento no valor digital. Esse processo é chamado de discretização, ou digitalização.

| Passo da resolução | Tensão lida | Equivalente binário |
|---|---|---|
| 0 | 0 * 0.0196 = 0V | 00000000 |
| 1 | 1 * 0.0196 = 0.0196V | 00000001 |
| 2 | 2 * 0.0196 = 0.0392V | 00000010 |
| ... | ... | ... |
| 150 | 150 * 0.0196 = 2,94V | 10010110 |
| ... | ... | ... |
| 255 | 255 * 0.0196 ≈ 5V | 11111111 |

# Conversor Analógico/Digital

•Após ativar (setar) o canal ADC, usando o comando set_adc_channel(canal); (sendo o valor do canal de 0 a 7, em um PIC com 8 canais), estes valores resultantes da variação pode ser tratados como variáveis através do comando de leitura v=read_adc(); (a variável v recebe o valor da conversão AD. • Se o valor da variável for usado para setar o ciclo ativo (duty cicle) em um controle PWM, há que se fazer a relação entre a resolução do canal ADC e a carga (período) do Timer2.

• Exemplo: Se a resolução do ADC for de 8 bits (0 a 255) e a carga do timer2 variar de 0 a 149, então temos:

| Valor ADC | Valor PWM | Tensão equivalente |
|---|---|---|
| 0 | 0 | 0V |
| 255 | 149 | 5V |

# Conversor Analógico/Digital

- Algumas características relativas a Conversão Analógica Digital devem ser levadas em consideração, são elas:
- Canais se conversão são portas (pinos) de entrada para conversão AD. ▫ Os módulos fazem a conversão do valor lido do canal para um equivalente binário.
- Alguns microcontroladores possuem 8 canais para conversão (16F877A), outros 10 canais (18F2550).
- Embora alguns microcontroladores tenham vários canais, todos eles possuem apenas um módulo de conversão.
- Não se pode utilizar mais de um canal simultaneamente num mesmo microcontrolador, dessa forma, é necessário desabilitar um para poder habilitar outro.

# Conversor Analógico/Digital

1. Diferencie carga do Timer2, prescaler do Timer2 e Duty Cicle.
2. Sendo a carga do Timer2 igual a 200, qual será a tensão média se o PWM for setado em 75?
3. Em um canal ADC com resolução de 8 bits o valor lido e armazenado em uma variável é 85, qual a tensão está sendo aplicada na entrada?
4. Qual a frequência do PWM em um PIC 16F877A com a carga do timer definido em 255 e prescaler definido em 16?

5. Implementar o sistema da máquina copiadora, onde um sinal luminoso acende quando falta papel na bandeja (sensor da bandeja for 0) ou o papel atola em dois sensores (passagem de papel forem iguais a 1). Escolha Arduino ou PIC.
6. Com um PIC implemente um sistema de irrigação que libera água somente no primeiro minuto do dia e depois desligue, acionando novamente no dia seguinte.
7. Implemente um PWM com dois botões onde um aumenta a largura do pulso e outro diminui usando o PIC.
8. Montar no laboratório pelo menos um circuito com PIC e outro com Arduino.

OBS: Mandar o código e o esquemático de cada questão para o email murilo.leal@centec.org.br