

Minicurso: Programação Arduino Aplicada à Robótica Móvel

Alexandre C. de Oliveira

UFPR - Grupo VRI - Visão Robótica e Imagem

Junho 2020



Table of Contents

1 Apresentação do Arduino

- Hardware - Arduino Uno (ATmega328)
 - Ambiente de Desenvolvimento (IDE)

2 Comunicação, Sensores e Atuadores

- Comunicando com hardware externo
 - Exemplos de sensores e atuadores

3 Programando um Robô Móvel

- Programando sensores e atuadores
 - Programando o Robô Capivara VSSS 2017



Table of Contents

1 Apresentação do Arduino

- Hardware - Arduino Uno (ATmega328)
- Ambiente de Desenvolvimento (IDE)

2 Comunicação, Sensores e Atuadores

- Comunicando com hardware externo
- Exemplos de sensores e atuadores

3 Programando um Robô Móvel

- Programando sensores e atuadores
- Programando o Robô Capivara VSSS 2017



O que é Arduino

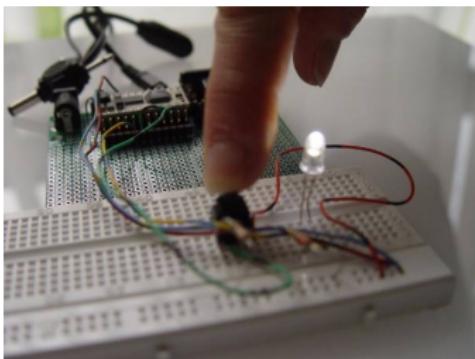
Arduino é uma plataforma de prototipagem eletrônica de hardware livre e código aberto em placa única. Foi idealizado para que pessoas, mesmo com conhecimento básico em hardware e programação possam implementar projetos eletrônicos que interajam com pessoas e com o ambiente. Teve origem no *Wiring* (outra plataforma de prototipagem) e possui linguagem própria, baseada na estrutura C/C++ como também um ambiente de desenvolvimento (IDE).

Site oficial: <https://www.arduino.cc>

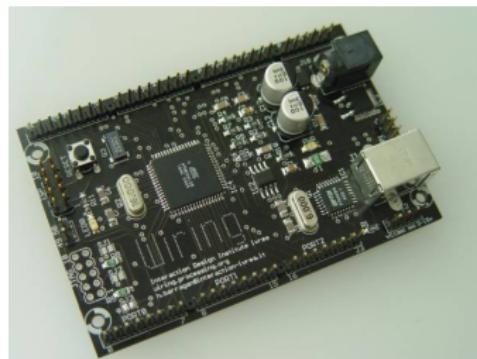


Wiring

Wiring: Trabalho de mestrado de Hernando Barragán do *Interaction Design Institute Ivrea (IDII)* na Itália, (2003).



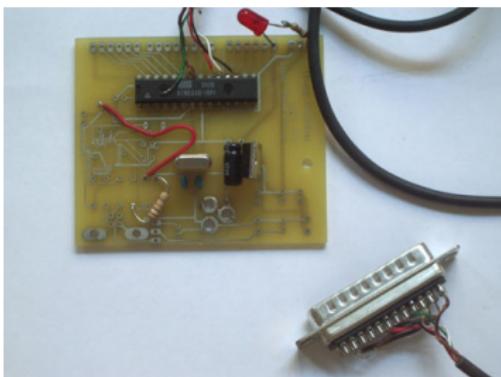
Primeiro protótipo de *hardware* do Wiring.



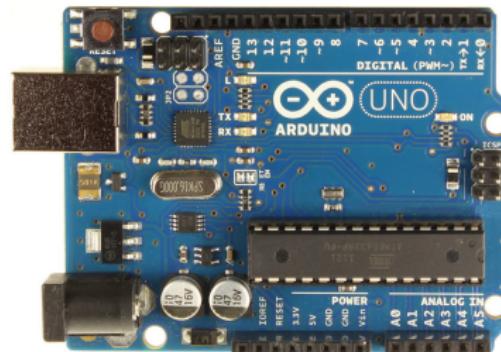
Placa do Wiring finalizada,
modelo atual.

Arduino

Arduino: Baseado no *Wiring*, com suporte ao microcontrolador ATmega8, por Massimo Banzi e David Mellis, (2005).

