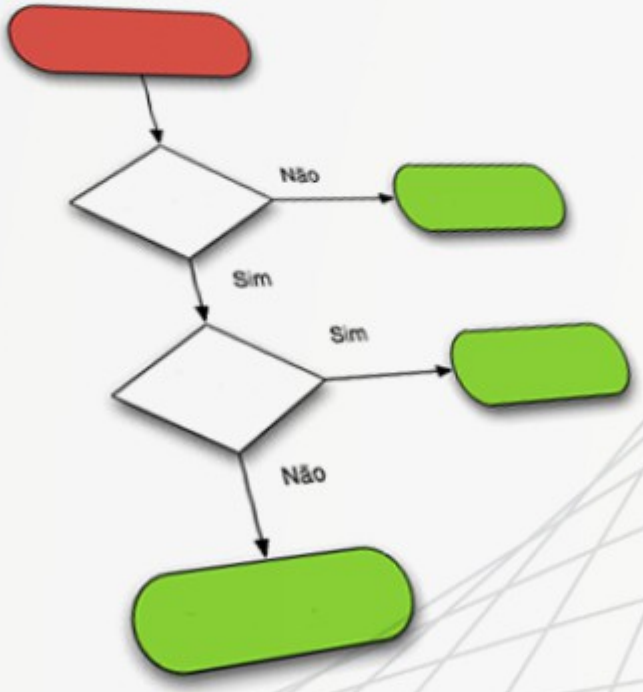
Programação II

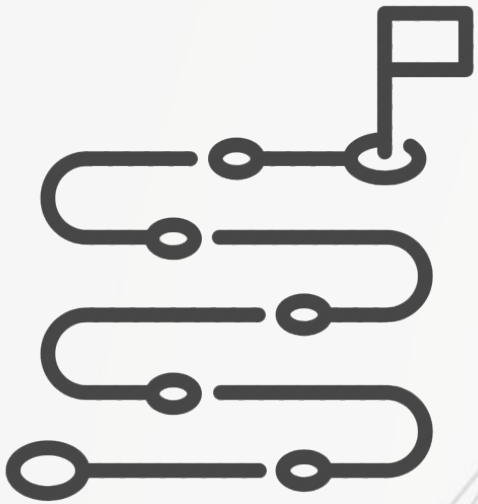
Comunicação Serial



Prof. Maurício Edgar Stivanello

Prof. Delcino Picinin Júnior

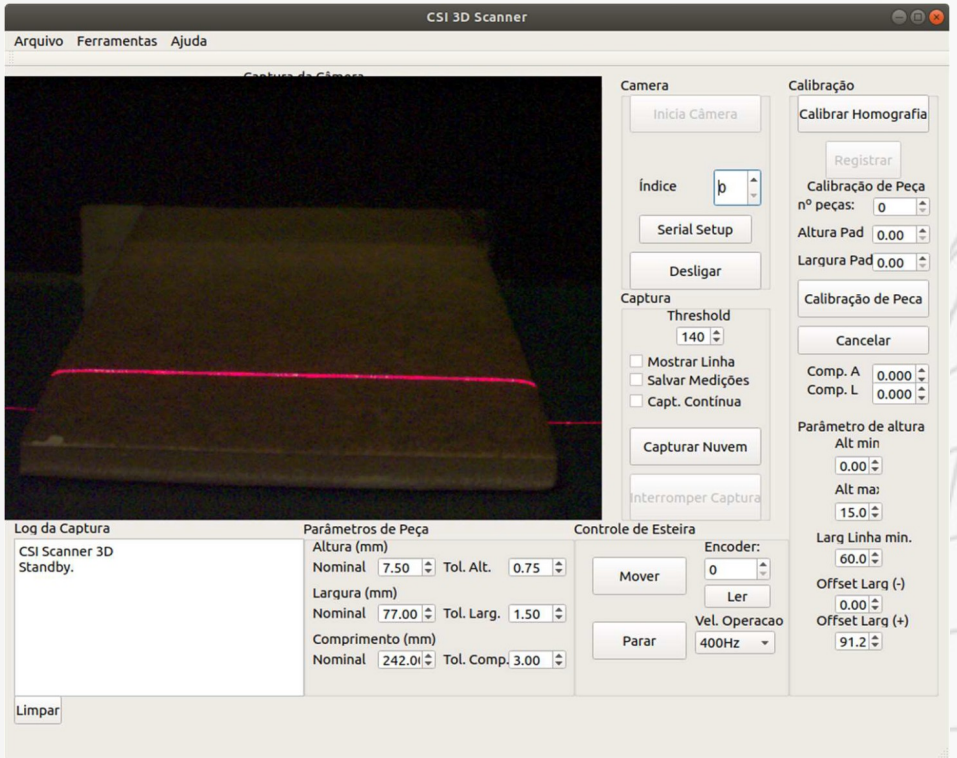
- Plano de Aula
  - **Exemplo de aplicação**
  - Comunicação Serial
  - Comunicação Serial no Arduino
  - Exemplos e Exercícios



Comunicação Serial

- Exemplo de equipamento: Esteira Transportadora
  - Utilizada em um sistema de medição de peças cerâmicas
  - Através da IHM é possível controlar a esteira e também obter dados dos sensores

