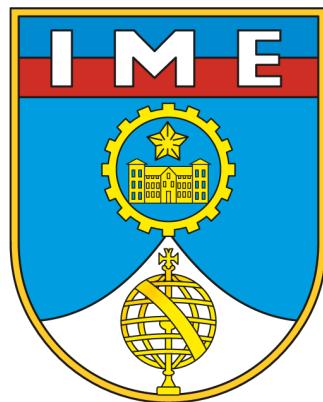


**MINISTÉRIO DA DEFESA  
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA  
INSTITUTO MILITAR DE ENGENHARIA**  
(REAL ACADEMIA DE ARTILHARIA, FORTIFICAÇÃO E DESENHO, 1792)



**SEÇÃO DE ENGENHARIA ELÉTRICA (SE/3)**

**PROGRAMAÇÃO APLICADA**

**RELATÓRIO FINAL**

**PROFESSOR: TEN NÍCOLAS OLIVEIRA**

**COMPONENTES DO GRUPO:**  
GABRIEL AIRES LIMA  
VICTOR KAUÃ DE SOUSA VIANA

# Contents

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Fluxograma</b>	<b>4</b>
<b>3</b>	<b>Diagrama de Classes</b>	<b>5</b>
<b>4</b>	<b>Instruções de Compilação e Uso</b>	<b>6</b>
4.1	Clonagem do Projeto . . . . .	6
4.2	Instalação da Toolchain ARM . . . . .	6
4.3	Compilação do Programa . . . . .	6
4.4	Transferência e Execução na Placa . . . . .	6
<b>5</b>	<b>Documentação do Projeto e do Protocolo</b>	<b>7</b>
5.1	Estrutura Geral . . . . .	7
5.2	Protocolo de Comunicação UDP . . . . .	7
5.3	Funcionamento do Cliente UDP . . . . .	7
5.4	Funcionamento do Servidor e Interface . . . . .	7
<b>6</b>	<b>Descrição do Sensor e Funcionamento</b>	<b>8</b>
6.1	Componentes do Módulo . . . . .	8
6.2	Princípio de Operação . . . . .	9
<b>7</b>	<b>Análise dos Valores Medidos</b>	<b>10</b>
<b>8</b>	<b>Conclusão</b>	<b>10</b>

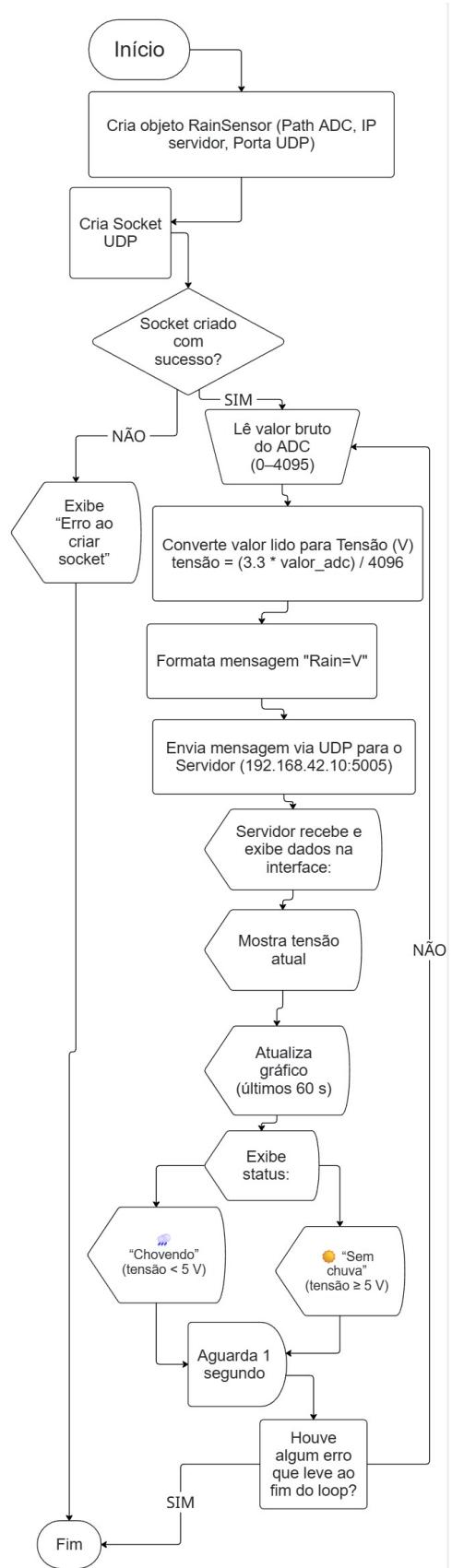
# 1 Introdução

Este relatório apresenta o desenvolvimento de um sistema embarcado voltado ao **monitoramento inteligente de cargas sensíveis**, com ênfase na **deteção de umidade e infiltração de líquidos**. O projeto foi desenvolvido no âmbito da disciplina *Programação Aplicada*, do Instituto Militar de Engenharia (IME).

