

UNIVERSIDAD DE LAS FUERZAS
ARMADAS ESPE

ESPE



HERRAMIENTAS DE SOFTWARE PARA LA INGENIERIA

Creación de un chat serial en python

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

Docente: ING. Darwin Omar Alulema Flores

Nombre: Jhony Borja, Kevin Casagallo, David Campoverde

NRC: 5035

SANGOLQUI 08 DE JULIO DEL 2019

1 Introducción

GNU Linux es un sistema operativo libre tipo Unix POSIX; multiplataforma, multiusuario y multitarea. El sistema es la combinación de varios proyectos, entre los cuales destacan GNU (encabezado por Richard Stallman y la Free Software Foundation) y el núcleo Linux (encabezado por Linus Torvalds). Su desarrollo es uno de los ejemplos más prominentes de software libre: todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera, bajo los términos de la GPL (Licencia Pública General de GNU) y otra serie de licencias libres.

La comunicación serial es de implementación muy sencilla desde lo computacional, y nos permite enviar y recibir cualquier información que necesitemos entre puertos seriales de una computadora y otro dispositivo, de forma que pueda ser visualizada empleando nuestro emulador de terminal.

2 Índice

1. Planteamiento del problema
2. Objetivos
3. Estado del arte
4. Marco teórico
 - (a) Definición
 - (b) Clases
 - (c) Herencias
5. Diagramas
6. Lista de componentes
7. Mapa de variables
8. Explicación del código fuente
9. Descripción de prerequisites y configuraciones
10. Conclusiones
11. Recomendaciones
12. Bibliografía
13. Anexos
14. Manual de usuario
15. Hojas técnicas

3 Planteamiento del problema

La comunicación serie o comunicación secuencial, es el proceso de envío de datos de un bit a la vez, de forma secuencial, sobre un canal de comunicación o un bus, en el presente proyecto se va a desarrollar y aportar información sobre la creación de un chat en Python utilizando la comunicación serial mediante la instalación de paquetes en el sistema operativo Ubuntu y Raspbian, que pertenecen a Linux dando a conocer ¿Cómo podemos usar el sistema operativo?

- ¿Cómo lograr instalar en la consola de Linux los diferentes paquetes?
- ¿Cómo conectar de forma adecuada los cables puente entre Módulo Adaptador USB a Serie TTL RS232 – USB y la PC?
- ¿De que manera vamos a conseguir la comunicación entre la PC y la Raspberry Pi mediante

4 Objetivos

El objetivo general del proyecto es lograr implementar una comunicación serial entre una PC con Ubuntu y una Raspberry Pi con Raspbian para lograr obtener un chat serial en Python.

A continuación se presentan los objetivos específicos que se pretenden alcanzar:

- Observar y estudiar el funcionamiento del lenguaje python en el desarrollo de este proyecto
- Investigar sobre los diferentes paquetes que se necesitan tanto en Ubuntu para la PC como en Raspbian en la Raspberry Pi.
- Conocer de que manera podemos lograr conectar los cables puente ya que existen pines específicos que deben ser correctamente identificados para así lograr una correcta comunicación serial.

5 Estado del arte

Wakerly, J. (1981). Comunicación serial. Microprocesadores y microsistemas, 5 (6), 247–253. doi: 10.1016 / 0141-9331 (81) 90582-2

La comunicación en serie significa que la información transmitida desde el origen hasta el destino se transporta por una sola vía. Dentro de los confines físicos inmediatos de una computadora. En general, el uso de enlaces seriales suele estar motivado por Consideraciones de costos: los enlaces seriales pueden reducir el empaquetado y el costo de cableado y reducir el número y la complejidad de componentes para enviar y recibir datos. Fuera de El sistema informático, uso de un enlace de comunicación serial. A menudo se ve forzado por la naturaleza misma de los datos disponibles. Medios de transmisión - líneas telefónicas y ondas de radio. Solo se puede enviar una señal analógica a la vez.

