O'REILLY®

2a Edición

Simon Monk

Ejercicios prácticos con Raspberry Pi

PROBLEMAS Y SOLUCIONES DE SOFTWARE Y HARDWARE

Marcombo

Ejercicios prácticos con Raspberry Pi

Simon Monk

Marcombo

Edición original publicada en inglés por O'Reilly con el título *Raspberry Pi Cookbook*, ISBN 978-14-919-3910-9 © Simon Monk 2016.

Título de la edición en español:

Ejercicios prácticos con Raspberry Pi

Segunda edición en español, 2017

Gran Via de les Corts Catalanes, 594
08007 Barcelona
www.marcombo.com

Traducción: Loly Martos
Revisión técnica: Ferran Fàbregas
Diseño de la cubierta: ENEDENÚ DISEÑO GRÁFICO

ISBN versión digital en PDF: 978-84-267-2413-7

ISBN versión en papel: 978-84-267-2450-2

Tabla de contenidos

Prefacio

Lanzado en 2011, Raspberry Pi ha encontrado un papel como un ordenador basado en Linux de muy bajo coste y como una plataforma para los sistemas informáticos incorporados. Resulta popular para los educadores y para los aficionados por igual.

Desde la primera edición de este libro, se han vendido varios millones de Raspberry Pi y se han producido nuevos modelos. Algunos modelos, como los modelos B+, A+ y el modelo B+ de la Pi 2, mejoran la especificación de este dispositivo, surgiendo así Raspberry Pi 2 con procesador de cuatro núcleos y el modelo Raspberry Pi Compute, que proporciona una placa de extensión que puede ser parte de un sistema más grande.

Esta edición ha sido completamente actualizada para abarcar los nuevos modelos de Raspberry Pi, así como los muchos cambios y mejoras de su sistema operativo Raspbian.

Esta edición contiene un nuevo capítulo sobre visión artificial y otro capítulo para hacer proyectos del Internet de las Cosas con Raspberry Pi.

Este libro está diseñado de tal manera que usted puede leerlo linealmente como lo haría con cualquier libro, o acceder a los capítulos de forma individual. Puede buscar en la tabla de contenido aquello que desee y saltar directamente a ese capítulo. Si el capítulo requiere que usted sepa sobre otras cosas, hará referencia a otros capítulos, igual que un libro de cocina podría referirse a una salsa concreta antes de mostrar cómo cocinar algo más elaborado.

El mundo Raspberry Pi se mueve rápidamente. Con una comunidad grande y activa se están desarrollando constantemente nuevas tarjetas de interfaz y bibliotecas de *software*. Además de los ejemplos que utilizan tarjetas de interfaz o *softwares* específicos, el libro también cubre los principios básicos para que pueda tener una mejor comprensión de cómo usar las nuevas tecnologías que aparecen a medida que se desarrolla el ecosistema Raspberry Pi.

Como era de esperar, al libro le acompaña una gran cantidad de código (principalmente los programas de Python). Estos programas son de código abierto y están disponibles en GitHub.

Para la mayoría de los capítulos basados en *software*, lo único que necesita es una Raspberry Pi. Recomiendo la Raspberry Pi 2 o 3. Cuando se trata de capítulos que

involucran la fabricación del propio *hardware* para interconectarlo con Raspberry Pi, he tratado de usar módulos ya preparados, así como las placas de prueba sin soldadura y cables puente, para evitar tener que soldar.

Para aquellos que desean hacer más duraderos los proyectos basados en placas de prueba, aconsejo el uso de placas de prototipos con el mismo diseño que una placa de prueba de tamaño medio, como las vendidas por Adafruit, para que el diseño pueda soldarse.

Convenciones utilizadas en este libro

En este libro se utilizan las siguientes convenciones tipográficas:

Cursiva

Indica nuevos términos, URL, direcciones de correo electrónico, nombres de archivos y extensiones de archivo.

`Ancho constante`

Utilizado para listados de programas, así como dentro de párrafos para referirse a elementos del programa tales como nombres de variables o funciones, bases de datos, tipos de datos, variables de entorno, enunciados y palabras clave.

`Ancho constante en negrita`

Comandos u otros textos que deben ser escritos literalmente por el usuario.

`Ancho contante en cursiva`

Muestra el texto que debe ser reemplazado por valores proporcionados por el usuario o por valores determinados por el contexto.

Este icono significa un consejo, sugerencia o nota general.

Este icono indica una advertencia o precaución.

Este icono le señala el vídeo relacionado para esta sección.

Uso de ejemplos de código

El material complementario (ejemplos de código, ejercicios, etc.) está disponible para descargar en *http://www.raspberrypicookbook.com*.

Este libro está aquí para ayudarle a hacer su trabajo. En general, si se ofrece código de ejemplo con este libro, puede utilizarlo en sus programas y documentación. No necesita ponerse en contacto con nosotros para obtener ningún permiso a menos que esté reproduciendo una parte significativa del código. Por ejemplo, escribir un programa que utiliza varios fragmentos de código de este libro no necesita permiso. La venta o distribución de un CD-ROM de ejemplos de los libros de O'Reilly necesita permiso. Responder a una pregunta citando este libro y citando el código de ejemplo no requiere permiso. Incorporar una cantidad significativa de código de ejemplo desde este libro a la documentación de su producto requiere permiso.

Apreciamos, pero no pedimos, la atribución. Una atribución generalmente incluye el título, el autor, la edición y el ISBN. Por ejemplo: "*Raspberry Pi Cookbook*, Segunda Edición, por Simon Monk (O'Reilly). Copyright 2016 Simon Monk, 978-1-491-93910-9."

Agradecimientos

Como siempre, doy las gracias a mi esposa Linda por su paciencia y apoyo.

También agradezco al revisor técnico Duncan Amos por su buen ojo, buen humor y sus excelentes propuestas que, sin duda, han contribuido en gran medida a este libro.

Gracias también a todo el equipo de O'Reilly, especialmente a los que conocí en la oficina de Cambridge, que fueron muy acogedores cuando fui. Y, por supuesto, a Nan Reinhardt por su copiosa edición.

CAPÍTULO 1

Ajustes y configuración

1.1 Introducción

Al comprar Raspberry Pi está comprando esencialmente una placa ensamblada de circuitos impresos. No incluye ni una fuente de alimentación ni un sistema operativo.

Este capítulo explicará cómo tener su Raspberry Pi configurada y lista para su uso.

Debido a que Raspberry Pi solo utiliza teclados y ratones USB estándares, la mayor parte de la configuración es bastante sencilla, por lo que se concentrará tan solo en las tareas específicas de Raspberry Pi.

1.2 Seleccionar un modelo de Raspberry Pi

Problema

Hay muchos modelos de Raspberry Pi y no está seguro de cuál utilizar.

Solución

Si quiere una Raspberry Pi para uso general, debe comprar el modelo B de la Raspberry Pi 3 o 2. Con cuatro veces más memoria y un procesador de cuatro núcleos, afrontará la mayoría de las tareas mucho mejor que con Pi Zero o con el modelo A+ y sus procesadores individuales. El modelo B de la Raspberry Pi 3 tiene la gran ventaja de tener wifi incorporado, así no necesitará un adaptador wifi USB adicional.

Si por el contrario ha incorporado una Raspberry Pi a un proyecto para un solo propósito, la mejor opción es utilizar el modelo A+ o Pi Zero y así ahorrará algunos euros.

Observaciones

La Figura 1.1 muestra la Pi Zero, el modelo A+ y el modelo B de la Raspberry Pi 2.

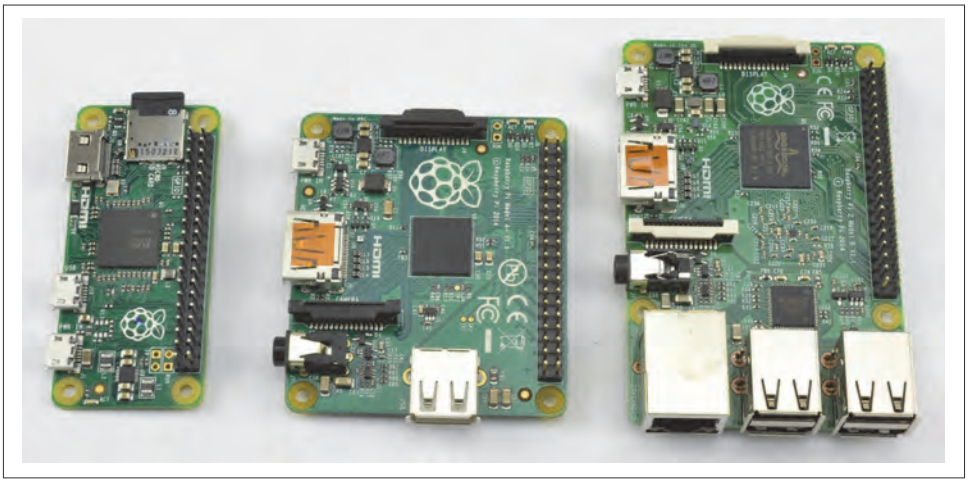


Figura 1.1. *La Raspberry Pi Zero (izquierda), el modelo A+ (centro) y el modelo B de la Raspberry Pi 2 (derecha).*

Como puede ver en la Figura 1.1, el modelo A+ es más pequeño que la Pi 2 y tiene solamente un puerto USB y ningún conector Ethernet RJ45. La Pi Zero es incluso más pequeña, ahorrando espacio al usar una ranura mini para el HDMI y un micro puerto USB. Si desea conectar un teclado y un ratón a la Pi Zero, necesitará adaptadores, tanto para los puertos USB como HDMI para poder conectar los periféricos estándar.

Las diferencias entre todos los modelos de Raspberry Pi hasta la fecha se resumen en la Tabla 1.1.

Tabla 1.1. *Modelos de Raspberry Pi.*

Modelo	RAM	Puertos USB	Puerto Ethernet	Notas
3 B	1 GB	4	sí	Incluye wifi
Zero	512 MB	1	no	Bajo coste
2 B	1 GB	4	sí	Cuatro núcleos
A+	256 MB	1	no	
B+	512 MB	4	sí	Interrumpido
A	256 MB	1	no	Interrumpido
B rev2	512 MB	2	sí	Interrumpido
B rev1	256 MB	2	sí	Interrumpido

Si usted tiene uno de los modelos antiguos interrumpidos de Raspberry Pi, son igualmente útiles. No tienen el rendimiento que tiene la última, el modelo B de la Raspberry Pi 3, pero para muchas situaciones no importa.

En el Capítulo 9.22, se introducirá el módulo Raspberry Pi Compute. Está diseñado específicamente para permitir que Raspberry Pi sea incorporado en un producto.

Para saber más

Para obtener más información sobre los modelos de Raspberry Pi, consulte *http://es.wikipedia.org/wiki/Raspberry_Pi*.

El coste bajo de la Pi Zero hace que sea ideal para añadirla en los proyectos de electrónica sin tener que preocuparse por el precio. Véase el Capítulo 9.23.

1.3 Proteger Raspberry Pi

Problema

Necesita una carcasa para su Raspberry Pi.

Solución

Raspberry Pi no viene con carcasa a menos que usted la compre como parte de un kit. Esto hace que sea vulnerable, ya que hay conexiones descubiertas en la parte inferior de la placa del circuito que pueden producir fácilmente un corto circuito si se coloca en un objeto de metal.

Comprar algo que proteja su Raspberry Pi es una buena idea, por ejemplo, una carcasa. Si tiene intención de utilizar los pines GPIO de Raspberry Pi, puede ver el PiBow Coupé en la Figura 1.2 con un diseño bonito y práctico.

Observaciones

Existe una gran variedad de estilos para elegir, incluyendo:

- Simple, de dos partes, que encajan en cajas de plástico.
- Carcasas montables VESA (para adjuntar a la parte trasera de un monitor o un televisor).
- Carcasa estilo Lego.
- Diseños de carcasas impresas en 3D.
- Diseños acrílicos a presión cortados con láser.

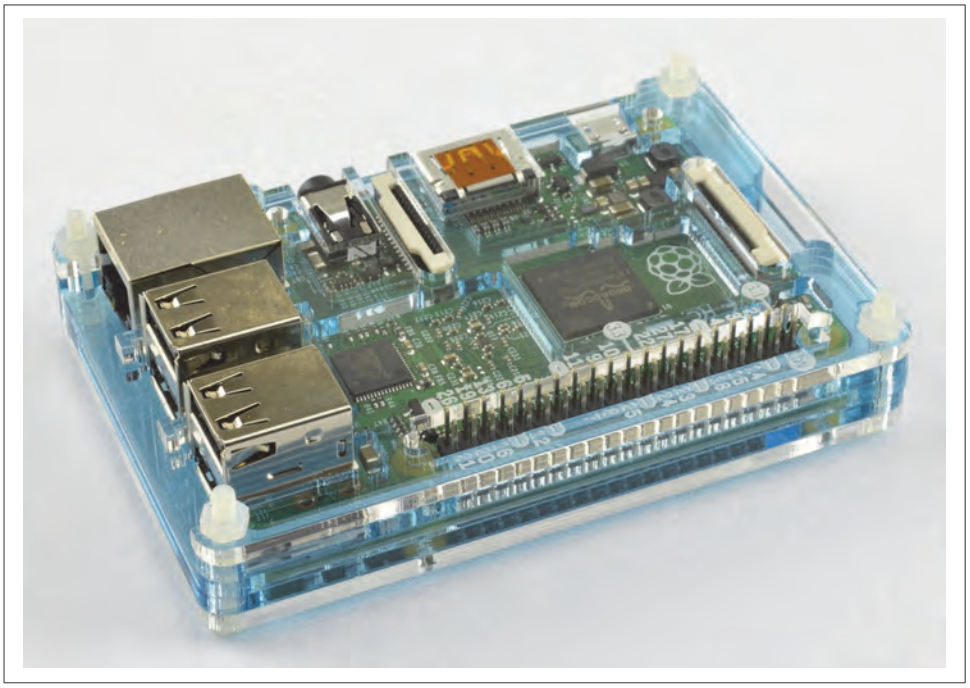


Figura 1.2. *Raspberry Pi 2 en un PiBow Coupé*

La que compre es cuestión de gustos personales. Sin embargo, algunas de las cosas que hay que considerar son las siguientes:

- ¿Necesita tener acceso al conector GPIO? Esto es importante si va a conectar aparatos electrónicos externos a su Raspberry Pi.
- ¿Está bien ventilada la carcasa? Es importante si va a sobrecargar su Raspberry Pi (Capítulo 1.11) o si va a reproducir vídeos o a jugar, ya que estos generan más calor.

Encontrará kits de disipador de calor que tienen pequeños disipadores autoadhesivos para unir a los chips de su Raspberry Pi. Estos pueden ser de utilidad si está exigiéndole mucho a su Raspberry Pi, como al reproducir muchos vídeos.

Para saber más

Adafruit tiene una gran variedad de carcasas para Raspberry Pi.

También encontrarás muchos estilos de carcasas en los proveedores de Raspberry Pi en eBay.

1.4 Seleccionar una fuente de alimentación

Problema

Necesita seleccionar una fuente de alimentación para su Raspberry Pi.

Solución

La especificación eléctrica básica para una fuente de alimentación adecuada para Raspberry Pi es la que proporciona 5 V de CC (corriente continua).

La cantidad de corriente que la fuente de alimentación debe ser capaz de proporcionar depende tanto del modelo de Raspberry Pi como de los periféricos conectados. Vale la pena conseguir una fuente de alimentación que pueda hacer frente fácilmente a Raspberry Pi y debe considerar 700 mA como mínimo.

Si usted compra su fuente de alimentación en el mismo lugar donde compre la Raspberry Pi, el vendedor le podrá decir si funcionará.

Si va a utilizar un dongle wifi u otros periféricos USB que utilizan cantidades significativas de energía, debería tener una fuente de alimentación que soporte 1,5 A o incluso 2 A. Tenga cuidado con las fuentes de alimentación de muy bajo coste, ya que pueden no proporcionar 5 V de manera precisa y fiable.

Observaciones

En realidad, la fuente de alimentación y el conector son los mismos que los que se encuentran en muchos cargadores de teléfonos inteligentes. Si terminan en un conector micro USB, son, casi con toda seguridad, de 5 V (pero asegúrese). La única pregunta que debe formular es si pueden suministrar suficiente corriente.

Si no pueden, pueden suceder algunas cosas malas:

- Es posible que se calienten y puede haber riesgo de incendio.
- Pueden fallar.
- En momentos de alta carga (por ejemplo, cuando Pi está utilizando un dongle wifi) el voltaje puede caer y Raspberry Pi se reiniciará solo.

En general, busque una fuente de alimentación que pueda suministrar 700 mA o más. Si se especifica un número de vatios (W) en lugar de mA, divida el número de vatios por 5 para obtener la cifra en A. Por lo tanto, una fuente de alimentación de 5 V (10 W) puede suministrar 2 A (2000 mA).

El uso de una fuente de alimentación con, por ejemplo, una corriente máxima de 2 A no utilizará más electricidad que una fuente de alimentación de 700 mA. Raspberry Pi tomará la mayor cantidad de corriente que necesite.

En la Figura 1.3 se mide la corriente tomada por el modelo B de la Raspberry Pi y se compara con modelo B de la Pi 2.

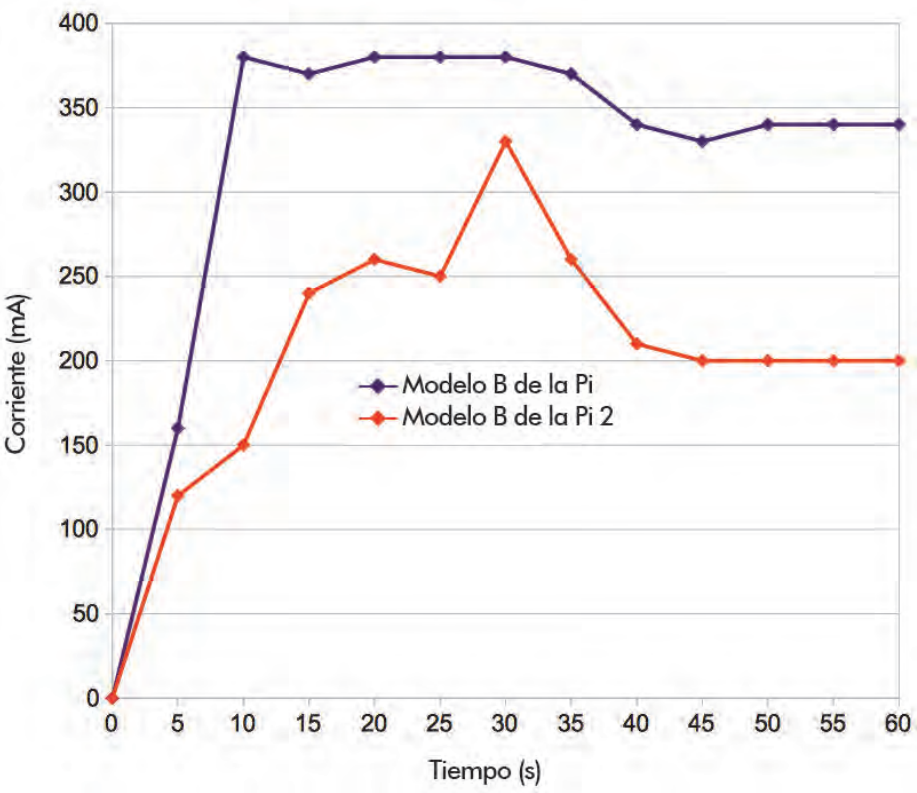


Figura 1.3. *Consumo de corriente durante el inicio de Raspberry Pi.*

Las Raspberry Pi más recientes (el A+, B+ o Raspberry Pi 2) son mucho más eficientes en cuanto a energía que los modelos más antiguos. Sin embargo, cuando el procesador está totalmente ocupado y tiene una gran cantidad de periféricos conectados, pueden llegar a ser similares.

En la Figura 1.3 puede ver que la corriente rara vez se pone por encima de 500 mA. Sin embargo, el procesador no está haciendo nada realmente. Si reprodujese un vídeo HD, la corriente aumentaría considerablemente. Cuando se trata de fuentes de alimentación, por lo general, es mejor tener algo en reserva.

Para saber más

Puede comprar un módulo que se desconecte de la alimentación cuando la Raspberry Pi se apague en *http://www.pi-supply.com/*.

1.5 Seleccionar la distribución del sistema operativo

Problema

Hay un número elevado de distribuciones para Raspberry Pi. No sabe cuál utilizar.

Solución

La respuesta a esta pregunta depende de lo que vaya a hacer con su Raspberry Pi.

Para un uso general como ordenador, o para usarlo en proyectos electrónicos, debe utilizar Raspbian, la distribución estándar y oficial para Raspberry Pi.

Si va a utilizar su Raspberry Pi como centro multimedia, existe una serie de distribuciones específicamente para ese propósito (véase Capítulo 4.2).

En este libro utilizamos la distribución Raspbian casi exclusivamente, aunque la mayoría de los ejemplos funcionarán con cualquier distribución basada en Debian.

Observaciones

Las tarjetas microSD no son caras, por lo que consiga unas cuantas y pruebe algunas distribuciones. Si hace esto, es buena idea mantener sus archivos en una unidad flash USB, de manera que no tenga que estar copiándolos en cada tarjeta microSD.

Tenga en cuenta que, si va a seguir los siguientes pasos para escribir su propia tarjeta SD, necesitará tener un equipo con ranura para tarjetas SD (muchos portátiles la tienen), o puede comprar un lector USB de tarjetas SD de bajo coste.

Para saber más

La lista oficial de distribuciones de Raspberry Pi

1.6 Grabar una tarjeta microSD con NOOBS

Problema

Quiere grabar una tarjeta microSD utilizando NOOBS (*New Out of the Box Software*).

Solución

NOOBS es, con diferencia, la forma más sencilla de obtener un sistema operativo en su Raspberry Pi.

Descargue el archivo de NOOBS de *http://www.raspberrypi.org/downloads*, extráigalo y colóquelo en una tarjeta microSD. Para ello necesitará un ordenador con ranura para tarjetas SD o un adaptador USB y un adaptador de SD a micro SD.

Una vez haya descargado el archivo de almacenamiento NOOBS, extráigalo y copie los contenidos de la carpeta en la tarjeta SD. Tenga en cuenta que si el archivo extrae la carpeta llamada *NOOBS_v1_3_12* o similar, es el contenido de la carpeta el que debe ser copiado en la raíz de la tarjeta micro SD, no la propia carpeta.

Coloque la tarjeta microSD que contiene los archivos extraídos de su NOOBS en su Raspberry Pi y después encienda Raspberry Pi. Cuando arranque, aparecerá la ventana que se muestra en la Figura 1.4. Desde esta pantalla, seleccione Raspbian y luego haga clic en el botón *Instalar*.

Si está utilizando NOOBS en una A+, verá una lista más corta de opciones, ya que solo se mostrarán las distribuciones para esa plataforma simplificada.

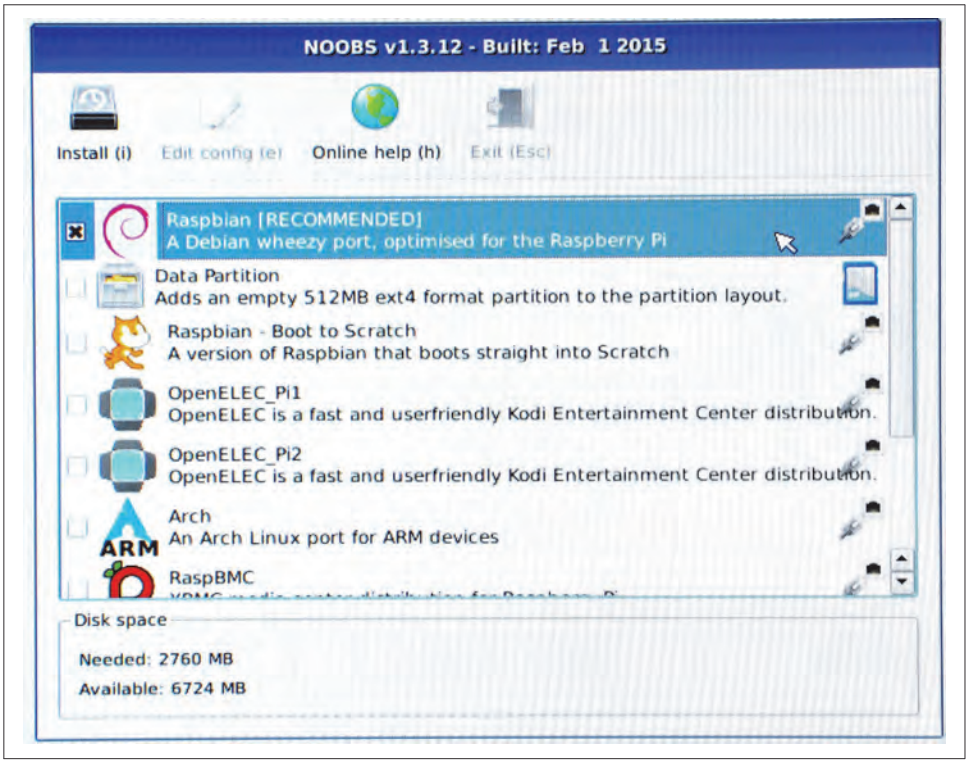


Figura 1.4. *Primera pantalla NOOBS.*

Recibirá un mensaje de advertencia diciendo que la tarjeta SD será sobrescrita (que está bien) y luego, como la distribución está instalada en la tarjeta SD, verá el progreso en la pantalla acompañado de información útil acerca de la distribución (Figura 1.5).

Figura 1.5. *NOOBS sobrescribiendo la tarjeta SD.*

Una vez que la copia de archivos se haya completado, recibirá el mensaje *Image applied successfully*. Raspberry Pi se reiniciará y, automáticamente, se ejecutará *raspi_config* para que pueda configurar la nueva instalación.

Una vez que esté en funcionamiento, la primera cosa que debe hacer es conectar su Raspberry Pi a Internet (Capítulo 2.2 y 2.6), abra una línea de comandos mediante LXTerminal (Capítulo 3.3), e introduzca el siguiente comando para actualizar su sistema a la última versión.

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

Esto tardará un rato.

Observaciones

Para instalar NOOBS correctamente en una tarjeta microSD, la tarjeta debe estar formateada como FAT32. La mayoría de las tarjetas microSD se suministran ya formateadas en FAT32. Si va a reutilizar una tarjeta antigua y necesita darle formato FAT32, utilice las herramientas de su sistema operativo para formatear los dispositivos extraíbles.

El tipo de tarjeta microSD que tenga también afectará a la velocidad de ejecución de su Raspberry Pi cuando su sistema operativo esté instalado. Busque una tarjeta microSD descrita como "clase 10."

Para saber más

Puede encontrar más información sobre la instalación de su sistema operativo con NOOBS, así como información sobre las diferentes distribuciones disponibles, en *https://www.raspberrypi.org/help/noobs-setup/*.

1.7 Conectar el sistema

Problema

Tiene todo lo que necesita para su Raspberry Pi y quiere conectarlo todo junto.

Solución

A menos que incorpore su Raspberry Pi en un proyecto o la utilice como centro multimedia, necesita conectar un teclado, un ratón, un monitor y probablemente un dongle wifi, excepto si tiene una Raspberry Pi 3.

La Figura 1.6 muestra un típico sistema de Raspberry Pi.

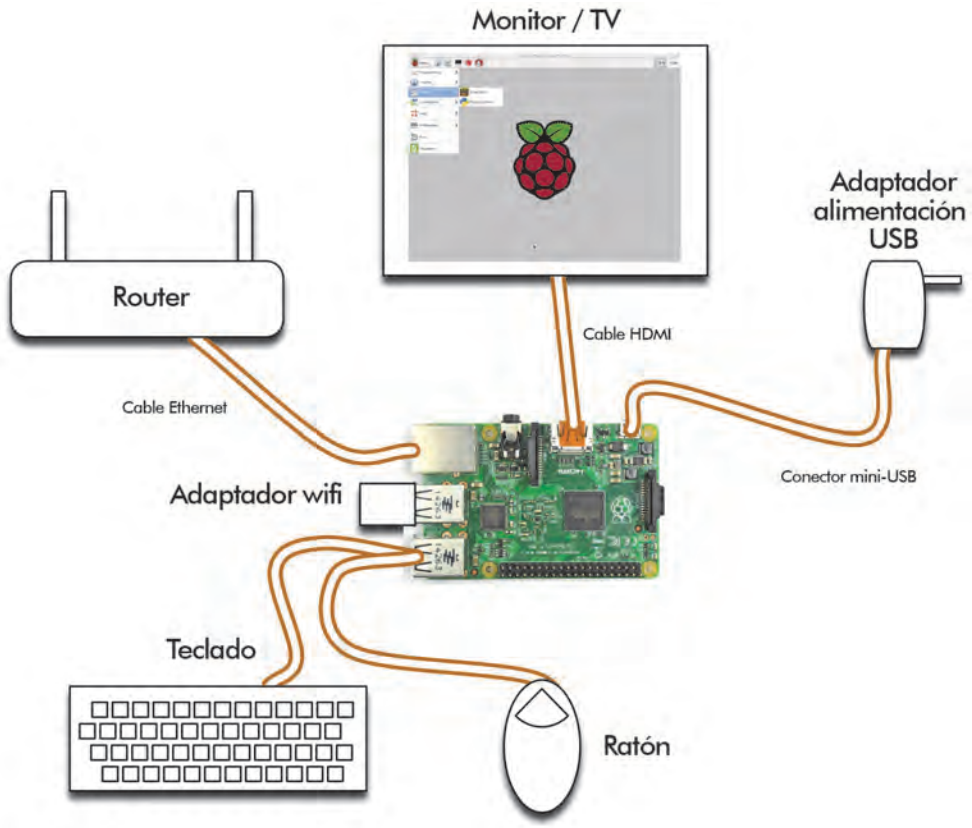


Figura 1.6. *Típico sistema de Raspberry Pi.*

Observaciones

A Raspberry Pi le va perfectamente bien cualquier teclado o ratón, alámbrico o inalámbrico. Excepto los teclados y ratones conectados por Bluetooth, que no funcionan con Raspberry Pi.

Si tiene una Raspberry Pi vieja o el modelo A o A+ y se queda sin conexiones USB, necesitará un hub USB.

Para saber más

La guía oficial de inicio rápido de Raspberry Pi

1.8 Conectar un monitor DVI o VGA

Problema

Su monitor no tiene HDMI, pero desea utilizarlo con su Raspberry Pi.

Solución

Mucha gente ha tenido este problema. Afortunadamente, es posible comprar adaptadores para monitores con entrada a DVI o VGA.

Los adaptadores DVI son los más simples y baratos. Se pueden encontrar por menos de 5 euros si se busca "convertidor macho HDMI para hembra DVI".

Observaciones

El uso de adaptadores VGA es más complejo, ya que requieren un poco de electrónica para convertir la señal de digital a analógico, así que cuidado con los cables que no los contengan. El convertidor oficial se llama *Pi-View* y está disponible en los distribuidores oficiales de Raspberry Pi. Pi-View tiene la ventaja de haber sido probado para trabajar con Raspberry Pi. Es posible encontrar alternativas más baratas a través de Internet, pero a veces no funcionan.

Para saber más

eLinux tiene consejos sobre lo que debe buscar en un convertidor.

1.9 Usar un monitor de vídeo compuesto

Problema

El texto en su monitor compuesto de baja resolución es ilegible. Es necesario ajustar la resolución a Raspberry Pi para una pantalla pequeña.

Solución

Raspberry Pi tiene dos tipos de salida de vídeo: HDMI y vídeo compuesto desde el conector de audio, para el que necesita un cable especial. De estos, el HDMI es de mejor calidad. Si usted tiene la intención de utilizar la salida de vídeo compuesto para su pantalla principal, es posible que se lo deba replantear.

Si está utilizando la pantalla de ese modo —por ejemplo, porque necesita una pantalla muy pequeña— entonces es necesario hacer algunos ajustes para adaptar la salida de vídeo a la pantalla. Necesitará hacer algunos cambios en el archivo */boot/config.txt*.

Puede editarlo en Raspberry Pi utilizando el siguiente comando en una sesión de terminal:

```
$ sudo nano /boot/config.txt
```

Si el texto es demasiado pequeño para leerlo y no tiene monitor HDMI, también puede editar el archivo quitando la tarjeta SD de su Raspberry Pi e insertándola en su ordenador. El archivo estará en el directorio raíz de la tarjeta SD y puede utilizar un editor de texto desde su ordenador para modificarlo.

Es necesario conocer la resolución de la pantalla. Para la mayoría de las pequeñas pantallas será 320 por 240 píxeles. Encuentre las dos líneas en el archivo que digan:

```
#framebuffer_width=1280
#framebuffer_height=720
```

Suprima el # del principio de cada línea y cambie los dos números a la anchura y altura de la pantalla. En el siguiente ejemplo, las líneas se han modificado para que sean 320 por 240:

```
framebuffer_width=320
framebuffer_height=240
```

Guarde el archivo y reinicie su Raspberry Pi. Debe notar que todo es más fácil de leer. Es probable que también vea un borde grande y grueso alrededor de la pantalla. Para ajustarlo véase el Capítulo 1.10.

Observaciones

Hay muchos monitores de circuito cerrado de bajo coste que pueden ir muy bien con su Raspberry Pi cuando esté haciendo algo como una consola retro (Capítulo 4.7). Sin embargo, estos monitores son a menudo de baja resolución.

Para saber más

Para otro tutorial sobre el uso de monitores compuestos, véase este tutorial Adafruit.

También, véase el Capítulo 1.8 y 1.10 para ajustar la imagen cuando se está utilizando la salida de vídeo HDMI.

1.10 Ajustar el tamaño de la imagen al monitor

Problema

Cuando conecte por primera vez Raspberry Pi a un monitor, es posible que algunos textos no se puedan leer al extenderse fuera de la pantalla, o que la imagen no utilice todo el espacio disponible en la pantalla.

Solución

Si el texto se extiende fuera de la pantalla, use la herramienta `raspi-config` para desactivar `overscan`.

Para ello, ejecute `raspi-config` mediante la apertura de una sesión de terminal y ejecute el comando:

```
$ sudo raspi-config
```

A continuación, utilice las flechas para desplazarse hacia abajo a Advanced Options y después Overscan, y desactive overscan (Figura 1.7).

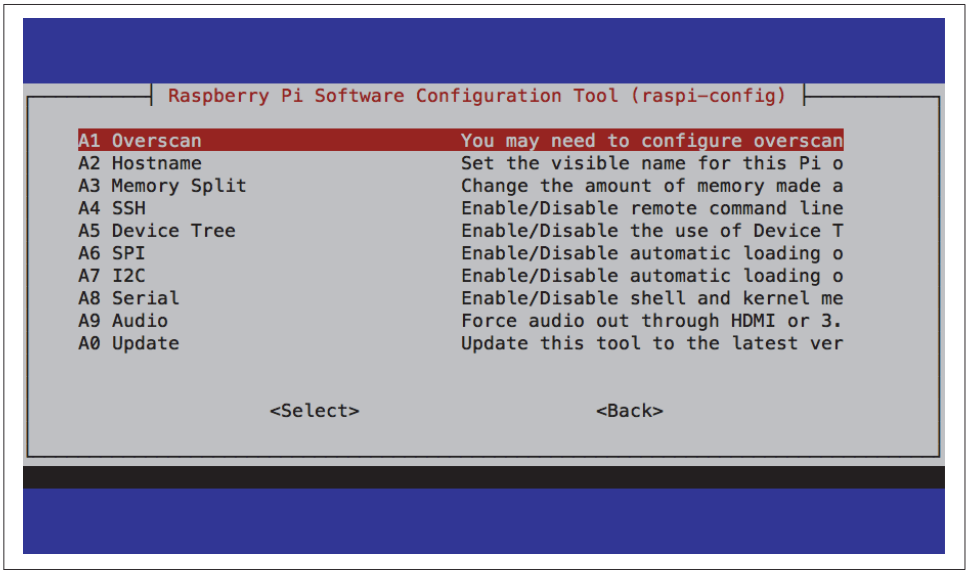


Figura 1.7. *Seleccionar la opción Overscan.*

Si su problema es que hay un gran borde negro alrededor de la imagen, lo puede reducir (y posiblemente eliminar por completo) al editar el archivo */boot/config.txt* utilizando el comando:

```
$ sudo nano /boot/config.txt
```

Busque la sección que va sobre overscan. Las cuatro líneas que hay que cambiar se muestran en el medio de la Figura 1.8.

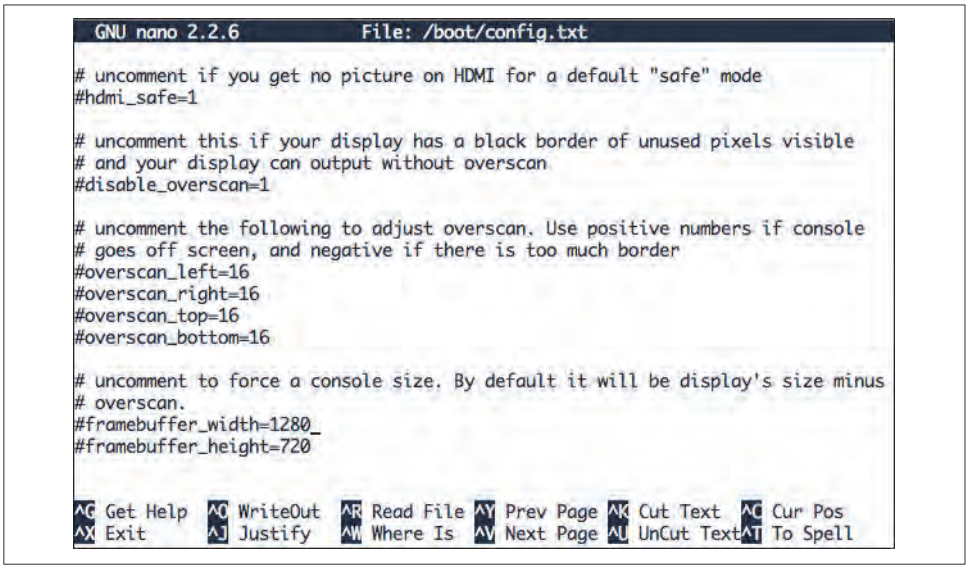


Figura 1.8. *Ajuste de overscan.*

Para que las líneas tengan efecto, primero tiene que *descomentarlas* quitando el carácter # del principio de cada línea.

Luego, utilizando ensayo y error, cambie los ajustes hasta que se llene la pantalla y el monitor tanto como sea posible. Tenga en cuenta que los cuatro números deben ser negativos. Pruebe a poner todos a –20 para empezar. Esto aumentará el área de la pantalla que utiliza.

Observaciones

Tener que reiniciar varias veces Raspberry Pi para ver los efectos de los cambios en la resolución es un poco tedioso. Afortunadamente, solo tendrá que hacer este procedimiento una vez. Muchos monitores y televisores funcionan muy bien sin ningún ajuste.

Para saber más

Puede encontrar más información acerca de la herramienta `raspi-config` en *http://elinux.org/RPi_raspi-config*.

1.11 Maximizar el rendimiento

Problema

Su Raspberry Pi parece que va lenta, por lo que desea mejorar la velocidad para que funcione más rápido.

Solución

Si usted tiene una Raspberry Pi 2 con un procesador de cuatro núcleos, es muy poco probable que vaya lenta. Sin embargo, las Raspberry Pi de un solo núcleo más antiguas pueden ir bastante lentas.

Puede aumentar la velocidad de su Raspberry Pi para que funcione más rápido. Esto hará que use un poco más de potencia y esté un poco más caliente (véase el siguiente apartado de «Observaciones»).

El método de overlocking descrito se llama *dynamic overclocking*, ya que controla automáticamente la temperatura de la Raspberry Pi y deja caer la velocidad si se calienta demasiado.

Para hacer overclock en su Pi, ejecute la utilidad de `raspi_config` a través del siguiente comando en un terminal:

```
$ sudo raspi-config
```

Seleccione la opción *Overclock* en el menú y se presentarán las opciones de la Figura 1.9.

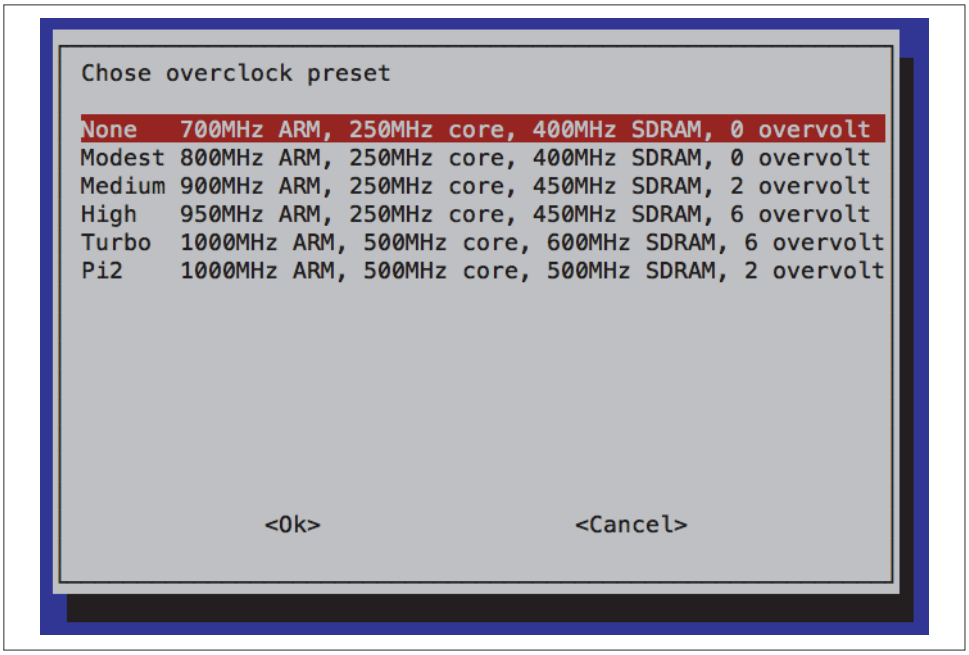


Figura 1.9. *Opciones de Overclocking.*

Seleccione una opción. Si usted ve que su Raspberry Pi empieza a ser inestable y que se bloquea inesperadamente, puede que tenga que elegir una opción más conservadora o configure el overclocking de nuevo en `None`.

Observaciones

Las mejoras en el rendimiento de overclocking pueden ser dramáticas. Para medirlas, use la revisión 2 del modelo B de la Raspberry Pi sin carcasa y a una temperatura ambiente de 15 °C.

La siguiente secuencia de comandos de Python es el programa de pruebas. Esto solo repercute en el procesador, y no es realmente representativo de las demás cosas que pasan en un ordenador, como, por ejemplo, escribir en una tarjeta SD, los gráficos, etc. Pero nos da una buena señal sobre el rendimiento de la CPU si desea evaluar el efecto de overclocking en su Raspberry Pi.

```
import time

def factorial(n):
    if n ==0:
        return 1
    else:
        return n * factorial(n-1)

before_time = time.clock()
for i in range(1, 10000):
    factorial(200)
after_time = time.clock()

print(after_time - before_time)
```

Compruebe los resultados del test en la Tabla 1.2.

Tabla 1.2. *Overclocking.*

	Test de velocidad	Corriente	Temperatura (grados C)
700 MHz	15,8 segundos	360 mA	27
1 GHz	10,5 segundos	420 mA	30

Como puede ver, el rendimiento ha aumentado en un 33 % pero con más corriente y a una temperatura ligeramente superior.

Un recinto bien ventilado ayudará a mantener el funcionamiento de la Raspberry Pi a toda velocidad. También ha habido algunos esfuerzos para añadir refrigeración por agua a Raspberry Pi. Francamente, es una tontería.

Para saber más

Puede encontrar más información acerca de la herramienta `raspi-config` en *http://elinux.org/RPi_raspi-config*.

1.12 Cambiar la contraseña

Problema

Por defecto, la contraseña de Raspberry Pi será *raspberry*. Quiere cambiarla.

Solución

Puede utilizar la herramienta `raspi-config` para cambiar su contraseña. Ejecute `raspi_config` mediante el siguiente comando en un terminal (véase el Capítulo 3.3):

```
$ sudo raspi-config
```

A continuación, seleccione la opción change_pass en el menú y siga las instrucciones que se muestran en la Figura 1.10.

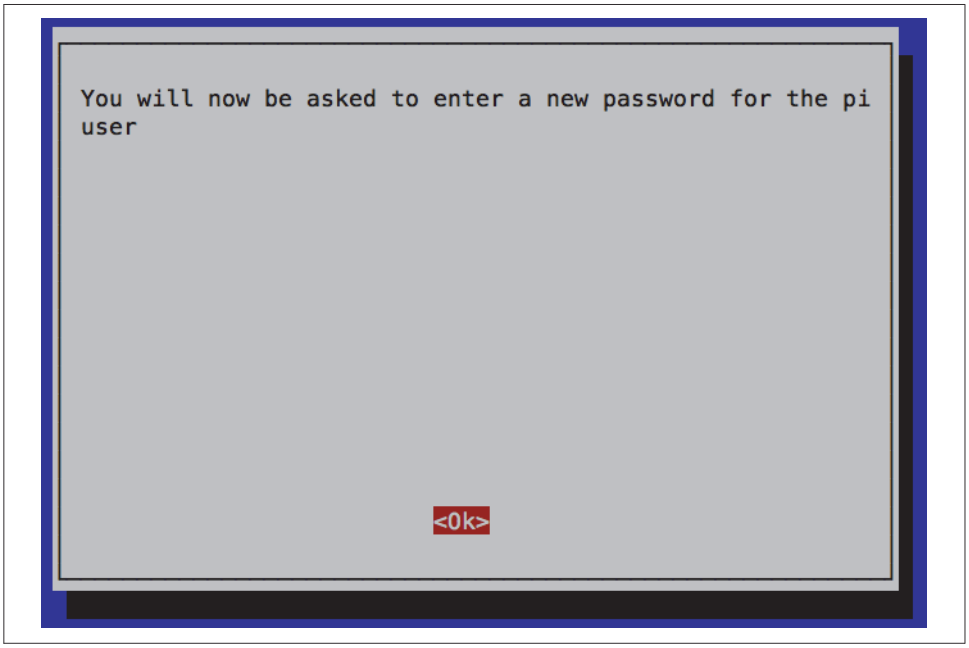


Figura 1.10. *Cambiar la contraseña.*

Para cambiar la contraseña no es necesario reiniciar Raspberry Pi para que los cambios surtan efecto.

Observaciones

También puede cambiar la contraseña en una sesión de terminal simplemente usando el comando `passwd` de la siguiente manera:

```
$ passwd
Changing password for pi.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Para saber más

Puede encontrar mucha más información sobre la herramienta `raspi-config` en *http://elinux.org/RPi_raspi-config*.

1.13 Ajustar Pi para empezar directamente con un sistema de ventanas

Problema

Cada vez que reinicie su Raspberry Pi tiene que iniciar sesión y luego iniciar el escritorio manualmente. Quiere hacer esto automáticamente.

Solución

Puede utilizar la herramienta `raspi-config` para cambiar el comportamiento de arranque para que Raspberry Pi se registre de manera automática y se inicie el escritorio. Ejecute la utilidad `raspi_config` con el siguiente comando en un terminal:

```
$ sudo raspi-config
```

Ahora, seleccione la opción Enable Boot to Desktop/Scratch y después Desktop Log in as user *pi* (Figura 1-11).

Cuando usted cambie la opción de arranque, se le pedirá que reinicie Raspberry Pi para que los cambios surtan efecto.

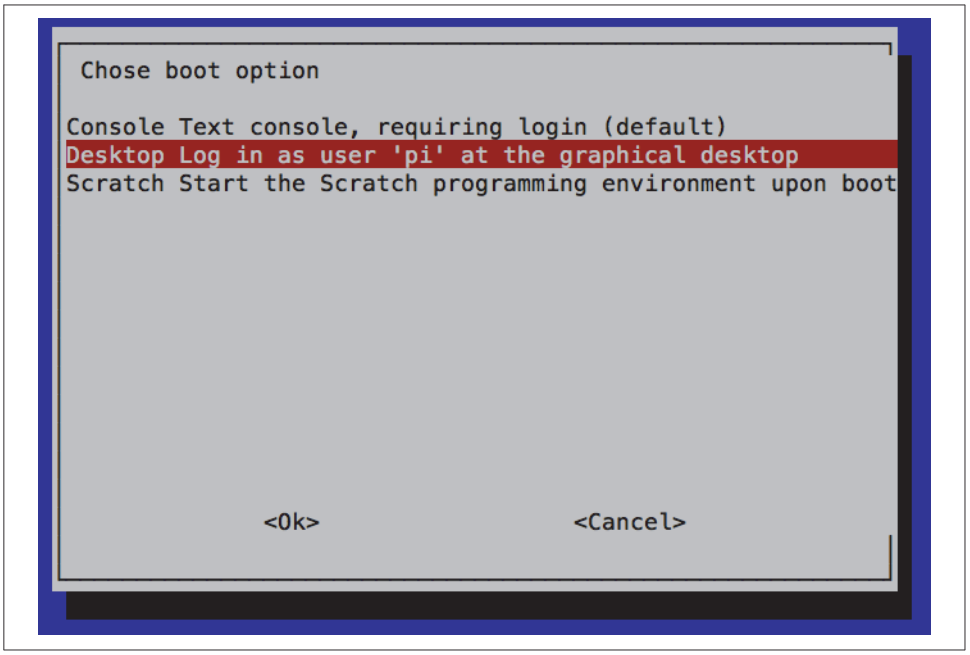


Figura 1.11. *Arranque automático en un escritorio.*

Observaciones

Evidentemente hay implicaciones de seguridad al permitir que Raspberry Pi inicie sesión en el entorno de ventanas de forma automática. Pero, como Raspberry Pi generalmente se usará como un ordenador personal, en lugar de ser compartido, normalmente la conveniencia pesa más que las desventajas.

Para saber más

Puede encontrar mucha más información sobre la herramienta `raspi-config` en *http://elinux.org/RPi_raspi-config*.

1.14 Apagar su Raspberry Pi

Problema

Quiere apagar su Raspberry Pi.

Solución

Haga clic en el menú Raspberry, en la esquina superior izquierda del escritorio. Esto mostrará una serie de opciones (Figura 1.12).

Shutdown

Apaga Raspberry Pi. Tendrá que desconectar la alimentación y volver a enchufarlo para que Raspberry Pi arranque de nuevo.

Reboot

Reinicia Raspberry Pi.

Logout

Finaliza la sesión y muestra un aviso para introducir sus credenciales de acceso para que pueda entrar de nuevo.

Figura 1.12. *Apagar su Raspberry Pi.*

También puede reiniciar desde la línea de comandos con el comando:

```
sudo reboot
```

Puede que tenga que hacer esto después de instalar algún *software*. Cuando lo reinicie, verá el mensaje que se muestra en la Figura 1.13, que ilustra la naturaleza multiusuario de Linux y advierte a todos los usuarios conectados a Pi.

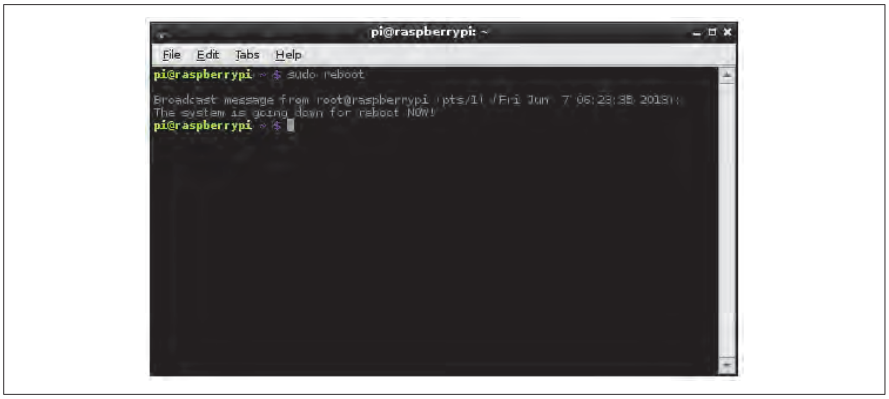


Figura 1.13. *Apagando su Raspberry Pi del terminal.*

Observaciones

Es mejor cerrar su Raspberry Pi como se describió anteriormente que simplemente tirar del enchufe, ya que Raspberry Pi puede estar en medio de la escritura de la tarjeta microSD. Esto podría llevar a la corrupción de archivos.

A diferencia de apagar la mayoría de los ordenadores, apagar Raspberry Pi no desconecta la alimentación. Entra en un modo de baja potencia; es un dispositivo de muy baja potencia (pero el *hardware* de Raspberry Pi no tiene control sobre su fuente de alimentación).

Para saber más

Puede comprar un módulo que se desconecte a la alimentación cuando Raspberry Pi se apague en *http://www.pi-supply.com/*.

1.15 Instalar el módulo de cámara de Raspberry Pi

Problema

Desea utilizar el módulo de cámara de Raspberry Pi (véase la Figura 1.14).

Solución

El módulo de cámara de Raspberry Pi (Figura 1.14) está unido a Raspberry Pi con un cable de cinta.

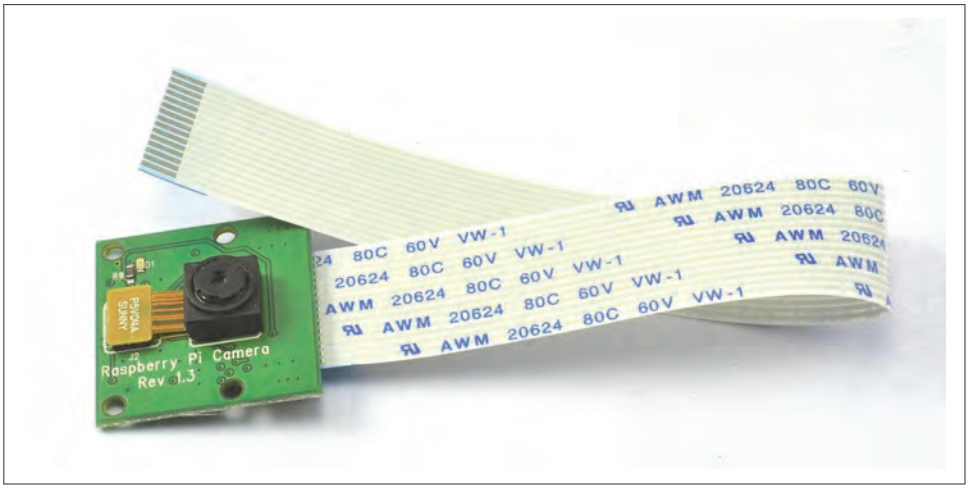


Figura 1.14. *Módulo de cámara de Raspberry Pi.*

Este cable se conecta a un conector especial entre el audio y las tomas HDMI en la Raspberry Pi 2. En el modelo B original de la Raspberry Pi, el conector está justo detrás de la toma Ethernet. Para ajustarlo, tire hacia arriba de las palancas a ambos lados del conector para que se desbloquee y luego presione el cable en la ranura, con los conectores planos del cable lejos de la toma Ethernet. Presione las dos palancas del conector hacia abajo para bloquear el cable (Figura 1.15).

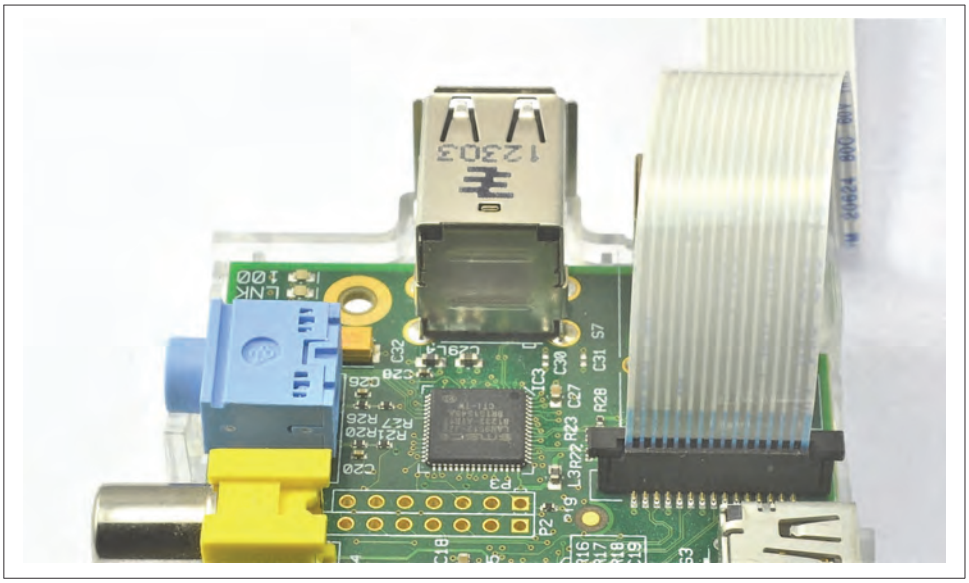


Figura 1.15. *Montaje de un módulo de cámara de Raspberry Pi unido a una Raspberry Pi modelo B.*

El embalaje del módulo de cámara afirma que es sensible a la electricidad estática. Antes de manipularla, toque algo que esté en el suelo como la tapa metálica de su ordenador.

El módulo de la cámara requiere algo de configuración de *software*. La forma más fácil de configurarlo es usar `raspi_config`. Para ejecutar `raspi-config` introduzca el siguiente comando en una sesión del terminal:

```
$ sudo raspi-config
```

Verá la opción Enable Camera (Figura 1.16).

Hay dos comandos disponibles para capturar imágenes fijas y vídeos: `raspiStill` y `raspivid`.

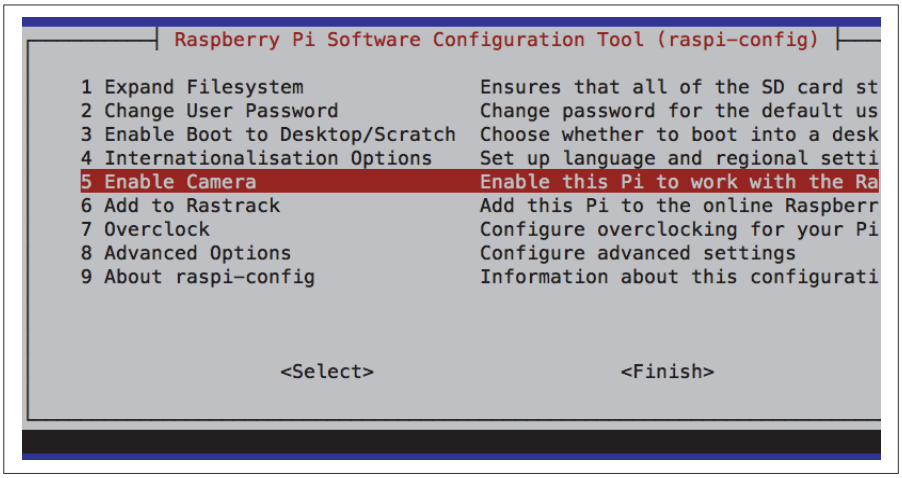


Figura 1.16. *La herramienta de configuración raspi-config actualizada.*

Para capturar una imagen fija utilice el comando `raspistill` como se muestra aquí:

```
$ raspistill -o image1.jpg
```

Aparecerá una pantalla de vista previa durante unos cinco segundos y luego hará la fotografía y la almacenará en el archivo *image1.jpg* del directorio actual.

Para capturar un vídeo, utilice el comando `raspivid`:

```
$ raspivid -o video.h264 -t 10000
```

El número que aparece en el extremo es la duración de la grabación en milisegundos, en este caso, 10 segundos.

Observaciones

Tanto `raspstill` como `raspivid` tienen un gran número de opciones. Si escribe cualquier comando sin ningún parámetro, el texto de ayuda mostrará las opciones que estén disponibles.

El módulo de la cámara tiene capacidad para imágenes de alta resolución y para grabación de vídeo. Estas son algunas de las características de la cámara:

- Sensor de 5 megapíxeles
- Lentes f/2 de foco fijo
- Resolución 1920×1080
- Vídeo 1080p, 30fps

Una alternativa al módulo de la cámara es utilizar una cámara web USB (véase el Capítulo 8.3).

Para saber más

La documentación de RaspiCam incluye `raspstill` y `raspivid`.

1.16 Usar *bluetooth*

Problema

Quiere usar *bluetooth* con Raspberry Pi.

Solución

Conecte un adaptador *bluetooth* USB a Raspberry Pi e instale el *software* Bluetooth de apoyo.

No todos los adaptadores *bluetooth* son compatibles con Raspberry Pi. La mayoría lo son, pero para estar seguro, compre una marca que trabaje con Raspberry Pi. La Figura 1.17 muestra una Raspberry Pi 2 equipada con un adaptador *bluetooth* USB (el más cercano a la cámara) y un adaptador wifi USB.

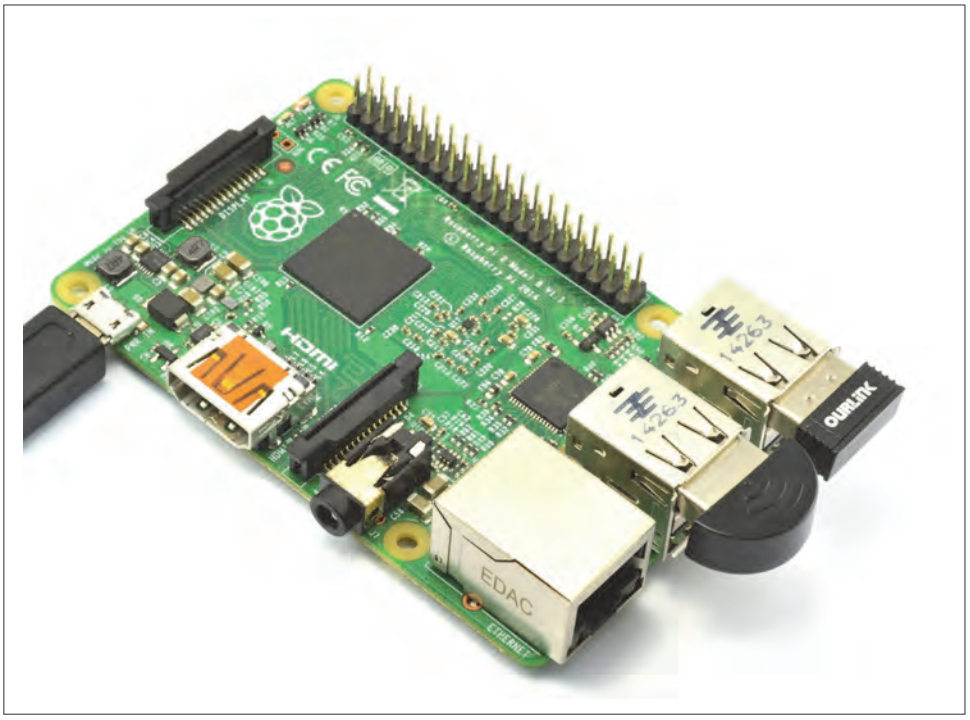


Figura 1.17. *Raspberry Pi 2 con adaptadores USB para bluetooth y wifi.*

Para instalar el *software* necesario para admitir *bluetooth*, introduzca los siguientes comandos:

```
$ sudo apt-get update
$ sudo apt-get install bluetooth bluez-utils blueman bluez
$ sudo usermod -G bluetooth -a pi
```

Estos comandos deben funcionar para todos los adaptadores *bluetooth* compatibles con Raspberry Pi.

Conecte el adaptador *bluetooth* y reinicie Raspberry Pi (Capítulo 1.14).

Encontrará una nueva entrada en el menú de inicio de Raspbian en la sección *Preferencias* llamada *Administrador de Bluetooth*. Abra esta herramienta y haga clic en *Buscar* para buscar dispositivos *bluetooth* (Figura 1.18). Asegúrese de que los dispositivos *bluetooth* estén visibles.

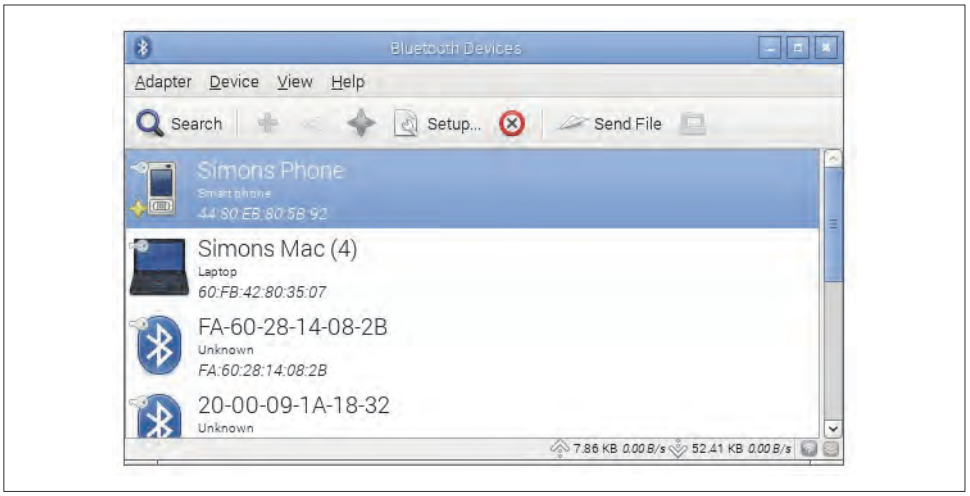


Figura 1.18. *Administración de dispositivos bluetooth.*

Observaciones

Desde el administrador de *bluetooth* puede vincularse con otros dispositivos *bluetooth*, enviarles archivos y configurar la visibilidad de su Raspberry Pi con los demás dispositivos *bluetooth*.

Para saber más

Para obtener una lista de los adaptadores Bluetooth compatibles con Raspberry Pi, vea *http://elinux.org/RPi_USB_Bluetooth_adapters*.

CAPÍTULO 2

Redes

2.1 Introducción

Raspberry Pi está diseñada para estar conectada a Internet. Su capacidad para comunicarse a través de Internet es una de sus características principales y abre todo tipo de usos posibles, incluyendo la automatización del hogar, servidores web, monitoreo de red, etc.

La conexión se puede llevar a cabo mediante un cable Ethernet (al menos en el caso del modelo B), o se puede usar un módulo wifi USB para proporcionar una conexión de red.

Tener una conexión a Internet en Raspberry Pi también significa que puede conectarse a ella de forma remota desde otro ordenador. Esto es muy útil en situaciones en las que la propia Raspberry Pi es inaccesible y no tiene teclado, ratón ni monitor conectado a ella.

En este capítulo se le darán pautas para conectar su Raspberry Pi a Internet y para controlarla remotamente a través de una red.

2.2 Conectar a una red por cable

Problema

Desea conectar su Raspberry Pi a Internet mediante una conexión por cable.

Solución

En primer lugar, si usted tiene un modelo A, A+, o Zero, de la Raspberry Pi no tendrá conector RJ45 para Ethernet. En este caso, la mejor opción para el acceso a Internet es el uso de un adaptador USB inalámbrico (véase el Capítulo 2.6).

Si usted tiene el modelo B de la Raspberry Pi, conecte un cable de conexión Ethernet en su enchufe RJ45 y luego conecte el otro extremo a la toma correspondiente en la parte posterior de su *router*. La Figura 2.1 muestra una tarjeta Raspberry Pi 1 original, donde los ledes de red están junto a la toma de audio. En una Raspberry Pi 2, los ledes se encuentran en el mismo enchufe Ethernet.

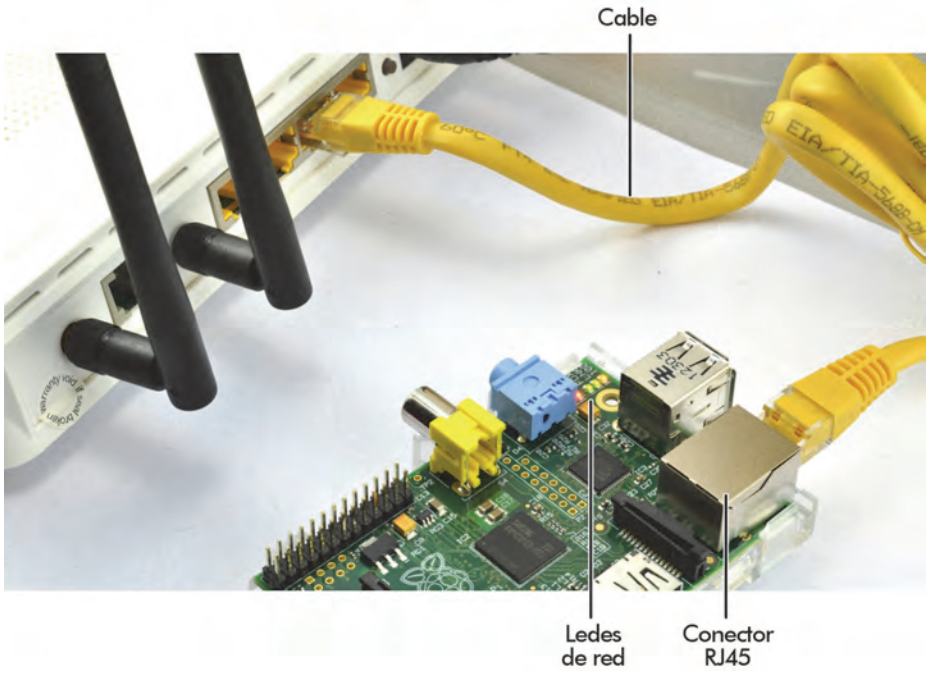


Figura 2.1. *Conectar Raspberry Pi a un concentrador.*

Los ledes de la red de su Raspberry Pi deben comenzar inmediatamente a parpadear cuando Raspberry Pi se conecta a la red.

Observaciones

Raspbian está preconfigurado para conectarse a cualquier red utilizando el protocolo de configuración dinámica de *host* (DHCP). Se le asignará automáticamente una dirección IP, siempre y cuando el DHCP no esté desactivado en la red.

Si los ledes de red en su Raspberry Pi no se encienden cuando se conecta el *router*, compruebe que está utilizando la toma Uplink RJ45 en el router o pruebe con otro cable.

Si los ledes parpadean, pero no se conecta a Internet su Raspberry Pi utilizando un navegador, compruebe que el DHCP está activado en la consola de gestión de red. Busque una opción como la que se muestra en la Figura 2.2.

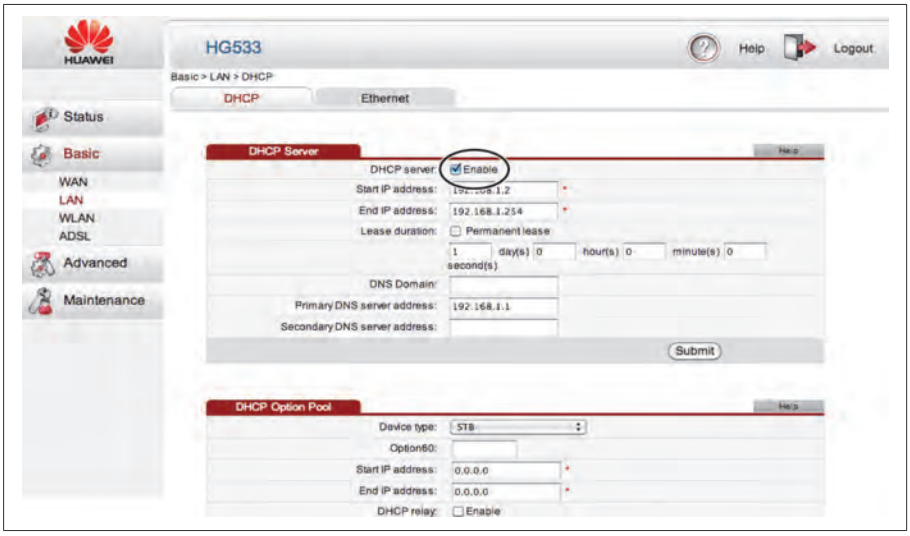


Figura 2.2. *Activar el DHCP en el router.*

Para saber más

Para conectarse a una red inalámbrica consulte el Capítulo 2.6.

2.3 Encontrar la dirección IP

Problema

Quiere saber la dirección IP de su Raspberry Pi para comunicarse con él, ya sea conectándolo como un servidor web, intercambiando archivos o controlándolo de forma remota con SSH (Capítulo 2.8) o VNC (Capítulo 2.9).

Una dirección IP es un número de cuatro partes que identifica la interfaz de red de un equipo de manera única dentro de una red. Cada parte está separada por un punto.

Solución

Para encontrar la dirección IP de su Raspberry Pi ejecute este comando en una ventana de terminal:

```
$ hostname -I
192.168.1.16
```

Esta es la dirección IP de su Raspberry en su red doméstica.

Observaciones

Raspberry Pi puede tener más de una dirección IP (es decir, una para cada conexión de red). Así que, si usted tiene una conexión por cable y una conexión inalámbrica a su Pi, tendrá dos direcciones IP. Normalmente, sin embargo, tendría que conectarlo solo mediante un método u otro, pero no ambos. Para ver todas las conexiones de red, utilice el comando `ifconfig`:

```
$ sudo ifconfig

eth0        Link encap:Ethernet  HWaddr b8:27:eb:d5:f4:8f
            inet addr:192.168.1.16 Bcast:192.168.255.255  Mask:255.255.0.0
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:1114 errors:0 dropped:1 overruns:0 frame:0
            TX packets:1173 errors:0 dropped:0 overruns:0
            carrier:0 collisions:0 txqueuelen:1000
            RX bytes:76957 (75.1 KiB)  TX bytes:479753 (468.5 KiB)

lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0
            carrier:0 collisions:0 txqueuelen:0
            RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0       Link encap:Ethernet  HWaddr 00:0f:53:a0:04:57
            inet addr:192.168.1.13 Bcast:192.168.255.255  Mask:255.255.0.0
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:38 errors:0 dropped:0 overruns:0 frame:0
            TX packets:28 errors:0 dropped:0 overruns:0
            carrier:0 collisions:0 txqueuelen:1000
            RX bytes:6661 (6.5 KiB)  TX bytes:6377 (6.2 KiB)
```

En cuanto a los resultados de la ejecución de `ifconfig`, puede ver que Pi está conectado mediante una conexión por cable (`eth0`) con una dirección IP de 192.168.1.16, y una inalámbrica (`wlan0`) con una dirección IP de 192.168.1.13. La interfaz de red *lo* es una interfaz virtual que permite al ordenador comunicarse con ella misma.

Otra manera de encontrar la dirección IP es conectarlo a la consola de administración del router, encontrar la sección LAN y luego consultar la tabla IP. Debe haber un dispositivo de la lista llamado *raspberrypi* con su dirección IP al lado.

Para saber más

Wikipedia tiene todo lo que desea saber sobre las direcciones IP.

2.4 Configurar una dirección IP estática

Problema

Desea configurar la dirección IP de su Raspberry Pi para que no cambie.

Solución

Para configurar la dirección IP de su Raspberry Pi, tanto si el uso de la red es por cable como inalámbrica, necesita editar el archivo de configuración */etc/network/interfaces*.

Si ve el archivo */etc/network/interfaces* utilizando el siguiente comando:

```
$ more /etc/network/interfaces
```

debe ser algo como esto:

```
auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

Esto le está diciendo que Raspbian tiene conocimiento de tres interfaces de red en su Raspberry Pi, cada una comienza con la palabra `iface`.

