PONTO DE CONTROLE 2
CONTROLE DE ACESSO VIA RECONHECIMENTO DE FACE HUMANA

Antônio Aldísio - 14/0130811 —- Vitor Carvalho de Almeida - 14/0165380

Programa de Graduação em Engenharia Eletrônica, Faculdade Gama
Universidade de Brasília
Gama, DF, Brasil

email: aldisiofilho@gmail.com —- vitorcarvalhoamd@gmail.com

RESUMO

O projeto consiste em construir um sistema de controle de acesso ativado por reconhecimento facial. Será possível enviar os dados de acesso via rede para um banco de dados. Como validação será confeccionada uma porta em miniatura.

Palavras-chave: Controle de acesso, Raspberry Pi, OpenCV, reconhecimento facial, segurança.

1. INTRODUÇÃO

Para este ponto de controle, é necessário comunicar a Raspberry Pi com os elementos que serão utilizados no projeto.

O projeto em questão utiliza uma trava solenoide, que trabalha com tensão e corrente maiores do que a placa consegue fornecer. Logo, é necessário usar um sistema de chaveamento.

Nas próximas seções são apresentadas as soluções para o problema.

O reconhecimento facial deste trabalho foi feito utilizando uma biblioteca específica, chamada OpenCV.

2. DESENVOLVIMENTO

2.1. Descrição do Hardware

Foi montado um sistema de ativação da trava eletrônica. Utilizando os seguintes materiais:

- Trava solenoide 12V (figura 1);
- Fonte DC 12V ;
- Resistor de 1 KOhm;
- Transistor NPN (TIP41);
- Jumpers
- Protoboard

Fig. 1. Trava eletrônica solenoide 12V

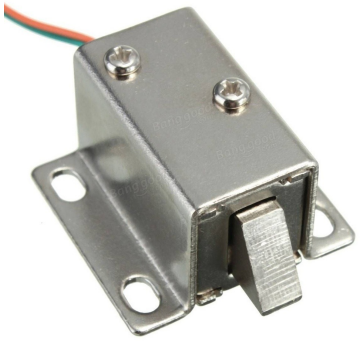
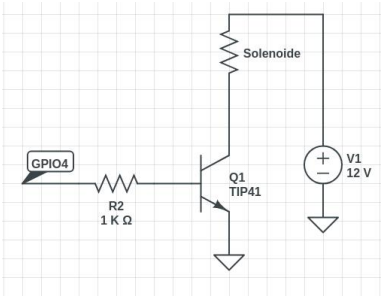


Fig. 2. Ativação da trava eletrônica solenoide 12V



Na protoboard foi montado o circuito da figura 2.

O pino de entrada foi conectado à GPIO4 da Raspberry Pi 3 para que fossem enviados os comandos para abrir a porta.

A trava solenoide mantem a porta fechada até que seja inserida uma tensão de 12V em seus terminais. Neste momento, o solenoide faz com que o ”dente” da trava seja retraído, liberando a abertura da porta. Ao retirar a tensão dos terminais, uma mola retorna a trava para a posição original, travando a porta novamente. [2]

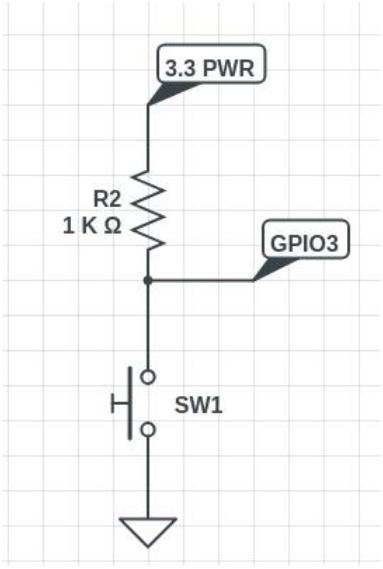
Foi utilizada uma fonte DC de 12V - 2A com conexão

Jack P4, ligada na protoboard com um conector Jack P4 fêmea.

Foi conectada uma caixa de som à saída P2 da Raspberry Pi para reproduzir sons de confirmação ou negação de acesso.