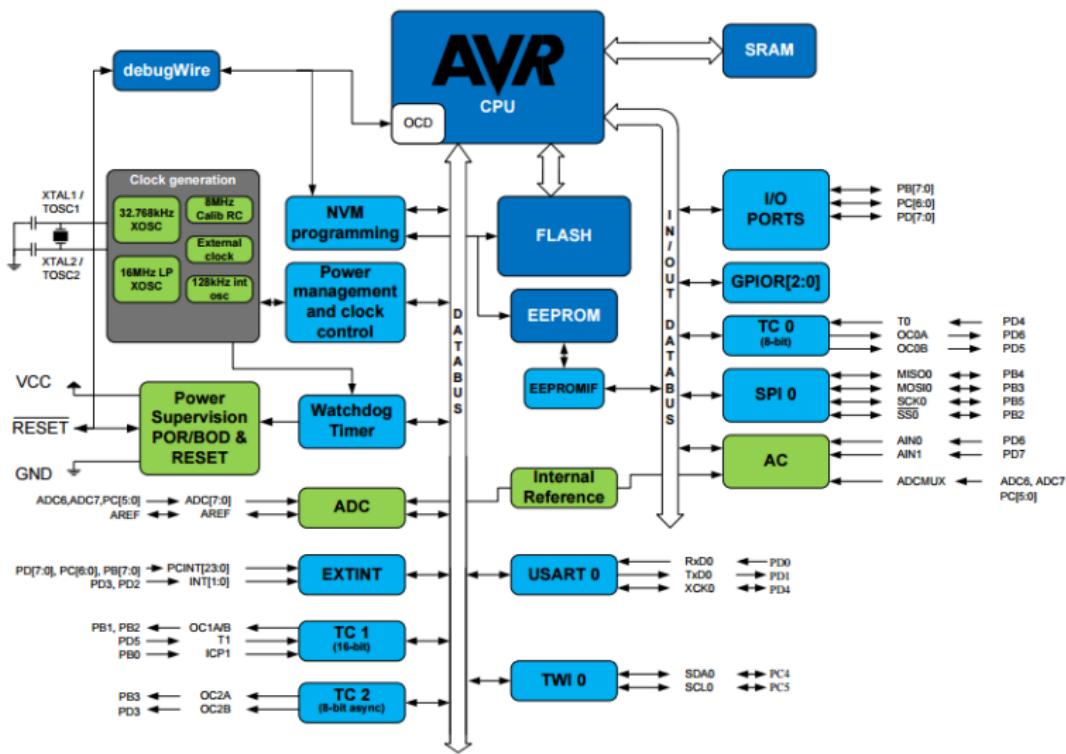


Primeiro protótipo de hardware do Arduino, com o ATmega8.



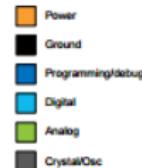
Arduino Uno modelo atual, com o ATmega328.

Diagrama interno do microcontrolador ATmega328



Microcontrolador ATmega328

(PCINT14/RESET) PC6	1	28	PC5 (ADC5/SCL/PCINT13)
(PCINT16/RXD) PD0	2	27	PC4 (ADC4/SDA/PCINT12)
(PCINT17/TXD) PD1	3	26	PC3 (ADC3/PCINT11)
(PCINT18/INT0) PD2	4	25	PC2 (ADC2/PCINT10)
(PCINT19/OC2B/INT1) PD3	5	24	PC1 (ADC1/PCINT9)
(PCINT20/XCK/T0) PD4	6	23	PC0 (ADC0/PCINT8)
VCC	7	22	GND
GND	8	21	AREF
(PCINT6/XTAL1/TOSC1) PB6	9	20	AVCC
(PCINT7/XTAL2/TOSC2) PB7	10	19	PB5 (SCK/PCINT5)
(PCINT21/OC0B/T1) PD5	11	18	PB4 (MISO/PCINT4)
(PCINT22/OC0A/AIN0) PD6	12	17	PB3 (MOSI/OC2A/PCINT3)
(PCINT23/AIN1) PD7	13	16	PB2 (SS/OC1B/PCINT2)
(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)



Pinagem e respectivas funcionalidades do ATmega328.



