Instituto Tecnológico Autónomo de México

Departamento Académico de Sistemas Digitales

*Nuevas Tecnologías Inalámbricas*

***Práctica 6***

***Linux embedido (Raspberry Pi)***

Integrantes:

Alfonso Venancio- 149211

Efraín Aguilar- 149643

Leandro Pantoja- 150883

Ulises Alejandre - 159235

26 de febrero de 2019

**Introducción**

Un sistema con Linux embebido se refiere a un aparato o software que funciona con el núcleo de Linux. Android y placas de desarrollo como la Raspberry Pi son ejemplos de sistemas con Linux embebido.

Una Raspberry Pi es una computadora de placa de bajo costo, producida en el Reino Unido, que tiene el propósito de enseñar a los niños programación de computadoras. En la práctica se usó la versión 3, la cual tiene un procesador ARM de 4 núcleos, 1 GB de memoria RAM, 40 pines GPIO, usa una tarjeta microSD como medio de almacenamiento y tiene 4 puertos USB y HDMI.

**Desarrollo**

Primero se conectó la Raspberry Pi a un monitor usando un adaptador VGA-HDMI, al igual que se conectó un teclado y un mouse para poder utilizarla. Primero se configuró el idioma, país y teclado a los valores correspondientes para México; luego se activaron los servidores ssh y vnc para poder usar la tarjeta remotamente desde otra computadora sin tener que dejar la Raspberry Pi conectada a un monitor.

El objetivo de la práctica fue reproducir el experimento del siguiente video <https://www.youtube.com/watch?v=mOaIZmyFuqY>, el cual consiste en controlar el parpadeo de un led usando un botón. Una vez hecho eso, tenía que modificar el código para que cuando se presione el botón, se envíe un mensaje a un servidor de eco hecho en python para que se alerte al usuario. Los dos ejercicios anteriores se tenían que realizar remotamente mediante ssh y vnc.

**Resultados**

Para la primera parte de la practica, se armo un circuito con un led y un push button, los cuales se conectaron los pines de la Raspberry Pi. Cabe mencionar que en este circuito se puso una resistencia para evitar que el led se quemara.

