Roteiro Nº 07	**Acesso Remoto a Dados de Sensores com ESP32 e Ngrok** ***(Monitoramento de vazão com Servidor Web Local e Túnel Seguro)***	
Curso Engenharia Elétrica	**Disciplina** ELT 1119 – Redes e Aplicações IoT	**Professor** Carlos Alberto Vasconcelos Bezerra
Nome do Estudante		**Data**

1. Objetivos da Aprendizagem

- Operar a Estação de Fluxo LabVolt Modelo 3502 com controle remoto via rede.
- Utilizar um servidor web embarcado no ESP32 para controlar uma variável de processo (vazão).
- Integrar a ferramenta ngrok para acesso remoto à interface do ESP32.
- Compreender a relação entre a saída DAC, a frequência da bomba e a vazão observada.
- Explorar os conceitos de IoT, tunelamento seguro e controle de processo..

2. Material Necessário

- Placa de desenvolvimento ESP32-DevKitC-V4;
- Computador com ambiente de desenvolvimento Thonny instalado e ngrok instalado e autenticado;
- Celular com modo roteador Wi-Fi;
- Cabo USB para comunicação entre o ESP32 e o computador;
- Planta de vazão didática modelo 3502 Lab Volt pertencente ao laboratório de instrumentação industrial da PUC Goiás;

3. Fundamentação Teórica

A planta de fluxo utilizada é a Estação de Vazão LabVolt Modelo 3502, representada na figura1 abaixo.

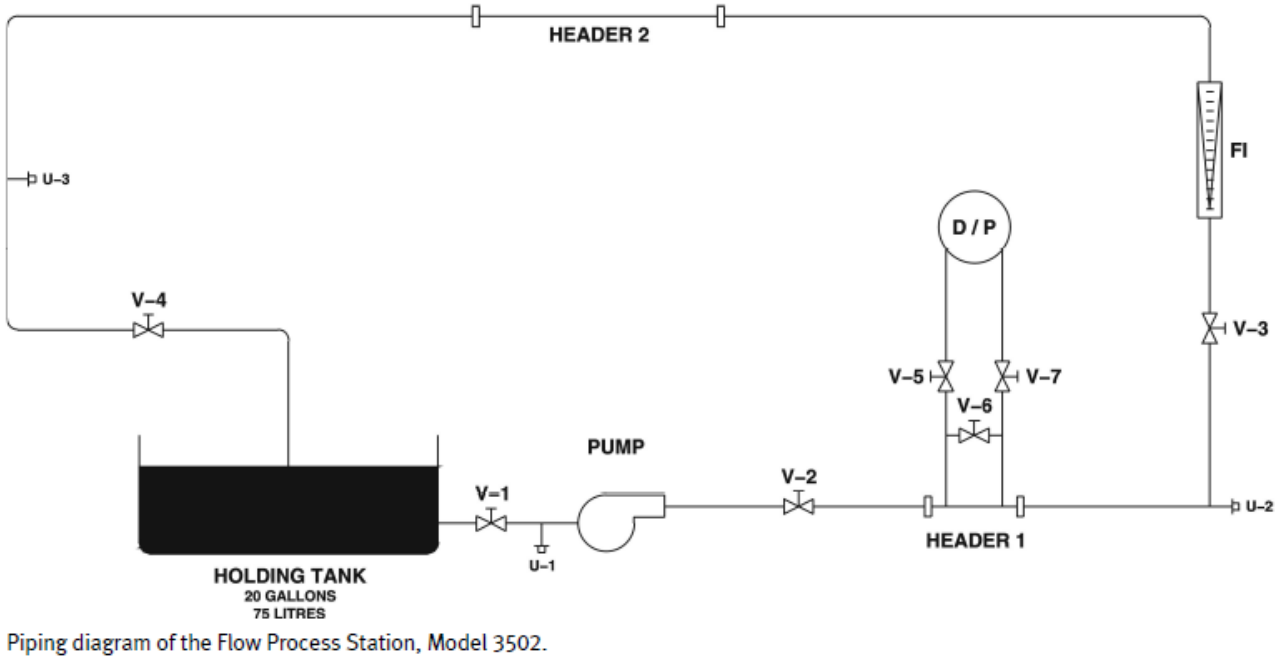


Figura 1 – Fluxograma do sistema de vazão

Neste experimento, o controle da planta será feito por uma interface web hospedada no ESP32. O valor ajustado pelo usuário (via slider) será convertido em sinal analógico através do DAC, que é conectado ao terminal de entrada analógica do inversor de frequência.
A página web exibirá em tempo real:

- O valor do DAC (0 a 255)
- A frequência aplicada (0 a 60 Hz)
- A vazão estimada (0 a 38 LPM)

O ngrok será utilizado para tornar essa interface acessível a partir de qualquer lugar com internet, por meio de uma URL pública segura.

