**UNIVERSIDADE DO VALE DO RIO DOS SINOS (UNISINOS)**
**GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO**

**MATHEUS DE FREITAS WEBER**
**GABRIEL BERWANGER SILVEIRA**

# EXERCÍCIO - AULA 03
## Sistema Cliente-Servidor e comunicação serial

**São Leopoldo**
**2025**

MATHEUS DE FREITAS WEBER
GABRIEL BERWANGER SILVEIRA

# EXERCÍCIO - AULA 03
## Sistema Cliente-Servidor e comunicação serial

Trabalho apresentado para a matéria Circuitos microprocessados pelo Curso de Engenharia de Controle e Automação e Engenharia da Computação da Universidade do Vale do Sinos (UNISINOS), ministrada pelo Prof.Jean Schmith.

**São Leopoldo**
**2025**

# Sumário


**1 Introdução** **3**

**2 Fundamentação teórica** **4**
2.1 Arquitetura Cliente-Servidor . . . . . 4
2.2 Comunicação Serial . . . . . 4
2.3 GPIO (General Purpose Input/Output) . . . . . 4
2.4 Chip Select (CS) . . . . . 4
2.5 Microcontroladores . . . . . 4

**3 Metodologia** **5**
3.1 Materiais Utilizados . . . . . 5
3.2 Diagrama de Conexões . . . . . 5
3.3 Configuração dos Microcontroladores . . . . . 5
3.3.1 Microcontrolador Servidor . . . . . 5
3.3.2 Microcontroladores Clientes . . . . . 6
3.4 Implementação do Código . . . . . 7
3.4.1 Código do Microcontrolador Servidor . . . . . 7
3.4.2 Código do Microcontrolador Cliente . . . . . 8

**4 Resultados** **9**

**5 Conclusão** **10**
5.0.1 Link do TinkerCAD . . . . . 10

# 1 Introdução

Em sistemas embarcados a compreensão da comunicação entre microcontroladores e integração efeciente entre hardware e software é essencial. Entro os modelos utilizados nesse contexto destaca-se a arquitetura Cliente-Servidor que permite centralizar o controle, dessa forma coordenando o funcionamento das unidades subordinadas.

Neste trabalho, exploramos a implementação de um sistema de comunicação serial entre três microcontroladores associando o uso de GPIOs como chip select para a correta identificação dos dispositivos que devem responder aos comandos enviados. Através de botões físicos vinculados aos microcontroladores do tipo cliente o usuário pode fazer o acionamento e desligamento individual dos LEDs conectados ao microcontrolador correspondente. O microcontrolador do tipo servidor é responsável por receber os comandos e repassá-los aos clientes corretos, garantindo a execução das ações solicitadas.

# 2 Fundamentação teórica

## 2.1 Arquitetura Cliente-Servidor

A arquitetura Cliente-Servidor é um modelo de comunicação onde um dispositivo (o cliente) solicita serviços ou recursos de outro dispositivo (o servidor). O servidor é responsável por fornecer esses serviços ou recursos, enquanto o cliente inicia a comunicação e faz as solicitações. Essa arquitetura é amplamente utilizada em redes de computadores, sistemas distribuídos e aplicações web.

## 2.2 Comunicação Serial

A comunicação serial é um método de transmissão de dados onde os bits são enviados sequencialmente, um após o outro, através de um único canal de comunicação. Esse método é eficiente para longas distâncias. Para comunicação entre microcontroladores é necessário definir parâmetros como taxa de transmissão (baud rate), paridade, bits de dados e bits de parada para garantir a integridade dos dados transmitidos.

## 2.3 GPIO (General Purpose Input/Output)

GPIOs são pinos de um microcontrolador que podem ser configurados como entradas ou saídas digitais. Eles são usados para interagir com outros dispositivos eletrônicos, como sensores, atuadores e outros microcontroladores. Em sistemas embarcados, os GPIOs são essenciais para a comunicação e controle de hardware.

## 2.4 Chip Select (CS)

O Chip Select (CS) é um sinal utilizado em sistemas de comunicação serial para selecionar um dispositivo específico em um barramento compartilhado. Quando o CS está ativo (geralmente em nível baixo), o dispositivo selecionado está habilitado para comunicação, enquanto os outros dispositivos permanecem inativos. Isso é crucial em sistemas onde múltiplos dispositivos compartilham a mesma linha de dados.