O sistema utiliza o **Sensor de Chuva SS29** conectado à placa **STM32MP1-DK1**, integrando hardware e software para leitura, processamento e envio de dados via protocolo **UDP** a um servidor central. No computador, uma **interface gráfica em Python** exibe as leituras em tempo real, com alertas visuais e gráficos históricos.

A proposta do projeto é proporcionar experiência prática na comunicação cliente-servidor e no tratamento de dados provenientes de sensores analógicos/digitais, explorando simultaneamente conceitos de programação embarcada, redes e visualização de dados.

## 2 Fluxograma



### 3 Diagrama de Classes

#### Atributos:

- `std::string adc_path`: caminho do canal ADC no sistema Linux embarcado (exemplo: `/sys/bus/iio/devices/iio:device0/in_voltage13_raw`).
- `std::string dest_ip`: endereço IP do servidor remoto para envio UDP.
- `int dest_port`: número da porta UDP de destino.
- `int max_envios`: quantidade máxima de transmissões.
- `int intervalo_seg`: intervalo entre cada envio (em segundos).

#### Funções principais:

`int lerSensor()` : Lê o valor bruto do ADC (inteiro entre 0 e 4096) a partir do arquivo indicado em `adc_path`.

**Retorno:** valor lido.

`float converterTensao(int valor_adc)` : Converte a leitura do ADC em tensão (em volts) usando a relação:

$$\text{tensão} = (3.3 * \text{valor\_adc}) / 4096$$

**Retorno:** valor convertido (em volts).

`std::string gerarMensagem(float tensao)` : Formata a mensagem a ser enviada via UDP no padrão:

"Rain=<tensão>V"

`bool enviarUDP(const std::string& mensagem)` : Cria o socket UDP e envia a string formatada para o IP e porta definidos.

**Retorno:** `true` em caso de sucesso, `false` se ocorrer falha no envio.

`void executar()` : Executa o loop principal de leitura e envio:

1. Lê o valor do ADC.
2. Converte para tensão.
3. Gera a mensagem.
4. Envia via UDP.
5. Aguarda `intervalo_seg` segundos entre os envios.

`int main()` : Instancia a classe `RainSensor`, inicializa os parâmetros e chama o método `executar()`.

## 4 Instruções de Compilação e Uso

### 4.1 Clonagem do Projeto

```
git clone https://github.com/seu-usuario/Sensor-de-chuva.git  
cd Sensor-de-chuva
```

### 4.2 Instalação da Toolchain ARM

Baxe e extraia o arquivo da toolchain para compilação cruzada da STM32MP1-DK1:

```
tar -xvf arm-buildroot-linux-gnueabihf_sdk-buildroot.tar.gz
```

### 4.3 Compilação do Programa

```
arm-linux-gnueabihf-g++ RainSensor.cpp -o RainSensor -std=c++17
```

### 4.4 Transferência e Execução na Placa

Envie o executável para a placa via scp:

```
scp RainSensor root@192.168.42.2:/home/root
```

Acesse a placa via SSH e execute:

```
ssh root@192.168.42.2  
chmod +x RainSensor  
.RainSensor
```

O programa exibirá leituras contínuas da tensão do sensor.

```
# chmod +x sensor  
# ./sensor  
Sent: Rain=14.691284V  
# ./sensor  
Sent: Rain=10.579980V  
# ./sensor  
Sent: Rain=43.751587V  
# ./sensor  
Sent: Rain=47.216747V  
# ./sensor  
Sent: Rain=42.105614V  
# ./sensor  
Sent: Rain=29.980371V
```

## 5 Documentação do Projeto e do Protocolo

### 5.1 Estrutura Geral

O projeto é composto por dois módulos principais:

- **RainSensor.cpp**: implementa a leitura do ADC, conversão de valor e envio via UDP;
- **Interface.py**: interface de visualização e registro de dados.

A classe **RainSensor** possui métodos dedicados à leitura analógica, formatação da mensagem e transmissão de pacotes UDP. Já a interface Python recebe as mensagens, exibe os dados em tempo real e permite o salvamento de logs.

### 5.2 Protocolo de Comunicação UDP

O protocolo escolhido foi o **UDP (User Datagram Protocol)**, devido à sua baixa latência e simplicidade. A Tabela 1 apresenta os principais parâmetros utilizados.

