**Nome:** Darlana Grando;

**Curso:** Técnico de Desenvolvimento de Sistemas;

**Professor:** Kevin S Guimaraes.

# **Trabalho de Pesquisa — Modularização PWM**

## **Tema: Modulador PWM (Pulse Width Modulation)**

**1. O que é o Modulador PWM**

**PWM** significa **Modulação por Largura de Pulso**.

É uma técnica para **controlar a energia enviada** a um dispositivo eletrônico (motor, LED, etc.) usando pulsos rápidos.

**Duty Cycle (ou Ciclo de Trabalho)**

É a **porcentagem de tempo que o sinal fica ligado (em nível alto)** dentro de um ciclo completo.

**Exemplo:**

* 25% = ligado 25% do tempo, desligado 75%
* 50% = ligado metade do tempo
* 75% = ligado 75% do tempo

**Diferença entre PWM e sinal analógico contínuo.**

* PWM liga e desliga rapidamente, parecendo um sinal digital.
* O analógico muda suavemente, sem interrupções.
* O PWM simula um sinal analógico **sem realmente ser contínuo**, o que é mais eficiente.

**2. Como o PWM funciona**

\* O PWM **controla a média da tensão** enviada para uma carga (como um LED ou motor).

Quanto maior o duty cycle, maior a tensão média.

**Frequência do sinal:**

* Frequência é **quantas vezes por segundo o pulso acontece**.
* Frequências mais altas fazem a carga (motor/LED) responder de forma **mais suave**, sem piscar ou vibrar.

**3. Para que serve o PWM**

**Aplicações:**

**Controle de motores DC:**

PWM regula a velocidade do motor ajustando a energia média enviada.

**Controle de brilho de LEDs**:

Quanto maior o duty cycle, mais tempo o LED fica aceso → mais brilho.

**Controle de temperatura/potência:**

PWM pode controlar a energia em resistências, aquecedores, etc.

**Robótica e IoT:**

Muito usado para controlar atuadores, servos, motores e iluminação com economia de energia.

**4. Importância do PWM**

**Mais eficiente que sinal analógico puro:**

Porque liga/desliga rapidamente, evitando perda de energia em forma de calor.

**Vantagem do Duty Cycle:**

Permite controlar potência com precisão, sem precisar de componentes caros.

**Importância no consumo e controle:**

* Menor consumo de energia.
* Mais controle e precisão nos dispositivos.
* Ideal para baterias e dispositivos IoT.

**5. Aplicação prática no Arduino**

**O Arduino gera PWM** com a função `analogWrite(pino, valor)`

(valor vai de 0 a 255, onde 255 = 100% ligado)

**Pinos com suporte PWM**.

No Arduino Uno, os pinos marcados com `~` (como 3, 5, 6, 9, 10, 11)

**Exemplo:** Controle de brilho do LED

cpp

int led = 9;

void setup() {

pinMode(led, OUTPUT);

}

void loop() {

for (int brilho = 0; brilho <= 255; brilho++) {

analogWrite(led, brilho);

delay(10);

}

for (int brilho = 255; brilho >= 0; brilho--) {

analogWrite(led, brilho);

delay(10);

}

}

**6. Interferência do PWM no ADC**

**ADC (Conversor Analógico-Digital)**:

Converte sinais analógicos (como tensão de um sensor) para valores digitais que o Arduino entende.

**Problema**:

Sinais PWM podem **gerar ruído** nas leituras analógicas, se estiverem muito próximos ou se usarem a mesma alimentação.

**Soluções**:

* Usar **filtro RC (resistor + capacitor)** para "suavizar" o sinal PWM.
* Separar alimentação do ADC e do PWM.
* Medir sinal analógico em momentos em que o PWM está desligado.

**Por que isso importa em IoT**:

Em projetos com sensores e atuadores juntos, interferência do PWM pode causar **leituras erradas**, o que afeta decisões do sistema.