Principais recursos de hardware do ATmega328

Recurso	ATmega328
Número de pinos	28
Pinos de I/O de propósito geral	23
Memória Flash	32K Bytes
Memória RAM	2K Bytes
Memória EEPROM	1K Bytes
Comunicação SPI	1
Two Wire Interface (I^2C)	1
Porta USART	1
Conversores Analógico-Digital (ADC)	6 (10bits - 15K SPS)
Saídas com PWM	6 (2x 16bits e 4x 8bits)
Contadores/Timers de 8bits	2
Contadores/Timers de 16bits	1



Acesso aos recursos de hardware do Arduino

Os recursos de hardware do microcontrolador, disponíveis na placa do Arduino para a comunicação e controle de hardware externo são multiplexadas, ou seja, cada pino pode desempenhar mais de uma função. Neste processo, é preciso configurar antecipadamente registradores internos ao microcontrolador.

O Ambiente de Desenvolvimento (IDE) do Arduino assim como a sua linguagem de programação, foram idealizado para facilitar muito esse processo, oferecendo abstração do hardware.



IDE do Arduino - Ambiente de Desenvolvimento

STORE SOFTWARE EDU PRO RESOURCES COMMUNITY HELP 

Download the Arduino IDE



ARDUINO 1.8.12

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board.
Refer to the [Getting Started](#) page for installation instructions.

Windows Installer, for Windows 7 and up

Windows ZIP file for non admin install

Windows app Requires Win 8.1 or 10

Mac OS X 10.10 or newer

Linux 32 bits

Linux 64 bits

Linux ARM 32 bits

Linux ARM 64 bits

[Release Notes](#)

[Source Code](#)

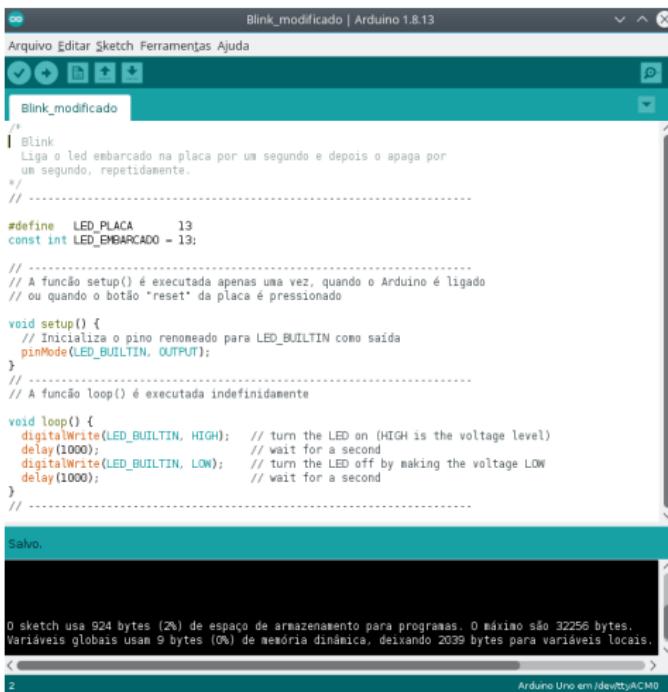
[Checksums \(sha512\)](#)

<https://www.arduino.cc/en/Main/Software>

11/36

IDE do Arduino - Ambiente de Desenvolvimento

Multiplataformas: Windows, Linux, Mac e versão web.



The screenshot shows the Arduino IDE interface with the title bar "Blink_modificado | Arduino 1.8.13". The menu bar includes "Arquivo", "Editar", "Sketch", "Ferramentas", and "Ajuda". Below the menu is a toolbar with icons for file operations. The main window displays the code for the "Blink" sketch:

```
/*
 * Blink
 * Liga o led embarcado na placa por um segundo e depois o apaga por
 * um segundo, repetidamente.
 */
// ----

#define LED_PLACA    13
const int LED_EMBARCADO = 13;

// -----
// A função setup() é executada apenas uma vez, quando o Arduino é ligado
// ou quando o botão "reset" da placa é pressionado

void setup() {
  // Inicializa o pino renomeado para LED_BUILTIN como saída
  pinMode(LED_BUILTIN, OUTPUT);
}

// -----
// A função loop() é executada indefinidamente

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
// ----

Salvo.
```

The status bar at the bottom indicates: "0 sketch usa 924 bytes (2%) de espaço de armazenamento para programas. O máximo são 32256 bytes. Variáveis globais usam 9 bytes (0%) de memória dinâmica, deixando 2039 bytes para variáveis locais." and "Arduino Uno em /dev/ttyACM0".