Un enlace de datos en serie simplex transmite datos en una sola dirección. (Una conexión de una computadora a un control remoto la impresora podría ser un enlace de datos serie simplex.) En la mayoría aplicaciones, hay dos enlaces de datos en serie entre una computadora y un terminal, uno para la transmisión en cada dirección.

6 Marco teorico

6.1 Definicion

La información en una cadena serial de bits esta contenida en su forma de onda dependiente del tiempo: los bits se representan por códigos que se transmiten por un periodo de tiempo fijo. El periodo de tiempo usado para transmitir cada código se conoce como periodo baud.

El mundo de las comunicaciones internas del computador se realiza en forma paralela alternada, por fuera del computador predominan las comunicaciones seriales; las redes de computadores se basan en dicha comunicación.

El PC utiliza la norma RS232, por lo que los niveles de tensión de los pines están comprendidos entre +15 y -15 voltios.

La información que maneja un computador puede transmitirse de un lugar a otro en dos formas básicas, en forma serial o en forma paralela. En una transmisión serial se forma un "tren" de bits, uno tras de otro viajan del lugar de emisión al receptor utilizando una sola vía, en este caso será un conductor eléctrico bus Serial, como en caso de los trenes con una sola vía si se desea transmitir en el sentido contrario, se debe esperar que la vía este libre. En la comunicación en paralelo cada bit tiene su vía exclusiva, con la condición de que todos viajen simultáneamente, como en el caso de la comunicación serial para transmitir en el sentido contrario se debe esperar que la vía este libre, a menos que se tenga una exclusiva para el sentido contrario. La comunicación en el interior del computador son básicamente transmisiones en paralelo utilizando una sola vía, en caso de que se transmita en los dos sentidos debe ser alternada por cuanto solo se cuenta con una sola vía.

Las cadenas seriales de bits generadas por los puertos serie de la PC usan una forma muy simple de codificación. Un bit se transmite durante cada periodo baud, con un bit "1" representado por un voltaje alto TTL y un "0" por un voltaje bajo TTL. Así la velocidad en baudios (baud rate, $1/[\text{periodo baud}]$) de un puerto serie de la PC es igual al número de bits por segundo que se transmiten o reciben.

Para enviar información codificada de esta manera, el transmisor y receptor registran el tiempo, el cual define el periodo baud, deben estar a la misma frecuencia y estar sincronizados. Los bits se transmiten como grupos separados, con una longitud típica de 7 u 8 bits, llamados caracteres. El nombre caracter se usa porque cada grupo de bits representan una letra del alfabeto cuando el texto esta codificado en ASCII. Cada carácter se envía en una armazón (frame) consistiendo de un bit "0" llamado un bit de inicio, seguido por el caracter mismo, seguido (opcionalmente) por un bit de paridad, y después un bit "1" llamado bit de paro. La lógica del bit bajo de inicio le dice al receptor que esta empezando una armazón, y la lógica del bit alto de paro denota el final de la armazón.

6.1.1 Mòdul PL230 Conversor USB a TTL

Permet aquell aparell que permet que un microcontrolador i una PC es comuniquen utilitzant el protocol USB de forma senzilla. És compatible amb qualsevol microcontrolador com PIC, Atmel AVR, Arduino i ESP8266. Funciona de forma similar als convertidors FTDI232 i PL2303HX, amb la ventaja de tenir un millor preu i major suport de drivers. A més pot funcionar com "programador" de l'Arduino Mini Pro, ja que inclou el pin DTR o RESET necessari per carregar fàcilment el sketch a l'Arduino Mini Pro. Els dispositius UART TTL són compatibles amb RS232 només des de la perspectiva del software, ja que ambdues interfícies transmeten amb la metodologia d'un bit per vegada (serialment), a una velocitat de baudios específica, amb/sin bits de paritat i amb/sin bits de parada. No obstant això, ambdues interfícies difereixen fonamentalment a nivell hardware (elèctric). La comunicació serial UART TTL transmetrà amb baixos voltatges continus, operant entre els límits de 0 volts i el Voltatge d'alimentació Vcc (el qual pot prendre el valor de 3,3 volts o de 5 volts). La seva lògica elèctrica també és opuesta a RS-232 ja que el valor actiu ("1") queda representat pel pic de tensió continu, mentre que la baixa de la lògica ("0") és 0 volts. Això fa que no puguem enllaçar directament un microcontrolador UART TTL a un port serial COMM de la nostra computadora ja que les interfícies a nivell hardware són incompatibles.

