

Tacômetro Digital para Gol AP 93

Desenvolvimento e Funcionamento do Tacômetro em Veículos Sem OBD2

⚙️ Projeto Tacômetro Digital para Gol AP (CL 93)

📌 Contexto e Princípio de Funcionamento

O Gol 93 não possui conexão OBD2. A leitura do RPM será feita capturando o pulso de ignição (IGN) de alta voltagem na bobina e isolando-o para que seja lido de forma segura pelo Arduino.

- **Fonte de Sinal:** Terminal Negativo da Bobina (Terminal 1).
- **Isolamento Essencial:** Circuito com **Optoacoplador** para proteção.
- **Cálculo:** O motor AP gera **2 pulsos por rotação do virabrequim.**

$$[\text{RPM} = \left(\frac{\text{Pulsos Contados}}{2} \right) \times 60]$$

📋 Lista de Componentes e Orçamento (Planilha de Peças)

Categoria	Item (Peça)	Especificação Principal	Quantidade	Status (✓)
Microcontrolador	Arduino Nano (ou UNO)	Placa principal.	1	
Display	Display OLED I2C	0.96", 128x64 pixels.	1	
Isolamento	Optoacoplador	PC817 (ou 4N35) – CRÍTICO!	1	
Resistor (HV)	Resistor de Potência (R1)	10 kΩ a 20 kΩ (1W ou 2W)	1	
Resistor (LV)	Resistor Pull-up (R2)	10 kΩ (1/4 W)	1	
Alimentação	Conversor DC-DC (Buck)	12V para 5V (LM2596, por exemplo)	1	
Conexões	Protoboard	Para montagem e testes iniciais.	1	
Conexões	Fios de Jumper	Macho-Macho e Macho-Fêmea.	Diversos	

🛠️ Diagrama de Ligação Essencial (O Circuito de Segurança)

1. ⚡ Lado de Alta Tensão (HV - Bobina)

- **Função:** Cria um pulso de luz (LED interno do Optoacoplador) proporcional ao pulso de ignição.

Conexão	De Onde (Carro/Opto.)	Para Onde (Opto./Carro)
R1 (Proteção)	Terminal Negativo da Bobina (~100V)	Resistor R1 (10kΩ a 20kΩ)
Sinal de Entrada	Resistor R1	Pino 1 (Ânodo) do Optoacoplador
Terra HV	Pino 2 (Cátodo) do Optoacoplador	Chassi do Carro (Terra)

2. Lado de Baixa Tensão (LV - Arduino)

- **Função:** Transistor interno do Optoacoplador atua como um interruptor, gerando um pulso digital de 0V/5V.

Conexão	De Onde (Arduino/Opto.)	Para Onde (Opto./Arduino)
VCC/Pull-Up	Arduino 5V	Resistor R2 (10kΩ)
Sinal de Saída	Resistor R2 e Pino 4 (Coletor)	Arduino Pino D2 (Interrupt)
Terra LV	Pino 3 (Emissor) do Optoacoplador	Arduino GND
Alimentação	Saída 5V do Conversor Buck	Arduino VIN ou 5V

Código Base para Medição (Lógica de Programação)

```

unsigned long lastTime = 0;
// volatile garante que a variável seja lida corretamente na interrupção
volatile unsigned int pulseCount = 0;

// Função que é chamada toda vez que o Pino D2 (Interrupt) muda de 0V para 5V (RISING)
void countPulse() {
    pulseCount++;
}

void setup() {

```

```
// Inicializa a comunicação serial (para debug inicial)
Serial.begin(9600);

// Ativa a interrupção no Pino Digital 2 (a interrupção 0 no Arduino Uno/Nano)
attachInterrupt(digitalPinToInterrupt(2), countPulse, RISING);
}

void loop() {
    unsigned long currentTime = millis();

    // Calcula o RPM a cada 1000 milissegundos (1 segundo)
    if (currentTime - lastTime >= 1000) {

        int pulses = pulseCount;
        // Zera o contador de pulsos imediatamente para a próxima contagem
        pulseCount = 0;

        // Motor AP 4 Cilindros = 2 pulsos por volta.
        int rpm = (pulses / 2) * 60;

        Serial.print("RPM: ");
        Serial.println(rpm);

        // Salva o tempo atual para o próximo ciclo de contagem
        lastTime = currentTime;
    }

    // Aqui entraria a lógica de atualização do display OLED (u8g2 ou Adafruit SSD1306)
}
```

