Controlo programado com Arduino

Prática 10: O Sistema Automatizado com Retorno de Dados

Chegaste à última paragem da nossa viagem de aprendizagem! Nesta prática final integraremos praticamente tudo o que aprendeste. Desenheaderemos um sistema de segurança inteligente. Para estruturar o código de maneira impecável, criaremos uma função de diagnóstico que avaliará o estado do botão de pressão e devolverá un valor lógico (`return`) ao ciclo principal, decidindo se ativa o relé de potência e os LEDs de aviso.

Objetivos de Aprendizagem

Ao finalizar esta sessão, serás capaz de:

- Desenhar funções personalizadas que processem informação e devolvam valores lógicos através da instrução `return`.
- Compreender os tipos de retorno de dados em C++ (em especial, o tipo booleano `bool`).
- Integrar múltiplos componentes de entrada e saída (botão de pressão, LEDs de sinalização e relé de potência) num único circuito complexo.
- Desenvolver a lógica de um sistema industrial de segurança de paragem de emergência.

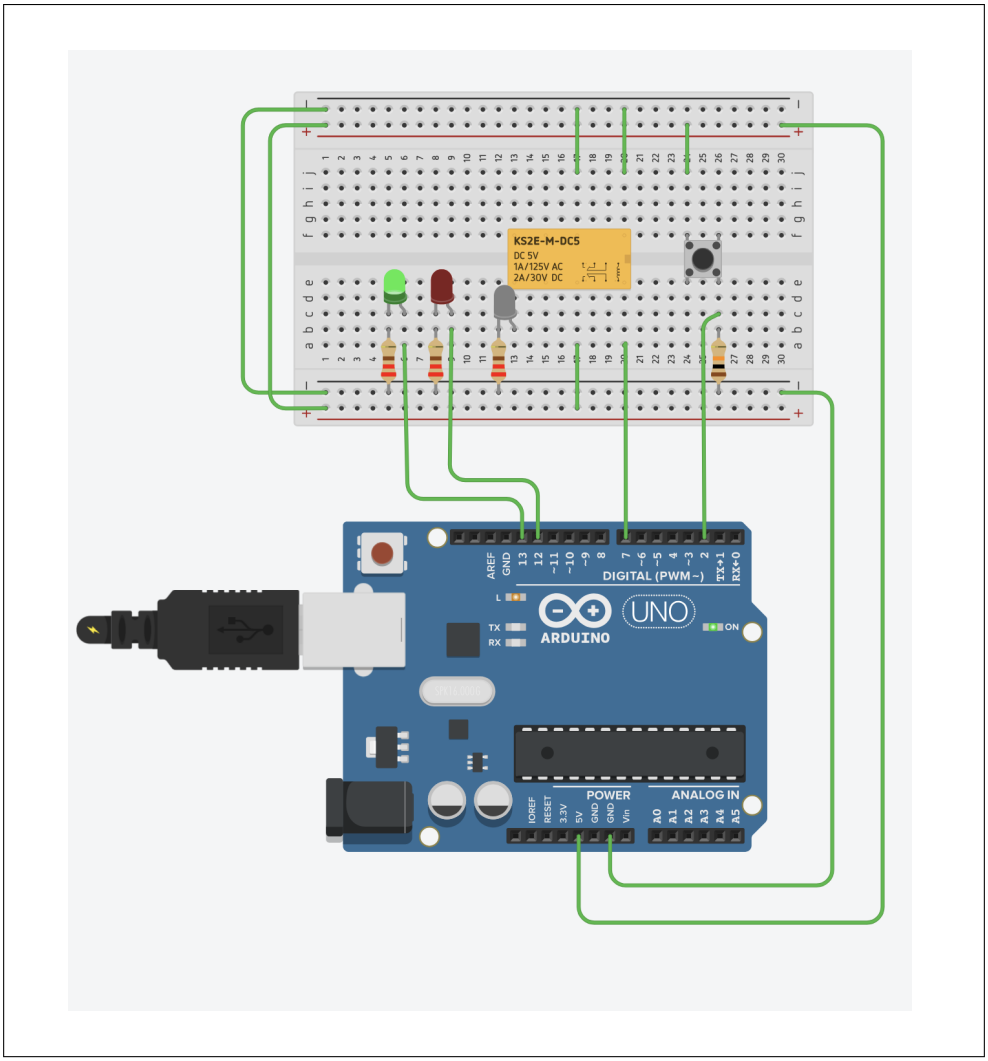
Componentes Necessários

Esta é a montagem mais completa do curso. Coloca os seguintes componentes no teu simulador TinkerCad:

- 1 Placa Arduino Uno.
- 1 Placa de ensaio pequena (Protoboard).
- 2 Díodos LED (1 Verde de ”Sistema Seguro” e 1 Vermelho de ”Perigo”).
- 2 Resistências de $220\,\Omega$ (para proteger os LEDs).
- 1 Botão de pressão de quatro pinos (Botão de emergência).
- 1 Resistência de $10\,\text{k}\Omega$ (para a configuração Pull-Down do botão).
- 1 Relé SPDT de 5 Volts (para simular o corte de alimentação de maquinaria de potência).
- Cabos de ligação virtuais.

