Alarme PIR-RFID

Alec Coelho

Novembro 2023

1 Introdução

O projeto consistirá, como mencionado no título, em um sistema de segurança que, ao detectar uma entrada não autorizada, soará um alarme. Para realizar o projeto, será utilizado um sensor de presença por infravermelho (PIR), um módulo leitor de radiofrequência (RFID) com chip MFRC522 além de LEDs e um buzzer. Estes periféricos estarão ligados à um microcontrolador ESP32, também conectado ao computador do usuário ou alimentado por bateria externa.

2 Classes

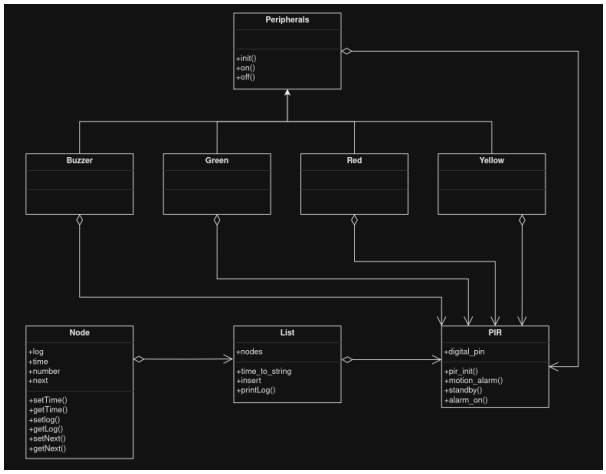

Figura 1: Diagrama de classes

Utilizando a linguagem de programação C++ com orientação a objetos, o software embarcado utiliza-se de diversas classes que representam, de forma geral, os periféricos e sensores utilizados, além das classes utilizadas para registro (logs). Primeiramente, a classe *Peripherals* foi utilizada como uma classe virtual herdada por todos os periféricos, que consistem em 3 LEDs e um buzzer. E possui apenas 3 métodos: *init*, *on* e *off*. Estes métodos são responsáveis, respectivamente, por inicializar, ligar e desligar o periférico de acordo com o pino definido em cada classe derivada.

Seguidamente, as classes "Nodo"e "*List*"foram utilizadas com o intuito de registrar cada entrada detectada pelo sensor de presença, seja esta autorizada ou não. Cada objeto da classe "Nodo"representa um destes registros. A classe *List* agregará variáveis do tipo nodo, mas terá apenas um objeto criado, e este será responsável por encadear os nodos contendo os registros, além de imprimi-los na porta serial quando solicitado.

Por fim, a classe "PIR"é a mais volumosa em termos de linhas de código, pois esta representa o sensor de movimento, que será responsável por administrar o uso dos periféricos e da lista de registros, agregando todas as outras classes no programa (exceto por "Nodo"). Seus métodos incluem uma função para inicialização, *standby* (declarado como *friend*), *motion alarm* e *alarm on*, explicados de forma mais detalhada na sessão abaixo.

3 Funcionamento

Ao energizar o ESP32 através de sua entrada USB ou pinos "VCC"e "GND", o programa será iniciado, e automaticamente entrará em modo de *stand-by* chamando o método de PIR com este mesmo nome.

Desta forma, o programa manterá o LED amarelo aceso até que seja detectado um cartão magnético no módulo RFID. Caso esta detecção ocorra, o sistema entrará em modo de ativação do sensor PIR, onde permacerá por 2 minutos, imprimindo o tempo restante a cada 10 segundos e piscando o LED verde a cada segundo (Caso seja detectado um cartão magnético novamente dentre o intervalo de 2 minutos, a ativação será cancelada e o sistema retornará para o modo de *stand-by*). Ao fim desta contagem, o LED verde permanecerá ligado de forma constante, indicando que o método *motion alarm* foi chamado. Durante a execução desta tarefa, o sistema mantém uma leitura constante da porta digital do sensor PIR. Caso haja tensão fornecida pelo dispositivo, indicando alguma detecção de movimento, uma nova contagem de 10 segundos será iniciada. Caso algum cartão magético seja detectado pelo módulo RFID antes que a contagem acabe, o movimento detectado será considerado uma entrada autorizada, e o sistema retornará ao modo de *stand-by*. Caso a contagem seja encerrada, esta entrada será considerada como não-autorizada, e o alarme será acionado pelo método *alarm on*, piscando os 3 LEDS juntos de um buzzer soando no mesmo ritmo, e o sistema permanecerá desta forma até que o módulo RFID detecte um cartão magnético.

```
//--------------    LOOP     --------------------------

void loop() {

  standby(); //Waits for card aproximation

  bool pir_activated = motion_sensor.pir_init(PIR_digital);

  if(!pir_activated){
    return;
  }//endif

  Serial.println("SYSTEM ACTIVATED");

  PRESENCE presence = motion_sensor.motion_alarm();
}  //endloop
```

Figura 2: Função *main*

4 Logs

Como requisitado nas especificações do presente projeto, o software implementado conta com a função de armazenar *data logs* na forma de registros de entradas detectadas através das classes "Nodo"e "Lista". De acordo com o que foi mencionado anteriormente, cada nodo possuirá um registro. Este registro conterá a data e hora de uma entrada, um número contendo a ordem cronológica de cada registro, e se esta entrada foi ou não autorizada. Para solicitar ou apagar a lista de registros, basta o usuário enviar os caracteres correspondentes descritos na saída serial enquanto o sistema estiver em modo de *stand-by*. Segue exemplo de impressão da lista:

```
---------- Log entries -------------
[22/11/2023 - 15:33:25] 1. Authorized entrance detected
[22/11/2023 - 15:33:66] 2. UNAUTHORIZED ENTRANCE DETECTED
[22/11/2023 - 15:34:22] 3. Authorized entrance detected
[22/11/2023 - 15:34:36] 4. Authorized entrance detected
Total of entries: 4
------------------------------------
```

Figura 3: Exemplo de requisição de logs