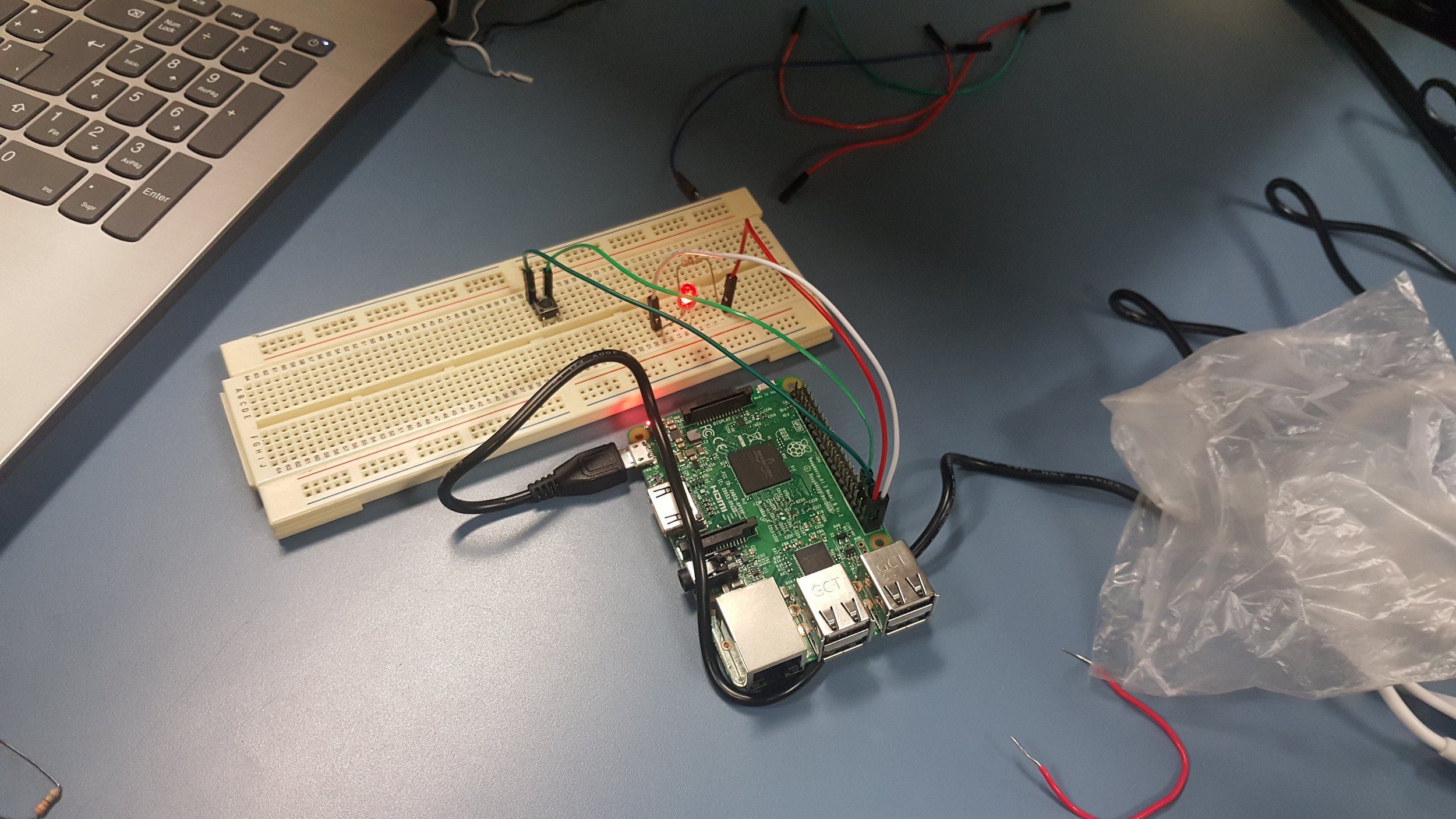


Figura 1: circuito led funcionando con Raspberry Pi

Posteriormente se agregó en el código de python un ciclo para monitorear el estado del botón, y en caso de que se presionara mandará el mensaje de “Pulsador está pulsado”.

while True:  
 **if GPIO.input(btn) == False:  
 print ("Pulsador está pulsado")**  
 tiempo = 0.1   
 else:   
 tiempo = 1  
 time.sleep(tiempo) #Esperamos un tiempo para que se vea el parpadeo del LED

al pulsarse el botón y led parpadea más rápido. En la consola de Raspberry se observa el mensaje si la entrada GPIO detecta la señal del botón pulsado.