`lo`

: Loopback. Puede pasar esto por alto.

`eth0`

: Una conexión de red mediante la toma Ethernet.

`wlan0`

: Interfaz de red con un adaptador wifi USB o con *hardware* wifi incorporado de Raspberry Pi 3.

Su Raspberry Pi tendrá una dirección IP diferente para cada conexión de red. En este ejemplo acaba de hacer la dirección IP de la interfaz Ethernet estática. Si desea hacer lo mismo para la interfaz wifi, en su lugar, edite la entrada en el archivo *interfaces*.

Para editar el archivo, escriba el siguiente comando:

```
$ sudo nano /etc/network/interfaces
```

En primer lugar, decida una dirección IP para usar. Es necesario escoger una que a la vez no sea utilizada por otro equipo en la red dentro del rango permitido de direcciones IP por su router. En este caso, se utilizará 192.168.1.116.

Modifique el contenido del archivo, cambiando la palabra `dhcp` por `static` y añadiendo las siguientes líneas:

```
address 192.168.1.116
netmask 255.255.255.0
gateway 192.168.1.1
```

Con el archivo cambiado como se muestra aquí, la dirección IP estática 192.168.1.116 ha sido asignada a la interfaz `eth0`.

```
auto lo

iface lo inet loopback
iface eth0 inet static
   address 192.168.1.116
   netmask 255.255.255.0
   gateway 192.168.1.1

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

Para la mayoría de las redes, la configuración de la máscara de red debe establecerse en `255.255.255.0` y la puerta de enlace debe ser la dirección IP de su propio *router* de casa. Esta será la misma que la dirección IP que utiliza para conectarse a la consola de administración de su *router*.

Después de editar y guardar el archivo, ejecute los siguientes comandos para borrar todas las entradas DHCP que existan y reinicie su Pi para que los cambios surtan efecto.

```
$ sudo rm /var/lib/dhcp/*
$ sudo reboot
```

Observaciones

Las direcciones IP internas son algo como 192.168.1.116, donde se cambia solo el último número de cada uno de los diferentes ordenadores. Otro formato común para las direcciones IP internas es 10.0.0.16.

Para saber más

Wikipedia tiene todo lo que desea saber sobre las direcciones IP.

2.5 Ajustar el nombre de Raspberry Pi

Problema

Desea cambiar el nombre de su Raspberry Pi para que no se llame simplemente "raspberrypi."

Solución

Cambiar el nombre de su Pi es bastante sencillo. Hay solo dos archivos que necesitan cambios.

Primero, edite el archivo */etc/hostname*. Puede hacerlo abriendo una ventana de terminal y escribiendo el comando:

```
$ sudo nano /etc/hostname
```

Reemplace "raspberrypi" por el nombre que elija. Debe ser una sola palabra, sin ningún tipo de puntuación o caracteres inusuales (incluyendo el carácter _.)

En segundo lugar, abra el archivo */etc/hosts* en un editor usando el comando:

```
$ sudo nano /etc/hosts
```

Se verá algo así:

```
127.0.0.1         localhost
::1               localhost ip6-localhost ip6-loopback
fe00::0           ip6-localnet
ff00::0           ip6-mcastprefix
ff02::1           ip6-allnodes
ff02::2           ip6-allrouters

127.0.1.1         raspberrypi
```

Cambie el texto del final, que utiliza el nombre antiguo ("raspberrypi"), por el nuevo nombre.

Reinicie Pi y verá que el nombre ha cambiado cuando aparezca en la red desde otro ordenador.

Observaciones

Cambiar el nombre de su Pi puede ser muy útil, especialmente si usted tiene más de uno conectado a la red.

Para saber más

Consulte el Capítulo 2.4 para cambiar la dirección IP de su Raspberry Pi.

2.6 Configurar una conexión inalámbrica

Problema

Desea conectar su Raspberry Pi a Internet a través de un adaptador inalámbrico USB.

Solución

Si tiene la versión más reciente de Raspbian, configurar el wifi es muy fácil. Solo tiene que conectar un adaptador wifi USB y luego hacer clic en el icono de red en la parte superior derecha de la pantalla (Figura 2.3). Se le presentará una lista de redes wifi. Seleccione su red y se le pedirá que introduzca la *Pre Shared Key* (contraseña). Introduzca la contraseña, y, después de un rato, el icono Network cambiará al símbolo estándar de wifi y se conectará.

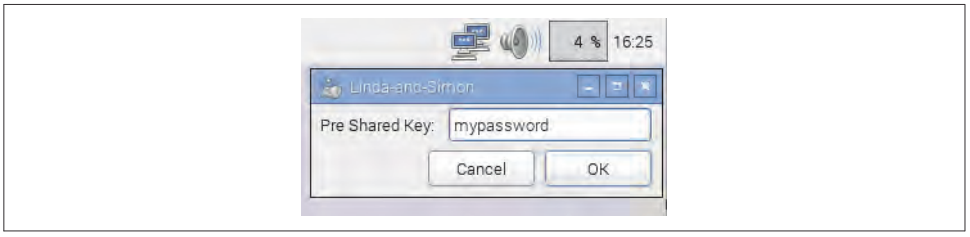


Figura 2.3. *Conectar a una red wifi.*

Si usted tiene una distribución más antigua de Raspbian, tendrá que utilizar la opción `WiFi Config` con un acceso directo en el escritorio. Si no está utilizando una distribución reciente, debe actualizarla de todos modos (véase el Capítulo 1.5).

Si tiene Raspberry Pi 3, ya lleva el *hardware* WiFi incorporado. Si usted tiene una Raspberry Pi más vieja, conecte un adaptador inalámbrico USB compatible (la mayoría son compatibles) en una de las tomas USB de su Raspberry Pi e inicie la utilidad `WiFi Config` (Figura 2-4). Encontrará la herramienta de configuración de wifi debajo de la sección Preferencias en el menú de inicio de Raspberry. Haga clic en el botón Scan para buscar los puntos de acceso. Haga doble clic en el punto de acceso (para su router) al que desea unirse y luego introduzca la contraseña en el campo PSK.

Por último, haga clic en Connect para unirse a la red.

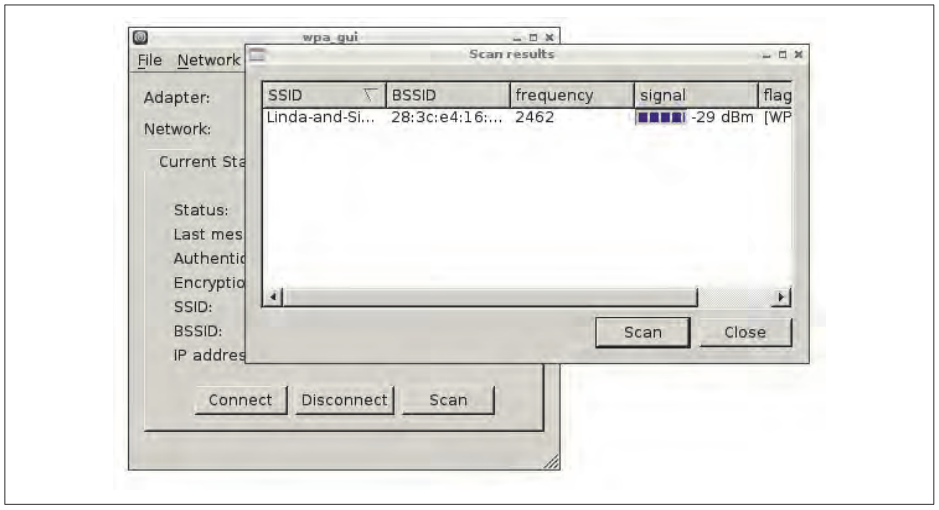


Figura 2.4. *Detectando las redes.*

Observaciones

Los adaptadores WiFi USB pueden utilizar una gran cantidad de energía, por lo que si ve que su Pi se reinicia inesperadamente o no arranca correctamente, puede que tenga que utilizar un suministro de energía más potente. Busquen una alimentación que sea de 1,5 A o más.

Si está utilizando también un teclado y un ratón, es posible que se le acaben las conexiones USB. En ese caso, la respuesta es un concentrador USB. La elección de un concentrador con su propia fuente de alimentación ayudará con el problema de la energía.

Si está utilizando su Raspberry Pi como un centro multimedia (véase el Capítulo 4.2), habrá una página de configuración que le permita conectarlo a la red mediante WiFi.

También puede configurar una conexión inalámbrica utilizando solo la línea de comandos. Para ello, primero debe editar el archivo */etc/network/interfaces* con el comando:

```
$ sudo nano /etc/network/interfaces
```

A continuación, busque la sección del archivo relativo a la interfaz *wlan0* y cámbielo para que sea:

```
allow-hotplug wlan0
iface wlan0 inet dhcp
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
```

La primera línea especifica que la conexión WiFi debe iniciarse automáticamente cuando se inserte un dongle WiFi USB. La segunda especifica que Raspberry Pi debe asignar la dirección IP usando DHCP. Si desea utilizar una dirección IP estática,

sustituya la palabra *dhcp* por *static* y añada las líneas que se describen en el Capítulo 2.4 para asignar una dirección IP estática.

La última línea especifica la ubicación del archivo solicitante. Este es el archivo que contiene realmente el SSID (nombre de la red) y el PSK (contraseña) para su red inalámbrica. Así, la próxima vez editará el archivo usando el comando:

```
$ sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Modifique el archivo ajustando los valores de `ssid` y `pskfoobar` para su red inalámbrica.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
        ssid="My-Network-Name"
        psk="My-password"
        proto=RSN
        key_mgmt=WPA-PSK
        pairwise=TKIP
        auth_alg=OPEN
}
```

Para que los cambios realizados surtan efecto, reinicie su Raspberry Pi.

Para saber más

Consulte el Capítulo 2.2 para las conexiones por cable. Si quiere conocer los adaptadores WiFi compatibles con Raspberry, consulte *http://elinux.org/RPi_VerifiedPeripherals*.

Para más información sobre cómo configurar una red por cable, véase el Capítulo 2.2.

2.7 Conectar un cable de consola

Problema

No hay ninguna conexión de red disponible, pero quiere controlar de forma remota su Raspberry Pi desde otro ordenador.

Solución

Utilice un cable de consola para conectarlo a Raspberry Pi.

Los cables de consola son ideales para una Pi que va a ser utilizada *headless*, es decir, sin teclado, ratón ni monitor. El cable de consola mostrado en la Figura 2.5 está disponible en Adafruit.

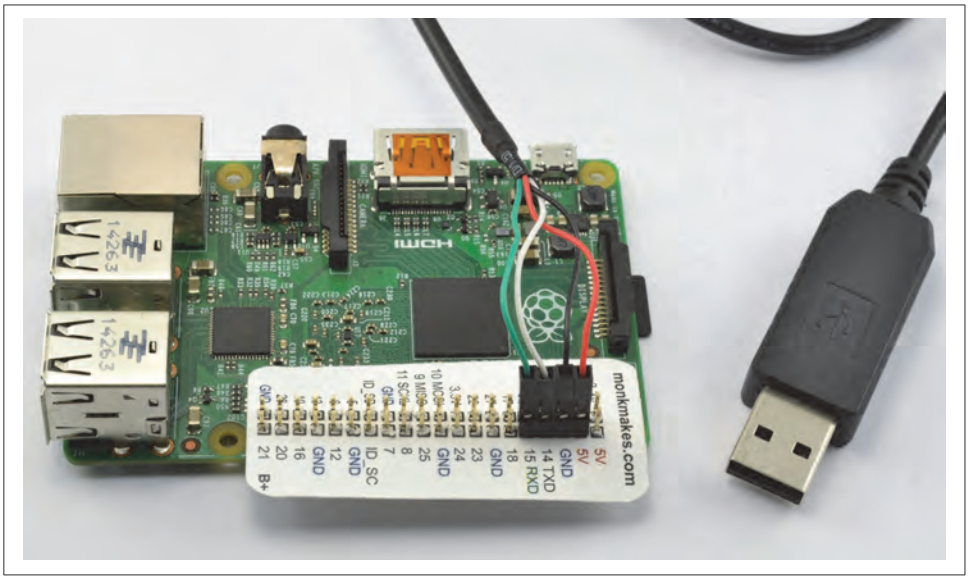


Figura 2.5. *Cable de consola.*

Conecte el cable de la siguiente manera:

1. Conectar el cable rojo (5 V) al pin de 5 V, el pin del borde izquierdo del encabezado GPIO.
2. Conectar el cable negro (GND) a GND, el siguiente pin de la izquierda.
3. Conectar el cable blanco (RX) al GPIO 14 (TXD), a la izquierda del cable negro.
4. Conectar el cable verde (RX) a 15 (RXD), a la izquierda del cable blanco.

Si se utiliza un cable diferente, los colores de los cables pueden ser diferentes, así que siempre consulte la documentación de los cables o podrá dañar su Raspberry Pi.

Tenga en cuenta que el cable USB también proporciona 5 V en el cable rojo con suficiente potencia para la misma Pi, pero no si tiene muchos dispositivos conectados.

Si usted es usuario de Windows o Mac OS X, tendrá que instalar drivers para el cable USB, que se encuentren disponibles para Windows y para Mac OS X. Los usuarios de Linux, por lo general, no necesitan instalar ningún driver para estos cables.

Para conectarse a Pi desde Mac OS X, necesitará ejecutar el terminal y escribir el comando:

```
$ sudo cu -l /dev/cu.NoZAP-PL2303-00001004 -s 115200
```

El nombre del dispositivo será diferente, pero si pulsa Tab y después `cu.P`, se autocompletará. Después de la conexión pulse *Entrar*, y debería aparecer el indicador de conexión de Raspberry (Figura 2-6). El nombre de usuario y contraseña por defecto es *pi* y *raspberry*, respectivamente.

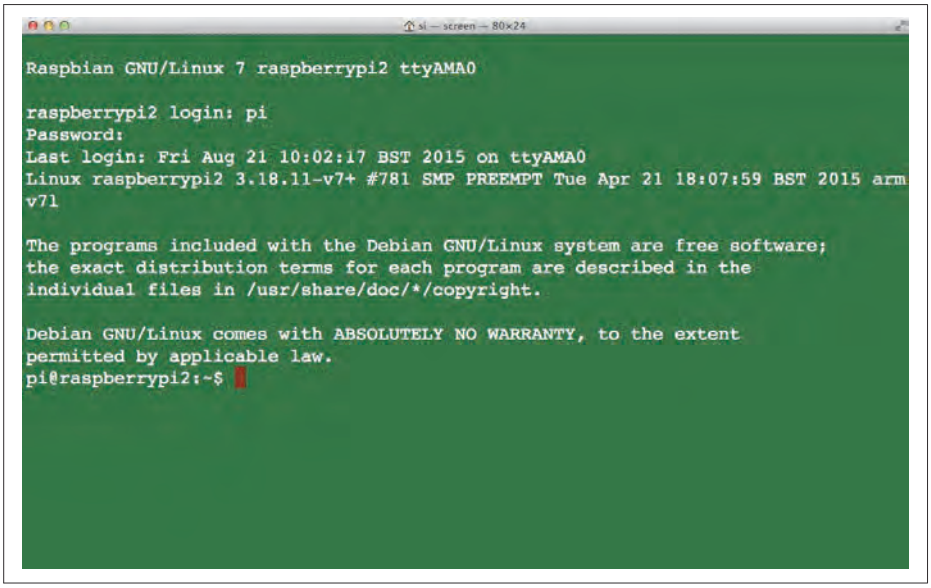


Figura 2.6. *Inicio de sesión con el cable de consola.*

Si está intentado conectarse a su Raspberry Pi desde un ordenador Windows, es necesario que descargue el *software* del terminal llamado Putty.

Cuando ejecute Putty, cambie "Connection type" a Serial y establezca la velocidad en 115200. También deberá establecer la "Serial line" para que esté el puerto COM en uso por el cable. Puede ser COM7, pero si eso no funciona, compruebe el Administrador de dispositivos de Windows.

Al hacer clic en *Open* y pulsar *Entrar*, la sesión del terminal debe comenzar con una pantalla de entrada.

Observaciones

El cable de la consola puede ser una forma muy conveniente de usar su Pi si viaja ligero, ya que proporciona energía y es una manera de controlarlo de manera remota.

El cable de consola tiene un chip al final del USB que proporciona una interfaz de USB a serie. Esto a veces (dependiendo de su sistema operativo) requiere la

instalación de drivers en el ordenador. El cable que se utiliza aquí es uno suministrado por Adafruit (código de producto: 954). Debe ser capaz de utilizar cualquier convertidor USB a serie mientras tenga los drivers necesarios para su ordenador.

Conectar las tomas de los cables en los lugares correctos es más fácil si cuidadosamente los pega juntos en el orden correcto para que se ajusten, en un bloque, al encabezado GPIO.

Encontrar la posición correcta en el encabezado GPIO es más fácil si se utiliza una plantilla GPIO como la hoja de Raspberry (vea el Capítulo 9.2).

Para saber más

Puede encontrar más información sobre el uso de consola serie en este tutorial de Adafruit. Adafruit también vende cables de consola.

2.8 Controlar Pi de forma remota con SSH

Problema

Desea conectarse a Pi de forma remota desde otro ordenador usando Secure Shell (SSH).

Solución

Antes de que pueda conectarse a su Raspberry Pi usando SSH, debe habilitar SSH. En las versiones más recientes de Raspbian puede utilizar la herramienta Configuración de Raspberry Pi (Figura 2.7) que se encuentra en el menú principal debajo de Preferencias. Simplemente marque la casilla de SSH y haga clic en OK. Se le pedirá que reinicie.

Si tiene una versión anterior de Raspbian, utilice la aplicación `raspi_config`. Comience en cualquier momento introduciendo el siguiente comando en el terminal:

```
$ sudo raspi-config
```

Desplácese hasta la opción SSH y actívela.

En las versiones más recientes de Raspbian, SSH está activado de forma automática y no hay ningún ajuste para cambiarlo.

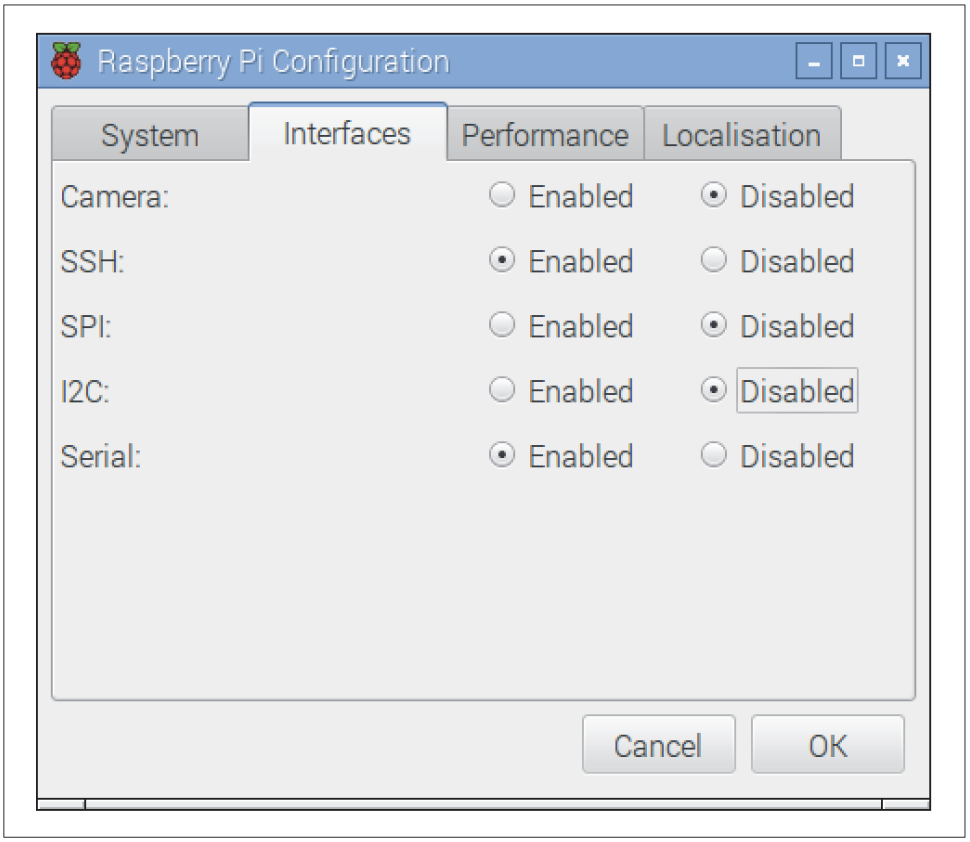


Figura 2.7. *Activar SSH usando la herramienta de configuración.*

Si está utilizando Mac OS X o tiene Linux instalado en el equipo desde el que desea conectar su Pi, lo único que tiene que hacer para conectarlo es abrir una ventana de terminal y escribir el comando:

```
$ ssh 192.168.1.16 -l pi
```

donde la dirección IP (192.168.1.16) es la dirección IP de su Pi (véase el Capítulo 2.3). Se le pedirá la contraseña y luego se conectará a su Pi (Figura 2.8).

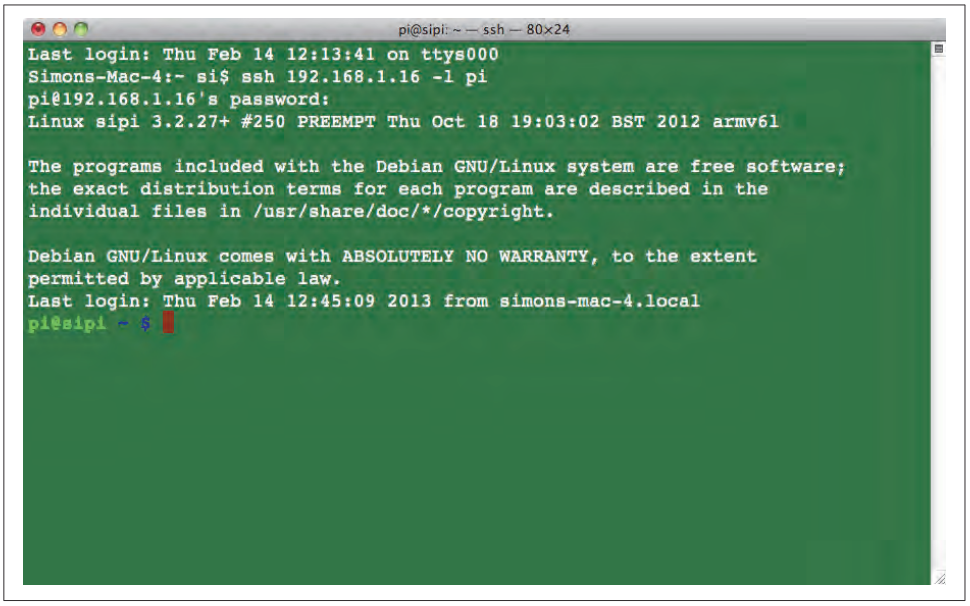


Figura 2.8. *Inicio de sesión con SSH.*

Para conectarse desde un ordenador con Windows, tendrá que utilizar Putty (Capítulo 2.7) para empezar una sesión SSH.

Observaciones

SSH es una forma muy común de conectarse a equipos remotos; cualquier comando que se pueda ejecutar en la propia Pi, puede utilizarse desde Secure Shell. También es, como su nombre indica, seguro, ya que la comunicación está encriptada.

Tal vez el único inconveniente es que se trata de una línea de comandos en lugar de un entorno gráfico. Si necesita acceso al entorno del escritorio de Raspberry Pi de forma remota, necesita usar VNC (Capítulo 2.9) o RDP (Capítulo 2.10).

Para saber más

Consulte este tutorial de Adafruit.

2.9 Controlar Pi de forma remota con VNC

Problema

Quiere tener acceso al escritorio gráfico Raspbian de su Pi desde un ordenador (Windows o Linux) o desde Mac OS X, usando VNC.

Solución

Instalar un servidor de Computación Virtual en Red (VNC).

Abra una sesión de terminal (o una sesión SSH) en Pi y ejecute los siguientes comandos:

```
$ sudo apt-get update
$ sudo apt-get install tightvncserver
```

Después de haber instalado el servidor VNC, ejecútelo con el comando:

```
$ vncserver :1
```

La primera vez que lo ejecute se le pedirá que cree una nueva contraseña para que cualquier persona que se quiera conectar de forma remota tenga que introducir la contraseña antes de obtener acceso a Pi.

Para conectarse a Pi desde un equipo remoto necesita instalar un cliente VNC. RealVNC es una opción conocida que se conecta bien con *tightvnc* y que está disponible para Windows, Linux y Mac OS X.

Cuando se ejecute el programa cliente en Mac OS X o en un PC, se le pedirá que introduzca la dirección IP del servidor VNC al que desea conectarse (la dirección IP de su Pi). Introduzca ":1" después de la dirección IP para indicar que desea conectarse al terminal número 1.

A continuación, se le pedirá la contraseña. Recuerde, es la contraseña que determinó previamente después de instalar *tightvncserver* y no es necesariamente la misma que la contraseña normal de Raspberry Pi.

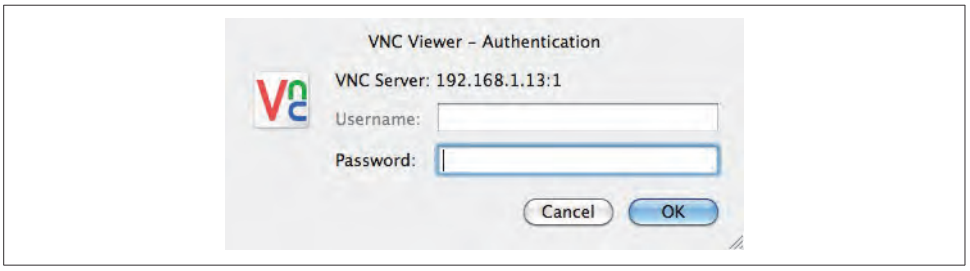


Figura 2.9. *Inicio de una sesión con VNC.*

Observaciones

Aunque se pueden hacer más cosas con SSH, a veces es útil tener acceso al entorno gráfico de Pi.

Si desea que el servidor VNC se inicie automáticamente cada vez que se reinicie su Raspberry Pi, siga estos pasos:

```
$ cd /home/pi
$ cd .config
$ mkdir autostart
$ cd autostart
$ nano tightvnc.desktop
```

Pegue lo siguiente en la ventana del editor:

```
[Desktop Entry]
Type=Application
Name=TightVNC
Exec=vncserver :1
StartupNotify=false
```

Mientras su Raspberry Pi se está configurado para iniciar sesión y arrancar en el entorno de ventanas, el servidor VNC se iniciará automáticamente cuando se reinicie.

Para saber más

Consulte el tutorial de Adafruit.

Véase también el Capítulo 2.10.

2.10 Controlar Pi de forma remota con RDP

Problema

Quiere tener acceso al escritorio gráfico Raspbian de su Pi desde un PC o Mac OS X mediante RDP.

Solución

Instalar el *software* XRDP en su Raspberry Pi introduciendo los siguientes comandos:

```
$ sudo apt-get update
$ sudo apt-get install xrdp
```

Una vez instalado el *software* se reiniciará automáticamente el servicio xrdp, lo que significa que se iniciará cuando se reinicie Raspberry Pi.

Si usted tiene Windows 7 o posterior, ya incluye un cliente RPD para conectar a su Raspberry Pi. Lo encontrará como *Conexión de Escritorio Remota* dentro de los *Accesorios de Windows* del menú de inicio. Para las versiones anteriores de Windows, puede descargarse el cliente desde: ModMyPi.

Los usuarios de Mac OS X pueden descargar el cliente Microsoft RDP para OS X desde la web de Microsoft.

Un cliente para Linux está disponible en *http://www.rdesktop.org/*.

Cuando inicie el cliente RDP se le preguntará a qué equipo quiere conectarse. Introduzca la dirección IP de su Raspberry Pi. A continuación, se le pedirá el *username* y *password* (Figura 2-10), que son los mismos que su habitual inicio de sesión en Raspberry Pi, es decir, el nombre de usuario de *pi* y la contraseña *raspberry* a menos que la haya cambiado.

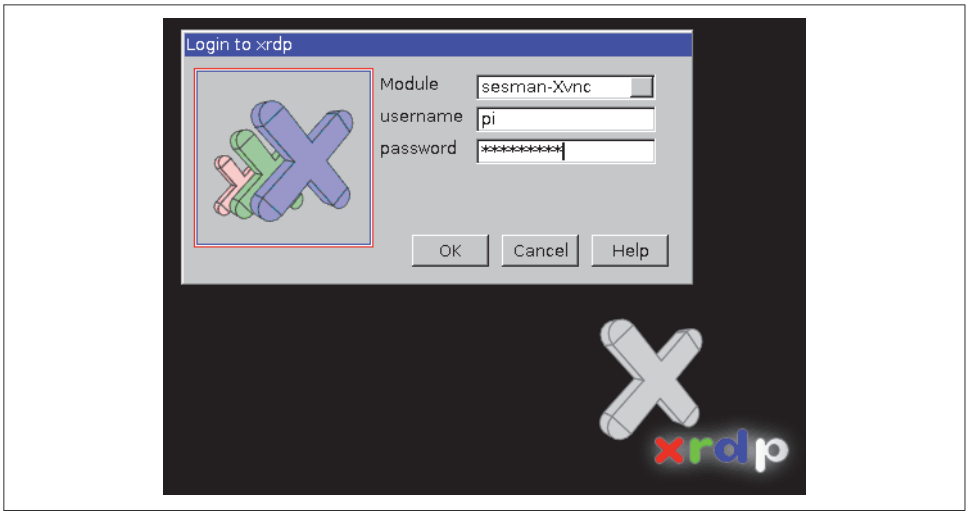


Figura 2.10. *Inicio de sesión con RDP.*

Observaciones

RDP hace el mismo trabajo que VNC, pero funciona de manera más eficiente y, por lo tanto, actualiza el contenido de la pantalla fluidamente. El único inconveniente es que para los usuarios de Mac OS X no se integra con la función Share Screen de OS X.

Para saber más

Consulte también el Capítulo 2.9.

2.11 Uso compartido de archivos en una red Mac

Problema

Quiere que su Raspberry Pi aparezca en la lista de ordenadores en el Finder de Mac OS X para que pueda conectarse a él y navegar por el sistema de archivos mediante el Finder.

Solución

El sistema operativo Mac OS X incluye una función de soporte para la visualización de archivos en el Finder a través de la red (Figura 2.11). Sin embargo, debe hacer algunos cambios de configuración en su Raspberry Pi para OS X.

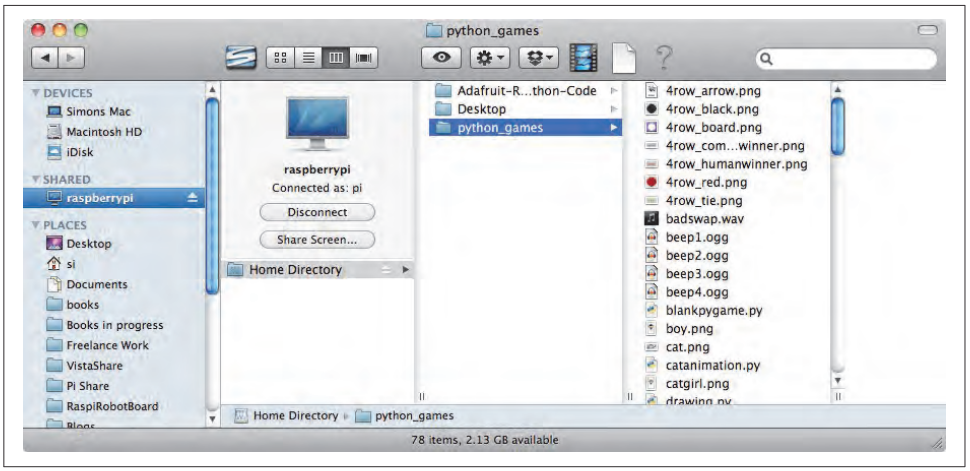


Figura 2.11. *Raspberry Pi en el Finder de Mac OS X.*

Necesita saber la dirección IP de su Raspberry Pi (Capítulo 2.3).

Ahora, en Raspberry Pi, instale `netatalk` usando el comando:

```
$ sudo apt-get install netatalk
```

Luego, de vuelta en su Mac, en el menú del Finder seleccione Go→Connect to Server e introduzca **`afp://192.168.1.16`** como dirección de servidor (pero utilice la dirección IP para su Raspberry Pi en lugar de la que se muestra aquí). A continuación, haga clic en Connect. Se le pedirá que inicie sesión. Tiene que reiniciar la Raspberry Pi antes de que llegue el mensaje de inicio de sesión.

Puede acceder usando el nombre *pi* y su contraseña, que será *raspberry* por defecto. El Finder ha de mostrarle el contenido de su directorio particular en Raspberry Pi.

Hay unos cuantos cambios de configuración que hacer en Raspberry Pi.

```
$ sudo apt-get install avahi-daemon
$ sudo update-rc.d avahi-daemon defaults
```

A continuación, escriba el comando:

```
$ sudo nano /etc/avahi/services/afpd.service
```

Pegue el siguiente código en el archivo:

```
<?xml version="1.0" standalone='no'?><!--*-nxml-*-->
<!DOCTYPE service-group SYSTEM "avahi-service.dtd">
<service-group>
   <name replace-wildcards="yes">%h</name>
   <service>
      <type>_afpovertcp._tcp</type>
      <port>548</port>
   </service>
</service-group>
```

Para activar el daemon escriba el comando:

```
$ sudo /etc/init.d/avahi-daemon restart
```

Cambie de nuevo a su Mac y deberá ver su Raspberry Pi en el Finder.

Observaciones

Ser capaz de mover archivos fácilmente entre su Mac y su Raspberry Pi es muy útil. Esto significa que puede utilizar archivos de su Pi sin tener que conectar ningún teclado separado, ratón o monitor.

También puede abrir archivos en Raspberry Pi como si estuvieran en su Mac. Esto tiene la ventaja de poder editarlos con TextMate o con su editor de textos de OS X favorito.

Si está utilizando Windows o Linux, también puede compartir archivos configurando su Raspberry Pi para trabajar como almacenamiento conectado en red (NAS); véase Capítulo 2.13.

Para saber más

Las instrucciones fueron adaptadas de este tutorial, que acredita el libro de Matt Richardson y Shawn Wallace *Getting Started with Raspberry Pi* (O'Reilly) como la fuente original.

2.12 Compartir la pantalla Pi en un Mac

Problema

Configura VNC, pero le gustaría compartir la pantalla de su Raspberry Pi como si fuera otro dispositivo Mac OS X en su red.

Solución

En primer lugar, siga el Capítulo 2.9 para instalar VNC. También necesitará el Capítulo 2.11.

A continuación, introduzca el comando:

```
$ sudo nano /etc/avahi/services/rfb.service
```

y pegue lo siguiente en el editor:

```
<?xml version="1.0" standalone='no'?>
<!DOCTYPE service-group SYSTEM "avahi-service.dtd">
<service-group>
   <name replace-wildcards="yes">%h</name>
   <service>
      <type>_rfb._tcp</type>
      <port>5901</port>
   </service>
</service-group>
```

Luego, introduzca el comando:

```
$ sudo /etc/init.d/avahi-daemon restart
```

Ahora debería ser capaz de ver la opción Share Screen mostrada en la Figura 2.12. Cuando se le pida una contraseña, utilice la contraseña que ha configurado para VNC, no su contraseña general de Raspberry.

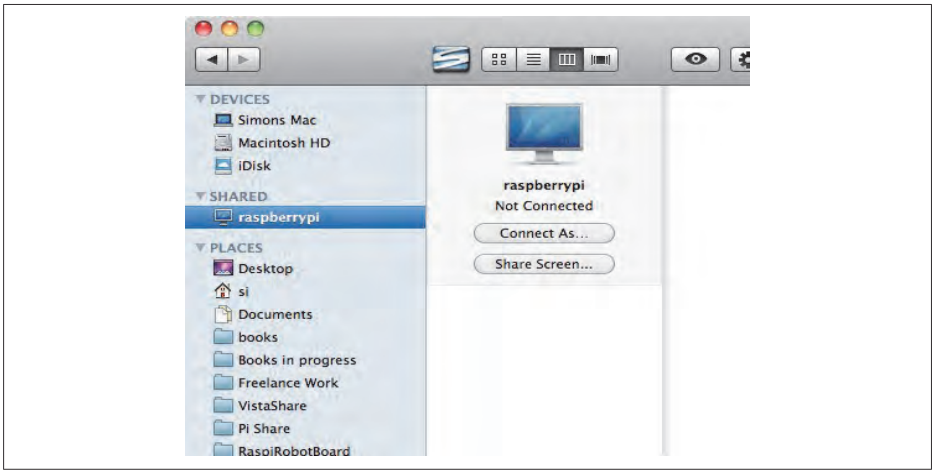


Figura 2.12. *Compartir la pantalla de Raspberry Pi con Mac OS X.*

Observaciones

Este capítulo solo añade un poco de comodidad al proceso de compartir la pantalla de su Raspberry Pi.

Si usted tiene más de una Raspberry Pi en su red, es necesario darles diferentes nombres para que pueda identificarlas en la red (Capítulo 2.5).

Si usted es un usuario de Windows o Linux, todavía puede conectarse a Raspberry Pi mediante VNC (Capítulo 2.9).

Para saber más

Las instrucciones han sido adaptadas de este tutorial, que acredita el libro de Matt Richardson y Shawn Wallace *Getting Started with Raspberry Pi* (O'Reilly) como fuente original.

2.13 Usar Raspberry Pi como almacenamiento conectado a la red (NAS)

Problema

Desea utilizar su Raspberry Pi como almacenamiento conectado a la red (NAS) mediante el acceso a una unidad USB conectada a su Raspberry Pi desde ordenadores de su red.

Solución

La solución a este problema es instalar y configurar Samba. Para ello ejecute los comandos:

```
$ sudo apt-get install samba
$ sudo apt-get install samba-common-bin
```

Ahora, conecte el disco duro USB a su Raspberry Pi. Automáticamente se montará en la carpeta */media*. Para comprobar que está ahí, utilice el comando:

```
$ cd /media
$ ls
```

La unidad deberá aparecer con el nombre que se le dio cuando se formateó. Se montará automáticamente cada vez que Raspberry Pi se inicie.

Ahora tiene que configurar Samba, por lo que la unidad puede ser compartida en la red. Para ello, primero tiene que añadir un usuario Samba (*pi*). Introduzca el siguiente comando y escriba una contraseña:

```
$ sudo smbpasswd -a pi
New SMB password:
Retype new SMB password:
Added user pi.
```

Después tendrá que hacer algunos cambios en el archivo */etc/samba/smb.conf*, por lo que introduzca el comando:

```
$ sudo nano /etc/samba/smb.conf
```

La primera línea que busca está cerca de la parte superior del archivo:

```
workgroup = WORKGROUP
```

Solo necesita cambiar esto si se va a conectar desde un ordenador Windows. Este debe ser el nombre de su grupo de trabajo (*workgroup*) de Windows. Para Windows XP, por defecto, es MSHOME; para las nuevas versiones de Windows es HOME (pero compruébelo en la red de Windows).

El siguiente cambio a realizar está más abajo del archivo en la sección de autenticación. Busque la línea:

```
# security = user
```

Quite el # de delante para activar la seguridad.

Finalmente, desplácese a la derecha hasta el final del archivo y añada las siguientes líneas:

```
[USB]
path = /media/NAS
comment = NAS Drive
valid users = pi
writeable = yes
browseable = yes
create mask = 0777
public = yes
```

Guarde el archivo y reinicie Samba introduciendo el comando:

```
$ sudo /etc/init.d/samba restart
```

Si todo va bien, la unidad USB ahora debe ser compartida en la red.

Observaciones

Para conectarse a la unidad en un Mac OS X, simplemente seleccione Go→Connect to Server desde el menú del Finder. A continuación, introduzca **`smb://raspberrypi/USB`** en el campo de dirección del servidor (Server Address). Verá un cuadro de diálogo de inicio de sesión, donde se tendrá que cambiar el nombre de usuario a *pi* (Figura 2-13).

Figura 2.13. *Conectar a NAS con Mac OS X.*

Si se está conectando a NAS desde un ordenador Windows, el procedimiento exacto puede variar dependiendo de la versión de Windows. Sin embargo, el principio básico es que en algún momento tendrá que introducir la dirección de red que debe ser *\\raspberrypi\USB* (Figura 2.14).

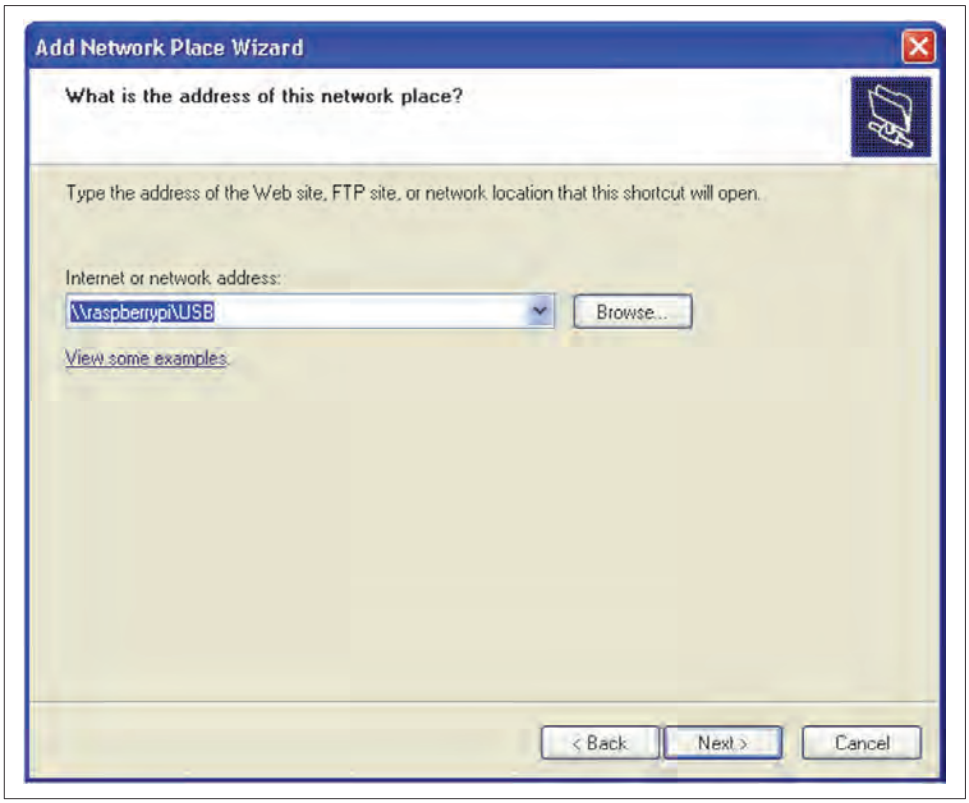


Figura 2.14. *Conectarse a NAS desde Windows.*

Se le pedirá el nombre de usuario y contraseña antes de poder utilizar el disco desde NAS (Figura 2.15).

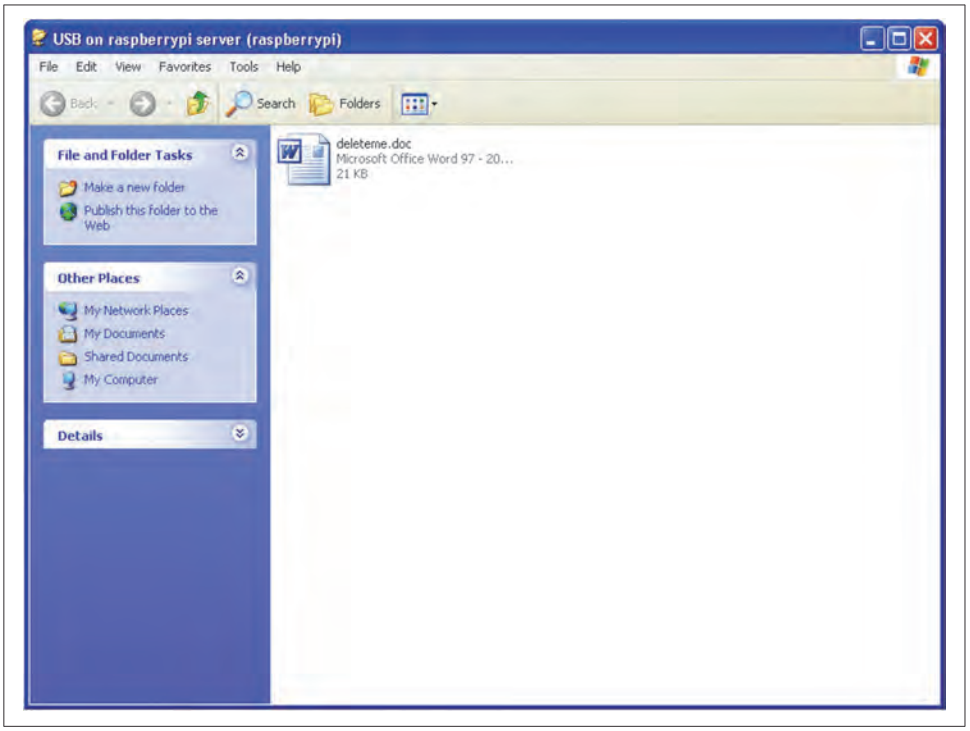


Figura 2.15. *Navegación NAS en Windows.*

Si usted es usuario de Linux, este comando montará la unidad NAS para usted:

```
$ sudo mkdir /pishare
$ sudo smbmount -o username=pi,password=raspberry //192.168.1.16/USB /pishare
```

Para saber más

Es posible que desee cambiar el nombre de la red de Raspberry Pi a algo más adecuado como *piNAS* (véase Capítulo 2.5).

2.14 Impresión en red

Problema

Desea imprimir en una impresora de red con Raspberry Pi.

Solución

Use el sistema de impresión común de Unix (CUPS).

Comience introduciendo el siguiente comando en un terminal para instalar CUPS. Esto puede tomar algo de tiempo.

```
$ sudo apt-get install cups
```

Dese privilegios de administrador para CUPS introduciendo el siguiente comando:

```
$ sudo usermod -a -G lpadmin pi
```

CUPS está configurado a través de una interfaz web. Los navegadores web Midori ni Dillo funcionan bien con las páginas de administración de CUPS, pero Iceweasel funciona muy bien. Puede instalar el navegador Iceweasel introduciendo el siguiente comando:

```
$ sudo apt-get install iceweasel
```

Inicie Iceweasel desde el grupo Internet del menú de inicio y vaya a la dirección *http://localhost:631*.

Vaya a la pestaña de administración y seleccione la opción Add Printer. Se mostrará la lista de impresoras que están en la red o conectadas directamente al puerto USB de Raspberry Pi (Figura 2-16).

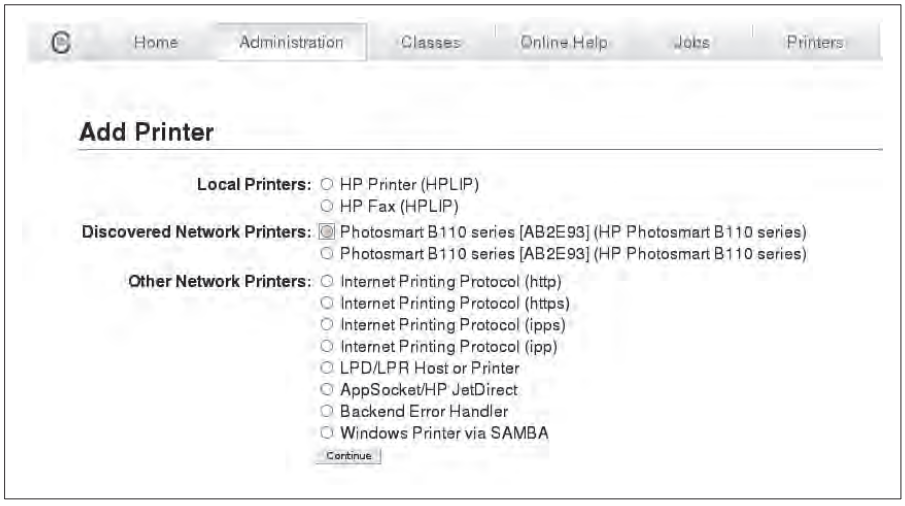


Figura 2.16. *Descubrir impresoras con CUPS.*

Siga los cuadros de diálogo para configurar la impresora.

Observaciones

Cuando haya terminado puede probar la impresora con AbiWord (véase el Capítulo 4.3). Escriba algún texto y cuando esté listo para imprimirlo deberá ver su impresora recién agregada disponible para imprimir (Figura 2-17).

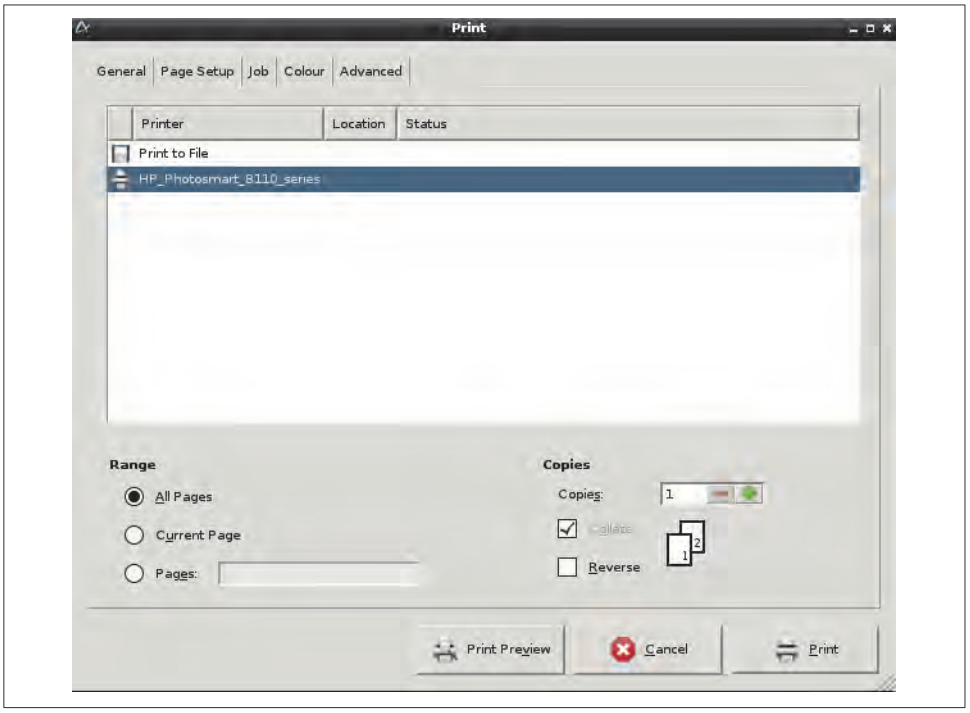


Figura 2.17. *Imprimir desde AbiWord.*

Para saber más

Consulte también la web oficial de CUPS.

CAPÍTULO 3

Sistema operativo

3.1 Introducción

Este capítulo explora muchos aspectos del sistema operativo Linux utilizado por Raspberry Pi. Esto en muchas ocasiones implica el uso de líneas de comandos.

3.2 Mover archivos de forma gráfica

Problema

Desea mover archivos a través de una interfaz gráfica como puede hacer en un Mac OS X o PC.

Solución

Utilice el Administrador de archivos.

Puede encontrar este programa en el grupo Accesorios del menú de inicio (Figura 3.1).

Con el Administrador de archivos puede arrastrar un archivo o directorio de un directorio a otro o utilizar el menú Edición para copiar un archivo en una ubicación y pegarlo en otra. Esto funciona igual que el Administrador de archivos de Windows o que el Finder de Mac OS X.

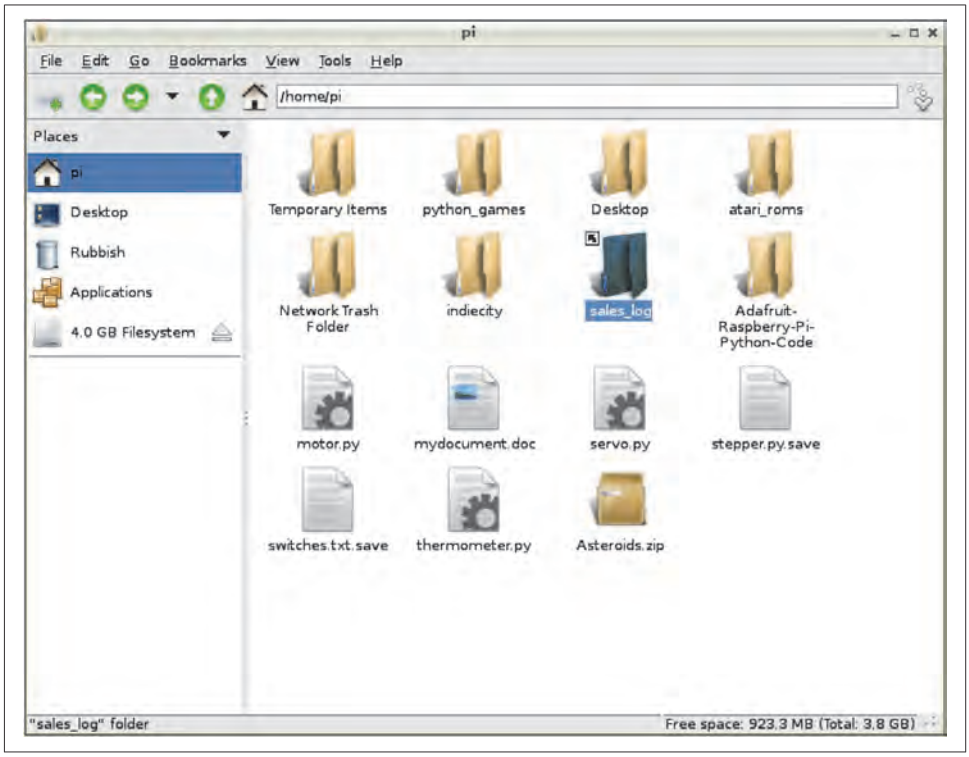


Figura 3.1. *Administrador de archivos.*

Observaciones

En el lado izquierdo del Administrador de archivos se muestran las unidades instaladas, es decir, si conecta una unidad flash USB o una unidad USB externa, aparecerá aquí.

El área central muestra los archivos de la carpeta actual, desde donde puede navegar utilizando los botones de la barra de herramientas o escribiendo una ubicación en el área de ruta de archivo en la parte superior.

Haga clic con el botón derecho en un archivo para ver las opciones que se pueden usar en ese archivo (Figura 3-2).

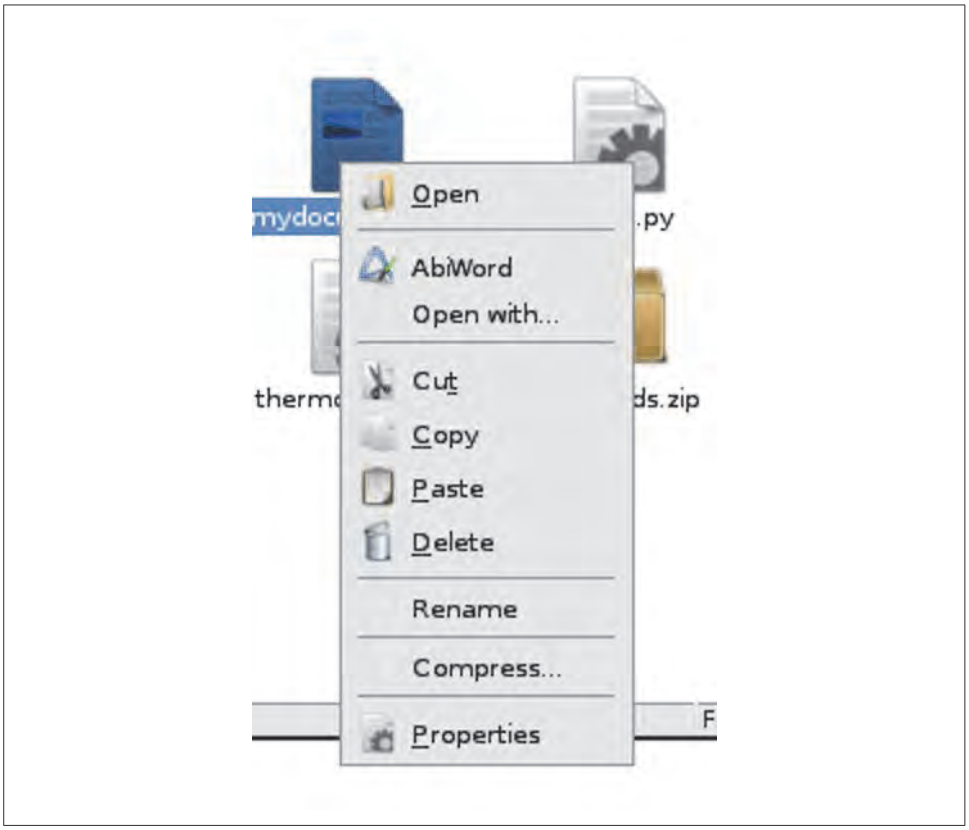


Figura 3.2. *Opciones del Administrador de archivos.*

Para saber más

Consulte también el Capítulo 3.5.

3.3 Comenzar una sesión de terminal

Problema

Cuando utiliza Raspberry Pi es necesario ejecutar comandos de texto en un terminal.

Solución

Seleccione el icono LXTerminal (parece un monitor de ordenador negro) en la parte superior del escritorio de Raspberry Pi, o seleccione la opción de menú Terminal en el menú de inicio del grupo Accesorios (Figura 3-3).

Figura 3.3. *Abrir LXTerminal.*

Observaciones

Cuando se inicia LXTerminal se establece en su directorio personal (*/home/pi*).

Puede abrir tantas sesiones de terminal como desee. A menudo es útil tener un par de sesiones abiertas en diferentes directorios para que no tenga que cambiar constantemente los directorios usando el comando `cd` (Capítulo 3.4).

Para saber más

En la siguiente sección (Capítulo 3.4) verá la estructura de directorios al navegar usando el terminal.

3.4 Navegar por el sistema de archivos mediante un terminal

Problema

Necesita saber cómo cambiar de directorio y cómo moverse por el sistema de archivos usando el terminal.

Solución

El comando principal utilizado para navegar por el sistema de archivos es `cd` (cambiar directorio). Después de `cd` debe especificar el directorio al que desea cambiar. Esto puede ser una ruta *relativa* a un directorio de dentro del directorio actual o una ruta *absoluta* a otro lugar del sistema de archivos.

Para ver cuál es el directorio actual puede utilizar el comando `pwd` (imprimir –mostrar– el directorio de trabajo).

Observaciones

Pruebe algunos ejemplos. Abra una sesión de terminal y verá un mensaje como este:

```
pi@raspberrypi ~ $
```

El mensaje de cada comando (`pi@raspberrypi ~ $`) es un recordatorio de su nombre de usuario (`pi`) y el nombre de su ordenador (`raspberrypi`). El carácter ~ es una abreviatura para su directorio de inicio (*/home/pi*). Por lo tanto, en cualquier momento puede cambiar de su directorio actual a su directorio personal de la siguiente manera:

```
$ cd ~
```

A lo largo del libro utilizo un $ al principio de cada línea donde se escribe el comando. La respuesta de la línea de comandos no será prefijada por nada; aparecerá igual que ocurre en la pantalla de Raspberry Pi.

Puede confirmar que el comando realmente estableció el directorio en el directorio de inicio mediante el comando `pwd`:

```
$ pwd
/home/pi
```

Si desea subir un nivel en la estructura de directorios, puede usar el valor especial .. (dos puntos) después del comando `cd`, como se muestra aquí:

```
$ cd ..
$ pwd
/home
```

Como ya se deduce, la ruta de acceso a un archivo o directorio en particular está formada por palabras separadas por /. Así que, la raíz de todo el sistema de archivos es /. Para acceder al directorio de inicio, dentro de /, será */home/*. Entonces, para encontrar el directorio *pi* dentro de ese utilizará */home/pi/*. El último / se puede omitir.

Las rutas también pueden ser absolutas (comenzando con / y especificando la ruta completa desde la raíz) o pueden ser relativas al directorio de trabajo actual, en cuyo caso no deben comenzar con /.

Tendrá acceso completo de lectura y escritura a los archivos en su directorio de inicio, pero una vez que se mueva a los lugares donde se guardan los archivos del sistema y las aplicaciones, su acceso a algunos archivos estará restringido solo a la lectura. Puede anular esto (Capítulo 3.12), pero se debe tener cuidado.

Compruebe la *raíz* de la estructura del directorio introduciendo los siguientes comandos:

```
$ cd /
$ ls
bin   dev  home   lost+found   mnt   proc  run   selinux   sys  usr
boot  etc  lib    media        opt   root  sbin  srv       tmp  var
```

El comando `ls` (lista) nos muestra todos los archivos y directorios bajo un / del directorio raíz. Verá que hay un directorio *home* (de inicio) en la lista; es el directorio del que acaba de llegar.

Ahora cambie a uno de esos directorios usando el comando:

```
$ cd etc
$ ls
adduser.conf                hosts.deny              polkit-1
alternatives                hp                      profile
apm                         iceweasel               profile.d
apparmor.d                  idmapd.conf             protocols
apt                         ifplugd                 pulse
asound.conf                 init                    python
```

Usted notará un par de cosas. Primero, hay una gran cantidad de archivos y carpetas en la lista, más de lo que puede caber en la pantalla a la vez. Utilice la barra de desplazamiento del lateral de la ventana del terminal para moverse hacia arriba y hacia abajo.

En segundo lugar, verá que los archivos y las carpetas tienen alguna codificación de color. Los archivos se muestran en blanco, mientras que los directorios son azules.

A menos que le guste escribir, la tecla Tab le ofrece un buen atajo. Si empieza a escribir el nombre de un archivo, al pulsar la tecla Tab la función de autocompletar intentará completar el nombre del archivo. Por ejemplo, si va a cambiar el directorio a *network*, escriba el comando `cd netw` y después pulse la tecla Tab. Debido a que *netw* es suficiente para identificar el archivo o directorio, se autocompletará al pulsar la tecla Tab.

Si lo que ha escrito no es suficiente para identificar de forma exclusiva el nombre del archivo o del directorio, pulsando otra vez la tecla Tab mostrará una lista de opciones posibles que coinciden con lo que ha escrito hasta ahora.

Por lo tanto, si se había detenido en *net* y pulsó la tecla Tab, verá lo siguiente:

```
$ cd net
netatalk/ network/
```

Puede proporcionar un argumento adicional después de `ls` para restringir las cosas que desea listar. Cambie el directorio a */etc* y luego ejecute lo siguiente:

```
$ ls f*
fake-hwclock.data  fb.modes  fstab  fuse.conf

fonts:
conf.avail  conf.d  fonts.conf  fonts.dtd

foomatic:
defaultspooler  direct  filter.conf

fstab.d:
pi@raspberrypi  /etc
$
```

El carácter `*` se llama comodín. Al especificar `f*` después de `ls`, estamos diciendo que queremos listar todo lo que empieza con *f*.

Con ayuda, los resultados primero muestran todos los archivos dentro de */etc* que empiezan con *f* y luego el contenido de todos los directorios en esa carpeta que comiencen con *f*.

Un uso común de comodines es listar todos los archivos con una cierta extensión (por ejemplo, `ls *.docx`).

Una convención en Linux (y en muchos otros sistemas operativos) es prefijar archivos que deben estar ocultos por el usuario iniciando su nombre con un punto. Los archivos o carpetas así denominados no aparecerán cuando escriba `ls` a menos que también proporcione `ls` con la opción `-a`. Por ejemplo:

```
$ cd ~
$ ls -a
.                                Desktop              .pulse
..                               .dillo               .pulse-cookie
Adafruit-Raspberry-Pi-Python-Code .dmrc               python_games
.advance                         .emulationstation    sales_log
.AppleDB                         .fltk                servo.py
.AppleDesktop                    .fontconfig          .stella
.AppleDouble                     .gstreamer-0.10      stepper.py.save
Asteroids.zip                    .gvfs                switches.txt.save
atari_roms                       indiecity            Temporary Items
.bash_history                    .local               thermometer.py
.bash_logout                     motor.py             .thumbnails
.bashrc                          .mozilla             .vnc
.cache                           mydocument.doc       .Xauthority
.config                          Network Trash Folder .xsession-errors
.dbus                            .profile             .xsession-errors.old
```

Como puede ver, la mayoría de los archivos y carpetas de su directorio de inicio están ocultos.

Para saber más

Consulte también el Capítulo 3.14.

3.5 Copiar un archivo o una carpeta

Problema

Desea copiar un archivo utilizando una sesión de terminal.

Solución

Utilice el comando `cp` para copiar archivos y directorios.

Observaciones

Por supuesto, usted puede copiar archivos utilizando el Administrador de archivos y sus opciones de copiar y pegar (Capítulo 3.2).

El ejemplo más simple de copiar en una sesión de terminal es hacer una copia de un archivo dentro de su directorio de trabajo. El comando `cp` va seguido por el archivo que se va a copiar y luego por el nombre que se le va a dar al nuevo archivo.

Por ejemplo, el ejemplo siguiente crea un archivo llamado *myfile.txt* y después hace una copia de él con el nombre *myfile2.txt*. Puede encontrar más información sobre el truco de crear un archivo usando el comando > en el Capítulo 3.9.

```
$ echo "hello" > myfile.txt
$ ls
myfile.txt
$ cp myfile.txt myfile2.txt
$ ls
myfile.txt myfile2.txt
```

Aunque en este ejemplo ambas rutas de archivo son las del directorio actual, las rutas del archivo pueden estar en cualquier parte del sistema de archivos donde tenga acceso de escritura. El siguiente ejemplo copia el archivo original en un área */tmp*, que es una ubicación para archivos temporales. No ponga nada importante en esa carpeta.