Table of Contents

1 Apresentação do Arduino

- Hardware - Arduino Uno (ATmega328)
 - Ambiente de Desenvolvimento (IDE)

2 Comunicação, Sensores e Atuadores

- Comunicando com hardware externo
 - Exemplos de sensores e atuadores

3 Programando um Robô Móvel

- Programando sensores e atuadores
 - Programando o Robô Capivara VSSS 2017



Interagindo com o meio externo

Os robôs são constituídos por um grupo de dispositivos eletromecânicos capazes de realizar trabalhos de maneira autônoma ou pré-programada, assim, precisam interagir com o ambiente a sua volta.

A placa do Arduino sozinha não é capaz de sentir e interagir com o ambiente externo. Para isso é preciso fazer uso de **sensores**, que transformam grandezas físicas em pulsos elétricos e **atuadores**, que transformam pulsos elétricos em grandezas físicas.



Comunicação com Hardware Periférico

Recursos do Arduino que permitem a comunicação com módulos e dispositivos externos:

Comunicação Serial: USART, SPI e I2C

Entradas e Saídas Digitais

Entradas Analógicas: Conversor A/D (Analógico/Digital)

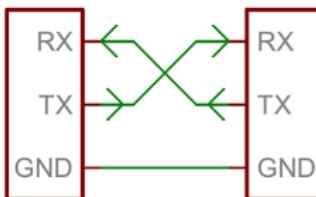
Saídas Analógicas: PWM (Modulação por Largura de Pulso)



USART

USART Universal Synchronous/Asynchronous Receiver Transmitter

- Conexão ponto-a-ponto (somente 2 dispositivos).
 - O Arduino se comunica com o PC no modo assíncrono.
- Exemplos de módulos que usam a USART: GPS, XBee
- Pinos no Arduino Uno: Rx 0, Tx 1



Conexão física entre dois dispositivos em modo assíncrono (UART).



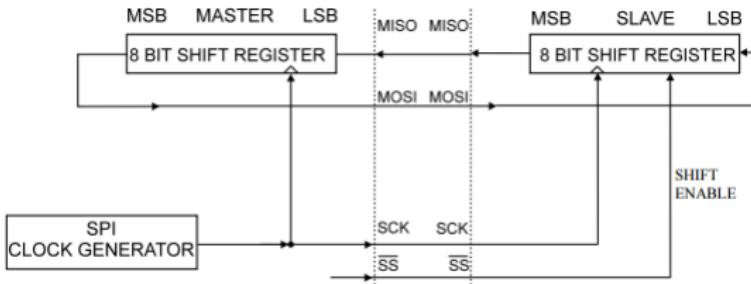
SPI

SPI Serial Peripheral Interface

- Barramento de conexão síncrona.
- Comunicação em altas velocidades e curtas distâncias.
- Um dispositivo *master* pode se comunicar com vários *slaves*.

Exemplos de módulos que usam a SPI: nRF24L01

Pinos no Arduino Uno: MOSI 11, MISO 12, SCK 13, SS 10



Interconexão SPI entre *master* e *slave*.



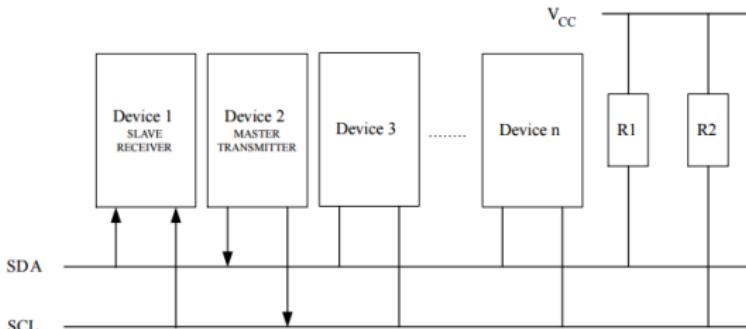
I2C

I²C Inter-Integrated Circuit (*Two Wire Interface*)

- Barramento de conexão síncrona.
 - Projetado para comunicação entre circuitos integrados.
 - Os dispositivo no barramento têm endereços únicos (7 bits).