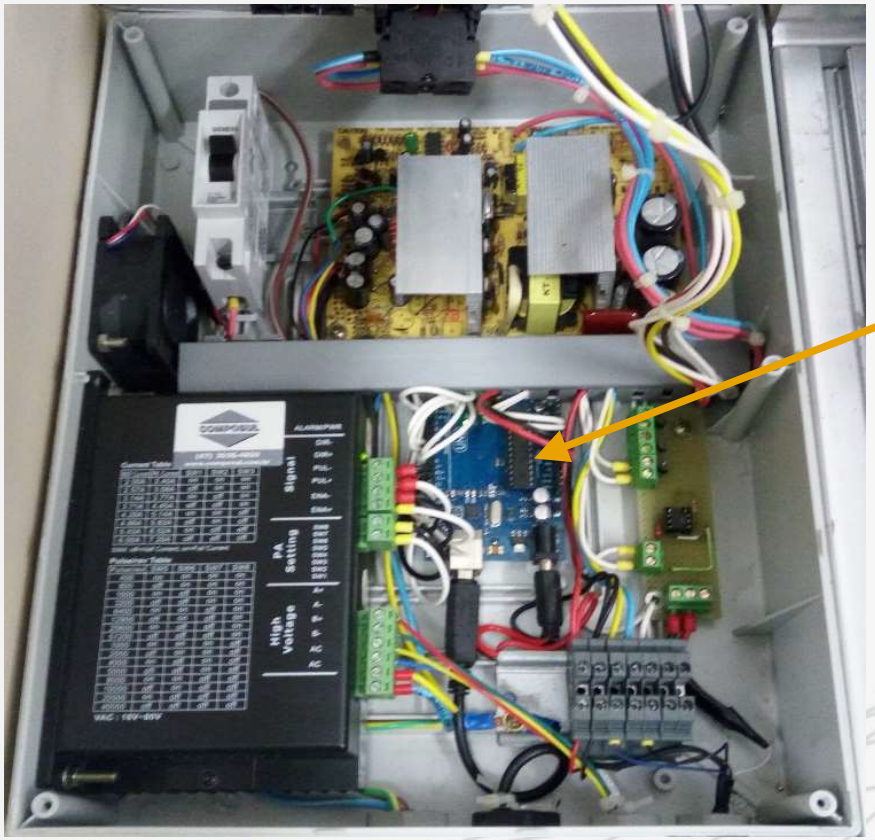
Esteira transportadora



Interface Homem-Máquina

Comunicação Serial

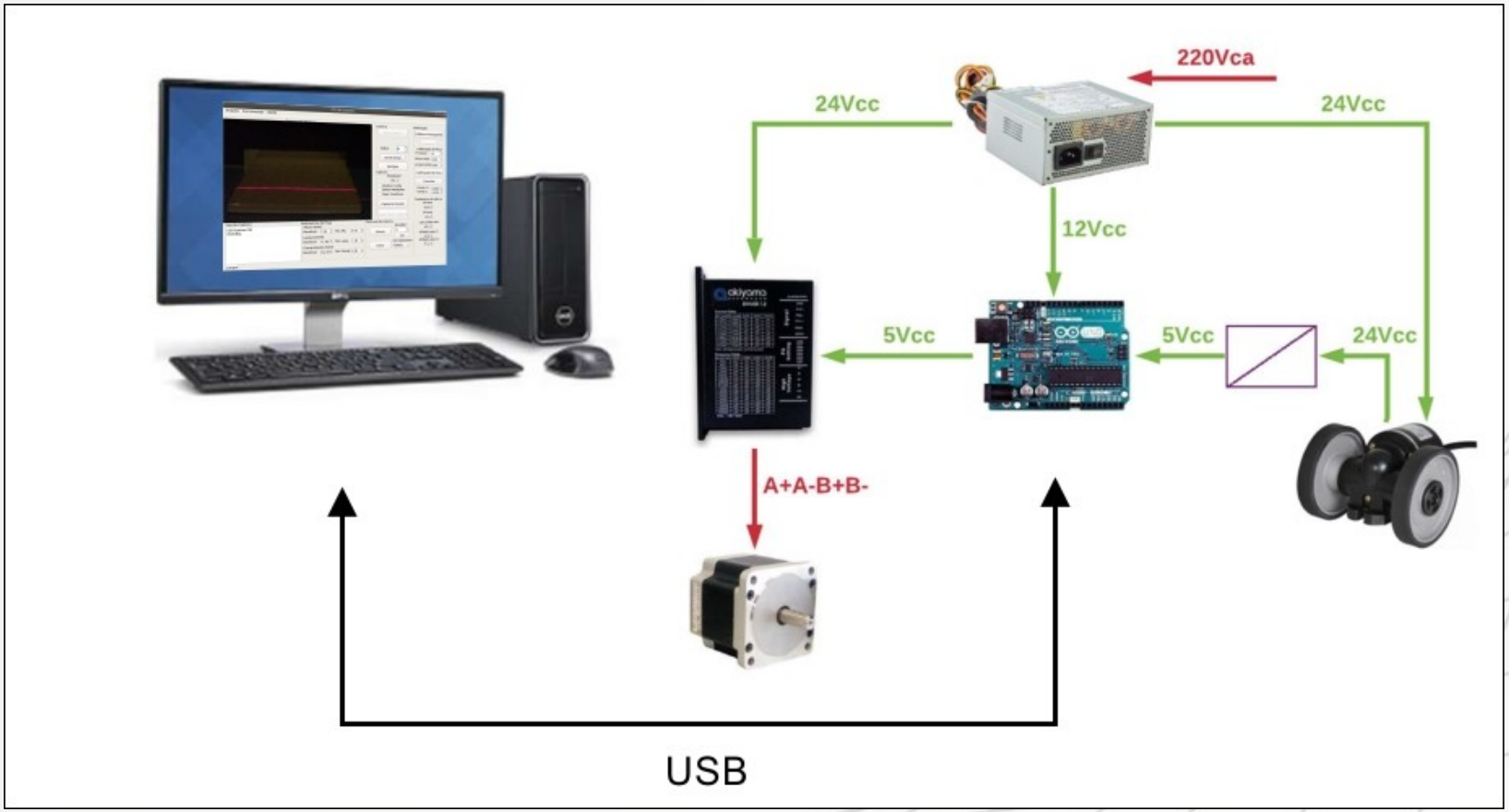
- Exemplo de equipamento: Esteira Transportadora
  - Os sensores e atuadores estão conectados a um microcontrolador
  - Deve haver uma transmissão de dados entre o PC com a IHM e o microcontrolador



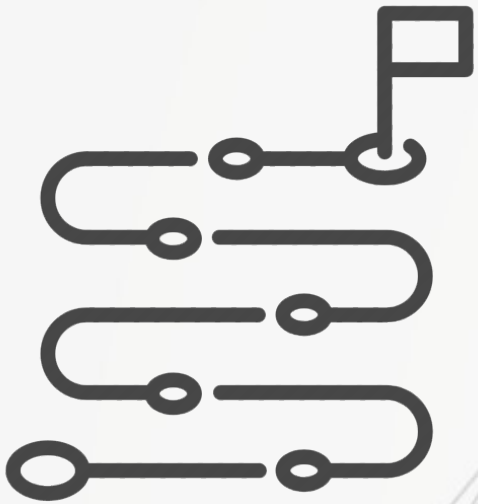
Microcontrolador

Comunicação Serial

- Exemplo de equipamento: Esteira Transportadora
  - O microcontrolador está ligado ao PC por uma interface serial (USB)