## 2.5 Microcontroladores

Microcontroladores são pequenos computadores em um único chip que contêm um processador, memória e periféricos de entrada/saída. Eles são amplamente utilizados em sistemas embarcados para controlar dispositivos eletrônicos, executar tarefas específicas e interagir com o ambiente.

# 3 Metodologia

## 3.1 Materiais Utilizados

- 2x Placas Arduino Uno R3
- 2x LEDs
- 2x Resistores 330Ω
- 2x Botões (Push Buttons)
- Fios de conexão (Jumpers)
- Protoboard (opcional)

## 3.2 Diagrama de Conexões

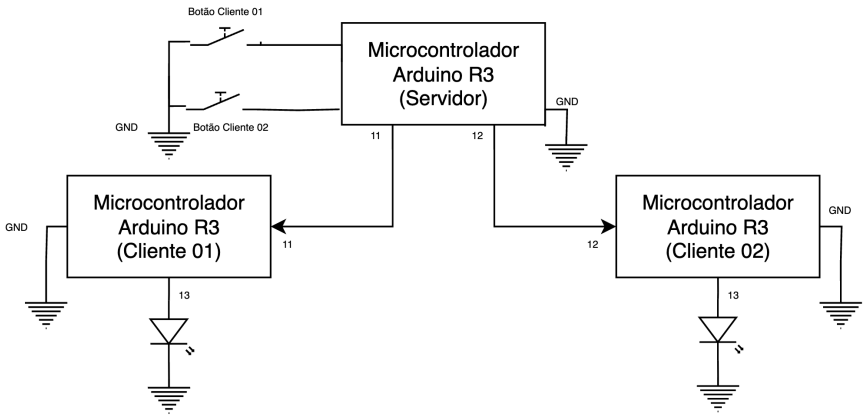
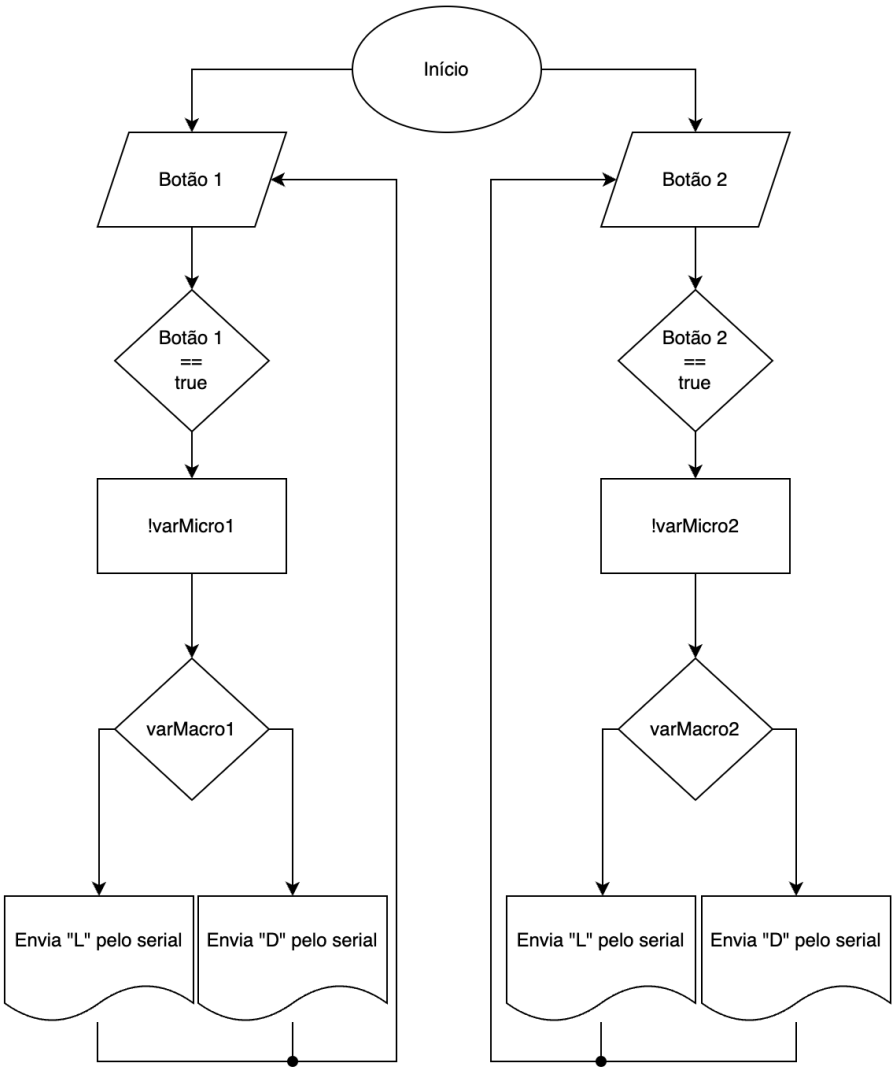



Diagrama de conexões entre os microcontroladores, LEDs e botões.