```
$ cp myfile.txt /tmp
```

Tenga en cuenta que, en ese caso, el nombre que se va a dar al nuevo archivo no se especifica, solo se especifica el directorio donde va a ir. Esto creará una copia de *myfile.txt* en */tmp* con el mismo nombre de *myfile.tmp*.

A veces, en lugar de copiar solo un archivo puede que desee copiar todo un directorio lleno de archivos y posiblemente otros directorios. Para copiar dicho directorio debe utilizar la opción `-r` (*recursivo*). Esto copiará el directorio y todo su contenido.

```
$ cp -r mydirectory mydirectory2
```

Siempre que esté copiando archivos o carpetas, si no tiene permiso, el resultado del comando le dirá eso. Tendrá que cambiar los permisos de la carpeta en la que está copiando (Capítulo 3.14) o copiar los archivos con privilegios de superusuario (Capítulo 3.12).

Para saber más

Consulte también los Capítulos 3.6 y 3.11.

3.6 Renombrar un archivo o carpeta

Problema

Desea cambiar el nombre de un archivo mediante una sesión de terminal.

Solución

Utilice el comando `mv` para renombrar archivos y directorios.

Observaciones

El comando `mv` (*mover*) se utiliza de manera similar al comando `cp`, a excepción de que el archivo o carpeta que se está moviendo simplemente se renombra en lugar de hacer un duplicado.

Por ejemplo, para simplemente renombrar un archivo de *my_file.txt* a *my_file.rtf*, utilice el comando:

```
$ mv my_file.txt my_file.rtf
```

Cambiar el nombre de un directorio es igual de sencillo y no necesita la opción recursiva `-r` que usó al copiar.

Para saber más

Consulte también los Capítulos 3.5 y 3.11.

3.7 Editar un archivo

Problema

Desea ejecutar un editor desde la línea de comandos para cambiar un archivo de configuración.

Solución

Utilice el editor *nano* incluido con la mayoría de las distribuciones de Raspberry Pi.

Observaciones

Para usar nano simplemente escriba el comando `nano` seguido del nombre o ruta del archivo que quiere editar. Si el archivo no existe, se creará cuando lo guarde desde el editor. Sin embargo, esto solo ocurrirá si tiene permisos de escritura en el directorio donde intente crear el archivo.

Desde su directorio de inicio escriba el comando `nano my_file.txt` para editar o crear el archivo *nano my_file.txt*. La Figura 3.4 muestra *nano* en acción.

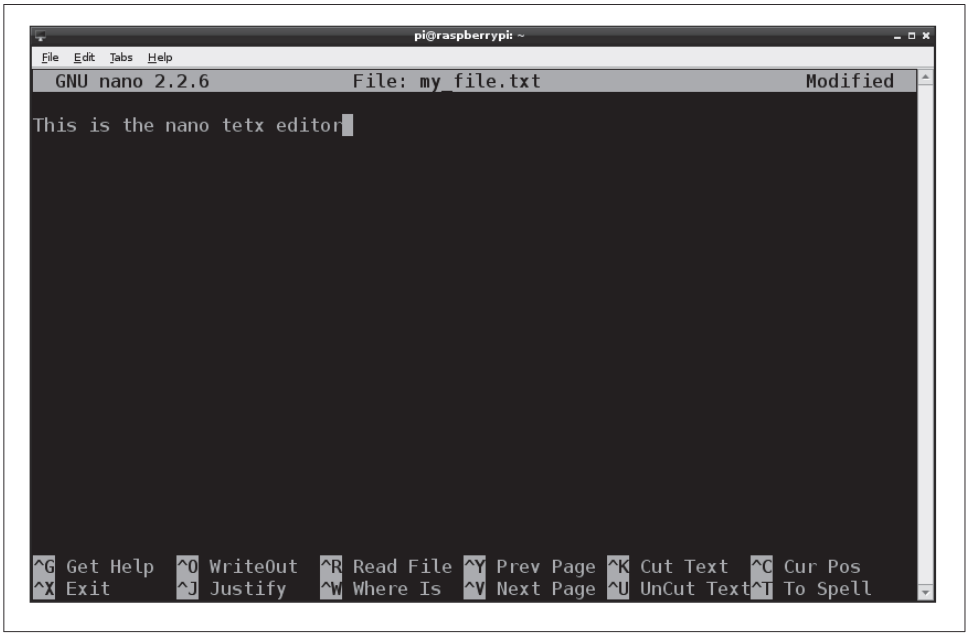


Figura 3.4. *Editar un archivo con nano.*

No puede utilizar el ratón para situar el cursor; utilice las teclas de flecha en su lugar.

El área de la parte inferior de la pantalla muestra una serie de comandos a los que puede acceder manteniendo pulsada la tecla Ctrl y la letra indicada. La mayoría de estos no son muy útiles. Los que probablemente utilice más son:

Ctrl-X

Salir. Se le pedirá que guarde el archivo antes de salir de nano.

Ctrl-V
: Siguiente página. Piense en ello como una flecha apuntando hacia abajo. Esto le permite moverse una pantalla cada vez a través de un archivo grande.

Ctrl-Y
: Página anterior.

Ctrl-W
: Dónde está. Esto le permite buscar un trozo de texto.

También hay algunas opciones del tipo cortar y pegar, pero en la práctica es más fácil utilizar el portapapeles normal desde el menú al que se accede con un clic derecho.

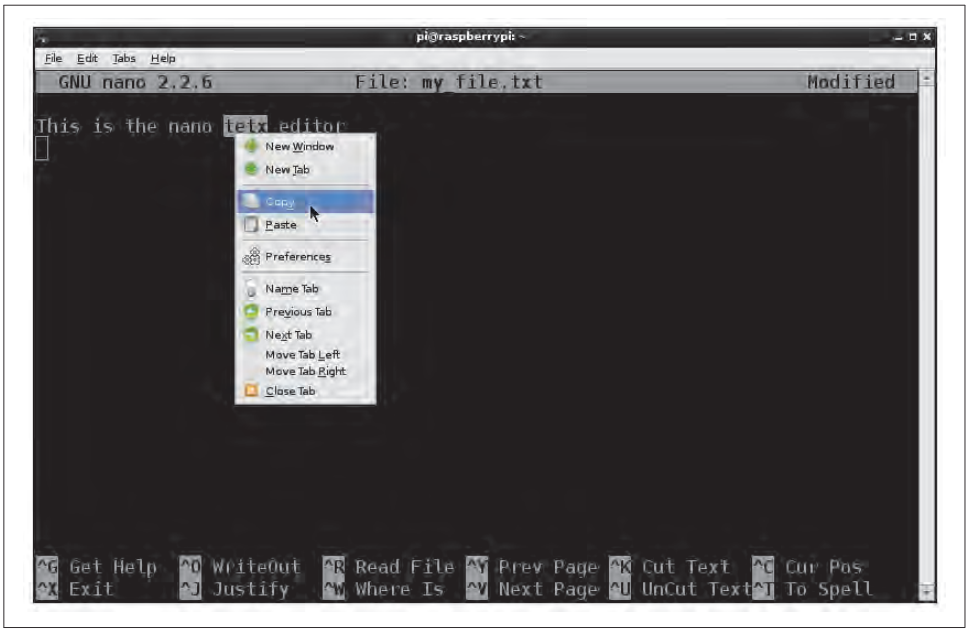


Figura 3.5. *Usar el portapapeles en nano.*

Usar este portapapeles también le permite copiar y pegar texto entre otras ventanas de su navegador.

Cuando esté listo para guardar los cambios en el archivo y salir de nano, utilice el comando Ctrl-X. Escriba **Y** para confirmar que desea guardar el archivo. nano muestra el nombre del archivo como el nombre predeterminado para guardarlo. Pulse *Intro* para guardar y salir.

Si desea abandonar los cambios que ha realizado, introduzca N en lugar de Y.

Para saber más

Los editores son cuestión de gustos personales. Muchos editores están disponibles para Linux y funcionan perfectamente en Raspberry Pi. El editor *vim* (vi mejorado) tiene muchos fans en el mundo de Linux. Está incluido también en las distribuciones populares de Raspberry Pi. Sin embargo, no es un editor fácil para principiantes. Puede ejecutarlo de la misma manera que nano, pero utilizando el comando `vi` en lugar de `nano`. Hay más detalles sobre el uso de vim en *http://newbiedoc.sourceforge.net/text_editing/vim.html.en*.

3.8 Visualizar el contenido de un archivo

Problema

Desea visualizar el contenido de un archivo pequeño sin editarlo.

Solución

Use el comando `cat` o `more` para ver el archivo. Por ejemplo:

```
$ more myfile.txt
This file contains
some text
```

Observaciones

El comando `cat` muestra todo el contenido del archivo, aunque sea más largo de lo que cabría en la pantalla.

El comando `more` solo muestra una pantalla de texto a la vez. Pulse la barra espaciadora para mostrar la siguiente pantalla.

Para saber más

También puede utilizar `cat` para concatenar (unir) un número de archivos (Capítulo 3.31).

Otro comando popular relacionado con `more` es `less`. `less` es como `more` a excepción de que permite moverse hacia atrás en el archivo y también hacia delante.

3.9 Crear un archivo sin utilizar un editor

Problema

Desea crear un archivo de una línea sin tener que utilizar un editor.

Solución

Utilice los comandos `>` y `echo` para redirigir lo que escribe en la línea de comandos a un archivo. Por ejemplo:

```
$ echo "file contents here" > test.txt
$   more   test.txt
file contents here
```

El comando `>` sobrescribe un archivo existente, así que úselo con precaución.

Observaciones

Puede ser útil para crear un archivo.

Para saber más

Para usar el comando `more` para ver archivos sin usar un editor, consulte el Capítulo 3.8. Para usar `>` para capturar otros tipos de salida del sistema, consulte el Capítulo 3.30.

3.10 Crear un directorio

Problema

Desea crear un nuevo directorio utilizando el terminal.

Solución

El comando `mkdir` creará un nuevo directorio.

Observaciones

Para crear un directorio, utilice el comando `mkdir.` Pruebe el siguiente ejemplo:

```
$ cd ~
$ mkdir my_directory
$ cd my_directory
$ ls
```

Debe tener permiso de escritura en el directorio en el que intenta crear el nuevo directorio.

Para saber más

Para obtener información general sobre el uso del terminal para navegar por el sistema de archivos, véase el Capítulo 3.4.

3.11 Eliminar un archivo o un directorio

Problema

Desea eliminar un archivo o un directorio utilizando el terminal.

Solución

El comando `rm` (eliminar) borrará un archivo o directorio y su contenido. Debe ser utilizado con extrema precaución.

Observaciones

Eliminar un solo archivo es simple y seguro. El siguiente ejemplo eliminará el archivo *my_file.txt* del directorio personal:

```
$ cd ~
$ rm my_file.txt
$ ls
```

Debe tener permisos de escritura en el directorio en el que está intentando realizar la eliminación.

También puede utilizar el comodín * para eliminar archivos. Este ejemplo eliminará todos los archivos que comiencen por *my_file.* en el directorio actual:

```
$ rm my_file.*
```

También puede eliminar todos los archivos del directorio escribiendo:

```
$ rm *
```

Si desea eliminar de manera recursiva un directorio y todo su contenido, incluidos los directorios que contiene, puede utilizar la opción `-r`:

```
$ rm -r mydir
```

Al eliminar archivos desde una ventana de terminal, recuerde que no tiene la red de seguridad de una papelera de reciclaje desde la que se pueden recuperar archivos eliminados. Además, en términos generales, no se le dará la opción de confirmar; los archivos se eliminarán inmediatamente. Esto puede ser totalmente devastador si se combina con el comando `sudo` (Capítulo 3.12).

Para saber más

Consulte también el Capítulo 3.4.

Si le preocupa la eliminación accidental de archivos o carpetas, puede forzar el comando `rm` mediante la configuración de un alias de comando (Capítulo 3.35).

3.12 Realizar tareas con privilegios de superusuario

Problema

Algunos comandos no funcionan porque no tiene *suficientes privilegios*. Necesita ejecutar comandos con privilegios de superusuario.

Solución

El comando `sudo` (Super User DO) le permite realizar acciones con privilegios de superusuario. Simplemente debe prefijar el comando con `sudo`.

Observaciones

La mayoría de las tareas que desea realizar en la línea de comandos normalmente se pueden realizar sin privilegios de superusuario. Las excepciones más comunes aparecen cuando se instala un nuevo *software* o se editan archivos de configuración.

Por ejemplo, si intenta usar el comando `apt-get update`, recibirá varios mensajes de permiso denegado:

```
$ apt-get update
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)
E: Unable to lock directory /var/lib/apt/lists/
E: Could not open lock file /var/lib/dpkg/lock - open (13: Permission denied)
E: Unable to lock the administration directory (/var/lib/dpkg/), are you root?
```

El mensaje del final —`are you root?`— le da juego. Si ejecuta el mismo comando con el prefijo `sudo`, el comando funcionará correctamente:

```
$ sudo apt-get update
Get:1 http://mirrordirector.raspbian.org wheezy InRelease [12.5 kB]
Hit http://archive.raspberrypi.org wheezy InRelease
Get:2 http://mirrordirector.raspbian.org wheezy/main Sources [6,241 kB]
Hit http://archive.raspberrypi.org wheezy/main armhf Packages
Ign http://archive.raspberrypi.org wheezy/main Translation-en_GB
Ign http://archive.raspberrypi.org wheezy/main Translation-en
40% [2 Sources 2,504 kB/6,241 kB 40%]
```

Si tiene una gran cantidad de comandos para ejecutar como superusuario y no quiere tener que prefijar cada uno con sudo, puede usar el siguiente comando:

```
$ sudo sh
#
```

Tenga en cuenta cómo cambia el indicador de $ a #. Todos los comandos posteriores se ejecutarán como superusuario. Cuando desee volver a ser un usuario normal, introduzca el comando:

```
# exit
$
```

Para saber más

Para saber más sobre los permisos de archivos, véase el Capítulo 3.13. Para instalar el *software* utilizando `apt-get`, véase el Capítulo 3.17.

3.13 Entender los permisos de archivo

Problema

Ha visto caracteres extraños en algunos nombres de archivo cuando aparecen en la lista. Usted quiere saber lo que significan.

Solución

Para ver los permisos y la información de propiedad relativa a los archivos y directorios, utilice el comando `ls` con la opción `-l`.

Observaciones

Ejecute el comando `ls -l` y verá un resultado como el siguiente:

```
$ ls -l
total 16
-rw-r--r-- 1 pi pi    5 Apr 23 15:23 file1.txt
-rw-r--r-- 1 pi pi    5 Apr 23 15:23 file2.txt
-rw-r--r-- 1 pi pi    5 Apr 23 15:23 file3.txt
drwxr-xr-x 2 pi pi 4096 Apr 23 15:23 mydir
```

La Figura 3.6 muestra las diferentes secciones de la información del listado. La primera sección contiene los permisos. En la segunda columna, el número 1 indica cuántos archivos están involucrados. Este campo solo tiene sentido si la entrada de la lista es para un directorio; si es un archivo será casi siempre 1. Las dos entradas siguientes (ambos `pi`) son el propietario y el grupo del archivo. El tamaño de la entrada (la quinta columna) indica el tamaño del archivo en bytes. La fecha cambiará cada vez que se edite o cambie el archivo. Y, por último, la entrada final es el nombre real del archivo o directorio.

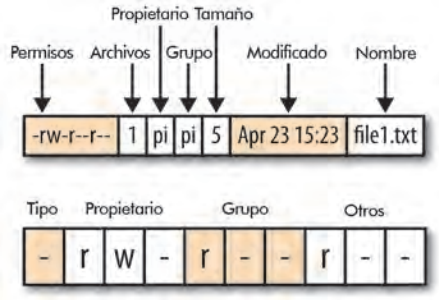


Figura 3.6. *Permisos de archivo.*

La cadena de permisos se divide en cuatro secciones (Tipo, Propietario, Grupo y Otros). La primera sección es el tipo de archivo. Si se trata de un directorio, será el carácter *d*; si es solo un archivo, la entrada será un -.

La siguiente sección de tres caracteres especifica los permisos de propietario para ese archivo. Cada carácter es un indicador que está o no. Así que si el propietario tiene permisos de lectura habrá una *r* en la primera posición del primer carácter. Si tiene permisos de escritura habrá una *w* en la segunda ranura. La tercera posición, que es – en este ejemplo, puede ser *x* si el archivo es ejecutable (un programa o script).

La tercera sección tiene los mismos tres indicadores, pero para cualquier usuario del grupo. Los usuarios pueden organizarse en grupos. Por lo tanto, en este caso, el archivo tiene un usuario *pi* y un grupo de propiedad *pi*. Si hubiera otros usuarios en el grupo *pi*, tendrían los permisos especificados ahí.

La sección final especifica los permisos para cualquier otro usuario que no sea *pi* ni del grupo *pi*.

Dado que la mayoría de la gente solo utilizará Raspberry Pi como usuario *pi*, los permisos de mayor interés están en la primera sección.

Para saber más

Para cambiar los permisos de archivo véase el Capítulo 3.14.

3.14 Cambiar los permisos de archivo

Problema

Necesita cambiar los permisos en un archivo.

Solución

El comando `chmod` se utiliza para modificar los permisos de archivo.

Observaciones

Las razones más comunes por las que puede que desee cambiar los permisos de archivo son tener que editar un archivo con permiso de solo lectura y darle a un archivo permisos de *ejecución* para que pueda ejecutarse como un programa o un script (secuencia de comandos).

El comando `chmod` le permite añadir o quitar permisos a un archivo. Hay dos sintaxis para hacer esto; una requiere el uso de octal (base 8) y el otro está basado en texto. Utilizaremos el método de texto, que es más fácil de entender.

El primer parámetro de `chmod` es el cambio que se va a realizar, y el segundo el archivo o carpeta a la que debe aplicarse. El parámetro de cambio toma la forma del ámbito de permiso (`+`, `−`, `=` para añadir, quitar y establecer, respectivamente) y, a continuación, el tipo de permiso.

Por ejemplo, el siguiente código dará derechos de ejecución (*x*) al archivo para el propietario del archivo *file2.txt*.

```
$ chmod u+x file2.txt
```

Si ahora enumeramos el directorio, podemos ver que el permiso *x* se ha añadido:

```
$  ls  -l
total 16
-rw-r--r-- 1 pi pi    5 Apr 23 15:23 file1.txt
-rwxr--r-- 1 pi pi    5 Apr 24 08:08 file2.txt
-rw-r--r-- 1 pi pi    5 Apr 23 15:23 file3.txt
drwxr-xr-x 2 pi pi 4096 Apr 23 15:23 mydir
```

Si queremos agregar permisos de ejecución para el grupo u otros usuarios, podríamos usar *g* y *o*, respectivamente. La letra *a* añadirá el permiso a todos.

Para saber más

Para obtener más información sobre los permisos de archivo consulte el Capítulo 3.13. Véase el Capítulo 3.15 para cambiar la propiedad del archivo.

3.15 Cambiar la propiedad del archivo

Problema

Quiere cambiar la propiedad de un archivo.

Solución

El comando `chown` (cambio de propietario) se utiliza para modificar la propiedad de un archivo o directorio.

Observaciones

Como hemos descubierto en el Capítulo 3.13, cualquier archivo o directorio tiene un propietario y un grupo asociado a él. Puesto que la mayoría de los usuarios de Raspberry Pi solo tendrán el usuario *pi*, no necesitamos preocuparnos de los grupos.

De vez en cuando, encontrará archivos en su sistema que han sido instalados con un usuario diferente a *pi*. Si este es el caso, puede cambiar la propiedad del archivo utilizando el comando `chown`.

Para cambiar el propietario de un archivo utilice `chown` seguido del nuevo propietario y grupo, separados por dos puntos, y después el nombre del archivo.

Probablemente verá que necesita privilegios de superusuario para cambiar la propiedad. En este caso prefije el comando con `sudo` (Capítulo 3.12).

```
$ sudo chown root:root file2.txt
$   ls   -l
total 16
-rw-r--r-- 1 pi   pi      5 Apr 23 15:23 file1.txt
-rwxr--r-- 1 root root    5 Apr 24 08:08 file2.txt
-rw-r--r-- 1 pi   pi      5 Apr 23 15:23 file3.txt
drwxr-xr-x 2 pi   pi   4096 Apr 23 15:23 mydir
```

Para saber más

Para obtener más información sobre los permisos de archivo, consulte el Capítulo 3.13. También puede consultar el Capítulo 3.14 para cambiar los permisos de archivo.

3.16 Hacer una captura de pantalla

Problema

Desea capturar una imagen de la pantalla de Raspberry Pi y guardarla en un archivo.

Solución

Instale y utilice el *software* de captura de pantalla llamado `scrot`.

Observaciones

Para instalar `scrot` ejecute el siguiente comando desde el terminal:

```
$ sudo apt-get install scrot
```

La forma más sencilla de capturar una pantalla es simplemente introduciendo el comando `scrot`. Esto tomará inmediatamente una imagen de la pantalla principal y la guardará en un archivo llamado algo así como *2013-04-25-080116_1024x768_scrot.png* dentro del directorio actual.

Alguna vez querrá una captura de pantalla para mostrar un menú que se abre o algo que generalmente desaparece. Para estas situaciones puede especificar un retraso antes de que se realice la captura usando la opción `-d`.

```
$ scrot -d 5
```

El retraso se especifica en segundos.

Si captura toda la pantalla, puede recortarla posteriormente con un *software* de edición de imágenes, como Gimp (Capítulo 4.12). Sin embargo, es más conveniente capturar directamente un área de la pantalla utilizando la opción `-s`.

Para utilizar esta opción, escriba este comando y arrastre el área de la pantalla que desea capturar con el ratón.

```
$ scrot -s
```

El nombre del archivo incluirá las dimensiones en píxeles de la imagen capturada.

Para saber más

El comando `scrot` tiene otras opciones para controlar cosas como utilizar múltiples pantallas y cambiar el formato del archivo guardado. Puede obtener más información sobre `scrot` escribiendo el siguiente comando:

```
$ man scrot
```

Para obtener más información sobre la instalación con `apt-get` véase el Capítulo 3.17.

3.17 Instalar un *software* con apt-get

Problema

Desea instalar un *software* utilizando la línea de comandos.

Solución

La herramienta más utilizada para instalar un *software* desde una sesión de terminal es `apt-get`.

El formato básico del comando que debe ejecutar como superusuario es:

```
$ sudo apt-get install <name of software>
```

Por ejemplo, para instalar el *software* del procesamiento de texto AbiWord, introduzca el comando:

```
$ sudo apt-get install abiword
```

Observaciones

El gestor de paquetes `apt-get` utiliza una lista de *software* disponible. Esta lista está incluida con la distribución del sistema operativo de Raspberry Pi que usted utiliza, pero es probable que esté desactualizada. Por lo tanto, si el *software* que intenta instalar es retornado por `apt-get` como *not found* (no encontrado), ejecute el siguiente comando para actualizar la lista:

```
$ sudo apt-get update
```

La lista y los paquetes de *software* para la instalación están todos en Internet, por lo que nada de esto funcionará si su Raspberry Pi no tiene conexión a Internet.

Si ve que al actualizar obtiene un error como `E: Problem with MergeList /var/lib/dpkg/status`, pruebe a ejecutar estos comandos:

```
sudo rm /var/lib/dpkg/status
sudo touch /var/lib/dpkg/status
```

El proceso de instalación puede tardar un poco porque los archivos se tienen que descargar e instalar. Algunas instalaciones también agregarán accesos directos en su escritorio o grupos de programas en su menú de inicio.

Puede buscar *software* para instalar con el comando `apt-get search` seguido por una cadena de búsqueda como `abiword`. Esto mostrará una lista de paquetes coincidentes que podría instalar.

Para saber más

Consulte el Capítulo 3.18 para eliminar programas que no necesita y así liberar espacio.

Consulte el Capítulo 3.21 para descargar el código fuente de GitHub.

3.18 Eliminar un *software* instalado con apt-get

Problema

Después de haber instalado una gran cantidad de programas con `apt-get`, desea eliminar algunos de ellos.

Solución

`apt-get` tiene una opción (`remove`) que eliminará paquetes, pero solo quitará los que se hayan instalado con `apt-get install`. Por ejemplo, si desea eliminar AbiWord, debería utilizar el comando:

```
$ sudo apt-get remove abiword
```

Observaciones

Eliminar un paquete como este no siempre lo elimina todo, ya que los paquetes a menudo tienen paquetes previos que también están instalados. Para eliminar estos puede utilizar la opción `autoremove`, como se muestra aquí:

```
$ sudo apt-get autoremove abiword
$ sudo apt-get clean
```

La opción `apt-get clean` limpiará un poco más los archivos de instalación de paquetes no utilizados.

Para saber más

Consulte el Capítulo 3.17 para instalar paquetes mediante `apt-get`.

3.19 Instalar paquetes de Python con pip

Problema

Desea utilizar el gestor de paquetes pip para instalar las bibliotecas de Python.

Solución

Si tiene la última versión de Raspbian, pip ya estará instalado y lo podrá ejecutar desde la línea de comandos como se muestra en el siguiente ejemplo, tomado del Capítulo 8.2, donde se utiliza para instalar la biblioteca `svgwrite` de Python.

```
$ sudo pip install svgwrite
```

Si pip no está instalado en su sistema, puede instalarlo usando el comando:

```
$ sudo apt-get install python-pip
```

Observaciones

Aunque muchas bibliotecas de Python pueden instalarse usando `apt-get` (véase el Capítulo 3.17) algunas no pueden y deben usar pip.

Para saber más

Para instalar un *software* usando `apt-get`, véase el Capítulo 3.17.

3.20 Buscar archivos desde la línea de comandos

Problema

Desea descargar un archivo desde Internet sin necesidad de utilizar un navegador web.

Solución

Utilice el comando `wget` para obtener un archivo de Internet. Por ejemplo:

```
$ wget http://www.icrobotics.co.uk/wiki/images/c/c3/Pifm.tar.gz
--2013-06-07 07:35:01-- http://www.icrobotics.co.uk/wiki/images/c/c3/Pifm.tar.gz
Resolving www.icrobotics.co.uk (www.icrobotics.co.uk)... 155.198.3.147
Connecting to www.icrobotics.co.uk (www.icrobotics.co.uk)|155.198.3.147|
:80... connected.
HTTP request sent, awaiting response... 200 OK
Length:  5521400  (5.3M)  [application/x-gzip]
Saving to: `Pifm.tar.gz'

100%[================================================>] 5,521,400    601K/s

2013-06-07 07:35:11 (601 KB/s) - `Pifm.tar.gz' saved [5521400/5521400]
```

Si su URL contiene caracteres especiales, es buena idea escribirlo entre comillas dobles. Este ejemplo de URL es del Capítulo 4.11.

Observaciones

Encontrará instrucciones para instalar *software* que dependen del uso de `wget`. A menudo es más conveniente hacer esto desde la línea de comandos en lugar de usar un navegador, encontrar el archivo, descargarlo y luego copiarlo en el lugar que lo necesite.

El comando `wget` toma la URL para descargarla como argumento y la descarga en el directorio actual. Normalmente se utiliza para descargar un archivo de algún tipo, pero también descargará cualquier página web.

Para saber más

Para obtener más información sobre la instalación con `apt-get`, consulte el Capítulo 3.16. Para obtener más información sobre cómo seleccionar y utilizar un navegador, consulte el Capítulo 4.4.

3.21 Buscar códigos fuente con Git

Problema

A veces, las bibliotecas Python y otros *softwares* se suministran a través de una URL del repositorio Git. Tiene que ser capaz de buscarlas en Raspberry Pi.

Solución

Para usar códigos en los repositorios de Git necesita usar el comando `git clone` para descargar los archivos.

Observaciones

Por ejemplo, el siguiente comando descargará todos los ejemplos de código fuente de este libro:

```
$ git clone https://github.com/simonmonk/raspberrypi_cookbook_ed2.git
```

Para saber más

Aprende más sobre Git y el servicio de hosting GitHub.

Consulte también el Capítulo 3.17.

3.22 Ejecutar un programa o una secuencia de comandos al iniciar

Problema

Desea que un programa o una secuencia de comandos se inicie automáticamente cuando su Raspberry Pi arranque.

Solución

Modifique su archivo *rc.local* para ejecutar el programa que desea.

Edite el archivo */etc/rc.local* utilizando el comando:

```
$ sudo nano /etc/rc.local
```

Agregue la siguiente línea después del primer bloque de líneas de comentario que comienza con #:

```
$ /usr/bin/python /home/pi/my_program.py &
```

Es importante incluir `&` al final de la línea de comandos para que se ejecute en segundo plano; de lo contrario su Raspberry Pi no arrancará.

Observaciones

Esta manera de autoejecutar un programa necesitará una edición muy cuidadosa de *rc.local*, o tendrá que parar su Raspberry Pi en el arranque.

Para saber más

Una forma más segura de autoejecutar un programa se detalla en el Capítulo 3.23.

3.23 Ejecutar un programa o secuencia de comandos como servicio

Problema

Desea que una secuencia de comandos o programa concreto se inicie automáticamente cada vez que se reinicie Raspberry Pi.

Solución

Debian Linux, en el que se basan la mayoría de las distribuciones de Raspberry Pi, utiliza un mecanismo basado en la dependencia para automatizar el funcionamiento de los comandos al inicio. Esto es un poco complicado de usar e implica crear un archivo de configuración para el script o programa que desea ejecutar en la carpeta llamada *init.d*.

Observaciones

El siguiente ejemplo muestra cómo ejecutar un script de Python en su directorio personal. El script podría hacer cualquier cosa, pero en este caso, ejecuta un servidor web de Python simple, que se describe más adelante en el Capítulo 7.18.

Los pasos involucrados en esto son:

1. Cree un script *init*.
2. Haga que *init* sea ejecutable.
3. Informe al sistema acerca del nuevo script *init*.

Primero, cree el script *init*. Debe crearlo en la carpeta */etc/init.d/*. Se puede llamar como quiera, pero, en este ejemplo, lo llamará `my_server`.

Cree el archivo nuevo utilizando `nano` con el siguiente comando:

```
$ sudo nano /etc/init.d/my_server
```

Pegue el siguiente código en la ventana del editor y guarde el archivo:

```
### EMPIEZA LA INFO INIT
# Provides: my_server
# Required-Start: $remote_fs $syslog $network
# Required-Stop: $remote_fs $syslog $network
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Simple Web Server
# Description: Simple Web Server
###ACABA LA INFO INIT

#! /bin/sh
# /etc/init.d/my_server

export HOME
case "$1" in
  start)
    echo "Starting My Server"
    sudo /usr/bin/python /home/pi/myserver.py  2>&1 &
  ;;
stop)
  echo "Stopping My Server"
  PID=`ps auxwww | grep myserver.py | head -1 | awk '{print $2}'`
  kill -9 $PID
  ;;
*)
  echo  "Usage:  /etc/init.d/my_server  {start|stop}"
  exit 1
;;
esac
exit 0
```

Es bastante trabajo para automatizar la ejecución de un script, pero casi todo es código repetitivo. Para ejecutar uno diferente, modifique la ruta a través del script, cambiando las descripciones y el nombre del archivo Python que desea ejecutar.

El siguiente paso es hacer que este archivo sea ejecutable por el propietario, cosa que se hace con este comando:

```
$ sudo chmod +x /etc/init.d/my_server
```

Ahora que el programa está configurado como servicio, puede probar que todo está bien antes de configurarlo de inicio automático en el arranque usando el siguiente comando:

```
$ /etc/init.d/my_server start
Starting My Server
Bottle v0.11.4 server starting up (using WSGIRefServer())...
Listening on http://192.168.1.16:80/
Hit Ctrl-C to quit.
```

Por último, si se ejecuta correctamente, utilice el siguiente comando para que el sistema sea consciente del nuevo servicio que ha definido:

```
$ sudo update-rc.d my_server defaults
```

Para saber más

Para obtener un enfoque más sencillo sobre la ejecución automática de un programa, consulte el Capítulo 3.22. Para más información sobre cómo cambiar permisos de archivos y carpetas, consulte el Capítulo 3.13.

3.24 Ejecutar automáticamente un programa o una secuencia de comandos en intervalos regulares

Problema

Desea ejecutar una secuencia de comandos una vez al día o en intervalos regulares.

Solución

Utilice el comando `crontab` de Linux.

Para ello, Raspberry Pi necesita saber la hora y la fecha y, por lo tanto, necesita una conexión a la red o a un reloj en tiempo real. Consulte el Capítulo 12.14.

Observaciones

El comando `crontab` le permite programar eventos en intervalos regulares. Esto puede ser diario o por horas, e incluso puede definir patrones complicados para que tenga lugar en diferentes días de la semana. Esto puede ser útil para las tareas de copia de seguridad, que es posible que desee que se ejecuten en mitad de la noche.

Puede editar los eventos programados mediante el siguiente comando:

```
$ crontab -e
```

Si el script o el programa que desea ejecutar necesita ser ejecutado por un superusuario, prefije todos los comandos `crontab` con `sudo` (Capítulo 3.12).

La línea de comentarios indica el formato de una línea `crontab`. Los dígitos son, en orden, minuto, hora, día del mes, mes, día de la semana y el comando que desea ejecutar.

La línea que comienza por un `#` es solo una línea de comentarios para recordarle el formato de una línea `crontab`.

Si hay un `*` en la posición del dígito, significa *every* (cada); si hay un número en su lugar, la secuencia de comandos solo se ejecutará en ese minuto/hora/día del mes.

Por lo tanto, para ejecutarlo cada día a la 1 de la madrugada: Figura 3.7.

Figura 3.7. *Editar crontab.*

Utilizando solamente un solo dígito puede especificar los intervalos para ejecutar el script solo los días laborables, por ejemplo:

```
0 1 * * 1-5 /home/pi/myscript.sh
```

Si su script necesita ejecutarse desde un directorio en particular, puede usar un punto y coma (;) para separar varios comandos como se muestra aquí:

```
0 1 * * * cd /home/pi; python mypythoncode.py
```

Para saber más

Puede ver la documentación completa de `crontab` introduciendo este comando:

```
$ man crontab
```

3.25 Encontrar cosas

Problema

Desea encontrar un archivo que sabe que está en alguna parte del sistema.

Solución

Utilice el comando de Linux `find`.

Observaciones

Comenzando con un directorio especificado en el comando, el comando `find` buscará y, si encuentra el archivo, mostrará su ubicación. Por ejemplo:

```
$ find /home/pi -name gemgem.py
/home/pi/python_games/gemgem.py
```

Puede iniciar la búsqueda más arriba del árbol, incluso en la raíz de todo el sistema de archivos (/). Esto hará que la búsqueda tarde mucho más, y también producirá mensajes de error. Puede redirigir estos mensajes de error añadiendo `2>/dev/null` al final de la línea.

Para buscar el archivo en todo el sistema de archivos utilice el siguiente comando:

```
$ find / -name gemgem.py 2>/dev/null
/home/pi/python_games/gemgem.py
```

Puede también usar comodines con `find`, como se muestra aquí:

```
$ find /home/pi -name match*
/home/pi/python_games/match4.wav
/home/pi/python_games/match2.wav
/home/pi/python_games/match1.wav
/home/pi/python_games/match3.wav
/home/pi/python_games/match0.wav
/home/pi/python_games/match5.wav
```

Para saber más

El comando `find` tiene otras funciones avanzadas para la búsqueda. Para ver la documentación completa de `find` use este comando:

```
$ man find
```

3.26 Utilizar el historial de la línea de comandos

Problema

Quiere ser capaz de repetir comandos en la línea de comandos sin tener que escribirlos de nuevo.

Solución

Utilice las teclas (arriba y abajo) para seleccionar comandos anteriores desde el historial de comandos y el comando `history` con `grep` para encontrar comandos antiguos.

Observaciones

Puede acceder al comando anterior que ejecutó pulsando la tecla de la flecha hacia arriba. Al pulsarla de nuevo le llevará al comando anterior, y así sucesivamente. Si se pasa el comando que desea, la flecha hacia abajo lo llevará de nuevo en la otra dirección.

Si desea cancelar sin ejecutar el comando, utilice Ctrl-C.

Con el tiempo, su historial de comandos crecerá demasiado como para encontrar un comando que usó hará siglos. Para encontrarlo hacia atrás, use el comando `history`.

```
$ history
    1  sudo nano /etc/init.d/my_server
    2  sudo chmod +x /etc/init.d/my_server
    3  /etc/init.d/my_server start
    4  cp /media/4954-5EF7/sales_log/server.py myserver.py
    5  /etc/init.d/my_server start
    6  sudo apt-get update
    7  sudo apt-get install bottle
    8  sudo apt-get install python-bottle
```

Esto muestra todo el historial de comandos y es probable que tenga demasiadas entradas para encontrar la que desea. Para remediarlo, puede *canalizar* el comando `history` con el comando `grep`, el cual solo mostrará resultados que coincidan con una cadena de búsqueda. Así, por ejemplo, para encontrar todos los comandos `apt-get` (Capítulo 3.17) que ha emitido, puede usar la línea:

```
$ history | grep apt-get
    6  sudo apt-get update
    7  sudo apt-get install bottle
    8  sudo apt-get install python-bottle
   55  history | grep apt-get
```

Cada elemento del historial tiene un número junto a él. Así que si encuentra la línea que buscaba, puede ejecutarla usando `!` seguido del número del historial como se muestra aquí:

```
$ !6
sudo apt-get update
Hit http://mirrordirector.raspbian.org wheezy InRelease
Hit http://mirrordirector.raspbian.org wheezy/main armhf Packages Hit
http://mirrordirector.raspbian.org wheezy/contrib armhf Packages
.....
```

Para saber más

Para encontrar archivos en lugar de comandos consulte el Capítulo 3.25.

3.27 Supervisar la actividad del procesador

Problema

Raspberry Pi a veces puede funcionar un poco lento, por lo que desea ver qué está acaparando el procesador.

Solución

Utilice el Administrador de tareas, lo encontrará dentro del menú de inicio en el grupo de programas Herramientas del sistema (Figura 3.8).

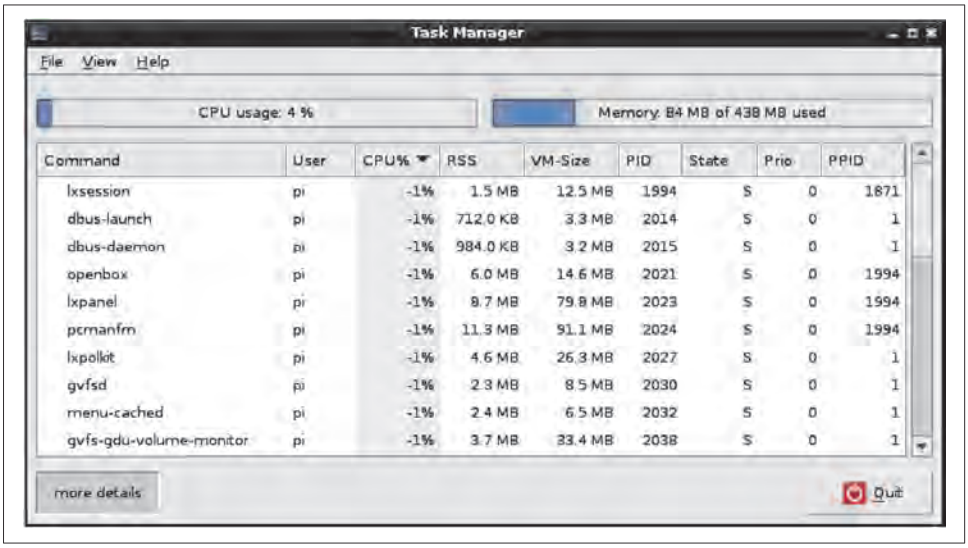


Figura 3.8. *Administrador de tareas.*

El Administrador de tareas le permite ver cuánta CPU y memoria se está utilizando. También puede hacer clic con el botón derecho del ratón en un proceso y seleccionar la opción para eliminarlo en el menú emergente.

Los gráficos de la parte superior de la ventana muestran el uso total de la CPU y la memoria. Los procesos se enumeran debajo, y puede ver la CPU que cada uno está utilizando.

Observaciones

Si prefiere hacer este tipo de cosas desde la línea de comandos, utilice el comando `top` de Linux para mostrar datos similares sobre el procesador y la memoria y los procesos que utilizan más recursos (Figura 3.9). Puede utilizar el comando `kill` para detener un proceso. Tendrá que hacerlo como superusuario.

En este caso, puede ver que es un programa de Python que utiliza el 97 % de la CPU. La primera columna muestra el ID del proceso (2447). Para detener este proceso, introduzca el comando:

```
$ kill 2447
```

Es posible eliminar procesos vitales del sistema operativo de esta manera. Si lo hace, apagar y volver a encender su Pi restablecerá las cosas a la normalidad.

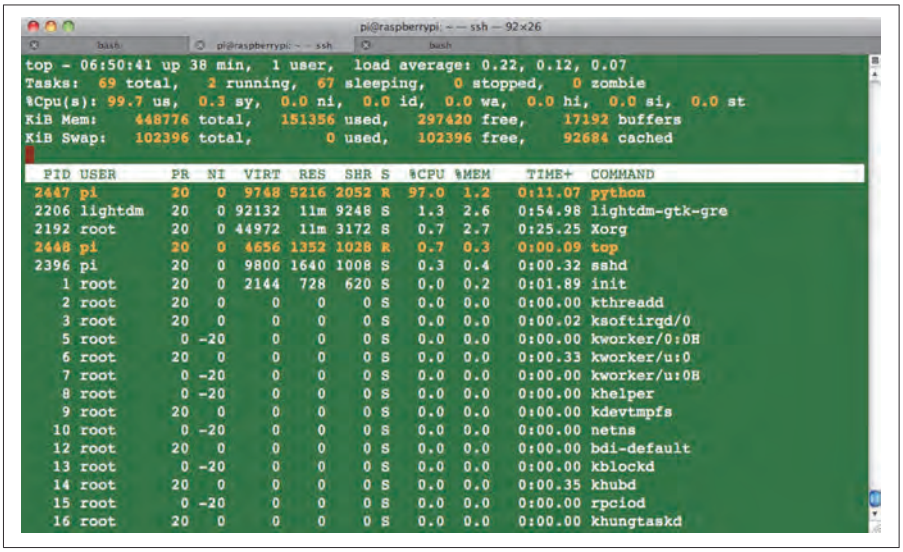


Figura 3.9. *Utilizar el comando top para ver el uso de los recursos.*

A veces, es posible que tenga un proceso en ejecución que no sea visible inmediatamente cuando utilice `top`. Si este es el caso, puede buscar todos los procesos que se ejecutan usando el comando `ps` y canalizando (Capítulo 3.32) los resultados con el comando `grep`, que buscará los resultados y resaltará los elementos de interés.

Por ejemplo, para encontrar el ID del proceso de Python que acapara la CPU, ejecutaríamos el siguiente comando:

```
$ ps -ef | grep "python"
pi        2447  2397 99 07:01 pts/0    00:00:02 python speed.py
pi        2456  2397  0 07:01 pts/0    00:00:00 grep --color=auto python
```

En este caso, el ID del proceso del programa `speed.py` de Python es 2447. La segunda entrada en la lista es el propio proceso del `ps`.

Una variación en el comando `kill` es el comando `killall`. Utilícelo con precaución, ya que elimina todos los procesos que coinciden con su argumento. Así, por ejemplo, el siguiente comando eliminará todos los programas de Python que se estén ejecutando en Raspberry Pi:

```
$ sudo killall python
```

Para saber más

Puede consultar las páginas de la documentación para `top`, `ps`, `grep`, `kill` y `killall`. Puede verlas escribiendo `man` seguido del nombre del comando, como se muestra aquí:

```
$ man top
```

3.28 Trabajar con archivos comprimidos

Problema

Ha descargado un archivo comprimido y desea descomprimirlo.

Solución

Dependiendo del tipo de archivo, necesitará utilizar los comandos `tar` o `gunzip`.

Observaciones

Si el archivo que desea descomprimir solo tiene la extensión *.gz*, puede descomprimirlo usando el comando:

```
$ gunzip myfile.gz
```

También puede encontrar archivos (llamados *tarballs*) que contienen un directorio que ha sido archivado con la utilidad de Linux `tar` y luego son comprimidos con `gzip` en un archivo con un nombre como *myfile.tar.gz*.

Puede extraer los archivos y las carpetas originales de un tarball utilizando el comando `tar`:

```
$ tar -xzf myfile.tar.gz
```

Para saber más

Puede obtener más información sobre `tar` desde las páginas de la documentación, a las que puede acceder mediante el comando `man tar`.

3.29 Mostrar los dispositivos USB conectados

Problema

Ha conectado un dispositivo USB y quiere asegurarse de que Linux lo reconoce.

Solución

Utilice el comando `lsusb`. Esto mostrará todos los dispositivos conectados a los puertos USB de su Raspberry Pi:

```
$ lsusb
Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.
Bus 001 Device 004: ID 15d9:0a41 Trust International B.V. MI-2540D [Optical mouse]
```

Observaciones

Este comando le indicará si un dispositivo está conectado o no, pero no garantiza que esté funcionando correctamente. Puede haber *drivers* para instalar y cambios de configuración para hacer en el *hardware*.

Para saber más

Para obtener un ejemplo de uso de `lsusb` al conectar una webcam externa consulte el Capítulo 4.6.

3.30 Redirigir la salida desde la línea de comandos a un archivo

Problema

Desea crear rápidamente un archivo con algo de texto o grabar una lista de directorios en un archivo.

Solución

Utilice el comando `>` para redirigir la salida que aparecería en la línea de comandos.

Por ejemplo, para copiar una lista de directorios en un archivo llamado *myfiles.txt*, haga lo siguiente:

```
$ ls > myfiles.txt
$  more  myfiles.txt
Desktop
indiecity
master.zip
mcpi
```

Observaciones

Puede usar el comando `>` en cualquier comando de Linux que produzca una salida, incluso si está ejecutando, por ejemplo, un programa de Python.

También puede usar el comando opuesto (`<`) para redirigir la entrada del usuario, aunque esto no es tan útil como `>`.

Para saber más

Para utilizar `cat` para unir varios archivos consulte el Capítulo 3.31.

3.31 Concatenar archivos

Problema

Usted tiene un número de archivos de texto y desea unirlos en un archivo grande.

Solución

Utilice el comando `cat` para *concatenar* un numero de archivos en un archivo de salida (*output*). Por ejemplo:

```
$ cat file1.txt file2.txt file3.txt > full_file.txt
```

Observaciones

Unir archivos es el verdadero propósito del comando `cat`. Puede proporcionar tantos nombres de archivos como desee, y todos se escribirán en el archivo al que los dirija. Si no redirecciona la salida, aparecerá en la ventana del terminal. Si son archivos grandes, puede tardar.

Para saber más

Consulte también el Capítulo 3.8, donde se utiliza `cat` para mostrar el contenido de un archivo.

3.32 Usar *pipes* (tuberías)

Problema

Desea utilizar la salida de un comando de Linux como entrada de otro comando.

Solución

Utilizar el comando `pipe`, que es el símbolo de *barra* (|) en el teclado, para `canalizar` la salida de un comando a otro. Por ejemplo:

```
$ ls -l *.py | grep Jun
-rw-r--r-- 1 pi 226 Jun  7 06:49 speed.py
```

Aquí encontrará todos los archivos con la extensión *py* que también tengan *Jun* en su lista de directorios, indicando que fueron modificados por última vez en junio.

Observaciones

A primera vista, esto se parece bastante a la redirección de salida usando > (Capítulo 3.30). La diferencia es que > no funcionará si el objetivo es otro programa. Solo funcionará para redirigir a un archivo.

Puede encadenar tantos programas como quiera, como se muestra aquí, aunque no es algo que hará a menudo.

```
$ command1 | command2 | command3
```

Para saber más

Puede consultar el Capítulo 3.27 para un ejemplo de uso de `grep` para encontrar un proceso.

Véase el Capítulo 3.26 para buscar en su historial de comandos, usando un `pipe` y `grep`.

3.33 Ocultar la salida al terminal

Problema

Desea ejecutar un comando, pero no quiere que la salida aparezca en pantalla.

Solución

Redireccione la salida a */dev/null* utilizando >. Por ejemplo:

```
$ ls > /dev/null
```

Observaciones

Este ejemplo ilustra la sintaxis, pero es bastante inútil. Un uso más común se da cuando está ejecutando un programa y el desarrollador deja muchos mensajes de rastreo en su código que realmente no quiere ver. El siguiente ejemplo oculta la salida innecesaria del comando `find` (véase el Capítulo 3.35).

```
$ find / -name gemgem.py 2>/dev/null
/home/pi/python_games/gemgem.py
```

Para saber más

Para más información acerca de cómo redirigir la salida estándar, consulte el Capítulo 3.30.

3.34 Ejecutar programas en segundo plano

Problema

Desea ejecutar un programa, pero también trabaja en otras tareas.

Solución

Ejecute el programa o el comando en segundo plano utilizando el comando `&`.

Por ejemplo:

```
$ python speed.py &
[1] 2528
$ ls
```

En lugar de esperar hasta que el programa haya terminado de ejecutarse, la línea de comandos muestra el ID del proceso (el segundo número) e inmediatamente le permite continuar con cualquier otro comando que quiera ejecutar. Puede utilizar el ID del proceso para eliminar procesos en segundo plano (Capítulo 3.27).

Para llevar un proceso del segundo plano al primer plano utilice el comando `fg`:

```
$ fg
python speed.py
```

Esto informa del comando o programa que se está ejecutando y luego espera a que finalice.

Observaciones

La salida del proceso en segundo plano seguirá apareciendo en el terminal.

Una alternativa para poner procesos en segundo plano es simplemente abrir más de una ventana de terminal.

Para saber más

Para obtener información sobre cómo administrar procesos consulte el Capítulo 3.27.

3.35 Crear alias de comandos

Problema

Desea crear alias para los comandos que utiliza con frecuencia.

Solución

Edite el archivo *~/.bashrc* utilizando `nano` (Capítulo 3.7) y luego muévase al final del archivo y añada tantas líneas como desee como estas:

```
alias l='ls -a'
```

Esto crea un alias llamado *l* que, cuando lo introduzca, será interpretado como el comando `ls -a`.

Guarde y salga del archivo usando Ctrl-X y Ctrl-Y, y luego, para actualizar el terminal con el nuevo alias, escriba el siguiente comando:

```
$ source .bashrc
```

Observaciones

Muchos usuarios de Linux configuran un alias para `rm` como el siguiente para que confirme las eliminaciones.

```
$ alias rm='rm -i'
```

No es una mala idea, siempre y cuando no se olvide cuando utilice el sistema de otra persona que no tenga este alias configurado.

Para saber más

Para más información sobre `rm`, consulte el Capítulo 3.11.

3.36 Ajustar la fecha y la hora

Problema

Desea configurar manualmente la fecha y la hora de su Raspberry Pi, porque no tiene conexión a Internet.

Solución

Utilice el comando de Linux `date`.

El formato de fecha y hora es MMDDHHMMYYYY, donde MM es el número del mes, DD es el día del mes, HH y MM son las horas y los minutos, respectivamente, y YYYY es el año.

Por ejemplo:

```
$ sudo date 010203042013
Wed Jan  2 03:04:00 UTC 2013
```

Observaciones

Si Raspberry Pi está conectada a Internet, a medida que arranque, automáticamente establecerá la hora utilizando un servidor de tiempo de Internet.

También puede utilizar `date` para mostrar la hora UTC simplemente introduciendo `date` (fecha):

```
$ date
Wed Jan  2 03:08:14 UTC 2013
```

Para saber más

Si desea que su Raspberry Pi mantenga la hora correcta incluso cuando no hay red, puede utilizar un módulo de reloj a tiempo real (RTC) (Capítulo 12.14).

3.37 Saber cuánto espacio tiene en la tarjeta SD

Problema

Desea saber cuánto espacio hay libre en la tarjeta SD.

Solución

Utilice el comando de Linux `df`:

```
$ df -h
Filesystem     Size   Used  Avail Use% Mounted on
rootfs         3.6G  1.7G  1.9G   48% /
/dev/root      3.6G  1.7G  1.9G   48% /
devtmpfs       180M  0     180M    0% /dev
tmpfs          38M   236K   38M    1% /run
tmpfs          5.0M  0     5.0M    0% /run/lock
tmpfs          75M   0      75M    0% /run/shm
/dev/mmcblk0p1 56M   19M    38M   34% /boot
```

Observaciones

En cuanto a la primera línea de los resultados, puede ver que hay 3,6 GB de almacenamiento en la tarjeta SD, de los cuales se utilizan 1,7 GB.

Cuando se quede sin espacio en el disco es probable que obtenga un comportamiento inesperado, como mensajes de error que indiquen que no se pudo escribir un archivo.

Para saber más

Puede encontrar la página de la documentación para `df` utilizando `man df`.

CAPÍTULO 4

Software

4.1 Introducción

Este capítulo contiene una serie de fórmulas para utilizar *softwares* preconfigurados en Raspberry Pi.

Algunas fórmulas de este capítulo están relacionadas con la conversión de Raspberry Pi a un solo uso, mientras que otras utilizan piezas específicas de *softwares* en Raspberry Pi.

4.2 Crear un centro multimedia

Problema

Desea convertir su Raspberry Pi en un centro multimedia.

Solución

Para utilizar su Raspberry Pi como centro multimedia debe tener una Raspberry Pi 3 o el modelo B de la Raspberry Pi 2, ya que la reproducción de vídeo es muy intensa en los procesadores.

Puede configurar su Raspberry Pi como centro multimedia durante el proceso de instalación de NOOBS (Capítulo 1.6). En lugar de seleccionar Raspbian como la distribución a instalar, seleccione OpenELEC_Pi1 si tiene la Raspberry Pi 1 o OpenELEC_Pi2 si tiene la Raspberry Pi 2.

OpenELEC es una distribución que optimiza su Raspberry Pi como centro multimedia. Incluye el *software* del centro multimedia Kodi, basado en el proyecto de código abierto XBMC que fue desarrollado originalmente para convertir las consolas Xbox en centros multimedia. Desde entonces, el código ha sido llevado a muchas plataformas, incluyendo Raspberry Pi (Figura 4.1).

Figura 4.1. *Raspberry Pi como centro multimedia.*

Raspberry Pi es perfectamente capaz de reproducir vídeos en *full* HD, así como música en *streaming*, archivos MP3 y radio por Internet.

Observaciones

Kodi es una potente pieza de *software* con muchas características. Quizás la manera más simple de comprobar que está funcionando es poner algunos archivos de música y/o vídeo en una unidad flash USB o en un disco duro externo y conectarlo a Raspberry Pi. Debería poder reproducirlos desde Kodi.

Puesto que Raspberry Pi es probable que esté cerca de su TV, puede ver si su TV tiene un puerto USB que pueda proporcionar corriente suficiente para ejecutar Raspberry Pi. Si este es el caso, no necesitará una fuente de alimentación aparte.

Es una buena idea tener un teclado y un ratón inalámbricos, si los compra como un conjunto usarán un solo puerto USB para el dongle, ya que evitará tener los cables por todas partes. También puede comprar miniteclados con *trackpads* incorporados que son útiles en esta situación.

Una conexión de red por cable ofrece generalmente un rendimiento superior y mejor que una conexión WiFi, pero no siempre es posible tener Pi cerca de una toma Ethernet. Si este es el caso, puede configurar XBMC para utilizar un dongle WiFi para una conexión de red.

La configuración de Kodi es muy intuitiva. Puede encontrar las instrucciones completas sobre el uso del *software* en *http://kodi.wiki/*.

Para saber más

Puede agregar un control remoto IR a Raspberry Pi para controlar XBMC.

4.3 Instalar programas de ofimática

Problema

Necesita abrir procesadores de texto y documentos de hojas de cálculo en Raspberry Pi.

Solución

Raspberry Pi es, después de todo, un ordenador Linux, por lo que hay varias aplicaciones de ofimática que puede instalar para utilizar hojas de cálculo y documentos de procesador de textos.

Los programas para Raspberry Pi se descargan de Internet, por lo que necesitará tener una conexión.

Antes de instalar cualquier *software* nuevo, una buena idea es abrir un terminal y ejecutar el siguiente comando:

```
$ sudo apt-get update
```

Para instalar el procesador de texto AbiWord, ejecute este comando:

```
$ sudo apt-get install abiword
```

Se le pedirá que confirme la instalación escribiendo **Y**; después de un minuto aproximadamente la instalación estará completa. Si observa su menú de Inicio, verá una nueva sección llamada Office, donde encontrará AbiWord (Figura 4-2).

AbiWord abrirá *.doc*, *.docx*, y otros formatos comunes de documentos de procesadores de textos.

Si necesita usar una hoja de cálculo, Gnumeric es una buena opción. Instálelo utilizando el comando:

```
$ sudo apt-get install gnumeric
```

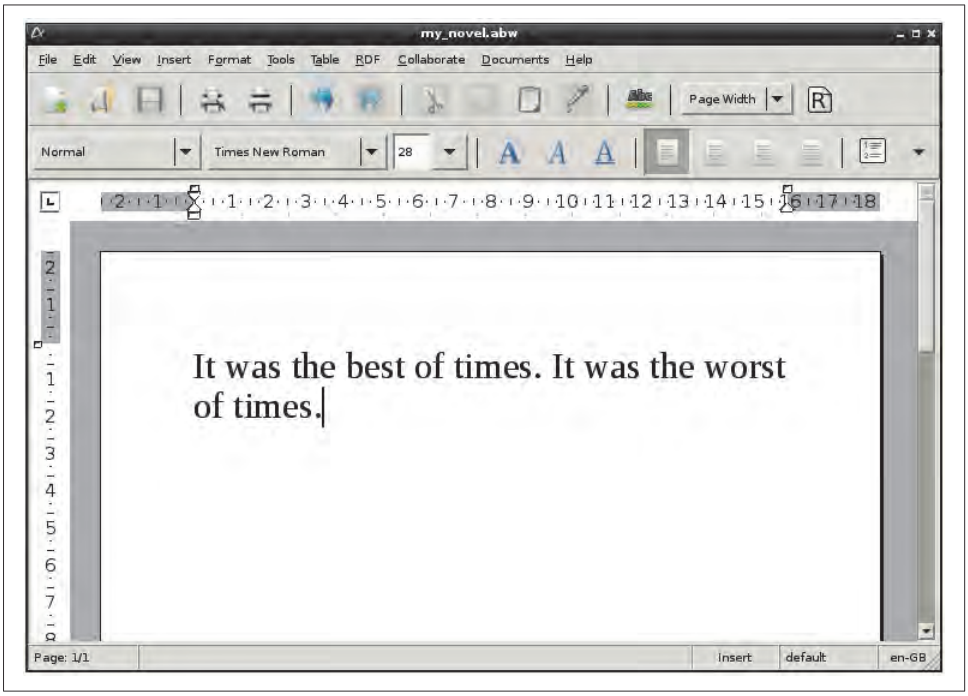


Figura 4.2. *AbiWord en Raspberry Pi.*

Observaciones

Si ve que las aplicaciones de ofimática funcionan lentamente, intente hacer *overclock* en su Raspberry Pi (Capítulo 1.11) para acelerarlo. Las Raspberry Pi 3 y 2 ejecutarán las aplicaciones de ofimática mucho mejor que una Raspberry Pi más vieja.

Para saber más

Se está intentando migrar otro *software* de ofimática a Raspberry Pi, como LibreOffice (parecido a Open Office). Consulte en Internet las últimas noticias sobre *software* de ofimática.

Consulte el Capítulo 3.17 para obtener información sobre el uso de `apt-get`.

4.4 Instalar otros navegadores

Problema

Desea usar un navegador que no sea Midori.

Solución

Puede utilizar varios navegadores en su Raspberry Pi. Pi no es un ordenador potente, y los navegadores modernos y páginas web pueden suponer una gran carga. Esto significa que cuando se utilice un navegador en Raspberry Pi, siempre debe haber un equilibrio entre sus características y el rendimiento.

Chromium (Figura 4-3), como su nombre indica, será familiar para los usuarios de Google Chrome. Sus funciones son completas, pero es notablemente lento cuando intenta desplazarse hacia arriba y abajo en una página web. Puede instalar Chromium mediante el siguiente comando, que pondrá un nuevo enlace en la sección Internet de su menú de inicio.

```
$ sudo apt-get install chromium-browser
```

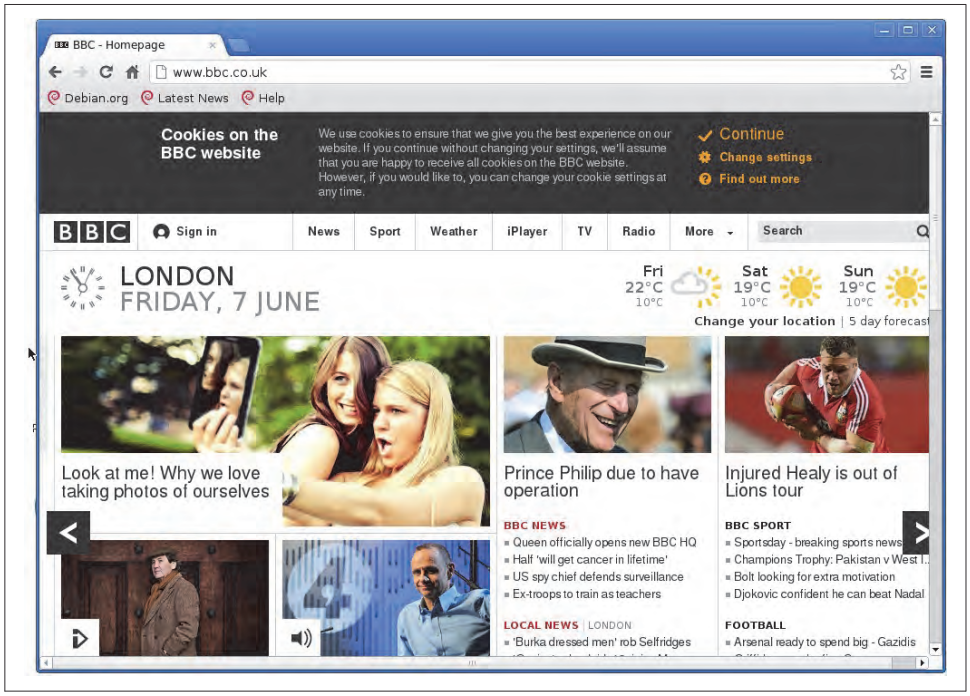


Figura 4.3. *El navegador Chromium.*

Otra alternativa a Midori es Iceweasel (Figura 4.4). Este navegador se basa en Firefox y funciona más rápido que Chromium porque utiliza la versión móvil de los sitios web (cuando está disponible), que normalmente tiene un código HTML más simple. Puede descargar e instalar Iceweasel utilizando este comando:

```
$ sudo apt-get install iceweasel
```

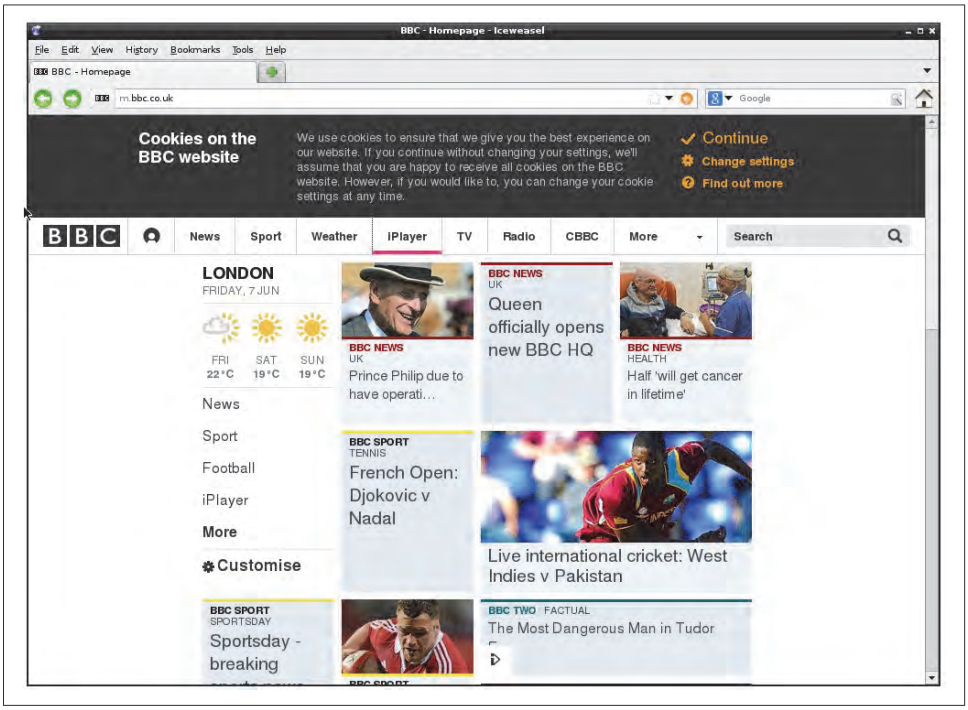


Figura 4.4. *El navegador Iceweasel.*

Observaciones

Dado que los navegadores necesitan una elevada potencia y memoria para muchos sitios web, necesitará una Raspberry Pi 3 o 2 para evitar una experiencia muy lenta.

Para saber más

Para obtener más información sobre la instalación con `apt-get` consulte el Capítulo 3.17.

4.5 Usar Pi Store

Problema

Desea instalar programas y juegos utilizando Pi Store.

Solución

Pi Store (Figura 4.5) equivale a Raspberry Pi lo que App Store a Apple o Play Store a Google; es un lugar desde el que puedes descargar, instalar y ejecutar todo tipo de aplicaciones, tanto gratuitas como de pago.

Encontrará un acceso directo en el menú de inicio para la aplicación Pi Store en la sección Internet.

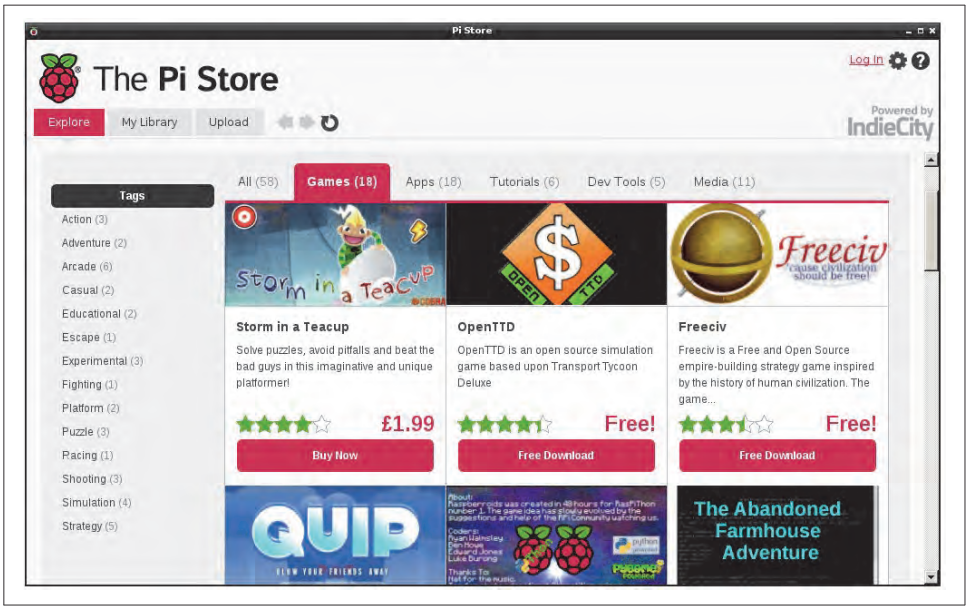


Figura 4.5. *Pi Store.*

La primera vez que intente descargar una aplicación, se le pedirá que se registre. Después de registrarse, la aplicación se descargará y aparecerá en la pestaña My Library. Para ejecutar la aplicación haga doble clic en ella.

Observaciones

Esta es una buena manera de buscar programas interesantes para usar en su Raspberry Pi. Puede encontrar más aplicaciones en Pi Store.

Para saber más

Consulte la web oficial de Pi Store.

4.6 Crear un servidor webcam

Problema

Desea configurar Raspberry Pi como servidor webcam.

Solución

Descargue el *software motion*. Esto le permitirá configurar una Raspberry Pi con una webcam USB conectada a él para que pueda conectarse a una página web y ver la webcam.

Para instalar el *software*, escriba el siguiente comando en una ventana terminal:

```
$ sudo apt-get install motion
```

Conecte su webcam USB y escriba `lsusb` para ver si la webcam está conectada:

```
$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp.
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.
Bus 001 Device 004: ID 3538:0059 Power Quotient International Co., Ltd
Bus 001 Device 006: ID eb1a:299f eMPIA Technology, Inc.
```

Si no puede ver ninguna entrada que parezca la de la webcam, intente desconectarla, ejecute el comando de nuevo y vea si una de las entradas desaparece de la lista. En este caso, la entrada final de la lista es para la webcam.

Después, necesita hacer algunos cambios de configuración. En primer lugar, edite el archivo */etc/motion/motion.conf* utilizando el comando:

```
$ sudo nano /etc/motion/motion.conf
```

Ese es un gran archivo de configuración. Justo cerca de la parte superior del archivo, encontrará la línea de `daemon off`; cámbielo a `daemon on`.

El otro cambio está mucho más abajo del archivo. Necesita cambiar `webcam_localhost = on` a `webcam_localhost = off`.

Necesitará cambiar otro archivo. Introduzca el comando:

```
$ sudo nano /etc/default/motion
```

Cambie `start_motion_daemon=no` a `start_motion_daemon=yes`.

Para poner el servicio web en ejecución, ejecute el comando:

```
$ sudo service motion start
```

Ahora debe poder abrir su navegador web y ver la webcam. Para ello, necesitará conocer la dirección IP de su Raspberry Pi (Capítulo 2.3).

Desde otro equipo de la misma red, abra un navegador y vaya a la URL http://192.168.1.16:8081/. Tendrá que cambiarla para que coincida con la dirección IP de su Raspberry Pi, pero mantenga el número de puerto `:8081` al final de la URL.

Si todo va bien, debería ver algo como la Figura 4.6.

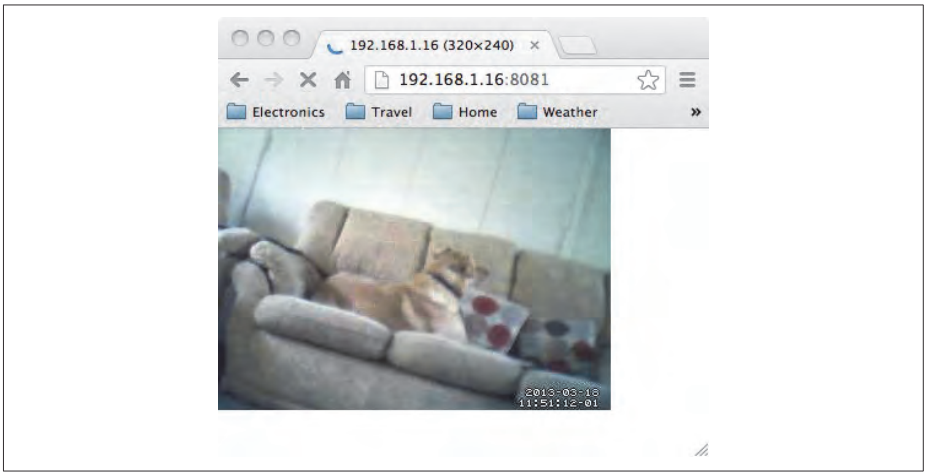


Figura 4.6. *Una webcam en Raspberry Pi.*

Observaciones

El *software motion* es realmente muy potente; hay muchos otros ajustes que puede modificar para cambiar el funcionamiento de su webcam.

De forma predeterminada, la webcam solo podrá ser vista desde su red. Si desea que su webcam sea visible para todo Internet, debe configurar el *port forwarding* (reenvío de puertos) en su router. Para esto tendrá que iniciar sesión en la consola de administración del router, buscar la opción *port forwarding*, y habilitarla para el puerto 8081 de la dirección IP de Raspberry Pi.

Después, podrá ver la webcam mediante la dirección IP externa asignada por su ISP. Normalmente se muestra en la portada de la consola de administración. Pero tenga cuidado: a menos que pague a su ISP por una dirección IP estática, es probable que esta dirección IP cambie cada vez que reinicie su módem.

Los servicios como No-ip también se pueden utilizar para proporcionar un DNS dinámico, de modo que pueda registrar un nombre de dominio que siga automáticamente los cambios de su dirección IP.

Para saber más

Hay documentación más completa en la página web de motion.

Puede encontrar una lista de webcams compatibles con Raspberry Pi en *http://elinux.org/ RPi_USB_Webcams*.

Raspberry Pi tiene un módulo de cámara (Capítulo 1.15). En este momento no es compatible con *motion*, pero quizá lo sea cuando lea esto.

Para usar una webcam para proyectos de visión artificial, consulte el Capítulo 8.

En el Capítulo 3.17 puede obtener más información sobre el uso de `apt-get`.

4.7 Ejecutar un emulador de juegos de consola *vintage*

Problema

Quiere utilizar un emulador de juegos en Raspberry Pi para hacer una consola de juegos *vintage*.

Solución

Hay bastantes emuladores de consolas de juegos de los años ochenta. Uno de los más populares es Stella, un emulador para Atari 2600 (Figura 4.7).

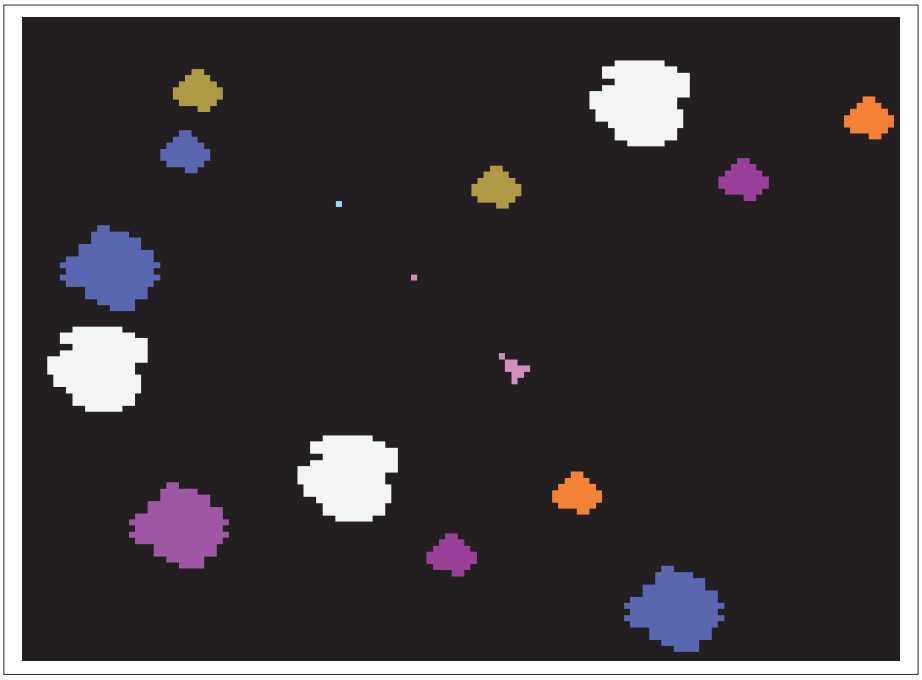


Figura 4.7. *Asteroides en el emulador Stella Atari 2600.*

Se debe señalar que, aunque estos juegos son antiguos, son propiedad de alguien. Los archivos de imagen ROM que necesita para jugar en un emulador como Stella—aunque sean fáciles de encontrar en la red—no necesariamente los puede coger. Así que adhiérase a la ley.

Para instalar Stella, simplemente introduzca el siguiente comando en una ventana del terminal:

```
$ sudo apt-get install stella
```

Una vez instalado, encontrará un nuevo programa en Games del menú de inicio. No lo ejecute todavía; necesita obtener la imagen ROM para un juego.

Si está fuera de Estados Unidos y posee el juego real, en muchos países tiene derecho a hacer una copia de la imagen ROM con fines de copia de seguridad, pero no en todos. También puede encontrar imágenes ROM para algunos juegos que se han liberado y no tienen restricciones de licencia.

Una vez tenga la imagen ROM para el juego al que desea jugar, cree una carpeta llamada *roms* y coloque la imagen ROM dentro. Después inicie Stella.

Haga clic en el archivo de imagen para empezar el juego. Por defecto, las teclas del cursor se asignan a los controles del *joystick* y la barra espaciadora es fuego. Puede cambiar algunos ajustes para el emulador; por ejemplo, puede ir a Video Settings y configurarlo en modo pantalla completa.

También, con Input Settings, en la pestaña Emul Events, puede asignarle los botones de control al teclado.

Observaciones

El emulador utiliza una cantidad sorprendentemente grande de los escasos recursos de Raspberry Pi, por lo que quizá necesite usar una Raspberry Pi 3 o 2.

Si busca en Internet, verá que muchas personas han llevado a cabo esta configuración básica y han añadido un controlador USB, como el USB Nintendo Retrolink Super SNES Classic Controller (disponible y barato), y han transformado Pi y un monitor en una gran consola estilo arcade.

Para saber más

Hay más emuladores de consola disponibles —en distintas fases de madurez y, por lo tanto, fiabilidad— para Raspberry Pi. Uno a considerar es Mame, que emula varias plataformas de juego diferentes.

Para obtener más información sobre la instalación con `apt-get`, consulte el Capítulo 3.17.

4.8 Ejecutar Minecraft Pi Edition

Problema

Quiere ejecutar el popular juego Minecraft en su Raspberry Pi.

Solución

Mojang, los desarrolladores originales de Minecraft, lo han llevado a Raspberry Pi y Minecraft Pi ya está preinstalado en la última distribución de Raspbian.

Figura 4.8. *Minecraft en Raspberry Pi.*

Observaciones

Para obtener Minecraft en Raspberry Pi los desarrolladores hicieron algunos recortes en el código de gráficos. Esto significa que solo puede jugar directamente en Raspberry Pi con el teclado, el ratón y el monitor conectados directamente. No funcionará con una conexión VNC remota.

Minecraft Pi Edition se basa en la versión móvil de Minecraft y carece de algunas características, sobre todo de las Redstone.

Para saber más

Obtenga más información sobre el puerto de Raspberry Pi de Minecraft.

También puede ejecutar una versión completa del servidor Minecraft en Raspberry Pi (Capítulo 4.9), aunque las Raspberry Pi 3 y 2 lo harán mucho mejor (consulte el Capítulo 1.2).

Minecraft Pi también incluye una interfaz de programación de Python, y en el Capítulo 7.21 aprenderá a enviar comandos a Python a través de una conexión SSH para hacerlo de forma automática.

4.9 Ejecutar un servidor de Minecraft

Problema

Desea alojar un servidor Minecraft de bajo consumo y de bajo coste para que usted y sus amigos puedan jugar.

Solución

El primer paso es instalar el entorno de ejecución de Java utilizando estos comandos:

```
$ sudo apt-get update
$ sudo apt-get install openjdk-7-jdk
```

A continuación, cree un directorio que contenga el código del servidor y obtenga el servidor de Minecraft mediante los comandos:

```
$ mkdir mcserver
$ cd mcserver
$ wget http://getspigot.org/spigot18/spigot_server.jar
```

Inicie el servidor con el comando:

```
$ java -Xms512M -Xmx800M -jar spigot_server.jar nogui
```

El servidor rápidamente le dará el siguiente mensaje de error:

```
[15:51:26 WARN]: Failed to load eula.txt
[15:51:26 INFO]: You need to agree to the EULA in order to run
the server. Go to eula.txt for more info.
[15:51:26 INFO]: Stopping server
```

Esto le indica que debe aceptar los términos de licencia editando el archivo llamado *eula.txt* que el servidor acaba de crear para usted. Por lo tanto, abra *eula.txt* utilizando `nano` mediante el siguiente comando:

```
$ nano eula.txt
```

Cambie la línea que dice `eula=false` a `eula=true` y guarde el archivo, luego ejecútelo y después reinicie el servidor. Esta vez le llevará varios minutos poner en marcha el servidor. No se preocupe, la próxima vez que lo ejecute será mucho más rápido.

Eventualmente, el servidor se iniciará y podrá probarlo conectándolo a otro ordenador que ejecute Minecraft en su red.

Inicie Minecraft, haga clic en Multiplayer y, a continuación, en Direct Connect. Introduzca la dirección IP de su Raspberry Pi (Figura 4-9) y haga clic en Connect.

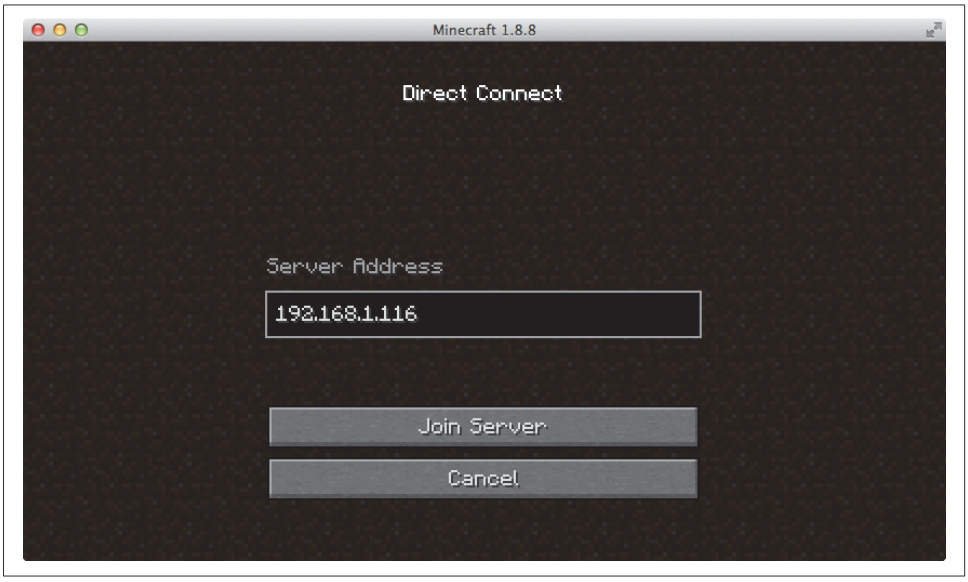


Figura 4.9. *Conectarse al servidor de Minecraft.*

Una vez conectado, usted y cualquier otra persona de su red puede unirse al servidor y comenzar a jugar (Figura 4.10).

Figura 4.10. *Jugar en un servidor Minecraft en Raspberry Pi.*

Probablemente querrá darse el estatus de *operador* en el juego escribiendo el comando:

```
> op YourName
```

donde `YourName` es su apodo en el juego.

Observaciones

Raspberry Pi 2 con sus cuatro núcleos de procesador y 1 GB de memoria ejecutará un servidor de Minecraft de manera bastante decente. Los parámetros `-Xms512M` y `-Xmx800M` establecen las cantidades mínima y máxima de memoria que Java debe utilizar para el servidor.

La ventana LXTerminal o SSH le informará cuando alguien se conecte al servidor y también podrá emitir varios comandos de administración del servidor desde aquí. Escriba `help` para obtener una lista de los comandos.

Cuando desee detener el servidor introduzca el comando `stop`:

```
[17:41:10 INFO]: UUID of player SimonMonk is 4e3e1a27-4e7e-4bfb-b331-a30af5bd49ec
[17:41:11 INFO]: SimonMonk[/192.168.1.5:58831] logged in with entity id
382 at ([world]257.23244892748414, 67.0, 185.3728463324922)
>stop
[17:41:20 INFO]: Stopping the server
[17:41:20 INFO]: Stopping server [17:41:20 INFO]: Saving players
[17:41:20 INFO]: SimonMonk lost connection: Server closed
[17:41:20 INFO]: SimonMonk left the game.
[17:41:21 INFO]: Saving worlds
[17:41:21 INFO]: Saving chunks for level 'world'/Overworld
[17:41:24 INFO]: Saving chunks for level 'world_nether'/Nether
[17:41:24 INFO]: Saving chunks for level 'world_the_end'/The End
>pi@Pi2headless ~/mcserver $
```

Para permitir que sus amigos de Internet jueguen en su servidor necesitará usar la consola de administración de su *router* para configurar el reenvío de puertos.

Para saber más

No necesita estar conectado directamente a Raspberry Pi para iniciar un servidor de Minecraft; funcionará correctamente si se inicia desde SSH (Capítulo 2.8).

Para obtener información sobre cómo ejecutar su propio servidor de Minecraft, consulte *http://uk.ign.com/ wikis/minecraft/Admin_and_Server_Commands*.

También puede consultar Minecraft Pi Edition (Capítulo 4.8).

4.10 Ejecutar Open Arena

Problema

Quiere ejecutar Open Arena en su Raspberry Pi.

Solución

Descargue y ejecute Open Arena (Figura 4.11) desde Pi Store.

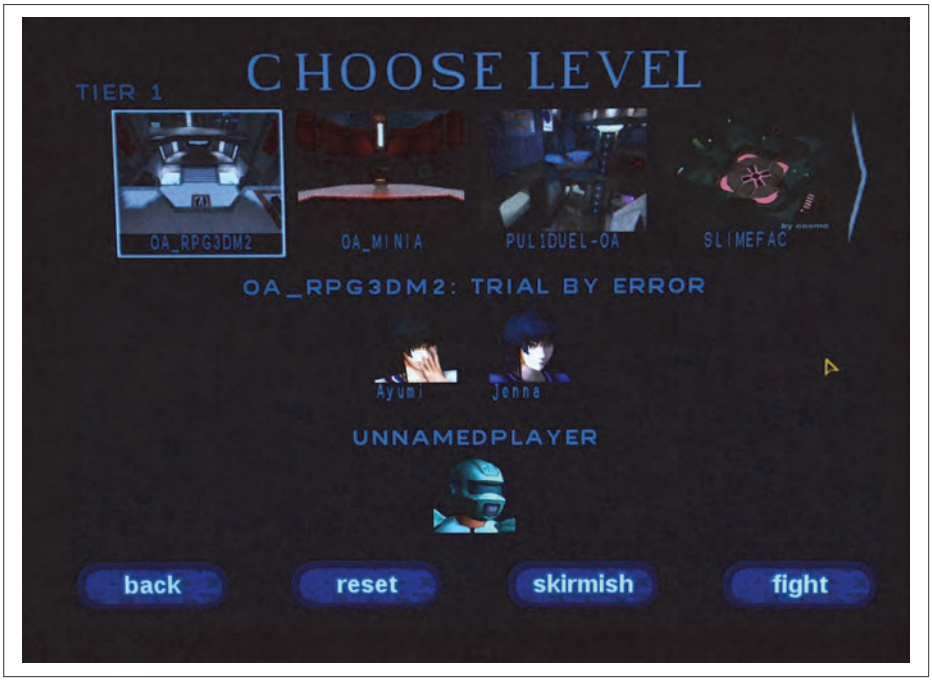


Figura 4.11. *Open Arena en Raspberry Pi.*

Observaciones

No es sorprendente que encuentre Open Arena en la sección Games de Pi Store. Aunque el juego es bastante violento y sangriento, se presenta con un entorno sencillo.

Para saber más

Obtenga más información sobre Open Arena.

Para obtener más información sobre el uso de Pi Store consulte el Capítulo 4.5.

4.11 Transmisor de radio Raspberry Pi

Asegúrese de revisar el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea convertir su Raspberry Pi en un transmisor FM de alta potencia que envíe una señal de radio a un receptor de radio FM normal (Figura 4.12).

Solución

Especialistas del Imperial College (Londres) han creado un código C y un script Python que le permite hacerlo. La descarga incluye el tema de *Star Wars* para que lo use como muestra.

Todo lo que necesita es un trozo corto de alambre conectado al pin 4 de su GPIO. Un cable encabezado de hembra a macho funcionará bien para esto. De hecho, debe funcionar teniendo la radio justo al lado de su Pi sin ningún tipo de antena, ya que la transmisión tiene mucha fuerza.

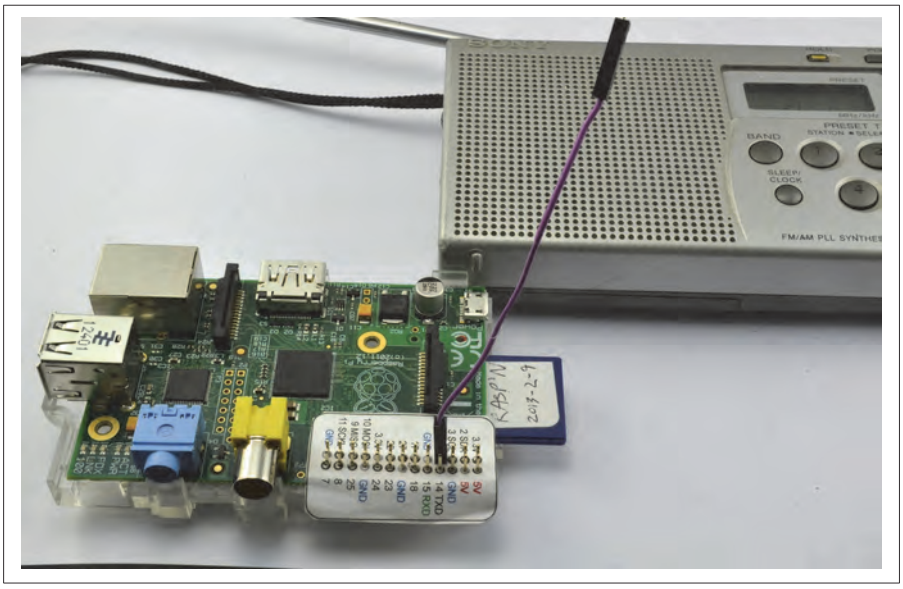


Figura 4.12. *Raspberry Pi como transmisor FM.*

El primer paso es instalar la biblioteca *pifm* utilizando los siguientes comandos:

```
$ mkdir pifm
$ cd pifm
$ wget http://www.icrobotics.co.uk/wiki/images/c/c3/Pifm.tar.gz
$ tar -xzf Pifm.tar.gz
```

A continuación, encuentre un receptor de radio FM y sintonice 103.0 MHz. Si esta frecuencia ya está ocupada por alguna otra transmisión, escoja otra frecuencia y tome nota de ella.

Luego ejecute el siguiente comando (cambiando el parámetro final de `103.0` si ha tenido que utilizar otra frecuencia):

```
sudo ./pifm sound.wav 103.0
```

Si todo está bien, debería oír los tonos de la canción de *Star Wars*.

Observaciones

Debe saber que este proyecto puede no ser legal en su país. La salida de energía es más alta que el de los transmisores de FM utilizados con reproductores MP3.

Puede reproducir otros archivos *.wav*, pero deben ser de 16 bits a 44.1 kHz.

El código también incluye una biblioteca de Python que puede utilizar dentro de sus propios programas de Python. Por lo tanto, podría escribir una interfaz de usuario para permitir la selección y reproducción de canciones.

El siguiente fragmento de código ilustra el uso de la interfaz de Python:

```
pi@raspberrypi ~/pifm $ sudo python
Python 2.7.3 (default, Jan 13 2013, 11:20:46)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import PiFm
>>> PiFm.play_sound("sound.wav")
```

Si pusiera Raspberry Pi en su vehículo, sería una manera excelente de tener sonido a través del sistema de audio, especialmente si utiliza las características de la última versión de *pifm*, que le permiten transmitir música al transmisor FM desde varias fuentes.

Para saber más

Esto se basa en la publicación original del Imperial College London. También incorpora trucos para canalizar la música desde varias fuentes a *pifm*, incluyendo un micrófono USN, que abre todo tipo de posibilidades.

4.12 Ejecutar GIMP

Problema

Quiere editar una imagen.

Solución

Descargue y ejecute el programa de edición de imágenes GNU (GIMP: vea la Figura 4.13).

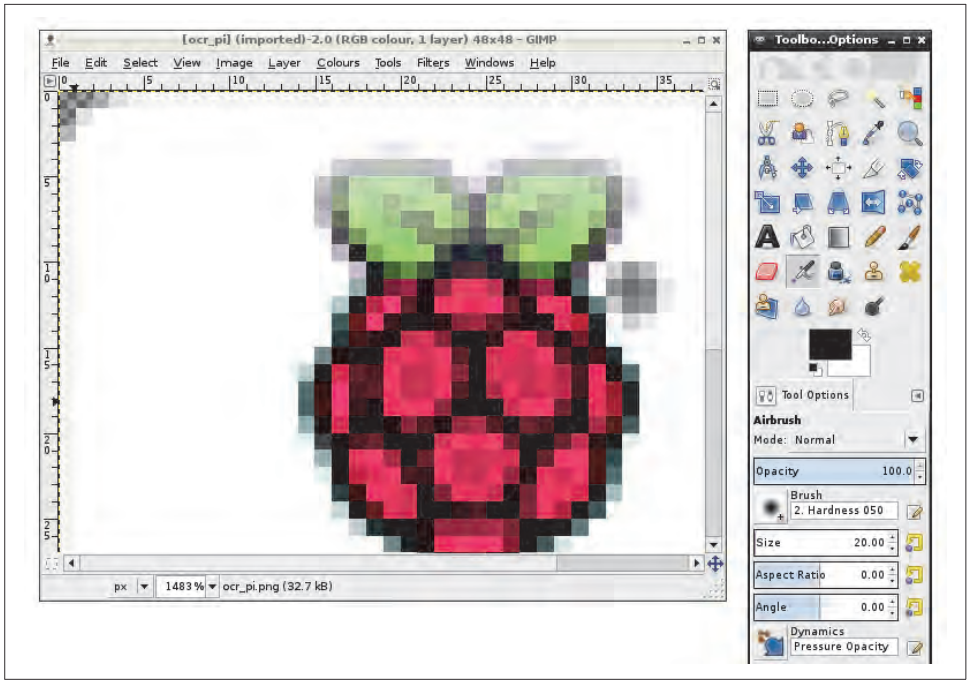


Figura 4.13. *GIMP en Raspberry Pi.*

Para instalar GIMP, abra una sesión de terminal y escriba el siguiente comando:

```
$ sudo apt-get install gimp
```

Una vez que GIMP esté instalado, encontrará una nueva entrada GNU Image Manipulation Program en el menú de inicio bajo el encabezado Graphics.

Observaciones

A pesar de que necesita mucha memoria y procesador, GIMP se puede utilizar en el modelo B de la Raspberry Pi B.

Para saber más

Obtenga más información en el sitio web de GIMP.

GIMP tiene un montón de características y es un programa de edición de imágenes muy sofisticado, por lo que necesitará un poco de aprendizaje. Encontrará un manual en línea en el sitio web de GIMP, en la pestaña Documentation.

Para obtener más información sobre la instalación con `apt-get`, consulte el Capítulo 3.17.

4.13 Radio por Internet

Problema

Quiere ser capaz de reproducir la radio por Internet en su Raspberry Pi.

Solución

Instale el reproductor multimedia VLC ejecutando el siguiente comando:

```
sudo apt-get install vlc
```

Una vez instalado, encontrará VLC en la sección Sound & Video de su menú de inicio.

Ejecute el programa y seleccione la opción Open Network Stream en el menú Media. Esto abrirá un cuadro de diálogo (véase la Figura 4-14) donde puede introducir la URL de la emisora de radio por Internet que desea reproducir.

Tendrá que enchufar auriculares o altavoces amplificados en la salida de audio en su Raspberry Pi.

Observaciones

VLC también puede ejecutarse desde la línea de comandos:

```
$ vlc http://www.a-1radio.com/listen.pls -I dummy
```

VLC probablemente producirá una serie de mensajes de error, pero después reproducirá el audio correctamente.

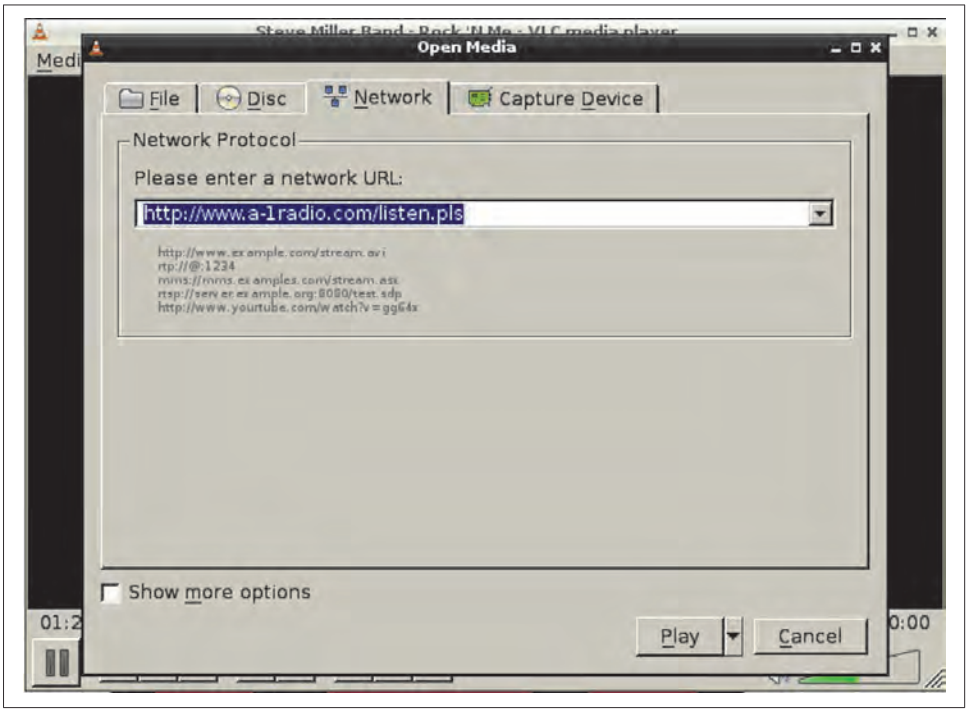


Figura 4.14. *VLC en Raspberry Pi.*

Para saber más

Este capítulo utiliza mucha información de este tutorial, en el que Jan Holst va un paso más allá y añade controles de estilo de radio al proyecto.

CAPÍTULO 5

Fundamentos de Python

5.1 Introducción

Existen muchos lenguajes para programar Raspberry Pi, pero Python es el más popular. De hecho, *Pi* en *Raspberry Pi* está inspirado en la palabra Python.

Aquí encontrará unas claves que le ayudarán a programar con Raspberry Pi.

5.2 Decidir entre Python 2 y Python 3

Problema

Es necesario utilizar Python, pero no está seguro de qué versión utilizar.

Solución

Utilice ambas versiones. Utilice Python 3 hasta que se encuentre con un problema que se resuelva mejor con la versión 2.

Observaciones

Aunque la última versión de Python es Python 3, mucha gente prefiere Python 2. De hecho, en la distribución de Raspbian, ambas versiones están instaladas, la versión 2 se llama Python, mientras que la versión 3 se llama Python 3. Incluso ejecutará Python 3 utilizando el comando `python3`. Los ejemplos de este libro están escritos para Python 3, a menos que se indique lo contrario. La mayoría se ejecutarán tanto en Python 2 como en Python 3 sin modificaciones.

Esta reticencia por parte de la comunidad Python de abandonar el antiguo Python 2 se debe, en gran parte, a que Python 3 introdujo algunos cambios que rompieron la compatibilidad con la versión 2. Esto significa que parte de las enormes bibliotecas de terceros que se han desarrollado para Python 2 no funcionan con Python 3.

Mi estrategia es escribir en Python 3 cuando sea posible, y volver a Python 2 cuando tenga problemas de compatibilidad.

Para saber más

Para leer un buen resumen del debate de Python 2 versus Python 3, vea el wiki de Python.

5.3 Editar programas de Python con IDLE

Problema

No está seguro de qué utilizar para escribir sus programas de Python.

Solución

Las distribuciones comunes de Raspberry Pi vienen con la herramienta de desarrollo IDLE Python en las versiones de Python 2 y Python 3. Si está utilizando Raspbian, encontrará accesos directos a ambas versiones de IDLE denominadas Python 2 y Python 3 en la sección Programming de su menú de inicio (Figura 5.1).

Figura 5.1. *Inicio de Python con IDLE.*

Observaciones

IDLE y IDLE 3 parecen ser idénticos; la única diferencia es la versión de Python que usan. Por lo tanto, abra IDLE 3 (Figura 5.2).

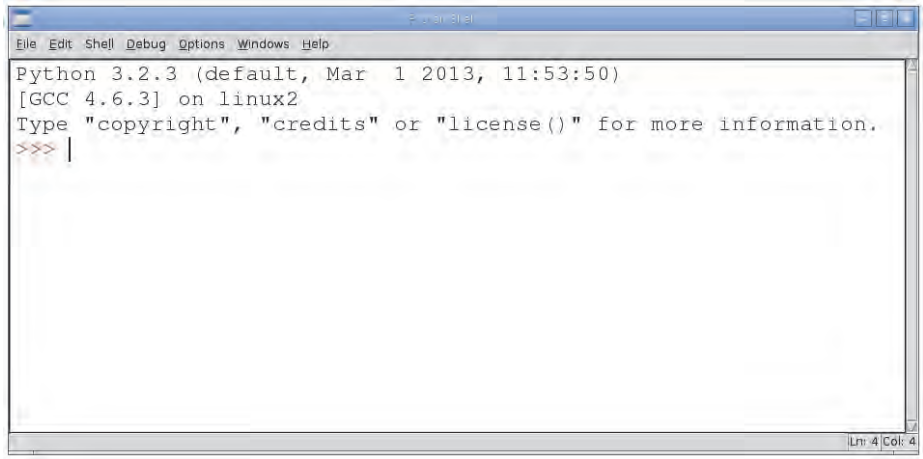


Figura 5.2. *Consola IDLE Python.*

La ventana que se abre se llama Python Shell. Es un área interactiva donde puede escribir **`Python`** y ver los resultados inmediatamente. Por lo tanto, intente escribir lo siguiente en el shell junto al indicador >>>:

```
>>> 2 + 2
4
>>>
```

Python ha evaluado la expresión `2 + 2` y ha dado la respuesta `4`.

Python Shell está bien para probar cosas, pero si quiere escribir programas necesita utilizar un editor. Para abrir un nuevo archivo para editar con IDLE, seleccione New Window desde el menú File (Figura 5.3).

Ahora puede escribir su programa en la ventana del editor. Solo para probar, pegue el siguiente texto en el editor, guarde el archivo como *count.py* y luego seleccione Run Module (ejecutar módulo) desde el menú Run. Puede ver los resultados del programa en la consola de Python:

```
for i in range(1, 10):
    print(i)
```

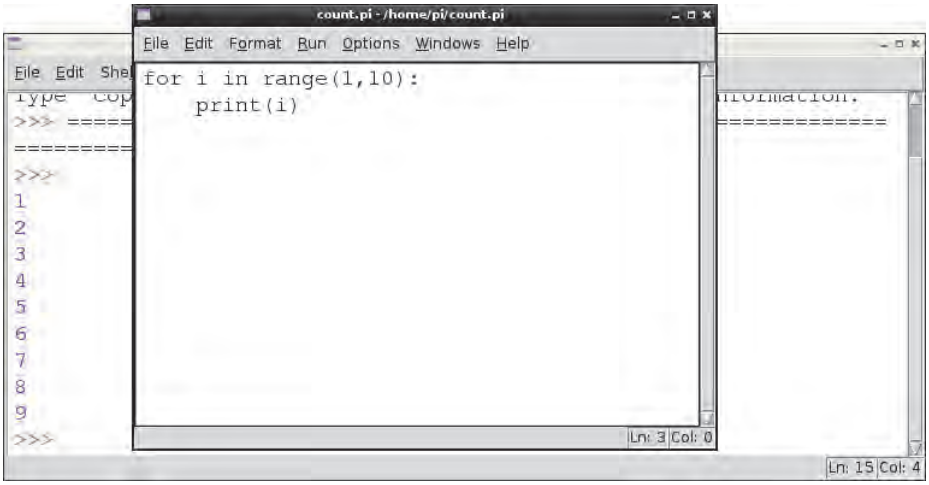


Figura 5.3. *Editor IDLE.*

En Python la sangría es una parte fundamental del lenguaje. Mientras que muchos lenguajes basados en C usan { y } para delimitar un bloque de código, Python utiliza la sangría. Por lo tanto, en el ejemplo anterior, Python sabe que `print` debe ser invocada repetidamente como parte del bucle `for`, porque está sangrado con cuatro espacios.

La convención usada en este libro es utilizar cuatro espacios como un nivel de sangría.

Cuando comience en Python, verá que es normal que aparezcan errores como `IndentationError: unexpected indent`, que significa que en algún lugar las cosas no están sangradas correctamente. Si todo parece alineado, compruebe que ninguna de las sangrías contenga caracteres de tabulación. Python trata las tabulaciones de manera distinta.

Observe cómo IDLE utiliza código de colores para resaltar la estructura del programa.

Para saber más

En este capítulo se utiliza IDLE para editar ejemplos de Python.

Además de usar IDLE para editar y ejecutar archivos Python, también puede editar archivos en `nano` (Capítulo 3.7) y luego ejecutarlos desde una sesión de terminal (Capítulo 5.5).

5.4 Usar la consola de Python

Problema

Desea introducir comandos Python sin escribir el programa entero.

Solución

Utilice la consola de Python, ya sea dentro de IDLE (Capítulo 5.3) o en una sesión de terminal.

Observaciones

Para iniciar una consola de Python 2 en una ventana de terminal, simplemente escriba el comando `Python`, y para iniciar una consola de Python 3 introduzca el comando `python3`.

El indicador `>>>` señala que puede escribir comandos de Python. Si necesita escribir comandos multilínea, la consola proporcionará automáticamente una línea de continuación indicada por tres puntos. Será necesario sangrar cualquiera de estas líneas con cuatro espacios, como se muestra a continuación:

```
>>> for i in range(1, 10):
...   print(i)
...
1
2
3
4
5
6
7
8
9
>>>
```

Tendrá que pulsar Intro dos veces después de su último comando para que la consola reconozca el final del bloque sangrado y ejecute el código.

La consola de Python también proporciona un historial de comandos para que pueda desplazarse hacia adelante y hacia atrás a través de sus comandos anteriores usando las teclas de arriba y abajo.

Para saber más

Si desea escribir más de un par de líneas, es mejor que utilice IDLE (Capítulo 5.3) para editar y ejecutar el archivo.

5.5 Ejecutar programas de Python desde el terminal

Problema

Ejecutar programas desde IDLE es correcto, pero a veces querrá ejecutar un programa de Python desde una ventana de terminal.

Solución

Utilice el comando `python` o `python3` en un terminal seguido del nombre del archivo que contiene el programa que desea ejecutar.

Observaciones

Para ejecutar un programa de Python 2 desde la línea de comandos, utilice un comando como este:

```
$ python myprogram.py
```

Si desea ejecutar el programa utilizando Python 3, cambie el comando `python` a `python3`. El programa de Python que desea ejecutar debe estar en un archivo con extensión *.py*.

Puede ejecutar la mayoría de los programas de Python como un usuario normal. Sin embargo, hay algunos, especialmente aquellos que utilizan el puerto GPIO, que necesitan ejecutarlo como superusuario. Si este es su caso, añada el prefijo `sudo`:

```
$ sudo python myprogram.py
```

En los ejemplos anteriores tiene que incluir `Python` en el comando para ejecutar el programa, pero también puede agregar una línea al inicio del programa para que Linux sepa que es un programa de Python. Esta línea especial se llama *shebang* (una contracción de *hash-bang*) y se puede ver en el siguiente ejemplo:

```
#!/usr/bin/python
print("I'm a program, I can run all by myself")
```

Antes de poder ejecutar esto directamente desde la línea de comandos tiene que darle los permisos de escritura mediante el siguiente comando (véase el Capítulo 3.14).

```
$ chmod +x test.py
```

El parámetro `+x` significa agregar el permiso de ejecución.

Ahora podrá ejecutar el programa de Python *test.py* mediante el siguiente comando:

```
$ ./test.py
I'm a program, I can run all by myself
$
```

Es necesario ./ al inicio de la línea para que la línea de comandos encuentre el archivo.

Para saber más

El Capítulo 3.24 le permite ejecutar un programa de Python como evento programado.

Para ejecutar automáticamente un programa al iniciar, vea el Capítulo 3.23.

5.6 Variables

Problema

Desea darle un nombre a un valor.

Solución

Asígnele un valor a un nombre usando =.

Observaciones

En Python no tiene que declarar el tipo de la variable, puede solamente asignarle un valor como se muestra en los siguientes ejemplos:

```
a = 123
b = 12.34
c = "Hello"
d = 'Hello'
e = True
```

Puede definir constantes de cadenas de caracteres utilizando comillas simples o dobles. Las constantes lógicas en Python son `True` o `False` y distinguen mayúsculas y minúsculas.

Por convención, los nombres de las variables comienzan con una letra minúscula y si el nombre de la variable consta de más de una palabra, las palabras se unen con un guion bajo. Siempre es buena idea dar a sus variables nombres descriptivos.

Algunos ejemplos de nombres de variable válidos son `x`, `total` y `number_of_chars`.

Para saber más

A las variables se les puede asignar un valor que sea una lista (Capítulo 6.2) o un diccionario (Capítulo 6.13).

Para obtener más información sobre la aritmética con variables, consulte el Capítulo 5.9.

5.7 Visualizar la salida

Problema

Desea ver el valor de una variable.

Solución

Utilice el comando `print`. Puede probar el siguiente ejemplo en la consola de Python (Capítulo 5.4):

```
>>> x = 10
>>> print(x)
10
>>>
```

Observaciones

En Python 2 puede utilizar el comando `print` sin paréntesis. Sin embargo, esto no lo puede hacer en Python 3, por lo que, para la compatibilidad con ambas versiones de Python, utilice paréntesis para el valor que quiere mostrar.

Para saber más

Para leer la entrada del usuario consulte el Capítulo 5.8.

5.8 Leer la entrada del usuario

Problema

Desea solicitar al usuario que introduzca un valor.

Solución

Utilice el comando `input` (Python 3) o `raw_input` (Python 2). Puede probar el siguiente ejemplo en la consola de Python 3 (Capítulo 5.4):

```
>>> x = input("Enter Value:")
Enter Value:23
>>> print(x)
23
>>>
```

Observaciones

En Python 2 `raw_input` debe sustituirse por `input` en el ejemplo anterior.

Python 2 también tiene la función `input`, pero esto valida la entrada e intenta convertirlo en un valor Python de un tipo apropiado, mientras que `raw_input` hace lo mismo que `input` en Python 3 y simplemente devuelve una cadena.

Para saber más

Obtenga más información sobre la entrada de Python 2.

5.9 Aritmética

Problema

Quiere hacer cálculos en Python.

Solución

Utilice los operadores +, -, * y /.

Observaciones

Los operadores más comunes para los cálculos son +, -, * y /, que son, respectivamente, sumar, restar, multiplicar y dividir.

También puede agrupar partes de la expresión con paréntesis como se muestra en el siguiente ejemplo, que, dada una temperatura en grados Celsius, lo convierte a grados Fahrenheit:

```
>>> tempC = input("Enter temp in C: ")
Enter temp in C:20
>>> tempF = (int(tempC) * 9) / 5 + 32
>>> print(tempF)
68.0
>>>
```

Otros operadores aritméticos incluyen % (módulo restante) y ** (elevar potencia). Por ejemplo, para elevar 2 a la potencia de 8 escribiría lo siguiente:

```
>>> 2 ** 8
256
```

Para saber más

Consulte el Capítulo 5.8 sobre el uso del comando `input`, y el Capítulo 5.13 sobre la conversión del valor de una cadena desde `input` a un número.

La biblioteca `Math` tiene muchas funciones matemáticas útiles que puede utilizar.

5.10 Crear cadenas

Problema

Desea crear una variable cadena.

Solución

Utilice el operador de asignación y una constante de cadena para crear una nueva cadena. Puede utilizar comillas dobles o simples alrededor de la cadena, pero deben coincidir.

Por ejemplo:

```
>>> s = "abc def"
>>> print(s)
abc def
>>>
```

Observaciones

Si va a incluir comillas dobles o simples dentro de una cadena, elija el tipo de comillas que no vaya a utilizar en los marcadores de inicio y fin de esta. Por ejemplo:

```
>>> s = "Isn't it warm?"
>>> print(s)
Isn't it warm?
>>>
```

A veces necesitará incluir caracteres especiales como la tabulación o una nueva línea dentro de la cadena. Esto requiere el uso de lo que se llama *caracteres de escape*. Para incluir una tabulación use `\t`, y para una nueva línea use `\n`. Por ejemplo:

```
>>> s = "name\tage\nMatt\t14"
>>>
print(s)
name    age
Matt    14
>>>
```

Para saber más

Para obtener una lista completa de los caracteres de escape, consulte el Manual de referencia de Python.

5.11 Concatenar (unir) cadenas

Problema

Desea unir varias cadenas.

Solución

Use el operador + (concatenar).

Por ejemplo:

```
>>> s1 = "abc"
>>> s2 = "def"
>>> s = s1 + s2
>>> print(s)
abcdef
>>>
```

Observaciones

En muchos lenguajes puede concatenar una cadena de valores donde algunos sean cadenas y otros sean de otro tipo como números. En ese caso, los números se convertirán automáticamente en cadenas durante la concatenación. Esto no ocurre en Python y, si intenta el siguiente comando, obtendrá un error:

```
>>> "abc" + 23
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object to str implicitly
```

Necesita convertir cada componente que quiera unir en una cadena antes de concatenar, como se muestra a continuación:

```
>>> "abc" + str(23)
'abc23'
>>>
```

Para saber más

Consulte el Capítulo 5.12 para obtener más información sobre cómo convertir números en cadenas utilizando la función `str`.

5.12 Convertir números en cadenas

Problema

Desea convertir un número en una cadena.

Solución

Utilice la función de Python `str`. Por ejemplo:

```
>>>  str(123)
'123'
>>>
```

Observaciones

Una razón común para querer convertir un número en una cadena es para poder unirla a otra cadena (Capítulo 5.11).

Para saber más

Para la operación inversa de convertir una cadena en un número, consulte el Capítulo 5.13.

5.13 Convertir cadenas en números

Problema

Desea convertir una cadena en un número.

Solución

Utilice la función de Python `int` o `float`.

Por ejemplo, para convertir `-123` en un número, puede utilizar:

```
>>> int("-123")
-123
>>>
```

Esto funcionará con número positivos y negativos.

Para convertir un número con coma flotante utilice `float` en lugar de `int`:

```
>>> float("00123.45")
123.45
>>>
```

Observaciones

Tanto `int` como `float` manejarán correctamente los ceros y tolerarán cualquier espacio o carácter de espacio en blanco que haya alrededor del número.

También puede utilizar `int` para convertir una cadena que representa un número en una base numérica distinta de la predeterminada de 10, suministrando la base numérica como segundo argumento. El siguiente ejemplo convierte la representación de la cadena del número binario 1001 en un número:

```
>>> int("1001", 2)
9
>>>
```

El segundo ejemplo convierte el número hexadecimal AFF0 en un número entero:

```
>>> int("AFF0", 16)
45040
>>>
```

Para saber más

Para la operación inversa de convertir un número en una cadena, consulte el Capítulo 5.12.

5.14 Conocer la longitud de una cadena

Problema

Necesita saber cuántos caracteres hay en una cadena.

Solución

Utilice la función `len` de Python.

Observaciones

Por ejemplo, para encontrar la longitud de la cadena `abcdef` debería utilizar:

```
>>>  len("abcdef")
6
>>>
```

Para saber más

El comando `len` también funciona en listas (Capítulo 6.4).

5.15 Conocer la posición de una cadena dentro de otra

Problema

Necesita encontrar la posición de una cadena dentro de otra.

Solución

Utilice la función de Python `find`.

Por ejemplo, para encontrar la posición inicial de la cadena `def` dentro de la cadena `abcdefghi` debería utilizar:

```
>>> s = "abcdefghi"
>>> s.find("def")
3
>>>
```

Tenga en cuenta que las posiciones de los caracteres comienzan en 0, por lo que una posición de 3 significa que es el cuarto carácter de la cadena.

Observaciones

Si la cadena que busca no existe en la cadena que indica, `find` devolverá el valor -1.

Para saber más

La función `replace` se utiliza para buscar y luego reemplazar todas las apariciones de una cadena (Capítulo 5.17).

5.16 Extraer una parte de una cadena

Problema

Desea cortar una parte de una cadena entre determinadas posiciones de caracteres.

Solución

Utilice la notación de Python `[:]`.

Por ejemplo, para cortar una parte de la cadena `abcdefghi` desde el segundo carácter al quinto utilizaría:

```
>>> s = "abcdefghi"
>>> s[1:5]
'bcde'
>>>
```

Tenga en cuenta que las posiciones de los caracteres comienzan en 0, por lo que una posición 1 significa que es el segundo carácter de la cadena y 5 significa que es el sexto. El rango de caracteres es exclusivo por lo que la letra *f* no se incluye en este ejemplo.

Observaciones

La notación `[:]` es en realidad muy potente. Usted puede omitir cualquiera de los argumentos, en el caso de que el principio o el final de la cadena se asuma como apropiado. Por ejemplo:

```
>>> s[:5]
'abcde'
>>>
```

y

```
>>> s = "abcdefghi"
>>> s[3:]
'defghi'
>>>
```

También puede utilizar índices negativos para contar hacia atrás desde el final de la cadena. Esto puede ser útil en situaciones como cuando desea encontrar la extensión de tres letras de un archivo, como en el siguiente ejemplo:

```
>>> "myfile.txt"[-3:]
'txt'
```

Para saber más

El Capítulo 5.11 describe cómo unir cadenas en lugar de separarlas.

El Capítulo 6.11 utiliza la misma sintaxis, pero con listas en lugar de cadenas.

5.17 Sustituir una cadena de caracteres por otra dentro de una cadena

Problema

Desea remplazar todas las apariciones de una cadena dentro de otra cadena.

Solución

Utilice la función `replace`.

Por ejemplo, para reemplazar todas las apariciones de `X` por `times` utilizaría:

```
>>> s = "It was the best of X. It was the worst of X"
>>> s.replace("X", "times")
'It was the best of times. It was the worst of times'
>>>
```

Observaciones

La cadena que busca debe coincidir exactamente. Es decir, la búsqueda distingue entre mayúsculas, minúsculas y espacios.

Para saber más

Consulte el Capítulo 5.15 para buscar una cadena sin realizar un reemplazo.

5.18 Convertir una cadena en letras mayúsculas o minúsculas

Problema

Desea convertir todos los caracteres de una cadena en letras mayúsculas o minúsculas.

Solución

Utilice la función `upper` o `lower` según corresponda.

Por ejemplo, para convertir `aBcDe` en mayúsculas utilice:

```
>>> "aBcDe".upper()
'ABCDE'
>>>
```

Para convertirlo en minúsculas use:

```
>>> "aBcDe".lower()
'abcde'
>>>
```

Observaciones

La mayoría de las funciones manipulan una cadena de alguna manera. `upper` y `lower` no modifican la cadena, sino que devuelven una copia modificada de esta.

Por ejemplo, el siguiente código devuelve una copia de la cadena `s`, pero observe que la cadena original no ha cambiado.

```
>>> s = "aBcDe"
>>> s.upper()
'ABCDE'
>>> s
'aBcDe'
>>>
```

Si desea cambiar el valor de `s` para que esté todo en mayúsculas, haga lo siguiente:

```
>>> s = "aBcDe"
>>> s = s.upper()
>>> s
'ABCDE'
>>>
```

Para saber más

Consulte el Capítulo 5.17 para reemplazar texto dentro de cadenas.

5.19 Ejecutar comandos de forma condicional

Problema

Desea ejecutar algunos comandos de Python cuando alguna condición sea verdadera.

Solución

Utilice el comando `if` de Python.

El siguiente ejemplo mostrará el mensaje `x is big` solo si `x` tiene un valor mayor que 100:

```
>>> x = 101
>>> if x > 100:
...     print("x is big")
...
x is big
```

Observaciones

Después de la palabra clave `if`, hay una *condición*. Esta condición a menudo, pero no siempre, compara dos valores y da una respuesta que es `True` o `False`. Si es `True`, todas las líneas sangradas posteriores serán ejecutadas.

Es bastante común querer que haga una cosa si una condición es `True` y querer otra diferente si la respuesta es `False`. En este caso, el comando `else` se utiliza con `if`, como se muestra en el ejemplo:

```
x = 101
if x > 100:
    print("x is big")
else:
    print("x is small")

print("This will always print")
```

También puede encadenar una serie larga de condiciones `elif`. Si alguna de las condiciones tiene éxito, ese bloque de código se ejecuta y ninguna de las otras se intentan. Por ejemplo:

```
x = 90
if x > 100:
    print("x is big")
elif x < 10:
    print("x is small")
else:
    print("x is medium")
```

Este ejemplo mostrará `x is medium`.

Para saber más

Consulte el Capítulo 5.20 para obtener más información sobre los diferentes tipos de comparaciones que puede hacer.

5.20 Comparar valores

Problema

Desea comparar valores.

Solución

Utilice uno de los operadores de comparación: <, >, <=, >=, ==, o !=.

Observaciones

Utilice los operadores < (menor que) y > (mayor que) del Capítulo 5.19. Aquí está el conjunto completo de operadores de comparación:

<	Menor que
>	Mayor que
<=	Menor que o igual a
>=	Mayor que o igual a
==	Exactamente igual a
!=	Diferente a

Algunas personas prefieren usar el operador <> en lugar de !=. Funcionan igual.

Puede probar los comandos utilizando la consola de Python (Capítulo 5.4), como se muestra en el siguiente intercambio:

```
>>> 1 != 2
True
>>> 1 != 1
False
>>> 10 >= 10
True
>>> 10 >= 11
False
>>> 10 == 10
True
>>>
```

Un error común es utilizar = (establecer un valor) en vez de == (doble igual) en las comparaciones. Esto puede ser difícil de detectar porque si la mitad de la comparación es una variable, es una sintaxis perfectamente legal y se ejecutará, pero no producirá el resultado esperado.

Además de comparar números, también puede comparar cadenas usando estos operadores de comparación, como se muestra aquí:

```
>>> 'aa' < 'ab'
True
>>> 'aaa' < 'aa'
False
```

Las cadenas se comparan lexicográficamente, es decir, en el orden en el que las encontraría en un diccionario. Esto no es del todo correcto porque para cada letra la versión en mayúsculas de la letra se considera menor que el equivalente en minúsculas. Cada letra tiene un valor que es su código ASCII.

Para saber más

Consulte también los Capítulos 5.19 y 5.21.

5.21 Operadores lógicos

Problema

Tiene que especificar una condición compleja en un enunciado `if`.

Solución

Utilice uno de los operadores lógicos: `and`, `or` y `not`.

Observaciones

Como ejemplo, puede que desee comprobar si una variable *x* tiene un valor entre 10 y 20. Para ello, debería utilizar el operador `and`:

```
>>> x = 17
>>> if x >= 10 and x <= 20:
...     print('x is in the middle')
...
x is in the middle
```

Puede combinar tantos enunciados `and` y `or` como desee y también puede utilizar paréntesis para agruparlos si las expresiones se complican.

Para saber más

Consulte los Capítulos 5.19 y 5.20.

5.22 Repetir instrucciones un número exacto de veces

Problema

Debe repetir un código de programa un número exacto de veces.

Solución

Utilice el comando de Python `for` y repita el comando en base a un rango.

Por ejemplo, para repetir un comando 10 veces utilice el siguiente ejemplo:

```
>>> for i in range(1, 11):
...     print(i)
...
1
2
3
4
5
6
7
8
9
10
>>>
```

Observaciones

El segundo parámetro en el comando `range` es exclusivo, es decir, para contar hasta 10 debe especificar un valor de 11.

Para saber más

Si la condición para detener el bucle es más complicada que simplemente repetir varias veces, consulte el Capítulo 5.23.

Si quiere repetir los comandos para cada elemento de una lista o de un diccionario, consulte los Capítulos 6.8 o 6.16, respectivamente.

5.23 Repetir instrucciones hasta que alguna condición cambie

Problema

Necesita repetir algunos códigos de programa hasta que algo cambie.

Solución

Utilice la instrucción `while` de Python. `while` repite los comandos hasta que su condición se vuelva falsa. El siguiente ejemplo permanecerá en bucle hasta que el usuario introduzca **X** para salir:

```
>>> answer = ''
>>> while answer != 'X':
...     answer = input('Enter command:')
...
Enter command:A
Enter command:B
Enter command:X
>>>
```

Observaciones

El ejemplo anterior utiliza el comando `input` porque funciona en Python 3. Para ejecutar el ejemplo en Python 2 sustituya el comando `input` por `raw_input`.

Para saber más

Si solo desea repetir algunos comandos varias veces, vea el Capítulo 5.22.

Si intenta repetir comandos para cada elemento de una lista o diccionario, consulte los Capítulos 6.8 o 6.16, respectivamente.

5.24 Interrumpir bucles

Problema

Está en un bucle y necesita salir si se producen ciertas condiciones.

Solución

Utilice la instrucción `break` de Python para salir de un bucle `while` o `for`.

El siguiente ejemplo se comporta igual que el código del ejemplo del Capítulo 5.23:

```
>>> while True:
...     answer = input('Enter command:')
...     if answer == 'X':
...             break
...
Enter command:A
Enter command:B
Enter command:X
>>>
```

Observaciones

Este ejemplo utiliza el comando `input`, ya que funciona en Python 3. Para ejecutar el ejemplo en Python 2, sustituya el comando `input` por `raw_input`.

Se comporta exactamente igual que el ejemplo del Capítulo 5.23. Sin embargo, en este caso, la condición para el bucle `while` es solo `True`, por lo que el bucle nunca terminará a menos que usemos `break` para salir de él cuando el usuario introduzca **X**.

Para saber más

También puede salir del bucle `while` utilizando una condición. Consulte el Capítulo 5.19.

5.25 Definir una función en Python

Problema

Quiere evitar repetir el mismo código una y otra vez en un programa.

Solución

Cree una función que agrupe líneas de código, permitiendo ser llamadas desde varios sitios.

La creación y la llamada de una función en Python se ilustra en el siguiente ejemplo:

```
def count_to_10():
    for i in range(1, 11):
        print(i)

count_to_10()
```

En este ejemplo se ha definido la función utilizando el comando `def` que mostrará los números entre 1 y 10 cada vez que se llame:

```
count_to_10()
```

Observaciones

Las convenciones para nombrar funciones son las mismas que para las variables (Capítulo 5.6), es decir, deben comenzar por una letra minúscula y, si el nombre consta de más de una palabra, deben están unidas con guiones bajos.

El ejemplo de función es un poco inflexible porque solo puede contar hasta 10. Si queremos hacerla más flexible, de modo que pueda contar hasta cualquier número, podemos incluir el número máximo como *parámetro* para la función, como se ve en este ejemplo:

```
def count_to_n(n):
    for i in range(1, n + 1):
        print(i)

count_to_n(5)
```

El nombre del parámetro `n` se incluye dentro de los paréntesis y luego se utiliza dentro del comando `range`, pero no antes de que se añada 1.

Utilizar un parámetro para el número hasta el que desea contar significa que cada vez que tenga que contar hasta un número diferente tendrá que especificar el número. Sin embargo, usted puede especificar un valor predeterminado para un parámetro y, por lo tanto, tener lo mejor de ambas cosas, como se muestra en el ejemplo:

```
def count_to_n(n=10):
    for i in range(1, n + 1):
        print(i)

count_to_n()
```

Esto contará hasta 10 a menos que se especifique un número diferente cuando llame a la función.

Si su función necesita más de un parámetro, tal vez para contar entre dos números, los parámetros se separarán por comas:

```
def count(from_num=1, to_num=10):
    for i in range(from_num, to_num + 1):
                print(i)

count()
count(5)
count(5,10)
```

Todos estos ejemplos son funciones que no devuelven ningún valor, solo hacen algo. Si necesita una función que le devuelva un valor, utilice el comando `return`.

La siguiente función toma una cadena como argumento y añade la palabra *please* al final de la cadena.

```
def make_polite(sentence):
    return sentence + " please"

print(make_polite("Pass the cheese"))
```

Cuando una función devuelve un valor puede asignar el resultado a una variable o, como en este ejemplo, mostrar el resultado.

Para saber más

Para devolver más de un valor desde una función consulte el Capítulo 7.4.

CAPÍTULO 6

Listas y diccionarios de Python

6.1 Introducción

En el Capítulo 5 hemos visto los conceptos básicos del lenguaje de Python. En este capítulo veremos las dos estructuras de datos clave de Python: listas y diccionarios.

6.2 Crear una lista

Problema

Quiere utilizar una variable que contenga una serie de valores en lugar de uno solo.

Solución

Utilice una lista. En Python una lista es una colección de valores almacenados en orden para que pueda acceder a ellos a través de su posición.

Cree una lista utilizando [y]:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>>
```

A diferencia de los conjuntos más rígidos en lenguajes como C, no tiene que especificar el tamaño de la lista en Python cuando la declare. También puede cambiar el número de elementos de la lista en el momento que desee.

Observaciones

Como ilustra este ejemplo, los elementos de una lista pueden no ser del mismo tipo, aunque a menudo lo sean.

Si desea crear una lista vacía a la que pueda añadir elementos más tarde, puede escribir:

```
>>> a = []
>>>
```

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.3 Acceder a elementos de una lista

Problema

Desea encontrar elementos individuales de una lista o cambiarlos.

Solución

Utilice [] para acceder a elementos de una lista por su posición en ella. Por ejemplo:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> a[1]
'Fred'
```

Observaciones

Las posiciones de la lista (índices) comienzan en 0 para el primer elemento.

Además de usar [] para leer valores de una lista, también puede usarlos para cambiar valores en una posición determinada. Por ejemplo:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> a[1] = 777
>>> a
[34, 777, 12, False, 72.3]
```

Si intenta cambiar (o, de hecho, leer) un elemento de una lista utilizando un índice que es demasiado grande, obtendrá error de "Índice fuera de rango":

```
>>> a[50] = 777
Traceback (most recent call last): File
  "<stdin>", line 1, in <module>
IndexError: list assignment index out of range
>>>
```

Para saber más

Todos los capítulos comprendidos entre 6.2 y 6.12 implican el uso de listas.

6.4 Conocer la longitud de una lista

Problema

Necesita saber cuántos elementos hay en una lista.

Solución

Utilice la función de Python `len`. Por ejemplo:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> len(a)
5
```

Observaciones

El comando `len` también funciona en cadenas (Capítulo 5.14).

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.5 Añadir elementos a una lista

Problema

Tiene que agregar un elemento a una lista.

Solución

Utilice las funciones de Python `append`, `insert` o `extend`.

Para añadir un solo elemento al final de una lista utilice `append`:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> a.append("new")
>>> a
[34, 'Fred', 12, False, 72.3, 'new']
```

Observaciones

A veces no desea agregar los elementos nuevos al final de la lista, sino que desea insertarlos en una posición determinada. Para ello, utilice el comando `insert`. El primer argumento es el índice en el que debe insertarse el elemento y el segundo es el elemento que se va a insertar:

```
>>> a.insert(2, "new2")
>>> a
[34, 'Fred', 'new2', 12, False, 72.3]
```

Observe como todos los elementos de después del elemento recién insertado se mueven una posición.

Tanto `append` como `insert` agregan solo un elemento a una lista. La función `extend` añade todos los elementos de una lista al final de otra:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> b = [74, 75]
>>> a.extend(b)
>>> a
[34, 'Fred', 12, False, 72.3, 74, 75]
```

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.6 Eliminar elementos de una lista

Problema

Tiene que eliminar un elemento de una lista.

Solución

Utilice la función de Python `pop`.

El comando `pop` sin parámetros elimina el último elemento de la lista:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> a.pop()
72.3
>>> a
[34, 'Fred', 12, False]
```

Observaciones

Observe que `pop` devuelve el valor eliminado de la lista.

Para quitar un elemento de una posición en concreto, en vez del último elemento, utilice `pop` con el parámetro de la posición del elemento que quiere eliminar:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> a.pop(0)
34
```

Si utiliza un índice con una posición que está más allá del final de la lista, obtendrá el mensaje: "Índice fuera de rango".

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.7 Crear una lista dividiendo una cadena

Problema

Tiene que convertir una cadena de palabras separadas por algún carácter en un conjunto de cadenas, con cada cadena del conjunto siendo una de las palabras.

Solución

Utilice la función de cadena `split`.

El comando `split` sin parámetros separa las palabras de una cadena en elementos individuales de un conjunto:

```
>>> "abc def ghi".split()
['abc', 'def', 'ghi']
```

Si aplica `split` con un parámetro, se dividirá la cadena utilizando el parámetro como separador. Por ejemplo:

```
>>> "abc--de--ghi".split('--')
['abc', 'de', 'ghi']
```

Observaciones

Este comando puede ser muy útil cuando, por ejemplo, esté importando datos de un archivo. El comando `split` puede, opcionalmente, tomar un argumento de la cadena para utilizarlo de delimitador cuando la esté dividiendo. Por lo tanto, si utiliza comas como separador, puede dividir la cadena de la siguiente manera:

```
>>> "abc,def,ghi".split(',')
['abc', 'def', 'ghi']
```

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.8 Iterar sobre una lista

Problema

Tiene que aplicar unas líneas de código a todos los elementos de una lista a la vez.

Solución

Utilice el comando de Python `for`:

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> for x in a:
...     print(x)
...
34
Fred
12
False
72.3
>>>
```

Observaciones

La palabra clave `for` va seguida inmediatamente por un nombre variable (en este caso `x`). Esto se llama variable bucle y se aplicará a cada uno de los elementos de la lista especificada después de `in`.

Las líneas sangradas que siguen se ejecutarán una vez por cada elemento de la lista. Cada vez, a `x` se le dará el valor del elemento de la lista en esa posición. Puede utilizar `x` para mostrar (*print*) el valor, como se muestra en el ejemplo anterior.

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.9 Enumerar una lista

Problema

Necesita ejecutar unas líneas de código para cada elemento de una lista a la vez, pero también necesita saber la posición de cada elemento.

Solución

Utilice el comando de Python `for` junto con el comando `enumerate`.

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> for (i, x) in enumerate(a):
...     print(i, x)
...
(0, 34)
(1, 'Fred')
(2, 12)
(3, False)
(4, 72.3)
>>>
```

Observaciones

Es normal que necesite conocer la posición de algo en una lista mientras se enumeran todos los valores. Un método alternativo es simplemente contar con una variable de índice y luego acceder al valor usando la sintaxis [] :

```
>>> a = [34, 'Fred', 12, False, 72.3]
>>> for i in range(len(a)):
...     print(i, a[i])
...
(0, 34)
(1, 'Fred')
(2, 12)
(3, False)
(4, 72.3)
>>>
```

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

Consulte el Capítulo 6.8 para iterar sobre una lista sin necesidad de conocer la posición de cada elemento.

6.10 Clasificar una lista

Problema

Debe ordenar los elementos de una lista.

Solución

Utilice el comando de Python `sort`:

```
>>> a = ["it", "was", "the", "best", "of", "times"]
>>> a.sort()
>>> a
['best', 'it', 'of', 'the', 'times', 'was']
```

Observaciones

Cuando ordene una lista en realidad la estará modificando en vez de devolver una copia ordenada de la lista original. Esto significa que, si también necesita la lista original, tendrá que usar el comando `copy` en la biblioteca estándar para hacer una copia antes de ordenarla:

```
>>> import copy
>>> a = ["it", "was", "the", "best", "of", "times"]
>>> b = copy.copy(a)
>>> b.sort()
>>> a
['it', 'was', 'the', 'best', 'of', 'times']
>>> b
['best', 'it', 'of', 'the', 'times', 'was']
>>>
```

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.11 Cortar una lista

Problema

Desea hacer una lista de una lista utilizando una serie de elementos de la lista original.

Solución

Utilice `[:]`. El siguiente ejemplo devuelve una lista que contiene los elementos de la lista original desde la posición 1 a la posición 2 (el número después de `:` es exclusivo):

```
>>> l = ["a", "b", "c", "d"]
>>> l[1:3]
['b', 'c']
```

Tenga en cuenta que las posiciones comienzan en 0, por lo que una posición 1 significa que es el segundo carácter de la cadena y 3 significa que es el cuarto. Sin embargo, el rango de caracteres es exclusivo por el final, por lo que la letra *d* no se incluye en el ejemplo.

Observaciones

`[:]` es en realidad muy poderoso. Puede omitir cualquiera de los argumentos en los casos en los que el inicio o el final de la lista se asuman como apropiados. Por ejemplo:

```
>>> l = ["a", "b", "c", "d"]
>>> l[:3]
['a', 'b', 'c']
```

```
>>> l[3:]
['d']
>>>
```

También puede utilizar índices negativos para contar desde el final de la lista. El siguiente ejemplo devuelve los dos últimos elementos:

```
>>> l[-2:] ['c', 'd']
```

Por cierto, `l[:-2]` devuelve `['a', 'b']` en el ejemplo anterior.

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

Consulte el Capítulo 5.16, donde se utiliza la misma sintaxis para las cadenas.

6.12 Aplicar una función a una lista

Problema

Desea aplicar una función a cada elemento de una lista y recopilar los cambios.

Solución

Utilice la función de Python llamada *comprensiones*.

El siguiente ejemplo convierte cada elemento de una cadena de la lista en mayúsculas y devuelve una nueva lista que tiene la misma longitud que la anterior, pero con todas las cadenas en mayúsculas:

```
>>> l = ["abc", "def", "ghi", "ijk"]
>>> [x.upper() for x in l]
['ABC', 'DEF', 'GHI', 'IJK']
```

A pesar de que puede llegar a confundir, no hay razón por la que no pueda combinar este tipo de expresiones, anidando una comprensión dentro de otra.

Observaciones

Esta es una manera muy concisa de hacer comprensiones. Toda la expresión debe estar entre corchetes ([]). El primer elemento de la comprensión es el código que va a ser evaluado para cada elemento de la lista. El resto de la comprensión se parece a un comando de repetición de una lista normal (Capítulo 6.8). La variable bucle sigue a la palabra `for`, y la lista que se va a utilizar sigue a la palabra `in`.

Para saber más

Todos los capítulos comprendidos entre el 6.2 y el 6.12 implican el uso de listas.

6.13 Crear un diccionario

Problema

Necesita crear una tabla de búsqueda donde se asocien valores con claves.

Solución

Utilice un diccionario de Python.

Los conjuntos van bien cuando necesite acceder a una lista de elementos ordenados o conozca la posición del elemento que desea utilizar. Los diccionarios son una alternativa a las listas para almacenar colecciones de datos, pero se organizan de manera diferente.

La Figura 6-1 muestra cómo se organiza un diccionario.

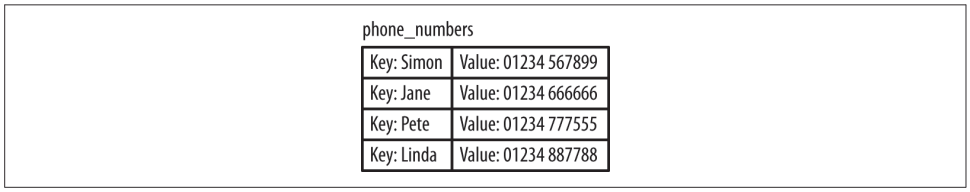


Figura 6.1. *Un diccionario de Python.*

Un diccionario almacena las parejas clave/valor de tal manera que pueda utilizar la claves (*key*) para recuperar su valor (*value*) de manera eficiente y sin tener que buscar en todo el diccionario.

Para crear un diccionario utilice {}:

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
```

Observaciones

En este ejemplo, las claves del diccionario son cadenas, pero no lo tienen que ser. Pueden ser números o cualquier tipo de datos, aunque las cadenas son las que normalmente se utilizan más.

Los valores también pueden ser de cualquier tipo de datos, incluyendo otros diccionarios o listas. El siguiente ejemplo crea un diccionario (`a`) y lo usa como valor en un segundo diccionario (`b`):

```
>>> a = {'key1':'value1', 'key2':2}
>>> a
{'key2': 2, 'key1': 'value1'}
```

```
>>> b = {'b_key1':a}
>>> b
{'b_key1': {'key2': 2, 'key1': 'value1'}}
```

Al mostrar el contenido de un diccionario, verá que el orden de los elementos del diccionario puede no coincidir con el orden que se especificó cuando se creó:

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> phone_numbers
{'Jane': '01234 666666', 'Simon': '01234 567899'}
```

A diferencia de las listas, los diccionarios no tienen el concepto de mantener los elementos en orden. Debido a la forma en la que se representan internamente, el orden de los contenidos en un diccionario será, a todos los efectos, aleatorio.

La razón por la que el orden es aleatorio se debe a que la estructura de datos es una *tabla hash*. Estas tablas utilizan la *función hashing* para decidir dónde almacenar el valor. Dicha función calcula un equivalente numérico a cualquier objeto.

Puede encontrar más información sobre las tablas hash en Wikipedia.

Para saber más

Los capítulos comprendidos entre el 6.13 y el 6.16 implican el uso de diccionarios.

6.14 Acceder a un diccionario

Problema

Necesita encontrar y cambiar entradas en un diccionario.

Solución

Use []. Introduzca la clave de la entrada a la que necesita acceder dentro de los corchetes.

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> phone_numbers['Simon']
'01234 567899'
>>> phone_numbers['Jane']
'01234 666666'
```

Observaciones

El proceso de búsqueda está en una sola dirección, desde la clave hasta el valor.

