Solução do Desafio HW/SW

Alex Borges dos Reis

Introdução

Esse documento tem como objetivo expor e esclarecer os caminhos escolhidos, decisões tomadas e conhecimentos adquiridos durante o processo de solução do desafio.

O problema consistia em estabelecer comunicação entre uma Raspberry Pi Zero W e um servidor, e quando um botão conectado na Raspberry fosse pressionado seu estado deveria ser enviado para servidor que o retornaria, alterando o estado de um Led, também ligado na Raspberry.

A fim de solucionar esse desafio foram feitas pesquisas na Internet, olhando em sites e vídeos, cujos links podem ser encontrados no trecho de Referências ao final do texto. Através deles foi possível entender como é a Raspberry e o funcionamento dos seus pinos GPIO, no intuito de usá-los para fazer as ligações do botão e do Led.

Baseando-se nessas pesquisas foi tomado como pressuposto, para desenvolver o código, que a comunicação entre cliente/servidor seria via socket.

Vale ressaltar que não há confirmação se tal código irá alcançar êxito, devido não somente ao meu conhecimento superficial no assunto de interações com servidores, mas principalmente a falta de meios materiais para realizar testes empíricos.

Desenvolvimento

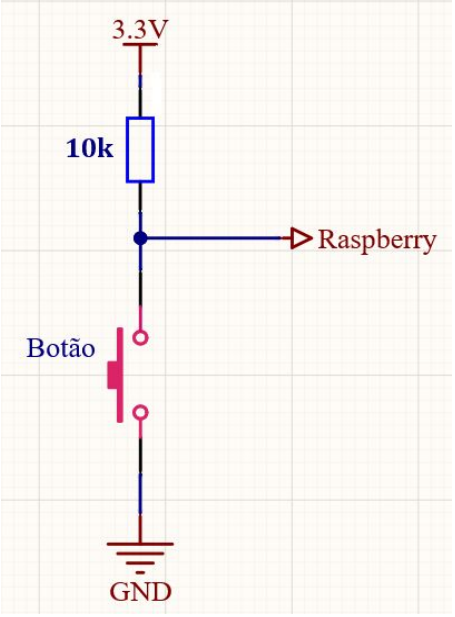
Primeiramente foi averiguado que era necessário fazer alguns ajustes, seguindo certos passos que serão mostrados abaixo, para conectar a Raspberry à Internet e, consequentemente, poder programar nela.

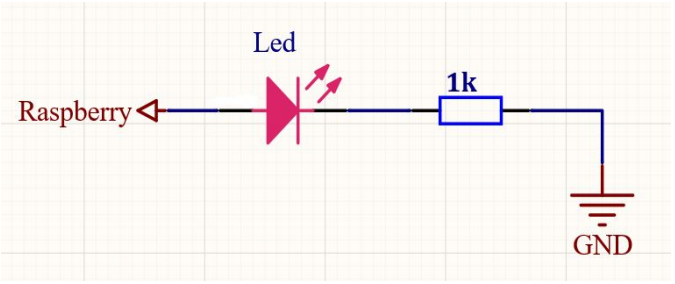
1. Deve-se conectar o cartão de memória da Raspberry ao computador, e abrir a unidade disco correspondente.
2. Lá dentro é necessário criar um arquivo de texto com nome “ssh” e apagar sua extensão .txt, assim a Raspberry estará apta a aceitar requisições usado o protocolo SSH.
3. Após isso, deve-se criar um outro arquivo com nome wpa_supplicant.conf, sendo .conf a nova extensão.
4. Abra o arquivo com um editor de texto e escreva da seguinte forma:
   ```
   network={
       ssid="NomeDaRede"
       psk="SenhaDaRede"
   }
   ```
5. Nos campos entre as aspas coloque o nome da rede Wi-Fi na qual quer se conectar e a senha, salve e desconecte o cartão.
6. Ao ligar a Raspberry ela se conectará automaticamente à rede, logo é preciso identificar o IP assumido por ela. Isso é possível através de softwares que fazem

varreduras da rede, os quais permitem distinguir os dispositivos conectados, e assim achar o IP da Raspberry.

7. Com o endereço IP em mãos agora é preciso estabelecer um acesso via SSH. Para isso pode-se usar também softwares específicos.
8. Quando isso for feito, basta inserir o usuário e senha padrões da Raspberry e, finalmente, o processo estará finalizado.

Em seguida foi trabalhado a parte do Hardware, desenvolvendo os circuitos do botão e do Led. As imagens abaixo mostram como devem ser as montagens de ambos.





O circuito à esquerda consiste em um botão normalmente aberto e um resistor de 10kΩ que serve de pull-up. Assim a porta GPIO em que estará conectado receberá nível lógico alto, a menos que o botão seja pressionado. Vale observar que é possível utilizar o resistor de pull-up interno da Raspberry via programação, dispensando assim o resistor convencional.

O circuito à direita consiste em um Led que indicará quando o servidor responder ao aperto do botão, e um resistor de 1kΩ para limitar a corrente.

E por último foi feito um estudo acerca de como configurar e estabelecer uma comunicação cliente/servidor via socket. Os sockets são recursos os quais permitem a comunicação entre máquinas diferentes, ou programas distintos em uma mesma máquina.

