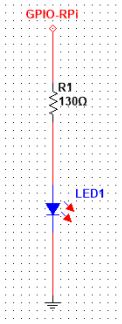
Desafio

Nome: Helio Abreu Marques Rocha Curso: GES Matrícula: 79

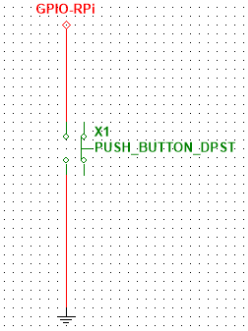
Primeiramente, eu desenvolveria o hardware.

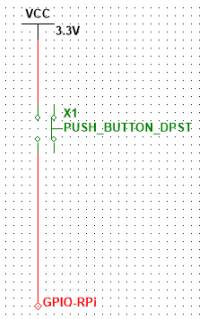
LED - Considerando uma corrente de 10mA e uma tensão de 2V para o funcionamento do LED, o circuito ficaria como o da imagem.



Botão - Para o circuiro do botão pode-se simplesmente ligar um dos terminais do botão no GPIO da Raspberry pi, e o outro no Vcc (Pulldown) ou GND(Pullup), e configurar qual dos modos será utilizado no próprio script.

Pull-up Pull-down





Configuração do SSH - Pode-se interagir com a Raspberry utilizando-se os periféricos básicos de um computador (teclado, mouse e monitor), mas comumente se utiliza o protocolo SSH para acessá-la de outro computador. Assim, considerando-se a utilização do SSH, deve-se criar um arquivo em branco, sem extensão e com o nome "SSH" no cartão de memória da Raspberry e asssim ela estará pronta para receber comandos de um terminal SSH.

Configuração do Wifi - Se estiver utilizando o SSH para acessar a raspberry, já é aconselhável criar um arquivo chamado wpa_supplicant.conf com as seguinte linhas dentro.

```
network={
    ssid="NomeDaRede"
    psk="SenhaDaRede"
}
```

(Caso estiver utilizando um monitor, o Wifi pode ser facilmente configurado através da interface da Raspberry).

Teste do HW utilizando script teste - Com a Raspberry configurada e o hardware montado, pode-se então criar um programa teste básico para testar o botão e o LED de forma isolado do resto do projeto. Quando o botão for pressionado, o LED acende e quando ele não for pressionado o LED fica apagado.

Este script pode ser facilmente desenvolvido utilizando a biblioteca RPi.GPIO com alguns comandos e estruturas básicas. Abaixo está o script desenvolvido utilizando o polling por questões de demonstração, mas para uma aplicação prática pode ser considerado a utilização de uma interrupção para maior eficiência do projeto.

```
/-----------------------------------------------------------------------------------------------------------------------------------------------------------------------

import RPi.GPIO as gpio

# BOT = pino do botão
# LED = pino do LED

gpio.setmode(gpio.BCM)   #Pode-se utilizar também gpio.BOARD o que alteraria somente a nomenclatura dos pinos

gpio.setup(BOT,gpio.IN,pull_up_down = gpio.PUD_UP) #Configura o botão como entrada
# Para utilizar pull down trocar gpio.PUD_UP por gpio.PUD_UP

gpio.setup(LED,gpio.OUT) #Configura o LED como saida



while True: #loop infinito
        if gpio.input(BOT) == gpio.LOW: # Se o estado do botão for nivel lógico baixo (botão pressionado no caso de pull-up)
                gpio.output(LED,1) # Liga o LED
        else: # Senão
                gpio.output(LED,0) # Desliga o LED

/-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Teste da conexão RPi-Servidor - Estando então conectado à rede, é necessário fazer a requisição ao servidor. Para isso, é possível utilizar a biblioteca “Requests”. Pode-se, utilizando o método GET, enviar a informação do estado do botão e assim o servidor retornará o estado do LED. Então, basta receber a informação do estado do LED e refleti-lo no LED.

Segue abaixo um código que pode ser utilizado:

```
/-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
import RPi.GPIO as gpio
import requests

# BOT = pino do botão
# LED = pino do LED

url = '192.168.0.110/get?state='


gpio.setmode(gpio.BCM)   #Pode-se utilizar também gpio.BOARD o que alteraria somente a nomenclatura dos pinos

gpio.setup(BOT,gpio.IN,pull_up_down = gpio.PUD_UP) #Configura o botão como entrada
# Para utilizar pull down trocar gpio.PUD_UP por gpio.PUD_UP

gpio.setup(LED,gpio.OUT) #Configura o LED como saida


while True: #loop infinito
        r = requests.get(url+gpio.input(BOT)) #Manda o get para o servidor
        state = int(r.text) # O estado do LED é igual à resposta do servidor.
        gpio.output(LED,state) # Muda o estado do LED

/-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

O código não pôde ser testado, então pode ser que na prática ocorram erros, mas a lógica é a seguinte:
-Importa as bibliotecas
-Configura os pinos e a url
-Manda a url concatenada com o valor do botão
-Recebe a resposta do servidor e reflete o estado no LED

Obs:
Como não houveram mais informações foram feitas várias considerações como:

-O servidor manda somente a informação do estado do LED
-O servidor usa a mesma codificação que é utilizada pelo r.text para decodificar
-Somente é necessário enviar o estado do botão para o servidor

Se essas considerações não se aplicarem à situação prática, devem ser realizados alguns ajustes como:
- “Limpeza” da resposta para encontrar a informação desejada
- Configurar o tipo de codificação utilizada
- Enviar as outras informações requeridas pelo servidor como por exemplo um token de autenticação.

Boas práticas, identificação de erros e notas adicionais

- É aconselhável definir os IPs do servidor e da Raspberry como estáticos para evitar problemas de troca de IP.
- Caso no teste do hardware o led e/ou o botão não funcionarem, pode-se testá-los utilizando uma fonte de 3.3V para ver se o problema está nos pinos e/ou no script
- Caso a parte da comunicação com o servidor estiver dando problemas, pode-se isolá-la do resto do projeto e analisar a resposta. Se ainda persistirem os problemas, pode-se ainda utilizar de ferramentas que permitem a visualização de todo o pacote de requisição e resposta.
- Dependendo da aplicação pode-se ainda programar a Raspberry Pi para executar o script assim que a Raspberry for ligada.