Per establir comunicació hem d'utilitzar un adaptador corresponent, sent els més senzills i fiables els de tipus USB a UART TTL

6.1.2 Paquet Pip

Pip és un sistema de gestió de paquets utilitzat per instal·lar i administrar paquets de software escrits en Python. Molts paquets poden ser trobats a l'Índex de Paquets de Python (PyPI). Python 2.7.9 i posteriors (en la sèrie Python2), Python 3.4 i posteriors inclouen pip (pip3 per Python3) per defecte. pip és un acrònim recursiu que es pot interpretar com Pip Instal·lador de Paquets o Pip Instal·lador de Python. Una ventaja important de pip és la facilitat de interfície de línia de comandaments: el seu acrònim recursiu el qual permet instal·lar paquets de software de Python fàcilment des de solament una ordre permet gestionar llistes de paquets i els seus nombres de versió corresponents a través d'un arxiu de requisits. Això ens permet una recreació eficaç d'un conjunt de paquets en un entorn separat (p. ex. un altre ordinador) o entorn virtual. Això es pot aconseguir amb un arxiu correctament format. Amb pip és possible instal·lar un paquet per una versió concreta de Python, només és necessari reemplaçar per la versió de Python que volem: 2, 3, 3.4, etc.

La Raspberry Pi 3 se li inclou un bluetooth dedicat sense necessitat de tenir una sola UART, ja que n'hi ha dues; una per al bluetooth i una altra per als GPIO. Aquesta modificació implica que els pins GPIO ja no estan direccionats a ttyAMA0 com ocorre en les versions 1 i 2 de Raspberry, ara passa la interfície UART a la direcció ttyS0, per tant s'ha de fer el canvi corresponent en el fitxer de configuració del sistema operatiu de la Raspberry, en aquest article t'indicarem com fer la respectiva configuració per treballar des de la UART.

6.1.3 Paquete Pyserial

PySerial es una librería de Python que permite comunicarse a través de comunicaciones por serial (RS-232).

Esto puede ser muy útil para mandar o recibir datos de periféricos que sean comunes o que tu mismo hayas hecho de una manera increíblemente sencilla y sin tener que complicarse para nada con este tipo de programación.

7 Diagramas

8 Lista de componentes

- Laptop que contenga el sistema operativo Ubuntu
- Tener instalado Python, los paquetes pip y debemos tener agregada la libreria Pyserial.
-

