

Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Puebla



Diseño de sistemas en chip
TE2003B.501

Dr. Emmanuel Torres

Raspberry y UART

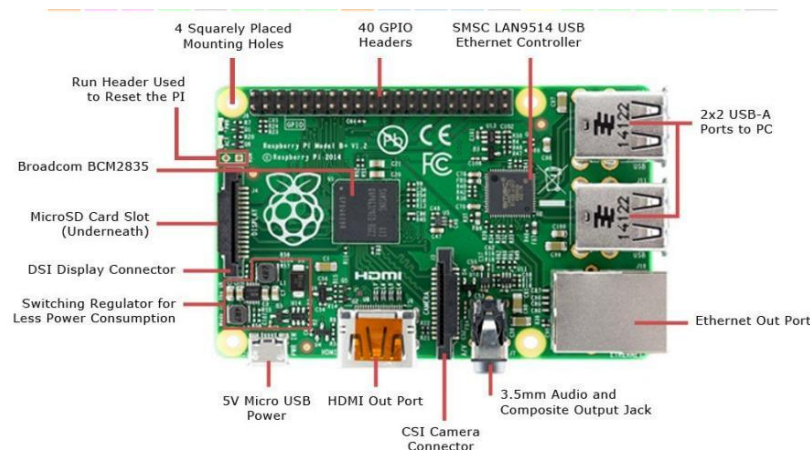
RoboInges | Integrantes:

Alan Iván Flores Juárez	A01736001
Givka Morales López	A01423534
Ulises Hernandez Hernández	A01735823

Abril 21, 2023

ambos sistemas embebidos, esto debido a la diferencia de voltajes que cada una de estas placas usa para sus puertos GPIO.

Se utiliza también una tarjeta Arduino Mega para mandar datos por el serial con un código que se realiza en arduino, para que al conectarlo a la raspberry por medio del puerto USB, este pueda leer e imprimir los datos que recibe por el puerto UART con un código que se realiza en python.



- Código Arduino:

```
enviar_datos_serial
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Holaaaaa");
  Serial.println("mundooo");
  Serial.println("Esta es la materia de:");
  Serial.println("Diseno");
  Serial.println("en");
  Serial.println("Chip");
  Serial.println("Hasta luego companero");
  delay(3000);
}
```

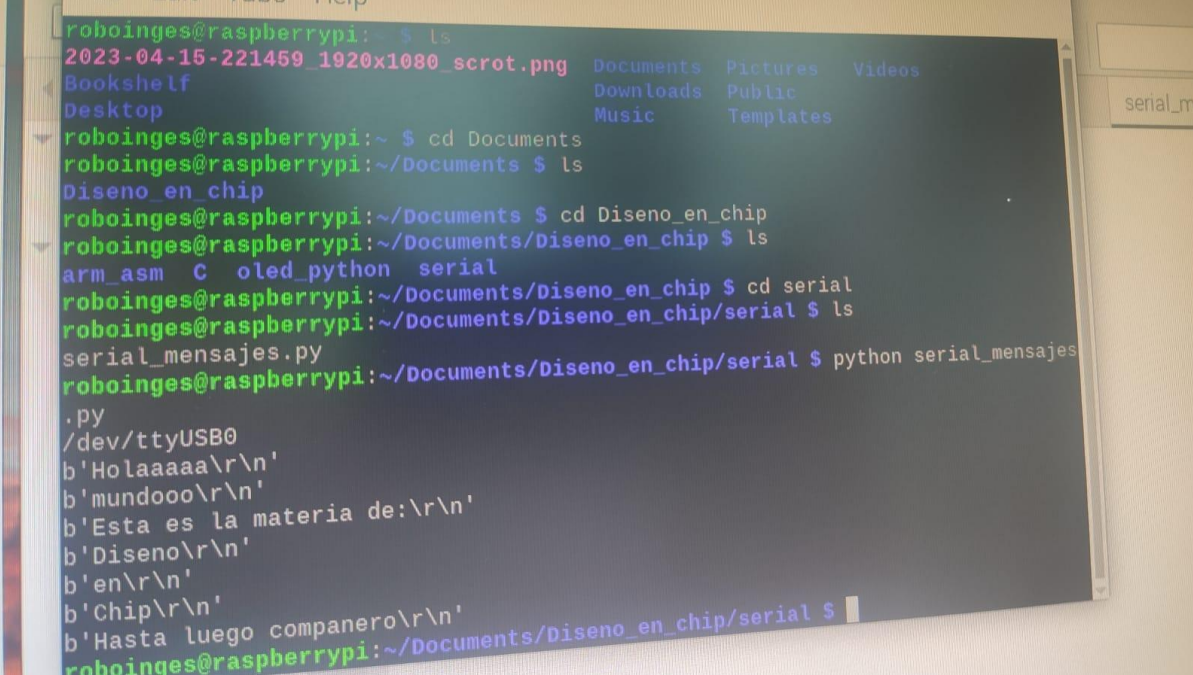
Este código se utiliza para establecer la comunicación serial entre el Arduino y la RaspBerry Pi 4, primero establece la velocidad de transmisión de datos o baudios a los cuales se comunican ambos dispositivos en la función setup() como se requiere en el protocolo UART para finalmente mandar los 7 mensajes arbitrarios en la función loop utilizando Serial.println("") y un delay de 3 segundos una vez enviados los 7 mensajes.