## 3.3 Configuração dos Microcontroladores

### 3.3.1 Microcontrolador Servidor

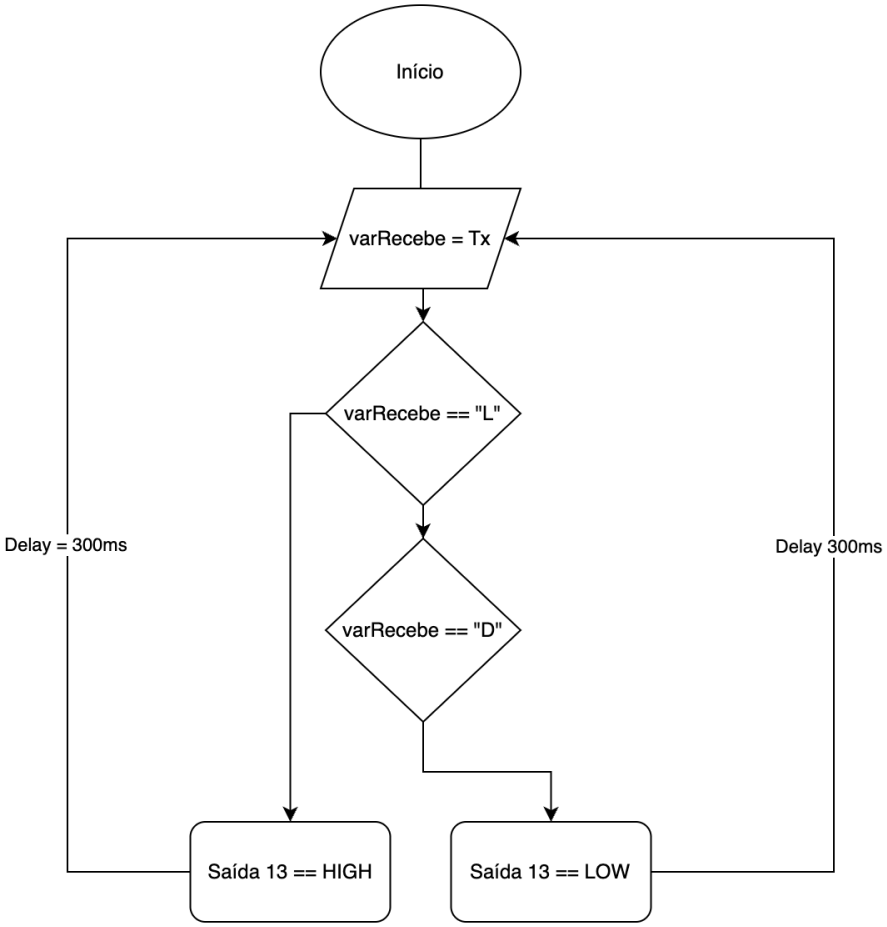
O microcontrolador servidor é responsável por receber os comandos dos microcontroladores clientes e repassá-los para os dispositivos corretos. Ele utiliza a comunicação serial para interagir com os clientes e controla os GPIOs para selecionar o dispositivo correto através do sinal Chip Select (CS).




Fluxograma do microcontrolador servidor.

### 3.3.2 Microcontroladores Clientes

Os microcontroladores clientes são responsáveis por monitorar os botões físicos e enviar comandos ao microcontrolador servidor quando um botão é pressionado. Cada cliente está associado a um LED específico, que será acionado ou desligado com base nos comandos recebidos do servidor.




Fluxograma do microcontrolador cliente.