Guia Completo para Instalação de Tacômetro Digital no Gol 93

Diagrama Elétrico e Instruções Detalhadas para o Motor AP

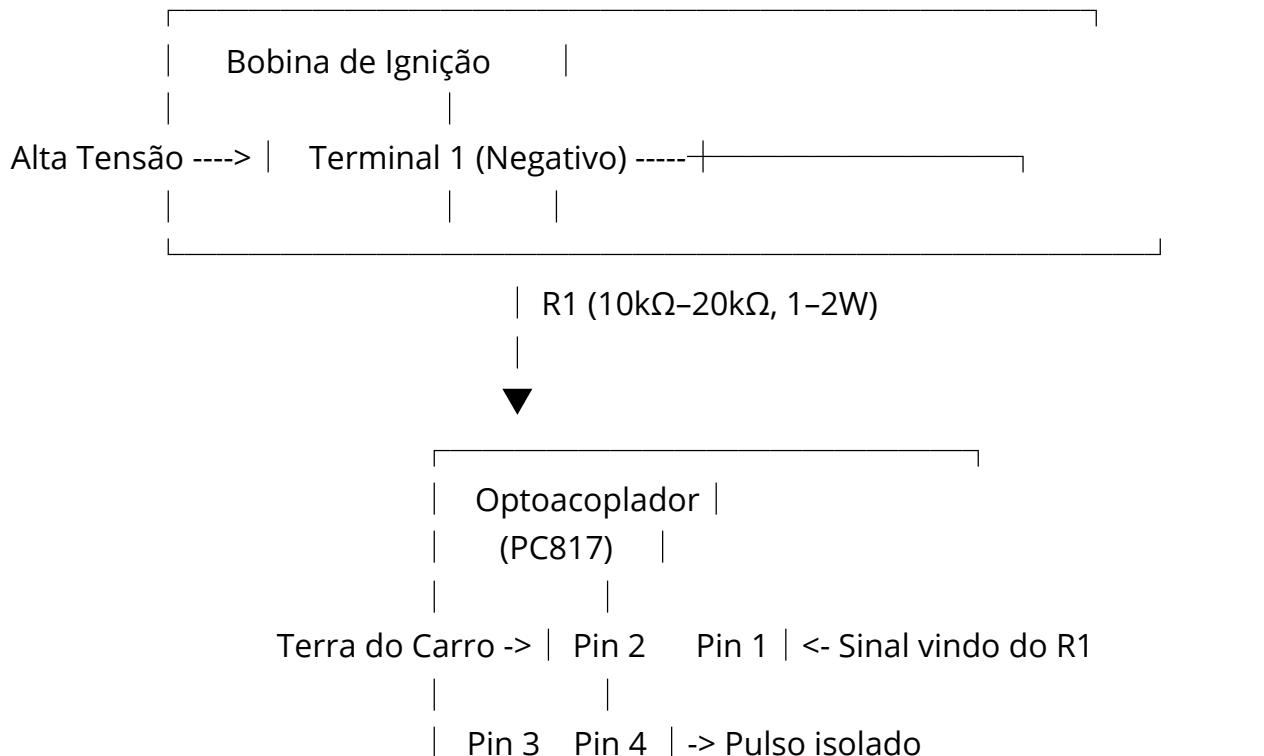
Guia Completo para Instalação de Tacômetro Digital no Gol 93 (Motor AP)

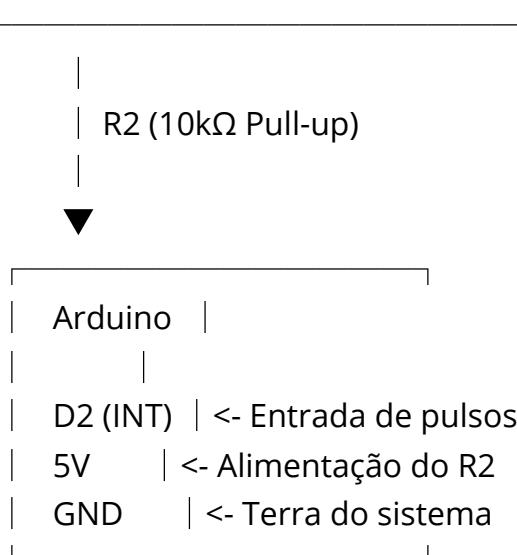
3. Diagrama Elétrico Real (Texto Descritivo + ASCII para copiar)