Com esses passos realizados foi possível pensar e elaborar um código com a finalidade requerida pelo desafio. Segue abaixo a imagem do programa:

```
1   #Importação dos módulos necessários
2   import RPi.GPIO as GPIO
3   import socket
4
5   GPIO.setmode(GPIO.BOARD) #Numeração dos pinos GPIO de acordo com a placa
6   GPIO.setwarnings(False)  #Desativação dos avisos dos GPIOs
7   led = 7       #Configurando pino do Led
8   botao = 12    #Configurando pino do botão
9   GPIO.setup(led,GPIO.OUT)  #Setando o Led como saída
10  GPIO.setup(botao,GPIO.IN) #Setando o botão como entrada
11
12  #Declarações de variáveis
13  int ult_est_bot = 1
14  int estado_botao
15  int estado_led = 0
16
```

```
17  #Configurando o host e a porta para comunicação
18  host = '192.168.0.110'
19  porta = 8000
20  sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #Criando o socket
21  sock.connect((host, porta))  #Estabelendo a conexão com servidor
22
23  while True:   #Loop infinito
24      estado_botao = GPIO.input(botao) #Leitura do estado do botão
25      if estado_botao != ult_est_bot:  #Averiguando se houve mudança no botão
26          sock.sendall(str.encode('change')) #Sinalizando p/ servidor
27          ult_est_bot = estado_botao
28          resp = sock.recv(100)  #Recebendo a resposta
29          if resp == 'change':  #Caso seja igual a mandada, altera estado do Led
30              if estado_led == 0:
31                  GPIO.output(led,GPIO.HIGH)
32                  estado_led = 1
33              else:
34                  GPIO.output(led,GPIO.LOW)
35                  estado_led = 0
36
```

Um detalhamento mais minucioso do código linha a linha:

Linhas 2 e 3 → Importação dos módulos necessários ao programa, sendo o de utilização das portas GPIO e do socket, respectivamente.

Linha 5 → Selecionando a opção para referir-se aos pinos GPIO de acordo com a numeração impressos na placa. Há também a opção de se referir a eles de acordo com o Broadcom SOC Channel, que varia dependendo da versão da Raspberry.

Linha 6 → O comando dessa linha desativa os avisos de segurança caso haja mais de um programa utilizando o mesmo pino GPIO.

Linhas 7 e 8 → Configurando que o pino do Led será o de número 7, e do botão o de número 12.

Linhas 9 e 10 → Estabelecendo o pino do Led como saída, e o do botão como entrada.

Linhas 13 a 15 → Declarando as variáveis que serão utilizadas no programa. A primeira variável guarda o último estado do botão, começando com o valor 1, equivalente a

HIGH. A segunda guarda o valor atual da leitura do botão. E a terceira armazena o estado do Led.

Linha 18 → Colocando o IP do host de acordo com o que foi mostrado no documento do desafio.

Linha 19 → Estabelecendo uma porta para comunicação. Tal número foi escolhido de forma aleatória, mas de preferência que seja um valor alto para não interferir em outras comunicações preexistentes.

Linha 20 → Esse comando cria o socket com certas características. O parâmetro AF_INET estabelece a utilização de endereços IPv4. E o parâmetro SOCK_STREAM estabelece a utilização do protocolo TCP na camada de transporte.

Linha 21 → Nessa linha é estabelecida a comunicação entre o cliente e o servidor, passando o endereço do servidor e a porta.

Linha 23 → Aqui é criado um While com a condição True, o que fará com que ele se torne um loop infinito, sempre realizando as mesmas operações.

Linha 24 → O estado do botão é lido e armazenado na variável “estado_botao”.

Linha 25 → Tem-se aqui uma estrutura de decisão if, caso o estado lido do botão na linha anterior for diferente do último estado assumido por ele, as próximas linhas serão executadas, senão ele voltará ao início do loop.

Linha 26 → Com a confirmação da mudança do estado, um sinal que consiste na palavra "change", é emitido para o servidor através do comando sendall.

Linha 27 → Agora a variável ult_est_botao é atualizada com o estado lido do botão.

Linha 28 → Aqui a variável “resp” armazena a informação recebida pelo socket. O valor 100 indica que ele receberá até 100 bytes.

Linha 29 → Caso a informação recebida seja o sinal mandado para o servidor as próximas linhas, que se referem ao Led, serão executadas.

Linhas 30 a 32 → Se o Led estiver desligado essa parte da estrutura será executada, fazendo com que o Led acenda e armazenando esse estado na variável “estado_led”.

Linhas 33 a 35 → E se o Led estiver ativado as linhas dentro do else serão executadas, desligando o Led e armazenando seu estado na variável.

Referências

CURVELLO, ANDRÉ. **Raspberry Pi Zero W: configuração rápida para Rede e SSH,** 2017. Disponível em: https://www.filipeflop.com/blog/raspberry-pi-zero-w-para-rede-e-ssh/. Acesso em 08 Jan. 2021.

Raspberry Pi: Como acender um LED? , 2015. Disponível em: https://pplware.sapo.pt/tutoriais/raspberry-pi-como-acender-um-led/. Acesso em 09 Jan. 2021.

What is the difference between BOARD and BCM for GPIO pin numbering?, 2017. Disponível em: https://raspberrypi.stackexchange.com/questions/12966/what-is-the-difference-between-board-and-bcm-for-gpio-pin-numbering. Acesso em 09 Jan. 2021.

VICENTE, ELDER. **Introdução a Sockets em Python,** 2019. Disponível em: https://medium.com/@urapython.community/introdu%C3%A7%C3%A3o-a-sockets-em-python-44d3d55c60d0. Acesso em: 10 Jan. 2021.