MÁQUINA DE REFRIGERANTE COM TRAVA DE SEGURANÇA

Kewin Kuster, Rodrigo Sousa Santos

Programa de Graduação em Engenharia Eletrônica, Faculdade Gama
Universidade de Brasília
Gama, DF, Brasil
email: kevinkiister0@gmail.com, rodrigo.sousa2711@gmail.com

1. JUSTIFICATIVA

A motivação inicial para a criação de uma máquina de refrigerantes com um sistema de controle de usuário partiu do reconhecimento do problema enfrentado por empresas alimentícias em locais públicos aglomerados como shoppings, hipermercados, grandes centros de comercio entre outros exemplos. O intuito da construção do sistema consiste em monitorar e controlar o consumo de bebidas em ambientes abertos, de modo a evitar que pessoas que não sejam clientes utilizem dos produtos disponibilizados como refil fornecidos pela loja. Uma vez que a utilização da máquina com a trava tornaria muito mais rara eventos como esses.

O sistema funciona através de uma simples implementação aos modelos de máquinas utilizadas hoje em dia, com a adição de uma câmera fotográfica capaz de realizar o processamento de um código adicionado aos copos fornecido pela empresa distribuidora da bebida, podemos realizar todo o controle de acesso a máquina de refrigerantes.

2. OBJETIVOS

O objetivo deste projeto é construir um sistema capaz de controlar uma máquina de refrigerante de refil na forma de evitar que pessoas burlem o sistema, realizando desse modo um maior controle sobre o fluxo de serviço prestado pela máquina.

3. REQUISITOS

Para realização do projeto, será impresso em cada copo um código qr, também chamado de qr code. Ao realizar a compra do copo para refil, será inserido no qr code uma validação que irá durar um certo período de tempo sem prejudicar o modo refil de utilização. Após o cadastramento do qr code, a pessoa passará o copo em um leitor de qr code, onde será utilizada uma câmera para leitura. Esta câmera estará conectado a uma raspberry pi3 para o processamento da imagem e análise dos dados. A mesma será utilizada para acionar a máquina de refrigerante, liberando o funcionamento.

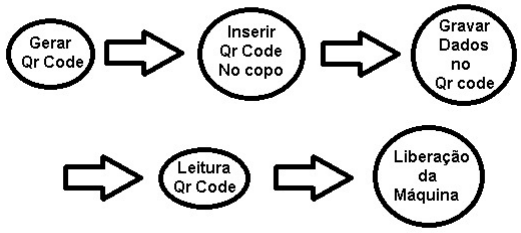

Fig. 1. Fluxograma do projeto

4. BENEFÍCIOS

O intuito da realização deste projeto está em conseguir que a utilização de um produto seja dada somente pelos clientes da loja de alimentos, fazendo com que haja a diminuição ou até mesmo a extinção desse mesmo consumo por terceiros, para que não haja prejuízo para a empresa que optar pela utilização do mesmo.

5. ASPECTOS DE HARDWARE

Segue abaixo uma tabela de materiais utilizados na implementação do projetos, quantidades e uma breve descrição de cada componente.

Tabela 1. Materiais utilizados no projeto.

Componentes	Descrição
RaspberryPi3	Sistema Embarcado
Câmera	Leitura do QrCode
Módulo Relé	Acionamento da Bomba
Bomba de Aquário	Líquido liberado

5.1. Raspberry Pi3

Raspberry Pi é um computador de baixo custo e que tem o tamanho de um cartão de crédito desenvolvido no Reino Unido pela Fundação Raspberry Pi. Ela possui Wifi e bluetooth integrado, um processador quad-core de 64 bits (Broadcom BCM2837), clock de 1.2 GHz e ainda conta ainda com uma arquitetura avançada, da Cortex-A53. Na parte gráfica, usa um processador gráfico VideoCore IV 3D, que consegue rodar vídeos em 1080p com relativa tranquilidade.[3] A placa possui 4 portas USB, saída de áudio e vídeo composto no mesmo conector, porta HDMI e conectores para câmera e display, além do conector de 40 pinos GPIO.[3] Para o projeto, a Raspberry Pi será utilizada para o processamento de imagens do QrCode feitas pela câmera , e a própria placa acionará o relé e consequentemente a bomba permitindo ou não a utilização da máquina. Foi escolhida a versão 3 por causa do seu poder de processador de 64 bits, processador gráfico para processamento de imagens, além do seu clock superior a outras versões.