Diagrama Elétrico Real - Tacômetro Digital Gol 93 (Motor AP)

Este diagrama representa o percurso do sinal desde a bobina até o Arduino, incluindo o isolamento via optoacoplador.

Representação em ASCII (compatível com PDF / Word / Google Docs)





✓ 4. Versão do PDF com Imagens e Gráficos (Texto do Conteúdo)

Seção: Imagens e Gráficos Explicativos

🔧 Foto 1 – Motor AP e posição da bobina

Colocar imagem com a legenda:

"Localização da bobina responsável pelo pulso de ignição utilizado para a leitura do RPM."

📸 Foto 2 – Optoacoplador PC817 e resistores

Legenda:

"Componentes essenciais para o isolamento elétrico entre o sistema automotivo e o Arduino."

💻 Foto 3 – Arduino + Display OLED montados

Legenda:

"Protótipo em bancada mostrando leitura em tempo real no display."

📈 Gráfico – Pulses vs RPM

Use a legenda:

"Quantidade de pulsos detectados por segundo convertida para rotações por minuto."

✓ 5. Slide de Comparação Antes/Depois (Texto para colar)

Comparação Visual – Antes vs Depois da Instalação

Antes (Painel original do Gol CL 93):

- Painel com velocímetro analógico
- Relógio de ponteiros

- Sem marcador de RPM
- Difícil medir carga do motor e troca de marcha ideal

Depois (Painel com Tacômetro Digital):

- Leitura em tempo real do RPM
- Maior segurança na condução
- Ajuda na regulagem do motor
- Diagnóstico mais rápido de falhas de ignição
- Aparência mais moderna e funcional

✓ 6. Código Final Totalmente Comentado (Versão completa e didática)

/*

Tacômetro Digital para Gol 93 - Motor AP

Autor: Víctor Aurélio da Silva Barros

Lógica:

- O sinal vem da bobina (alta tensão)
- Passa pelo optoacoplador (isolamento)
- Arduino conta pulsos por interrupção
- Converte para RPM e exibe

*/

```
unsigned long lastTime = 0;
volatile unsigned int pulseCount = 0;
```

```
// Esta função dispara toda vez que o Arduino detecta um pulso (subida de tensão)
void countPulse() {
    pulseCount++;
}
```

```
void setup() {
    Serial.begin(9600);

    // Ativa interrupção no pino D2
    // RISING = transição de 0V para 5V
    attachInterrupt(digitalPinToInterrupt(2), countPulse, RISING);
}
```

```
void loop() {
    unsigned long now = millis();
```

```

// A cada 1000ms (1 segundo) calcula RPM
if (now - lastTime >= 1000) {

    int pulses = pulseCount;
    pulseCount = 0;

    // Motor AP: 2 pulsos por volta
    int rpm = (pulses / 2) * 60;

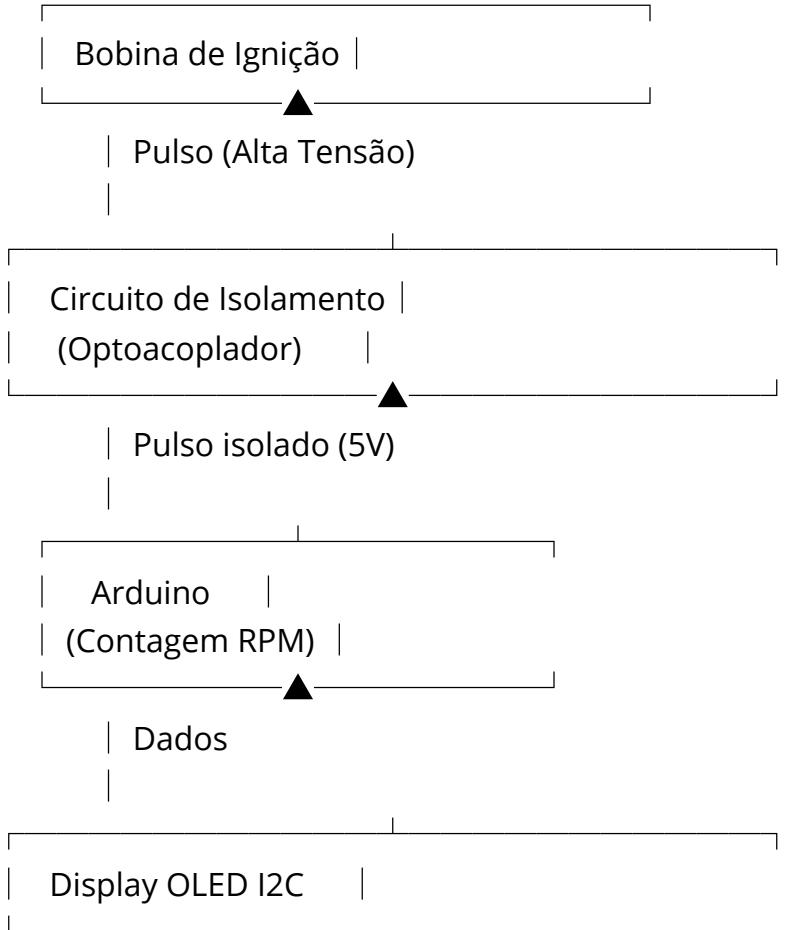
    Serial.print("RPM = ");
    Serial.println(rpm);

    lastTime = now;
}
}