Si usa una *key* que no está en el diccionario, obtendrá un *key error*. Por ejemplo:

```
{'b_key1': {'key2': 2, 'key1': 'value1'}}
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> phone_numbers['Phil'] Traceback
(most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'Phil'
>>>
```

Además de usar [] para leer valores de un diccionario, también los puede usar para agregar nuevos valores o sobrescribir los existentes.

El siguiente ejemplo añade una nueva entrada en el diccionario: clave `Pete` y valor `01234 777555`:

```
>>> phone_numbers['Pete'] = '01234 777555'
>>> phone_numbers['Pete']
'01234 777555'
```

Si la clave no está en uso en el diccionario, se agregará automáticamente una nueva entrada. Si la clave ya está presente, el valor será sobrescrito por el nuevo.

Esto contrasta con intentar leer un valor del diccionario, donde si la clave es desconocida causará un error.

Para saber más

Los capítulos comprendidos entre el 6.13 y el 6.16 implican el uso de diccionarios. Para obtener más información sobre cómo tratar errores consulte el Capítulo 7.11.

6.15 Eliminar elementos de un diccionario

Problema

Debe eliminar un elemento de un diccionario.

Solución

Utilice el comando `pop`, especificando la clave del elemento que desea eliminar:

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> phone_numbers.pop('Jane')
'01234 666666'
>>> phone_numbers
{'Simon': '01234 567899'}
```

Observaciones

El comando pop devuelve el valor del elemento eliminado del diccionario.

Para saber más

Todos los capítulos comprendidos entre el 6.13 y el 6.16 implican el uso de diccionarios.

6.16 Iterar sobre diccionarios

Problema

Debe hacer algo a cada uno de los elementos del diccionario a la vez.

Solución

Utilice el comando for para iterar sobre las claves de un diccionario:

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> for name in phone_numbers:
...     print(name)
...
Jane
Simon
```

Observaciones

Hay un par de técnicas que puede utilizar para iterar sobre un diccionario. El siguiente formulario puede ser útil si necesita acceder a los valores o a las claves.

```
>>> phone_numbers = {'Simon':'01234 567899', 'Jane':'01234 666666'}
>>> for name, num in phone_numbers.items():
...     print(name + " " + num)
...
Jane 01234 666666
Simon 01234 567899
```

Para saber más

Los capítulos comprendidos entre el 6.13 y el 6.16 implican el uso de diccionarios.

Consulte el comando for utilizado en los capítulos 5.22, 6.8 y 6.12.

CAPÍTULO 7

Python avanzado

7.1 Introducción

En este capítulo exploraremos algunos de los conceptos más avanzados del lenguaje Python, en particular Python orientado a objetos, lectura y escritura de archivos, manejo de excepciones, uso de módulos y programación de Internet.

7.2 Dar formato a números

Problema

Desea mostrar números con un cierto número de posiciones decimales.

Solución

Aplique la cadena `format` al número. Por ejemplo:

```
>>> x = 1.2345678
>>> "x={:.2f}".format(x)
'x=1.23'
>>>
```

Observaciones

La cadena de formato puede contener una mezcla de texto regular y marcadores delimitados por { y }. Los parámetros de la función de formato (puede haber tantos como desee) sustituirán al marcador, de acuerdo con el especificador de formato.

En el ejemplo anterior, el especificador de formato es `:.2f`, que significa que el número se especificará con dos dígitos después del punto decimal y que es flotante `f`.

Si desea que el número tenga el formato para que la longitud del número sea siempre de siete dígitos (o espacios de relleno), agregue otro número antes del decimal:

```
>>> "x={:7.2f}".format(x)
'x=   1.23'
>>>
```

En este caso, dado que el número es solo de tres dígitos, hay cuatro espacios de relleno antes del 1. Si quiere que dichos espacios tomen forma de ceros, use:

```
>>> "x={:07.2f}".format(x)
'x=0001.23'
>>>
```

Un ejemplo más complicado podría ser mostrar la temperatura en grados Celsius y en grados Fahrenheit, como se muestra aquí:

```
>>> c = 20.5
>>> "Temperature {:5.2f} deg C, {:5.2f} deg F.".format(c, c * 9 / 5 + 32)
'Temperature 20.50 deg C, 68.90 deg F.'
>>>
```

Para saber más

Dar formato en Python implica un lenguaje de formato completo.

7.3 Dar formato a la fecha y hora

Problema

Desea convertir una fecha en una cadena y darle formato de una manera concreta.

Solución

Aplique la cadena `format` a la fecha. Por ejemplo:

```
>>> from datetime import datetime
>>> d = datetime.now()
>>> "{:%Y-%m-%d %H:%M:%S}".format(d)
'2015-12-09 16:00:45'
>>>
```

Observaciones

El lenguaje de formato de Python incluye algunos símbolos especiales para formatear la fecha. `%Y`, `%m` y `%d` corresponden a los números del año, mes y día, respectivamente.

Otros símbolos útiles para dar formato a la fecha son `%B`, que da el nombre completo del mes, y `%Y`, que da el año en forma de cuatro dígitos, como se muestra aquí:

```
>>> "{:%d %B %Y}".format(d)
'09 December 2015'
```

Para saber más

Consulte el Capítulo 7.2 para dar formato a los números.

Dar formato en Python realmente implica un lenguaje de formato completo. Encontrará todos los detalles de esto en *http://strftime.org/*.

7.4 Devolver más de un valor

Problema

Necesita escribir una función que devuelva más de un valor.

Solución

Devuelva una *tupla* y use la sintaxis de asignación de variables múltiples. Una *tupla* es una estructura de datos de Python que es como una lista, a excepción de que las tuplas están encerradas entre paréntesis en vez de entre corchetes. Su tamaño es fijo.

Por ejemplo, podría tener una función que convierta una temperatura en Kelvin a Fahrenheit y Celsius. Puede hacer que esta función devuelva la temperatura en ambas unidades, separando los valores de retorno por comas:

```
>>> def calculate_temperatures(kelvin):
...     celsius = kelvin - 273
...     fahrenheit = celsius * 9 / 5 + 32
...     return celsius, fahrenheit
...
>>> c, f = calculate_temperatures(340)
>>>
>>> print(c)
67
>>> print(f)
152.6
```

Cuando llame a la función, proporcione el mismo número de variables antes del = y cada uno de los valores de retorno será asignado a las variables en la misma posición.

Observaciones

A veces, cuando solo tiene uno o dos valores para devolver, esta es la forma correcta de devolver múltiples valores. Sin embargo, si los datos son complejos, es posible que la solución sea más nítida si utiliza las funciones orientadas a objetos de Python y

define una clase que contenga los datos. De este modo, puede devolver una instancia de clase en lugar de una tupla.

Para saber más

Consulte el Capítulo 7.5 para obtener información sobre la definición de clases.

7.5 Definir una clase

Problema

Necesita agrupar datos y funcionalidades relacionadas en una clase.

Solución

Defina una clase y proporciónele las variables de miembro que necesite.

El siguiente ejemplo define una clase para representar una entrada de la libreta de direcciones:

```
class Person:
    '''This class represents a person object'''

    def __init__ (self, name, tel):
        self.name = name
        self.tel = tel
```

La primera línea dentro de la definición de clase utiliza las comillas triples para indicar una cadena de documentación. Esto debe explicar el propósito de la clase. Aunque es totalmente opcional, añadir una cadena de documentación a una clase permite a los demás ver lo que hace esa clase. Esto es particularmente útil si la clase está disponible para otros usuarios.

Las cadenas de doc no son como los comentarios normales porque, aunque no son líneas activas de código, se asocian con la clase, por lo que en cualquier momento puede leer la cadena de doc para una clase mediante el siguiente comando:

```
Person.__doc__
```

Dentro de la definición de clase está el método constructor, que se le llamará automáticamente cada vez que cree una nueva instancia de la clase. Una clase es como una plantilla, por lo que al definir una clase llamada `Person`, no creamos ningún objeto de `Person` hasta más tarde:

```
def __init__ (self, name, tel):
    self.name = name
    self.tel = tel
```

El método constructor debe ser nombrado con doble guion bajo a cualquier lado de la palabra `init`.

Observaciones

Una forma en la que Python difiere de la mayoría de los lenguajes orientados a objetos es que debe incluir la variable especial `self` como un parámetro a todos los métodos que defina dentro de la clase. Esta es una referencia a, en este caso, la instancia recién creada. La variable `self` es el mismo concepto que la variable especial `this` que se encuentra en Java y en otros lenguajes.

El código en este método transfiere los parámetros que se le suministraron en variables miembro. Las variables miembro no necesitan ser declaradas de antemano, pero necesitan tener el prefijo `self.`.

Así que la siguiente línea:

```
self.name = name
```

crea una variable llamada `name` que es accesible a todos los miembros de la clase `Person` y lo inicia con el valor pasado en la llamada para crear una instancia, que tiene el siguiente aspecto:

```
p = Person("Simon", "1234567")
```

Podemos entonces comprobar que nuestro nuevo objeto `Person`, `p`, tiene el nombre `"Simon"` escribiendo lo siguiente:

```
>>> p.name
Simon
```

En un programa complejo, es una buena práctica colocar cada clase en su propio archivo con un nombre de archivo que coincida con el nombre de la clase. Esto también facilita la conversión de la clase en un módulo (consulte el Capítulo 7.12).

Para saber más

Consulte el Capítulo 7.6 para obtener información sobre la definición de métodos.

7.6 Definir un método

Problema

Necesita agregar un método a una clase.

Solución

El siguiente ejemplo muestra cómo puede incluir un método dentro de una definición de clase:

```
class Person:
    '''This class represents a person object'''
```

```
    def __init__ (self, first_name, surname, tel):
        self.first_name = first_name
        self.surname = surname
        self.tel = tel

    def full_name(self):
        return self.first_name + " " + self.surname
```

El método `full_name` une los atributos de nombre y apellidos de la persona con un espacio entre ellos.

Observaciones

Puede pensar en métodos como solo funciones que están vinculadas a una clase específica y que pueden o no utilizar variables miembro de esa clase en su proceso. Así, como con una función, puede escribir cualquier código que quiera en un método y también tener uno que llame a otro.

Si un método llama a otro dentro de la misma clase, la llamada al método tiene que tener el prefijo `self.`

Para saber más

Consulte el Capítulo 7.5 para obtener información sobre la definición de una clase.

7.7 Herencia

Problema

Necesita una versión especializada de una clase que ya existe.

Solución

Utilice *inheritance* (herencia) para crear una subclase de una clase existente y agregue nuevas variables miembro y métodos.

De forma predeterminada, todas las clases nuevas que crea son subclases de `object`. Puede cambiarlo especificando la clase que desea utilizar como superclase entre paréntesis después del nombre de la clase en una definición de esta. El siguiente ejemplo define una clase (`Employee`) como subclase de `Person` y añade una nueva variable miembro (`salary`) y un método adicional (`give_raise`):

```
class Employee(Person):

    def __init__ (self, first_name, surname, tel, salary):
        super(). init (first_name, surname, tel)
        self.salary = salary
```

```
    def give_raise(self, amount):
        self.salary = self.salary + amount
```

Tenga en cuenta que el ejemplo anterior es para Python 3. Para Python 2 no puede utilizar `super` de la misma manera. En su lugar debe escribir:

```
class Employee(Person):

    def init (self, first_name, surname, tel, salary):
        Person. init (self, first_name, surname, tel)
        self.salary = salary

    def give_raise(self, amount):
        self.salary = self.salary + amount
```

Observaciones

En ambos ejemplos el método inicializador de la subclase utiliza primero el método inicializador de la clase primaria (superclase) y luego agrega la variable miembro. Esto tiene la ventaja de que no tiene que repetir el código de inicialización en la nueva subclase.

Para saber más

Consulte el Capítulo 7.5 para obtener información sobre la definición de una clase.

El mecanismo de herencia de Python es realmente muy potente y soporta *herencia múltiple*, donde una subclase hereda de más de una superclase. Para obtener más información sobre la herencia múltiple consulte la documentación oficial de Python.

7.8 Escritura en un fichero

Problema

Necesita escribir algo en un fichero.

Solución

Utilice las funciones `open`, `write` y `close` para abrir un archivo, escribir algunos datos y después cerrar el archivo, respectivamente:

```
>>> f = open('test.txt', 'w')
>>> f.write('This file is not empty')
>>> f.close()
```

Observaciones

Una vez haya abierto el archivo, puede hacer tantas escrituras como desee antes de cerrarlo. Tenga en cuenta que es importante utilizar `close` porque, aunque cada escritura debe actualizar el archivo inmediatamente, puede almacenarse en la memoria intermedia y los datos podrían perderse. También podría dejar el archivo bloqueado para que otros programas no puedan abrirlo.

El método `open` toma dos parámetros. El primero es la ruta al archivo que se va a escribir. Esto puede ser relativo al directorio de trabajo actual o, si comienza con /, un camino absoluto.

El segundo parámetro es el modo en el que desea abrir el archivo. Para sobrescribir un archivo existente o crear el archivo con el nombre especificado si no existe, simplemente use `w`. La Tabla 7.1 muestra la lista completa de caracteres de modos de archivo. Los puede combinar con +. Por lo tanto, para abrir un archivo en modo de lectura binario utilizaría:

```
>>> f = open('test.txt', 'r+b')
```

Tabla 7.1. *Modos de archivo.*

Modo	Descripción
r	Leer
w	Escribir
a	Añadir al final de un archivo existente en lugar de sobrescribir
b	Modo binario
t	Modo de texto (predeterminado)
+	Atajo para r+w

El modo binario le permite leer o escribir flujos binarios de datos, como imágenes, en lugar de texto.

Para saber más

Para leer el contenido de un archivo consulte el Capítulo 7.9. Para obtener más información sobre el manejo de excepciones consulte el Capítulo 7.11.

7.9 Lectura de un fichero

Problema

Necesita leer el contenido de un archivo en una variable de cadena.

Solución

Para leer el contenido de un archivo debe utilizar los métodos de archivo `open`, `read` y `close`. El siguiente ejemplo lee todo el contenido del archivo en la variable `s`:

```
f = open('test.txt')
s = f.read()
f.close()
```

Observaciones

También puede leer archivos de texto una línea cada vez usando el método `readline`.

El ejemplo anterior lanzará una excepción si el archivo no existe o no es legible por alguna otra razón. Puede solucionarlo encerrando el código en una construcción `try/except` como esta:

```
try:
    f = open('test.txt')
    s = f.read()
    f.close()
except IOError:
    print("Cannot open the file")
```

Para saber más

Para escribir en un archivo y ver la lista de modos de apertura de un fichero vea el Capítulo 7.8. Para saber cómo manejar las excepciones consulte el Capítulo 7.11.

7.10 Pickling

Problema

Desea guardar todo el contenido de una estructura de datos en un archivo para que pueda leerse la próxima vez que se ejecute el programa.

Solución

Utilice la función de Python *pickling* para volcar la estructura de datos al archivo en un formato en el que se pueda volver a leer automáticamente en la memoria como una estructura de datos equivalente más adelante.

El siguiente ejemplo guarda una estructura de lista compleja en el archivo:

```
>>> import pickle
>>> mylist = ['some text', 123, [4, 5, True]]
>>> f = open('mylist.pickle', 'w')
>>> pickle.dump(mylist, f)
>>> f.close()
```

Para deshacer el *pickle* en el contenido del archivo en una nueva lista utilice:

```
>>> f = open('mylist.pickle')
>>> other_array = pickle.load(f)
>>> f.close()
>>> other_array
['some text', 123, [4, 5, True]]
```

Observaciones

Pickling funcionará en casi cualquier estructura de datos. No necesariamente ha de ser una lista.

El archivo se guarda en una especie de formato legible por el usuario, pero normalmente no necesitará buscar o editar el archivo de texto.

Para saber más

Para escribir en un archivo y ver la lista de modos de apertura de un fichero consulte el Capítulo 7.8.

7.11 Manejar excepciones

Problema

Si algo sale mal mientras se ejecuta un programa, quiere detectar el error o la excepción y mostrar un mensaje de error más sencillo.

Solución

Utilice `try/except` de Python.

El siguiente ejemplo, tomado del Capítulo 7.9, detecta cualquier problema al abrir un archivo:

```
try:
    f = open('test.txt')
    s = f.read()
    f.close()
except IOError:
    print("Cannot open the file")
```

Debido a que envolvió los comandos potencialmente propensos a errores al abrir el archivo en una construcción `try/except`, cualquier error que se produzca será capturado antes de que muestre cualquier mensaje de error, permitiéndole manejarlo a su manera. En este caso, significa que se mostrará el mensaje `"Cannot open the file"`.

Observaciones

Una situación común en la que se pueden producir excepciones en tiempo de ejecución, además de durante el acceso a archivos, es cuando se está accediendo a una lista y el índice que está utilizando esté fuera de los límites de la lista. Por ejemplo, sucede si intenta acceder al quinto elemento (índice 4) de una lista de tres:

```
>>> list = [1, 2, 3]
>>> list[4]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

Los errores y las excepciones se organizan en una jerarquía y puede ser tan específico o general como quiera al captar las excepciones.

`Exception` está en la parte superior de ese árbol y atrapará casi cualquier excepción. Puede tener secciones `except` separadas para capturar diferentes tipos de excepciones y manejarlas, cada una, de manera diferente. Si no especifica ninguna clase de excepción, se capturarán todas.

Python también le permite tener cláusulas `else` y `finally` en su manejo de errores:

```
list = [1, 2, 3]
try:
    list[8]
except:
    print("out of range")
else:
    print("in range")
finally:
    print("always do this")
```

La cláusula `else` se ejecutará si no hay excepción y la cláusula `finally` se ejecutará tanto si hay excepción como si no.

Cada vez que se produce una excepción puede obtener más información sobre ella mediante el objeto de excepción, que solo está disponible si utiliza la palabra clave `as`, como se muestra en el siguiente ejemplo:

```
>>> list = [1, 2, 3]
>>> try:
...     list[8]
... except Exception as e:
...     print("out of range")
...     print(e)
...
out of range
list index out of range
>>>
```

Esto le permite manejar el error a su manera, pero también mantener el mensaje de error original.

Para saber más

La documentación de Python para la jerarquía de clases de excepción de Python.

7.12 Usar módulos

Problema

Desea utilizar un módulo Python en su programa.

Solución

Utilice el comando `import`:

```
import random
```

Observaciones

Hay un gran número de módulos (a veces llamados *bibliotecas*) disponibles para Python. Muchas están incluidas con Python como parte de la biblioteca estándar y otras se pueden descargar e instalar en Python.

Las bibliotecas estándar de Python incluyen módulos para números aleatorios, acceso a bases de datos, varios protocolos de Internet, serialización de objetos y mucho más.

Una consecuencia de tener tantos módulos es que existe el potencial de conflicto, por ejemplo, si dos módulos tienen una función con el mismo nombre. Para evitar estos conflictos, al importar un módulo puede especificar cuánto de este es accesible.

Por ejemplo, si utiliza un comando como este:

```
import random
```

no hay posibilidad de conflicto porque solo podrá acceder a cualquier función o variable en el módulo escribiendo el prefijo `random` (por cierto, conocerá el paquete `random` en el siguiente capítulo).

Si, por otro lado, utiliza el siguiente comando, todo en el módulo será accesible sin tener que poner nada delante. A menos que sepa cuáles son todas las funciones en todos los módulos que está utilizando, existe una posibilidad mucho mayor de conflicto.

```
from random import *
```

Entre estos dos extremos puede especificar explícitamente los componentes de un módulo que necesite dentro de un programa para que puedan usarse convenientemente sin ningún prefijo.

Por ejemplo:

```
>>> from random import randint
>>> print(randint(1,6))
2
>>>
```

Una tercera opción es utilizar la palabra clave `as` para proporcionar un nombre más conveniente o significativo para el módulo al referenciarlo.

```
>>> import random as R
>>> R.randint(1, 6)
```

Para saber más

Para obtener una lista definitiva de los módulos incluidos con Python consulte el siguiente enlace *http://docs.python.org/2/ library/*.

7.13 Números aleatorios

Problema

Necesita generar un número aleatorio entre un rango de números.

Solución

Utilice la biblioteca `random`:

```
>>> import random
>>> random.randint(1, 6)
2
>>> random.randint(1, 6)
6
>>> random.randint(1, 6)
5
```

El número generado estará entre los dos argumentos (incluidos), en este caso, simulando un dado.

Observaciones

Los números generados no son verdaderamente aleatorios, sino que son lo que se conoce como *secuencia de números pseudoaleatorios*; es decir, son una larga secuencia de números que, cuando se toman en una cantidad suficientemente grande, muestran lo que los estadísticos llaman una *distribución aleatoria*. Para los juegos va bien, pero si está generando números de lotería, debería mirar el *hardware*

de asignación aleatoria. Los ordenadores son malos siendo aleatorios, no es realmente su naturaleza.

Un uso común de números aleatorios es seleccionar algo al azar de una lista. Puede hacerlo generando una posición índice y utilizándolo, pero hay un comando en el módulo `random` específicamente para esto. Pruebe el siguiente ejemplo:

```
>>> import random
>>> random.choice(['a', 'b', 'c'])
'a'
>>> random.choice(['a', 'b', 'c'])
'b'
>>> random.choice(['a', 'b', 'c'])
'a'
```

Para saber más

Consulte la referencia oficial del paquete aleatorio para obtener más información al respecto.

7.14 Hacer peticiones web desde Python

Problema

Necesita leer el contenido de una página web en una cadena usando Python.

Solución

Python tiene una extensa biblioteca para hacer peticiones HTTP.

El siguiente ejemplo de Python 2 lee el contenido de la página de inicio de Google en la cadena `contents`:

```
import urllib2
contents = urllib2.urlopen("https://www.google.com/").read()
print(contents)
```

Observaciones

Si está utilizando Python 3 en lugar de Python 2, tendrá que cambiar su ejemplo a:

```
import urllib.request
response = urllib.request.urlopen('http://python.org/')
contents = response.read()
print(contents)
```

Después de haber leído el código HTML es probable que desee buscar y extraer las partes del texto que realmente quiera. Para ello, tendrá que utilizar las funciones de manipulación de cadenas (consulte los Capítulos 5.15 y 5.16).

Para saber más

Para obtener más ejemplos relacionados con Internet en Python vea el Capítulo 15.

7.15 Argumentos de la línea de comandos en Python

Problema

Desea ejecutar un programa Python desde la línea de comandos y pasarle los parámetros.

Solución

Importar `sys` y utilizar su propiedad `argv`, como se muestra en el siguiente ejemplo. Esto devuelve un conjunto, cuyo primer elemento es el nombre del programa. Los otros elementos son los parámetros (separados por espacios) que se escribieron en la línea de comandos después del nombre del programa.

```
import sys

for (i, value) in enumerate(sys.argv):
    print("arg: %d %s " % (i, value))
```

Ejecutar el programa desde la línea de comandos con algunos parámetros da como resultado la siguiente salida:

```
$ python cmd_line.py a b c
arg: 0 cmd_line.py
arg: 1 a
arg: 2 b
arg: 3 c
```

Observaciones

Ser capaz de especificar los argumentos de la línea de comandos puede ser útil para automatizar la ejecución de los programas de Python, ya sea al inicio (Capítulo 3.22) o sobre una base cronometrada (Capítulo 3.24).

Para saber más

Para obtener información básica sobre la ejecución de Python desde la línea de comandos, consulte el Capítulo 5.5. Al imprimir `argv` usamos enumeraciones de listas (Capítulo 6.9).

7.16 Ejecutar comandos de Linux desde Python

Problema

Desea ejecutar un comando o programa Linux desde su programa Python.

Solución

Utilice el comando `system`.

Por ejemplo, para eliminar un archivo llamado *myfile.txt* del directorio desde el que inició Python, podría hacer lo siguiente:

```
import os
os.system("rm myfile.txt")
```

Observaciones

A veces, en lugar de simplemente ejecutar un comando como en el ejemplo anterior, necesita capturar la respuesta del comando. Digamos que quería usar el comando `'hostname'` para encontrar la dirección IP (vea el Capítulo 2.3) de Raspberry Pi. En este caso, puede utilizar la función `check_output` en la biblioteca `subprocess`.

```
import subprocess
ip = subprocess.check_output(['hostname', '-I'])
```

En este caso, la variable `ip` contendrá la dirección IP de Raspberry Pi. A diferencia de `system`, `check_output` necesita el comando y cualquier parámetro que se suministre como elementos separados de un conjunto.

Para saber más

Para obtener información sobre la biblioteca del sistema operativo, vaya al siguiente enlace:

http://www.pythonforbeginners.com/os/ pythons-os-module.

Para obtener más información sobre la biblioteca de `subprocess` consulte:

https://docs.python.org/3/ library/subprocess.html.

En el Capítulo 14.5 encontrará un ejemplo que utiliza subprocesos para mostrar la dirección IP, el hostname y la hora de su Raspberry Pi en una pantalla LCD.

7.17 Enviar correos electrónicos desde Python

Problema

Desea enviar un correo electrónico desde un programa de Python.

Solución

Python tiene una biblioteca para el Protocolo para Transferencia Simple de Correo (SMTP), que puede utilizar para enviar correos electrónicos:

```
import smtplib

GMAIL_USER = 'your_name@gmail.com'
GMAIL_PASS = 'your_password'
SMTP_SERVER = 'smtp.gmail.com'
SMTP_PORT = 587

def send_email(recipient, subject, text):
    smtpserver = smtplib.SMTP(SMTP_SERVER, SMTP_PORT)
    smtpserver.ehlo()
    smtpserver.starttls()
    smtpserver.ehlo
    smtpserver.login(GMAIL_USER, GMAIL_PASS)
    header = 'To:' + recipient + '\n' + 'From: ' + GMAIL_USER
    header = header + '\n' + 'Subject:' + subject + '\n'
    msg = header + '\n' + text + ' \n\n'
    smtpserver.sendmail(GMAIL_USER, recipient, msg)
    smtpserver.close()

send_email('destination_email_address', 'sub', 'this is text')
```

Para utilizar este ejemplo y enviar un correo electrónico a una dirección concreta, cambie las variables `GMAIL_USER` y `GMAIL_PASS` para que coincidan con sus credenciales de correo electrónico. Si no está utilizando Gmail, también tendrá que cambiar los valores de `SMTP_SERVER` y, posiblemente, `SMTP_PORT`.

También debe cambiar la dirección de correo electrónico de destino en la última línea.

Observaciones

El mensaje `send_email` simplifica el uso de la biblioteca `smtplib` en una sola función que puede reutilizar en sus proyectos.

Ser capaz de enviar correos electrónicos desde Python abre todo tipo de oportunidades de proyectos. Por ejemplo, puede utilizar un sensor, como el sensor PIR, para enviar un correo electrónico cuando se detecte movimiento.

Para saber más

Para ver un ejemplo similar que utiliza el servicio web IFTTT para enviar correos electrónicos consulte el Capítulo 15.4.

Para realizar peticiones HTTP desde Raspberry Pi consulte el Capítulo 7.14.

Para obtener más información sobre `smtplib` vea *http://docs.python.org/2/library/smtplib.html*.

Para tener más información relacionada con Internet revise el Capítulo 15.

7.18 Escribir un servidor web simple en Python

Problema

Necesita crear un servidor web Python simple, pero no quiere tener que ejecutar una pila completa de servidores web.

Solución

Utilice la biblioteca `bottle` de Python para ejecutar un servidor web Python que responda a las peticiones HTTP.

Para instalar `bottle` utilice el siguiente comando:

```
sudo apt-get install python-bottle
```

El siguiente programa (*bottle_test* en los recursos del libro) presenta un mensaje que muestra la hora que Pi cree que es. La Figura 7.1 muestra la página que verá si conecta Raspberry Pi desde un navegador en cualquier parte de su red:

```
from bottle import route, run, template
from datetime import datetime

@route('/')
def index(name='time'):
    dt = datetime.now()
    time = "{:%Y-%m-%d %H:%M:%S}".format(dt)
    return template('<b>Pi thinks the date/time is: {{t}}</b>', t=time)

run(host='0.0.0.0', port=80)
```

Para iniciar el programa debe ejecutarlo con privilegios de superusuario:

```
$ sudo python bottle_test.py
```

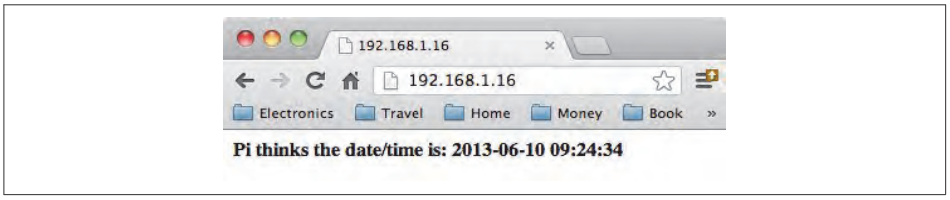


Figura 7.1. *Navegar en un servidor web bottle de Python.*

Este ejemplo requiere una pequeña explicación.

Después de los comandos `import`, el comando `@route` enlaza la ruta URL / con la función del manipulador que la sigue.

La función del manipulador formatea la fecha y la hora y luego devuelve una cadena del código HTML que debe representar el navegador. En este caso, utiliza una plantilla en la que los valores pueden ser sustituidos.

La línea `run` final inicia el proceso del servicio web. El puerto 80 es el predeterminado por los servidores web, por lo que, si desea utilizar un puerto diferente, debe agregar el número de puerto con `:` después de la dirección del servidor.

Observaciones

Puede definir tantas rutas y manipuladores como desee dentro del programa.

`bottle` es perfecto para proyectos de servidor web pequeños y sencillos. Está escrito en Python y es muy fácil escribir una función de manipulador para controlar el *hardware* en respuesta al usuario que interactúa con la página en un navegador. Encontrará otros ejemplos del uso de `bottle` en el Capítulo 15.

Para saber más

Para obtener más información consulte la documentación del proyecto `bottle`.

Para obtener más información sobre el formato de fechas y horas en Python consulte el Capítulo 7.3.

Para tener más información relacionada con Internet consulte el Capítulo 15.

7.19 Hacer más de una cosa a la vez

Problema

Su programa Python está ocupado haciendo una cosa y quiere que haga otra al mismo tiempo.

Solución

Utilice la biblioteca `thread` de Python.

El siguiente ejemplo, que puede descargar desde la sección de descargas de la página web del libro, establece un proceso de ejecución que interrumpe el conteo del hilo (*thread*) principal.

```
import thread, time, random

def annoy(message):
    while True:
        time.sleep(random.randint(1, 3))
        print(message)

thread.start_new_thread(annoy, ('BOO !!',))

x = 0
while True:
    print(x)
    x += 1
    time.sleep(1)
```

La salida de la consola se verá así:

```
$ python thread_test.py
0
1
BOO !!
2
BOO !!
3
4
5
BOO !!
6
7
8
```

Cuando inicie un nuevo *subproceso de ejecución* mediante la biblioteca de subprocesos de Python, debe especificar la función que se va a ejecutar como subproceso. En este ejemplo, la función se llama `annoy` y contiene un bucle que continuará indefinidamente imprimiendo un mensaje después de un intervalo aleatorio de entre 1 y 3 segundos.

Para iniciar el subproceso se llama a la función `start_new_thread` del módulo de subproceso. Esta función espera dos parámetros: el primero es el nombre de la función a ejecutar (en este caso `annoy`) y el segundo es una tupla que contiene los parámetros que se van a pasar a la función (en este caso `'BOO !!'`).

Puede ver que el hilo principal que está contando será interrumpido cada pocos segundos por el subproceso de ejecución en la función `annoy`.

Observaciones

Los subprocesos como estos también se llaman a veces *procesos ligeros* porque son similares a tener más de un programa o proceso ejecutándose a la vez. Sin embargo, tienen la ventaja de que los subprocesos que se ejecutan en el mismo programa tienen acceso a las mismas variables y, cuando el hilo principal del programa acaba, también lo hacen los subprocesos que se inician en él.

Para saber más

Para una buena introducción a los subprocesos en Python vaya a *http://www.tutorialspoint.com/ python/python_multithreading.htm*.

7.20 No hacer nada en Python

Problema

Quiere que Python pare el proceso durante un rato. Es posible que desee hacerlo para retrasar el envío de mensajes al terminal.

Solución

Utilice la función `sleep` en la biblioteca `time` como se ilustra en el siguiente ejemplo de código que puede encontrar en las descargas del libro llamado *sleep_test.py*.

```
import time

x = 0
while True:
    print(x)
    time.sleep(1)
    x += 1
```

El bucle principal del programa se demorará un segundo antes de mostrar el siguiente número.

Observaciones

La función `time.sleep` toma un número de segundos como parámetro. Si desea retrasos más cortos puede especificarlo con decimales. Por ejemplo, para retrasarlo un milisegundo utilizaría `time.sleep(0.001)`.

Es buena idea poner un pequeño retraso en cualquier bucle que continúe indefinidamente, o incluso que solo continúe durante más de una fracción de segundo, porque cuando se llama a *sleep*, el procesador se libera para permitir a otros procesos trabajar.

Para saber más

Para una discusión interesante sobre cómo `time.sleep` puede reducir la carga de CPU de su programa Python vaya al siguiente enlace:

http://raspberrypi.stackexchange.com/questions/8077/how-can-i-lower-the-usage-of-cpu-for-this-python-program.

7.21 Usar Python con Minecraft Pi Edition

Problema

Ahora que conoce Python quiere usarlo con Minecraft.

Solución

Utilice la interfaz de Python que viene con la edición Minecraft Pi para interactuar con Minecraft Pi mientras se está ejecutando.

Aunque se puede cambiar entre una sesión de LXTerminal y el juego de Minecraft, el juego se detendrá cada vez que la ventana pierda el foco, por lo que será mucho más fácil conectarse a Raspberry Pi mediante SSH en un segundo ordenador (Capítulo 2.8). Esto tiene el beneficio adicional de que puede ver el *script* de Python en acción y estar en vivo en el juego.

El siguiente ejemplo, que se puede descargar de la sección de descargas de la página web del libro, construye una escalera en su ubicación actual (Figura 7-2):

```
from mcpi import minecraft, block
mc = minecraft.Minecraft.create()

mc.postToChat("Lets Build a Staircase!")

x, y, z = mc.player.getPos()

for xy in range(1, 50):
    mc.setBlock(x + xy, y + xy, z, block.STONE)
```

La biblioteca de Python viene preinstalada en Raspbian, así que, si no está allí, probablemente debería actualizar su sistema (Capítulo 1.6).

Después de importar la biblioteca, la variable `mc` se asigna a una nueva instancia de la clase Minecraft.

El método `postToChat` envía un mensaje a la pantalla del jugador diciéndole que la escalera está a punto de ser construida.

Las variables `x`, `y`, y `z` están enlazadas a la posición del jugador y, luego, el bucle `for` incrementa tanto la `x` como la `y` (`y` es la altura) repetidamente para construir la escalera usando el método `setBlock` (Figura 7.2).

Figura 7.2. *Construyendo una escalera en Minecraft Pi con Python.*

Observaciones

La biblioteca `mcpi` no solo permite publicar en el chat, encontrar la ubicación del jugador y colocar un bloque, sino que también proporciona una gran cantidad de otros métodos que le permiten:

- Encontrar el ID del bloque en las coordenadas especificadas.
- Conocer quién está jugando.
- Establecer la posición de un jugador.
- Obtener la dirección hacia quien se enfrenta un jugador

No hay documentación definitiva aparte de esta útil publicación de blog.

Para saber más

Para obtener más información sobre Minecraft Pi Edition consulte el Capítulo 4.8.

Para aprender cómo ejecutar un servidor Minecraft en Raspberry Pi, revise el Capítulo 4.9.

CAPÍTULO 8

Visión artificial

8.1 Introducción

La visión artificial (*Computer Vision* - CV) le permite a su Raspberry Pi ver cosas. En términos prácticos, esto significa que su Raspberry Pi puede analizar una imagen buscando cosas de interés e incluso reconociendo caras y texto.

Si vincula esto con una cámara para capturar las imágenes, se abre un abanico de posibilidades.

8.2 Instalar SimpleCV

Problema

Desea instalar el *software* SimpleCV en su Raspberry Pi.

Solución

Para instalar SimpleCV primero instale los paquetes de requisitos previos mediante estos comandos:

```
$ sudo apt-get update
$ sudo apt-get install ipython python-opencv python-scipy
$ sudo apt-get install python-numpy python-setuptools python-pip
$ sudo pip install svgwrite
```

A continuación, instale SimpleCV con el comando:

```
$ sudo pip install https://github.com/sightmachine/SimpleCV/zipball/master
```

Una vez completada la instalación puede comprobar que todo ha ido bien ejecutando este comando:

```
$ simplecv
+-----------------------------------------------------------+
 SimpleCV 1.3.0 [interactive shell] - http://simplecv.org
+-----------------------------------------------------------+
Commands:
    "exit()" or press "Ctrl+ D" to exit the shell
    "clear()" to clear the shell screen
    "tutorial()" to begin the SimpleCV interactive tutorial
    "example()" gives a list of examples you can run
    "forums()" will launch a web browser for the help forums
    "walkthrough()" will launch a web browser with a walkthrough
```

Esto abrirá la consola SimpleCV. Se trata de una consola Python con funciones adicionales para SimpleCV.

Observaciones

SimpleCV es un entorno de Python alrededor del *software* OpenCV. SimpleCV, como su nombre indica, simplifica el uso de OpenCV. Si quieres conocer toda la potencia de OpenCV, echa un vistazo a *http://opencv.org/*.

La visión artificial tiene un uso de procesador y de memoria intensivo, por lo que, aunque SimpleCV y OpenCV funcionen con una Raspberry Pi más antigua, puede ser frustrantemente lento en comparación con las Raspberry Pi 3 o 2.

Para saber más

Para obtener información sobre OpenCV vea *http://opencv.org/*.

La página principal del proyecto SimpleCV es *http://simplecv.org*.

El primer punto en el que se usa SimpleCV es el Capítulo 8.5, donde encontrará detalles útiles para comenzar con SimpleCV.

8.3 Configurar una webcam USB para la visión artificial

Problema

Desea configurar una webcam USB para usarla en proyectos de visión artificial.

Solución

Utilice una webcam USB que sea compatible con Raspberry Pi (vea *http://elinux.org/RPi_USB_Webcams*). Elija una cámara de buena calidad y, si está trabajando en un proyecto en el que necesita tener la cámara cerca del sujeto, seleccione una que tenga enfoque manual. Para conseguir estar realmente cerca del sujeto puede ser útil un endoscopio USB barato.

Dependiendo de su proyecto de CV puede que desee configurar un área bien iluminada. La Figura 8.1 muestra una simple caja de luz hecha a partir de una caja de plástico translúcida iluminada, por los lados y por la parte superior, para proporcionar una iluminación uniforme. La webcam está conectada por un agujero en la parte superior de la caja. Esta disposición se utiliza en el Capítulo 8.5.

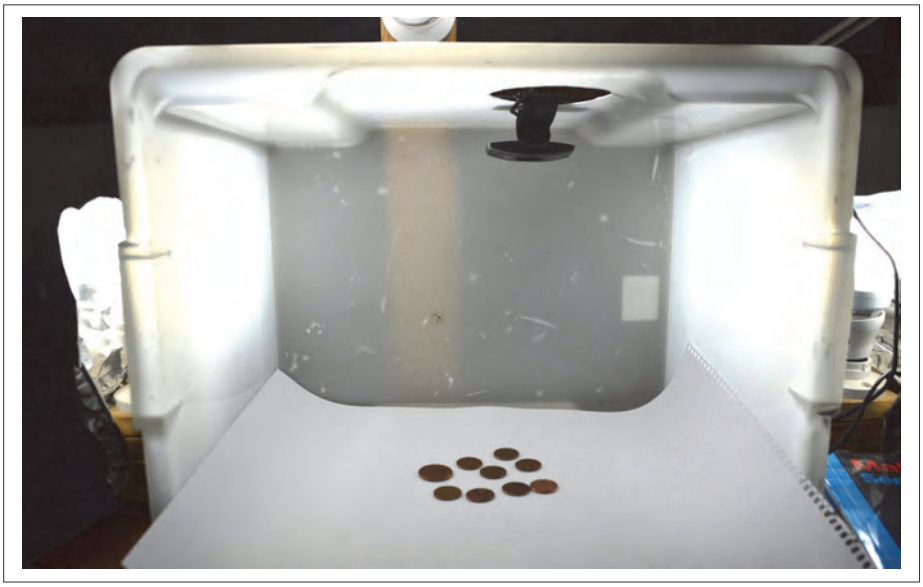


Figura 8.1. *Uso de una "tienda de campaña" casera para una iluminación uniforme.*

También puede comprar *tiendas de campaña* de luz diseñadas para la fotografía que funcionan bien.

Es posible que necesite un poco de prueba y error para obtener su sistema brillante y uniformemente iluminado. Las sombras pueden ser realmente problemáticas.

Observaciones

Puede probar su cámara USB desde la consola SimpleCV. Inicie SimpleCV y luego escriba los comandos que se muestran a continuación en negrita:

```
SimpleCV:1> c = Camera()
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
```

```
SimpleCV:2> i = c.getImage()
SimpleCV:3> i
SimpleCV:3: <SimpleCV.Image Object size:(640, 480), filename: (None),
at memory location: (0x2381af8)>
SimpleCV:4> i.show()
```

No se preocupe por los mensajes de *argumento no válido*.

Cuando emita el comando `i.show()` se abrirá una segunda ventana que mostrará la imagen recién capturada de la cámara.

Aunque puede usar el módulo de la cámara de Raspberry Pi (Capítulo 8.4) con SimpleCV, probablemente prefiera trabajar con una webcam de alta calidad.

Para saber más

Para utilizar un módulo de cámara de Raspberry Pi con SimpleCV vea el Capítulo 8.4.

8.4 Usar un módulo de cámara de Raspberry Pi para la visión artificial

Problema

Desea utilizar un módulo de cámara de Raspberry Pi que se conecte directamente a su Raspberry Pi con Simple CV.

Solución

El módulo de cámara de Raspberry Pi no aparece automáticamente como un dispositivo de cámara. La forma más fácil de hacer que el módulo funcione con SimpleCV es utilizar el módulo de Python `picamera` para capturar una imagen con la cámara.

El siguiente fragmento de código utilizará `picamera` para capturar una imagen, guardarla en un archivo temporal y cargarla como `Image` adecuada para usar con SimpleCV.

```
import picamera
from SimpleCV import *

def get_camera_image():
    with picamera.PiCamera() as camera:
        camera.capture('tmp.jpg')
    return Image('tmp.jpg')
```

En el Capítulo 8.5 el programa asume una webcam USB. Se proporciona una segunda versión del programa que utiliza la función anterior para el módulo de cámara de Raspberry Pi en el archivo *coin_count_pi_cam.py*.

Observaciones

Puede parecer ineficiente escribir un archivo cada vez que se captura una imagen, y de hecho lo es, pero cualquier procesamiento que aplique a la imagen con SimpleCV en Raspberry Pi probablemente tome más tiempo del necesario para cargar y guardar la imagen.

El módulo `picamera` le permite configurar varias funciones de la cámara. Puede controlar, de forma más útil, la resolución, la exposición automática y los ajustes de balance de blancos, lo que puede facilitarle la obtención de resultados con CV.

La resolución nativa de la cámara es 2592 × 1944, lo que hará que el procesamiento de imágenes sea muy lento, por lo que puede que desee ajustar la resolución a algo más cercano a 1024 × 768 y desactivar el balance de blancos automático. La función `get_camera_image` se puede modificar para incluir esta configuración como se muestra aquí:

```
import picamera
from SimpleCV import *

def get_camera_image():
    with picamera.PiCamera() as camera:
        camera.resolution = (1024, 768)
        camera.awb_mode = 'off'
        camera.capture('tmp.jpg')
    return Image('tmp.jpg')
```

Para saber más

Consulte el Capítulo 1.15 para obtener información sobre la instalación del módulo de cámara de Raspberry Pi.

Para obtener información sobre el módulo de Python `picamera` vea *http://picamera.readthe- docs.org/*.

Para utilizar una cámara USB con SimpleCV consulte el Capítulo 8.3.

8.5 Contar monedas

Problema

Desea usar la visión artificial para contar el número de monedas con su webcam.

Solución

Utilice SimpleCV y su función `findCircle` para proporcionar un recuento a tiempo real del número de monedas colocadas debajo de la webcam.

Este es un uso de CV en el que se necesita una buena iluminación y una cámara con posición fija. Utilicé la configuración mostrada en la Figura 8.1.

Antes de escribir un programa de Python que simplemente le diga el número de monedas que su Raspberry Pi puede ver, es necesario experimentar con la consola SimpleCV para obtener los parámetros correctos de reconocimiento de círculo.

Inicie SimpleCV con el comando `simplecv` y luego introduzca los siguientes comandos para iniciar la cámara, capturar una imagen y mostrarla en una ventana aparte.

```
SimpleCV:1> c = Camera()
SimpleCV:2> i = c.getImage()
SimpleCV:3> i.show()
```

Esto debería abrir una imagen de sus monedas como aparece en la Figura 8.2.

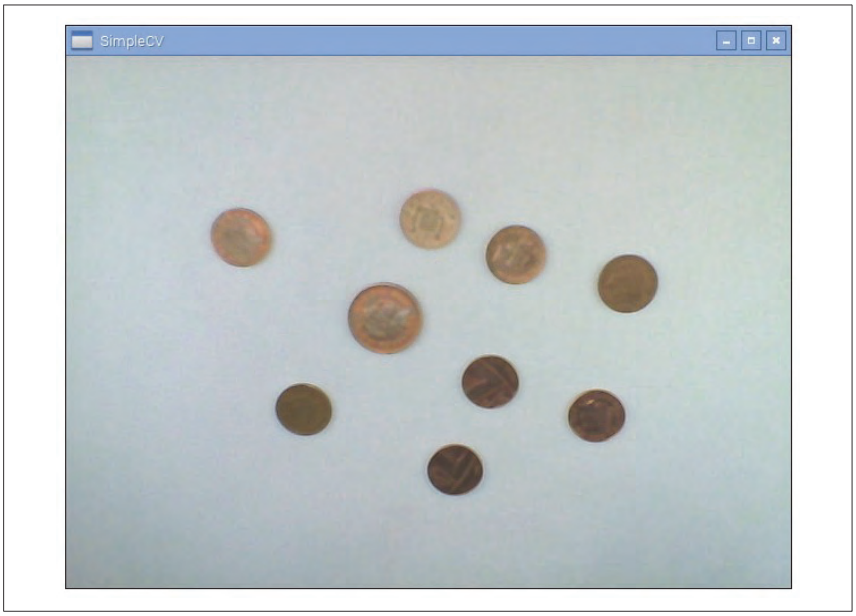


Figura 8.2. *Imagen básica de algunas monedas.*

La detección del círculo necesita la imagen que se va a invertir, o puede utilizar un fondo negro. Para invertir la imagen y luego mostrarla ejecute el siguiente comando:

```
SimpleCV:4> i2 = i.invert()
SimpleCV:5> i2.show()
```

Este comando hace una copia invertida de `i` como se ve en la Figura 8.3.

Figura 8.3. *Imagen invertida de algunas monedas.*

Su imagen ya está preparada, así que el siguiente paso es que SimpleCV busque círculos usando el comando `findCircle`. Este comando toma tres parámetros que necesitará configurar para evitar una identificación errónea. Los parámetros son:

`canny`

Este es el umbral para la detección de bordes. En términos de CV, un borde es la línea entre los cambios significativos en los colores del píxel de la imagen. El valor predeterminado para este parámetro es 100 y, si disminuye este valor, encontrará más bordes. Puede que no se encuentren más círculos si esos bordes adicionales corrompen las formas de un buen círculo. En el caso de una moneda, los bordes podrían ser la escritura o gráficos en esta.

`thresh`

Después de encontrar los bordes, la detección del círculo necesita decidir cuáles son lo suficientemente fuertes para representar los círculos. Si disminuye este valor, se obtendrán más círculos.

`distance`

Este parámetro establece el espacio requerido (en píxeles) entre círculos adyacentes.

Ejecute el siguiente comando para buscar algunos círculos:

```
SimpleCV:6> coins = i2.findCircle(canny=100, thresh=70, distance=15)
SimpleCV:7> coins
SimpleCV.Features.Detection.Circle at (237,297),
SimpleCV.Features.Detection.Circle at (307,323),
SimpleCV.Features.Detection.Circle at (373,305),
SimpleCV.Features.Detection.Circle at (305,261),
SimpleCV.Features.Detection.Circle at (385,253),
SimpleCV.Features.Detection.Circle at (243,231),
SimpleCV.Features.Detection.Circle at (307,383),
SimpleCV.Features.Detection.Circle at (407,371),
SimpleCV.Features.Detection.Circle at (235,373)]
```

Si no recibe ninguna moneda, trate de disminuir los parámetros `canny` y `thresh`. Si obtiene demasiados, aumente `thresh`. Podemos comprobar que SimpleCV realmente encuentra monedas superponiendo los círculos de las monedas a la imagen original usando el siguiente comando:

```
SimpleCV:8> coins.draw(width=4)
SimpleCV:9> coins.show()
```

Esto mostrará los círculos superpuestos sobre las monedas reales (Figura 8.4).

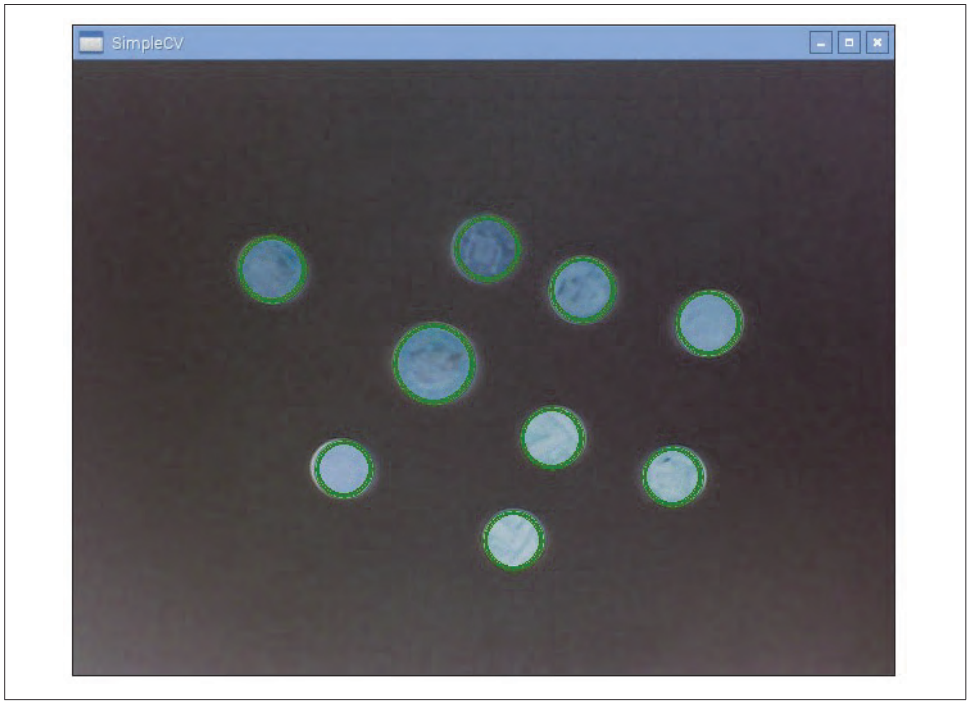


Figura 8.4. *¡Monedas encontradas!*

Trate de añadir y quitar monedas antes de capturar otra foto y de repetir los pasos anteriores para asegurarse de que es fiable. Puede ajustar los parámetros hasta obtener los resultados deseables.

Podemos agrupar los comandos que usamos en la consola SimpleCV en un programa de Python que pueda mostrar (con la rapidez de Raspberry Pi) el número de monedas detectadas. El programa se puede encontrar con las otras descargas del libro en el archivo *coin_count.py*.

```
from SimpleCV import *

c = Camera()

while True:
    i = c.getImage().invert()
    coins = i.findCircle(canny=100, thresh=70, distance=1)
    print(len(coins))
```

Después de importar la biblioteca SimpleCV, los comandos son los mismos que los que escribió en la consola. La única diferencia es que, en lugar de mostrar las monedas, la función `len` se utiliza para mostrar el recuento.

```
$ sudo python count_coins.py
9
9
9
10
10
```

Trate de añadir una moneda y de moverlas para ver cómo funciona el proyecto.

Observaciones

Después de la demora inicial mientras la biblioteca se carga y la cámara se configura, vi que podía obtener cerca de dos "recuentos" por segundo utilizando la Raspberry Pi B+. Con la Raspberry Pi 2 aumentó a cerca de cinco lecturas por segundo.

Aunque poner una máquina expendedora no sea algo que le interese, sería un proyecto interesante utilizar el diámetro de las monedas para identificar su valor monetario y sumar el valor de las monedas de encima de la mesa.

Puede acceder al diámetro usando el método `diameter` en una de las monedas, como aquí:

```
SimpleCV:10> coins[0].diameter()
SimpleCV:11> 60
```

Para saber más

Para obtener información sobre la instalación de SimpleCV vea el Capítulo 8.2.

Para obtener información sobre cómo configurar una cámara consulte el Capítulo 8.3.

8.6 Detección facial

Problema

Desea encontrar las coordenadas de las caras en una fotografía o imagen de webcam.

Solución

Utilice la detección de características *Haar-like* en SimpleCV para analizar una imagen y seleccionar las caras.

Si aún no lo ha hecho, instale SimpleCV (consulte el Capítulo 8.2). Abra la consola de SimpleCV y cargue una imagen que contenga caras. Encontrará un archivo adecuado en las descargas para el libro llamado *faces.jpg*. A continuación, ejecute los siguientes comandos:

```
SimpleCV:1> i=Image("faces.jpg")
SimpleCV:2> faces = i.findHaarFeatures('face.xml', min_neighbors=5)
SimpleCV:3> faces.draw(width=4)
SimpleCV:4> i.show()
```

Esto abrirá la ventana de visualización de imágenes con las caras marcadas con rectángulos, como se muestra en la Figura 8.5.

Observaciones

Además de interactuar con una cámara, también puede cargar archivos existentes en SimpleCV. En el ejemplo anterior la imagen *i* se carga desde el archivo *faces.jpg*. El método `findHaarFeatures` tiene un archivo obligatorio, que es el tipo de características que está buscando. Estas características se denominan funciones `haar` y son descritas en un archivo XML. SimpleCV viene precargado con algunos de estos archivos, pero también puede encontrar archivos `haar` más específicos en Internet.

El segundo parámetro utilizado en este ejemplo (`min_neighbors`) ajustará la función `haar`. Disminuir `min_neighbors` aumenta el número de falsos positivos. Estos a menudo tienen un elemento facial (boca, nariz y ojos).

Hay muchos elementos incorporados en `haar`. Puedes listarlos todos usando el comando:

```
SimpleCV:5> i.listHaarFeature()
SimpleCV:4> fullbody.xml', 'face4.xml', 'face.xml',
'upper_body.xml', 'right_ear.xml', 'eye.xml', 'lower_body.xml',
'two_eyes_small.xml', 'nose.xml', 'face2.xml', 'lefteye.xml',
'right_eye.xml', 'two_eyes_big.xml', 'face3.xml', 'mouth.xml',
'glasses.xml', 'profile.xml', 'left_ear.xml', 'left_eye2.xml',
'upper_body2.xml', 'right_eye2.xml', 'face_cv2.xml'
```

Como puede ver, todos están asociados con partes del cuerpo. Encontrar los elementos de `haar` tomará algunos segundos, incluso en la Raspberry Pi 3.

Figura 8.5. *Detección de caras.*

Para saber más

Para obtener más información sobre la instalación de SimpleCV vea el Capítulo 8.2.

Para obtener información sobre cómo configurar una cámara consulte el Capítulo 8.3.

Para una gran cantidad de interesantes archivos Haar eche un vistazo a: *https://github.com/Itseez/ opencv/tree/master/data/haarcascades*.

8.7 Detección de movimiento

Problema

Desea utilizar una cámara conectada a su Raspberry Pi para detectar cualquier cosa que se mueva en su campo de visión.

Solución

Use SimpleCV para detectar cambios entre fotogramas sucesivos desde la cámara.

El siguiente programa compara cada imagen capturada con la imagen anterior. Después detecta cualquier mancha (área de color similar) en la imagen de diferencia y, si hay más que en `MIN_BLOB_SIZE`, imprime un mensaje que indica que se detectó movimiento.

```
from SimpleCV import *

MIN_BLOB_SIZE = 1000

c = Camera()

old_image = c.getImage()

while True:
    new_image = c.getImage()
    diff = new_image - old_image
    blobs = diff.findBlobs(minsize=MIN_BLOB_SIZE)
    if blobs:
        print("Movement detected")
    old_image = new_image
```

Observaciones

Los fotogramas sucesivos de la imagen podrían parecerse a las Figuras 8.6 y 8.7. Cuando la primera imagen se reste de la primera, la imagen resultante se verá como la Figura 8.8. La detección de manchas producirá una imagen como la Figura 8.9.

Figura 8.6. *Marco de detección de movimiento 1.*

Figura 8.7. *Marco de detección de movimiento 2.*

Figura 8.8. *Detección de movimiento, la imagen de diferencia.*

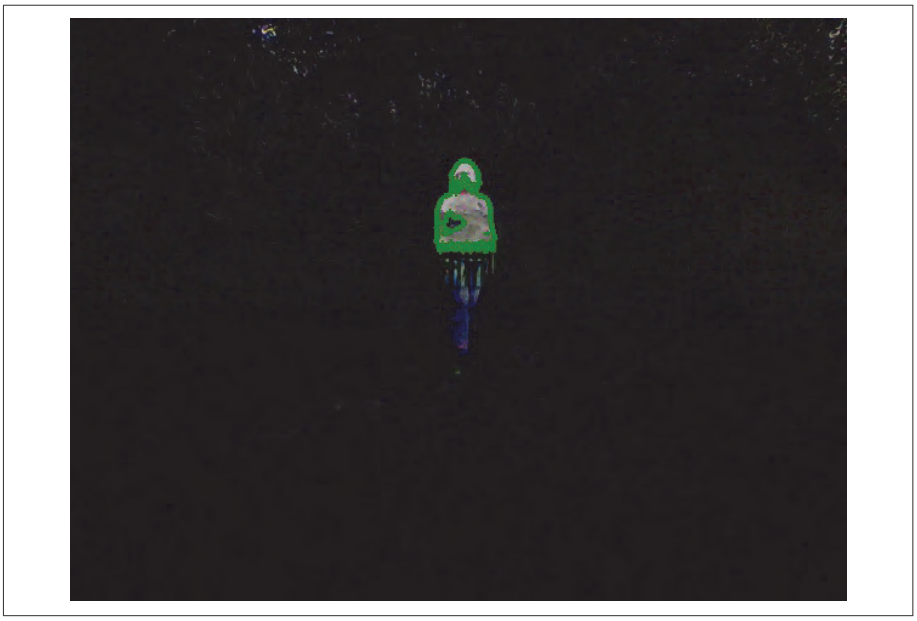


Figura 8.9. *Manchas de detección de movimiento.*

Utilizando una Raspberry Pi 2, el programa de detección de movimiento anterior procesará aproximadamente cinco fotogramas por segundo.

Para saber más

Para obtener más información sobre la instalación de SimpleCV consulte el Capítulo 8.2.

Para obtener más información sobre cómo configurar una cámara consulte el Capítulo 8.3.

Una forma alternativa de detectar movimiento es usar un sensor de infrarrojos pasivo (PIR); consulte el Capítulo 12.10.

8.8 Reconocimiento óptico de caracteres

Problema

Desea poder convertir una imagen que contenga texto en texto real.

Solución

Utilice el *software* de Reconocimiento Óptico de Caracteres (OCR) `tesseract` para extraer texto de la imagen.

Para instalar `tesseract` ejecute los siguientes comandos:

```
$ sudo apt-get install python-distutils-extra tesseract-ocr tesseract-ocr-eng
libopencv-dev libtesseract-dev libleptonica-dev python-all-dev swig
libcv-dev python-opencv python-numpy python-setuptools
build-essential subversion
$ sudo apt-get install tesseract-ocr-eng tesseract-ocr-dev libleptonica-dev
python-all-dev swig libcv-dev
$ sudo svn checkout
http://python-tesseract.googlecode.com/svn/python-tesseract-0.7.4/
$ cd python-tesseract-0.7.4
$ sudo python setup.py build
$ sudo python setup.py install
```

Para probar `tesseract`, necesitará un archivo de imagen que contenga texto. Encontrará uno llamado *ocr_example.png* en las descargas del libro.

Para convertir la imagen en texto ejecute el comando:

```
$ tesseract ocr_example.png found
Tesseract Open Source OCR Engine v3.02 with Leptonica
$ more found.txt
The  quick  brown  fox  jumped
over the lazy dogs back.
$
```

Observaciones

La biblioteca `tesseract` funcionará con la mayoría de los tipos de imágenes, incluidos los archivos PDF, PNG y JPG.

Para saber más

Para obtener más información sobre la biblioteca `tesseract`, consulte *https://github.com/tesseract-ocr/ tesseract/wiki/TrainingTesseract*.

CAPÍTULO 9

Fundamentos de *hardware*

9.1 Introducción

Este capítulo contiene algunas claves básicas para configurar y usar el conector de entrada y salida de uso general (GPIO) de Raspberry Pi. Este conector le permite conectar todo tipo de electrónica interesante a su Raspberry Pi.

9.2 Conocer el camino al conector GPIO

Asegúrese de revisar el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Necesita conectar electrónica al conector GPIO, pero primero necesita saber más sobre qué hacen los pines.

Solución

En realidad, ha habido tres versiones del conector GPIO de Raspberry Pi. Dos diseños de 26 pines para la Raspberry Pi original y uno de 40 pines que va con los modelos "+" de Raspberry Pi.

La Figura 9.1 muestra las patillas del GPIO para las revisiones 1 y 2 del modelo B original de Raspberry Pi. La manera rápida de distinguir las tarjetas es saber que, si es una de las primeras tarjetas de revisión 1, tendrá un zócalo de audio negro. Las tarjetas de revisión 2, en cambio, tienen un zócalo de audio azul.

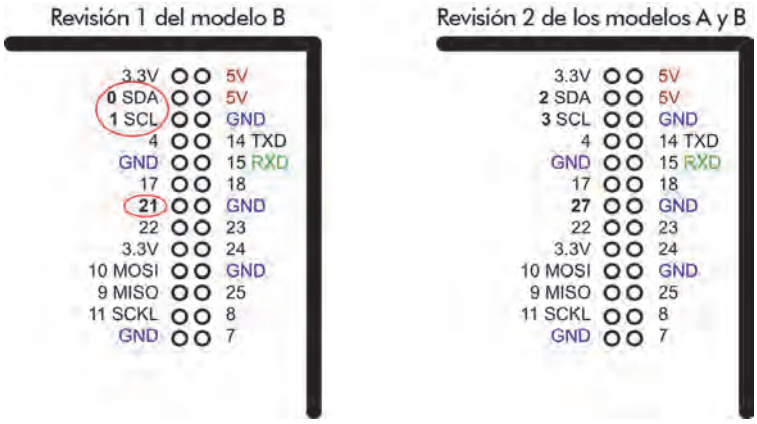


Figura 9.1. *Patillaje GPIO (Modelos de 26 pines).*

Hubo tres cambios en el conector GPIO entre la revisión 1 y la revisión 2. Estos se resaltan en la Figura 9.1. En primer lugar, se intercambió el puerto I2C. Los dos pines SDA y SCL siguen siendo SDA y SCL, pero utilizan una interfaz I2C interna diferente. Esto significa que, si está utilizando los pines como GPIO en lugar de I2C, se referirá a ellos como 2 y 3 en una tarjeta de revisión 2. Además, GPIO 21 fue reemplazado por GPIO 27 en la revisión 2.

En la parte superior del conector hay fuentes de alimentación de 3,3 V y 5 V. El GPIO utiliza 3,3 V para todas las entradas y salidas. Cualquier pin con un número cerca de él puede actuar como pin GPIO. Aquellos que tienen otro nombre después de ellos también tienen algún otro propósito especial, por lo que 14 TXD y 15 RXD son los pines de transmisión y recepción de la interfaz serie. SDA y SCL forman la interfaz I2C, y MOSI, MISO y SCKL la interfaz SPI.

La Figura 9.2 muestra el diseño actual de 40 pines, que es el mismo para todos los modelos GPIO de las Raspberry Pi de 40 pines.

Los 26 pines de la parte superior son los mismos que los 26 pines de la revisión 2 del modelo B original de Raspberry Pi. Esto permite que los modelos de 40 pines de Raspberry Pi utilicen el *hardware* y el *software* diseñados para los anteriores diseños de 26 pines de Raspberry Pi. Los pines adicionales del conector de 40 pines se componen de tres conexiones GND adicionales útiles y 9 pines GPIO. Los pines ID_SD y ID_SC están diseñados para ser utilizados en la comunicación con un chip de memoria en serie especial, que puede incluirse en las tarjetas de interfaz que conforman el Hardware At Top (HAT) estándar y permite que Raspberry Pi identifique la tarjeta (véase el apartado «Observaciones»).

Figura 9.2. *Las clavijas GPIO (modelos de 40 pines).*

Observaciones

Ver qué pin es cuál en Raspberry Pi es bastante propenso a errores si confía en contar conectores para encontrar el pin que necesita. La mejor manera de encontrar el pin correcto es usar una plantilla GPIO como la hoja Raspberry mostrada en la Figura 9.3.

Esta plantilla de papel se ajusta sobre los pines GPIO y le dice cuál es cuál. Otras plantillas GPIO incluyen la tarjeta de referencia GPIO Pi.

El hardware *At Top* estándar es un estándar de interfaz que puede utilizar con las Raspberry Pi 2, B+ y A+. Este estándar no le impide utilizar los pines GPIO directamente; sin embargo, las tarjetas de interfaz que se ajustan al estándar pueden denominarse HAT y difieren de las tarjetas de interfaz regulares de Raspberry Pi, ya que un HAT debe contener un pequeño chip de memoria de solo lectura, programable y borrable eléctricamente (EEPROM) que se usa para identificar el HAT de modo que, en última instancia, Raspberry Pi pueda autoinstalar el *software* necesario. En el momento de escribir este artículo los HAT no han alcanzado ese nivel de sofisticación, pero la idea es buena. Los pines ID_SD y ID_SC se utilizan para comunicarse con una EEPROM HAT.

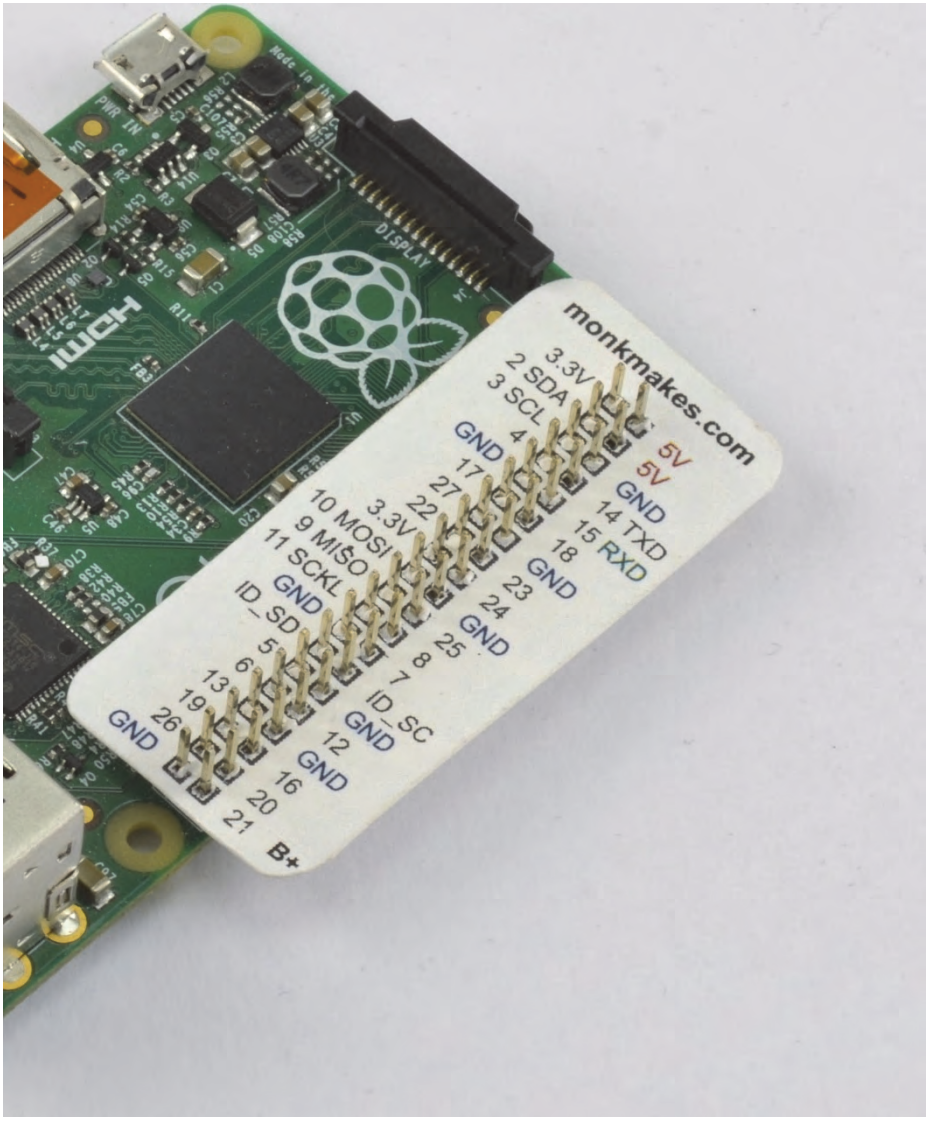


Figura 9.3. *La hoja Raspberry.*

Para saber más

El conector GPIO de Raspberry Pi solo tiene entradas y salidas digitales; no tiene entradas analógicas como sí pasa en tarjetas similares. Puede evitar este problema utilizando un chip ADC separado (Capítulo 13.6) o utilizando sensores resistivos (Capítulo 13.2).

Para ver un ejemplo de HAT consulte el Sense HAT descrito en el Capítulo 9.17.

9.3 Mantener su Raspberry Pi segura cuando utilice el conector GPIO

Problema

Desea conectar electrónica externa a su Raspberry Pi, pero no desea dañarla o romperla de forma accidental.

Solución

Obedezca estas sencillas reglas para reducir el riesgo de dañar su Raspberry Pi al usar el conector GPIO:

- No ponga más de 3,3 V en ningún pin GPIO que se utilice como entrada.
- No extraiga más de 16 mA por salida y mantenga el total de todas las salidas por debajo de 50 mA para una Raspberry Pi vieja de 26 pines, y por debajo de 100 mA en una Raspberry Pi de 40 pines.
- Cuando se utilizan ledes, 3 mA es suficiente para encender un led rojo con una resistencia de 470 Ω.
- No golpee el conector GPIO con un destornillador ni con ningún objeto metálico cuando Pi esté encendido.
- No encienda Pi con más de 5 V.
- No extraiga más de un total de 250 mA de los pines de suministro de 5 V.

Observaciones

No hay duda al respecto, Raspberry Pi es un poco frágil cuando se le agrega electrónica externa. Los modelos más nuevos de Raspberry Pi son un poco más robustos, pero todavía bastante fáciles de romper. Tenga cuidado y compruebe lo que ha conectado *antes* de encender su Raspberry Pi o correrá el riesgo de tener que reemplazarla.

Para saber más

Encontrará más información en el sitio de las capacidades de las salidas GPIO de Raspberry Pi.

9.4 Configurar I2C

Problema

Tiene un dispositivo I2C que quiere usar con su Raspberry Pi, pero no sabe cómo hacerlo.

Solución

En las últimas versiones de Raspbian la habilitación de I2C se encuentra en la herramienta de Configuración de Raspberry Pi que encontrará en el menú principal (Figura 9.4). Solo marque la casilla para I2C y haga clic en OK. Se le pedirá que reinicie.

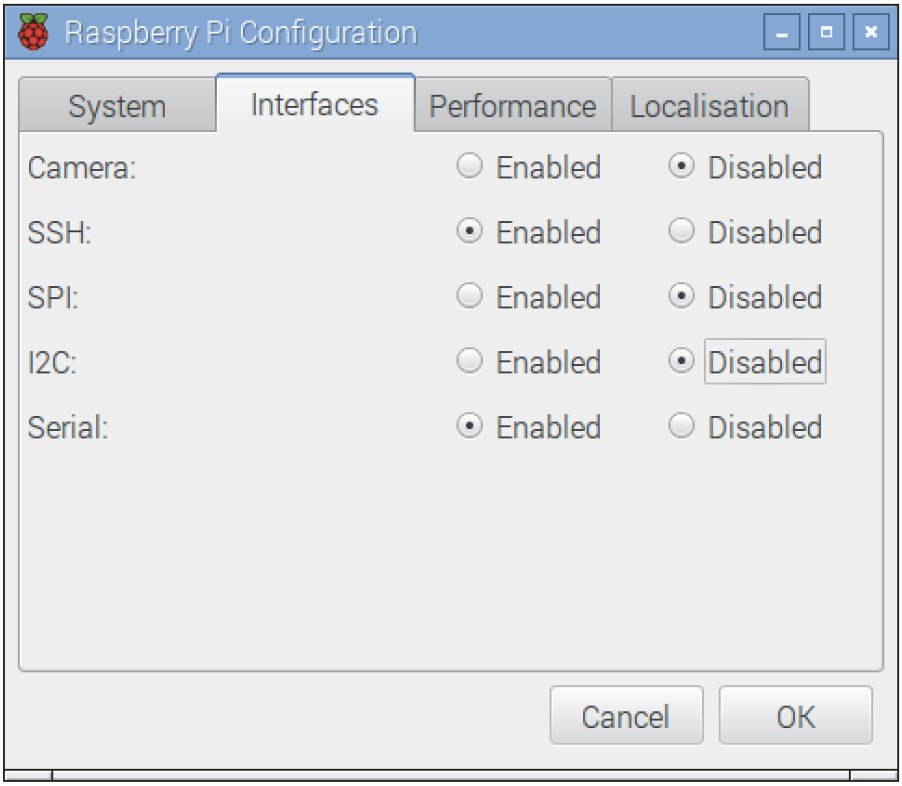


Figura 9.4. *Activar I2C mediante la herramienta de configuración.*

En versiones anteriores de Raspbian la herramienta `raspi-config` hace la misma función.

Inicie `raspi-config` usando el siguiente comando:

```
$ sudo raspi-config
```

A continuación, seleccione Advanced en el menú y desplácese a I2C (Figura 9-5).

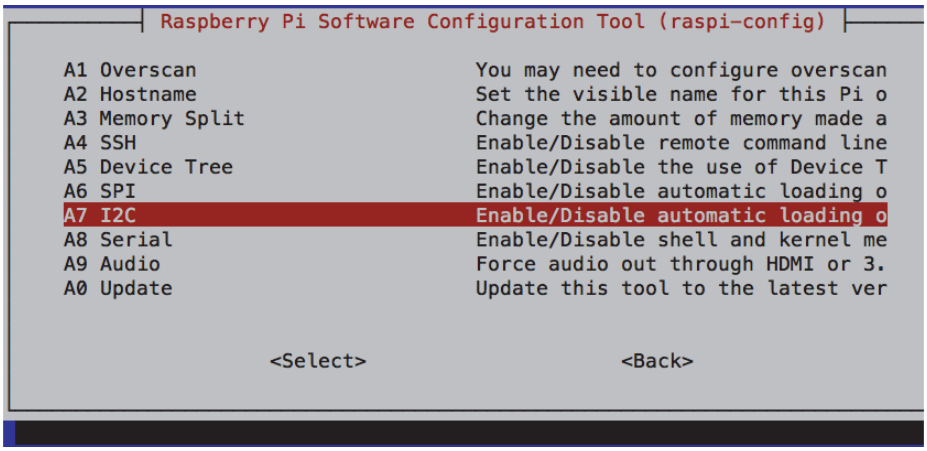


Figura 9.5. *Activar I2C mediante raspi-config.*

Después se le preguntará si quiere activar la interfaz I2C ARM *(Would you like the ARM I2C interface to be enabled*?), a lo que deberá decir «Yes». También se le preguntará si desea cargar el módulo I2C al inicio, a lo que también deberá decir «Yes».

En este punto probablemente también desee instalar la biblioteca de Python I2C mediante el comando:

```
$ sudo apt-get install python-smbus
```

A continuación, tendrá que reiniciar Raspberry Pi para que los cambios funcionen.

Observaciones

Utilizar módulos I2C es en realidad una muy buena forma de interactuar con Pi. Reduce el número de cables que necesita para conectarlo todo (a solo cuatro), y hay algunos módulos I2C realmente geniales.

Sin embargo, no olvide calcular el total de la corriente utilizada por los módulos I2C y asegúrese de que no exceda los especificados en el Capítulo 9.3.

La Figura 9.6 muestra una selección de módulos I2C disponibles de Adafruit. Otros proveedores, como SparkFun, también tienen dispositivos I2C. De izquierda a derecha está la pantalla de matriz de led, una de cuatro dígitos y siete segmentos, un controlador PWM/servo de 16 canales y un módulo de reloj a tiempo real.

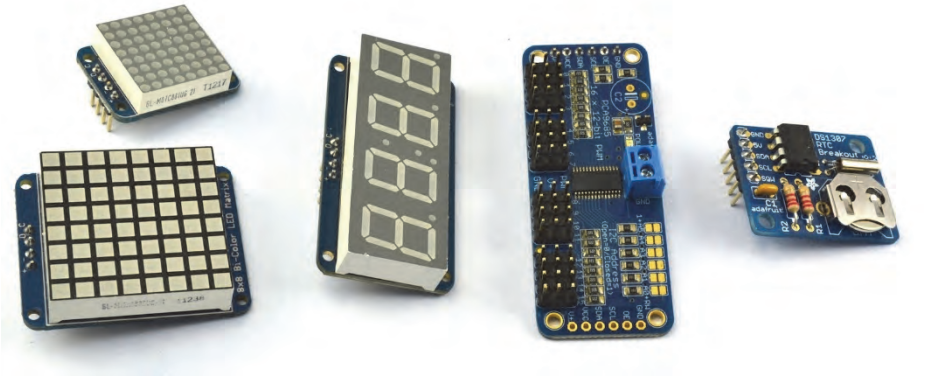


Figura 9.6. *Módulos I2C.*

Otros módulos I2C incluyen transmisores de radio FM, telémetros ultrasónicos, pantallas OLED y varios tipos de sensores.

Para saber más

Vea algunos de los capítulos sobre I2C en este libro, incluyendo los Capítulos 11.4, 14.2, 14.3 y 14.5.

9.5 Usar herramientas I2C

Problema

Tiene un dispositivo I2C conectado a su Raspberry Pi y desea comprobar que está conectado y encontrar su dirección I2C.

Solución

Instalar y utilizar `i2c-tools`.

En distribuciones más recientes, es posible que `i2c-tools` ya esté instalado.

Desde una ventana de terminal en su Pi, escriba los siguientes comandos para buscar e instalar `i2c-tools`:

```
$ sudo apt-get install i2c-tools
```

Conecte su dispositivo I2C a Pi y ejecute el comando:

```
$ sudo i2cdetect -y 1
```

Tenga en cuenta que, si está utilizando una tarjeta de la revisión 1 muy antigua de Raspberry Pi, debe cambiar 1 a 0 en la línea de código anterior.

Si I2C está disponible, verá una salida como la que se muestra en la Figura 9.7. Esto muestra que dos direcciones I2C están en uso (0x68 y 0x70).

Figura 9.7. *Herramientas i2C.*

Observaciones

i2cDetect es una herramienta de diagnóstico y vale la pena ejecutarla la primera vez que utilice un nuevo dispositivo I2C.

Para saber más

Consulte algunos de los capítulos sobre I2C en este libro, incluyendo los Capítulos 11.4, 14.2, 14.3 y 14.5. Para más información sobre la instalación con `apt-get` revise el Capítulo 3.17.

9.6 Configurar SPI

Problema

Tiene un bus de Interfaz de Periféricos Serie (SPI) que desea utilizar con su Raspberry Pi.

Solución

De forma predeterminada, Raspbian no está configurado para que la interfaz SPI de Raspberry Pi esté habilitada. Para habilitarla, el procedimiento es casi el mismo que en el Capítulo 9.4. Utilice la herramienta de configuración de Raspberry Pi que encontrará en el menú principal o, en versiones anteriores de Raspbian, utilice `raspi-config` usando el comando:

```
$ sudo raspi-config
```

Después seleccione Advanced, seguido de SPI y, a continuación, Yes antes de reiniciar su Raspberry Pi. Después de reiniciar, SPI estará disponible.

Observaciones

SPI permite la transferencia en serie de datos entre Raspberry Pi y los dispositivos periféricos, como el convertidor analógico-digital (ADC) y los chips de expansión de puertos, entre otros dispositivos.

Puede encontrar algunos ejemplos de interfaz con SPI que no usan la interfaz SPI, sino que utilizan un método llamado *bit banging*, donde la biblioteca `RPi.GPIO` se utiliza para interactuar con los cuatro pines GPIO utilizados por la interfaz SPI.

Para saber más

Utilizamos un chip convertidor analógico-digital SPI en el Capítulo 13.6.

9.7 Instalar PySerial para acceder al puerto serie desde Python

Problema

Desea utilizar el puerto serie (pines Rx y Tx) en Raspberry Pi usando Python.

Solución

Instale la biblioteca PySerial:

```
$ sudo apt-get install python-serial
```

Observaciones

La biblioteca es bastante fácil de usar. La creación de una conexión utiliza la siguiente sintaxis:

```
ser = serial.Serial(DEVICE, BAUD)
```

donde `DEVICE` es el dispositivo para el puerto serie (`/dev/ttyAMA0`) y `BAUD` es la velocidad en baudios como un número, no una cadena. Por ejemplo:

```
ser = serial.Serial('/dev/ttyAMA0', 9600)
```

Una vez que se establece una conexión puede enviar datos como series como esto:

```
ser.write('some text')
```

Escuchar una respuesta normalmente implica un bucle que lee o imprime, como se ilustra en este ejemplo:

```
while True:
    print(ser.read())
```

Para saber más

Tendrá que utilizar esta técnica en los capítulos que conecten el *hardware* al puerto serie, como el Capítulo 12.11.

9.8 Instalar Minicom para probar el puerto serie

Problema

Desea enviar y recibir comandos serie desde una sesión de terminal.

Solución

Instalar Minicom:

```
$ sudo apt-get install minicom
```

Una vez instalado Minicom puede iniciar una sesión de comunicaciones en serie con un dispositivo serie conectado a los pines RXD y TXD del conector GPIO mediante este comando:

```
$ minicom -b 9600 -o -D /dev/ttyAMA0
```

El parámetro después de `-b` es la velocidad en baudios y después de `-D`, el puerto serie. Recuerde utilizar la misma velocidad en baudios que la del dispositivo con el que se está comunicando.

Esto iniciará una sesión de Minicom. Seguramente, una de las primeras cosas que quiera hacer es activar el *eco local* para que pueda ver el comando que está escribiendo. Para ello, pulse Ctrl-A y después Z. Verá la lista de comandos mostrada en la Figura 9.8. Pulse E para activar el eco local.

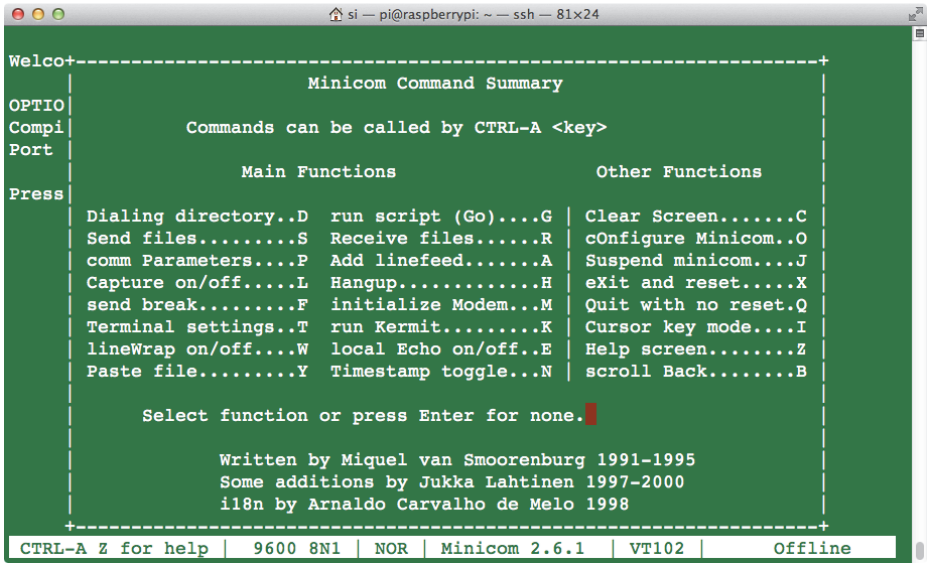


Figura 9.8. *Comandos Minicom.*

Ahora, cualquier cosa que usted escriba será enviada al dispositivo serie y todos los mensajes que vengan del dispositivo también aparecerán.

Observaciones

Minicom es una gran herramienta para comprobar los mensajes que se reciben de un dispositivo serie o para cerciorarse de que funciona.

Para saber más

Consulte la documentación de Minicom.

Si desea escribir un programa Python para manejar las comunicaciones en serie, necesitará la biblioteca *serial* de Python (Capítulo 9.7).

9.9 Usar una placa de pruebas con cables puente

Problema

Quiere hacer algunos prototipos electrónicos con su Raspberry Pi y una placa de pruebas sin soldar.

Solución

Utilice cables puente macho-hembra y una plantilla de pines GPIO como la hoja de Raspberry (Figura 9.9).

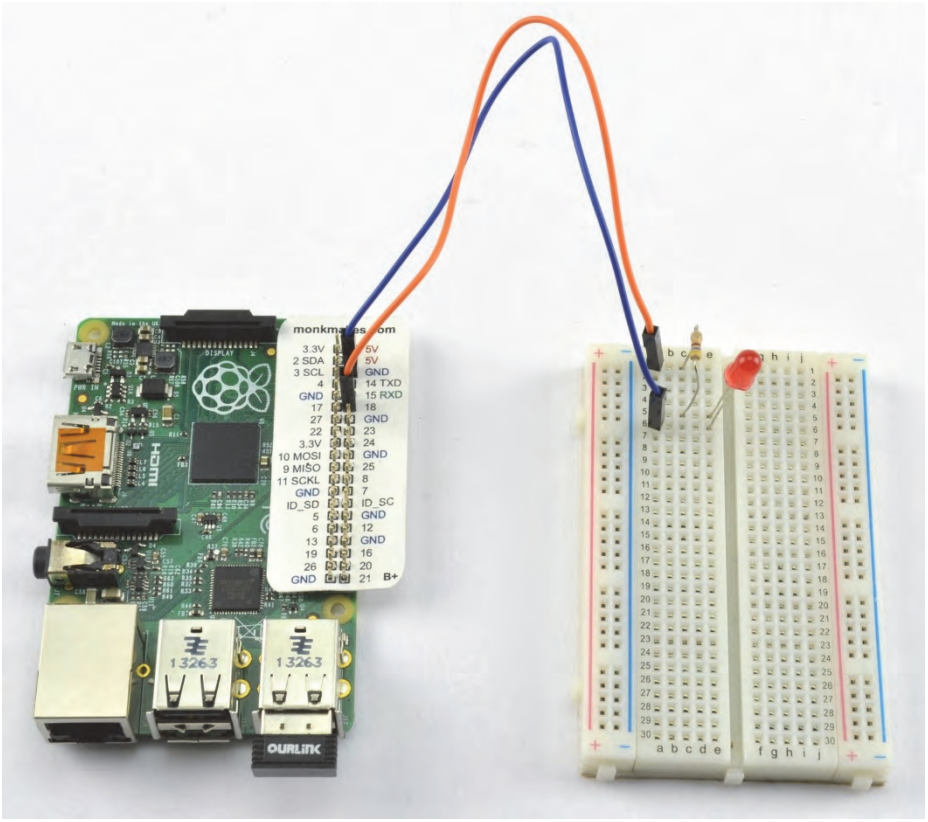


Figura 9.9. *Conectar Pi a la placa de pruebas usando cables puente macho-hembra.*

Observaciones

No siempre es fácil identificar los pines que quiere en una tarjeta Raspberry Pi. Puede simplificarlo imprimiendo una plantilla, como la hoja de Raspberry, para ponerla sobre las clavijas.

También es útil tener una selección de cables puente macho-macho para hacer conexiones desde una parte de la placa de pruebas a otra.

Los cables puente hembra-hembra son útiles para conectar módulos con pines de cabezal ancho directamente a Raspberry Pi cuando ningún otro componente pueda justificar el uso de la placa de pruebas.

Una buena manera de conseguir una placa de pruebas, la hoja de Raspberry y un conjunto de cables puente es comprando un kit básico basado en una placa de pruebas, como el kit básico electrónico para Raspberry Pi desde MonkMakes (consulte el Apéndice A).

Para saber más

Este enfoque funciona bien cuando hay un número pequeño de conexiones. Cuando necesite más será mejor un Pi Cobbler (consulte el Capítulo 9.10).

9.10 Usar una placa de pruebas con Pi Cobbler

Problema

Quiere hacer algunos prototipos electrónicos con su Raspberry Pi y una placa de pruebas sin soldar.

Solución

Utilice Adafruit Pi Cobbler, un dispositivo que consiste en una pequeña placa de circuito impreso (PCB) con pines que están diseñados para encajar en una placa de soldadura como un chip de placa doble en línea (DIL). Los pines están etiquetados y el PCB tiene una toma. El cable de cinta se utiliza para conectar Pi Cobbler a Raspberry Pi (vea la Figura 9.10).

El Pi Cobbler que se muestra en la Figura 9.10 es la versión de 26 pines del producto. También hay una versión de 40 pines diseñada para la nueva Raspberry Pi, pero esto deja poco espacio en la placa de pruebas para otros componentes. Por lo que, si tiene suficientes pines GPIO para su proyecto con los 26 pines superiores del conector GPIO, será mejor utilizar la versión de 26 pines de Pi Cobbler, incluso con una Raspberry Pi de 40.

Observaciones

Una gran ventaja de Cobbler es que puede montar los componentes en la placa de pruebas y luego conectar el cable de cinta cuando esté listo.

Asegúrese de que el borde rojo del cable de cinta vaya hacia el borde de la tarjeta SD de Raspberry Pi. El cable solo encaja en el zócalo del Pi Cobbler de manera correcta.

Una vez haya construido su prototipo de placa de pruebas puede decidir hacer una versión soldada del mismo. Una manera es usar una placa Adafruit Perma-Proto como la que se muestra en la Figura 9-11.

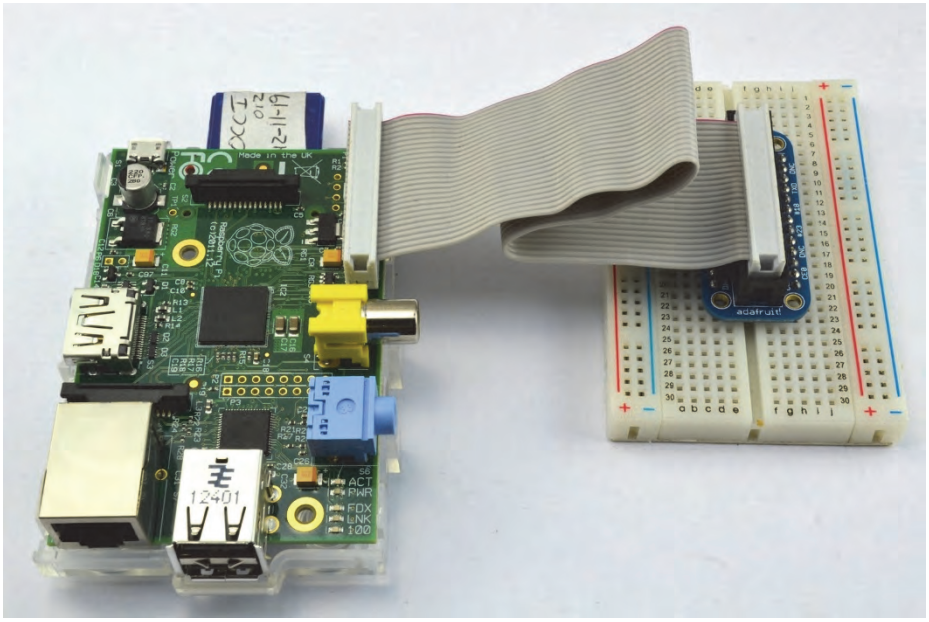


Figura 9.10. *Conectar Pi a una placa de pruebas mediante Pi Cobbler.*

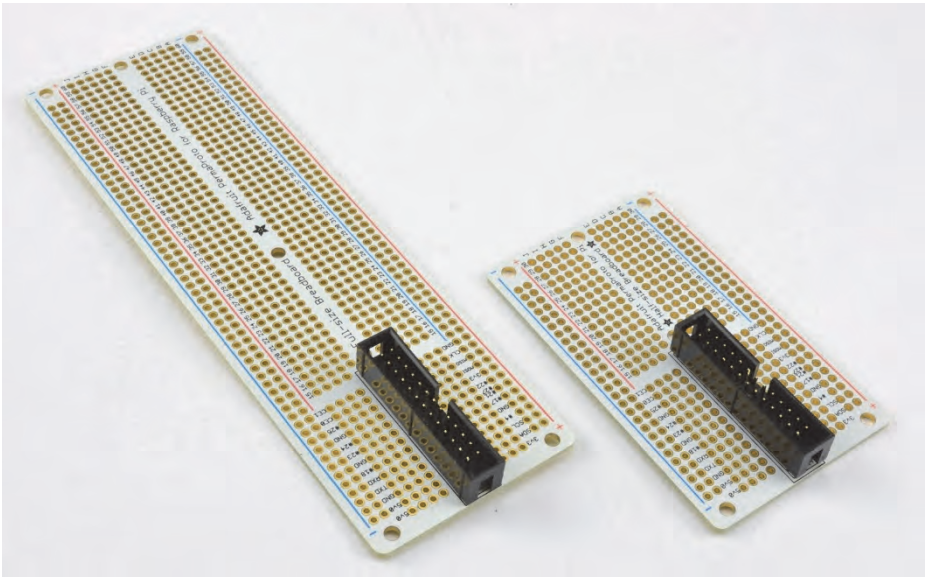


Figura 9.11. *Placas Adafruit Perma-Proto.*

Estas placas son placas de circuido impreso (PCB) que tienen la misma disposición de vías y agujeros que las placas de pruebas. Esto le permite transferir su diseño de placa de pruebas a las placas Perma-Proto sin tener que rediseñarla. La placa viene completa con un zócalo que acepta el cable y el enchufe del Cobbler.

Para saber más

Si quiere hacer una placa que se ajuste directamente a Raspberry Pi, consulte el Capítulo 9.20 y 9.21.

9.11 Usar una Raspberry Squid

Problema

Desea conectar un led RGB a su Raspberry Pi sin tener que construir nada en su placa de pruebas.

Solución

Use una Raspberry Squid (Figura 9.12).

Raspberry Squid es un led RGB con los resistores en serie incorporados y los cables principales hembra, de modo que puedan ser conectados directamente a los pines GPIO de la Raspberry Pi. El Squid tiene los cables codificados por colores. El negro va a uno de los pines GPIO GND y los cables rojo, verde y azul van a los pines GPIO utilizados para los canales rojo, verde y azul. Las salidas de color rojo, verde y azul pueden ser salidas digitales o PWM (Capítulo 10.4), que permiten mezclar diferentes colores.

Las instrucciones para hacer tu propia Squid se pueden encontrar en *https://github.com/simon-monk/squid*, pero también puedes comprar una Squid ya hecha en *http://monkmakes.com*.

La Squid tiene una biblioteca Python que se instala usando los siguientes comandos:

```
$ git clone https://github.com/simonmonk/squid.git
$ cd squid
$ sudo python setup.py install
```

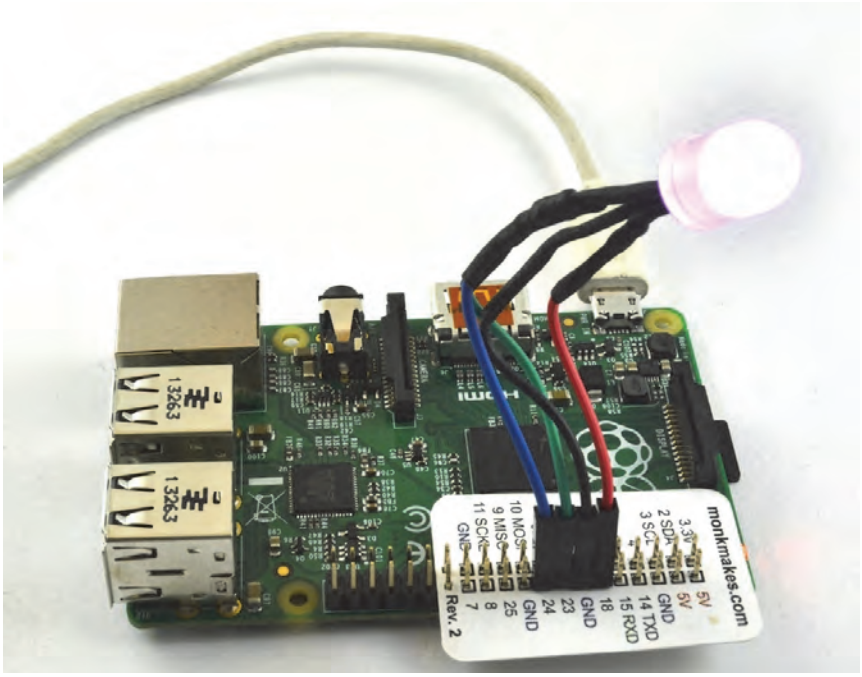


Figura 9.12. *Una Raspberry Squid.*

Dentro de la carpeta de ejemplo encontrará el programa *test_led.py* que le indicará prácticamente todo lo que necesita saber sobre el uso de Raspberry Squid.

```
from squid import *
import time

rgb = Squid(18, 23, 24)
rgb.set_color(RED)
time.sleep(2)
rgb.set_color(GREEN)
time.sleep(2)
rgb.set_color(BLUE)
time.sleep(2)
rgb.set_color(WHITE)
time.sleep(2)
rgb.set_color(WHITE, 300)
time.sleep(2)
```

Una vez importada la biblioteca `squid`, puede crear un nuevo objeto Squid suministrando los tres pines que se utilizarán para los canales rojo, verde y azul (en este caso, 18, 23, y 24). A continuación, puede configurar el color utilizando en el método `set_color`, que toma el nombre de una constante de color (RED, GREEN, WHITE, etc.) o una tupla de tres valores entre 0 y 100 para los canales rojo, verde y azul. El segundo parámetro opcional para `set_color` es el brillo general.

Después de ajustar el color, `time.sleep(2)` se utiliza para crear un retraso de dos segundos antes del siguiente cambio de color.

Observaciones

No es necesario utilizar los tres canales de color de una Squid, y puede ser muy práctico para comprobar que un pin GPIO se está encendiendo y apagando como cuando espera antes de conectar otros aparatos electrónicos a él.

Para saber más

Para obtener más información sobre la biblioteca Squid consulte el siguiente enlace:

https://github.com/simonmonk/squid.

Para más información sobre el botón Squid revise el Capítulo 9.12.

El Capítulo 10.11 es un ejemplo de proyecto que controla un led RGB (basado en una Squid o en una placa de pruebas).

9.12 Usar el botón Raspberry Squid

Problema

Desea conectar un interruptor a su Raspberry Pi sin tener que montar nada en la placa de pruebas.

Solución

Use un botón Squid.

El botón Squid (Figura 9.13) es un pulsador con cables hembra conectados a los contactos para que pueda enchufarlo directamente al conector GPIO de Raspberry Pi. El botón Squid también incluye un resistor que limita la corriente que fluiría si el botón Squid fuera conectado accidentalmente a una salida digital en lugar de a una entrada digital.

Figura 9.13. *Botón Squid.*

El botón Squid es un complemento natural del led RGB Squid y comparte la misma biblioteca Python (consulte el Capítulo 9.11 para instalar esta biblioteca).

El botón Squid podría utilizarse directamente con la biblioteca `RPi.GPIO`, pero la librería `squid` proporciona un *debouncing* (consulte *https://github.com/simonmonk/squid*) para evitar presionar de manera múltiple y accidentalmente el botón. El archivo *test_button.py* en la carpeta de ejemplos de la librería `squid` ilustra el uso del botón.

```
from button import *
import time

b = Button(7)

while True:
    if b.is_pressed():
        print(time.time())
```

El número (en este caso, 7) indica el pin GPIO al que está conectado el botón. El otro pin está conectado a GND.

Observaciones

El botón Squid es útil para probar proyectos que utilizan una entrada digital, pero, dado que el botón es adecuado para montar el panel, también puede montarlo en proyectos más permanentes.

Para saber más

Para obtener más información sobre la biblioteca `squid` consulte *https://github.com/simonmonk/squid*.

Para obtener más información sobre el uso de pulsadores consulte los Capítulos 12.2, 12.3, 12.4, 12.5, 12.6 y 12.7.

Para obtener información sobre el led RGB Squid revise el Capítulo 9.11.

9.13 Convertir señales de 5 V a 3,3 V con dos resistores

Problema

Raspberry Pi funciona a 3,3 V, pero quiere saber cómo conectar la salida de 5 V de un módulo a un pin GPIO sin dañarlo.

Solución

Utilice un par de resistores como divisor de potencias para reducir el voltaje de salida. La Figura 9.14 muestra cómo se puede utilizar la conexión en serie de 5 V de un Arduino a Raspberry Pi.

Para hacerlo necesitará:

- Un resistor de 270 Ω (vea "Resistores y Condensadores" en la página 474 en el Apéndice A)
- Un resistor de 470 Ω (vea "Resistores y Condensadores" en la página 474 en el Apéndice A)

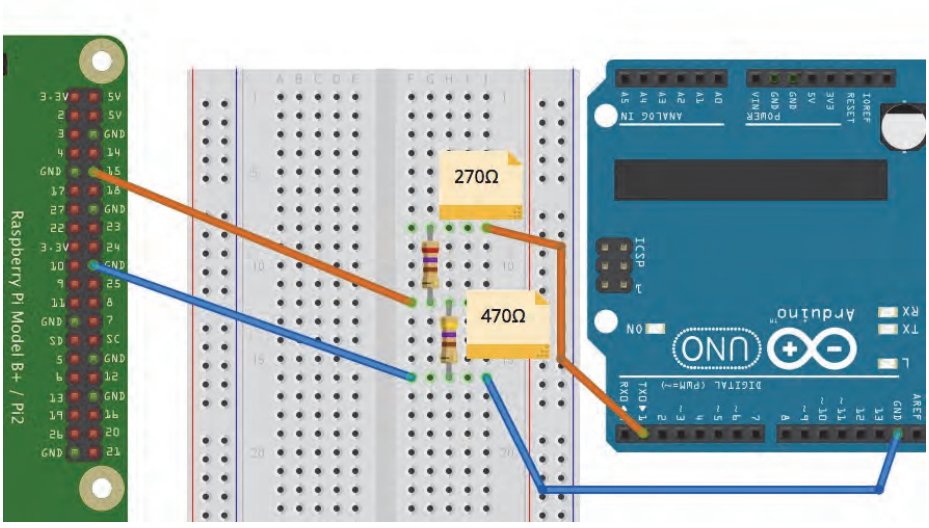


Figura 9.14. *Uso de resistores para convertir una señal de 5 V a 3,3 V.*

La señal TXD de Pi es una salida de 3,3 V. Se puede conectar directamente a una entrada de 5 V en Arduino sin problemas. El módulo Arduino reconocerá como alta cualquier cosa por encima de 2,5 V.

El problema surge cuando se tiene que conectar la salida de 5 V del módulo Arduino al pin RXD de Pi. Este *no debe* conectarse directamente a la entrada RXD. La señal de 5 V puede dañar a su Pi. En su lugar, se usan los dos resistores mostrados en la Figura 9.14.

Observaciones

Los resistores usados aquí crearán una corriente de 6 mA. Dado que Pi utiliza 500 mA, no se verá afectado, notablemente, el consumo actual de Pi.

Si desea minimizar la corriente utilizada por el divisor de potencia, utilice resistores de mayor valor, proporcionalmente a escala, por ejemplo, 27 kΩ y 47 kΩ, que extraerán unos miseros 60 μA.

Para saber más

Para obtener más información sobre cómo conectar Raspberry Pi a Arduino consulte el Capítulo 16.

Si tiene varias señales para convertir entre 3,3 V y 5 V, probablemente sea mejor utilizar un módulo conversor multicanal de niveles. Consulte el Capítulo 9.14.

9.14 Convertir señales de 5 V a 3,3 V con un módulo convertidor de nivel

Problema

Raspberry Pi funciona a 3,3 V. Desea conectar un número de pines digitales de 5 V a pines GPIO en Pi sin dañarlos.

Solución

Utilice un módulo conversor de nivel bidireccional, como los mostrados en la Figura 9.15.

Estos módulos son muy fáciles de usar. Un lado tiene la fuente de alimentación a un voltaje y un número de canales que pueden ser entradas o salidas a ese voltaje. Las clavijas en el otro lado del módulo tienen un pin de potencia en el segundo voltaje y todas las entradas y salidas de ese lado se convierten, automáticamente, al nivel de voltaje asignado para ese lado.

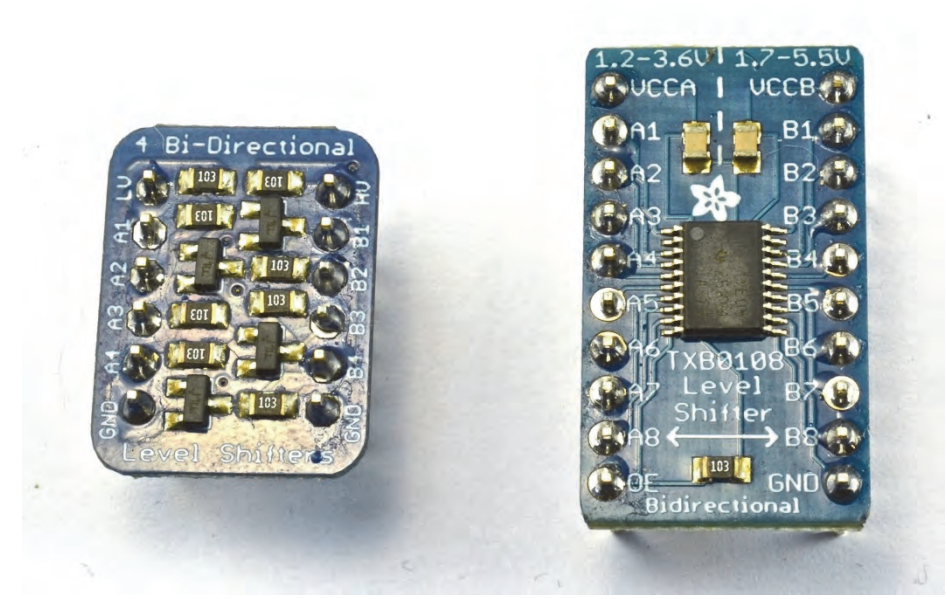


Figura 9.15. *Módulos conversores de nivel.*

Observaciones

Estos conversores de nivel están disponibles con diferentes números de canales. Los dos mostrados en la Figura 9.15 tienen cuatro y ocho canales.

Los recursos para estos conversores de nivel se encuentran en el Apéndice A.

Para saber más

Revise el Capítulo 9.13, especialmente si tiene solamente uno o dos niveles para convertir.

Normalmente, las entradas lógicas de 5 V aceptarán salidas de 3,3 V sin problemas. Sin embargo, en algunos casos como cuando se utilizan tiras de led (Capítulo 14.8), este puede no ser el caso, pueden utilizarse un transistor o uno de los módulos descritos anteriormente para elevar el nivel lógico.

9.15 Alimentar Raspberry Pi con pilas

Problema

Desea conectar su Raspberry Pi a un robot y alimentarlo usando pilas alcalinas.

Solución

Un proyecto típico que usa una Raspberry Pi requiere 5 V hasta aproximadamente 600 mA (revise el Capítulo 1.4). El requisito para 5 V es estricto. No debe tratar de alimentar su Pi desde más o menos 5 V. En la práctica esto significa que es más probable que utilice una pila con un voltaje superior a 5 V —por ejemplo, 9 V— y utilice un regulador de voltaje para bajarlo a 5 V.

El requisito de potencia relativamente alta de Pi lo hace inadecuado para la alimentación de, por ejemplo, una pequeña pila de 9 V o incluso un conjunto de pilas AAA. Lo más adecuado sería un paquete formado por seis pilas recargables AA y un regulador de voltaje.

La Figura 9.16 muestra cómo puede utilizar un regulador de voltaje con un paquete de pilas para alimentar una Pi a través del pin 5 V del conector GPIO.

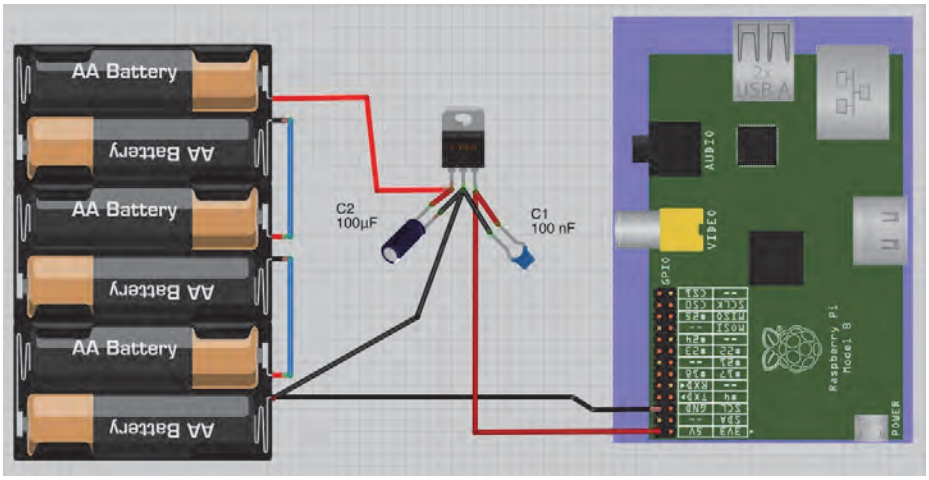


Figura 9.16. *Alimentación de Raspberry Pi con pilas AA.*

El regulador de voltaje 7805 se calentará mucho. Si se calienta demasiado, el protector térmico se activará y el voltaje de la salida caerá, probablemente causando que Pi se reinicie. Un disipador térmico fijo en el Circuito Integrado (IC) sería de gran ayuda.

Una solución alternativa, y más fría, es utilizar una tarjeta RasPiRobot V3 (Capítulo 9.19), como se muestra en la Figura 9.17.

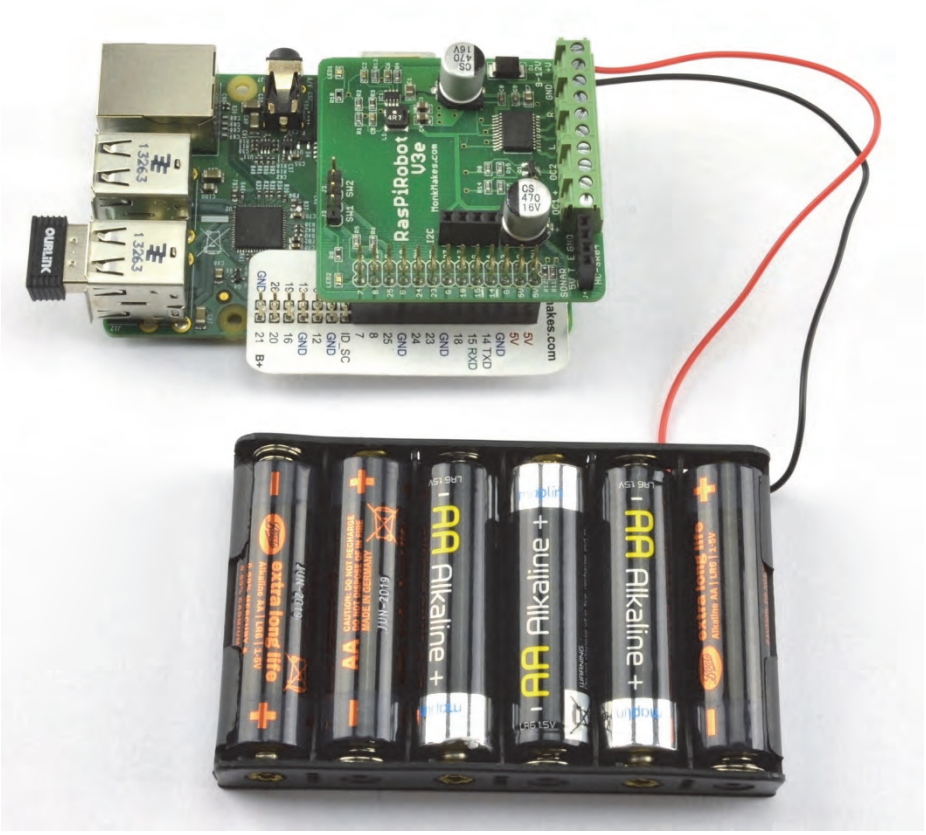


Figura 9.17. *Alimentación de Raspberry Pi con una tarjeta RasPiRobot V3.*

Además de ser un driver de motor, la tarjeta RasPiRobot V3 tiene una fuente de alimentación conmutada incorporada que toma una entrada de entre 7 V y 12 V y suministra 5 V a Raspberry Pi. Debido a que la fuente de alimentación de la tarjeta RasPiRobot V3 es una fuente de alimentación de conmutación eficiente, en lugar de un regulador lineal como el 7805, no se calentará con el uso normal de Raspberry Pi.

Observaciones

El 7805 requiere que el voltaje de entrada sea por lo menos 2 V por encima de 5 V. También puede comprar reguladores de baja caída (LDO), como el LM2940. LM2940 tiene el mismo patillaje que el 7805 pero solo requiere que la entrada sea 0,5 V por encima de la salida de 5 V. Sin embargo, recuerde que nominalmente las pilas de 1,5 V AA caen rápidamente a aproximadamente 1,2 V. Así que es poco probable que un paquete de cuatro de ellas proporcione suficiente tensión durante más de unos minutos. Un paquete de seis será adecuado.

El ejemplo anterior utiliza un paquete de pilas de 9 V, pero si desea que su Raspberry quepa en un automóvil o RV de 12 V, funcionará.

Para saber más

Otra manera de alimentar su Raspberry Pi con pilas es mediante el uso de un cargador de teléfono móvil que acepte pilas AA. Asegúrese de que puede hacer frente a 600 mA o más.

El Capítulo 9.16 muestra cómo encender Raspberry Pi con una batería LiPo. Para obtener más información sobre el uso de tarjetas RasPiRobot V3, consulte el Capítulo 9.19.

9.16 Alimentar Raspberry Pi con una batería LiPo

Problema

Desea conectar su Raspberry Pi a un robot y alimentarlo desde una batería LiPo de 3,7 V.

Solución

Utilice un módulo regulador de potencia (Figura 9.18). El módulo mostrado es de SparkFun, pero hay diseños similares más baratos disponibles en eBay.

Como siempre, con precios tan bajos en eBay, debe probar bien el módulo antes de usarlo. No siempre funcionan exactamente como se anuncia y la calidad puede ser bastante variable.

La ventaja de este tipo de módulo es que actúa como regulador de voltaje para suministrar 5 V a Pi y también tiene una toma USB para suministrar energía al circuito de carga. Si conecta el adaptador de alimentación de Pi en el enchufe del cargador, Pi se alimentará y la batería se cargará, lo que permitirá desconectar la alimentación USB y utilizar Pi con la batería mientras tenga suficiente energía.

Con una batería LiPo de 1300 mA, Pi se encenderá durante dos o tres horas.

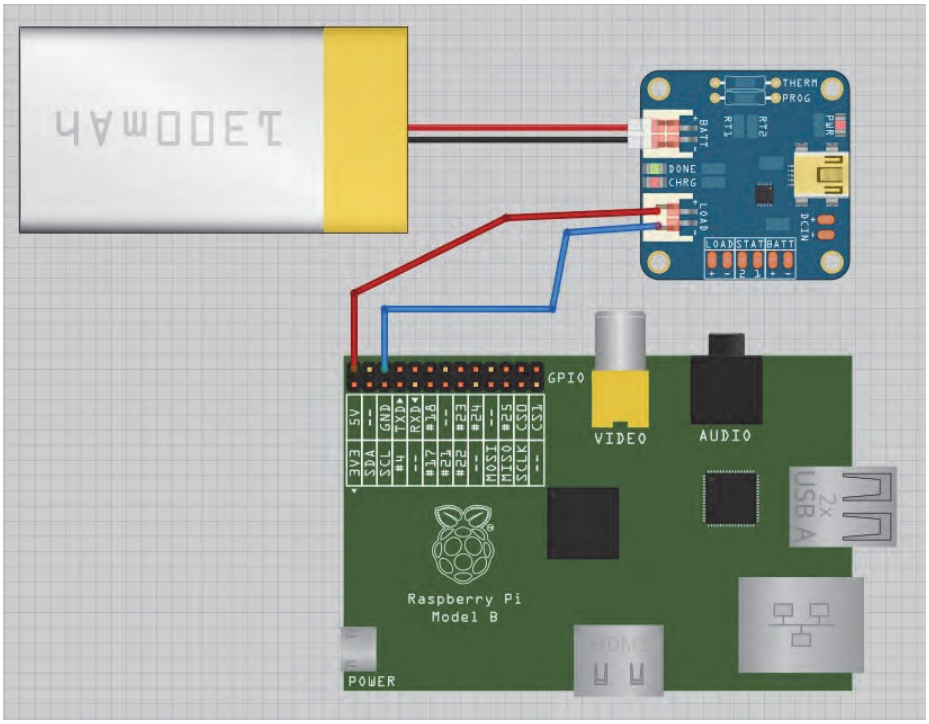


Figura 9.18. *Alimentación de Raspberry Pi con una batería LiPo de 3,7 V.*

Observaciones

Si usted planea utilizar la carga de la batería en otra parte, puede conseguir un módulo convertidor de potencia, sin el cargador, a un coste más bajo.

Para saber más

También puede encontrar paquetes de baterías LiPo USB preprogramadas, a menudo con una batería LiPo de alta capacidad, que alimentará su Pi durante horas.

9.17 Introducción a Sense HAT

Problema

Quiere saber cómo utilizar Raspberry Pi Sense HAT.

Solución

Raspberry Pi Sense HAT (Figura 9.19) es una tarjeta de interfaz para Raspberry Pi. Incluye sensores, de hecho puede medir la temperatura, la humedad relativa y la presión atmosférica (Capítulo 13.11). Tiene un acelerómetro, un giroscopio (Capítulo 13.14) y un magnetómetro (Capítulo 13.15) para proyectos de navegación. También tiene una pantalla matriz de led 8 × 8 de colores (Capítulo 14.4).

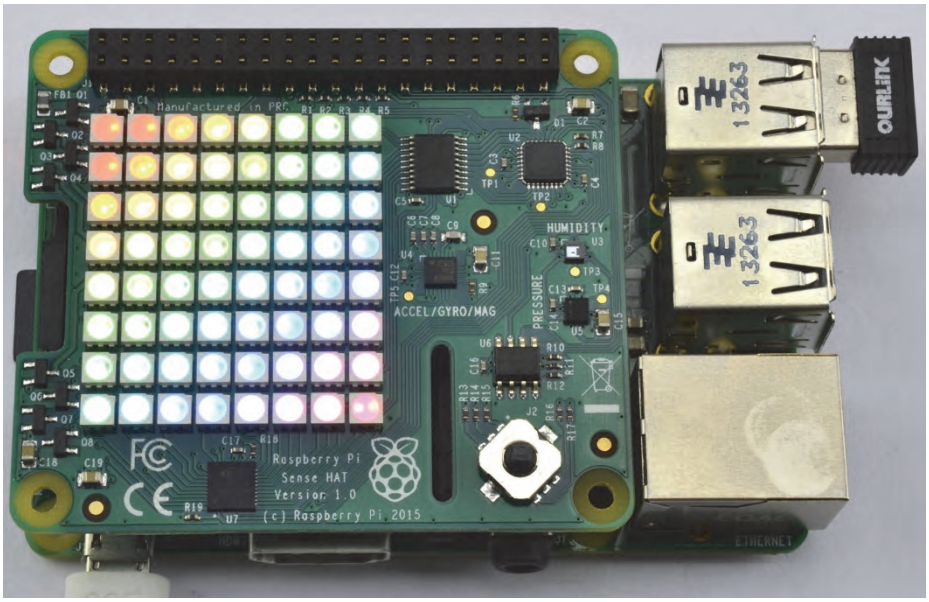


Figura 9.19. *Raspberry Pi Sense HAT.*

Sense HAT necesita una Raspberry Pi con un GPIO de 40 pines, por lo que no será capaz de utilizarlo en una Raspberry Pi antigua con un encabezado de 26 pines.

Sense HAT necesita que se instale *software* antes de poder usarla. El proceso no es complicado, pero puede tardar bastante rato.

Tenga en cuenta que quizás cuando lea esto Raspbian ya incluya todo el *software* necesario para Sense HAT. Si ese es el caso, la ejecución de estos comandos no le causará ningún problema. El instalador solo le dirá que no tiene que hacer nada.

Instale la biblioteca Sense HAT de Python usando el comando:

```
$ sudo apt-get install sense-hat
```

Este instalador habilitará I2C automáticamente, por lo que no será necesario seguir la configuración habitual de I2C (Capítulo 9.4).

La pantalla de Sense HAT utiliza una biblioteca de gráficos llamada Python Image Library (PIL), que debe instalarse mediante el comando:

```
$ sudo pip-3.2 install pillow
```

La instalación tardará un rato, después deberá reiniciar si I2C no estaba habilitado.

Observaciones

Hay más apartados en este libro donde se utiliza Sense HAT, pero por ahora puede comprobar que funciona abriendo una consola de Python usando:

```
$ sudo python
```

A continuación, introduzca los siguientes comandos en la consola de Python:

```
>>> from sense_hat import SenseHat
>>> hat = SenseHat()
>>> hat.show_message('The Raspberry Pi Cookbook')
```

La matriz de ledes debe mostrar el texto del mensaje anterior, desplazándolo a través de la pantalla.

Para saber más

Consulte la referencia de programación para Sense HAT en *https://pythonhosted.org/sense-hat/ api/*.

Para medir la temperatura, la humedad y la presión atmosférica consulte el Capítulo 13.11.

Para usar el acelerómetro y el giroscopio de Sense HAT consulte el Capítulo 13.14.

Para usar el magnetómetro para detectar el norte y detectar la presencia de un imán consulte los Capítulos 13.15 y 13.17, respectivamente.

9.18 Introducción a Explorer HAT Pro

Problema

Quiere saber cómo empezar con Pimoroni Explorer HAT Pro.

Solución

Conecte el HAT en su Raspberry Pi e instale la biblioteca Explorer HAT Pro de Python.

La Figura 9.20 muestra un Pimoroni Explorer HAT Pro en una Raspberry Pi B+. Tenga en cuenta que este HAT solo funcionará en una Raspberry Pi de 40 pines.

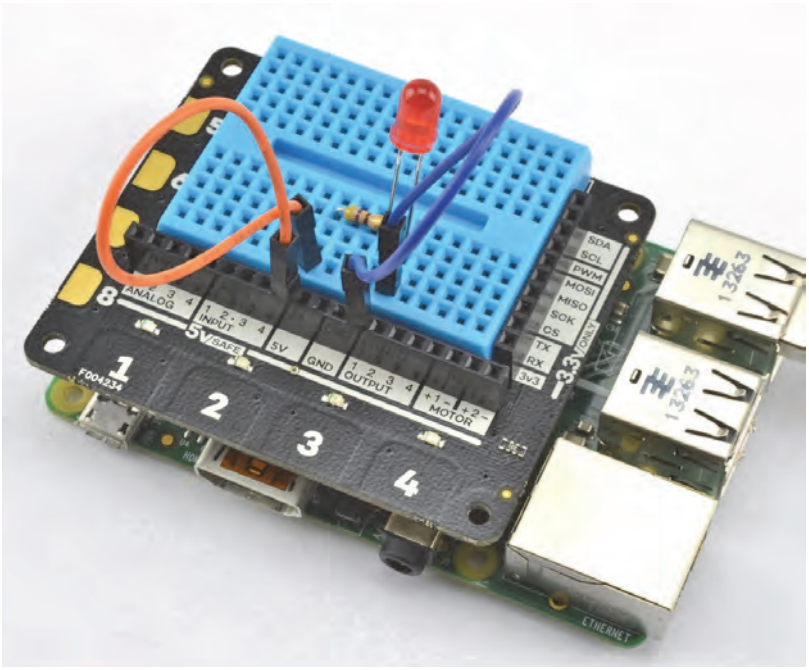


Figura 9.20. *Pimoroni Explorer HAT Pro.*

Explorer HAT Pro tiene algunas opciones útiles de entrada/salida, así como un área donde se puede conectar una pequeña placa de pruebas sin soldadura. Sus características incluyen:

- 4 ledes
- 4 entradas almacenados en búfer
- 4 salidas almacenadas en búfer (hasta 500 mA)
- 4 entradas analógicos
- 2 drivers de motor de baja potencia (máximo 200 mA)
- 4 paneles táctiles capacitivos
- 4 paneles de clip cocodrilo capacitivos

Para instalar la biblioteca de Python para el Explorer HAT Pro ejecute el siguiente comando:

```
$ sudo pip install explorerhat
```

Una vez instalada la biblioteca, puede probar un pequeño experimento para que el led rojo incorporado parpadee.

```
import explorerhat, time

while True:
    explorerhat.light.red.on()
    time.sleep(0.5)
    explorerhat.light.red.off()
    time.sleep(0.5)
```

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas en este libro, puede descargar el programa desde la sección de descargas de la web del libro, que se llama *explorer_blink.py*.

Observaciones

Explorer HAT Pro proporciona cuatro entradas y salidas de búfer; entradas y salidas que no están conectadas directamente a Raspberry Pi, sino a chips en el Explorer HAT Pro. Esto significa que, si accidentalmente conecta cosas incorrectamente, el Explorer HAT Pro se dañará en lugar de su Raspberry Pi.

Para saber más

Puede usar el Explorer HAT para la detección de elementos capacitivos (Capítulo 13.19).

9.19 Introducción a la placa RaspiRobot

Problema

Quiere saber cómo usar una placa RaspiRobot.

Solución

La Figura 9.21 muestra la placa RaspiRobot (versión 3). La placa tiene un controlador de doble motor que se puede utilizar para dos motores de corriente continua o para un solo motor paso a paso. Puede suministrar energía de 5 V a Raspberry Pi usando el regulador de voltaje de modo conmutado incorporado. La placa tiene dos entradas de interruptor, dos salidas de corriente (2 A) y proporciona conexiones fáciles a un telémetro HC-SR04 y a las interfaces I2C de Raspberry Pi.

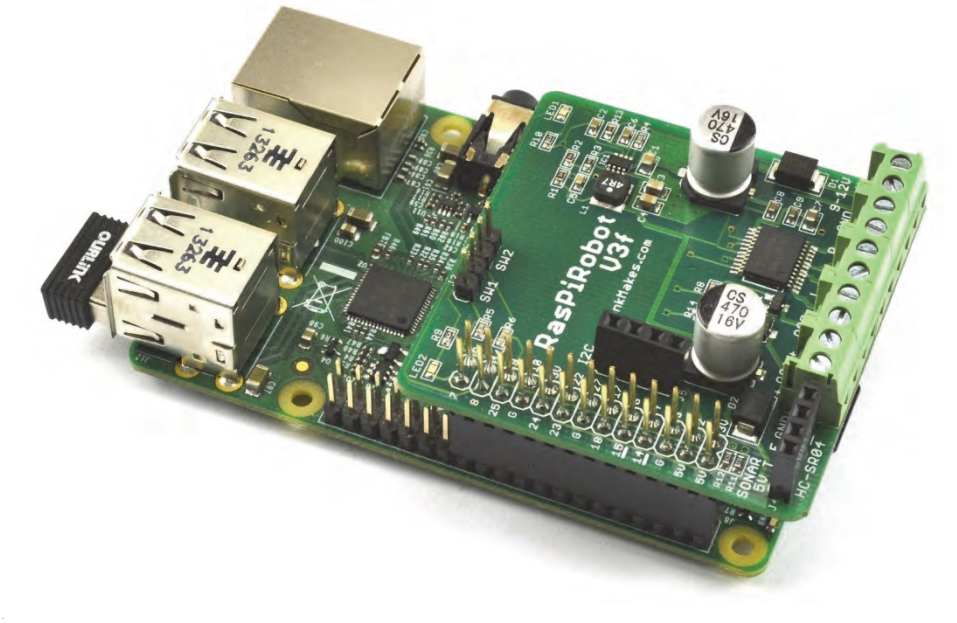


Figura 9.21. *Placa RaspiRobot V3.*

La placa RaspiRobot V3 tiene su propia biblioteca Python que debe descargar e instalar usando los siguientes comandos:

```
$ git clone https://github.com/simonmonk/raspirobotboard3.git
$ cd raspirobotboard3/python
$ sudo python setup.py install
```

Observaciones

Coloque RasPiRobot V3 en Raspberry Pi y luego encienda Raspberry Pi. Puede probar algunos comandos con la placa RaspiRobot utilizando la consola Python sin conectar energía externa o motores a la placa RaspiRobot. Dado que la biblioteca de la placa RaspiRobot utiliza el puerto GPIO, necesitará iniciar Python como superusuario utilizando este comando:

```
$ sudo python
```

Cuando lo encienda por primera vez verá que, en la placa RaspiRobot, ambos ledes se encienden. Introduzca los siguientes comandos y los ledes se apagarán cuando la biblioteca se inicialice:

```
>>> from raspirobotboard import *
>>> rr = RRB3()
```

Intente encender y apagar uno de los ledes usando estos comandos:

```
>>> rr.set_led1(1)
>>> rr.set_led1(0)
```

La placa RaspiRobot tiene dos pares de pines, cada uno diseñado para ser conectado a un interruptor. Están etiquetados como SW1 y SW2 en la placa. Puede probar si SW1 está cerrado utilizando el comando:

```
>>> print(rr.sw1_closed())
False
```

Cortocircuitar los dos pines de SW1 con un destornillador y ejecutar el comando otra vez devolverá `True`.

Existen otros comandos disponibles para configurar las dos salidas de colector abierto (`set_oc1` y `set_oc2`) y los comandos de control del motor (`forward`, `reverse`, `left`, `right` y `stop`). Vea *http://www.raspirobot.com* para obtener la referencia completa de comandos.

Para saber más

Para aprender a usar esta placa para hacer un robot itinerante consulte el Capítulo 11.11.

Para utilizar la placa RaspiRobot para controlar un motor bipolar paso a paso consulte el Capítulo 11.10.

9.20 Utilizar una placa de prototipado para Pi

Problema

Desea utilizar la placa de prototipado para Pi.

Solución

Esta placa (Figura 9.22) es una placa de prototipado en lugar de una placa interfaz como RaspiRobot (Capítulo 9.19). En otras palabras, no incluye ninguna electrónica, ya que está diseñada para que pueda soldar sus propios componentes a un área prototipada.

La placa tiene un área donde puede utilizar un chip de montaje de 16 pines y una fila de cuatro agujeros, espaciados ampliamente, donde puede soldar un terminal de tornillo.

La placa tiene terminales de tornillo en dos lados, que están conectados a todos los pines GPIO. Puede ignorar el área de creación de prototipos y solo utilizar los terminales de tornillo para conectar los cables a los pines GPIO.

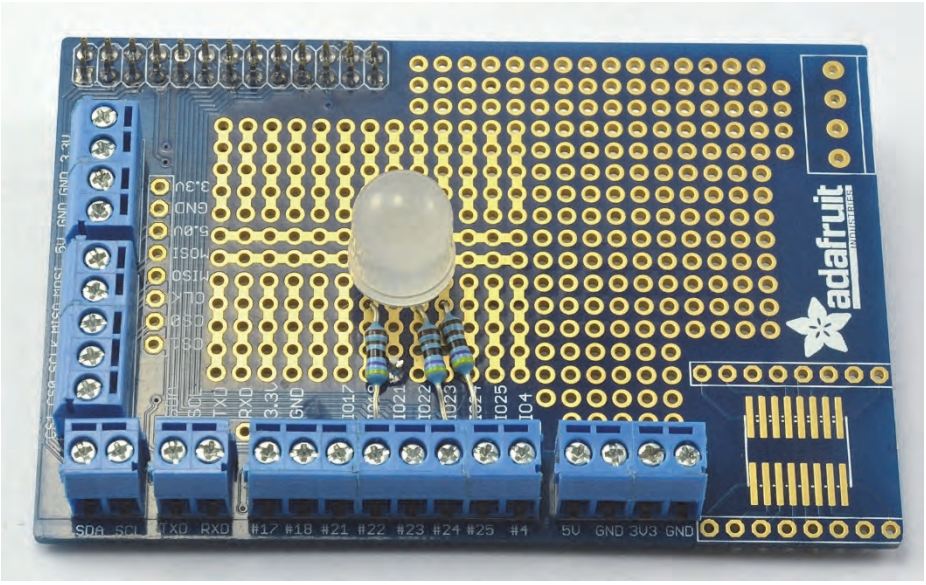


Figura 9.22. *Placa de prototipado para Pi.*

Observaciones

El diseño de la placa tiene una rejilla de agujeros que se utiliza en la mayoría de los componentes, incluyendo los circuitos integrados DIL. Suelde los componentes a la placa empujando los cables a través de los agujeros por la parte superior. Suelde las conexiones por debajo.

Las vías que conectan los agujeros son claramente visibles en la parte superior de la placa y esta se divide en varias zonas. Existe un área con buses centrales de potencia destinados a los circuitos integrados DIL, así como un área de prototipado general y áreas para chips de montaje en superficie y terminales de tornillo adicionales.

Habiendo soldado los componentes en su lugar, necesitará cables extra para enlazarlo todo. Se pueden unir en la parte inferior o superior de la placa, o en ambos si el diseño es complicado.

Es buena idea planear el diseño antes de empezar a soldar.

Con las siguientes instrucciones construirá un led RGB en la placa utilizando el diseño mostrado en la Figura 9.23. Es una versión del Capítulo 10.11, que es similar a diferencia de que el diseño se construye sin soldadura.

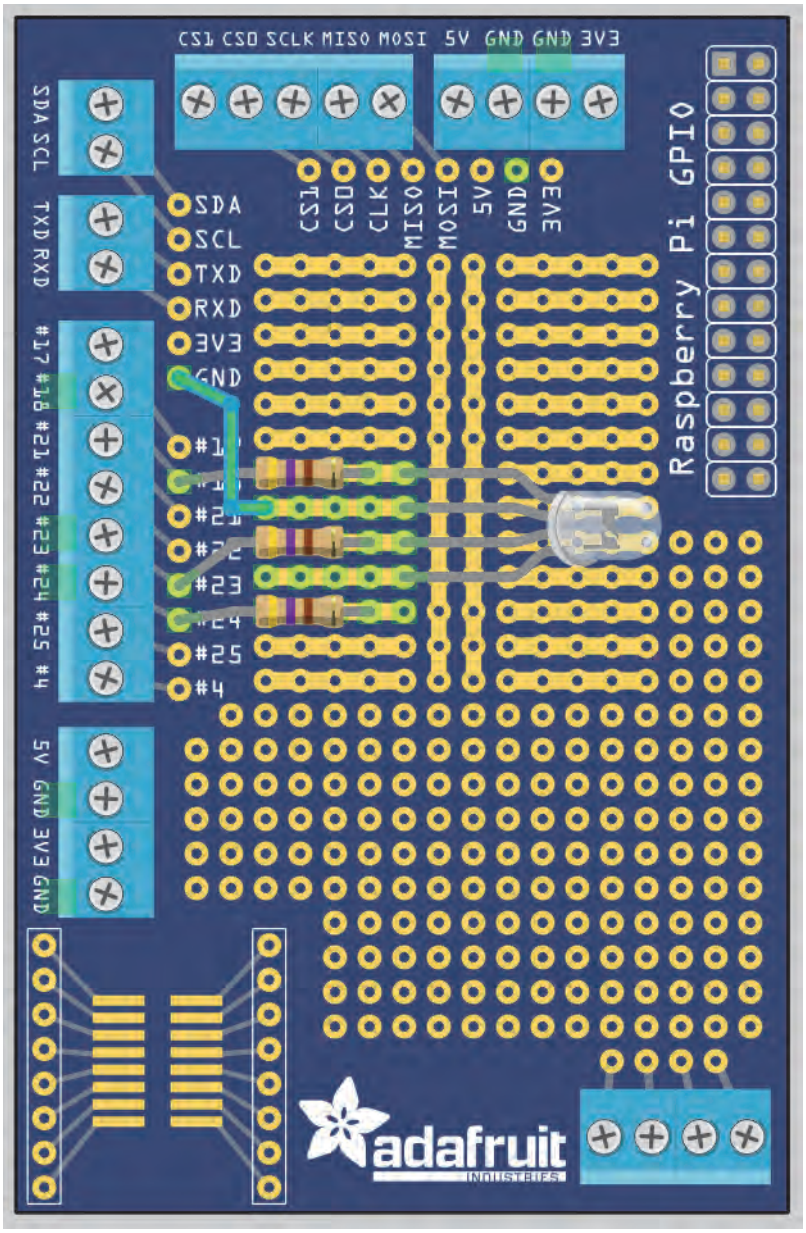


Figura 9.23. *Diseño de la placa de prototipado para un led RGB.*

El primer paso es soldar las resistencias. Doble los cables y empújelos a través de los agujeros de la placa. Después, dele la vuelta a la placa y toque con el soldador los cables que salen de los agujeros durante un segundo más o menos antes de aplicar la soldadura, que debe fluir alrededor del cable (Figura 9.24).

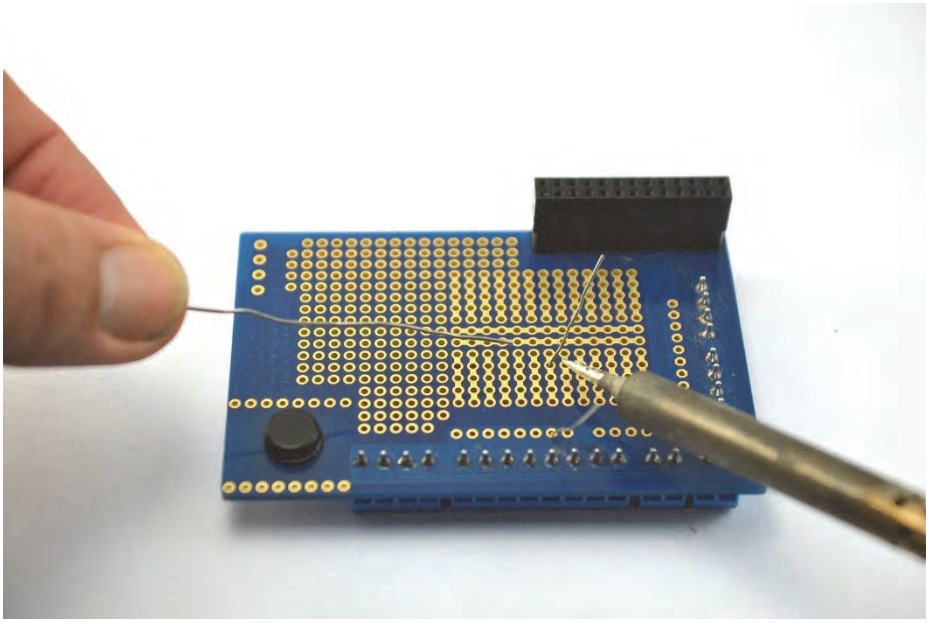


Figura 9.24. *Soldar una resistencia a la placa de prototipado.*

Cuando haya soldado ambos extremos corte el exceso de cable y repita el proceso para los otros resistores (Figura 9.25).

Luego suelde el led, asegurándose de hacerlo de manera correcta. El cable más largo es el cátodo común y debe ser el único cable led conectado a la tira de agujeros que no está conectado a ninguna resistencia. Muy ocasionalmente encontrará ledes en los que el cable más largo no sea el positivo. Esto pasa a menudo con los ledes infrarrojos. Si no está seguro consulte la hoja de datos del led o la página de información de proveedor.

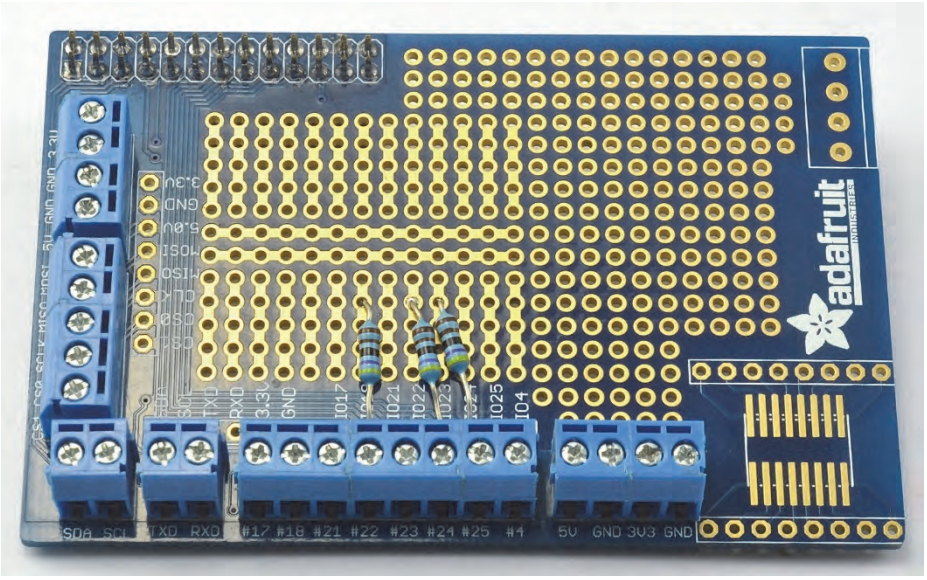


Figura 9.25. *Resistencias soldadas a la placa de prototipado.*

También necesitará soldar un trozo corto de alambre de esa fila a la conexión GND en la placa (Figura 9.26).

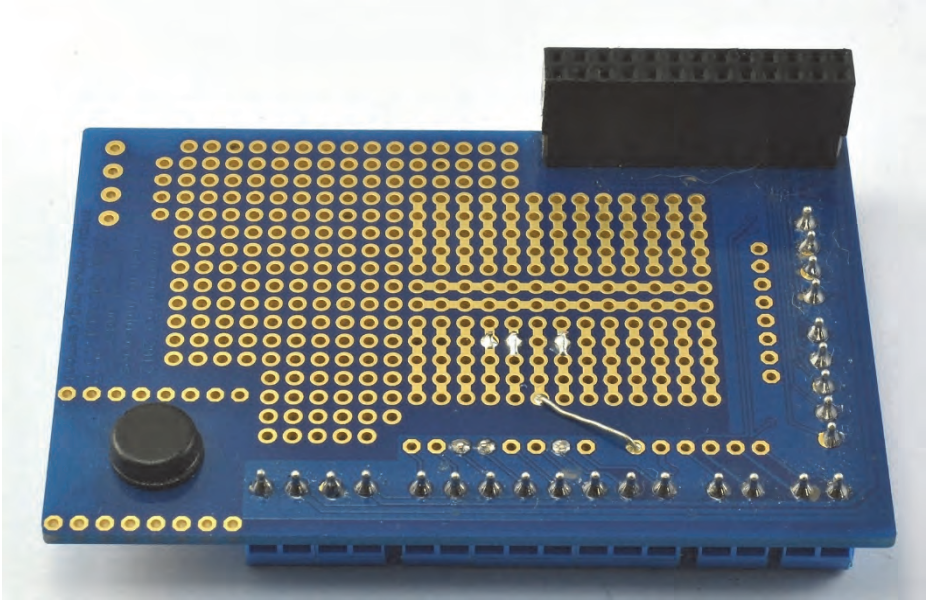


Figura 9.26. *Soldar enlace de cable a la placa de prototipado para Pi.*

Cuando la placa esté completa deberá estar como la de la Figura 9.27.

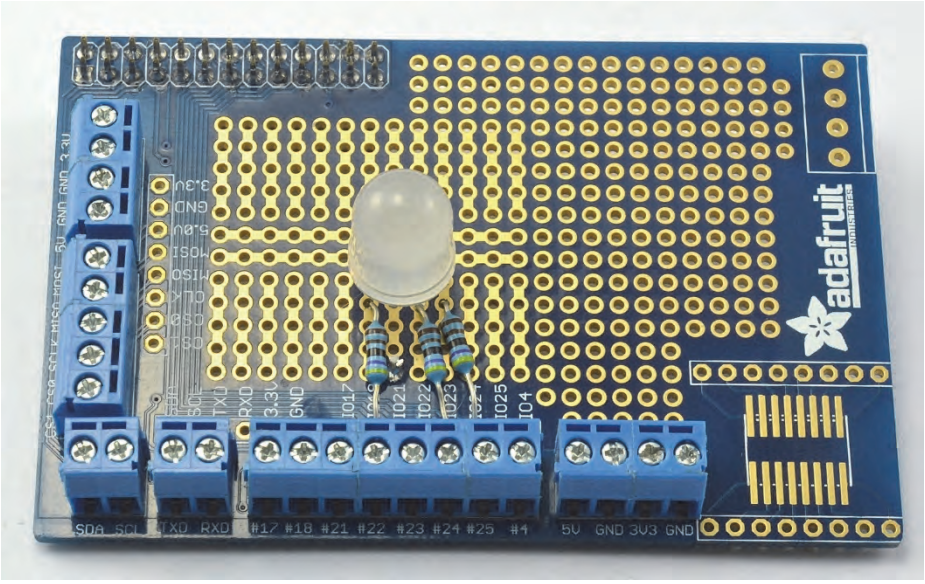


Figura 9.27. *Led RGB acabado en una placa de prototipado para Pi.*

Puede probar el led usando el programa de Python del Capítulo 10.11.

Para saber más

Encontrará más información sobre este producto en la página web de Adafruit.

9.21 Crear un *Hardware At Top* (HAT)

Problema

Desea crear un prototipo de tarjeta de interfaz Raspberry Pi que cumpla con el estándar HAT.

Solución

Utilice un HAT Perma-Proto Pi (Figura 9.28).

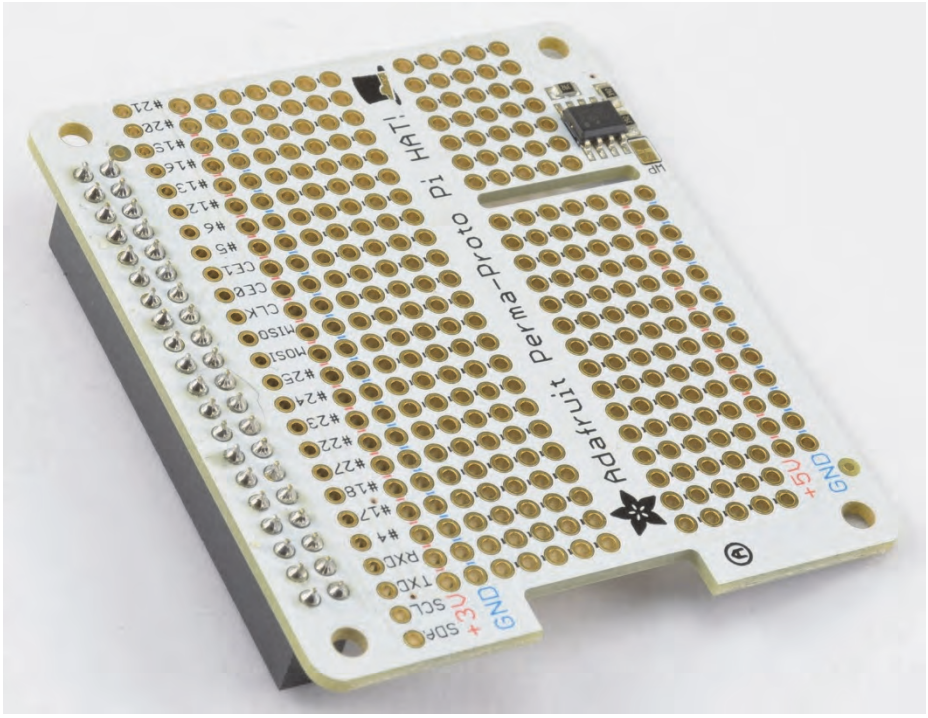


Figura 9.28. *HAT Perma-Proto Pi.*

Con la llegada de Raspberry Pi B+ con un GPIO de 40 pines se definió un nuevo estándar para placas adicionales para Raspberry Pi llamado HAT (*Hardware At Top*). No debe atenerse a este estándar, especialmente si está haciendo un producto únicamente para usted. En cambio, si está diseñando un producto para venderlo, debería ajustarse al estándar HAT.

El estándar HAT define el tamaño y la forma del PCB y también obliga a que el PCB tenga un chip de memoria de solo lectura borrable electrónicamente (EEPROM) soldado en la tarjeta. Este chip está conectado a los pines ID_SD y ID_SC de GPIO y, en el futuro, permitirá que se realice una configuración del Pi e incluso que se cargue automáticamente el *software* cuando arranque Raspberry Pi con un HAT adjunto.

El área de prototipos de la placa se compone de algunos carriles de energía y un diseño de formato de placas de dos filas de cinco agujeros, más los carriles de alimentación a ambos lados de la placa.

Si no le interesa programar la EEPROM puede parar aquí. Sin embargo, si desea añadir su propia información personalizada a la EEPROM de HAT, siga leyendo el apartado «Observaciones».

Observaciones

El estándar HAT tiene mucho sentido. Sin embargo, en el momento en el que escribo esto, Raspbian no utiliza ninguna información escrita en el EEPROM del HAT. Esto probablemente cambiará en un futuro y seguramente nos lleve a la excitante posibilidad de que HAT haga cosas automáticamente, como habilitar I2C e instalar las bibliotecas de Python para su *hardware*, solo por estar presente en Raspberry Pi.

Para escribir datos en la EEPROM primero debe habilitar el puerto oculto I2C utilizado por los pines ID_SD ID_SC, que se utilizan para leer y escribir en la EEPROM. Para hacer eso necesitará editar */boot/config.txt* añadiendo o no comentando la línea:

```
dtparam=i2c_vc=on
```

Una vez hecho esto reinicie su Raspberry Pi y deberá ser capaz de detectar que la EEPROM de I2C está conectada al bus de I2C con herramientas de i2c (Capítulo 9.5).

