Internet das Coisas

Aula 13 – Leitura de sensores de resposta binária com uso de interrupções

Prof. Rodrigo Barreto rodrigo.barreto@qi.edu.br / Fone/Whats: (51) 98412-5651

Utilizando interrupções para contar pulsos

Todos estes sensores, como visto, ficam realizando a modificação do seu estado de acordo com a situação que estão observando. Porém, alguns deles, fazem essa mudança de forma constante e rítmica de acordo com o movimento de algum componentes interno ou externo. Isso faz com que ele, de tempos em tempos, emita pulsos e a quantidade de pulsos em um determinado período de tempo pode nos informar uma grandeza medida.

Utilizando interrupções para contar pulsos

Este é o caso do encoder óptico, encoder magnético e da válvula de vazão de água. Os dois primeiros são ligados geralmente a um motor, polia ou correia e seu elemento interno emite um pulso a cada volta ou a cada movimento de retorno após um movimento de ida. Já a válvula de vazão, possui uma hélice com um elemento magnético, emitindo também um pulso a cada volta.

encoder óptico

encoder magnético

sensor de vazão

Utilizando interrupções para contar pulsos

A dificuldade em trabalhar com sensores do tipo é que não basta apenas realizar um digitalRead a cada janela de tempo e contar um pulso se o estado for HIGH, pois não sabemos quanto tempo esperar (sendo inviável a configuração de um delay). Essa inviabilidade se deve a, dependendo da velocidade de leitura, o tempo de cada pulso mudar. Além disso, cada sensor é construído para que este pulso dure um tempo diferente.

Utilizando interrupções para contar pulsos

Neste caso, podemos fazer uso das interrupções do Arduino. Como vimos na aula sobre o pinout da placa, o Arduino Uno possui duas interrupções: a interrupção 0 (localizada na porta digital 2) e a interrupção 1 (localizada na porta digital 3). Nós podemos vincular um método a ser executado sempre que a interrupção mudar o seu estado e ligar o pino de dados do nosso sensor à esta porta, para que a cada pulso (que é uma mudança do estado LOW para o estado HIGH) ative a interrupção (e, consequente, execute o método vinculado à ela).

Utilizando interrupções para contar pulsos

Para isso, nós utilizamos o método attachInterrupt, como mostrado abaixo:

```
attachInterrupt( <interrupcao>, <metodo>, <modo> );
```

<interrupcao>
número da interrupção que queremos monitorar (0 - pino 2 ou 1 - pino 3)

<metodo>
método que será chamado a cada nova chamada da interrupção

<modo>
modo de disparo da interrupção. Pode receber os seguintes valores:

- LOW: dispara quando o estado da porta for para LOW
- CHANGE: dispara quando o estado da porta mudar (LOW para HIGH ou HIGH para LOW)
- RISING: dispara quando o estado do porta mudar de LOW para HIGH apenas
- FALLING: dispara quando o estado da porta mudar de HIGH para LOW apenas

Utilizando interrupções para contar pulsos

Após vincularmos um método à uma interrupção através do attachInterrupt, podemos ligar ou desligar essa interrupção conforme desejarmos. Para isso, utilizamos os métodos abaixo:

sei() → ativa a interrupção

cli() → desativa a interrupção

Utilizando interrupções para contar pulsos

Exemplo de um contador de pulsos por segundo genérico:

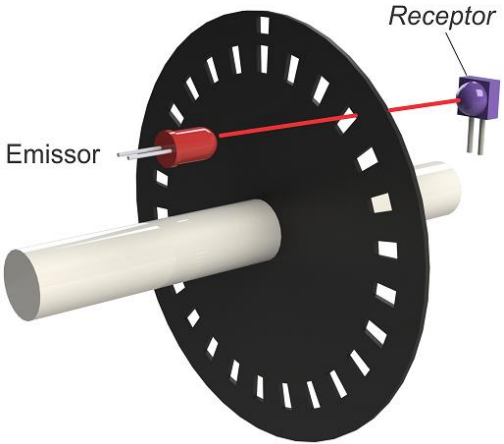
```
int pulsos;

void contaPulso(){
  pulsos++;
}

void setup(){
  attachInterrupt(0, contaPulso, RISING);
  Serial.begin(9600);
}

void loop(){
  pulsos = 0;
  sei();
  delay(1000);
  cli();

  Serial.println("Pulsos por segundo: " + (String)pulsos);
}
```