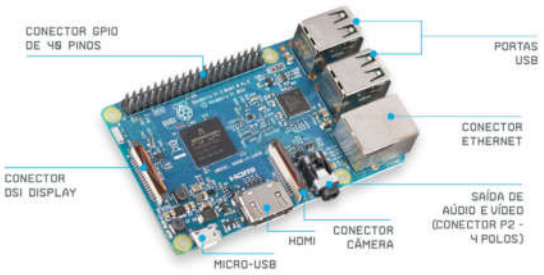

Fig. 2. Placa Raspberry Pi 3.

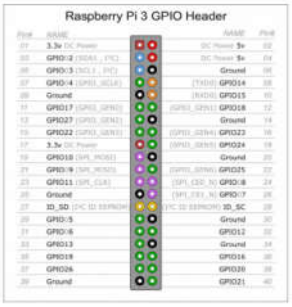

Fig. 3. Pinos GPIO;

5.2. Câmera Raspberry Pi3

A câmera Raspberry Pi é uma câmera digital em um módulo bastante leve, pesando apenas 3 gramas, e compacto com medidas de (25 x 20 x 9mm). Ela gera fotos com resolução de até 25921944 pixels e vídeos com resolução de até 1080p.[4]

Fig. 4. Pinos GPIO;

A imagem abaixo apresenta um esquemático mostrando a conexão entre a câmera e a placa Raspberry Pi 3.

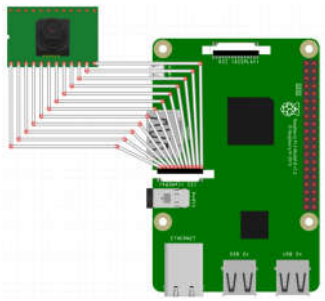

Fig. 5. Conexão entre câmera e a placa.

5.3. Bomba de Aquário

Para realização da máquina de refrigerantes, era necessário encontrar um modo de entrega esse refrigerante ao usuário, e o modo encontrado pela equipe foi de utilizar uma bomba de aquário com o intuito de injetar ar no interior do recipiente contendo liquido, fazendo com que o mesmo seja entregue ao cliente. Para isso foi utilizado a bomba sarlobetter mimi a de 2W, que por possuir um baixo consumo e uma vazão de até 1L/min se encaixa no escopo do projeto. As imagens abaixo apresenta uma figura da bomba utilizada no projeto e também um esquemático mostrando o modo que foi feita a conexão entre a bomba e a placa Raspberry Pi 3.

Fig. 6. Bomba de Aquário.

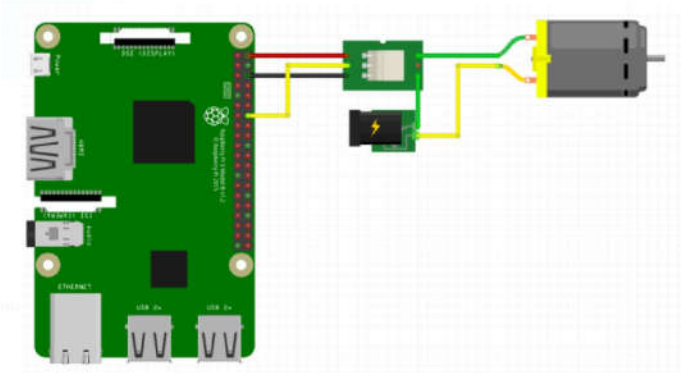

Fig. 7. Conexão entre a bomba e a placa.

Tabela 2. Pinagem entre o módulo e a placa.

Raspberry Pi3	Módulo Relé
Pin 02	VCC
Pin 06	GCC
Pin 12	Signal

5.4. Módulo Relé

A utilização do módulo relé vem da necessidade de controle do acionamento da bomba uma vez que a mesma precisa ser alimentada com uma tensão de 220V e a placa trabalha com valores muito inferiores.