9 Mapa de variables

10 Explicacion del codigo fuente

```
#llamo a las librerias usadas en el programa\
import serial
import sys
import time
#Bienvenida

nombre=str(input('\n Bienvenido al programa SOPHA2.0 por favor ingresar su nombre: '))
print("\n Sea bienvenido ", nombre, "empecemos\n ")

#Menu de ingreso

print ("-----Programa de comunicacion entre dos computadoras por medio USB-----\n")
print (" 1.- Determinar Que Puertos estan disponibles \n")
print (" 2.- Ser El Pc Emisor \n")
print (" 3.- Ser El Pc Receptor \n")
print (" 4.- Configuracion del puerto serie \n")
print (" ----- \n")

# Comparador si el valor ingresado esta dentro de 1-3

while True:
    menu = input("\n Por favor escoja opciones del 1 al 4: \t")
    try:
        menu = int(menu)
        if menu<1 or menu>4:
            print ("No Ingreso un valor en el rango:")
            raise ValueError("No Ingreso un valor en el rango:")
```

```
while True:
    menu = input("\n Por favor escoja opciones del 1 al 4: \t")
    try:
        menu = int(menu)
        if menu<1 or menu>4:
            print ("No Ingreso un valor en el rango:")
            raise ValueError("No Ingreso un valor en el rango:")
    except ValueError:
        print ("ERROR!")
        print ("-----")
    else:
        break
if menu==1:
    disponibles = []

    for i in range(10):
        try:
            s = serial.Serial(i)
            disponibles.append(s.portstr)
            s.close()
        except: pass

    for usb in range(10):
        try:
            dato="/dev/ttyUSB"+str(usb)
            s = serial.Serial(dato)
            disponibles.append(s.portstr)
            s.close()
        except: pass

    print("\n Puertos serials disponibles Para su Uso son: ")
```

```

File Edit Format Run Options Window Help
    disponibles.append(s.portstr)
    s.close()
except: pass

print(" \n Puertos seriales disponibles Para su Uso son: ")

for nombres in range(len(disponibles)):
    print(disponibles[nombres])

print ("Gracias por usar nuestro programa",nombre,"Hasta la proxima")

if menu==2:
    p=input(" Elija el puerto que desea para ser Emisor (RECUERDE QUE SOLO TIENE LOS PUERTOS DISPONIBLES QUE SE ENLISTARON EN LA OPCION 1 DEL MENU)\t")
    print("\n---- CP Emisora ----")
    print("\n Para finalizar el envio escribe Fin.\n")
    while True:
        try:
            ser=serial.Serial("/dev/ttyUSB"+str(p))
            mensaje=input(" Ingrese su mensaje : ")
            finalizar=mensaje
            mensaje=mensaje+"\n"
            ser.write(mensaje.encode())
            print(" Mensaje Enviado con exito ","\n")
            if finalizar=="fin":
                ser.close()
                print(" Modo Emisor cerrador")
                time.sleep(1)
                print ("Gracias por usar nuestro programa",nombre,"Hasta la proxima")
        except:
            print(" No se encontro el puerto #",p)
            break

```

Sophaz.py - C:\Users\ALEXANUKA\Downloads\sophaz.py (3.7.5)

```

File Edit Format Run Options Window Help
    print(" No se encontro el puerto #",p)
    break

if menu==3:
    p=input(" Elija el puerto que desea para ser Receptor (RECUERDE QUE SOLO TIENE LOS PUERTOS DISPONIBLES QUE SE ENLISTARON EN LA OPCION 1 DEL MENU)\t")
    print("\n---- CP Receptora ----")
    print("\n Para finalizar la recepcion escriba Fin..\n")
    while True:
        try:
            ser=serial.Serial("/dev/ttyUSB"+str(p))
            print("Recibiendo Datos...")
            llega=ser.readline()
            finl=llega.decode('UTF-8')
            if llega=="fin\n":
                ser.close()
                print("Conexion finalizada...")
                print("Cerrando el puerto...")
                time.sleep(1)
                print ("Gracias por usar nuestro programa",nombre,"Hasta la proxima")
                break
            print(finl)
        except serial.SerialException:
            ser.close()
            print("No se encontro el puerto #",p)
            time.sleep(2)
            break

if menu==4:
    try:
        print(" \n A que velocidad de Transferencia de datos desea cambiar \n ")
        print("300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200")
        t=input("Escriba a que Velocidad Desea cambiar los dos Puertos: ")
    except:
        print("No se encontro el puerto #",p)
        break

```

11 Descripción de prerequisites y configuraciones

- Instalar Python en la PC mediante comandos en Ubuntu
- Instalar los paquetes necesarios para la realización del chat serial en la PC como lo son el Paquete Pip y el Paquete PySerial.
- Instalar los paquetes necesarios para la realización del chat serial en la Raspberry Pi y configurarla correctamente mediante comandos en la consola de Linux.
- Configurar en la consola de linux de la Raspberry para que el linux no use la UART para login Shell.
- Conectar los cables puente de manera correcta así como se muestra en el diagrama:

12 Aportaciones

La comunicación serial entre una Raspberry Pi y una Laptop es una de las aportaciones ya que se utiliza Ubuntu y Raspbian, los dos pertenecen a linux pero su configuración en la consola de linux mediante línea de comandos es diferente ya que tienen diferentes librerías y paquetes que deben ser instalados en cada uno de los equipos que se utilicen.