Exemplo 1 com interrupções

Exemplo prático 1: encoder óptico e magnético

Os encoderes óptico e magnético funcionam da mesma forma, apenas com elementos de operação diferentes que geram pulsos (o óptico um disco, o magnético um imã). Para ligarmos um encoder a um motor e descobrirmos a sua velocidade, necessitamos do uso de interrupções que nos enviem pulsos. No óptico, esse pulso acontece cada vez que a luz é interrompida pelo disco que fica posicionado entre seu emissor de IR e seu receptor de IR. No magnético, esse pulso acontece a cada volta quando um imã passa pelo seu elemento detector de campo magnético.

Exemplo 1 com interrupções



Funcionamento do encoder óptico

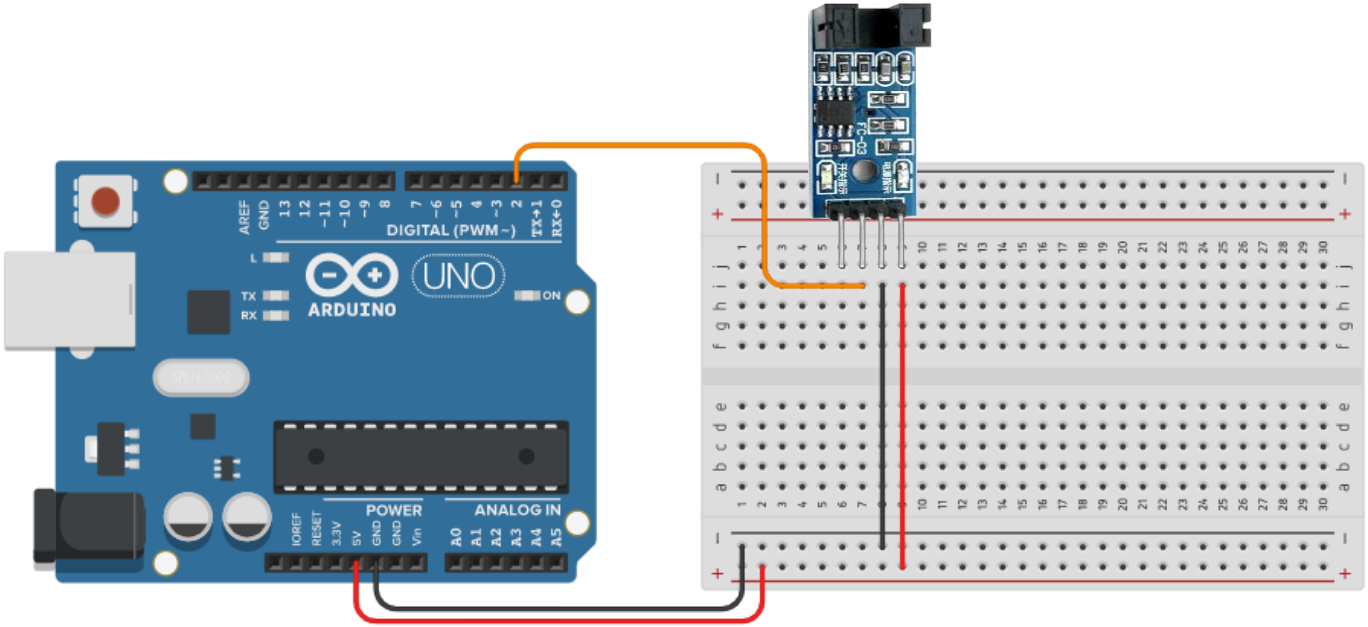
Funcionamento do encoder magnético

Exemplo 1 com interrupções

Para o nosso exemplo, vamos controlar a velocidade de um motor CC utilizando um encoder do tipo óptico, com o seu pino da saída de dados ligada na interrupção 0 (porta 2). Além desse pino, os encoderes possuem um pino VCC, um GND e alguns deles um pino especial analógico para a geração de ondas que permitam a visualização da aceleração e desaceleração em osciloscópios e simuladores. Essa pinagem muda de fabricante para fabricante. No nosso caso, utilizaremos um encoder óptico de 4 pinos: o primeiro é o analógico para produção de sinal e não será utilizado, o segundo é o digital de dados que ligaremos à porta 2 (interrupção 0), o terceiro é o GND e o quarto é o VCC.

Exemplo 1 com interrupções

Assim, teremos o seguinte esquema eletrônico:



Exemplo 1 com interrupções

Nosso código será então baseado na detecção dos pulsos vindos do encoder, sempre verificando ao final de 1 segundo o total de pulsos. No nosso caso, estamos utilizando um encoder com 20 furos (ou seja, cada 20 pulsos contam como uma volta) e com um raio de 1,5 cm. Assim, podemos calcular a velocidade em m/s através da fórmula abaixo (e depois converter para Km/h multiplicando por 3,6):