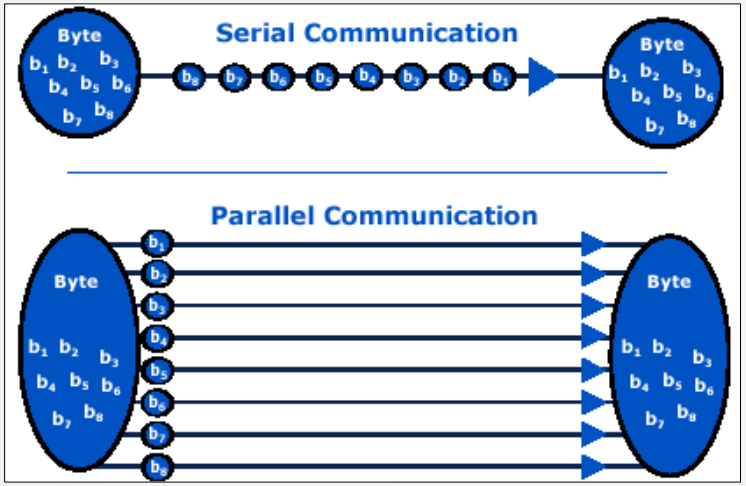


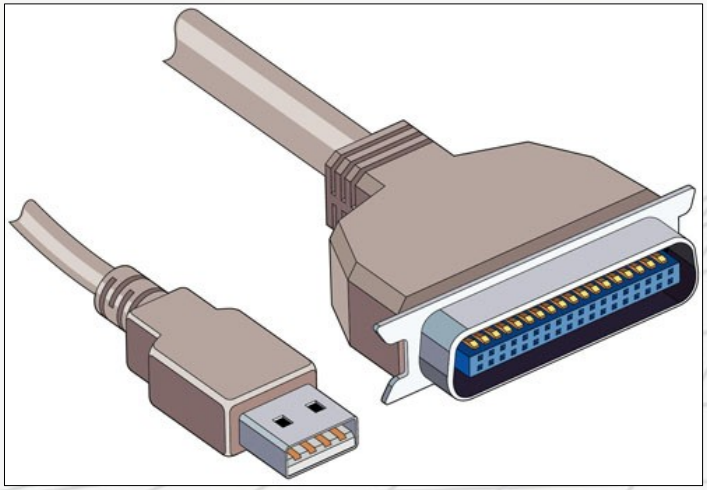
- Plano de Aula
  - Exemplo de aplicação
  - **Comunicação Serial**
  - Comunicação Serial no Arduino
  - Exemplos e Exercícios



Comunicação Serial

- Transmissão de dados digitais
  - Corresponde a transferência de dados (fluxo digital de bits ou sinais analógicos digitalizados) sobre um canal de comunicação.
  - A transmissão de dados pode acontecer de forma paralela ou serial:
    - Transmissão Paralela: todos os bits de um dado símbolo a ser enviados são transmitidos ao mesmo tempo.
    - Transmissão Serial: os bits de um dado símbolo são transmitidos um a um.

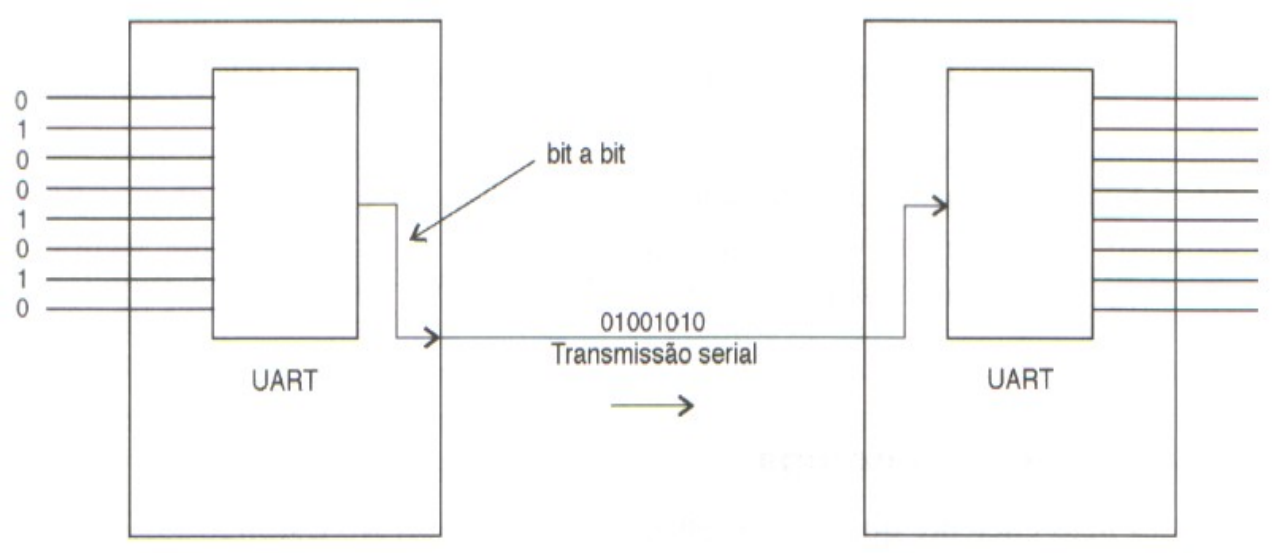




Comunicação Serial

Interfaces / Portas Seriais

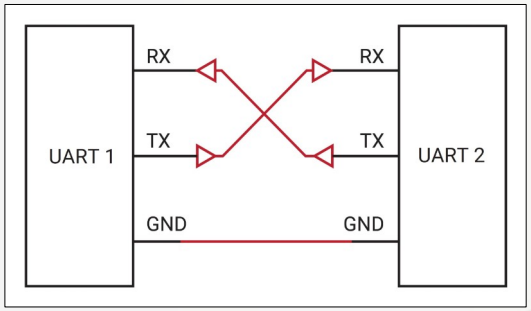
- Periféricos que convertem dados em paralelo para dados seriais: Transformam bytes de dados em bits individuais passíveis de transmissão.
- A operação inversa também é realizada: Os bits recebidos são convertidos para bytes novamente.