Para receber a requisição de acesso, foi montado um circuito com botão em modo Pull-Up, como mostra o esquematico da figura 3

Fig. 3. Botão em modo Pull-Up



2.2. Descrição do Software

2.2.1. Campainha

A GPIO3 foi configurada de modo a ficar em modo

```
1 Modo de espera
2 Botão foi pressionado? N–Espera S–
      Segue
3 Envia requisição de acesso para o
      código principal.
```

2.2.2. Resposta ao usuário

A rotina para liberar a porta foi feita em um arquivo de instruções bash. O código *abre.sh* é mostrado no apendice.

1 Primeiramente é reproduzido o som de uma confirmação de acesso pela caixa de som, para que o usuário saiba que pode entrar

2 Depois a GPIO4 é definida como saída e colocada em nível lógico alto.

3 É definido um tempo de espera, no caso 3 segundos, para que a trava se mantenha retraída e o usuário possa empurrar a porta.

4 Ao final da contagem a GPIO4 volta para o nível lógico baixo. Neste momento, ao encostar a porta, esta será trancada.

5 Por fim, a GPIO4 é liberada para uso em outra rotina.

Caso o usuário não tenha acesso cadastrado, ao tocar a campainha deve ser executado o código *negado.sh*, mostrado abaixo

```
1 #!/bin/bash
2
3 omxplayer -o local /home/pi/
      embarcados/projeto_final/sons/nao.
      mp3
```

Neste caso, a única função realizada é a reprodção de um som de negação na caixa de som.

3. RESULTADOS

[FOTO CIRCUITO]

A ativaÃ§Ã£o da trava eletrÃ´nica foi realizada com sucesso, sem sobreaquecimento do transistor, nem falha na comunicaÃ§Ã£o.

4. DISCUSSÃO E CONCLUSÕES

Pelos experimentos realizados, concluiu-se que o projeto pode ser implementado da forma como foi pensado inicialmente, pois tanto a comunicação da Raspberry Pi com os periféricos (trava, botão, câmera) quanto a biblioteca de reconhecimento facial do OpenCV funcionaram corretamente.

Decidiu-se usar o bot do Telegram como interface do administrador com o sistema, porque assim o acesso é muito mais prático e podem ser recebidas notificações no celular sem necessidade de instalação de outros aplicativos.

Ainda é necessário otimizar o uso do reconhecimento facial para que seja considerada a profundidade de campo, evitando que a porta seja aberta com a foto de um usuário cadastrado.

Para este documento, foram escritos programas cuja rotina principal é específica para cada ação. Porém, para o próximo ponto de controle, os códigos serão unidos como subrotinas de um código principal de controle do sistema.

5. REFERENCIAS

[1] https://github.com/opencv/opencv

[2] https://www.filipeflop.com/blog/acionando-trava-eletrica-com-rfid/

6. APENDICE

Códigos utilizados

Rotina do botão: *campainha.c*

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <fcntl.h>
4  #include <sys/poll.h>
5  #include <unistd.h>
6
7  int main(void)
8  {
9          struct pollfd pfd;
10         char buffer;
11         system("echo 3 > /sys/class/gpio/
               export");
12         system("echo falling > /sys/class/
               gpio/gpio3/edge");
13         system("echo in > /sys/class/gpio/
               gpio3/direction");
14         pfd.fd = open("/sys/class/gpio/
               gpio3/value", O_RDONLY);
15         if(pfd.fd < 0)
16         {
17                 puts("Erro abrindo /sys/
                       class/gpio/gpio3/value
                       ");
18                 puts("Execute este
                       programa como root");
19                 return 1;
20         }
21         read(pfd.fd, &buffer, 1);
22         pfd.events = POLLPRI | POLLERR;
23         pfd.revents = 0;
24         puts("Augardando campainha");
25         poll(&pfd, 1, -1);
26         if(pfd.revents) puts("Campainha
               pressionada: iniciar rotina de
               reconhecimento");
27         close(pfd.fd);
28         system("echo 3 > /sys/class/gpio/
               unexport");
29         return 0;
30 }
```

abre.sh:

```
1  #!/bin/bash
2
3  GPIO_PATH=/sys/class/gpio
4
5  omxplayer -o local /home/pi/embarcados/
       projeto_final/sons/sim.mp3
6  echo 4 >> $GPIO_PATH/export
7  sudo echo out > $GPIO_PATH/gpio4/direction
8  sudo echo 1 > $GPIO_PATH/gpio4/value
9  sleep 3
10 echo 0 > $GPIO_PATH/gpio4/value
11 echo 4 >> $GPIO_PATH/unexport
```