## 3.4 Implementação do Código

### 3.4.1 Código do Microcontrolador Servidor

**Código do Microcontrolador Servidor.**

```
// C++ code

bool varMicro01 = false;
bool varMicro02 = false;
void setup() {
  pinMode(8, INPUT);
  pinMode(9, INPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if (digitalRead(8) == LOW) {
    varMicro01 = !varMicro01;
    digitalWrite(11, HIGH);
    if (varMicro01) {
```

```
      Serial.println("L");
    } else {
      Serial.println("D");
    }
    delay(300);
    digitalWrite(11, LOW);
  }
  if (digitalRead(9) == LOW) {
    varMicro02 = !varMicro02;
    digitalWrite(12, HIGH);
    if (varMicro02) {
      Serial.println("L");
    } else {
      Serial.println("D");
    }
    delay(300);
    digitalWrite(12, LOW);
  }
}
```

### 3.4.2 Código do Microcontrolador Cliente

**Código do Microcontrolador Cliente.**

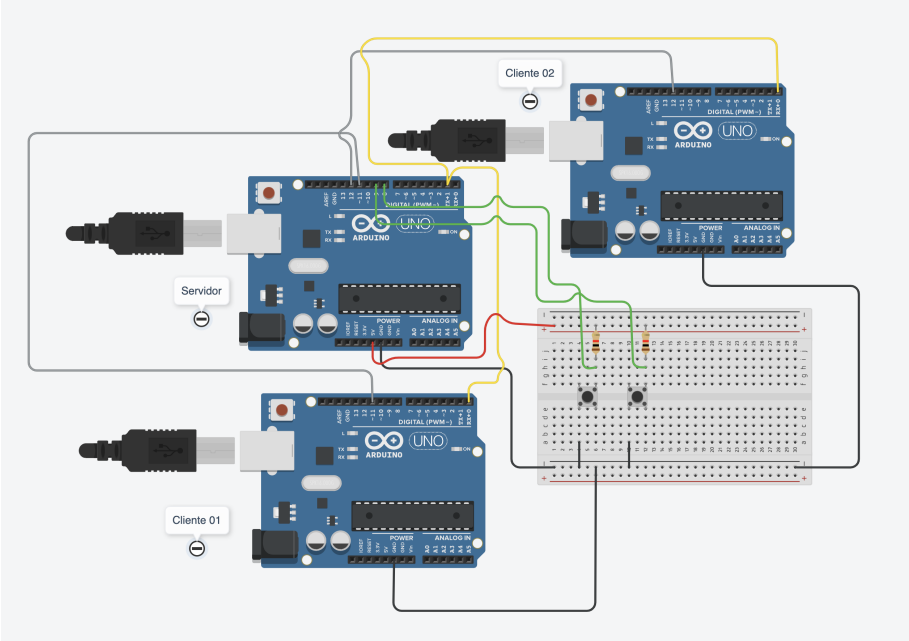
```
// C++ code
char varRecebe;

void setup() {
  pinMode(13, OUTPUT);
  pinMode(11, INPUT);
  Serial.begin(9600);
}

void loop() {
  varRecebe = Serial.read();
  Serial.println(varRecebe);
  if (digitalRead(11) == HIGH) {
    if (varRecebe == 'L') {
      digitalWrite(13, HIGH);
    } else if (varRecebe == 'D') {
      digitalWrite(13, LOW);
    }
  }
  delay(300);
}
```

# 4 Resultados

Através da implementação do sistema de comunicação serial entre os microcontroladores, foi possível observar o funcionamento correto da arquitetura Cliente-Servidor. Os microcontroladore servidor foram capazes de enviar comandos aos clientes quando os botões foram pressionados preservando seus estados tornando o efeito 'toggle', e o servidor respondeu adequadamente acionando ou desligando os LEDs correspondentes.




Montagem final do sistema com os microcontroladores, LEDs e botões no simulador TinkerCAD.

# 5 Conclusão

A implementação de um sistema de comunicação serial entre microcontroladores utilizando a arquitetura Cliente-Servidor demonstrou a eficácia dessa abordagem para o controle e monitoramento de dispositivos em sistemas embarcados. A utilização de GPIOs como Chip Select permitiu a seleção precisa dos dispositivos, garantindo que os comandos fossem direcionados corretamente. A integração de botões físicos para acionar os LEDs mostrou como a interação do usuário pode ser facilmente incorporada em sistemas embarcados. Este trabalho reforça a importância de compreender os conceitos de comunicação serial e arquitetura Cliente-Servidor para o desenvolvimento de soluções eficientes e escaláveis em aplicações de engenharia de controle e automação e engenharia da computação.

---

### 5.0.1 Link do TinkerCAD

https://www.tinkercad.com/things/5rhsM6DAYVq-brilliant-luulia-elzing/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard&sharecode=Q1B5XRHh_h2rFmrkxMFGBgP1fBbbUQ7nosW_mD-sV9Q