Comunicação Serial

UART - Universal Asynchronous Receiver Transmitter

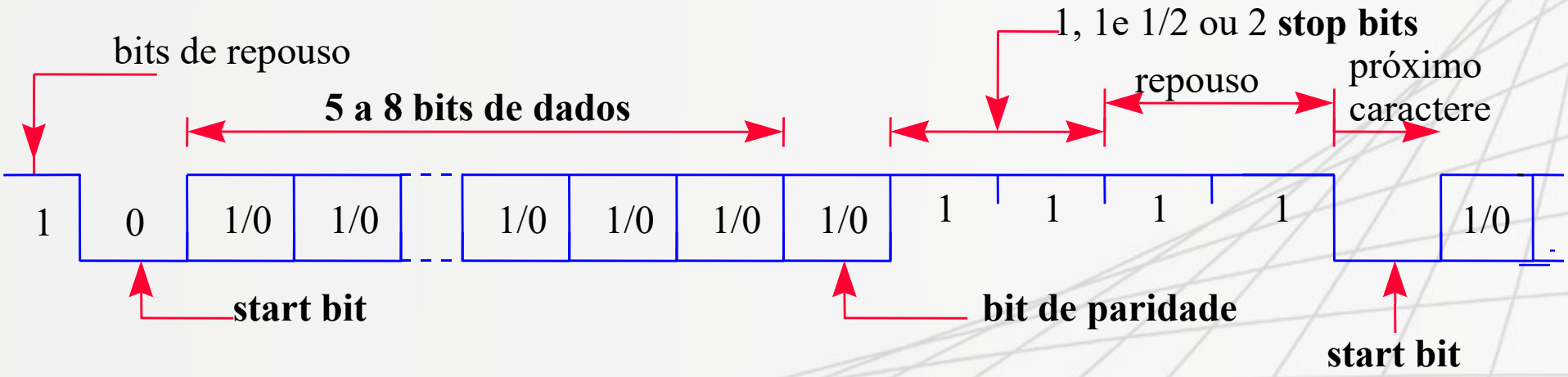
- Um dos tipos de chip responsável pela serialização das informações.
- Trabalha baseado em comunicação assíncrona:
  - Faz uso de bits marcadores que identificam o início, fim e corretude dos dados enviados.
- Deve ser configurado conforme as necessidades de transmissão.



Comunicação Serial

Configurações comuns da UART

- **Taxa de transmissão**: Velocidade de comunicação (bits por segundo). Exemplo: 9600 bps
- **Número de bits de dados**: Tamanho da palavra de dados. (*Padrão do Arduino: 8*).
- **Paridade**: Verificação de conscistência entre a informação transmitida e recebida. (Sem paridade / Par / Ímpar) (*Padrão do Arduino: sem paridade*)
- **Start Bit e Stop Bit**: Permite configurar como serão os bits que identificam o início e fim da das palavras transmitidas. (*Padrão do Arduino: 1 bit*)

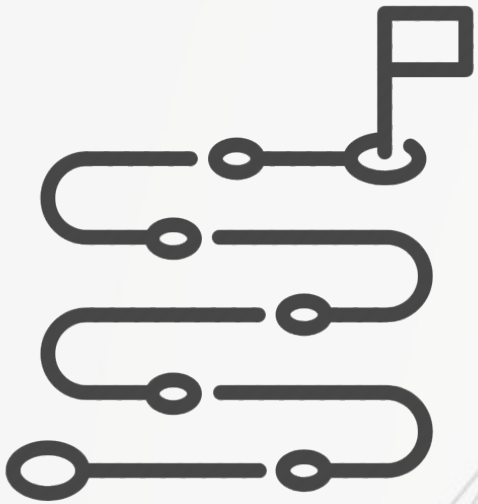


Comunicação Serial

Comunicação com uma UART no Windows

- **Portas de comunicação**: Ao se conectar um dispositivo com uma UART no barramento USB será criada uma porta de comunicação nomeada por COMX:
  - Exemplo: COM2
- As seguintes etapas estão envolvidas na comunicação através de uma porta:
  - Abertura da porta de comunicação
  - Configuração da porta de comunicação
  - Leituras e escritas
  - Fechamento da porta de comunicação
- A forma de utilização depende das ferramentas e da plataforma sobre a qual está sendo realizado o desenvolvimento.

- Plano de Aula
  - Exemplo de aplicação
  - Comunicação Serial
  - **Comunicação Serial no Arduino**
  - Exemplos e Exercícios