13 Conclusiones

En conclusion la comunicación serial o la creación de un chat serial con una Raspberry Pi y una Laptop con Raspbian y Ubuntu respectivamente

14 Recomendaciones

15 Bibliografía

Bibliografia

- [1] <https://es.wikipedia.org/wiki/GNU/Linux>(INTRODUCCION)
- [2] <https://ubuntuperonista.blogspot.com/2017/09/como-me-conecto-traves-de-conexion-serial-ttl-ubuntu.html> (MARCO TEORICO)
- [3] <https://www.digitaltechinstitute.com/phyton-lenguaje-programacion/>(PLANTEAMIENTO DEL PROBLEMA)
- [4] <https://doi.org/10.1080/07474938.2011.55357>(ESTADO DEL ARTE)
- [5] <https://doi.org/10.1145/1595496.1562955>(ESTADO DEL ARTE)
- [6] <https://doi.org/10.1145/2157136.2157353>(ESTADO DEL ARTE)
- [7] <https://github.com/pickcas/producto-unidad-2/Anexos>
- [8] <https://www.overleaf.com/project/Anexos>

16 Anexos

16.1 Repositorios

<https://github.com/pickcas/producto-unidad-2>

<https://www.overleaf.com/project>

The image shows two screenshots of web interfaces. The top screenshot is a GitHub repository page for 'pickcas / producto-unidad-2'. It displays the repository's structure, including files like '.gitignore', 'Maquina_dispensadora_con_lenguaje_python (2).pdf', 'articulo.pdf', and 'maquina.py'. The bottom screenshot is an Overleaf project page. It shows a list of projects on the left sidebar, including 'plantillas (2)', 'Producto unidad segundo parcial (3)', 'PROYECTO (4)', 'Proyecto de investigación segundo parcial (5)', 'proyecto unidad (2)', 'purevas para mejorar los informes (3)', and 'Uncategorized (22)'. The main area displays a table of projects with columns for Title, Owner, Last Modified, and Actions.

Title	Owner	Last Modified	Actions
Maquina dispensadora con lenguaje python	You	5 hours ago by campoverded3	Share Download Delete
articulo	You	a day ago by You	Share Download Delete
baposter Template	campoverded3	14 days ago by campoverded3	Share Download Delete

The screenshot shows a LaTeX Beamer presentation titled "Progrmacion en Python de una Maquina dispensadora" by David Campoverde, Jhonny Borja, and Kevin Casagallo. The presentation is displayed in a Beamer viewer interface. The left sidebar shows a file explorer with files like 1.png, 2.png, 3.png, 4.png, 5.png, 6.png, 7.png, baposter.cls, baposter_guide.pdf, poster.tex, and README. The main content area is divided into sections: INTRODUCCION, DEFINICION, BASIC CONCEPTS, and CARACTERISTICAS. The INTRODUCCION section defines Python as a general-purpose, versatile, and popular programming language. The DEFINICION section explains that learning a programming language is important in the field of Systems or Technical Computing and Informatics. The BASIC CONCEPTS section states that Python is a general-purpose, versatile, and popular programming language. The CARACTERISTICAS section lists features like object-oriented, imperative, and functional programming paradigms. A code snippet for a class 'Producto' is shown in the DEFINICION section.

The screenshot shows a LaTeX Beamer presentation titled "Maquina dispensadora con lenguaje python". The left sidebar shows a file explorer with files like imagenes and main.tex. The main content area is divided into sections: 1 abstrac, 2 Introducción, and 3 Desarrollo. The 1 abstrac section discusses the process of designing, coding, and protecting source code. The 2 Introducción section defines Python as a general-purpose, versatile, and popular programming language. The 3 Desarrollo section lists the most common programming languages: C, C++, and Java.