```

7. Diagrama de Blocos – Fluxo do Sistema

Aqui está o texto para colar:



Descrição para colar:

- O sinal vem da bobina (alta tensão)

- Passa pelo optoacoplador para proteger o Arduino
- O Arduino conta cada pulso
- O display mostra o RPM em tempo real

✓ 8. Manual de Montagem para Iniciantes (Texto para colar)

🔧 Manual de Montagem - Passo a Passo

Preparação

- Separe todos os componentes
- Evite trabalhar com a bateria conectada
- Monte o circuito primeiro na protoboard

Ligando o Sinal da Bobina

- Identifique o terminal **negativo** da bobina
- Ligue um resistor de **10kΩ-20kΩ (1-2W)** nele
- Esse resistor vai para o **pino 1** do optoacoplador

Ligando o Lado de Baixa Tensão

- Pino 2 do opto → terra do carro
- Pino 3 do opto → GND do Arduino
- Pino 4 do opto → pino D2 do Arduino (via resistor de 10kΩ pull-up)

Ligando o Display

- VCC → 5V
- GND → GND
- SDA → A4
- SCL → A5

Alimentação do Arduino

- Use um **buck converter** LM2596
- Ajuste saída para **5.00V** certinho
- Alimente o Arduino pelo pino 5V

Testes

- Abra o Serial Monitor
- Dê partida no carro
- Os valores de RPM devem aparecer imediatamente

Instalação Final

- Solde os componentes

- Isole bem os fios
 - Fixe o display onde desejar no painel
-

Código Base para Medição de RPM com Arduino

COMO UTILIZAR INTERRUPÇÕES PARA CALCULAR A ROTAÇÃO POR MINUTO DE MOTORES

Código Base para Medição de RPM

Este código é um exemplo básico para medir a rotação por minuto (RPM) de um motor usando um Arduino. Ele utiliza interrupções para contar pulsos e calcular o RPM a cada segundo. Esse tipo de medição é útil em diversas aplicações, como em automação industrial e projetos de robótica.

Explicação do Código

Variáveis Globais

- **lastTime**: Armazena o tempo em milissegundos desde o último cálculo de RPM.
- **pulseCount**: Variável volátil que conta os pulsos detectados por uma interrupção.

Funções

Função countPulse

```
void countPulse() {
    pulseCount++;
}
```

Esta função é chamada sempre que ocorre uma interrupção no pino digital 2 do Arduino (pino de interrupção 0 no Arduino Uno/Nano). A cada detecção de uma borda de subida (de 0V para 5V), o contador de pulsos é incrementado.