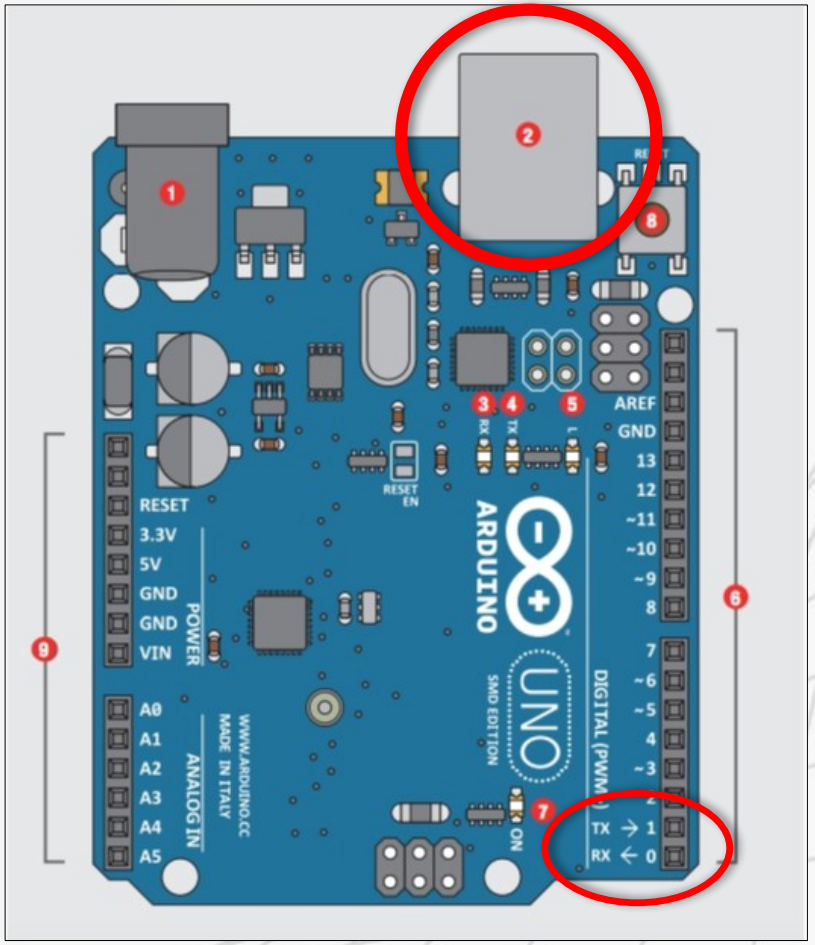


INSTITUTO FEDERAL
SANTA CATARINA

Comunicação Serial

Serial X Arduino

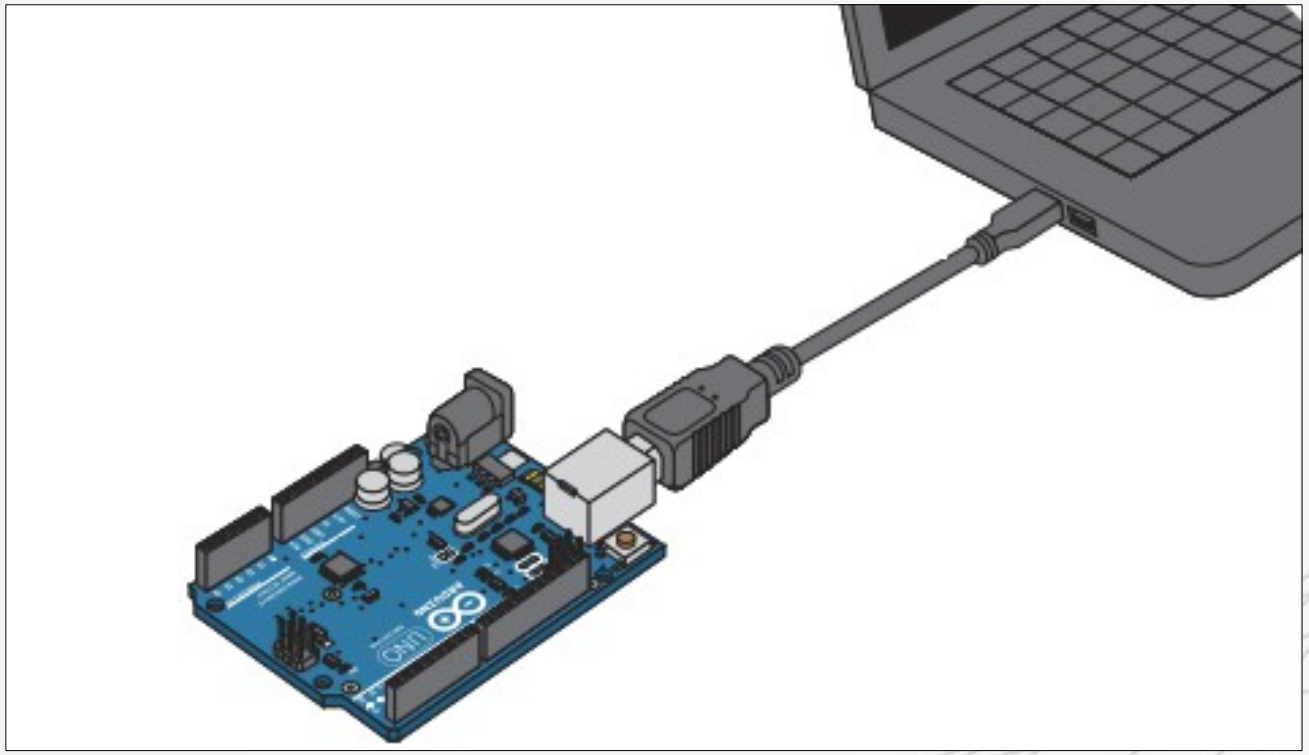
- As placas arduino possuem pelo menos uma UART
- Com ela é possível realizar comunicações através dos pinos 0 (RX) e 1 (TX), assim com o computador através da USB.
- É possível utilizar a ferramenta “**Serial monitor**” do ambiente de desenvolvimento para comunicações com o Arduino, ou ainda desenvolver um aplicativo customizado através de uma linguagem de programação.



