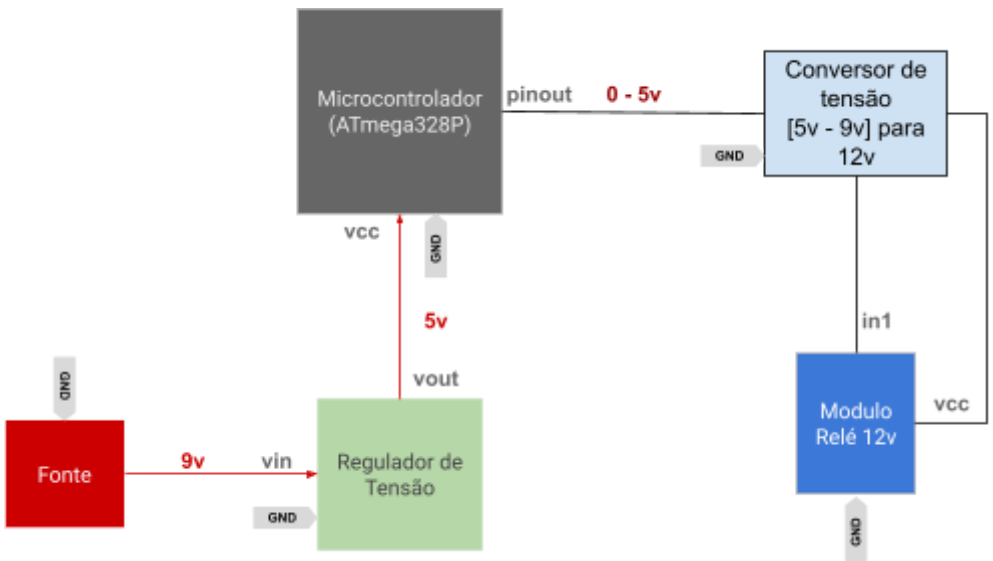
Atividade de Sistemas Embarcados

1. Esboce o diagrama em bloco e circuito elétrico utilizando um microcontrolador ATmega328P, para controlar um módulo Relé 12V.



Explicação do Diagrama:

Nesse diagrama é utilizado:

1. Fonte de tensão 9v
2. Circuito Regulador de 9v - 5v
3. Microcontrolador ATMEGA328P
4. Conversor de Tensão com entrada mínima de 5v e saída de 12v
5. Modulo Relé 12v.

A fonte fornece uma alimentação para o regulador de tensão que por sua vez alimenta o microcontrolador com 5v. Através do pino 1 do microcontrolador é enviado um sinal de 5v para um conversor de tensão e transmite o sinal acionar o relé

2. Calcule o valor a ser carregado no registrador **OCR1A** para que o TIMER1 gere uma interrupção por comparação a cada 250 ms.

Obs:
O microcontrolador é o ATmega328P utilizado em um Arduino UNO. Prescaler de 256

Solução 2:

OCR1A = (tempo * f_{clock}) / prescaler
OCR1A = 250ms * 16Mhz / 256
OCR1A = 0.250s * ($16*10^6$) / 256
OCR1A = 15625
OCR1A = 0011 1101 0000 1001

3. Vimos na última aula que a maior frequência alcançada na GPIO2 configurada como saída foi de ~136 KHz quando o microcontrolador ATmega328P utilizado na placa Arduino Nano foi carregada com o código apresentado no quadro abaixo. Reescreva o código para que seja possível aumentar a frequência máxima.