4. Procedimento Experimental

4.1 Conexão do Sistema:

1. Conecte o ESP32 à rede Wi-Fi gerada pelo celular (modo roteador).
2. Programe o ESP32 com o código de controle via DAC e servidor web. (Anexo 1)

4.2 Configuração do ngrok:
No notebook conectado à mesma rede do ESP32:

- Abra o terminal (cmd).
- Execute:

ngrok http 192.168.x.x:80
(Substitua x.x pelo IP do ESP32 mostrado no terminal do Thonny ou PuTTY).

4.3 Copie a URL gerada pelo ngrok (ex: https://abc123.ngrok.io).

4.4 Controle da Vazão:

- Acesse a interface:
  - Via IP local nos dispositivos da rede (ex: http://192.168.x.x)
  - Via link do ngrok de qualquer lugar com internet.

4.5 Use o **slider** para alterar o valor do DAC (0 a 255).

4.6 Observe a resposta da planta:
O medidor de vazão (FI)
A estabilidade do nível no tanque
O comportamento dos indicadores na interface web

5 Exercícios Propostos:

1. **Confiabilidade da Interface Remota:**
   - A resposta da planta é compatível com os valores exibidos na interface remota?
   - Há atrasos perceptíveis na comunicação?
2. **Segurança e Aplicações Reais:**
   - Quais seriam os riscos e cuidados necessários ao controlar uma planta real via web?
   - Em quais cenários industriais o uso do ngrok (ou outro tunelamento seguro) seria vantajoso?
3. **Modelagem da Planta:**
   - A relação entre DAC e vazão é linear? Como ela pode ser representada graficamente?
   - Como a variação da frequência influencia a dinâmica do processo?