INSTITUTO FEDERAL
SANTA CATARINA

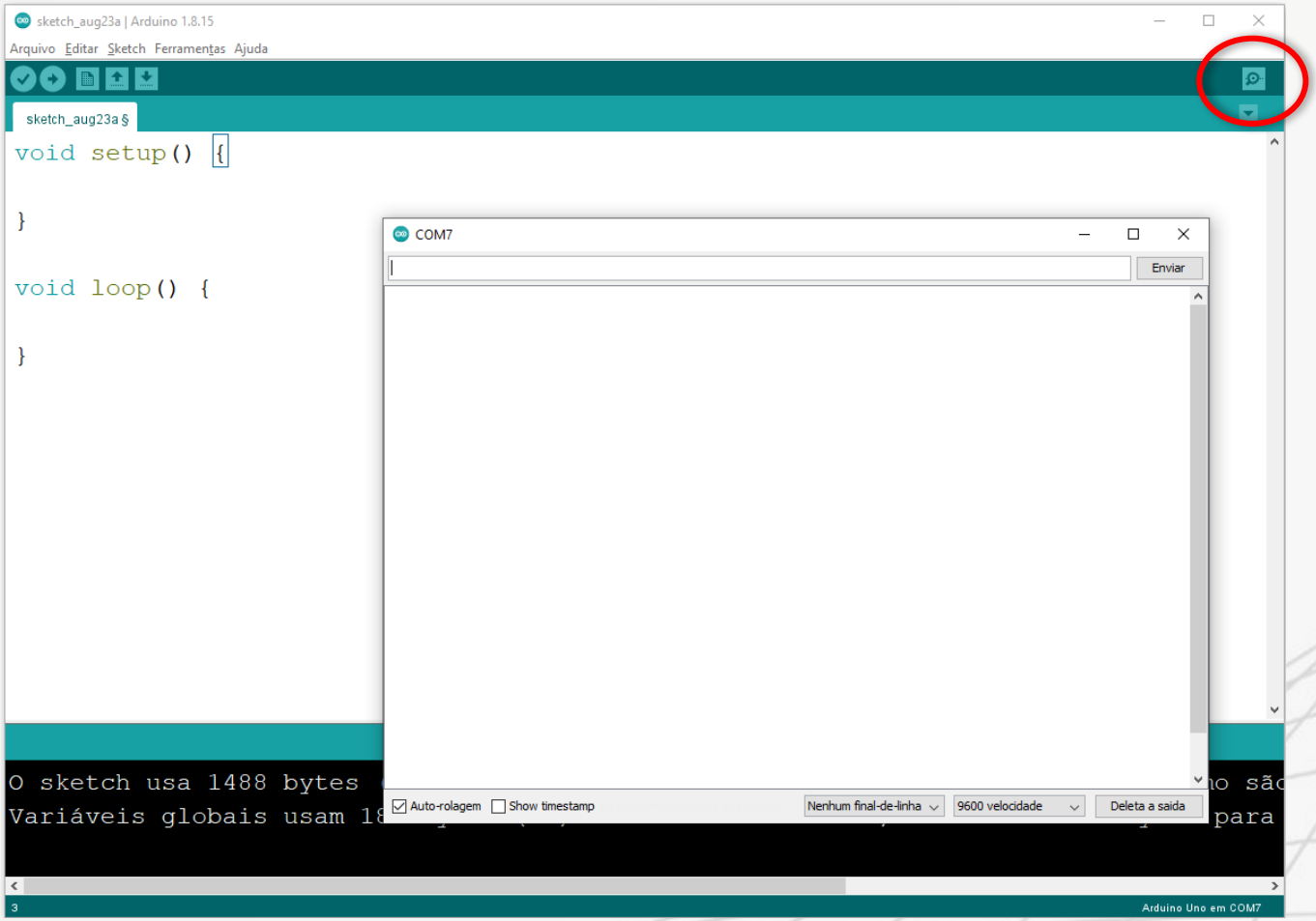
Comunicação Serial

- Conexão do Arduino ao barramento USB do computador



Comunicação Serial

- Serial Monitor



Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.begin()**

- Descrição:
  Configura a taxa de transferência e inicia a comunicação.
- Sintaxe
  Serial.begin(speed)
- Parâmetro
  speed: a taxa de transmissão em bits per second

```
void setup() {
    Serial.begin(9600); // Configura e inicia a porta serial
}
```

Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.available()**

- Descrição:
  Confere o número de bytes (caracteres) disponíveis para leitura da porta serial.
- Sintaxe
  = Serial.available()
- Retorna
  O número de bytes disponíveis para leitura.

```
void loop() {
  if (Serial.available() != 0)
  {
    ...
```

Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.read()**

- Descrição:
  Lê dados recebidos na porta serial.
- Sintaxe
  = Serial.read()
- Retorna
  O primeiro byte de dados recebidos disponível.

```
void loop() {
  // apenas responde quando dados são recebidos:
  if (Serial.available() > 0) {
    // lê do buffer o dado recebido:
    char caracter = Serial.read();
    ...
```

Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.print()**

- Descrição:
  Envia dados na porta serial em como texto ASCII
- Sintaxe
  Serial.print(val)
- Parâmetro
  val: o valor a ser impresso - qualquer tipo de dados

```
Serial.print(78) envia "78"

Serial.print(1.23) envia "1.23"

Serial.print('N') envia "N"

Serial.print("Bom dia.") envia "Bom dia."
```

INSTITUTO FEDERAL
SANTA CATARINA

Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.println()**

- Descrição:
  Essa função assume as mesmas formas que Serial.print(), porém adiciona ao final do valor a ser enviado um caratere de retorno de carruagem (ASCII 13, ou '\r') e um cactere de nova linha (ASCII 10, ou '\n')

INSTITUTO FEDERAL
SANTA CATARINA

Comunicação Serial

- Comandos da comunicação Serial da Linguagem de Programação do Arduino

Função **Serial.end()**

- Descrição:
  Desativa a comunicação serial
- Sintaxe
  Serial.end()