Esquema de Montagem na Protoboard

Organiza a cablagem com paciência. Lembra-te de que a bobina do relé é ativada através de um pino do Arduino, e os seus contactos governam la potência de forma totalmente independente.



Código Base: Funções com tipo de retorno 'bool'

Escreve o seguinte programa no TinkerCad. Observa que o tipo de dados da nossa função já não é `void`, mas sim `bool`, o que significa que nos devolverá um valor de verdadeiro (`true`) ou falso (`false`) utilizando a instrução `return`.

```
// Pratica 10: Sistema de seguranca industrial com retorno de estado
int pinBoton = 2;
int pinRele = 7;
int ledVerde = 13;
int ledRojo = 12;

void setup() {
  pinMode(pinBoton, INPUT);
  pinMode(pinRele, OUTPUT);
  pinMode(ledVerde, OUTPUT);
  pinMode(ledRojo, OUTPUT);
}

void loop() {
  // 1. CHAMADA DA FUNCAO: Avaliamos o botao e guardamos o resultado
  bool alarmaActiva = evaluarSeguridad(pinBoton);

  // 2. ACOES COM BASE NO RETORNO DA FUNCAO
  if (alarmaActiva == true) {
    // Se a alarme estiver ativa, ativamos o rele e acendemos o LED vermelho
    digitalWrite(pinRele, HIGH);
    digitalWrite(ledRojo, HIGH);
    digitalWrite(ledVerde, LOW);
  } else {
    // Se tudo estiver seguro, o rele descansa e o LED verde brilha
    digitalWrite(pinRele, LOW);
    digitalWrite(ledRojo, LOW);
    digitalWrite(ledVerde, HIGH);
  }
}

// 3. DEFINICAO DA FUNCAO COM RETORNO
// Indicamos 'bool' no inicio porque a funcao deve devolver um booleano
bool evaluarSeguridad(int pinSensor) {
  int lectura = digitalRead(pinSensor);

  // Se for detetada uma pressao (perigo detetado)
  if (lectura == HIGH) {
    return true;  // Devolvemos o valor 'verdadero' ao loop()
  } else {
    return false; // Devolvemos o valor 'falso' ao loop()
  }
}
```

Listing 1: Código utilizando uma função com retorno de dados para avaliar o sistema.

Como funciona o código e a instrução `return`?

Até agora, todas as nossas funções terminavam com a palavra `void` porque apenas realizavam ações físicas. Mas em programação é vital que as funções realizem cálculos ou leituras e nos "devolvam" uma resposta.

- **Tipo de retorno (`bool`)**: No cabeçalho, substituímos `void` por `bool`. Isto obriga a função a devolver um dado booleano (verdadeiro ou falso) antes de terminar.
- **A instrução `return`**: É a porta de saída da função. Envia o valor selecionado de volta para a linha exata onde a função foi chamada. Assim que o programa lê `return`, a função termina imediatamente.
- **A receção do dado**: Na linha `bool` , o Arduino executa a função e substitui todo esse bloco pelo resultado devolvido (`true` o `false`), guardando-o diretamente na variável local .

O Desafio da Prática 10: O Desafio Final!

Consolida os teus conhecimentos completando este grande desafio de programação estruturada de segurança:

1. **Alarme Acústica Temporizada**: Adiciona um padrão de piscar de emergência. Se a função devolver `true`, faz com que o LED Vermelho pisque rapidamente de forma contínua, enquanto o LED Verde se mantém apagado.
2. **Função de Retorno Numérico**: Desenha uma nova função de diagnóstico chamada `comprobarErrores()` que não devolva um valor booleano, mas sim um número inteiro (`int`). Se o sistema estiver desligado, deve devolver 0. Se o sistema se ativar, deve contar os segundos que permanece ativo e devolver esse valor numérico ao `loop()` para mostrar o tempo de atraso na resposta.
3. *Pergunta para refletir*: Parabéns por completares as 10 práticas! Escreve uma breve reflexão pessoal sobre como evoluiu a tua forma de programar desde a Prática 1 (onde tudo era sequencial) até esta Prática 10 (com funções com passagem de parâmetros e retornos). De que forma esta estrutura te ajuda a pensar como um/a engenheiro/a de software?