velocidade = voltas . 2 . π . r

Onde: v = velocidade
voltas = total de voltas por segundo (pulsos / 60)
r = raio em metros

Exemplo 1 com interrupções

Para contar os pulsos, será necessária a criação de um método que incrementa um contador de pulsos, para que ele possa ser vinculado à interrupção 0 (porta 2) e chamado a cada nova interrupção (gerada a cada novo pulso lido). Deste modo, nosso código ficará assim:

Exemplo 1 com interrupções

```
int pulsos;

void contaPulso(){
  pulsos++;
}

void setup(){
  attachInterrupt(0, contaPulso, RISING);
  Serial.begin(9600);
}

void loop(){
  pulsos = 0;
  sei();
  delay(1000);
  cli();

  float voltasSegundo = pulsos / 20;
  float metrosSegundo = voltasSegundo * 2 * 3.1416 * 0.015;
  float quilometrosHora = metrosSegundo * 3.6;

  Serial.println("Pulsos por segundo: " + (String)pulsos);
  Serial.println("Voltas por segundo: " + (String)voltasSegundo);
  Serial.println("Metros por segundo: " + (String)metrosSegundo);
  Serial.println("Quilômetros por hora: " + (String)quilometrosHora);
}
```

Exemplo 1 com interrupções

Entendendo o código: começamos criando uma variável inteira chamada de pulsos, que irá contar o número de pulsos lidos. Essa variável será incrementada pelo método contaPulso que a cada vez que for chamado, aumenta o seu valor em 1. Para que este método seja chamado a cada novo pulso do nosso sensor, utilizamos dentro do setup o método attachInterrupt, para vinculá-lo à interrupção 0 (da porta digital 2), usando o modo RISING pois enquanto o disco gira seu estado muda de LOW para HIGH, informando um novo pulso de luz (e o RISING é para justamente acionar a interrupção quando o estado muda de LOW para HIGH).

