Sistemas Embarcados e Robótica

Entrega 1 – Protótipo do Projeto em Arduino

Descrição do Projeto

O projeto consiste no desenvolvimento de uma solução de automação para salas de aula, voltada a instituições de ensino que desejam otimizar recursos, melhorar a gestão acadêmica e proporcionar maior organização no ambiente escolar.

A proposta integra diferentes funcionalidades de automação, como controle inteligente de iluminação e climatização, identificação de professores e alunos por meios de etiquetas RFID, gerenciamento de horários de aula e monitoramento de presença em avaliações. Essas funcionalidades têm como objetivo principal aumentar a eficiência operacional das instituições, reduzir desperdícios de energia elétrica e oferecer maior confiabilidade nos processos de controle acadêmico.

Além disso, a solução busca agregar valor para professores, alunos e gestores escolares ao proporcionar um ambiente mais confortável, organizado e transparente. Para a instituição, os dados coletados geram relatórios estratégicos sobre frequência, pontualidade e utilização dos recursos da sala, permitindo tomadas de decisão mais embasadas.

Arquitetura de Hardware

Para a primeira etapa do protótipo, fizemos um sistema que opera com base na criação de uma barreira invisível por meio de dois feixes de luz infravermelha paralelos, posicionados na entrada e na saída de uma sala. A lógica central reside na detecção de uma interrupção nesses feixes, na análise da sequência em que cada um deles foi interrompido e assim detectando se uma pessoa entrou ou saiu da sala. A quebra sequencial do feixe 1 seguido pelo feixe 2 caracteriza uma entrada, incrementando o contador de ocupantes na sala. O processo inverso sinaliza uma saída, decrementando o mesmo contador. O estado de ocupação da sala (vazia ou ocupada) é usado para ligar os periféricos da sala, como as lâmpadas, o ar-condicionado, o projetor e os computadores.

Para o protótipo, o Arduino Nano foi selecionado como cérebro devido a facilidade de trabalhar com ele, vasta documentação e facilidade de prototipagem. Ele é responsável por executar o software, ler os dados dos sensores e processar a lógica.

Como entrada, temos dois receptores e emissores de infravermelho bem simples ligados em um divisor de tensão, assim podemos construir uma solução confiável e barata, além de nos permitir ler a intensidade do sinal pelas portas analógicas e decidir quando considerar que o receptor foi interrompido ou não.

Como saída temos um LED simples que atua como uma representação bem simplificada do acionamento de sistemas de automação mais complexos, quando o contador de pessoas é maior que zero, o LED acende, simulando o envio de um sinal para ligar as luzes e o ar-condicionado.

Arquitetura de Software

O software embarcado no Arduino foi desenvolvido em C++ e estruturado como uma Máquina de Estados Finitos (MEF) para gerenciar o processo de contagem de forma confiável. Essa abordagem previne problemas como contagem múltipla e falsos positivos. Os principais estados do MEF são:

1. Ocioso: O sistema aguarda o início de uma passagem, monitorando ambos os sensores.
2. Sequência de Entrada: O primeiro sensor foi acionado. O sistema agora ignora este sensor e aguarda exclusivamente o acionamento do segundo para confirmar a entrada.
3. Sequência de Saída: O processo inverso da sequência de entrada, o segundo sensor foi acionado e o sistema aguarda o acionamento do primeiro.

A transição entre os estados é acionada por eventos de detecção de borda de descida. Ou seja, o código não reage ao nível ou a intensidade do sinal, mas sim à transição do estado livre para bloqueado. Isso é implementado comparando o estado atual do sensor com o estado registrado na iteração anterior do loop, garantindo que cada interrupção do feixe gere um único evento lógico. Adicionalmente, um mecanismo de timeout foi implementado para resetar a máquina de estados caso uma sequência de passagem seja iniciada, mas não concluída, dentro de um intervalo pré-definido, evitando que o sistema fique travado.

Componentes Utilizados

A lista abaixo detalha os componentes eletrônicos utilizados na montagem do protótipo:

1. 1x Arduino Nano – Controlar o sistema
2. 2x Emissores Infravermelhos – Emitir o feixe
3. 2x Receptores Infravermelhos – Receber o feixe

4. 1x LED de 10mm – Stub para as lâmpadas e climatização
5. Resistores de 220Ω - Ligar o LED e os emissores IR
6. Resistores de 10kΩ - Divisores de tensão com os receptores IR
7. Jumpers e Protoboard – Conectar os componentes

Validações e Resultados

Para validar a funcionalidade e a robustez do protótipo, foi executada uma série de testes controlados em bancada. A metodologia constituiu na simulação de cenários de uso real e o monitoramento em tempo real através do Monitor Serial da IDE do Arduino.

1. Diferenciação de sentido: O sistema conseguiu identificar corretamente se era uma entrada ou uma saída com base na sequência de acionamento dos sensores, atualizando o contador de pessoas e o estado do LED.
2. Prevenção de contagem dupla: Ao bloquear o sensor e manter o objeto parado, confirmamos que a contagem ocorrida apenas uma vez. Isso prova que o sistema é robusto e não registrará a mesma pessoa várias vezes.
3. Reset por Timeout: Simulamos uma passagem incompleta, bloqueando apenas o primeiro sensor. Após alguns segundos, o sistema reiniciou os estados, mostrando que é capaz de lidar com situações anormais sem travar.