Função setup

```
void setup() {
    Serial.begin(9600);
    attachInterrupt(digitalPinToInterrupt(2), countPulse, RISING);
}
```

A função setup é executada uma vez na inicialização do Arduino. Nela, a comunicação serial é inicializada para fins de depuração, e a interrupção é configurada para detectar mudanças no pino 2.

Função loop

```
void loop() {  
    unsigned long currentTime = millis();  
  
    if (currentTime - lastTime >= 1000) {  
        int pulses = pulseCount;  
        pulseCount = 0;  
        int rpm = (pulses / 2) * 60;  
        Serial.print("RPM: ");  
        Serial.println(rpm);  
        lastTime = currentTime;  
    }  
}
```

A função loop é executada continuamente. Ela verifica se um segundo se passou desde o último cálculo de RPM. Se sim, o número de pulsos é salvo, o contador é zerado, e o RPM é calculado. A fórmula $(\text{pulses} / 2) * 60$ é usada considerando que um motor de 4 cilindros gera 2 pulsos por volta. O valor de RPM é então impresso na porta serial.

Atualização do Display

No final da função loop, há um comentário indicando onde a lógica para atualizar um display OLED poderia ser inserida. Isso geralmente envolve bibliotecas específicas, como u8g2 ou Adafruit SSD1306, para desenhar o valor de RPM na tela.

Conclusão

Este código é uma base sólida para medir RPM usando um Arduino. Ele pode ser expandido para incluir funcionalidades adicionais, como exibir dados em um display ou armazenar leituras para análise posterior.

Tacômetro Digital Simples para Gol 93 - Motor AP

Este projeto mostra como criar um tacômetro digital para um Gol 93 com motor AP usando um Arduino. Ele conta os pulsos elétricos da bobina do motor e transforma em rotações por minuto (RPM). O código é comentado de forma simples para facilitar o entendimento.

Como Funciona

- **Sinal de Entrada:** O sinal vem da bobina do motor.
- **Isolamento:** Um optoacoplador isola eletricamente o circuito.
- **Contagem de Pulses:** O Arduino usa interrupções para contar os pulsos.
- **Cálculo de RPM:** Converte os pulsos para RPM e mostra no monitor serial.

Código

```
/*
Tacômetro Digital para Gol 93 - Motor AP
Autor: Víctor Aurélio da Silva Barros

Como Funciona:
- Sinal da bobina
- Passa pelo optoacoplador
- Arduino conta pulsos
- Converte para RPM e mostra
*/
```

```
// Variáveis para tempo e contagem de pulsos
unsigned long lastTime = 0;
volatile unsigned int pulseCount = 0;

// Função chamada quando há um pulso
void countPulse() {
    pulseCount++;
}

void setup() {
    // Configuração da comunicação serial
    Serial.begin(9600);

    // Configura a interrupção no pino D2
    attachInterrupt(digitalPinToInterrupt(2), countPulse, RISING);
}

void loop() {
    // Tempo atual em milissegundos
    unsigned long now = millis();

    // A cada segundo, calcula o RPM
    if (now - lastTime >= 1000) {
```

```

// Armazena e reseta a contagem de pulsos
int pulses = pulseCount;
pulseCount = 0;

// Calcula o RPM (motor AP gera 2 pulsos por volta)
int rpm = (pulses / 2) * 60;

// Mostra o RPM no monitor serial
Serial.print("RPM = ");
Serial.println(rpm);

// Atualiza o tempo
lastTime = now;
}

}

```

Funções Explicadas

- **countPulse():** Dispara quando o Arduino detecta um pulso, aumentando a contagem de pulsos.
- **setup():** Inicializa a comunicação serial e ativa a interrupção no pino 2, que detecta quando a tensão sobe de 0V para 5V.
- **loop():** Roda continuamente. A cada segundo, calcula o RPM com base nos pulsos. O motor AP gera dois pulsos por volta.

Notas Adicionais

- **Interrupções:** Garantem que todos os pulsos sejam contados sem precisar verificar constantemente o pino.
- **Optoacoplador:** Protege o Arduino de picos de tensão da bobina, garantindo isolamento seguro.

Este projeto é ótimo para quem gosta de carros e eletrônica e quer aprender a integrar sistemas eletrônicos com veículos.