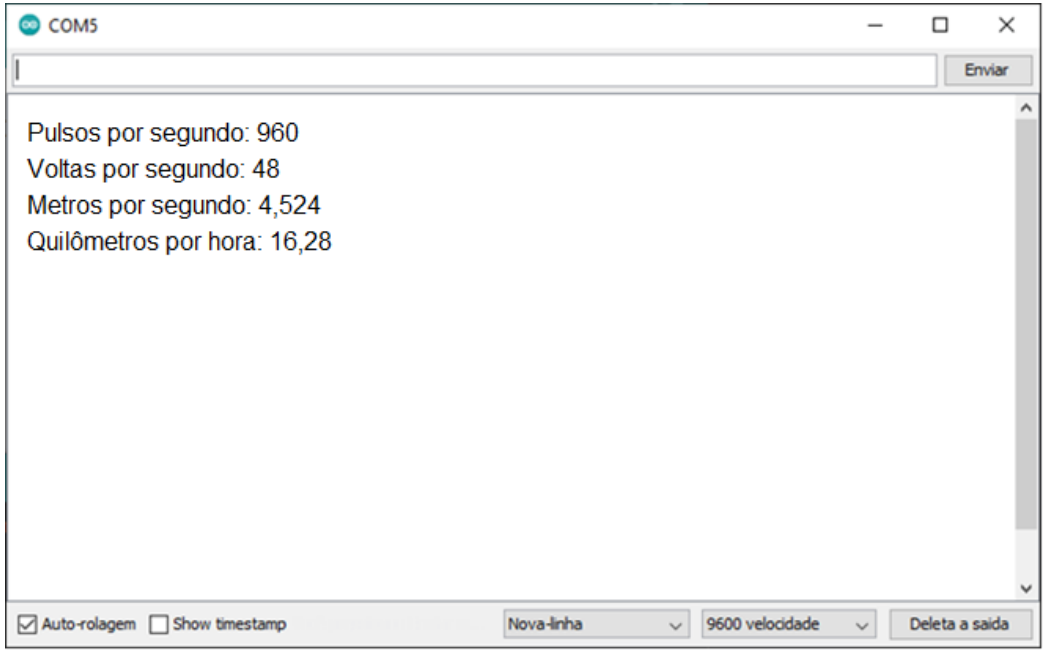
Exemplo 1 com interrupções

Entendendo o código: Além disso, também dentro do setup, inicializamos a nossa Serial para que possamos utilizá-la para a escrita dos resultados (e a posterior visualização no Monitor Serial). Dentro do loop, zeramos a variável pulsos e através do método sei() habilitamos o uso das interrupções e, nesse momento, cada novo pulso passa a ser contabilizado. Para que essa contagem ocorra por exatamente 1 segundo, utilizamos um delay logo após a habilitação do uso das interrupções e depois de uma espera de 1 segundo, desabilitamos o uso das interrupções com o uso do método cli(). A partir daqui os pulsos não são mais contabilizados.

Exemplo 1 com interrupções

Entendendo o código: Agora, podemos utilizar o total de pulsos contabilizados em um segundo, dividindo por 60 (transformando em voltas), com essas voltas calcular a velocidade em metros por segundo e finalmente em quilômetros por hora, escrevendo na serial. Se tudo der certo, deveremos ter uma saída similar à essa ao abrir o Monitor Serial da Arduino IDE:

Exemplo 1 com interrupções



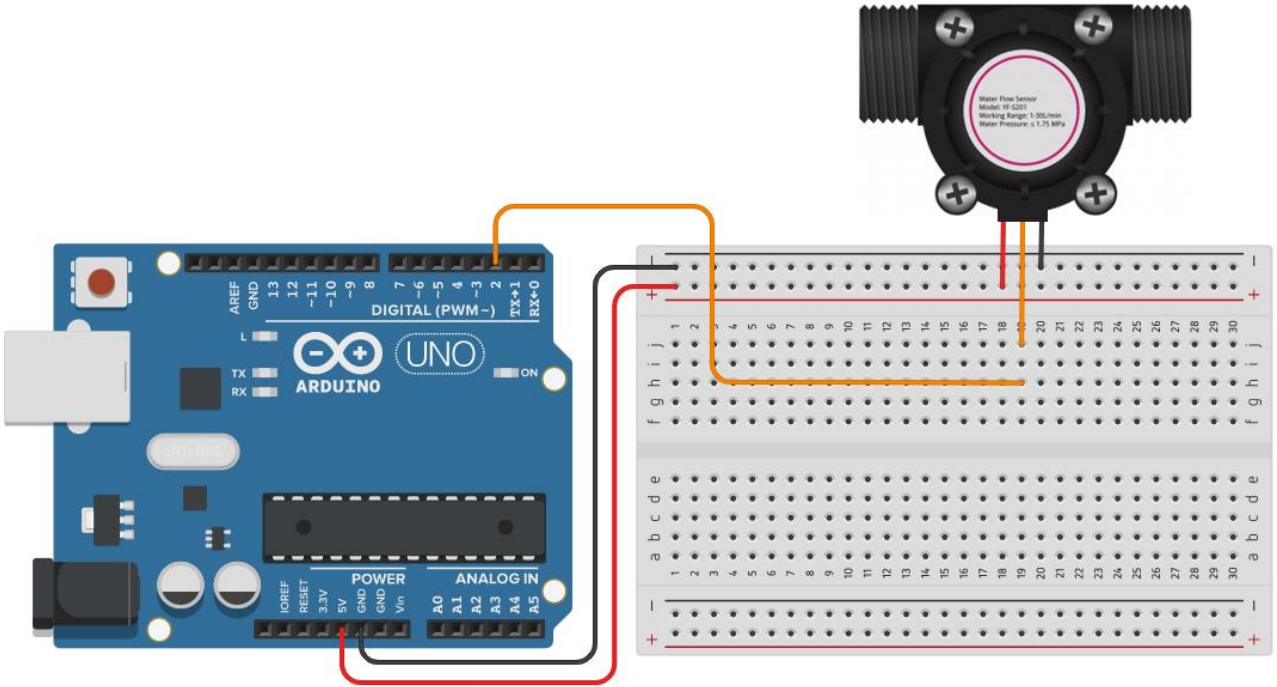
Exemplo 2 com interrupções

Exemplo prático 2: sensor de vazão por efeito hall

O sensor de vazão por efeito hall, funciona de forma similar aos encoderes, apenas com um elemento mecânico diferente. A cada volta de sua hélice interna, ele emite um pulso. Como seu diâmetro é conhecido, ele já nos entrega uma constante (com valor 5.5) para que possamos dividir o total de voltas em um segundo e converter isso em uma estimativa de vazão em litros por segundo. Seguimos aqui o mesmo princípio do exemplo anterior e o seu pino de dados será ligado na interrupção 0 (porta 2) neste exemplo.

Exemplo 2 com interrupções

Assim, teremos o seguinte esquema eletrônico:



Exemplo 2 com interrupções

Nosso código também será baseado na contagem de pulsos através do disparo da interrupção e por causa disso, também necessitamos da criação de um método que incremente o total de pulsos, para ser vinculado à ela. Ao final de um segundo dividindo os pulsos por 5.5, obtendo assim uma estimativa de litros por segundo e depois multiplicando por 60, transformando em litros por minuto. Deste modo, nosso código ficará assim:

Exemplo 2 com interrupções

```
int pulsos;

void contaPulso(){
  pulsos++;
}

void setup(){
  attachInterrupt(0, contaPulso, RISING);
  Serial.begin(9600);
}

void loop(){
  pulsos = 0;
  sei();
  delay(1000);
  cli();

  float litrosSegundo = pulsos / 5.5;
  float litrosMinuto = litrosSegundo * 60;
  float litrosHora = litrosMinuto * 60;

  Serial.println("Litros por segundo: " + (String)litrosSegundo);
  Serial.println("Litros por minuto: " + (String)litrosMinuto);
  Serial.println("Litros por hora: " + (String)litrosHora);
}
```

Exemplo 2 com interrupções

Entendendo o código: o código deste exemplo é análogo ao código do exemplo de uso do encoder óptico. Nossa única diferença são os cálculos realizados a partir do número de pulsos obtidos. Se tudo der certo, deveremos ter uma saída similar à essa ao abrir o Monitor Serial da Arduino IDE:

Exemplo 2 com interrupções