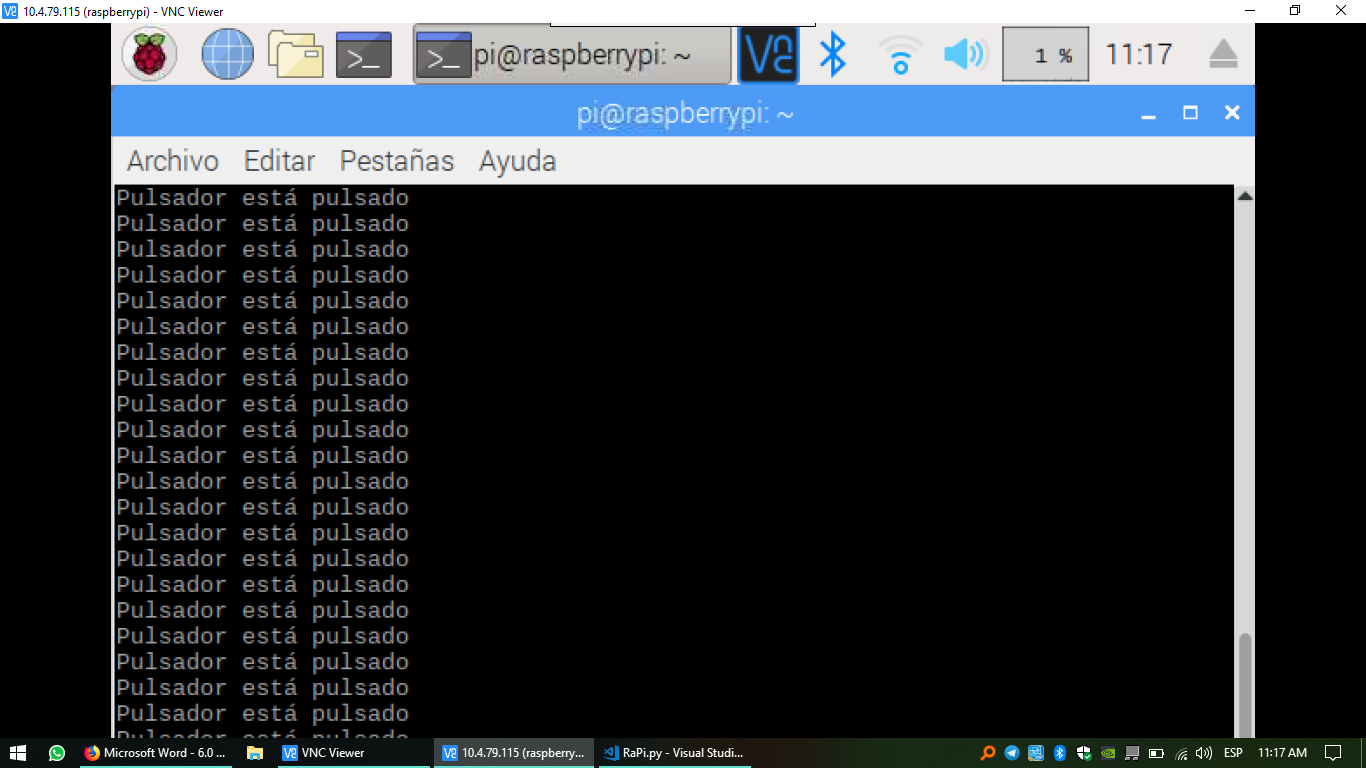


Figura 2: Ejemplo mensaje consola Raspberry cuándo botón está pulsado

Para la última parte de esta práctica se hizo un program servidor en python que mandara un mensaje , esto se logró gracias al uso de una conexión socket.

El método listen() se usó para levantar un servidor de eco e imprimir el mensaje en la consola. Se usó una secuencia de else-if para tener control del programa y saber cuándo salirse.

def listen():  
 connection = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
 connection.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)  
 connection.bind(('localhost', 5555))  
 connection.listen(10)  
 while True:  
 current\_connection, address = connection.accept()  
 while True:  
 data = current\_connection.recv(2048)  
 if data == 'quit\r\n' or data == 'salir':  
 current\_connection.shutdown(1)  
 current\_connection.close()  
 break  
  
 elif data == 'stop\r\n' or data == 'parar':  
 current\_connection.shutdown(1)  
 current\_connection.close()  
 exit()  
  
 elif data:  
 current\_connection.send(data)  
 print (data)

Se usó la función publish para mandar el mensaje del programa al servidor de eco. El argumento de la función es el mensaje y se le aplica el método encode para que el servidor lo interprete como un string y no como bytes.

def publish(msg=None):  
 sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
 server\_address = ('localhost',5555)  
 sock.connect(server\_address)  
 if msg != None:  
 msg = msg.encode()  
 print('Enviando mensaje ...')  
 sock.sendall(msg)  
 print('... Saliendo del programa')  
 sock.close()  
 #sock.shutdown(1)  
 else:  
 while True:  
 print('Escriba salir para terminar el programa')  
 msg = input('Escriba su mensaje:\n')  
 if msg == 'salir':  
 print('... Saliendo del programa')  
 sock.close()  
 #sock.shutdown(1)  
 break  
 msg = msg.encode()  
 sock.sendall(msg)

**Conclusión**

En esta práctica aprendimos a usar de manera básica la tarjeta Raspberry Pie usando un ejemplo con un led y un botón. Aprendimos cómo están organizados los GPIO de la tarjeta así como la funcionalidad. Por último experimentamos el uso de un programa servidor el cual mostraba un mensaje de evento al usuario y vimos la utilidad de una conexión socket.

**Referencias**

* <https://www.elinux.org/RPi_Hub>
* <https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/>
* <https://www.youtube.com/watch?v=mOaIZmyFuqY>