Parâmetro	Valor
IP do servidor	192.168.42.10
IP da placa STM32MP1-DK1	192.168.42.2
Porta UDP	5005
Frequência de envio	1 leitura/segundo
Formato da mensagem	"Rain=<tensão>V"

Table 1: Parâmetros de comunicação via UDP

A cada segundo, o cliente (na STM32MP1) lê o valor analógico do sensor, converte para tensão e envia ao servidor uma string contendo o valor atual.

### 5.3 Funcionamento do Cliente UDP

1. Cria um socket UDP (**SOCK\_DGRAM**);
2. Lê o ADC e converte em tensão;
3. Formata a mensagem no padrão “Rain= $x$ V”;
4. Envia para o IP e porta definidos;
5. O servidor (PC) recebe e exibe em tempo real.

### 5.4 Funcionamento do Servidor e Interface

O servidor é implementado em Python e possui uma interface gráfica que:

- Atualiza o valor da tensão a cada 1 s;
- Exibe o histórico dos últimos 60 s de leitura;
- Muda de cor conforme o estado (“Chovendo” ou “Sem chuva”);

- Permite salvar os dados registrados em um arquivo .csv.

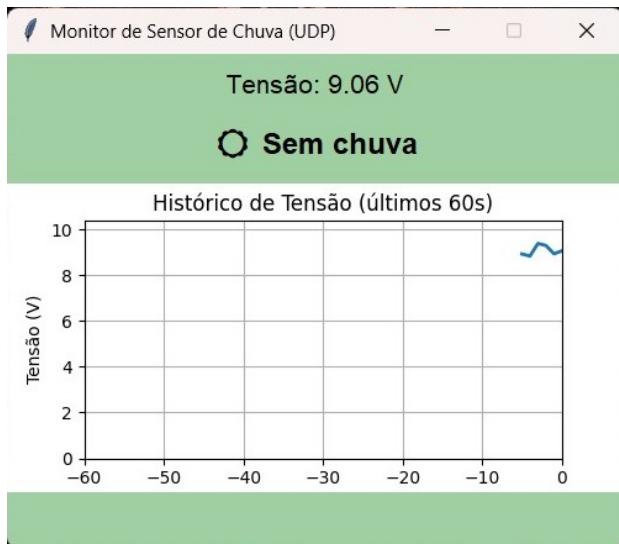


Figure 1: Condição de tempo seco



Figure 2: Condição de chuva detectada

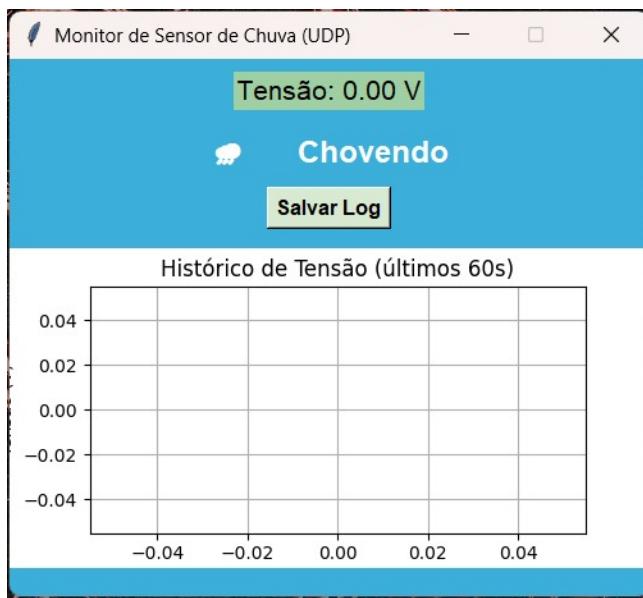


Figure 3: Implementação do salvamento do histórico em arquivos de log

## 6 Descrição do Sensor e Funcionamento

O sensor de chuva utilizado é o **Módulo SS29**, projetado para detectar presença de água ou alta umidade sobre sua superfície. O módulo possui saídas analógica e digital, sendo a primeira conectada ao conversor ADC da placa STM32MP1.

### 6.1 Componentes do Módulo

- **Matriz condutiva** – detecta gotas de água ou umidade;

- **Comparador LM393** – gera saída digital conforme limiar ajustado;
- **Potenciômetro** – permite ajustar a sensibilidade de detecção.

## 6.2 Princípio de Operação

- Ambiente seco  $\Rightarrow$  alta resistência  $\Rightarrow$  alta tensão lida (acima de 5 V);
- Ambiente úmido ou molhado  $\Rightarrow$  baixa resistência  $\Rightarrow$  baixa tensão lida.