Código

```
// --- Pinos ---
const int pinoRec1 = A3; // Sensor 1
const int pinoRec2 = A2; // Sensor 2
const int pinoLed = 4;   // LED que indica se a sala está ocupada

// --- Parâmetros de Detecção ---
const int LIMIAR = 100; // Desobstruído fica em torno de 300, bloqueado
cai para menos de 30, então 100 é seguro

// --- Variáveis de Lógica e Estado ---
int contadorPessoas = 0;

// Estados ATUAIS dos sensores: false = livre, true = bloqueado
bool estadoAtualSensor1 = false;
bool estadoAtualSensor2 = false;

// Estados ANTERIORES dos sensores (para detectar a mudança)
bool ultimoEstadoSensor1 = false;
bool ultimoEstadoSensor2 = false;
```

```
// Variável para controlar a sequência de passagem
// 0: Ninguém passando, sistema pronto
// 1: Aguardando o sensor 2 ser bloqueado (iniciou uma entrada)
// 2: Aguardando o sensor 1 ser bloqueado (iniciou uma saída)
int estadoPassagem = 0;

// Controle de tempo para evitar detecções falsas ou sequências travadas
unsigned long tempoInicioPassagem;
const long TIMEOUT_PASSAGEM = 2000; // 2 segundos de tempo máximo para
cruzar os dois sensores (por enquanto de teste, vamos ajustar )

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

  pinMode(pinoRec1, INPUT);
  pinMode(pinoRec2, INPUT);
  pinMode(pinoLed, OUTPUT);

  digitalWrite(pinoLed, LOW);

  Serial.println("Sistema de Contagem de Pessoas - V2 Robusto -
Iniciado");
  Serial.print("Pessoas na sala: ");
  Serial.println(contadorPessoas);
  Serial.println("-----------------------------------");
}

void loop() {
  // 1. Leitura dos sensores e definição do estado ATUAL
  estadoAtualSensor1 = (analogRead(pinoRec1) < LIMIAR);
  estadoAtualSensor2 = (analogRead(pinoRec2) < LIMIAR);

  // 2. Lógica principal baseada na MUDANÇA de estado (de livre para
bloqueado)

  // Se estivermos esperando o início de uma passagem (estado 0)
  if (estadoPassagem == 0) {
    // Se o sensor 1 ACABOU de ser bloqueado (antes estava livre)
    if (estadoAtualSensor1 && !ultimoEstadoSensor1) {
      estadoPassagem = 1; // Sequência de ENTRADA iniciada
      tempoInicioPassagem = millis();
      Serial.println("=> Sequência de ENTRADA iniciada (Sensor 1
bloqueado)...");
    }
    // Senão, se o sensor 2 ACABOU de ser bloqueado
    else if (estadoAtualSensor2 && !ultimoEstadoSensor2) {
      estadoPassagem = 2; // Sequência de SAÍDA iniciada
      tempoInicioPassagem = millis();
```

```
      Serial.println("<= Sequência de SAÍDA iniciada (Sensor 2
bloqueado)...");
    }
  }

  // Se uma sequência de ENTRADA já começou (estado 1)
  else if (estadoPassagem == 1) {
    // E o sensor 2 ACABOU de ser bloqueado
    if (estadoAtualSensor2 && !ultimoEstadoSensor2) {
      contadorPessoas++;
      Serial.print("ENTROU UMA PESSOA! Total na sala: ");
      Serial.println(contadorPessoas);
      // Reseta o sistema para a próxima pessoa
      estadoPassagem = 0;
      Serial.println("...Sistema pronto para nova contagem.");
      Serial.println("-----------------------------------");
    }
  }

  // Se uma sequência de SAÍDA já começou (estado 2)
  else if (estadoPassagem == 2) {
    // E o sensor 1 ACABOU de ser bloqueado
    if (estadoAtualSensor1 && !ultimoEstadoSensor1) {
      if (contadorPessoas > 0) {
        contadorPessoas--;
      }
      Serial.print("SAIU UMA PESSOA! Total na sala: ");
      Serial.println(contadorPessoas);
      // Reseta o sistema para a próxima pessoa
      estadoPassagem = 0;
      Serial.println("...Sistema pronto para nova contagem.");
      Serial.println("-----------------------------------");
    }
  }

  // 3. Lógica de Timeout
  // Se uma passagem começou mas não terminou em um tempo razoável
  if ((estadoPassagem == 1 || estadoPassagem == 2) && (millis() -
tempoInicioPassagem > TIMEOUT_PASSAGEM)) {
    Serial.println("!!! TIMEOUT: A passagem não foi completada. Resetando
sistema.");
    Serial.println("-----------------------------------");
    estadoPassagem = 0; // Reseta a máquina de estados
  }

  // 4. Controle do LED
  if (contadorPessoas > 0) {
    digitalWrite(pinoLed, HIGH);
  } else {
```

```
    digitalWrite(pinoLed, LOW);
  }

  // 5. ATUALIZAÇÃO DOS ESTADOS ANTERIORES - ESSENCIAL!
  // Prepara as variáveis para a próxima iteração do loop
  ultimoEstadoSensor1 = estadoAtualSensor1;
  ultimoEstadoSensor2 = estadoAtualSensor2;

  delay(20); // Um pequeno delay para estabilidade
}
```