Exemplos de módulos que usam a I2C: LCD (Serial), EEPROM

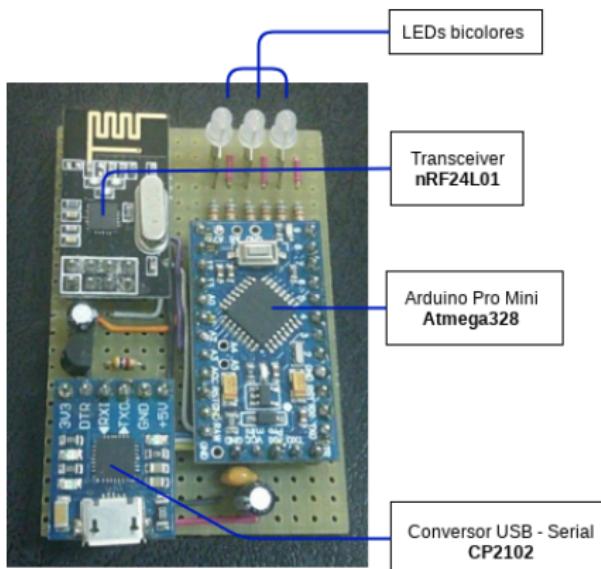
Pinos no Arduino Uno: SDA A4, SCL A5



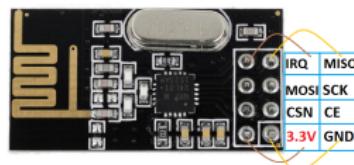
Típico barramento I2C.

Exemplo de Integração de Hardware Externo

Módulos que usam protocolos padrão (UART e SPI).



Placa de rádio de comando dos robôs Capivara VSSS 2017



Módulo de rádio nRF24L01



Conversor USB para
UART-TTL CP2102



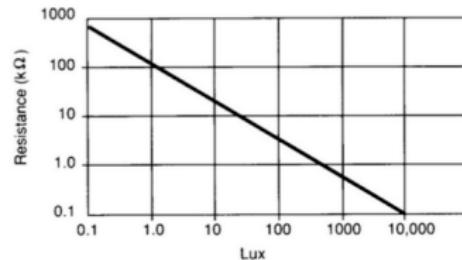
LDR (Light Dependent Resistor)

- Uso da entrada analógica (Conversor A/D).
 - Resposta linear (Resistência vs Intensidade luminosa).

Sketch: `ldr.ino`



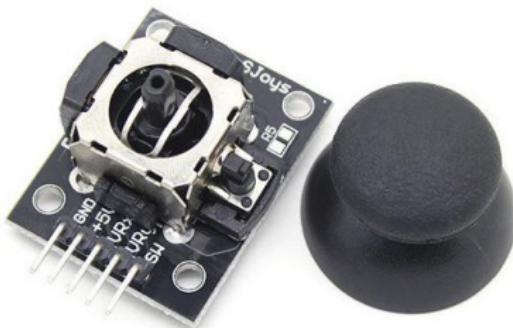
Robô seguidor de linha.



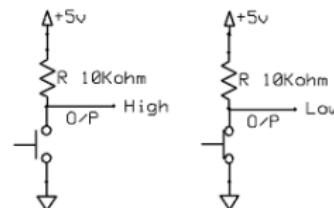
Curva característica de um LDR

Potenciômetro e Pull-up

- Lendo mais de uma entrada analógica simultaneamente.
 - Uso de resistor pull-up (interno e externo).
- Sketch: joystick.ino



Módulo JoyStick.

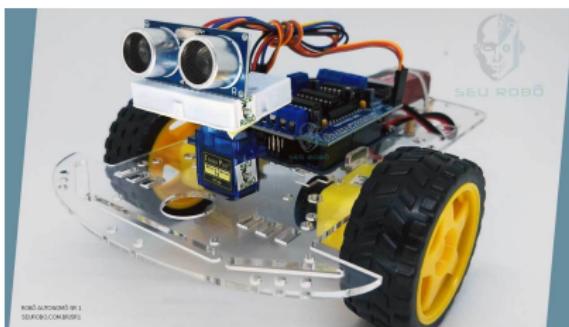


Conexão do resistor de pull-up



Sensor de Ultrassom

- Detectar obstáculos e medir distâncias.
- Sketch: ultrassom.ino



Robô autônomo com sensoreamento de ambiente por ultrassom usando o módulo HC-SR04 com servomotor.



Módulo ultrassom HC-SR04

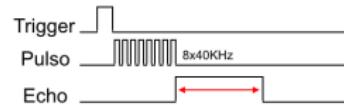
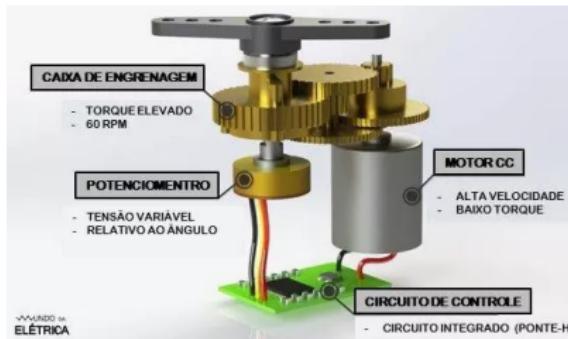


Diagrama de tempo do sensor
HC-SR04

Servo motor

- Controle de movimento com posicionamento de alta precisão.