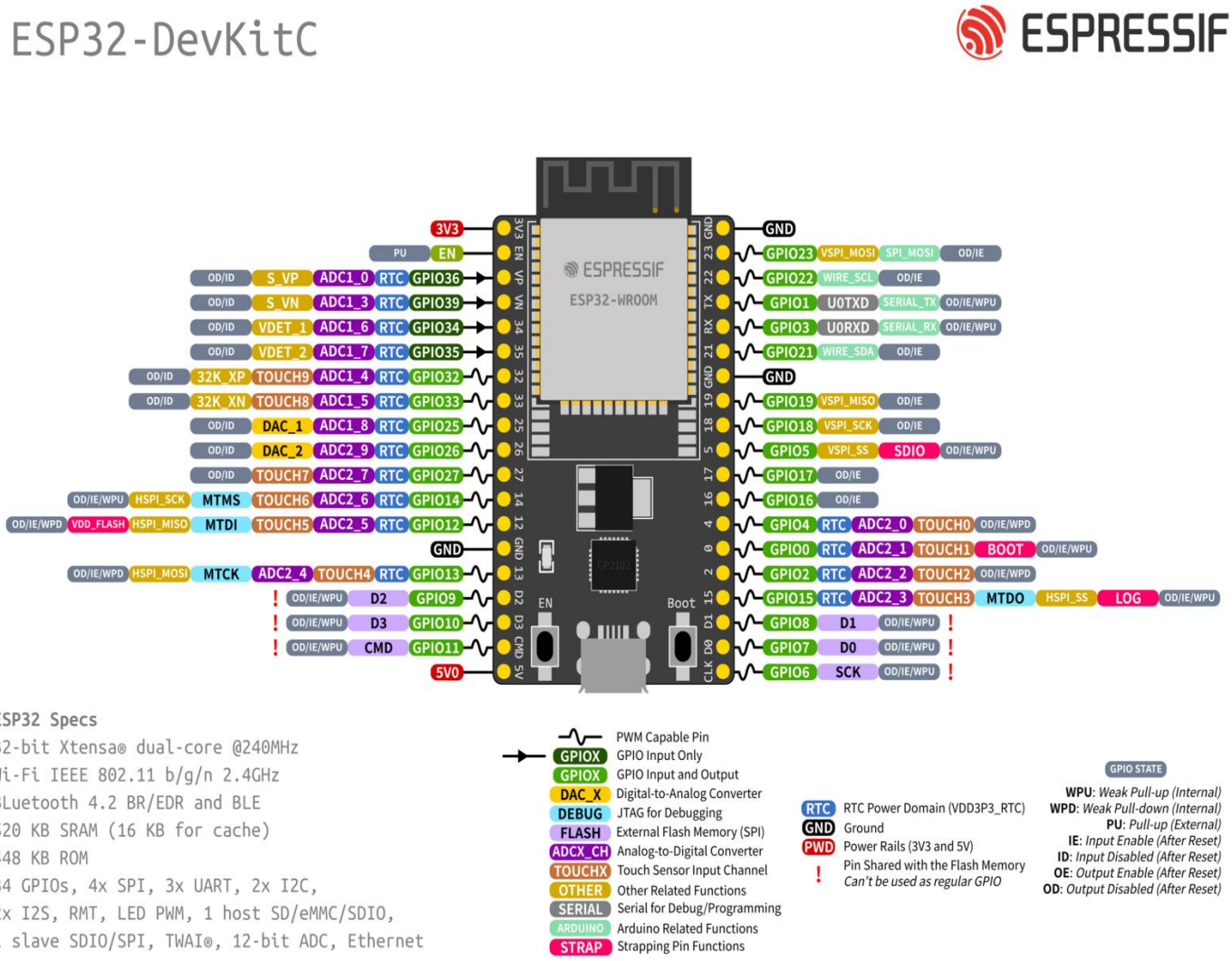
Bibliografia

ASCHER, D.; LUTZ, M. Aprendendo Python. Porto Alegre: Bookman, 2007.

MENEZES, N. N. C. Introdução à programação com Python. São Paulo: Ed. Novatec, 2014.

OLIVEIRA, S. Internet das coisas com ESP8266, Arduino e Raspberry Pi. São Paulo: Ed. Novatec, 2017.

OLIVEIRA, C. L. V. IoT com Micropython e NodeMCU. São Paulo: Ed. Novatec, 2022.



Anexo 1 – Código micropython

```
import network
import socket
import ujson
from machine import Pin, DAC

# === Configurações do Wi-Fi ===
SSID = "Alencar Oi 2.4G"
PASSWORD = "Carloshenrique"

# === Inicializa DAC no pino 25 ===
dac = DAC(Pin(25))
output_value = 0

# === Inicializa Wi-Fi ===
sta = network.WLAN(network.STA_IF)
sta.active(True)
sta.connect(SSID, PASSWORD)

while not sta.isconnected():
    pass

print("Conectado à rede!")
print("Endereço IP:", sta.ifconfig()[0])

# === Função HTML inicial ===
def html_page():
    html = """<!DOCTYPE html>
<html>
<head>
    <meta charset='utf-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <meta name='ngrok-skip-browser-warning' content='true'>
    <title>Controle de Frequência e Vazão</title>
    <link rel="icon" href="data:,">
    <style>
        .container {
            width: 40%;
            margin: auto;
            padding: 20px;
            border: 2px solid #000;
            border-radius: 10px;
            background-color: #e0f7fa;
            text-align: center;
            font-family: Arial, sans-serif;
        }
        input[type=range] {
            width: 100%;
```

```
        }
        p {
            font-size: 1.2em;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Controle do Inversor</h2>
        <input type="range" id="dacSlider" min="0" max="255" value="0" oninput="sendDAC(this.value)">
        <p>Valor DAC: <span id="dacVal">0</span></p>
        <p>Frequência: <strong><span id="freq">0.0</span> Hz</strong></p>
        <p>Vazão: <strong><span id="vazao">0.0</span> LPM</strong></p>
    </div>
    <script>
        function sendDAC(val) {
            document.getElementById('dacVal').innerText = val;
            fetch('/update?dac=' + val)
            .then(response => response.json())
            .then(data => {
                document.getElementById('freq').innerText = data.freq.toFixed(1);
                document.getElementById('vazao').innerText = data.vazao.toFixed(1);
            });
        }
    </script>
</body>
</html>"""
    return html

# === Criação do servidor ===
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(('', 80))
s.listen(1)
print("Servidor iniciado!")

while True:
    try:
        conn, addr = s.accept()
        request = conn.recv(1024).decode("utf-8")
        print("Requisição:", request)

        if "/favicon.ico" in request:
            conn.close()
            continue

        if "GET /update?dac=" in request:
            try:
                param = request.split("GET /update?dac=")[1].split(" ")[0]
                output_value = int(param)
                output_value = max(0, min(255, output_value))
```

```
            dac.write(output_value)
            print("DAC atualizado para:", output_value)
        except:
            print("Erro ao interpretar valor do DAC.")

        freq = (output_value / 255) * 60
        vazao = (output_value / 255) * 38
        data = {"freq": freq, "vazao": vazao}

        response = "HTTP/1.1 200 OK\r\nContent-Type: application/json\r\nConnection: close\r\n\r\n"
        response += ujson.dumps(data)
        conn.sendall(response.encode("utf-8"))
        conn.close()
        continue

    # Página HTML inicial
    response = (
        "HTTP/1.1 200 OK\r\n"
        "Content-Type: text/html\r\n"
        "Connection: close\r\n"
        "\r\n"
        + html_page()
    )
    conn.sendall(response.encode("utf-8"))
    conn.close()

except Exception as e:
    print("Erro:", e)
    conn.close()
```