

# Encoder

- [Conexão elétrica](#)
- [Categoria motor](#)
- [Categoria de variáveis](#)
- [Exercício](#)

# Conexão elétrica

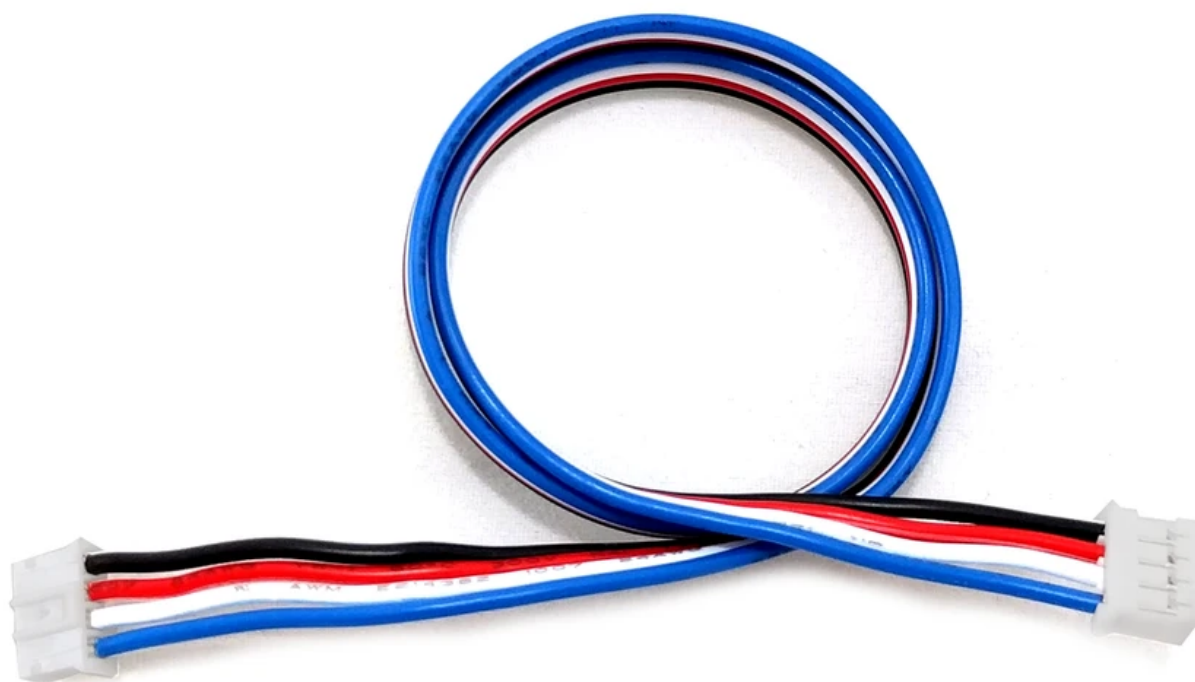
---

Encoder é um sensor que fornece a posição angular do motor ao qual está conectado - veja mais em: [Encoders](#)

A presente seção tem como objetivo explicar a utilização desse sensor com aplicação nos motores - será usado um HD Hex com ultraplanetária 60:1.

## Cabos

Visto que o encoder é um sensor digital - veja a seção [Conexão elétrica para sensores digitais](#) - o cabo que ele usa é o padrão JST-PH de 4 pinos.