```
$ i2cdetect -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Puede ver a partir del resultado del comando `i2cdetect` que la EEPROM tiene una dirección I2C de 50. Tenga en cuenta que la opción `-y 0` en lugar del usual `-y 1` se utiliza porque este no es el bus I2C normal en los pines 2 y 3, sino el bus I2C dedicado a la EEPROM del HAT.

Para leer y escribir la EEPROM debe descargar algunas herramientas utilizando los siguientes comandos:

```
$ git clone https://github.com/raspberrypi/hats.git
$ cd hats/eepromutils
$ make
```

Escribir la EEPROM es un proceso de tres pasos. Primero, debe editar el archivo *eeprom_settings.txt*. Cambie al menos los campos *product_id*, *product_version*, *vendor* y *product* para que sean el nombre de la empresa y el producto. Tenga en cuenta que hay muchas otras opciones en este archivo. Incluyen la especificación de las opciones de retroalimentación, los pines GPIO utilizados, etc.

En segundo lugar, después de editar el archivo, ejecute el siguiente comando para convertir el archivo de texto en un archivo adecuado para escribir en la EEPROM (*rom_file.eep*).

```
$ ./eepmake eeprom_settings.txt rom_file.eep
Opening file eeprom_settings.txt for read
```

```
UUID=7aa8b587-9c11-4177-bf14-00e601c5025e
Done reading
Writing out...
Done.
```

Finalmente, copie *rom_file.eep* en la EEPROM ejecutando el siguiente comando:

```
sudo ./eepflash.sh -w -f=rom_file.eep -t=24c32
This will disable the camera so you will need to REBOOT after this...
This will attempt to write to i2c address 0x50. Make sure there is...
This script comes with ABSOLUTELY no warranty. Continue only if you...
Do you wish to continue? (yes/no): yes
Writing...
0+1 records in
0+1 records out
127 bytes (127 B) copied, 2.52071 s, 0.1 kB/s
Done.
pi@raspberrypi ~/hats/eepromutils $
```

Una vez completada la escritura puede volver a leer la ROM utilizando los siguientes comandos:

```
$ sudo ./eepflash.sh -r -f=read_back.eep -t=24c32
$ ./eepdump read_back.eep read_back.txt
$ more read_back.txt
```

Para saber más

Puede encontrar la guía de diseño del HAT de Raspberry Pi aquí: *https://github.com/raspberrypi/ hats*.

Hay muchos HAT predefinidos en el mercado, incluyendo los HAT de los motores paso a paso (Capítulo 11.9), los táctiles capacitivos (Capítulo 13.19) y los HAT PWM de 16 canales (Capítulo 11.2) de Adafruit, así como el HAT Pro del Pimoroni Explorer (Capítulo 9.17).

9.22 Pi Compute Module

Problema

Desea crear un producto que utilice la tecnología de Raspberry Pi, pero no quiere incluir la Raspberry Pi entera.

Solución

Utilice Raspberry Pi Compute module.

Raspberry Pi Compute module (a la izquierda de la Figura 9.29 junto a su placa IO a la derecha) es una tarjeta que contiene el sistema de Raspberry Pi en un chip (SoC) con 512 MB de RAM con forma de un módulo de memoria SODIMM como el que cabría en un ordenador portátil al actualizar su memoria.

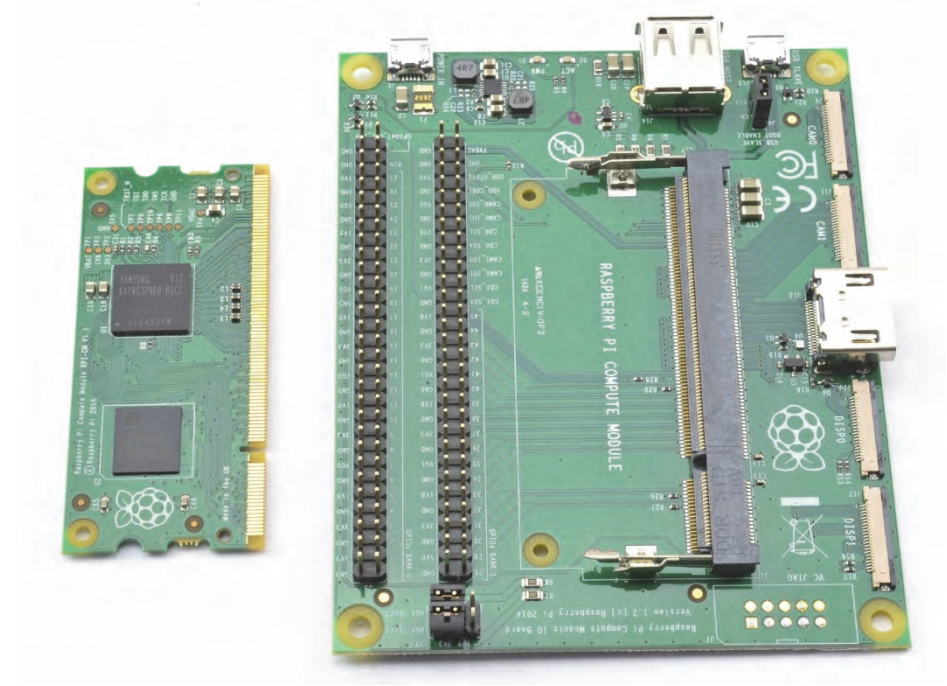


Figura 9.29. *Raspberry Pi Compute module y la tarjeta IO.*

El *hardware* se basa en la Raspberry Pi original, pero en lugar de una ranura de tarjeta SD, hay un chip flash de 4 GB.

La idea que hay detrás de esta placa es que los desarrolladores de productos pueden crear su propio PCB con un zócalo SODIMM en el que Pi Compute module podría estar conectado. La tarjeta IO está ahí para que prototipe sus diseños.

Observaciones

Esta placa fue lanzada antes de Raspberry Pi Zero (Capítulo 9.23), y en el momento en el que escribo este libro parece algo redundante, ya que es más caro que la Zero, no funciona tan bien y tan solo es un poco más pequeño. También carece de algunos de los conectores presentes en Pi Zero, lo que puede ser justo lo que necesita en su producto sin tener que diseñar su propio PCB.

Se han creado una gran cantidad de productos que usan el Compute module, pero, sin una actualización del *hardware* y/o una gran caída de precios, este dispositivo puede tener un futuro incierto.

Para saber más

Para obtener más información completa sobre Pi Compute module consulte *https://www.raspberrypi.org/blog/ raspberry-pi-compute-module-new-product/*.

9.23 Pi Zero

Problema

Quiere aprender más sobre Pi Zero y cómo utilizarla en proyectos de electrónica.

Solución

El coste extremadamente bajo de la Pi Zero y su pequeño tamaño la convierten en la opción ideal para proyectos de electrónica.

La Figura 9.30 muestra una Raspberry Pi Zero.

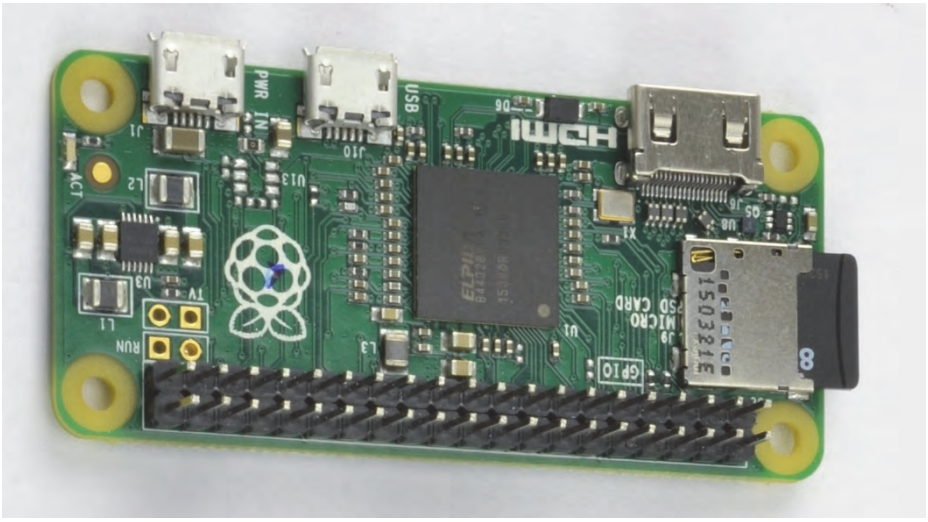


Figura 9.30. *Raspberry Pi Zero.*

Pi Zero se suministra sin encabezados de pines adjuntos, por lo que es probable que su primer trabajo sea soldarlos. Los adecuados están disponibles en kits de inicio de Pi Zero, como el suministrado por Pi Hut.

También puede utilizar encabezados de pines de una fila, simplemente soldando dos longitudes una al lado de la otra.

Observaciones

Con solo un conector USB y un conector micro USB OTG *(on the go)* necesitará un adaptador USB y un concentrador USB para poder conectar un dongle, un teclado y un ratón USB wifi para configurar Pi Zero.

Alternativamente, puede utilizar un cable de consola como el descrito en el Capítulo 2.7 para configurar el wifi editando */etc/network/interfaces* como se describe en el Capítulo 2.6. Una vez configurado puede conectarse a la Pi Zero sin cables usando SSH (Capítulo 2.8).

Para saber más

Para una comparación de los modelos de Raspberry Pi disponibles revise el Capítulo 1.2.

CAPÍTULO 10

Control de *hardware*

10.1 Introducción

Este capítulo trata el control de la electrónica a través del conector GPIO de Raspberry Pi.

En la mayoría de los capítulos se necesitan placas de prototipado sin soldadura y cables macho-hembra y macho-macho (revise el Capítulo 9.9). Para mantener la compatibilidad con los modelos más antiguos de Raspberry Pi de 26 pines, en todos los ejemplos de placas de pruebas solo se utilizan los 26 pines más comunes para los dos diseños de GPIO (revise el Capítulo 9.2).

10.2 Conectar un led

Asegúrese de consultar el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Quiere saber cómo conectar un led a Raspberry Pi.

Solución

Conecte un led a uno de los pines GPIO usando un resistor en serie de 470 Ω o 1 kΩ para limitar la corriente. En este apartado necesitará:

- Placa de pruebas y cables puente (consulte "Equipos para prototipos", página 474)
- Resistor de 470 Ω (consulte "Resistores y condensadores", página 474)
- Led (consulte "Optoelectrónica", página 476)

La Figura 10.1 muestra cómo se puede cablear este led utilizando una placa de pruebas sin soldar y cables puente macho a hembra.

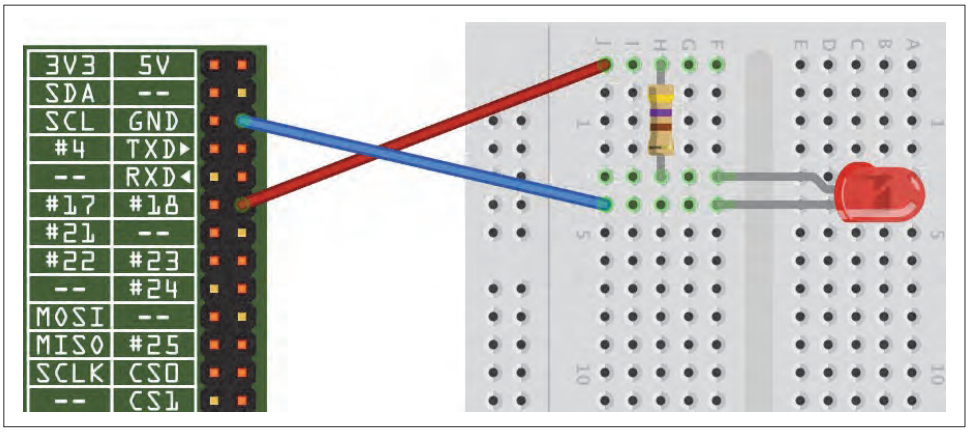


Figura 10.1. *Conectar un led a Raspberry Pi.*

Una vez conectado el led necesitamos poder encenderlo y apagarlo usando comandos de Python.

Inicie una consola de Python desde el terminal con acceso de superusuario e introduzca estos comandos:

```
$ sudo python
>>> import RPi.GPIO as GPIO
>>> GPIO.setmode(GPIO.BCM)
>>> GPIO.setup(18, GPIO.OUT)
>>> GPIO.output(18, True)
>>> GPIO.output(18, False)
```

Esto encenderá y apagará el led.

Observaciones

Los ledes son una forma muy útil, barata y eficiente de producir luz, pero hay que tener cuidado en cómo usarlos. Si están conectados directamente a una fuente de tensión (como una salida GPIO), que es mayor que aproximadamente 1,7 voltios, crearán una corriente muy grande. Esto puede ser suficiente para romper el led o lo que sea que esté proporcionando corriente, lo cual no es bueno si su Raspberry Pi está proporcionando la corriente.

Debe utilizar siempre un resistor en serie con un led, porque el resistor en serie se coloca *entre* el led y la fuente de tensión, lo que limita la cantidad de corriente que fluye a través del led a un nivel seguro, tanto para el led como para el pin GPIO.

Con los pines GPIO de Raspberry Pi solo se garantiza que proporcionen unos 3 mA o 16 mA de corriente (dependiendo de la placa y el número de pines en uso) Puede consultar el Capítulo 9.3. Los ledes se iluminarán generalmente con cualquier corriente mayor a 1 mA, pero serán más brillantes con más corriente. Utilice la Tabla 10.1 como guía para seleccionar un resistor en serie basado en el tipo de led. La tabla también indica la corriente aproximada que se extraerá del pin GPIO.

Tabla 10.1. *Selección de resistores en serie para led y un pin GPIO de 3,3 V.*

Tipo de led	Resistencia	Corriente (mA)
Rojo	470 Ω	3,5
Rojo	1 kΩ	1,5
Naranja, amarillo, verde	470 Ω	2
Naranja, amarillo, verde	1 kΩ	1
Azul, blanco	100 Ω	3
Azul, blanco	270 Ω	1

Como puede ver, en todos los casos es seguro usar un resistor de 470 Ω. Si está utilizando un led azul o blanco, puede reducir el valor de la resistencia en serie considerablemente sin riesgo de dañar su Raspberry Pi.

Si desea ampliar los experimentos que realizó en la consola de Python en un programa que hace que el led parpadee repetidamente, puede pegar el siguiente código en los editores IDLE (Capítulo 5.3) o nano (Capítulo 3.7). Guarde el archivo como *led_blink.py*. También puede descargar el programa de la sección de descargas en la página web del libro.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

while (True):
    GPIO.output(18, True)
    time.sleep(0.5)
    GPIO.output(18, False)
    time.sleep(0.5)
```

Recuerde que para ejecutar el programa debe tener privilegios de superusuario para la biblioteca `RPi.GPIO`, por lo que necesitará usar este comando:

```
$ sudo python led_blink.py
```

Para saber más

Eche un vistazo a esta útil calculadora de resistencia en serie.

Para obtener más información sobre el uso de placas de pruebas y cables puente con Raspberry Pi revise el Capítulo 9.9.

10.3 Dejar los pines GPIO en un estado seguro

Problema

Desea establecer todos los pines GPIO como entradas cuando su programa se cierre, para que haya menos posibilidades de un corte accidental en el GPIO, lo que podría dañar su Raspberry Pi.

Solución

Utilice la construcción `try: finally:` y el método `GPIO.cleanup`.

El ejemplo de parpadeo del Capítulo 10.2 se puede volver a escribir con seguridad como se muestra a continuación. El archivo para el código se llama *led_blink_safe.py*.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

try:
    while (True):
        GPIO.output(18, True)
        time.sleep(0.5)
        GPIO.output(18, False)
        time.sleep(0.5)
finally:
    print("Cleaning Up!")
    GPIO.cleanup()
```

Ahora, cuando pulse Ctrl-C en el programa para cerrarlo llamará a `GPIO.cleanup` antes de que el programa se cierre.

Observaciones

Si no se le llama o no se reinicia Pi, los pines establecidos como salidas permanecerán como salidas una vez finalizado el programa. Si fuese a iniciar el cableado para un nuevo proyecto, sin darse cuenta de este problema, su nuevo circuito podría cortar accidentalmente una salida GPIO en uno de los carriles de suministro u otro pin GPIO en el estado opuesto.

Esto podría ocurrir al conectar un pulsador, conectando un pin GPIO que haya configurado como salida y HIGH a GND, como en el Capítulo 12.2.

En resumen, tenga cuidado al intercambiar *hardware*, use `GPIO.cleanup` o reinicie su Pi. En cualquier caso, es buena idea apagar su Pi mientras se esté conectando *hardware* nuevo.

Para saber más

Para obtener más información sobre el tratamiento de excepciones en Python revise el Capítulo 7.11.

10.4 Controlar el brillo de un led

Problema

Desea variar el brillo de un led desde un programa Python.

Solución

La biblioteca RPi.GPIO tiene una función de modulación por ancho de pulsos (PWM) que le permite controlar la potencia de un led y su brillo.

Para probarlo conecte un led como se describe en el Capítulo 10.3 y ejecute este programa de prueba (*led_brightness.py*):

```
import RPi.GPIO as GPIO

led_pin = 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)

pwm_led = GPIO.PWM(led_pin, 500)
pwm_led.start(100)

while True:
        duty_s = raw_input("Enter Brightness (0 to 100):")
        duty = int(duty_s)
        pwm_led.ChangeDutyCycle(duty)
```

Si está utilizando Python 3 en lugar de Python 2, cambie el comando `raw_input` a `input`.

Ejecute el programa de Python y podrá cambiar el brillo introduciendo un número entre 0 y 100:

```
pi@raspberrypi ~ $ sudo python
led_brightness.py Enter Brightness (0 to 100):0
Enter Brightness (0 to 100):20
Enter Brightness (0 to 100):10
Enter Brightness (0 to 100):5
Enter Brightness (0 to 100):1
Enter Brightness (0 to 100):90
```

Salga del programa pulsando Ctrl-C.

Observaciones

PWM es una técnica en la que se varía la longitud de los pulsos, manteniendo constante el número total de pulsos por segundo (la frecuencia en Hz). La Figura 10.2 ilustra el principio básico de PWM.

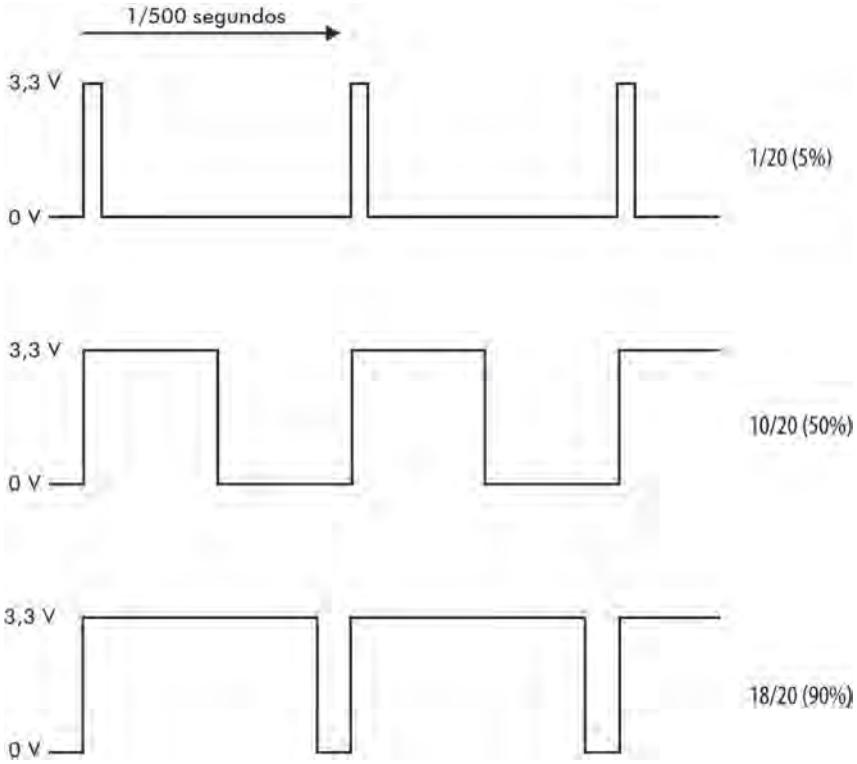


Figura 10.2. *Modulación por ancho de pulsos.*

En altas frecuencias, la frecuencia PWM medida varía algo de la frecuencia suministrada como argumento. Esto podría cambiar en versiones posteriores de la PWM de `RPi.GPIO`.

Puede cambiar la frecuencia PWM modificando esta línea:

```
pwm_led = GPIO.PWM(led_pin, 500)
```

El valor es en Hz, por lo que, en este caso, la frecuencia se establece en 500 Hz.

La Tabla 10.2 compara las frecuencias especificadas en el segundo parámetro con `GPIO.PWM` con la frecuencia real en el pin medida con un osciloscopio.

Tabla 10.2. *Frecuencia requerida contra frecuencia real.*

Frecuencia requerida	Frecuencia medida
50 Hz	50 Hz
100 Hz	98,7 Hz
200 Hz	195 Hz
500 Hz	470 Hz
1 kHz	890 Hz
10 kHz	4,4 kHz

También encontré que a medida que aumentaba la frecuencia, su estabilidad disminuía. Eso significa que la característica PWM no es buena para el audio, pero es lo suficientemente rápida para controlar el brillo de los ledes o la velocidad de los motores.

Para saber más

Para obtener más información sobre PWM consulte Wikipedia.

El Capítulo 10.11 usa PWM para cambiar el color de un led RGB, y el Capítulo 11.5 usa PWM para controlar la velocidad de un motor de corriente continua.

Para obtener más información sobre el uso de placas de pruebas y cables puente con Raspberry Pi revise el Capítulo 9.9. También puede controlar el brillo de un led con un control deslizante. Consulte el Capítulo 10.10.

Para tener otro enfoque para controlar el color de un led RGB usando la biblioteca led RGB Squid revise el Capítulo 9.11.

10.5 Producir un zumbido

Problema

Quiere producir un zumbido con Raspberry Pi.

Solución

Utilice un zumbador piezoeléctrico conectado a un pin GPIO.

La mayoría de los zumbadores piezoeléctricos funcionan muy bien con la disposición mostrada en la Figura 10.3. El que he usado es un componente suministrado por Adafruit (vea "Varios" en la página 477). Puede conectar los pines del zumbador directamente a Raspberry Pi usando cabezales hembra a hembra (consulte "Equipo de prototipos" en la página 474).

Estos zumbadores utilizan muy poca corriente. Pero, si tiene un zumbador grande o simplemente quiere usarlo con seguridad, ponga un resistor de 470 Ω entre el pin GPIO y el zumbador.

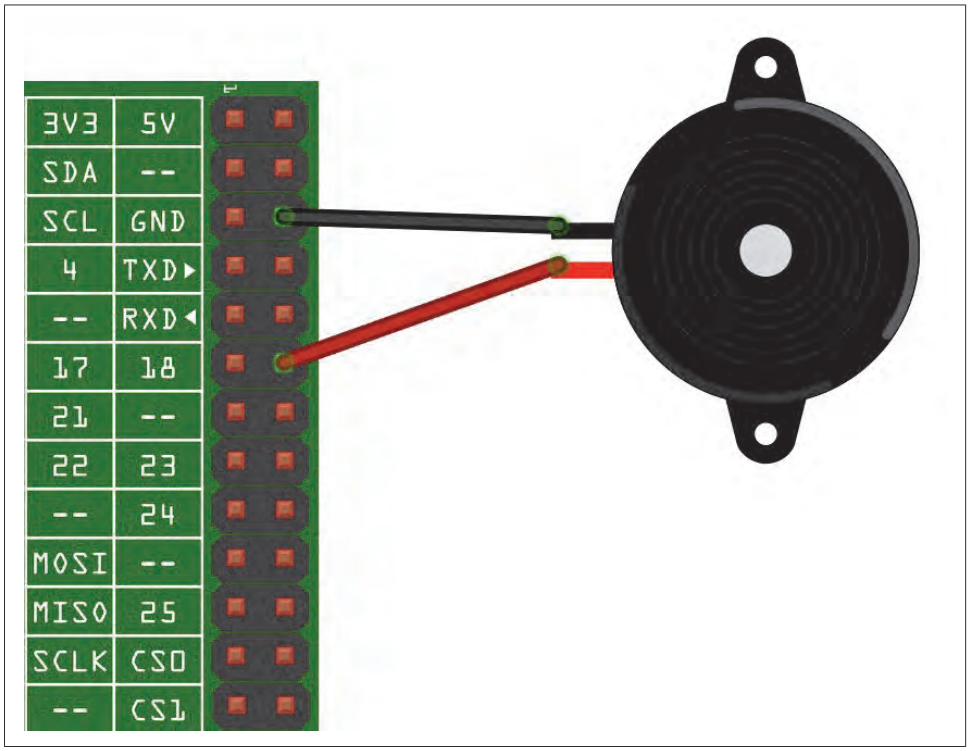


Figura 10.3. *Conexión de un zumbador a Raspberry Pi.*

Pegue el siguiente código en los editores IDLE (Capítulo 5.3) o nano (Capítulo 3.7). Guarde el archivo como *buzzer.py*. También puede descargar el programa desde la sección de descargas de la página web del libro.

```
import RPi.GPIO as GPIO
import time

buzzer_pin = 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer_pin, GPIO.OUT)

def buzz(pitch, duration):
    period = 1.0 / pitch
    delay = period / 2
    cycles = int(duration * pitch)
    for i in range(cycles):
        GPIO.output(buzzer_pin, True)
        time.sleep(delay)
        GPIO.output(buzzer_pin, False)
        time.sleep(delay)
```

```
while True:
    pitch_s = raw_input("Enter Pitch (200 to 2000): ")
    pitch = float(pitch_s)
    duration_s = raw_input("Enter Duration (seconds): ")
    duration = float(duration_s)
    buzz(pitch, duration)
```

Cuando ejecute el programa, primero le pedirá el tono en Hz y luego la duración del zumbido en segundos:

```
$ sudo python buzzer.py
Enter Pitch (2000 to 10000): 2000
Enter Duration (seconds): 20
```

Observaciones

Los zumbadores piezo no tienen una amplia gama de frecuencias, ni la calidad del sonido es especialmente buena.

El programa funciona simplemente conmutando el pin GPIO 18, apagándolo y encendiéndolo con un breve retraso en el medio. El retraso se calcula a partir del tono. Cuanto mayor sea el tono (frecuencia), más corto será el retraso.

Para saber más

Puede encontrar la hoja de datos para el zumbador piezo aquí: *http://bit.ly/Iwkv2R*.

10.6 Conmutar un dispositivo CC de alta potencia mediante un transistor

Problema

Desea controlar la potencia de un dispositivo de CC de alta potencia y bajo voltaje, como un módulo led de 12 V.

Solución

Estos ledes de alta potencia utilizan demasiada corriente para encenderse directamente desde un pin GPIO. También necesitan 12 V en lugar de 3,3 V. Para controlar una potencia tan alta es necesario utilizar un transistor.

En este caso, se utilizará un tipo de transistor de alta potencia llamado transistor de efecto de campo metal-óxido-semiconductor (MOSFET), que cuesta menos de un euro, pero puede manejar cargas de hasta 30 amperios, más de lo que se necesita para los ledes de alta potencia. El MOSFET utilizado es un FQP30N06L (vea "Transistores y diodos", página 475).

La Figura 10.4 muestra cómo se conecta un MOSFET a una placa de pruebas. Identifique correctamente los cables de alimentación positiva y negativa para el módulo led.

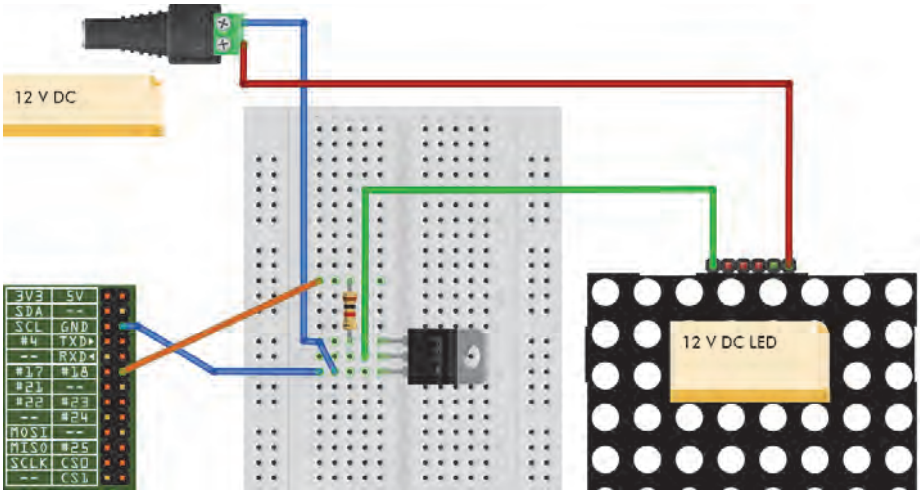


Figura 10.4. *Control de grandes corrientes con un MOSFET.*

Para hacer esto necesitará:

- Una placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 1 kΩ (vea "Resistores y condensadores", página 474)
- Transistor MOSFET FQP30N06L N-Channel o transistor TIP120 Darlington (vea "Transistores y diodos", página 475)
- Adaptador de corriente de 12 V
- Módulo de led de 12 V DC

El código Python para encender y apagar el panel led es exactamente el mismo que si controláramos un solo led de baja potencia sin el MOSFET (revise el Capítulo 10.2).

También puede utilizar PWM con el MOSFET para controlar el brillo del módulo led (consulte el Capítulo 10.4).

Observaciones

Siempre que necesite alimentar algo importante con el conector GPIO use baterías o un adaptador de alimentación externo. El conector GPIO solo puede suministrar corrientes relativamente bajas (Capítulo 9.3). En este caso, usará un adaptador de alimentación de 12 V DC para suministrar energía al panel led. Elija un adaptador de corriente que tenga suficiente energía. Por lo tanto, si el módulo led es de 5 W, necesitará al menos una fuente de alimentación de 12 V 5 W (6 W sería mejor). Si la fuente de alimentación especifica una corriente máxima en lugar de potencia, puede calcular su potencia multiplicando el voltaje por la corriente máxima. Por lo tanto, una fuente de alimentación de 500 mA 12 V puede proporcionar 6 W de energía.

El resistor es necesario para asegurar que las corrientes de pico que se producen cuando el MOSFET cambia de apagado a encendido y viceversa no sobrecarguen el pin GPIO. El MOSFET conmuta el lado negativo del panel led, por lo que el suministro positivo se conecta directamente al lado positivo del panel led, y el lado negativo del panel led se conecta al *drenaje* del MOSFET. La conexión de la *fuente* del MOSFET está conectada a GND, y el pin de la *puerta* MOSFET controla el flujo de corriente desde el drenaje a la fuente. Si el voltaje de la puerta está por encima de 2 V, el MOSFET se encenderá y la corriente fluirá a través de él y del módulo led.

El MOSFET utilizado aquí es un FQP30N06L. La L del final significa que es un MOSFET de nivel lógico cuya tensión de *umbral* de puerta es la adecuada para el uso con salidas digitales de 3,3 V. La versión de este MOSFET que no tiene L es probable que también funcione bien, pero no se puede garantizar que el rango especificado de tensión de umbral de puerta sea de 2 V a 4 V.

Una alternativa al MOSFET es utilizar un transistor Darlington de potencia como el TIP120. Esto tiene un patillaje compatible con FQP30N06L, por lo tanto, puede mantener el mismo diseño de la placa de pruebas.

Este circuito es adecuado para controlar la potencia de otros dispositivos de CC de baja tensión. Las únicas excepciones reales son los motores y los relés, que requieren algún tratamiento adicional (consulte el Capítulo 10.7).

Para saber más

Eche un vistazo a la hoja de datos para el MOSFET.

Si desea crear una interfaz gráfica de usuario para controlar su módulo led, consulte el Capítulo 10.9 para el control de apagado/encendido, y el Capítulo 10.10 para el control variable del brillo con un control deslizante.

10.7 Conmutar un dispositivo de alta potencia con un relé

Problema

Desea encender o apagar dispositivos que no sean adecuados para conmutar con un MOSFET.

Solución

Utilice un relé y un pequeño transistor.

La Figura 10.5 muestra cómo se puede conectar un transistor y un relé en una placa de pruebas. Asegúrese de que el transistor y el diodo estén colocados correctamente. El diodo tiene una raya en el extremo y el transistor utilizado aquí tiene un lado plano y otro curvado.

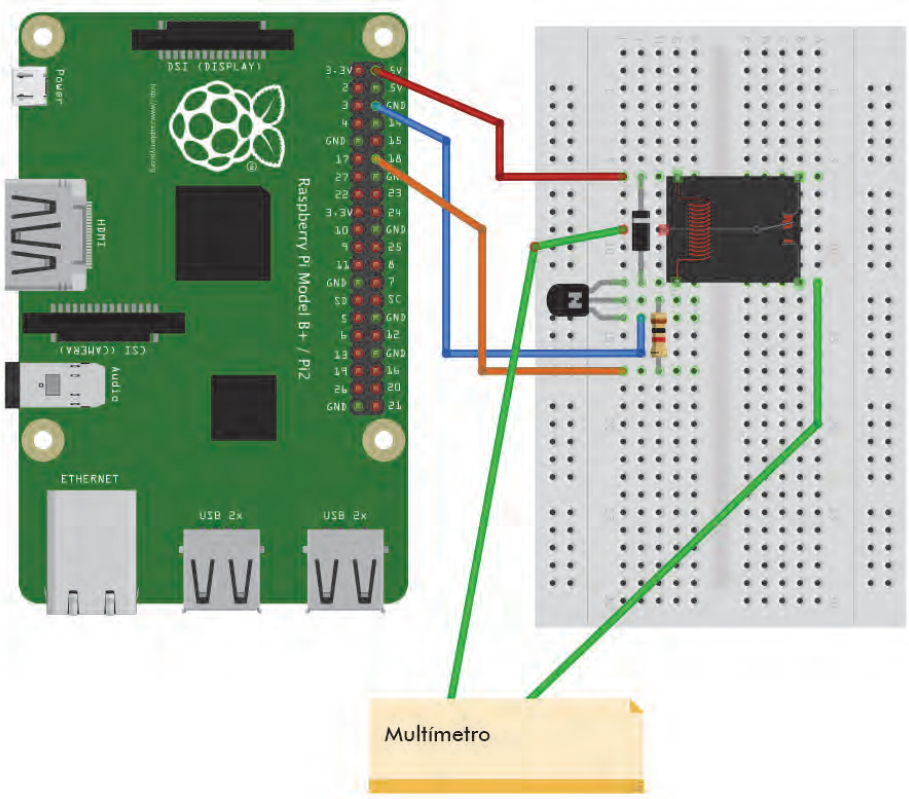


Figura 10.5. *Usar un relé con Raspberry Pi.*

Para hacer esto, necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 1 kΩ (vea "Resistores y condensadores", página 474)
- Transistor 2N3904 (vea "Transistores y diodos", página 475)
- Diodo 1N4001 (vea "Transistores y diodos", página 475)
- Relé de 5 V (vea "Varios" en la página 477)
- Multímetro

Puede utilizar el mismo programa de parpadeo de led que utilizó en el Capítulo 10.2. Si todo va bien, escuchará un clic del relé cada vez que los contactos estén cerrados. Sin embargo, los relés son dispositivos mecánicos lentos, así que no trate de usarlos con PWM. Puede dañar el relé.

Observaciones

Los relés han existido desde los primeros días de la electrónica y tienen la gran ventaja de ser fáciles de usar, además funcionan en cualquier situación en la que un conmutador funcione de manera normal —por ejemplo, cuando cambie a AC (corriente alterna)— o en situaciones en las que el cableado exacto del dispositivo conmutado sea desconocido.

Si se pide a los contactos del relé que excedan sus especificaciones, la vida del relé se acortará. Habrá arcos y los contactos eventualmente se podrán fusionar. También existe la posibilidad de que el relé se caliente peligrosamente.

La Figura 10.6 muestra el símbolo esquemático, el diseño del pin y un relé típico.

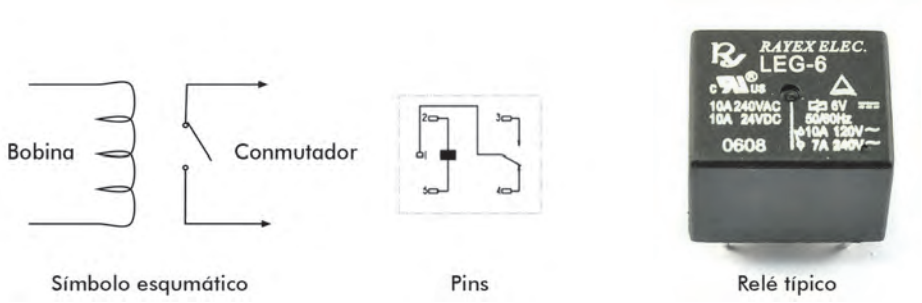


Figura 10.6. *Funcionamiento de un relé.*

Un relé es esencialmente un conmutador cuyos contactos están cerrados cuando un electroimán tira de ellos. Puesto que el electroimán y el conmutador no están conectados eléctricamente de ninguna manera, esto protege el circuito de la bobina del relé de los altos voltajes en el lado del conmutador.

La desventaja de los relés es que son lentos para operar y eventualmente se desgastarán después de cientos de miles operaciones. Esto significa que solo son adecuados para un control lento de encendido/apagado y no para una conmutación rápida como PWM.

La bobina de un relé requiere cerca de 50 mA para cerrar las conexiones. Debido a que un pin GPIO de la Raspberry Pi solo es capaz de suministrar alrededor de 3 mA, es necesario utilizar un pequeño transistor como conmutador. No es necesario utilizar un MOSFET de alta potencia como se hizo en el Capítulo 10.6, pero en su lugar puede utilizar un pequeño transistor. Este tiene tres conexiones. La base (cable intermedio) se conecta al pin GPIO a través de un resistor de 1 kΩ para limitar la corriente, el *emisor* está conectado a GND y el *colector* está conectado a un lado del relé. El otro lado del relé está conectado a 5 V en el conector GPIO. El diodo se utiliza para suprimir los impulsos de alta tensión que se producen cuando el transistor cambia rápidamente la alimentación a la bobina del relé.

Aunque los relés pueden utilizarse para conmutar CA de 110 V o 240 V, este voltaje es muy peligroso y no debe utilizarse en una placa de pruebas. Si desea conmutar los voltajes altos, lea el Capítulo 10.8.

Para saber más

Para conmutar la CC usando un MOSFET de potencia revise el Capítulo 10.6.

10.8 Controlar los dispositivos de CA de alto voltaje

Problema

Desea conmutar 110 o 240 V CA usando Raspberry Pi.

Solución

Utilice PowerSwitch Tail II (vea la Figura 10.7). Este práctico dispositivo hace que sea realmente fácil encender y apagar el equipo de CA desde Raspberry Pi. Tiene un enchufe de la CA en un extremo y un enchufe en el otro, como un cable de extensión; la única diferencia es que la caja de control tiene tres terminales de tornillo. Al conectar el terminal 2 a GND y el terminal 1 al pin GPIO, el dispositivo actúa como un interruptor para encender y apagar el aparato.

Puede usar el mismo código Python que usó en el Capítulo 10.2 para usar PowerSwitch Tail, como se muestra en la Figura 10.7.

Observaciones

PowerSwitch Tail utiliza un relé, pero para conmutar el relé utiliza un componente llamado *optoaislador*, que tiene un led que brilla sobre un fototriac (un interruptor de alta tensión sensible a la luz); cuando el led está iluminado el fototriac conduce, suministrando corriente a la bobina del relé.

El led dentro del optoaislador tiene la corriente limitada por un resistor, de modo que solo 3 mA fluyen a través de ella cuando se suministra con 3,3 V desde un pin GPIO.

También encontrará dispositivos similares a PowerSwitch Tail, y más baratos, en eBay y Amazon.

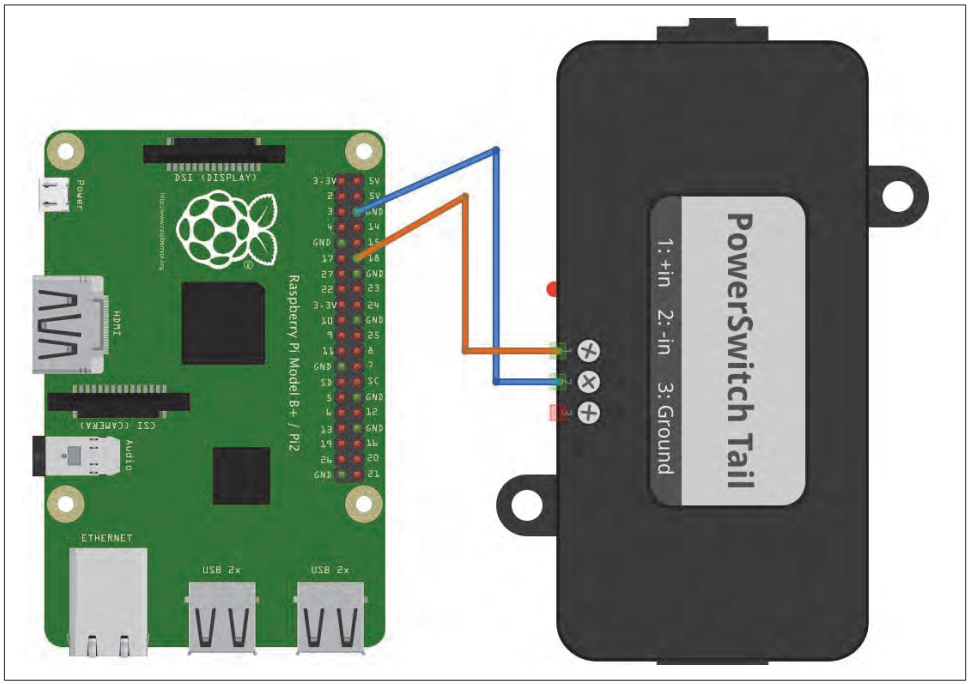


Figura 10.7. *Usar PowerSwitch Tail con Raspberry Pi.*

Para saber más

Para conmutar la CC usando un MOSFET de potencia revise el Capítulo 10.6; y para la conmutación usando un relé en una placa de pruebas consulte el Capítulo 10.7.

La versión de 240 V de PowerSwitch Tail está disponible como un kit.

10.9 Crear una interfaz de usuario para activar y desactivar cosas

Problema

Desea hacer una aplicación que se ejecute en Raspberry Pi y que tenga un botón para activar y desactivar cosas.

Solución

Utilizando el marco de la interfaz de usuario Tkinter escriba un programa Python que use una casilla de verificación para activar y desactivar un pin GPIO (Figura 10.8).

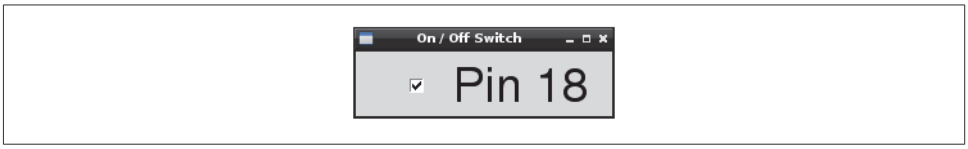


Figura 10.8. *Interfaz de usuario para activar y desactivar cosas.*

Necesitará conectar un led o algún otro tipo de dispositivo de salida al pin GPIO 18. Utilizar un led (Capítulo 10.2) es la opción más fácil para comenzar.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código en la página web del libro, donde pone *gui_switch.py*.

```
from Tkinter import *
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        self.check_var = BooleanVar()
        check = Checkbutton(frame, text='Pin 18',
                 command=self.update,
                 variable=self.check_var, onvalue=True, offvalue=False)
        check.grid(row=1)

    def update(self):
        GPIO.output(18, self.check_var.get())

root = Tk()
root.wm_title('On / Off Switch')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

Tenga en cuenta que tendrá que ejecutarlo con `sudo` porque el `RPi.GPIO` necesita que tenga privilegios de superusuario para acceder al hardware GPIO:

```
$ sudo python gui_switch.py
```

En Python 3 la biblioteca Tkinter ha sido renombrada `tkinter`, con *t* minúscula.

Observaciones

El programa de ejemplo define una clase llamada `App` que contiene la mayor parte del código de la aplicación. Su función de inicialización crea una variable miembro llamada `check_var` que contiene una instancia de `BooleanVar` que se suministra como `variable` a la casilla de verificación. Esto asegura que cada vez que se haga clic en la casilla de verificación, el valor de esta variable cambiará. La opción `command` ejecuta el comando `update` cada vez que ocurre un cambio.

La función `update` simplemente escribe el valor en `check_var` de la salida GPIO.

Para saber más

Puede utilizar este programa para controlar un led (Figura 10-8), un dispositivo CC de alta potencia (Capítulo 10.6), un relé (Capítulo 10.7) o un dispositivo de CA de alta tensión (Capítulo 10.8).

10.10 Crear una interfaz de usuario para controlar la potencia de PWM en ledes y motores

Problema

Desea hacer una aplicación que se ejecute en Raspberry Pi y que tenga un control deslizante para controlar la potencia de un dispositivo usando PWM.

Solución

Usando el marco de la interfaz de usuario Tkinter, escriba un programa Python que use un deslizador para cambiar el ciclo de trabajo de PWM entre 0 y 100 % (Figura 10.9).

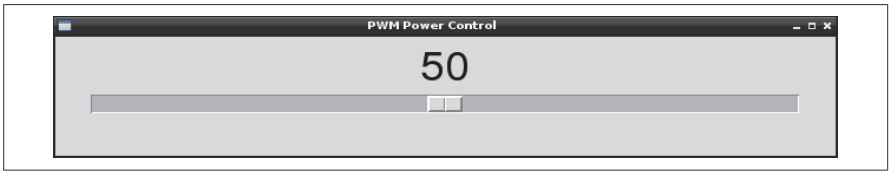


Figura 10.9. *Interfaz de usuario para controlar la potencia de PWM.*

Necesitará conectar un led o algún otro tipo de dispositivo de salida al pin 18 GPIO que sea capaz de responder a una señal PWM. El uso de un led (Capítulo 10.2) es la opción más fácil para empezar.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código en la página web del libro, donde pone *gui_slider.py*.

```
from Tkinter import *
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
pwm = GPIO.PWM(18, 500)
pwm.start(100)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        scale = Scale(frame, from_=0, to=100,
              orient=HORIZONTAL, command=self.update)
        scale.grid(row=0)


    def update(self, duty):
        pwm.ChangeDutyCycle(float(duty)
        )

root = Tk()
root.wm_title('PWM Power Control')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

Tenga en cuenta que tendrá que ejecutarlo con `sudo` porque el `RPi.GPIO` necesita que tenga privilegios de superusuario para acceder al hardware GPIO.

```
$ sudo python gui_slider.py
```

Observaciones

El programa de ejemplo define una clase llamada `App` que contiene la mayor parte del código de la aplicación. La opción `command` ejecuta el comando `update` cada vez que se cambia el valor del control deslizante. Esto actualiza el ciclo de trabajo del pin de salida.

Para saber más

Puede utilizar este programa para controlar un led (Capítulo 10.2), un motor de CC (Capítulo 11.5) o un dispositivo de CC de alta potencia (Capítulo 10.6).

10.11 Cambiar el color de un led RGB

Problema

Desea controlar el color de un led RGB.

Solución

Utilice PWM para controlar la potencia de cada uno de los canales rojo, verde y azul de un led RGB.

Para esto necesitará:

- Placa de pruebas y cables puente (vea "Equipo para prototipos", página 474)
- Tres resistores 470 Ω (vea "Resistores y condensadores", página 474)
- Led de cátodo común RGB ("Optoelectrónica", página 476)
- Un Perma-Proto (Capítulo 9.10) o una placa de prototipado para Pi (revise el Capítulo 9.20) para hacer un proyecto más permanente (opcional)

La Figura 10.10 muestra cómo puede conectar su led RGB a una placa de pruebas. Asegúrese de que el led está correcto; el cable más largo debe ser el segundo desde arriba del todo de la placa de pruebas. Esta conexión se llama *cátodo común*, ya que las conexiones negativas (cátodos) de los ledes rojo, verde y azul dentro de la carcasa led tienen todos sus lados negativos conectados entre sí para reducir el número de pines necesarios.

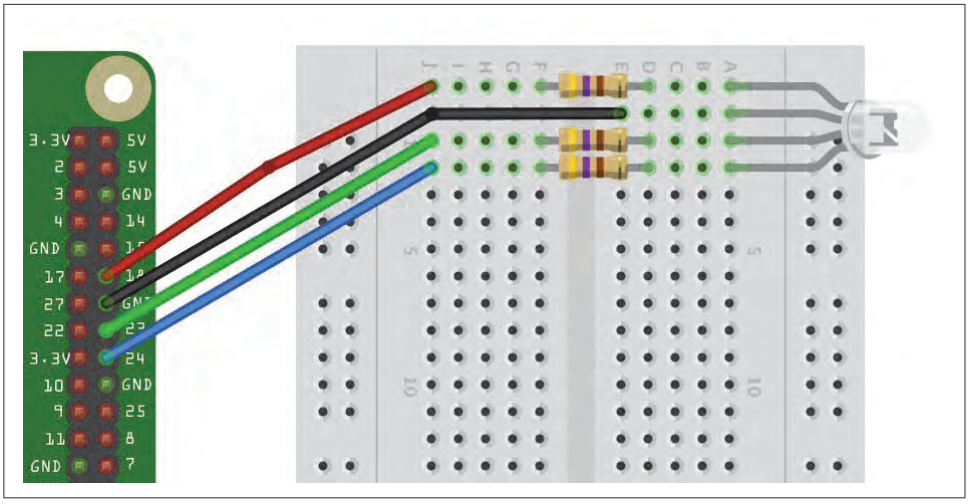


Figura 10.10. *Uso de un led RGB con Raspberry Pi.*

Una alternativa al uso de una placa de pruebas es usar una Raspberry Squid (revise el Capítulo 9.11).

El próximo programa tiene tres controles deslizantes para controlar los canales rojo, verde y azul del led (Figura 10.11).

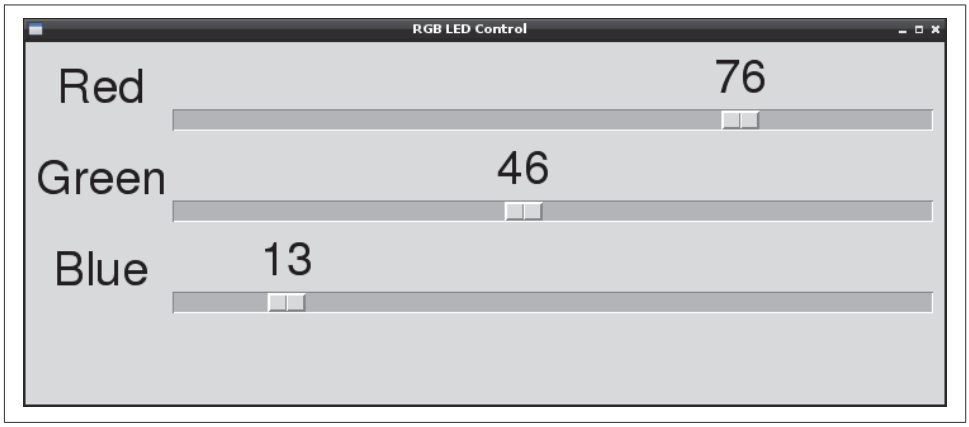


Figura 10.11. *Uso de una interfaz de usuario para controlar un led RGB.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *gui_sliderRGB.py*.

```
from Tkinter import *
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)

pwmRed = GPIO.PWM(18, 500)
pwmRed.start(100)

pwmGreen = GPIO.PWM(23, 500)
pwmGreen.start(100)

pwmBlue = GPIO.PWM(24, 500)
pwmBlue.start(100)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        Label(frame, text='Red').grid(row=0, column=0)
        Label(frame, text='Green').grid(row=1, column=0)
        Label(frame, text='Blue').grid(row=2, column=0)
        scaleRed = Scale(frame, from_=0, to=100,
              orient=HORIZONTAL, command=self.updateRed)
        scaleRed.grid(row=0, column=1)
```

```
        scaleGreen = Scale(frame, from_=0, to=100,
              orient=HORIZONTAL, command=self.updateGreen)
        scaleGreen.grid(row=1, column=1)
        scaleBlue = Scale(frame, from_=0, to=100,
              orient=HORIZONTAL, command=self.updateBlue)
        scaleBlue.grid(row=2, column=1)


    def updateRed(self, duty):
        pwmRed.ChangeDutyCycle(float(duty))

    def updateGreen(self, duty):
        pwmGreen.ChangeDutyCycle(float(duty))

    def updateBlue(self, duty):
        pwmBlue.ChangeDutyCycle(float(duty))

root = Tk()
root.wm_title('RGB LED Control')
app = App(root)
root.geometry("200x150+0+0")
root.mainloop()
```

Observaciones

El código es similar en operación al control para un solo canal PWM, descrito en el Capítulo 10.10. Sin embargo, en este caso, necesita tres canales PWM y tres controles deslizantes, uno para cada color.

El tipo de led RGB usado aquí es un cátodo común. Si tiene el tipo de ánodo común puede usarlo, pero conéctelo al pin de 3,3 V en el conector GPIO. Verá que el control deslizante se invierte, por lo que con un ajuste de 100 se apagará y con 0 se encenderá.

Cuando seleccione un led para este proyecto tenga en cuenta que los ledes *difusos* serán los mejores porque permiten que los colores se mezclen mejor.

Para saber más

Si solo desea controlar un canal PWM, revise el Capítulo 10.10.

Para conocer otro enfoque para controlar el color de un led RGB utilice la biblioteca Squid RGB LED (Capítulo 9.11).

10.12 Usar muchos ledes (Charlieplexing)

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Quiere controlar muchos ledes utilizando los mínimos pines GPIO posibles.

Solución

La manera de hacer esto es usando la técnica llamada *Charlieplexing*. El nombre proviene del inventor, Charlie Allen de la compañía Maxim, y la técnica aprovecha la característica de los pines GPIO que les permite cambiar de salidas a entradas mientras un programa está funcionando. Cuando se cambie un pin para que sea una entrada, no fluirá corriente suficiente para iluminar un led o influir en otros pines conectados al led configurados como salidas.

La Figura 10.12 muestra el esquema para controlar seis ledes con tres pines.

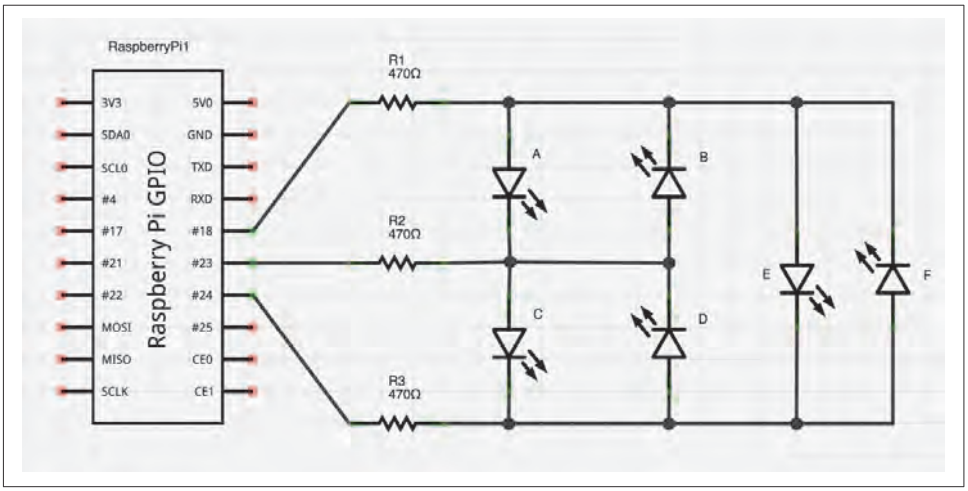


Figura 10.12. *Charlieplexing.*

La Figura 10.13 muestra la disposición de la placa de pruebas para los ledes y los resistores. Para hacer esto necesitará:

- Placa de pruebas y cables puente (consulte "Equipo para prototipos", página 474)
- Tres resistores de 470 Ω (revise "Resistores y condensadores", página 474)
- Seis ledes (consulte "Optoelectrónica" en la página 476)

Abra el editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la página web del libro, donde pone *charlieplexing.py*.

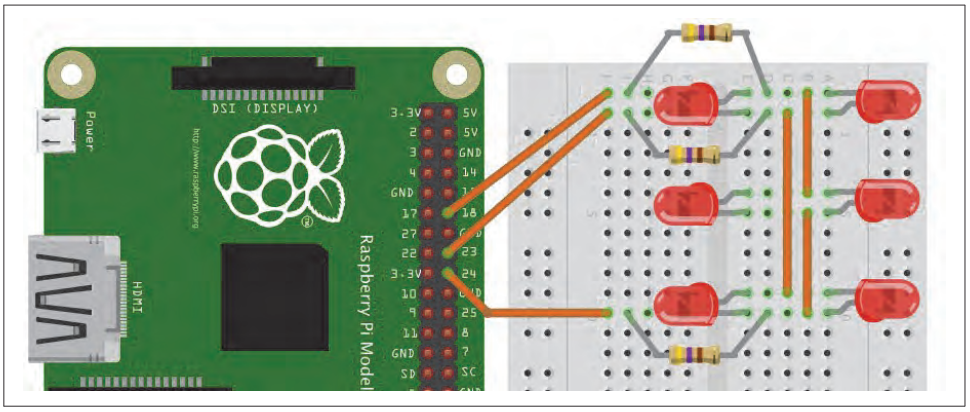


Figura 10.13. *Diseño de la placa de pruebas de Charlieplexing.*

Este código de ejemplo le pide que introduzca un número entre 0 y 5, que encenderá uno de los seis ledes:

```
import RPi.GPIO as GPIO

 pins = [18, 23, 24]

pin_led_states = [
  [1, 0, -1], # A
   [0, 1, -1], # B
   [-1, 1, 0], # C
   [-1, 0, 1], # D
   [1, -1, 0], # E
   [0, -1, 1]  # F
]

GPIO.setmode(GPIO.BCM)

def set_pin(pin_index, pin_state):
    if pin_state == -1:
        GPIO.setup(pins[pin_index], GPIO.IN)
    else:
        GPIO.setup(pins[pin_index], GPIO.OUT)
```

```
            GPIO.output(pins[pin_index], pin_state)

    def light_led(led_number):
        for pin_index, pin_state in enumerate(pin_led_states[led_number]):
            set_pin(pin_index, pin_state)

    set_pin(0, -1)
    set_pin(1, -1)
    set_pin(2, -1)

    while True:
        x = int(raw_input("Pin (0 to 5):"))
        light_led(x)
```

Observaciones

Para entender cómo funciona Charlieplexing, imagine que desea encender el led A en la Figura 10.12. Un led solo se encenderá cuando su cable positivo sea alto y su cable negativo esté bajo. Si el voltaje es al revés, no se encenderá. Para encender el led A necesitará tener su cable conectado a GPIO 18 (a través de un resistor) para que sea alto, y el otro cable al led A, conectado al GPIO 23 por un resistor, para que sea bajo. Sin embargo, también debe asegurarse de que el GPIO 24 esté configurado para que sea una entrada; de lo contrario, los ledes C o D también se iluminarán dependiendo de si el GPIO 24 está alto o bajo.

La matriz *pin_led_states* contiene los ajustes de cada GPIO para cada uno de los seis ledes. Si el valor es 0, el pin es bajo; 1 significa alto; y -1 significa que es una entrada.

El número de ledes que se pueden controlar por pin GPIO viene dado por la fórmula:

$$\text{ledes} = n^2 - n$$

Utilizando cuatro pines puede tener 16, 4, o 12 ledes, mientras que 10 pines le darían un número de 90 ledes.

En este ejemplo, está iluminando solo un led a la vez. Para encender más de uno a la vez debe ejecutar un bucle de actualización que mantenga el estado deseado de los ledes en una matriz y actualice la pantalla, activando los ledes que deben estar encendidos antes de pasar al siguiente. Debe hacerlo lo suficientemente rápido para que parezca que más de uno de los ledes está encendido al mismo tiempo.

Cuantos más ledes utilice cuando trate de hacer que parezca que más de un led está encendido a la vez, menos tiempo se iluminará el led realmente y se convertirá en un dimmer de led.

Para saber más

Para obtener más información sobre Charlieplexing, consulte la Wikipedia.

10.13 Usar un medidor analógico como pantalla

Problema

Desea conectar un voltímetro de panel analógico a Raspberry Pi.

Solución

Suponiendo que tiene un voltímetro de 5 V, puede usar una salida PWM para conducir el medidor directamente, conectando el lado negativo del medidor a tierra y el lado positivo a un pin GPIO (Figura 10.14). Si el medidor es de tipo común de 5 V, solo podrá mostrar voltajes de hasta 3,3 V.