Comunicação Serial

- Exemplo de utilização

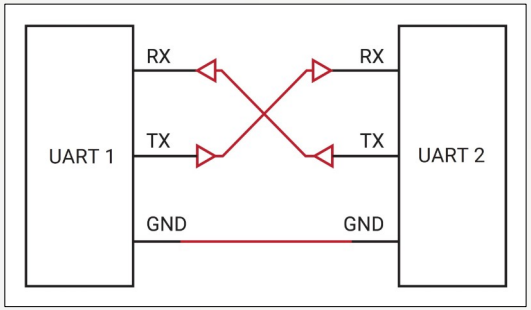
Depuração

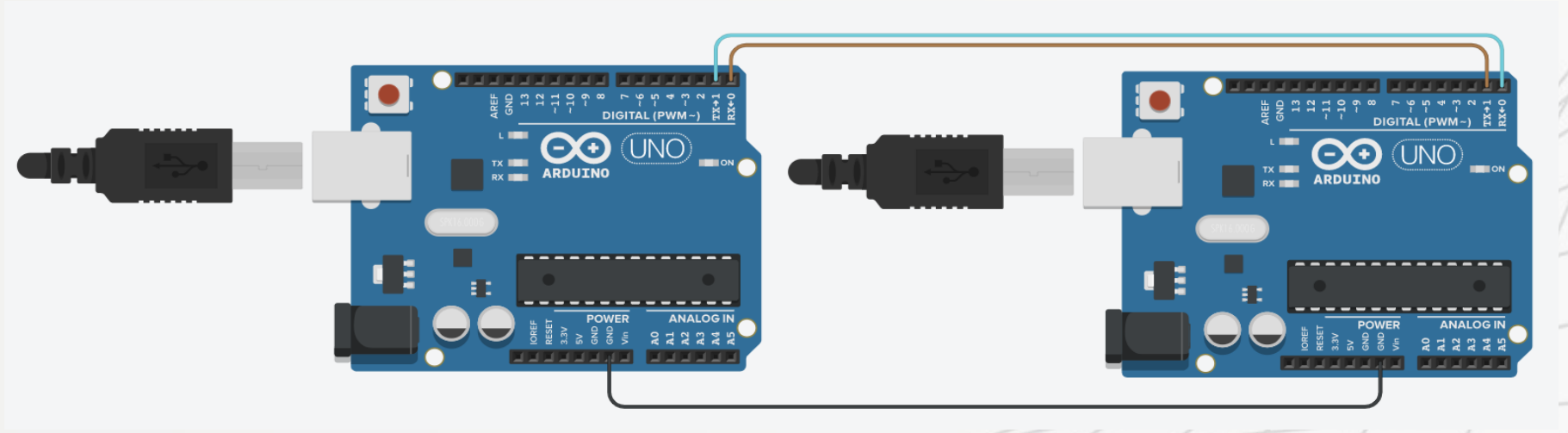
```
void setup() {
  pinMode(10, INPUT);
  Serial.begin(9600);
  Serial.println("Vai iniciar o monitoramento do botão ...");
}

void loop() {
  if (digitalRead(10) == HIGH)
  {
    Serial.println("O botão foi pressionado");
    ...
```

Comunicação Serial

- Exemplo de utilização

Comunicação entre dispositivos – Arduino ↔ Arduino

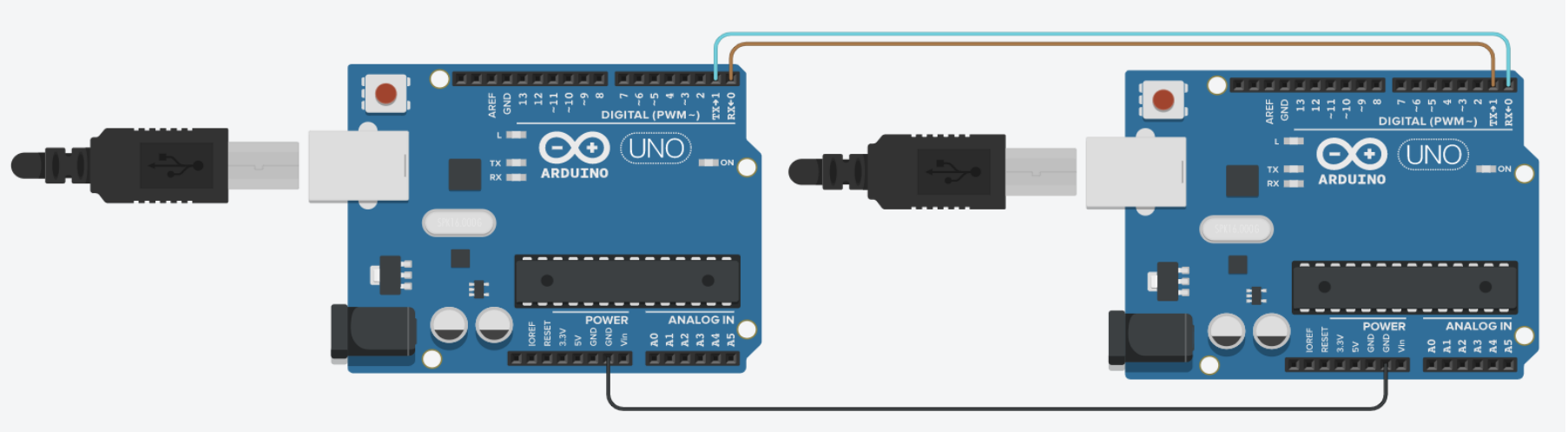




Exemplo 1 – Ping pong baseado em caractere

```
void setup() {
  Serial.begin(9600);
  pinMode(13,OUTPUT);
  delay(5000);
  Serial.print("L");
}

void loop() {
  if (Serial.available())
  {
    char c = Serial.read();
    if (c == 'L')
    {
          digitalWrite(13,HIGH);
          delay(500);
          digitalWrite(13,LOW);
          delay(500);
          Serial.print("L");
    }
  }
}
```

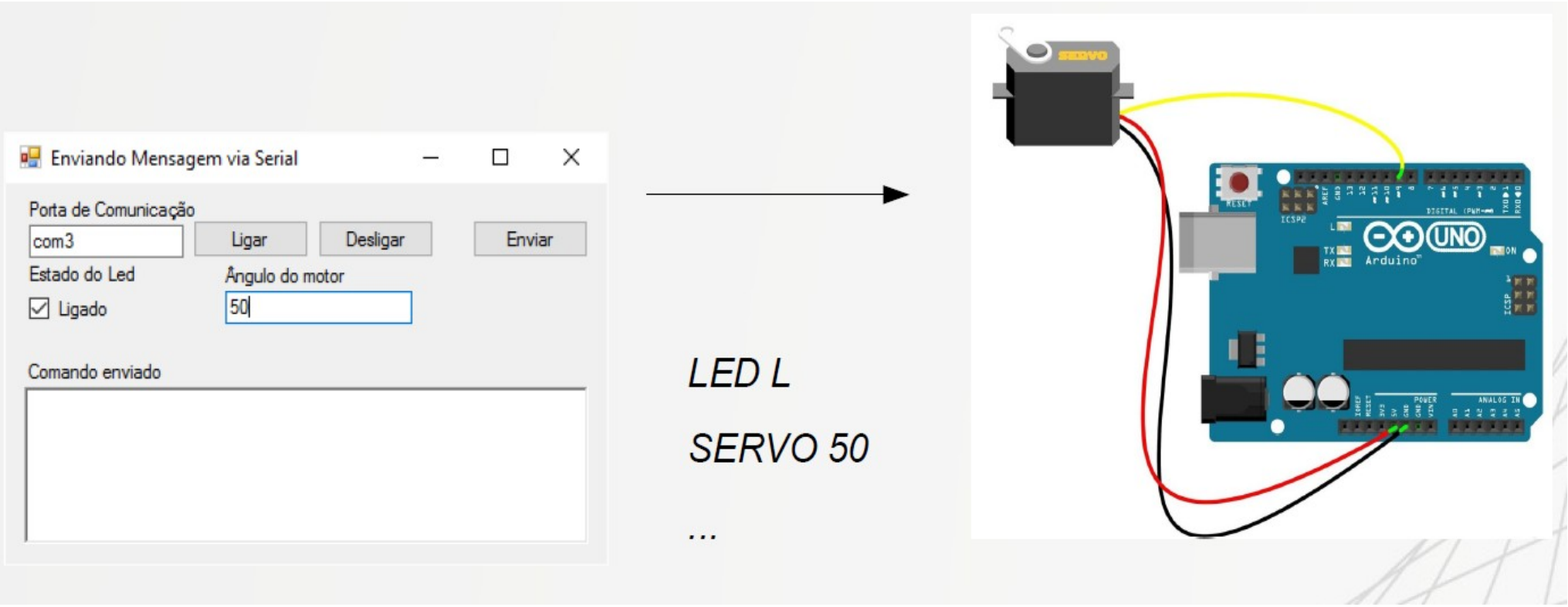
```
void setup() {
  Serial.begin(9600);
  pinMode(13,OUTPUT);
}

void loop() {
  if (Serial.available())
  {
    char c = Serial.read();
    if (c == 'L')
    {
          digitalWrite(13,HIGH);
          delay(500);
          digitalWrite(13,LOW);
          delay(500);
          Serial.print("L");
    }
  }
}
```