Sketch: servomotor.ino



Estrutura interna de um servomotor.



Micro Servo 9g

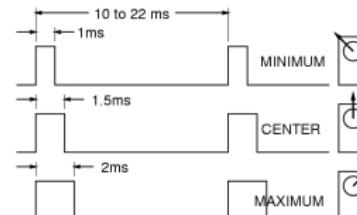


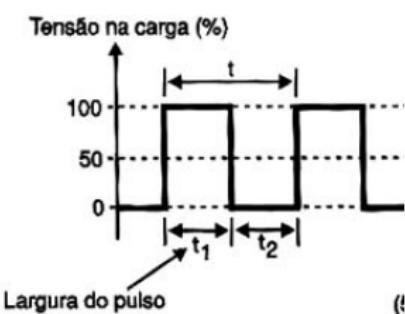
Diagrama de tempo.



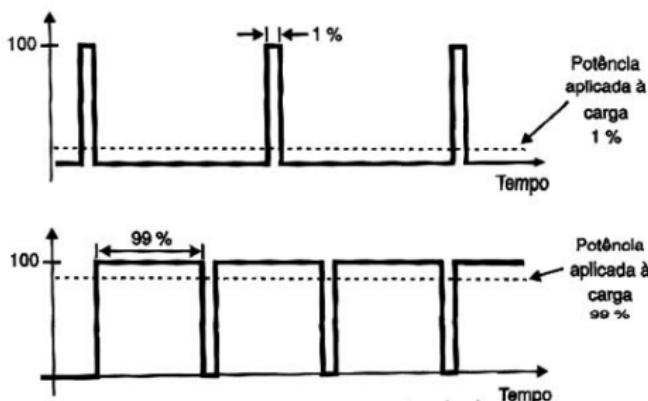
PWM - Pulse Width Modulation

- Saída "análogica", usada para controle de potência.

Sketch: led_pwm.ino



Tensão média na carga.

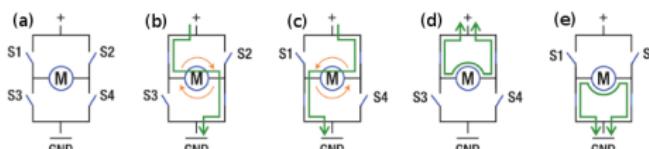
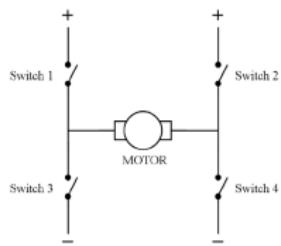


Controlando a potência pelo ciclo ativo.

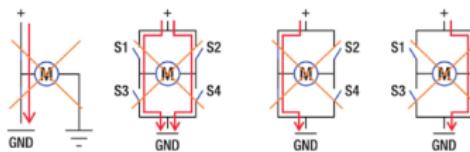


Ponte H

- Permite o controle de motores CC de forma reversível e gradual.

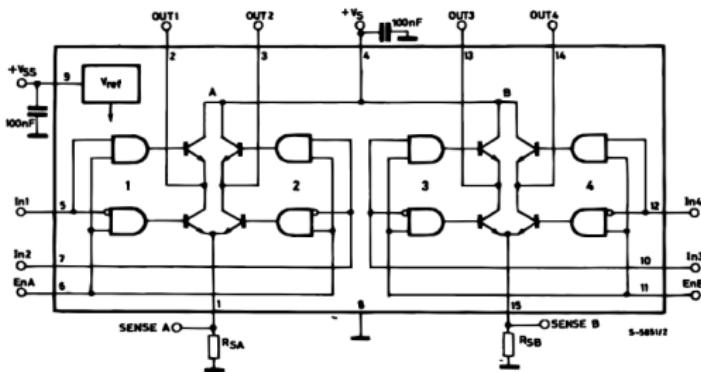
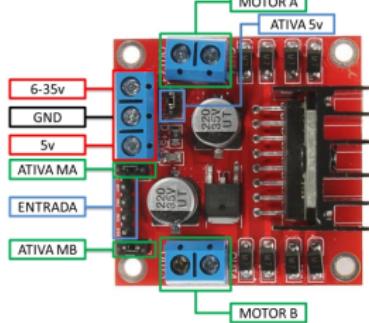


Configurações seguras para uma Ponte H



Ponte H - CI L298N

Módulo com o Circuito Integrado L298N.