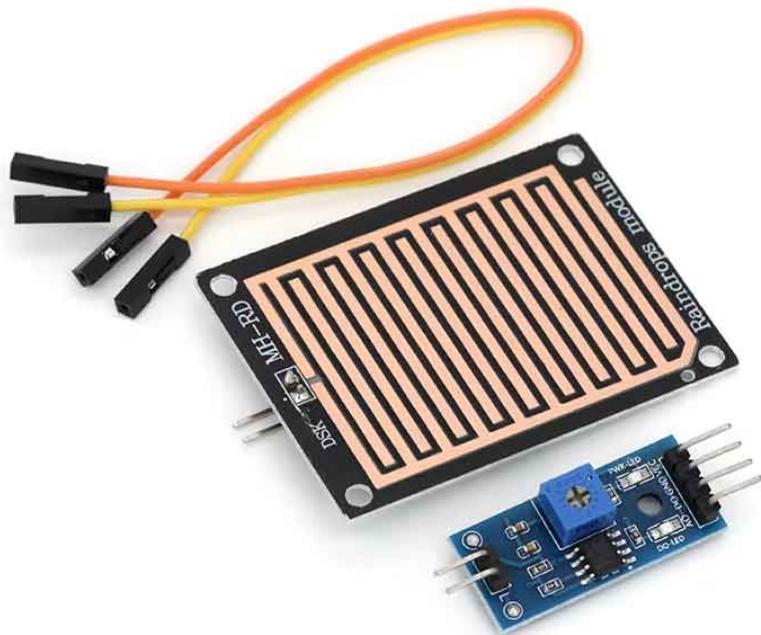


Figure 4: Sensor de Chuva SS29

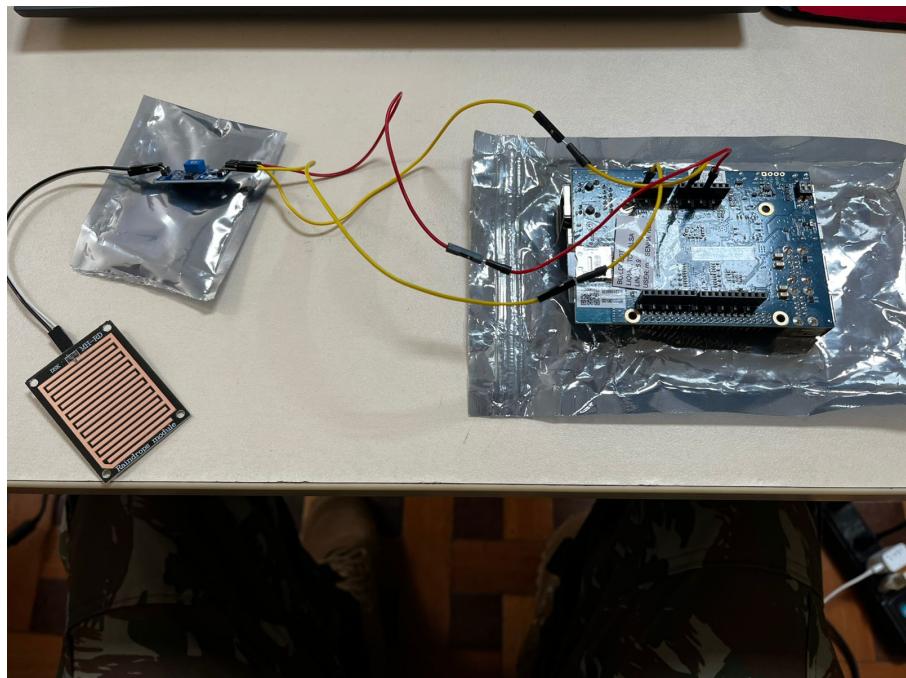


Figure 5: onfiguração de conexão entre placa STM32 e sensor de chuva.

## 7 Análise dos Valores Medidos

A tensão medida pelo sensor é representada em tempo real na interface. Valores acima de **5 V** indicam ausência de chuva (“Sem chuva”), enquanto tensões abaixo desse limiar são interpretadas como “**Chovendo**”.

O gráfico exibe os últimos 60 segundos de leituras, atualizando dinamicamente e permitindo a exportação dos dados para arquivo `.csv`. Essa funcionalidade garante rastreabilidade e análise posterior das condições ambientais.

## 8 Conclusão

O projeto demonstrou a integração bem-sucedida entre hardware e software em um ambiente embarcado. A leitura do sensor SS29, o envio dos dados via UDP e a interface de monitoramento em Python permitiram o acompanhamento preciso das condições de umidade em tempo real.

Além disso, o uso de ferramentas como **Doxxygen** e **Graphviz** reforçou a importância da documentação automatizada e da boa organização de código em sistemas de engenharia.

O sistema desenvolvido cumpre os requisitos propostos e pode ser expandido futuramente para monitorar múltiplos sensores ou enviar alertas automáticos ao detectar níveis críticos de umidade.