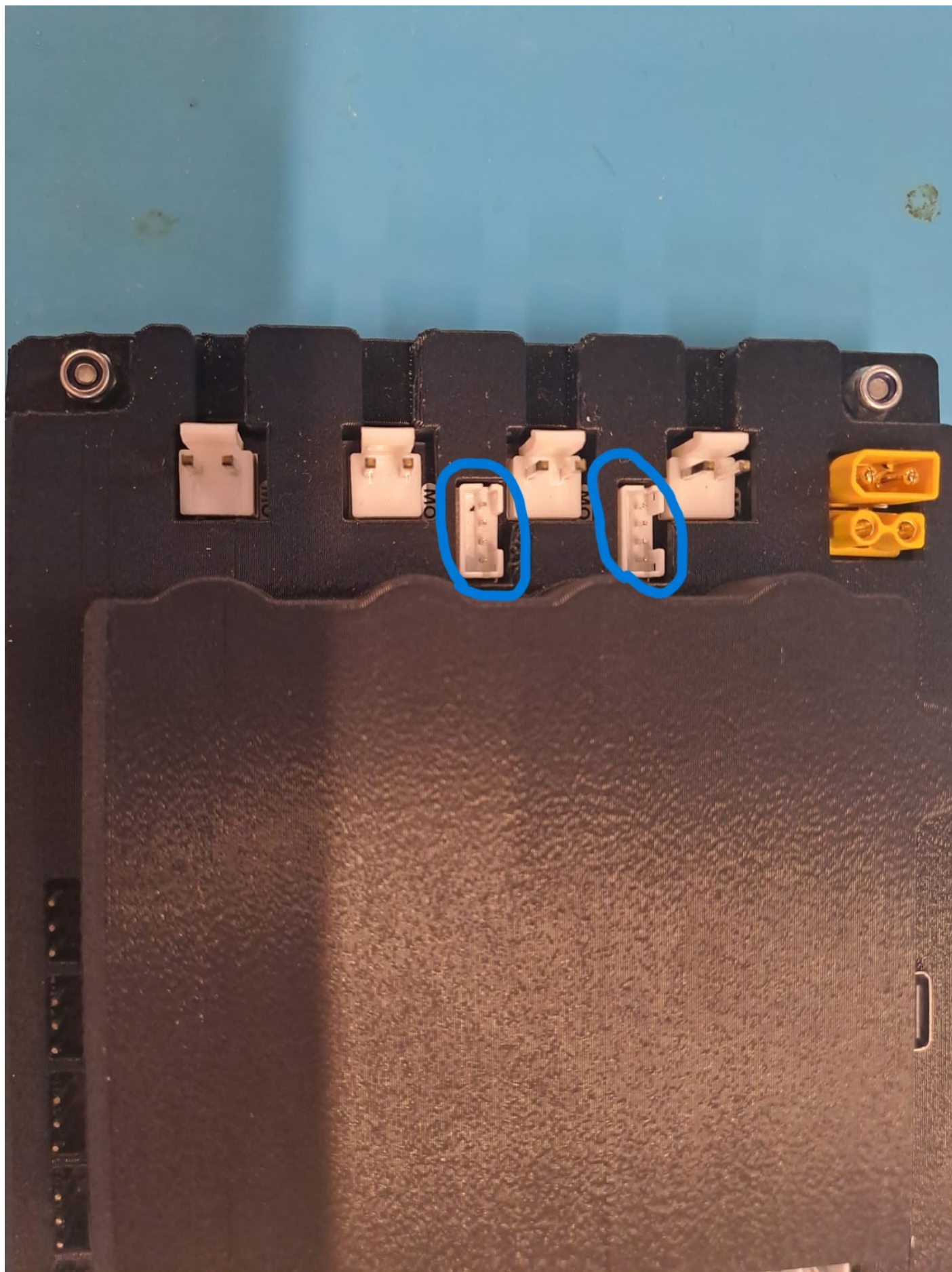
## Conexão no Motor

Uma das pontas do cabo é conectada diretamente ao motor, na entrada correspondente - como indicado:



## Conexão placa

A ponta restante é conectada a placa nas entradas mostrados abaixo:



# Conexão geral

Conectado os dois deve ficar da seguinte forma:





É interessante pensar que o motor e o encoder são dispositivos separados, tanto é que, o motor funciona com 12V, enquanto o encoder pode ser energizado apenas pelos 5V da conexão USB-C, portanto, nos testes seguintes não será utilizado bateria



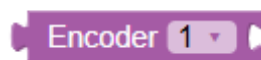
# Categoria motor

---

O encoder trabalha juntamente com a categoria de motor.

## Objeto do encoder

O objeto de encoder é semelhante aos outros objetos descritos neste documento - uma diferença é que como ele não é um acionador, sua função é retornar um valor.



## Bloco de posição

O seguinte bloco retorna a posição angular do motor especificado pela porta do encoder.



Nesse exemplo, o encoder retorna a posição angular do motor 1

## Bloco de *reset*

O seguinte bloco tem como objetivo zerar a contagem atual do encoder.



Existem outros blocos de encoder que não serão abordados neste documento

A próxima página tem como objetivo ensinar a utilização da categoria de variáveis.

# Categoria de variáveis

---

Na grande maioria dos códigos é importante salvar os valores que os blocos retornam, isso pode ser feito utilizando a categoria de variáveis - que será explicado abaixo.