Exemplo 2 – Mensagens em cadeias de caractere separados por quebra de linha

```
String comando = ""; // onde será montado o comando
bool comandoCompleto = false; // indica a ocorrência de um comando completo

void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  // enquanto existirem bytes na serial e o comando não está completo
  while (!comandoCompleto && Serial.available()) {
    char caracter = (char) Serial.read();
    // se o caracter é uma quebra de linha, indica conclusão do comando
    if (caracter == '\n') {
      comandoCompleto = true;
    } else{
      // adiciona caracter ao comando
      comando += caracter;
    }
  }

  // se o comando está completo, o executa
  if (comandoCompleto)
  {
    comando.trim();
    if (comando == "Ligar led")
    {
      digitalWrite(13, HIGH);
    }
    else if (comando == "Desligar led")
    {
      digitalWrite(13, LOW);
    }

    comando = "";
    comandoCompleto = false;
  }

}
```

Exemplo 3 – Comandos + Parâmetros (separados por espaço e quebra de linha)



Exemplo 3 – Comandos + Parâmetros (separados por quebra de linha e espaço)

Parte A

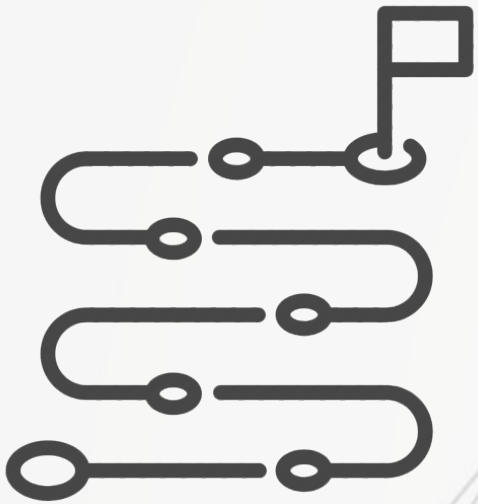
```
#include <Servo.h>
#define LED 13
#define SERVO 9
Servo servoMotor;
String comando = ""; // string para os comandos recebidos
boolean comandoCompleto = false; // indica quando um comando foi completamente recebido
void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
  servoMotor.attach(SERVO);
}
void loop() {
  // quando um comando completo chegou, interpreta
  if (comandoCompleto) {
    interpretaComando();
    // limpa a string do comando interpretado
    comando = "";
    comandoCompleto = false;
  } delay(50);
}
```

Exemplo 3 – Comandos + Parâmetros (separados por quebra de linha e espaço)

Parte B

```
//evento que é invocado quando dados chegam pela serial
void serialEvent() {
  while (Serial.available()) {
    // pega byte do buffer
    char caracter = (char)Serial.read();
    // se o caracter é uma quebra de linha, indica a flag comandoCompleto
    //para que o comando seja processado
    if (caracter == '\n') {
      comandoCompleto = true;
    } else {
      // adiciona caracter ao comando
      comando += caracter;
    }
  }
}
void interpretaComando() {
  String parte1, parte2;
  parte1 = comando.substring(0, comando.indexOf(' '));
  parte2 = comando.substring(comando.indexOf(' ') + 1);
  parte2.trim();
  // COMANDO LED
  if (parte1.equals("LED"))
  {
    if (parte2.equals("L"))
    {
      digitalWrite(LED, HIGH);
    } else if (
      parte2.equals("D"))
    {
      digitalWrite(LED, LOW);
    }
  } // COMANDO SERVO
  else if (parte1.equals("SERVO"))
  {
    int angulo = parte2.toInt();
    servoMotor.write(angulo);
  }
}
```

- Plano de Aula
  - Exemplo de aplicação
  - Comunicação Serial
  - Comunicação Serial no Arduino
  - **Exemplos e Exercícios**



INSTITUTO FEDERAL
SANTA CATARINA

Comunicação Serial

Exercícios

1) Reproduza os projetos de exemplo da aula no Tinkercad e avalie o funcionamento.
2) Crie um projeto contendo dois Arduinos interligados de modo que possam realizar comunicações seriais. Ao Arduino A estarão conectados 3 botões. No Arduino B estarão conectados 3 leds. Programe os Arduinos de modo que o pressionamento de cada botão no Arduino A pisque um dos leds no Arduino B.
3) Crie um projeto contendo dois Arduinos interligados de modo que possam realizar comunicações seriais. Ao Arduino A estarão conectados 1 led e um servo motor. Ao Arduino B estarão conectados 2 botões e um potênciometro. Neste circuito, um dos botões serve para ligar o led, enquanto o outro serve para desligar o led. Por sua vez, o potenciômetro serve para indicar o ângulo do servo motor.

Dúvidas?

mauricio.stivanello@ifsc.edu.br