- **Comandos:**
- Se utiliza el comando `ls -l /dev/tty*` para obtener la ruta de comunicación entre ambos dispositivos.
- **Código python:**

```
1 import Serial
2 s=serial.Serial('/dev/ttyUSB0',9600)
3 print(s.name)
4 for i in range (7):
5     lectura = s.readline()
6     print(lectura)
7
8 s.close()
```

Este código en python se utiliza para desplegar en la terminal los mensajes enviados por el Arduino a través de comunicación serial, para esto se importa la librería Serial con el fin de utilizar estas funciones. Primero se declara el puerto serial con su ruta de acceso y los baudios de comunicación, es importante recalcar que la velocidad de transmisión debe ser la misma en ambos códigos para que la comunicación sea exitosa. Posteriormente se realiza un ciclo for con la función readline() para desplegar python de manera correcta todos los mensajes.

Resultados



```
roboinges@raspberrypi:~$ ls
2023-04-15-221459_1920x1080_scr0t.png  Documents  Pictures  Videos
Bookshelf                               Downloads  Public
Desktop                                 Music      Templates
roboinges@raspberrypi:~$ cd Documents
roboinges@raspberrypi:~/Documents$ ls
Diseno_en_chip
roboinges@raspberrypi:~/Documents$ cd Diseno_en_chip
roboinges@raspberrypi:~/Documents/Diseno_en_chip$ ls
arm_asm  C  oled_python  serial
roboinges@raspberrypi:~/Documents/Diseno_en_chip$ cd serial
roboinges@raspberrypi:~/Documents/Diseno_en_chip/serial$ ls
serial_mensajes.py
roboinges@raspberrypi:~/Documents/Diseno_en_chip/serial$ python serial_mensajes
.py
/dev/ttyUSB0
b'Holaaaaa\r\n'
b'mundooo\r\n'
b'Esta es la materia de:\r\n'
b'Diseno\r\n'
b'en\r\n'
b'Chip\r\n'
b'Hasta luego companero\r\n'
roboinges@raspberrypi:~/Documents/Diseno_en_chip/serial$
```

Conclusión

Podemos concluir que los protocolos de comunicación como lo puede ser UART son de gran ayuda para establecer comunicación con una variedad de dispositivos, que pueden incluso no compartir el mismo voltaje entre sus pines de comunicación o que difieran en la lógica del controlador, siendo de gran utilidad en este caso para poder comunicar un Arduino Mega y una RaspBerry y enviar 7 mensajes a través de una comunicación serial entre ambos.

Podemos extrapolar este tipo de usos para infinidad de dispositivos, que como mencionamos anteriormente, no requieren de ni siquiera compartir la misma arquitectura para establecer comunicación y de esta manera aprovechar las ventajas que cada arquitectura o estándar posee dependiendo de la aplicación para la que se requiera.