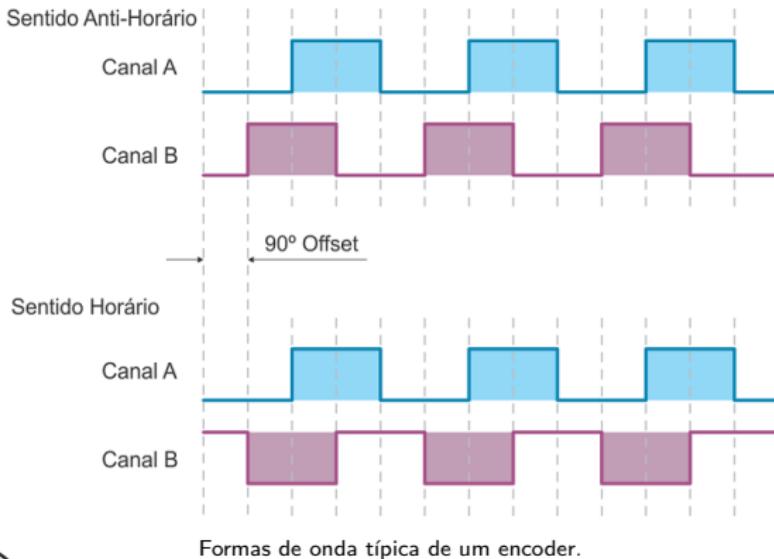
Módulo de Ponte H com o CI L298N.

Diagrama interno do CI L298N.

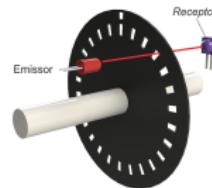


Encoder

- Quantizar distâncias, controlar velocidades, medir ângulos, número de rotações, realizar posicionamentos, rotacionar braços robóticos, etc.



Encoder magnético.



Princípio do encoder por barreira óptica

Interrupção Externa

- Interrompe a temporariamente tarefa da CPU, atende o dispositivo chamador e logo após, o programa retoma seu processamento do ponto onde havia parado.
- Resolve problemas de temporização, atendendo tarefas urgentes.

Pinos no Arduino Uno: INTA 2, INTB 3

Modos de acionamento da interrupção do Arduino:
LOW, CHANGE, RISING, FALLING.

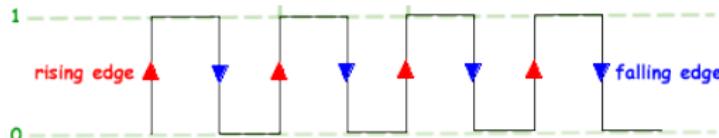


Table of Contents

1 Apresentação do Arduino

- Hardware - Arduino Uno (ATmega328)
- Ambiente de Desenvolvimento (IDE)

2 Comunicação, Sensores e Atuadores

- Comunicando com hardware externo
- Exemplos de sensores e atuadores

3 Programando um Robô Móvel

- Programando sensores e atuadores
- Programando o Robô Capivara VSSS 2017



Programando sensores e atuadores

Arduino Uno:

Sketch: ldr.ino

Sketch: joystick.ino

Sketch: led_pwm.ino

Sketch: ultrassom.ino

Sketch: servomotor.ino

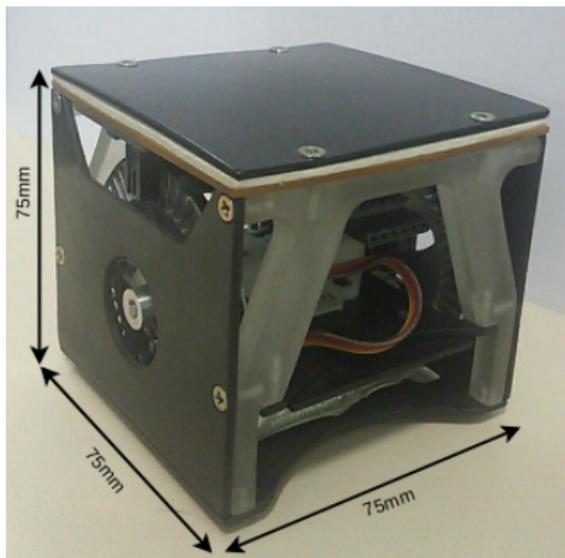


Categoria IEEE VSSS (Very Small Size Soccer)