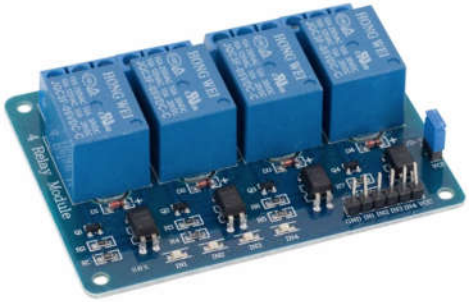

Fig. 8. Módulo Relé.

Tabela 3. Pinagem entre a bomba e o relé.

Módulo Relé	Bomba de Aquario
Pin 01	VCC Bomba
Pin 02	Power

6. ASPECTOS DE SOFTWARE

6.1. Raspberry Pi 3

O sistema operacional utilizado na Raspberry Pi é o Raspbian, sendo este um sistema livre baseado no Debian, otimizado para o hardware Raspberry Pi.[5] Ele foi instalado em um cartão SD para ser utilizado na placa, já que a Raspberry não possui HD interno. Já dentro do sistema operacional foi utilizado a IDE Python 3, pois toda a programação utilizada no projeto foi em Python.

6.2. Câmera

Para utilização da câmera na Raspberry Pi 3 após a conexão de hardware, foi acessada as configurações da placa clicando em Berry -> Preferences -> Raspberry Configuration. Clicando na aba Interfaces, ativamos a câmera clicando em Enabled. Após darmos reboot para a configuração ser validada, abrimos a IDE Python 3 e escrevemos o código que se encontra em anexo.[6]

No código utilizado em Python, primeiramente é definida as duas bibliotecas, sendo elas a PiCamera, que é a principal interface do módulo da câmera do Raspberry Pi, e a biblioteca time, chamando a função sleep, que pausa o funcionamento do programa. Com a função camera.start-preview() é possível iniciar a exibição ao vivo da entrada da câmera, esperamos 5 segundos com ela aberta utilizando sleep(5) e fechamos a câmera com camera. stop-preview(). [6]

6.3. Leitor de QR Code

Para realizar a leitura de uma imagem QR code foi utilizada a biblioteca zbar em python, onde através do acionamento da câmera é possível reconhecer os padrões de um QR code e com isso escanear uma imagem. Uma vez feita o reconhecimento a imagem passa pelo processo de decodificação onde o comando pyzbar.decode encontra os comandos que estão guardados dentro da imagem para assim para no final converter esses dados em uma string.

6.4. Módulo Relé

Para utilização do módulo relé, foi utilizado os pinos de GPIO. O código utilizado se encontra em anexo. No código iniciamos importando a biblioteca para manipulação dos pinos de entrada e saída. Em seguida, através da sentença GPIO.set mode() determinamos a maneira como vamos nos referenciar aos pinos da placa Raspberry Pi. Ao utilizarmos GPIO.BOARD como parâmetro, devemos nos referenciar aos pinos, no código, pela ordem em que estão anexados na mesma. Posteriormente, definimos o modo de operação dos pinos da placa Raspbarry Pi, de modo que, configuramos o pino 12 como um pino de saída digital. Por fim, criamos um loop infinito através da sentença while (True),e ativamos e desativamos o pino de saída 12 utilizando nível lógico alto (1) e baixo (0) respectivamente.[7]

7. BIBLIOGRAFIA

1 Raspberry Pi Cookbook for Python Programmers. Cox, Tim

2 Roubo de Refil. Disponível em http://varelanoticias.com.br/garotos-levam-galao-de-20-litros-para-encher-refil-de-refrigerante-no-burger-king-veja-video

3 Filipe Flop. Guia Raspberry para iniciantes, 2016

4 Câmera Raspberry Pi. Disponível em https://www.filipeflop.com/blog/modulo-camera-raspberry-pi/

5 Welcome to Raspbian. Disponível em https://www.raspbian.org/

6 PiCamera. Disponível em https://projects.raspberrypi.org/en/projects/getting-started-with-picamera

7 Ativar Relé. Disponível em https://blog.usinainfo.

com.br/utilizando-o-raspberry-pi-3-no-acionamentode-lampadaspronto-post-de-automacao-residencial 8 Rosebrock, Adrian. Practical Python and OpenCV, 3rd Edition. 8.