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux

Maquina dispensadora

1. Papas sin marca : 0.5
2. Galletas : 0.45
3. Agua : 1.35
4. Oreo : 0.65
5. Ruffles : 0.25
6. Pringles : 2.35
7. Doritos : 0.25
8. Sandwich : 1.65
9. Manzana : 45
10. Yogurt toni : 0.5
11. mmm's : 1.2
12. Coca cola : 0.35
13. Pepsi : 0.5
14. V220 : 0.65
15. Monster : 3.25
-> Elija un producto, q para salir: 5
5
Producto elegido: Ruffles : 0.25
Ingrese su pago separado por comas -> 1,1
Su cambio en monedas es: 1,50,25, total: 1.75
1. Papas sin marca : 0.5
2. Galletas : 0.45
3. Agua : 1.35
4. Oreo : 0.65
5. Ruffles : 0.25
6. Pringles : 2.35
7. Doritos : 0.25
8. Sandwich : 1.65
9. Manzana : 45
10. Yogurt toni : 0.5
11. mmm's : 1.2
12. Coca cola : 0.35
13. Pepsi : 0.5
14. V220 : 0.65
15. Monster : 3.25
-> Elija un producto, q para salir: 
```

17 Manual de usuario

17.1 Manual de usuario para configurar la Raspberry Pi

- Lo primero que debemos hacer es encender la Raspberry y ejecutar en la consola de Linux los comandos para verificar los puertos disponibles así en el terminal digitaremos: `dmesg | grep tty`, Ésta instrucción permite ver los puertos seriales conectados.
- Una vez verificados los puertos pasemos a configurar la Raspberry para que el linux no use la UART para login Shell, para ello en la consola de comandos digitaremos: `sudo raspi-config`, con esta instrucción abriremos el software de configuración general de la tarjeta nos dirigiremos a la opción 5 de opciones de interfaz, posteriormente inhabilitaremos el login Shell, pero mantendremos habilitado el Hardware de la UART.
- Una vez hecho el ajuste, salimos del `raspi-config` y reiniciaremos la tarjeta para confirmar que se hizo correctamente el cambio mientras inicia la tarjeta no volvera a enviar datos por la UART.
- Ahora nuevamente debemos abrir la consola de Linux en la Raspberry Pi y configurar el archivo de inicio para ajustar el reloj de la UART para ello editaremos el archivo digitando: `sudo nano /boot/config.txt`
- En el archivo debemos adicionar las líneas al final del texto

```
core_freq=250  
enable_uart=1
```

- Salimos digitando `CTR+C` confirmamos los cambios digitando y guardando el archivo con el mismo nombre. Finalmente reiniciaremos la Tarjeta.

17.2 Manual de usuario para configurar la PC con UBUNTU

- Compruebe si Python ya está instalado

```
$ python --version
```

o bien

```
$ python3 --version
```

- Si no está instalado Python 2.7 o una versión posterior, instale Python con el administrador de paquetes de su distribución.

```
$ sudo apt-get install python3
```

Instalar Pip en Ubuntu

- `sudo apt-get install python-pip`

Instalar pip en Python3

Para instalar pip3 para Python3 podemos utilizar los siguientes comandos:

- `sudo python3 get-pip.py`

Para usar pip en Linux

Debemos abrir la consola y ejecutar el comando:

- `pip install nombre_paquete`

Instalar el paquete Pyserial en Linux

Descargamos el paquete pyserial

- Se dirigen a la carpeta donde fue descargado
- Entran al terminal de Ubuntu
- Colocan `cd Descargas` si tienen contraseña les pedirá que la introduzcan
- Una vez hecho eso escriben `ls Pyserial` y la versión que se descargaron
- Comenzará la instalación

Introducen pyserial y la versión junto con `ls`

- Se les abre la carpeta
- Escriben `python setup.py build` y le dan enter
- Colamos el comando y se instalara

`sudo python setup.py install`