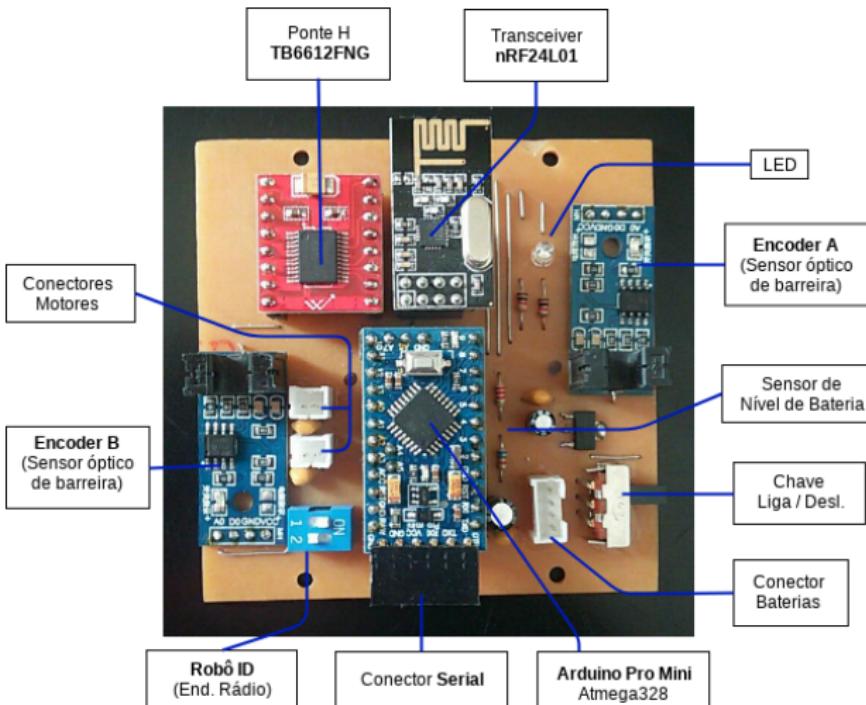
Robô Capivara VSSS 2017

LARC Latin-American Robotics Competition (2017)



Robô Capivara VSSS 2017

Placa eletrônica



Placa eletrônica do Robô Capivara VSSS 2017



Conexão dos Módulos

```
/* ****
 * Estas são as conexões de hardware mapeadas aos pinos do Arduino */
#define LED      4      // Led conectado a uma saída digital
#define RADIO_CE 7      // Pino CE do módulo de rádio
#define RADIO_CS 8      // Pino CS do módulo do rádio
#define RADIO_A0 A4     // Bit 0 do end. do rádio (LOW = ligado)
#define RADIO_A1 A5     // Bit 1 do end. do rádio (LOW = ligado)

#define IRQ_ENC_A 2      // Pino de interrupção do Encoder A
#define IRQ_ENC_B 3      // Pino de interrupção do Encoder B
#define IRQ_RADIO 5      // Pino de interrupção do Rádio

#define HBRID_EN 6      // Habilita a ponte H (High)
#define MTR_AIN1 A2     // Bit 0 - Controle da ponte H do Motor A
#define MTR_AIN2 A3     // Bit 1 - Controle da ponte H do Motor A
#define MTR_BIN1 A1     // Bit 0 - Controle da ponte H do Motor B
#define MTR_BIN2 A0     // Bit 1 - Controle da ponte H do Motor B
#define MTR_PWM_A 9      // Sinal de PWM para controle do Motor A
#define MTR_PWM_B 10     // Sinal de PWM para controle do Motor B

#define VOLT_BAT A7      // Tensão da bateria -> Vcc/10
```



Conexão dos Módulos do Robô Capivara VSSS 2017

Programando o Robô Capivara VSSS 2017

Robô Capivara VSSS 2017

- Acionamento da Ponte H
 - Encoder usando interrupções
 - Leitura de múltiplas chaves
 - Configuração alternativa do ADC

Sketch: robo_basico.ino



Muito Obrigado!

Os sketches referenciados nesta apresentação estão disponíveis em:
<https://github.com/alex-co/ROSIE2020>