8. ANEXOS

8.1. Câmera

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
camera.start-preview()
sleep(10)
camera.stop-preview()
```

8.2. Qr Code

```
#!/usr/bin/python3
#qrcodeGUI.py
import tkinter as TK
from tkinter import messagebox
import subprocess
import cameraGUI as camGUI

class SET(camGUI.SET):
  QR_SIZE=(640,480)
  READ_QR="zbarimg "

class cameraGUI(camGUI.cameraGUI):
  def run_p(cmd):
    print("RunP:"+cmd)
    proc=subprocess.Popen(cmd,shell=True,stdout=subprocess.PIPE)
    result=""
    for line in proc.stdout:
      result=str(line,"utf-8")
    return result
  def __init__(self,parent):
    super(cameraGUI,self).__init__(parent)
    self.parent=parent
    TK.Frame.__init__(self,self.parent,background="white")
    self.qrScan=TK.IntVar()
    self.qrRead=TK.IntVar()
    self.qrStream=TK.IntVar()
    self.resultQR=TK.StringVar()
    self.btnQrTxt=TK.StringVar()
    self.btnQrTxt.set("QR GO!")
    self.QRBtn=TK.Button(self.parent,textvariable=self.btnQrTxt,
                                              command=self.qrGet)
    readChk=TK.Checkbutton(self.parent,text="Read",
                           variable=self.qrRead)
    streamChk=TK.Checkbutton(self.parent,text="Stream",
                             variable=self.qrStream)
    labelQR=TK.Label(self.parent,textvariable=self.resultQR)
    readChk.grid(row=3,column=0)
    streamChk.grid(row=3,column=1)
    self.QRBtn.grid(row=3,column=3)
    labelQR.grid(row=4,columnspan=4)
```

Fig. 9. Código Leitor QR Code Parte 1.

```
    self.scan=False
  def qrGet(self):
    if (self.scan==True):
      self.btnQrTxt.set("QR GO!")
      self.btnState("active")
      self.scan=False
    else:
      self.msg("Get QR Code")
      self.btnQrTxt.set("STOP")
      self.btnState("disabled")
      self.scan=True
      self.qrScanner()
  def qrScanner(self):
    found=False
    while self.scan==True:
      self.resultQR.set("Taking image...")
      self.update()
      cameraGUI.camCapture(SET.PREVIEW_FILE,SET.QR_SIZE)
      self.resultQR.set("Scanning for QRCode...")
      self.update()
      #check for QR code in image
      qrcode=cameraGUI.run_p(SET.READ_QR+SET.PREVIEW_FILE)
      if len(qrcode)>0:
        self.msg("Got barcode: %s"%qrcode)
        qrcode=qrcode.strip("QR-Code:").strip('\n')
        self.resultQR.set(qrcode)
        self.scan=False
        found=True
      else:
        self.resultQR.set("No QRCode Found")
    if found:
      self.qrAction(qrcode)
      self.btnState("active")
      self.btnQrTxt.set("QR GO!")
    self.update()
  def qrAction(self,qrcode):
    if self.qrRead.get() == 1:
      self.msg("Read:"+qrcode)
      cameraGUI.run("sudo flite -t '"+qrcode+"'")
    if self.qrStream.get() == 1:
      self.msg("Stream:"+qrcode)
      cameraGUI.run("omxplayer '"+qrcode+"'")
    if self.qrRead.get() == 0 and self.qrStream.get() == 0:
      TK.messagebox.showinfo("QR Code",self.resultQR.get())
#End
```

Fig. 10. Código Leitor QR Code Parte 2.

8.3. Módulo Relé

```
import RPi.GPIO as GPIO
 GPIO.setmode(GPIO.BOARD)
GPIO.setup(12, GPIO.OUT)
while (True):
GPIO.output(12,0)
sleep(2)
GPIO.output(12,1)
sleep(2)
GPIO.setwarnings(False)
```