## Blocos de tipos

Os chamados tipos indicam quais espécies de valores podem ser armazenados nessas variáveis, como: ponto flutuante, inteiro ou texto. Veja mais em: [Data types](#)

double

int

string

## Bloco de nome

Para criar uma variável é preciso escolher um nome para ela juntamente do tipo de valores que devem ser armazenados. Segue abaixo um exemplo de criação de variável:

double **name**

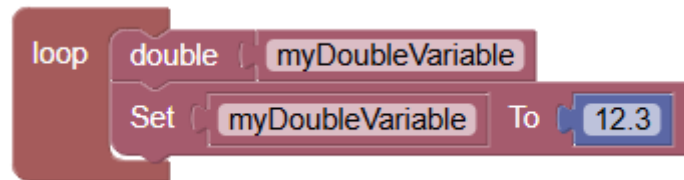
Esse é o nome padrão da variável, o qual deve ser mudado, para fazer isso apenas clique com o botão esquerdo sobre o texto e digite o novo nome.

Como:

double myDoubleVariable

## Definir um valor

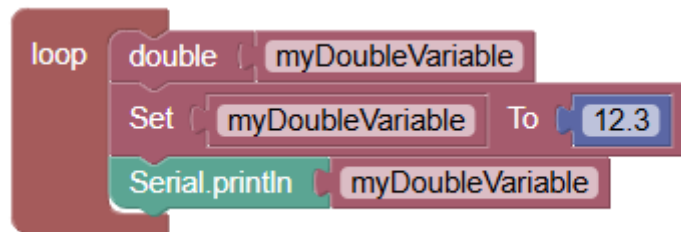
Para definir um valor se usa o bloco **set**, exemplo abaixo:



Para definir um valor é preciso declarar a variável antes, se não é gerado um erro de compilação.

## Retorno de valor

Para retornar o valor de um bloco é necessário usar somente o bloco de nome, como demonstra:



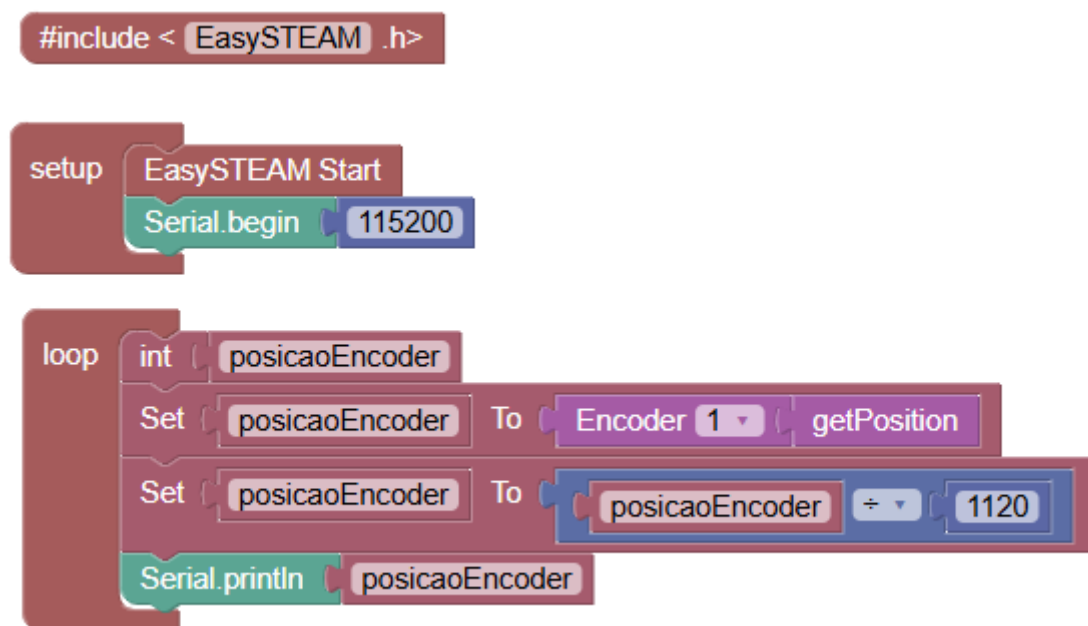
O valor mostrado no monitor deve ser 12.3

# Exercício

A presente seção tem como objetivo fazer um código que realize o seguinte:

- Retorna o valor do encoder para uma variável;
- Divide a variável por um valor arbitrário - você escolhe;
- Mostra o valor no monitor serial.

O seguinte exercício pode ser feito da forma mostrada abaixo:



1120 é considerado uma volta completa de um motor HD Hex 40:1, portanto, se esse motor fosse utilizado o código acima mediria o número de voltas.

Verifique no monitor serial a saída do código executado acima.