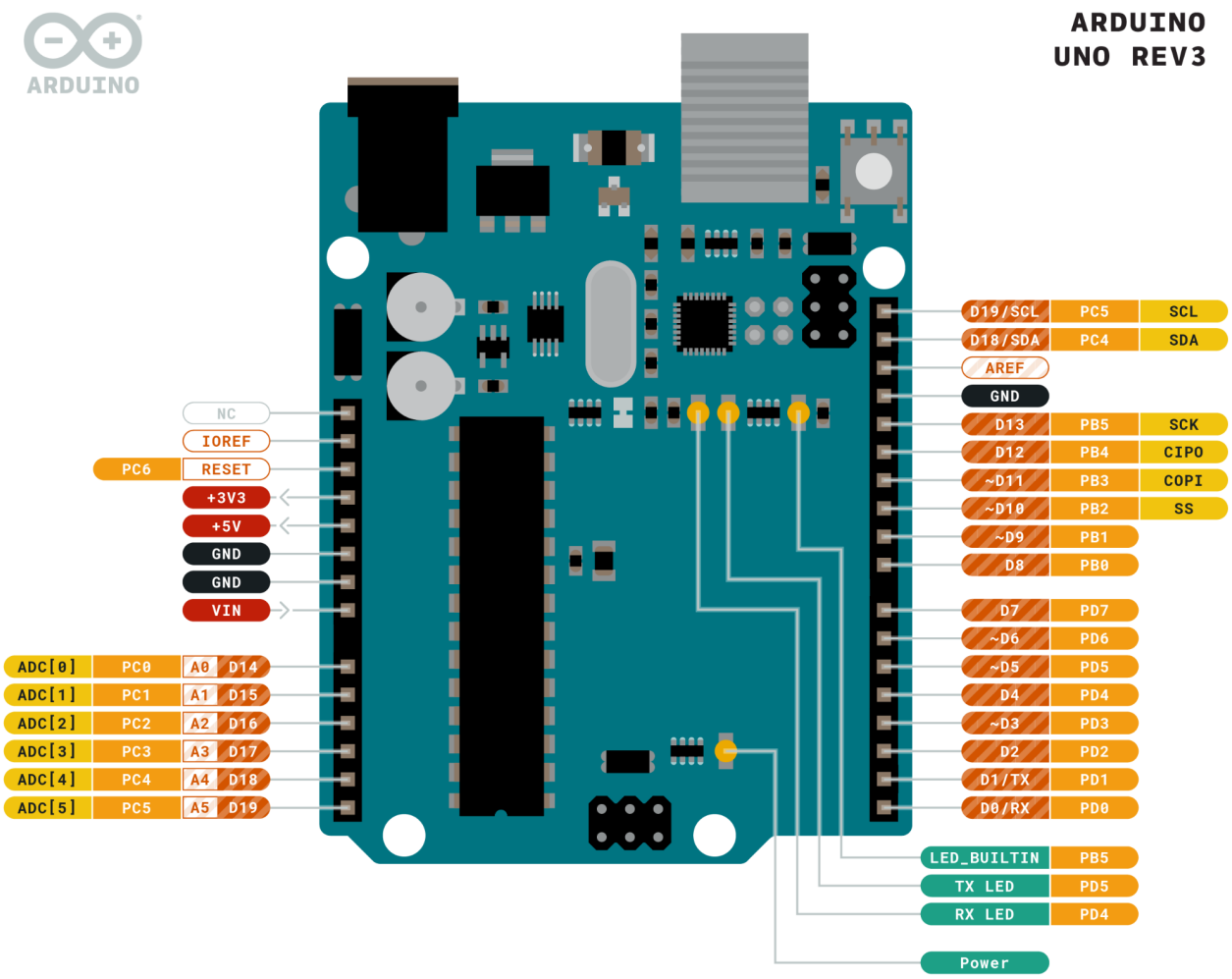
```arduino
#include <Arduino.h>

int pinLED = 2;

void setup() {
  pinMode(pinLED, OUTPUT);
}

void loop() {
 digitalWrite(pinLED, HIGH);
 // delay(delayPeriod);
 digitalWrite(pinLED, LOW);
 // delay(delayPeriod);
}
```

Solução 3:

Baseado no código apresentado anteriormente, as operações são realizadas utilizando comandos de alto nível ou funções abstraídas, como por exemplo:

`pinMode()`
`digitalWrite()`

Uma forma alternativa e mais rápida de fazer isso é acessando os pinos através de registradores.



Através da figura podemos ver que o PINO 2 do arduino uno está conectado a porta D2. Isso significa que acessando a PORTA D no bit 2 conseguimos alterar o estado do pino 2. Desse modo o código ficaria:

```
#include <Arduino.h>

void setup() {
 // Primeiro devemos setar a Porta D2 como Output
 // ativando o bit 2 usando operação OR bit a bit.
 DDRD |= (1 << 2);
}

void loop() {
 PORTD |= (1 << 2);  // Ativa o bit 2 da porta D: 0b00000100
 PORTD &= (0 << 2);  // Ativa o bit 2 da porta D: 0b00000000
}
```

Para testar foi usado o mesmo código para setar o pino 13 do arduino chegando no seguinte resultado:

```
#include <Arduino.h>

void setup() {
 DDRB |= (1 << 5);
```

```
}

void loop() {
 PORTB |= (1 << 5);
 delay(500);
 PORTB &= (0 << 5);
 delay(500);
}
```