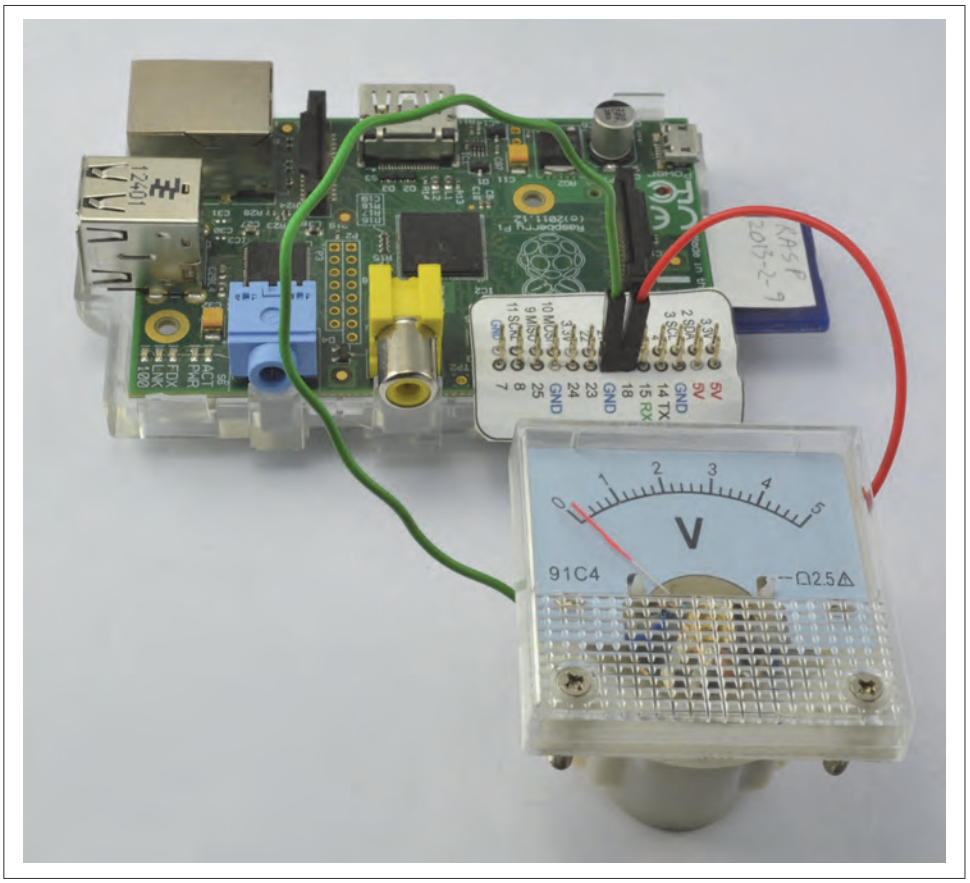


Figura 10.14. *Conectar un voltímetro directamente a un pin GPIO.*

Si desea utilizar el rango completo de un voltímetro de 5 V, necesitará un transistor para que actúe como interruptor para la señal PWM y un resistor de 1 kΩ para limitar la corriente a la base del transistor.

Para hacer esto necesitará:

- Panel del medidor de 5 V ("Varios", página 477)
- Placa de pruebas y cables puente (consulte "Equipos para prototipos", página 474)
- Dos resistores de 1 kΩ (vea "Resistores y condensadores", página 474)
- Transistor 2N3904 (revise "Transistores y diodos", página 475)

La distribución de la placa de pruebas para ello se muestra en la Figura 10.15.

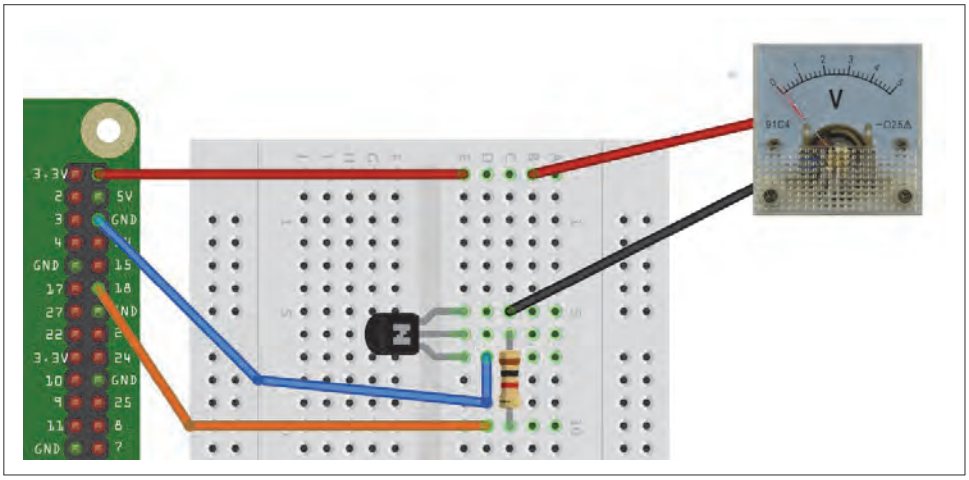


Figura 10.15. *Usar un panel de medidor de 5 V con GPIOde 3,3 V.*

Observaciones

Para probar el voltímetro utilice el mismo programa que utilizó para controlar el brillo de led en el Capítulo 10.10.

Probablemente notará que la aguja da una lectura constante en cualquiera de los extremos de la escala, pero que en cualquier otro lugar se agita un poco. Este es un efecto secundario de cómo se generan las señales PWM. Para obtener un resultado más estable puede utilizar *hardware* externo de PWM como el módulo de 16 canales utilizado en el Capítulo 11.4.

Para saber más

Para obtener más información sobre cómo funcionan los voltímetros antiguos consulte la Wikipedia.

Para obtener más información sobre el uso de la placa de pruebas y los cables puente con Raspberry Pi revise el Capítulo 9.9.

10.14 Programar con interrupciones

Problema

Desea responder a algún evento, como pulsar un botón, sin tener que consultar continuamente el pin de entrada para ver si su estado ha cambiado.

Solución

Utilice la función `add_event_detect` de la biblioteca `RPi.GPIO`.

El próximo ejemplo muestra cómo se puede conectar una rutina de servicio de interrupción para que se active cuando se presione un botón.

Conecte el interruptor a una placa de pruebas como se muestra en la Figura 10.16. Alternativamente, podría utilizar un botón Squid (Capítulo 9.12).

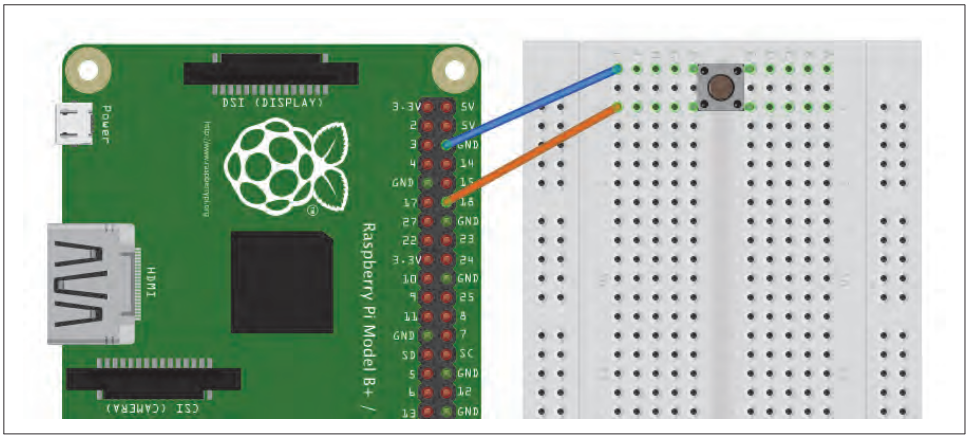


Figura 10.16. *Conexión de un interruptor a una entrada GPIO para mostrar interrupciones.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa desde la sección de código de la página web del libro, donde pone *interrupts.py*.

Este código de ejemplo actualiza continuamente un recuento en segundos y muestra un mensaje cuando se presiona el botón:

```
import RPi.GPIO as GPIO
import time
```

```
GPIO.setmode(GPIO.BCM)


def my_callback(channel):
    print('You pressed the button')

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.add_event_detect(18, GPIO.FALLING, callback=my_callback)

i = 0
while True:
    i = i + 1
    print(i)
    time.sleep(1)
```

Pruebe a ejecutar el programa con privilegios de superusuario. Debería ver algo como esto cuando presione el botón:

```
$ sudo python interrupts.py
1
2
3
You pressed the button
4
You pressed the button
5
You pressed the button
You pressed the button
6
```

Observaciones

Podría detectar cuándo se ha presionado un botón o cuándo una entrada GPIO ha cambiado simplemente comprobando repetidamente un bucle. Por ejemplo:

```
while True:
    if GPIO.input(18) == False:
        # ponga el código para que sea accionado aquí
    time.sleep(0.1)
```

La desventaja aquí es que no puede hacer mucho más mientras está revisando las presiones del botón. Otra desventaja es que, si la pulsación del botón es muy rápida, podría ir y venir antes de que la pueda registrar con `GPIO.input`. Este enfoque se llama *polling*.

Las interrupciones funcionan de manera diferente. Le permiten asociar una función con uno de los pines de modo que, cuando el voltaje en la entrada cambie de bajo a alto o viceversa, podrá activar la función que se va a ejecutar.

Puede ver cómo funciona esto en el programa de ejemplo anterior. Primero, defina una función llamada `my_callback` que tome un único argumento. Este argumento

especifica la entrada que activó la interrupción, lo que le permite utilizar la misma función de controlador para una serie de interrupciones.

```
def my_callback(channel):
    print('You pressed the button')
```

En este caso, la función de devolución solo muestra un mensaje.

La línea de código que realiza la vinculación real es:

```
GPIO.add_event_detect(18, GPIO.FALLING, callback=my_callback)
```

El primer parámetro especifica el pin (18). El segundo puede ser `GPIO.FALLING` o `GPIO.RISING`. Si se ajusta `FALLING`, la función solo llamará si el pin GPIO pasa de alto a bajo. Este es el caso de este ejemplo, ya que el interruptor tira de la entrada baja contra el resistor interno pull-up. Si, por otro lado, el segundo argumento se ajusta como `RISING`, la función solo llamará cuando la entrada pase de baja a alta (cuando se suelte el interruptor).

La función de controlador de eventos no detiene el bucle contador principal mientras se ejecuta; en realidad se ejecuta en su propio hilo de ejecución.

Los interruptores suelen *rebotar* cuando se presionan. Esto significa que no siempre hace la transición limpia de abierto a cerrado, sino que rebotan entre los dos, posiblemente varias veces, lo que puede parecer que el botón se haya pulsado varias veces de manera rápida cuando en realidad solo se ha pulsado una vez.

Si sigue presionando el botón, probablemente verá esto reflejado en la salida como el mensaje que aparece cuando se pulsa el botón.

La biblioteca tiene una opción para detener el rebote, de ser este un problema, evitando que la interrupción se active de nuevo dentro de un cierto periodo de tiempo. Para hacer uso de esta función, simplemente agregue el parámetro extra opcional `bouncetime` al final de la llamada `add_event_detect`. El valor de `bouncetime` es en milisegundos.

```
GPIO.add_event_detect(18, GPIO.FALLING, callback=my_callback, bouncetime=100)
```

Para saber más

Para obtener más información sobre el uso de pulsadores con Raspberry Pi consulte el Capítulo 12.2.

CAPÍTULO 11

Motores

11.1 Introducción

En este capítulo se investiga el uso de diferentes tipos de motores con Raspberry Pi.

11.2 Controlar servomotores

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Usted desea utilizar Raspberry Pi para controlar la posición de un servomotor.

Solución

Utilice PWM para controlarle el ancho de los pulsos a un servomotor para cambiar su ángulo. Aunque esto funcionará, el PWM generado no es completamente estable, por lo que habrá un poco de inestabilidad con el servo. Para obtener una solución alternativa que produzca una sincronización de pulsos más estables utilice el *software* controlador ServoBlaster (consulte el Capítulo 11.3).

Si tiene una Raspberry Pi 1 más antigua, debe alimentar el servo con una fuente de alimentación separada de 5 V porque los picos en la corriente de carga son muy propensos a colapsar o sobrecargar la Raspberry Pi. Si tiene una Raspberry Pi B+ o una más reciente, las mejoras en la regulación de voltaje harán que pueda conseguir alimentar servos pequeños directamente del pin 5 V en el puerto GPIO.

La Figura 11-1 muestra un pequeño servo de 9 g (vea "Varios" en la página 477) que funciona bastante bien con la Raspberry Pi B+.

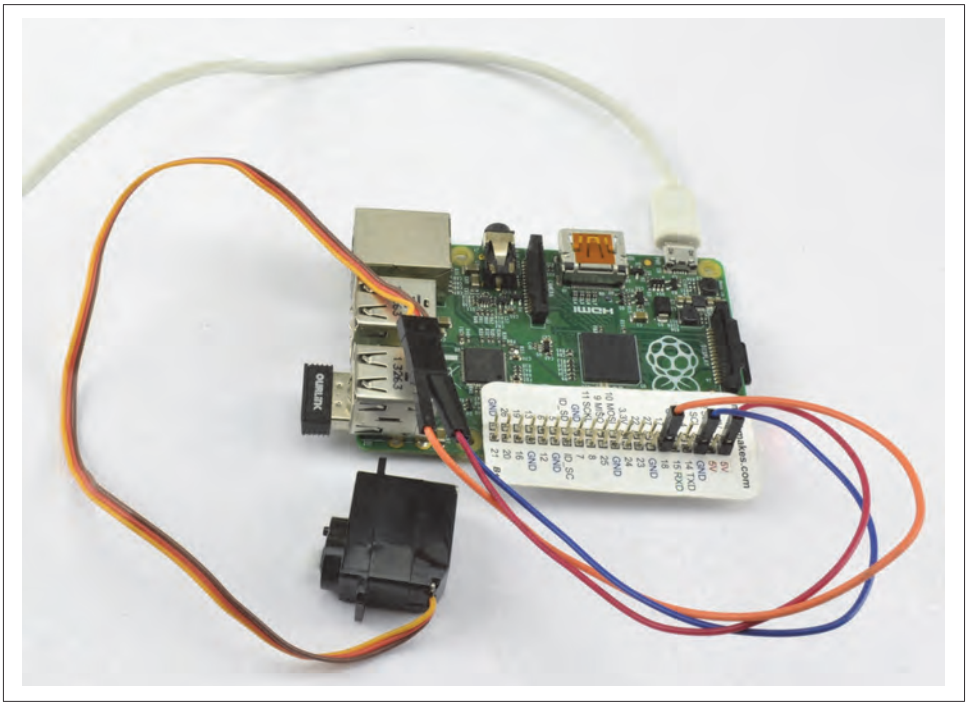


Figura 11.1. *Conexión directa de un pequeño servo a la Raspberry Pi B+.*

Los cables del servo suelen ser de 5 V de alambre como rojo, el de tierra marrón y el de control de color naranja. Los cables de 5 V y de tierra están conectados al pin del encabezado GPIO de 5 V y al GND, y el cable de control está conectado al pin 18. Las conexiones están hechas con cables de encabezado de hembra a macho.

Si está utilizando una fuente de alimentación separada, una placa de pruebas será la mejor manera de mantener todos los cables juntos.

En este caso necesitará:

- Servomotor de 5 V (vea "Varios" en la página 477)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 1 kΩ (vea "Resistores y condensadores" en la página 474)
- Fuente de alimentación de 5 V 1 A o batería de 4,8 V (vea "Varios" en la página 477)

El diseño de la placa de pruebas de esto se muestra en la Figura 11.2.

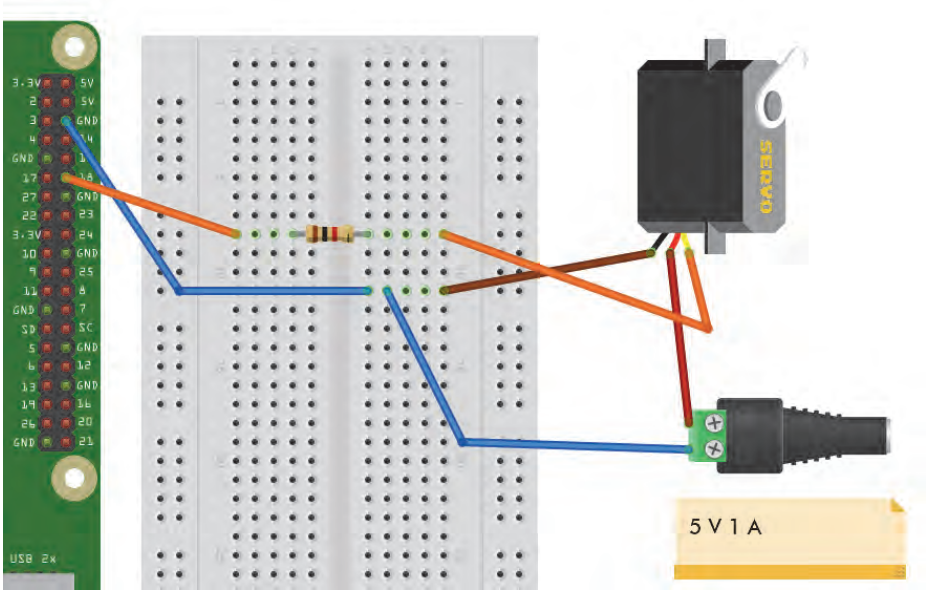


Figura 11.2. *Control de un servomotor.*

El resistor de 1 kΩ no es esencial, pero protege el pin GPIO de corrientes inesperadamente altas en la señal de control, cosa que podría ocurrir si se produjera un fallo en el servo.

Si lo prefiere, puede alimentar el servo con una batería en lugar de con una fuente de alimentación. El uso de un soporte de batería de cuatro pilas AA con pilas recargables proporcionará alrededor de 4,8 V y funcionará bien con un servo. El uso de cuatro pilas alcalinas AA para proporcionar 6 V irá bien con muchos servos, pero compruebe la hoja de datos de su servo para asegurarse de que funcionará correctamente con 6 V.

La interfaz de usuario para ajustar el ángulo del servo se basa en el programa *gui_slider.py*, destinado a controlar el brillo de un led (Capítulo 10.10). Sin embargo, puede modificarlo para que el deslizador establezca el ángulo entre 0 y 180 grados (Figura 11.3).

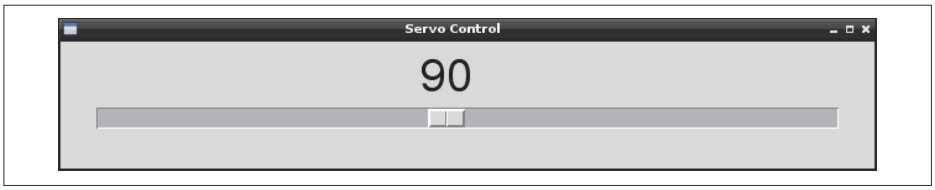


Figura 11.3. *Interfaz de usuario para controlar un servomotor.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *servo.py*.

Tenga en cuenta que este programa utiliza una interfaz gráfica de usuario, por lo que no se puede ejecutar desde SSH.

Debe ejecutarlo desde el entorno de ventanas en la propia Pi o a través del control remoto usando VNC (Capítulo 2.9) o RDP (Capítulo 2.10). También es necesario ejecutarlo como superusuario, así que ejecútelo con el comando `sudo python servo.py`:

```
from Tkinter import *
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
pwm = GPIO.PWM(18, 100)
pwm.start(5)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        scale = Scale(frame, from_=0, to=180,
              orient=HORIZONTAL, command=self.update)
        scale.grid(row=0)


    def update(self, angle):
        duty = float(angle) / 10.0 + 2.5
        pwm.ChangeDutyCycle(duty)

root = Tk()
root.wm_title('Servo Control')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

Observaciones

Los servomotores se utilizan en vehículos de control remoto y de robótica. La mayoría de los servomotores no son *continuos*, es decir, no pueden girar todo el recorrido, sino solo un ángulo de unos 180 grados.

La posición del servomotor se ajusta por la longitud del pulso. El servo espera a recibir un pulso por lo menos cada 20 milisegundos. Si ese pulso es alto durante 1 milisegundo, el ángulo del servo será cero; si es 1,5 milisegundos, estará en la posición central; y si son 2 milisegundos, estará a 180 grados (Figura 11.4).

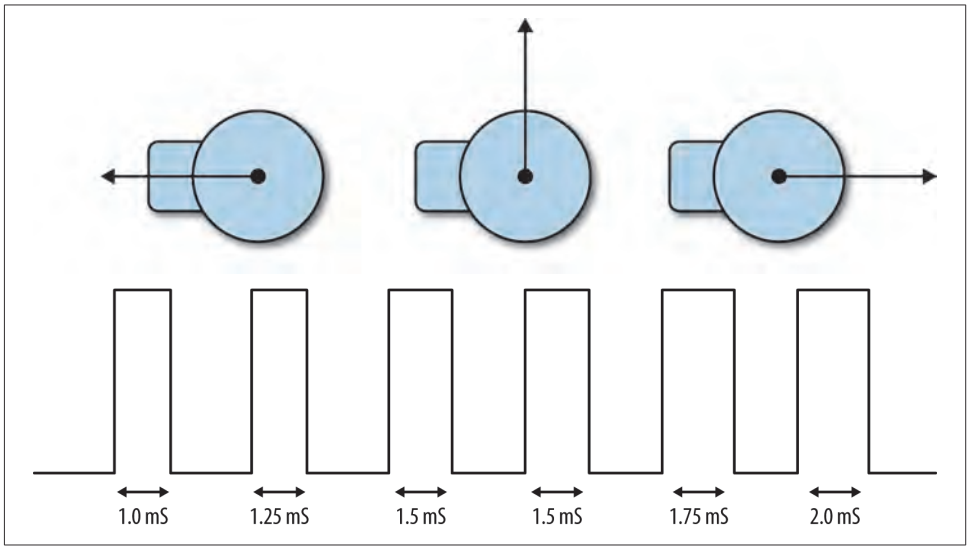


Figura 11.4. *Servomotores.*

El programa de ejemplo establece la frecuencia PWM a 100 Hz, que enviará un pulso al servo cada 10 milisegundos. El ángulo se convierte en un ciclo de trabajo entre 0 y 100.

Para saber más

Si tiene muchos servos para controlar, o necesita una mayor estabilidad y precisión, puede utilizar un módulo controlador de servos, como se describe en el Capítulo 11.4.

Adafruit ha desarrollado otro método para controlar servos.

Para obtener una solución alternativa que produzca una sincronización de pulsos mucho más estable utilizando el software controlador ServoBlaster, vaya al Capítulo 11.3.

11.3 Controlar servomotores de manera precisa

Problema

La función PWM de la biblioteca `RPi.GPIO` no es precisa ni libre de temblores para su aplicación en un servo.

Solución

Instale el controlador del dispositivo Servo Blaster.

El software Servo Blaster creado por Richard Hurst utiliza el hardware de la CPU de Raspberry Pi para generar pulsos con tiempos mucho más precisos que los que se pueden generar con la biblioteca `RPi.GPIO`. Instale esto usando los siguientes comandos y luego reinicie su Raspberry Pi.

```
git clone git://github.com/richardghirst/PiBits.git
cd PiBits/ServoBlaster/user
sudo make
sudo make install
```

El programa del Capítulo 11.2 se puede modificar para usar el código Servo Blaster. El programa modificado se puede encontrar en el archivo *servo_blaster.py* y asume que el pin controlador de servos está conectado a GPIO 18.

```
from Tkinter import *
import os
import time

servo_min = 500  # uS
servo_max = 2500  # uS

servo = 2 # GPIO 18

def map(value, from_low, from_high, to_low, to_high):
  from_range = from_high - from_low
  to_range = to_high - to_low
  scale_factor = float(from_range) / float(to_range)
  return to_low + (value / scale_factor)


def set_angle(angle):
  pulse = int(map(angle, 0, 180, servo_min, servo_max))
  command = "echo {}={}us > /dev/servoblaster".format(servo, pulse)
  os.system(command)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        scale = Scale(frame, from_=0, to=180,
              orient=HORIZONTAL, command=self.update)
        scale.grid(row=0)

    def update(self, angle):
        set_angle(float(angle))

root = Tk()
root.wm_title('Servo Control')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

El código de la interfaz de usuario es el mismo que el del Capítulo 11.2. Las diferencias están en la función `set_angle`. La función `set_angle` primero utiliza una función de utilidad llamada *map* que convierte el ángulo en una duración de pulso utilizando las constantes `servo_min` y `servo_max`. Después, se crea un comando que se ejecutará como si fuese desde la línea de comandos. El formato de esta línea comienza con el comando *echo*, seguido por el número de servo que se va a controlar, luego un signo igual y luego la duración del pulso en microsegundos. Esta parte de la cadena del comando estará dirigido al dispositivo */dev/servoblaster*. El servo entonces ajustará su ángulo.

Cuando ServoBlaster, o más específicamente, el servicio *servo.d*, esté ejecutándose, no podrá utilizar los pines del servo para nada y el audio no funcionará en Raspberry Pi. Así que cuando necesite usar los pines para otra cosa, use los siguientes comandos para desactivar el ServoBlaster y luego reinicie Pi.

```
$ sudo update-rc.d servoblaster disable
$ sudo reboot
```

Cuando su Raspberry Pi se reinicie, ServoBlaster no tendrá control sobre sus pines. Siempre puede volver a activar ServoBlaster utilizando:

```
$ sudo update-rc.d servoblaster enable
$ sudo reboot
```

Discusión

El controlador ServoBlaster es realmente muy potente y puede ser configurado para que pueda usar casi todos los pines GPIO para controlar servos. Su configuración predeterminada define ocho pines GPIO que actúan como pines de control de servo. Cada uno de ellos tiene un número de canal, como se ve en la Tabla 11.1.

Tabla 11.1. *Asignación predeterminada de los pines y los canales del servo para ServoBlaster.*

Canales servo	Pines GPIO
0	4
1	17
2	18
3	27
4	22
5	23
6	24
7	25

La conexión de tantos servos puede dar lugar a líos con los cables puente. Una placa como MonkMakes Servo Six simplifica enormemente el cableado de los servos en su Raspberry Pi.

Para saber más

Puede encontrar la documentación completa de Servo Blaster en *https://github.com/richard- ghirst/PiBits/tree/master/ServoBlaster*.

Si no necesita una sincronización exacta de Servo Blaster, la biblioteca `RPi.GPIO` también puede generar pulsos para su servo, como se describe en el Capítulo 11.2.

11.4 Controlar varios servomotores

Problema

Necesita controlar una gran cantidad de servos con alta precisión.

Solución

Aunque el código Servo Blaster (revise el Capítulo 11.3) le permite controlar muchos servos con precisión, no resuelve el problema del cableado de todos esos servos. Puede hacerlo con la placa de pruebas, pero será un desorden y los cables seguramente se suelten.

Para facilitar la conexión utilice un HAT servomotor como el que se muestra en la Figura 11.5.

El HAT servomotor le permite controlar hasta 16 servos o canales PWM usando la interfaz I2C de Raspberry Pi. Los servos se conectan directamente al HAT.

La alimentación se suministra a los circuitos lógicos del módulo desde la conexión de 3,3 V de Raspberry Pi. Es totalmente independiente de la fuente de alimentación para los servomotores, que proviene de un adaptador de alimentación externo de 5 V.

Si lo prefiere, puede alimentar los servos con una batería en lugar de con una fuente de alimentación. El uso de un soporte de batería de cuatro pilas AA con pilas recargables proporcionará alrededor de 4,8 V y funcionará correctamente con la mayoría de servos. El uso de cuatro pilas alcalinas AA para proporcionar 6 V irá bien con muchos servos, pero compruebe la hoja de datos de su servo para asegurarse.

Los cabezales de los pines para conectar los servos están convenientemente dispuestos para que el cable servo encaje directamente en los pines. Asegúrese de ponerlos de manera correcta.

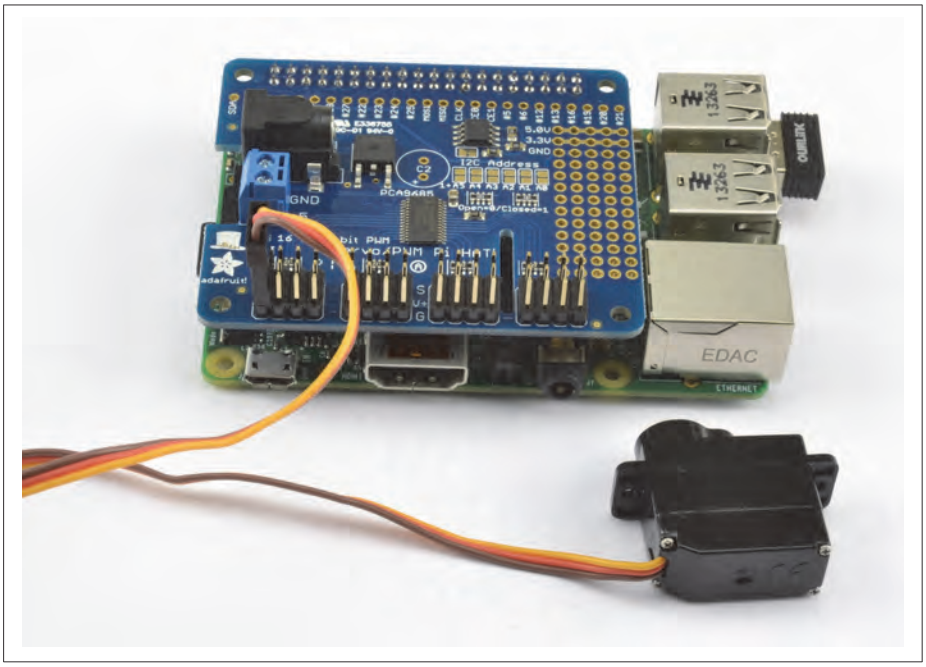


Figura 11.5. *Servomotor Adafruit HAT.*

Para usar el software de Adafruit para este módulo necesitará configurar el I2C en Raspberry Pi (Capítulo 9.4).

La biblioteca de Adafruit no es realmente una biblioteca adecuada que contenga un script de instalación, sino más bien es un directorio que contiene una serie de archivos. Por lo tanto, al usarlo debe estar en el directorio en el que se descargan o su programa no los encontrará.

Para descargar todas las bibliotecas del software de Adafruit para Raspberry Pi, escriba lo siguiente:

```
$ git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_PWM_Servo_Driver
```

Las dos últimas líneas configuran el directorio actual en el directorio que contiene el código para PWM, así como un programa de ejemplo proporcionado por Adafruit, que se puede ejecutar con el comando:

```
$ sudo python Servo_Example.py
```

Un ejemplo alternativo modifica el ejemplo del programa del Capítulo 10.3 para que pueda utilizar un deslizador para establecer la posición del servo entre 0 y 180 grados. El archivo de programa debe guardarse en el directorio *Adafruit_PWM_Servo_Driver*.

El deslizador cambiará las posiciones del servo de ambos canales 0 y 1 al mismo tiempo, de manera que ambos servos seguirán la posición del control deslizante.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *servo_module.py*. Tenga en cuenta que este programa utiliza una interfaz gráfica de usuario, por lo que no se puede ejecutar desde SSH. Debe ejecutarlo desde el entorno de ventana en la propia Pi o mediante control remoto usando VNC (Capítulo 2.9) o RDP (Capítulo 2.10).

```
from Tkinter import *
from Adafruit_PWM_Servo_Driver import PWM
import time

pwm = PWM(0x40)
pwm.setPWMFreq(50)

class App:

    def init (self, master):
        frame = Frame(master)
        frame.pack()
        scale = Scale(frame, from_=0, to=180,
              orient=HORIZONTAL, command=self.update)
        scale.grid(row=0)


    def update(self, angle):
        pulse_len = int(float(angle) * 500.0 / 180.0) + 110
        pwm.setPWM(0, 0, pulse_len)
        pwm.setPWM(1, 0, pulse_len)

root = Tk()
root.wm_title('Servo Control')
app = App(root)
root.geometry("200x50+0+0")
root.mainloop()
```

Observaciones

La primera línea después de la importación crea una nueva instancia de PWM utilizando la dirección I2C especificada como argumento; en este caso, 0×40. El módulo tiene conexiones que le permiten cambiar la dirección de I2C si esto entra en conflicto con otro dispositivo I2C que esté utilizando o si desea utilizar más de uno de estos módulos.

La siguiente línea establece la frecuencia PWM a 50 Hz, lo que proporcionará un pulso de actualización cada 20 milisegundos.

La línea que realmente establece el PWM para un canal en particular es:

```
pwm.setPWM(0, 0, pulse_len)
```

El primer argumento es el canal PWM cuyo ciclo de trabajo debe ser cambiado. Cada ciclo de PWM se divide en 4.096 marcas, y el segundo argumento es la marca en la que debe empezar el pulso. Este siempre será 0. El tercer argumento es la marca en la que el pulso debe terminar. Las constantes de `500,0` y `110` en la siguiente línea fueron ajustadas con un poco de ensayo y error para proporcionar un servo estándar lo más cerca posible de los 180 grados de movimiento:

```
pulse_len = int(float(angle) * 500.0 / 180.0) + 110
```

Al seleccionar una fuente de alimentación para este módulo, recuerde que un servo de control remoto estándar puede fácilmente necesitar 400 mA mientras se mueve, y más si está bajo carga. Así que, si usted planea tener muchos servos grandes moviéndose a la vez, necesitará un adaptador de corriente grande.

Para saber más

Un servo HAT será genial si su Raspberry Pi está cerca de los servomotores. Pero si sus servos están lejos de donde sea que esté Raspberry Pi, existe un módulo de servo, que vende Adafruit, (producto ID 815) que tiene el mismo *hardware* controlador de servos que el servo HAT, pero solo tiene cuatro pines para conectar la interfaz I2C a la placa de la interfaz I2C de Raspberry Pi.

Consulte la documentación de Adafruit para su biblioteca Raspberry Pi.

11.5 Controlar la velocidad de un motor CC

Problema

Desea controlar la velocidad de un motor CC usando su Raspberry Pi.

Solución

Puede utilizar el mismo diseño que en el Capítulo 10.6. Sin embargo, es buena idea colocar un diodo a través del motor para evitar que los picos de tensión dañen el transistor o incluso Raspberry Pi. 1N4001 es un diodo adecuado para esto. El diodo tiene una raya en un extremo, así que asegúrese de que lo esté colocando de manera correcta.

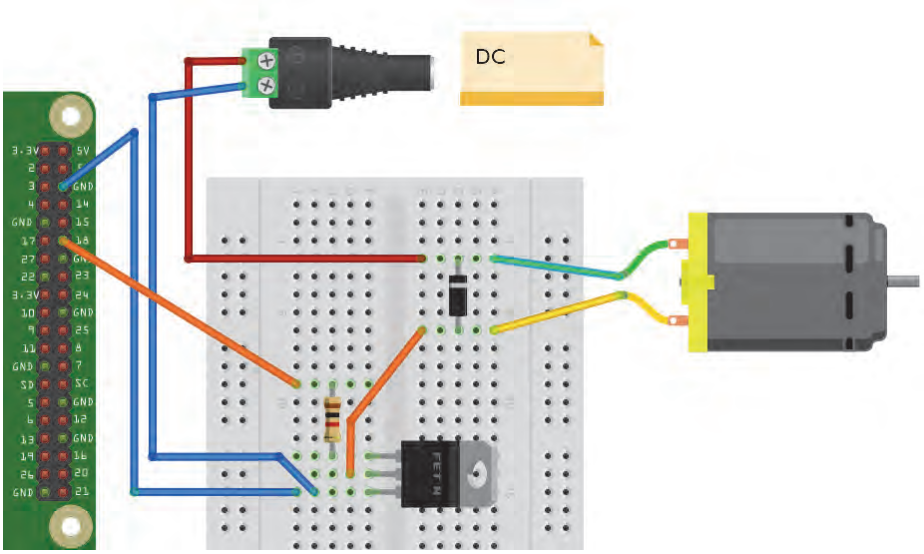


Figura 11.6. *Control de un motor de alta potencia.*

Necesitará:

- Motor CC de 3 V a 12 V
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 1 kΩ (vea "Resistores y condensadores", página 474)
- Transistor MOSFET FQP30N06L (vea "Transistores y diodos", página 475)
- Diodo 1N4001 (vea "Transistores y diodos", página 475)
- Fuente de alimentación con voltaje para que coincida con el motor

Para controlar la velocidad del motor, puede descargar el programa de la sección de código en la página web del libro, donde pone *gui_slider.py*. Tenga en cuenta que este programa utiliza una interfaz gráfica de usuario, por lo que no se puede ejecutar desde SSH. Debe ejecutarlo desde el entorno de ventana en el propio Pi o a través del control remoto mediante VNC (Capítulo 2.9) o RDP (Capítulo 2.10).

Observaciones

Si solo está utilizando un motor CC de baja potencia (menos de 200 mA), puede utilizar un transistor más pequeño (y más barato), como 2N3904 (vea "Transistores y diodos" en la página 475). En la Figura 11.7 se muestra el diseño de la placa de pruebas usando un 2N3904.

Probablemente puede funcionar bien con la alimentación de un pequeño motor desde la línea de suministro de 5 V en el conector GPIO. Si ve que Raspberry Pi se bloquea, utilice una fuente de alimentación externa, como se muestra en la Figura 11.6.

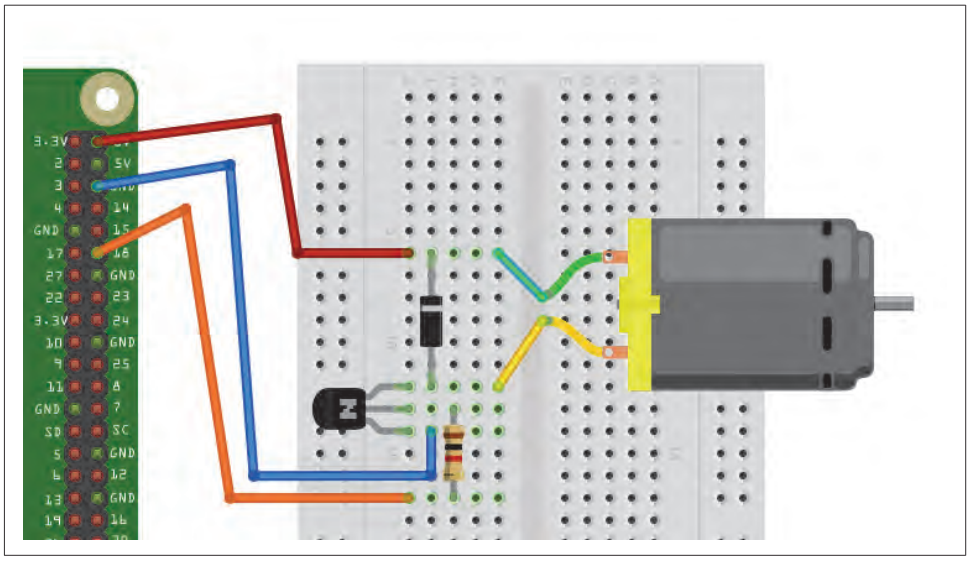


Figura 11.7. *Control de un motor de baja potencia.*

Para saber más

Este diseño solo controla la velocidad del motor. No puede controlar su dirección. Para ello necesita ver el Capítulo 11.6.

Para obtener más información sobre el uso de la placa de pruebas y los cables puente con Raspberry Pi vea el Capítulo 9.9.

11.6 Controlar la dirección de un motor CC

Problema

Desea controlar tanto la velocidad como la dirección de un pequeño motor CC.

Solución

Utilice un chip o un módulo H-Bridge.

Tiene dos fórmulas para controlar un motor. La primera, el "enfoque bricolaje", utiliza una placa sin soldadura y un chip L293D. El segundo diseño utiliza un módulo H-Bridge ya preparado de SparkFun conectado directamente a Raspberry Pi con cables puente.

Tanto el L293D como el módulo SparkFun son capaces de accionar dos motores sin ningún tipo de hardware adicional. En el apartado «Observaciones» también se mencionan otras opciones para controlar los motores de corriente continua.

Opción 1: chip L293D y placa de pruebas

Si toma la opción del L293D, necesitará:

- Motor CC de 3 V a 12 V
- Placa de pruebas y cables puente (macho a hembra; vea "Equipos para prototipos", página 474)
- Chip L293D (vea "Circuitos integrados" en la página 475)
- Fuente de alimentación con voltaje para que coincida con el motor

El diseño de la placa de pruebas se muestra en la Figura 11.8.

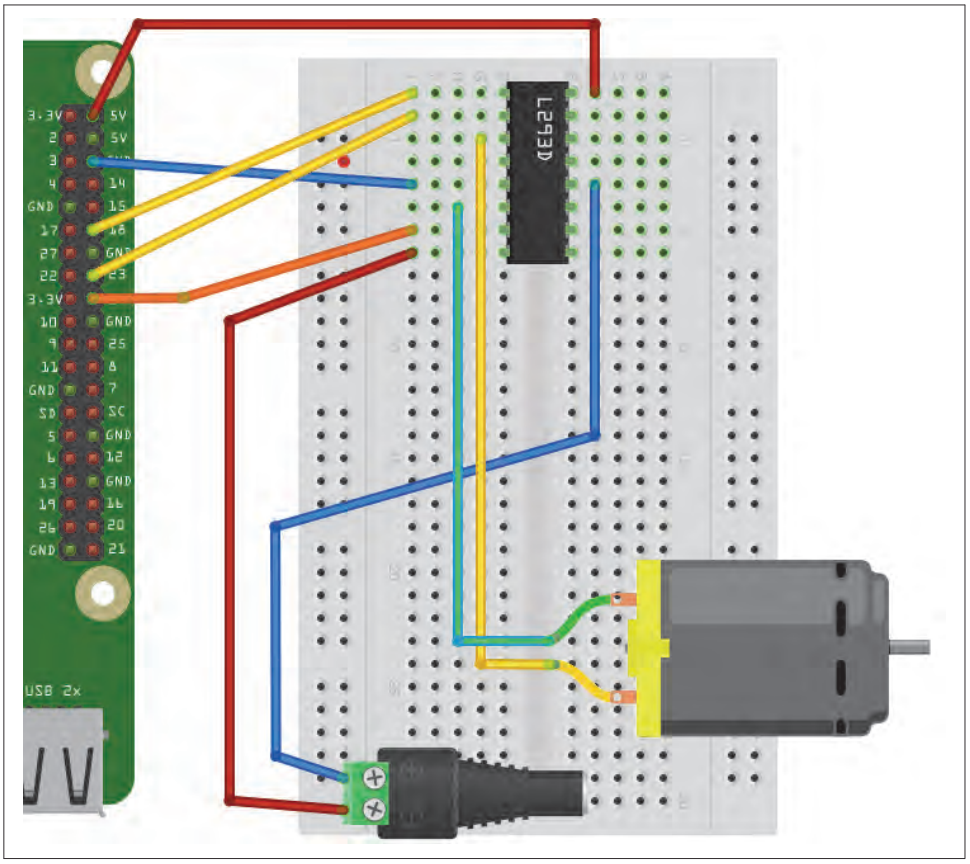


Figura 11.8. *Usar un chip L293D para controlar un motor.*

Asegúrese de que el chip está orientado para el lado correcto. Hay una muesca en la parte superior, que es el final, que debe estar en la parte superior de la placa de pruebas.

Opción 2: Módulo controlador del motor

Si decide utilizar el controlador del motor SparkFun, o un módulo controlador del motor similar, necesitará lo siguiente:

- Motor CC de 3 V a 12 V
- Cables puente (hembra a hembra; vea "Equipos para prototipos", página 474)
- Pines de encabezado (vea "Circuitos integrados", página 475)
- Controlador del motor SparkFun 1 A dual (consulte "Módulos", página 476)
- Fuente de alimentación con voltaje para que coincida con el motor

El diseño se muestra en la Figura 11.9. Tenga en cuenta que el motor mostrado es un motor de corriente continua. Si el proyecto está impulsado por ruedas, se debería usar un *motorreductor*, lo que combina un motor y una caja de cambios para reducir las r.p.m. y aumentar el torque.

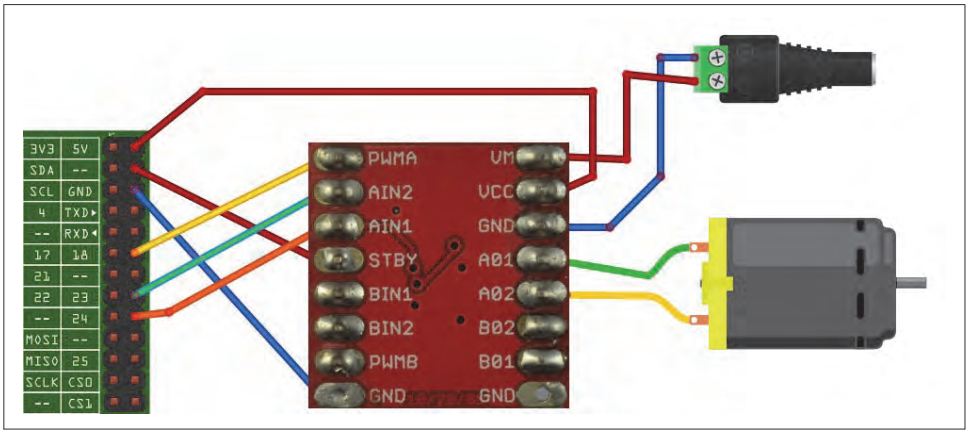


Figura 11.9. *Cableado de un controlador de motor SparkFun.*

El módulo controlador del motor viene sin pines de encabezado adjuntos, por lo que tendrá que añadirlos a la placa antes de comenzar. Las conexiones se realizan usando encabezados hembra a hembra.

La Figura 11.10 muestra el módulo cableado que va hasta un pequeño motorreductor de CC, según el diagrama de cableado de la Figura 11.9.

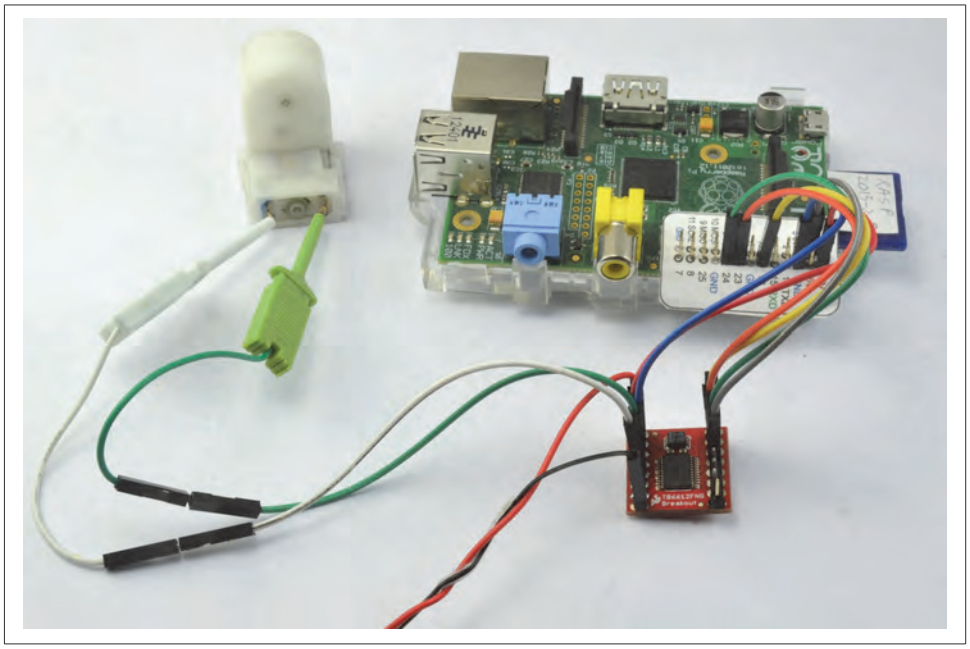


Figura 11.10. *Usar el controlador de motor SparkFun con un motorreductor.*

Software

Sea cual sea la opción de *hardware* que haya decidido, puede utilizar el mismo programa para probar el motor. Esto le permite introducir una letra `f` o `r` y después solo un dígito entre 0 y 9. El motor avanzará o retrocederá a una velocidad especificada por el dígito: 0 para parar y 9 para la velocidad máxima.

```
$ sudo python motor_control.py
Command, f/r 0..9, E.g. f5 :f5
Command, f/r 0..9, E.g. f5 :f1
Command, f/r 0..9, E.g. f5 :f2
Command, f/r 0..9, E.g. f5 :r2
```

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *motor_control.py*. Este programa utiliza la línea de comandos, por lo que se puede ejecutar desde SSH.

Si está utilizando Python 3, cambie el comando `raw_input` a solamente `input`:

```
import RPi.GPIO as GPIO
import time

enable_pin = 18
```

```
in1_pin = 23
in2_pin =24

GPIO.setmode(GPIO.BCM)
GPIO.setup(enable_pin, GPIO.OUT)
GPIO.setup(in1_pin, GPIO.OUT)
GPIO.setup(in2_pin, GPIO.OUT)

pwm = GPIO.PWM(enable_pin, 500)
pwm.start(0)

def clockwise():
    GPIO.output(in1_pin, True)
    GPIO.output(in2_pin, False)

def counter_clockwise():
    GPIO.output(in1_pin, False)
    GPIO.output(in2_pin, True)

while True:
    cmd = raw_input("Command, f/r 0..9, E.g. f5 :")
    direction = cmd[0]
    if direction == "f":
        clockwise()
    else:
        counter_clockwise()
    speed = int(cmd[1]) * 10
    pwm.ChangeDutyCycle(speed)
```

Observaciones

Antes de ver cómo funciona el programa necesita entender un poco más acerca de cómo funciona un H-Bridge.

La Figura 11.11 muestra cómo se trabaja utilizando conmutadores en lugar de transistores o un chip. Al invertir la polaridad a través del motor, H-Bridge también invierte la dirección en la que gira el motor.

En la Figura 11.11, S1 y S4 están cerrados y S2 y S3 están abiertos. Esto permite que la corriente fluya a través del motor, siendo el terminal A positivo y el terminal B negativo. Si volviésemos a invertir los conmutadores para que S2 y S3 estuvieran cerrados y S1 y S4 estuvieran abiertos, B sería positivo y A sería negativo, y el motor giraría en la dirección opuesta.

Sin embargo, es posible que haya detectado un peligro con este circuito. Si por alguna casualidad S1 y S2 están cerrados, la fuente positiva estará directamente conectada a la fuente negativa y habrá un cortocircuito. Lo mismo ocurre si S3 y S4 se cierran al mismo tiempo.

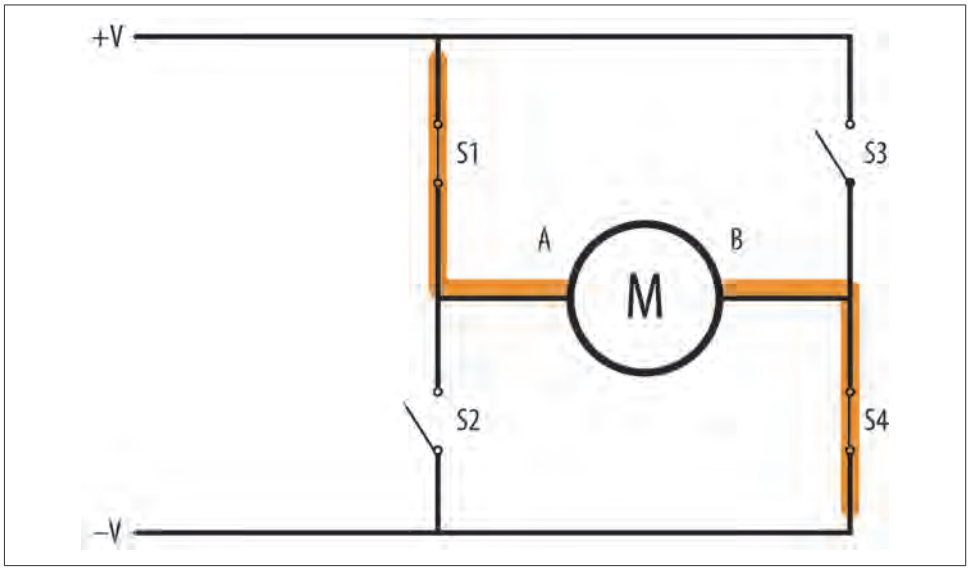


Figura 11.11. *H-Bridge.*

Aunque pueda utilizar transistores individuales para crear un H-Bridge, es más sencillo utilizar un H-Bridge IC como L293D. Este chip en realidad tiene dos H-Bridges en él, por lo que se puede utilizar para controlar dos motores.

L293 tiene tres pines de control para cada uno de los dos canales de control del motor. El pin `Enable` solo habilita o deshabilita el canal como un todo. En el programa de ejemplo se conecta a una salida PWM para controlar la velocidad del motor. Los otros dos pines (IN1 y IN2) controlan la dirección en la que se acciona el motor. Puede ver el uso de estos dos pines de control en las funciones `clockwise` y `counter_clock wise`:

```
def clockwise():
    GPIO.output(in1_pin, True)
    GPIO.output(in2_pin, False)

def counter_clockwise():
    GPIO.output(in1_pin, False)
    GPIO.output(in2_pin, True)
```

Si IN1 es alto e IN2 es bajo, el motor girará en una dirección. Si estos dos pines están invertidos, el motor girará en la dirección opuesta.

Como alternativa al uso de un L293D en una placa de pruebas, hay módulos de bajo coste disponibles en eBay que incluyen un L293D en un PCB con terminales de tornillo para conectar motores y encabezados directamente al conector GPIO de Raspberry Pi. Si necesita un módulo controlador de motor de mayor potencia, puede

encontrar uno que funcione con los mismos principios, pero con corrientes mucho más altas, hasta 20 A o más. Polulu tiene una gama impresionante de tales placas.

Para saber más

El motor paso a paso HAT de Adafruit (Capítulo 11.9) y la placa RaspiRobot (Capítulo 11.10) también se pueden usar para controlar la velocidad y la dirección de un motor de corriente continua.

Compruebe la hoja de datos de L293D y la página web del módulo controlador de motor SparkFun

Para obtener más información sobre el uso de la placa de pruebas y de los cables puente con Raspberry Pi, vea el Capítulo 9.9.

11.7 Utilizar motores paso a paso unipolares

Problema

Quiere accionar un motor paso a paso unipolar de cinco cables usando Raspberry Pi.

Solución

Utilice un chip controlador Darlington ULN2803 en la placa de pruebas.

Los motores paso a paso encajan entre los motores de corriente continua y los servomotores en el mundo de las tecnologías de motores. Pueden girar continuamente igual que un motor CC regular, pero también puede posicionarlos con mucha precisión moviéndolos paso a paso en cualquier dirección.

Para hacer esto necesitará:

- Motor paso a paso unipolar de cinco cables (vea "Varios", página 477)
- Controlador IC Darlington ULN2803 (vea "Circuitos integrados", página 475)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)

La Figura 11.12 muestra el diagrama de cableado para usar un ULN2803. Tenga en cuenta que el chip se puede utilizar para impulsar dos de estos motores. Para accionar un segundo motor paso a paso necesitará conectar cuatro pines más de control desde el conector GPIO a los pines 5, 6, 7 y 8 de ULN2803, y conectar los cuatros pines, del 11 al 14, del segundo motor de ULN2803.

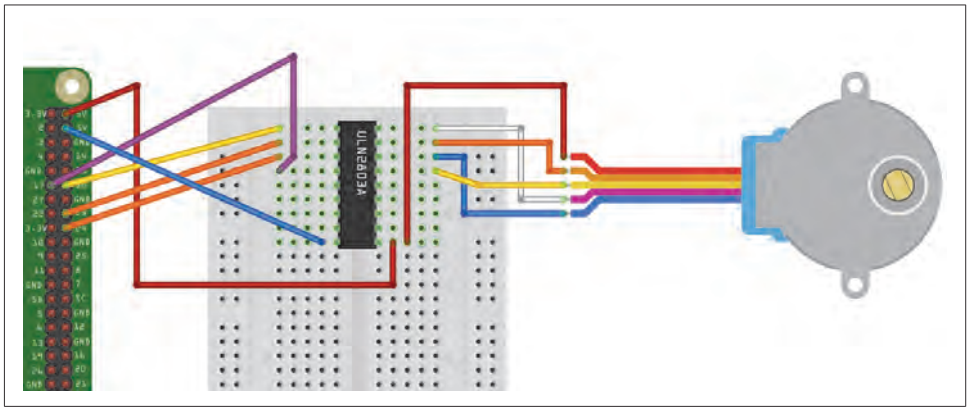


Figura 11.12. *Usar un ULN2803 para controlar un motor paso a paso unipolar.*

El suministro de 5 V del conector GPIO puede funcionar bien con un motor paso a paso pequeño. Si se le bloquea Raspberry Pi o necesita utilizar un motor paso a paso más grande, utilice un suministro separado para la alimentación del motor (pin 10 del ULN2803).

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas en este libro, puede descargar el programa de la sección de código de la página web del libro, donde pone *stepper.py*. Este programa utiliza la línea de comandos, por lo que se puede ejecutar desde SSH.

Si está utilizando Python 3, cambie el comando `raw_input` por `input`:

```
import RPi.GPIO as
GPIO import time

GPIO.setmode(GPIO.BCM)

coil_A_1_pin = 18
coil_A_2_pin = 23
coil_B_1_pin = 24
coil_B_2_pin = 17

GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
GPIO.setup(coil_B_1_pin, GPIO.OUT)
GPIO.setup(coil_B_2_pin, GPIO.OUT)

forward_seq = ['1010', '0110', '0101', '1001']
reverse_seq = list(forward_seq) # para copiar la lista
reverse_seq.reverse()

def forward(delay, steps):
 for i in range(steps):
```

```
    for step in forward_seq:
      set_step(step)
      time.sleep(delay)

def backwards(delay, steps):
  for i in range(steps):
    for step in reverse_seq:
      set_step(step)
      time.sleep(delay)


def set_step(step):
  GPIO.output(coil_A_1_pin, step[0] == '1')
  GPIO.output(coil_A_2_pin, step[1] == '1')
  GPIO.output(coil_B_1_pin, step[2] == '1')
  GPIO.output(coil_B_2_pin, step[3] == '1')

while True:
  set_step('0000')
  delay = raw_input("Delay between steps (milliseconds)?")
  steps = raw_input("How many steps forward? ")
  forward(int(delay) / 1000.0, int(steps)) set_step('0000')
  steps = raw_input("How many steps backwards? ")
  backwards(int(delay) / 1000.0, int(steps))
```

Al ejecutar el programa, se le pedirá un retraso entre los pasos. Este debe ser 2 o más. Después, se le pedirá el número de pasos en cada dirección:

```
$ sudo python stepper.py
Delay between steps (milliseconds)?2
How many steps forward? 100
How many steps backwards? 100
Delay between steps (milliseconds)?10
How many steps forward? 50
How many steps backwards? 50
Delay between steps (milliseconds)?
```

Observaciones

Los motores paso a paso utilizan un rotor dentado y electroimanes para empujar la rueda (Figura 11.13). Tenga en cuenta que los colores de los cables variarán.

La activación de las bobinas en un determinado orden impulsa el motor. El número de pasos que el motor paso a paso tiene en una rotación de 360 grados es en realidad el número de dientes del rotor.

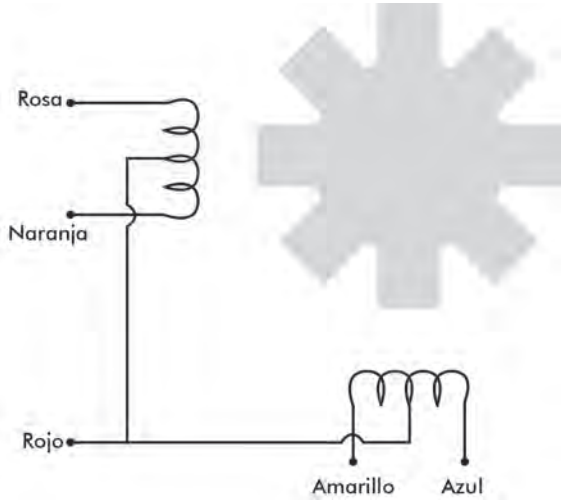


Figura 11.13. *Motor paso a paso.*

El programa de ejemplo utiliza una lista de cadenas para representar cada una de las cuatro etapas de la activación que forman un solo paso:

```
forward_seq = ['1010', '0110', '0101', '1001']
```

La secuencia para girar el motor en la dirección opuesta es justo lo inverso de la secuencia para avanzar.

Puede utilizar las funciones `forward` y `backward` en sus programas para mover el motor. El primer argumento para cualquier función es el retraso en milisegundos entre cada parte de la secuencia de pasos. El valor mínimo para esto depende del motor que utilice. Si es demasiado pequeño, el motor no girará. Normalmente, dos milisegundos o más estará bien. El segundo parámetro es el número de pasos a tomar.

```
def forward(delay, steps):
 for i in range(steps):
    for step in forward_seq:
      set_step(step)
      time.sleep(delay)
```

La función `forward` tiene dos bucles `for` anidados. El externo se repite para el número de pasos y el interno itera sobre la secuencia de activaciones del motor, llamando `setStep` para cada secuencia.

```
def set_step(step):
  GPIO.output(coil_A_1_pin, step[0] == '1')
  GPIO.output(coil_A_2_pin, step[1] == '1')
  GPIO.output(coil_B_1_pin, step[2] == '1')
  GPIO.output(coil_B_2_pin, step[3] == '1')
```

La función `set_step` establece cada uno de los pines de control en alto o bajo, dependiendo del patrón suministrado como argumento.

El bucle principal establece el paso a *0000* entre moverse hacia delante y hacia atrás, y pone todas las salidas a cero cuando el motor no está girando. De lo contrario, una de las bobinas se puede quedar encendida, haciendo que el motor consuma corriente innecesariamente.

Para saber más

Si tiene un motor paso a paso bipolar de cuatro cables, vaya al Capítulo 11.8.

Para obtener más información sobre los motores paso a paso, los diferentes tipos y cómo funcionan, vea Wikipedia, donde también encontrará una explicación animada del patrón de activación para arrancar el motor.

Para obtener información sobre el uso de servomotores, vaya el Capítulo 11.2; para el control de motores CC, revise los Capítulos 7.17 y 7.19.

Para obtener más información sobre el uso de una placa de pruebas y de los cables puente con Raspberry Pi, vea el Capítulo 9.9.

11.8 Utilizar motores paso a paso bipolares

Problema

Quiere arrancar un motor paso a paso bipolar de cuatro cables utilizando Raspberry Pi.

Solución

Utilice un chip de controlador H-Bridge L293D. H-Bridge es necesario para accionar un motor paso a paso bipolar porque, como la palabra *bipolar* indica, la dirección de la corriente a través de los bobinados necesita ser invertida, como conducir un motor CC en ambas direcciones (consulte el Capítulo 11.6).

Para hacerlo necesitará:

- 12 V, motor paso a paso bipolar de cuatro cables (vea "Varios", en la página 477)
- H-Bridge IC L293D (vea "Circuitos integrados", en la página 475)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)

El motor utilizado aquí de 12 V, es algo más grande que el ejemplo anterior del motor paso a paso unipolar. Por lo tanto, la potencia del propio motor se suministra a partir de una fuente de alimentación externa y no de Raspberry Pi. Vea el diagrama de cableado en la Figura 11.14.

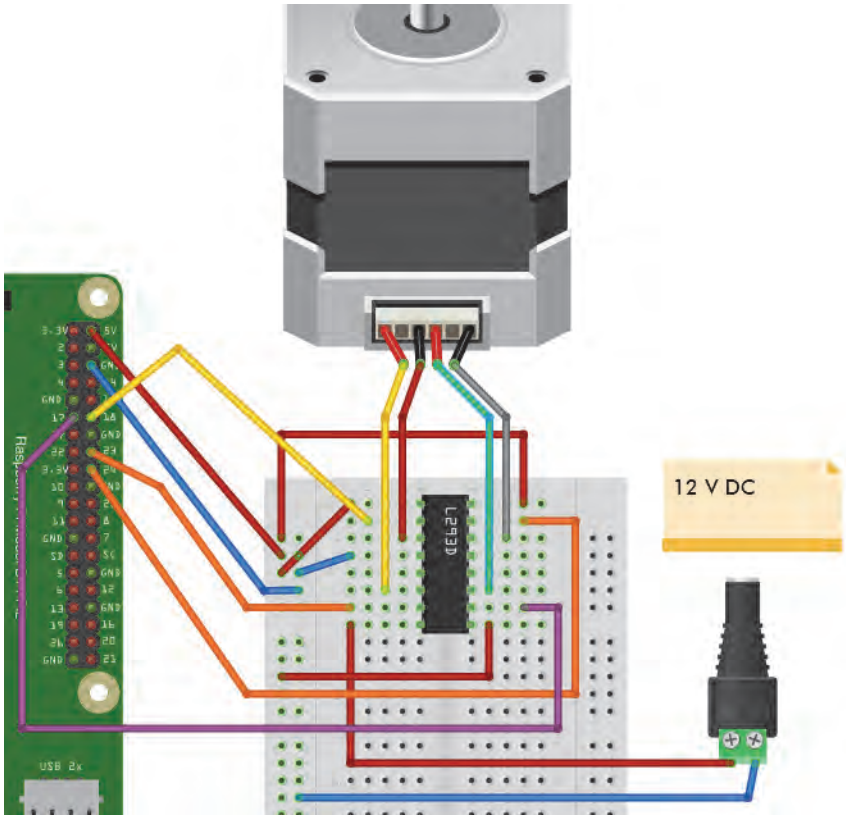


Figura 11.14. *Usar un L293D para controlar un motor paso a paso bipolar.*

Observaciones

Puede utilizar el mismo programa *stepper.py* para controlar este motor paso a paso (vea el Capítulo 11.7). El diseño utiliza H-Bridges L293D, por lo que necesita uno de estos chips para cada motor que desea controlar.

Para saber más

Si el tipo de motor paso a paso que tiene es un motor paso a paso unipolar de cinco cables, vea el Capítulo 11.7.

Para más información sobre los motores paso a paso —los distintos tipos y cómo funcionan— vea la Wikipedia, donde también encontrará una explicación animada del patrón de activación para arrancar el motor.

Para más información sobre el uso de servomotores, vea el Capítulo 11.2; y para controlar motores CC, vea los Capítulos 7.17 y 7.19.

Para obtener más información sobre el uso de una placa de pruebas y de los cables puente con Raspberry Pi, vea el Capítulo 9.9.

También puede manejar un motor paso a paso utilizando la tarjeta RasPiRobot (Capítulo 11.10).

11.9 Utilizar un HAT para accionar un motor paso a paso bipolar

Problema

Desea controlar varios motores paso a paso bipolares utilizando una tarjeta de interfaz.

Solución

Utilice un motor paso a paso HAT de Adafruit (Figura 11.15).

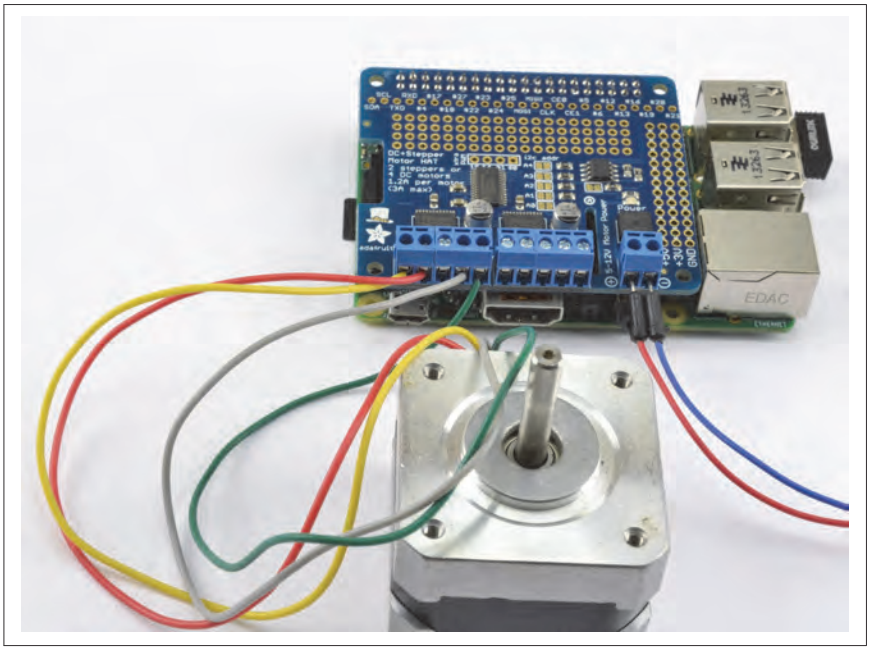


Figura 11.15. *Motor paso a paso HAT de Adafruit para controlar un motor paso a paso bipolar.*

Esta placa es capaz de accionar dos motores paso a paso bipolares. La Figura 11.15 muestra la placa con un motor paso a paso bipolar con una bobina conectada a los terminales M1 y la otra a los terminales M2. La potencia de los motores se suministra separadamente en los terminales de tornillo, a la derecha de la Figura 11.15.

Este HAT utiliza I2C, así que asegúrese de que tiene habilitado I2C (Capítulo 9.4).

Para facilitar el uso de HAT, Adafruit ha escrito una biblioteca de Python. Para instalarla introduzca los siguientes comandos:

```
$ git clone https://github.com/adafruit/Adafruit-Motor-HAT-Python-Library.git
$ sudo apt-get install python-dev
$ cd Adafruit-Motor-HAT-Python-Library/
$ sudo python setup.py install
```

La biblioteca incluye algunos ejemplos, así que cambie el directorio y luego ejecute el ejemplo básico usando los comandos:

```
$ cd examples
$ sudo python StepperTest.py
```

Buses I2C

Si siguió el Capítulo 9.20 para crear su propio HAT y activó el bus I2C 0 tal como se describe, deberá invertir el cambio en */boot/config.txt* porque el Adafruit detecta automáticamente el bus I2C y detectará un error si el bus 0 está habilitado.

En */boot/config.txt* quite el comentario de la línea:

```
dtparam=i2c_vc=on
```

Una vez hecho esto, reinicie Raspberry Pi.

Observaciones

Cuando ejecute el programa, el motor comenzará a girar y el programa hará un bucle de cuatro modos diferentes de pasos. Es posible que desee probar el siguiente sencillo ejemplo para controlar su motor paso a paso.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código de la página web del libro, donde pone *stepper_hat.py*. Este programa utiliza la línea de comandos, por lo que puede ejecutarlo desde SSH.

```
from Adafruit_MotorHAT import Adafruit_MotorHAT
import time

HAT = Adafruit_MotorHAT
stepper_hat = Adafruit_MotorHAT()

stepper = stepper_hat.getStepper(200, 1) # 200 pasos/rev, puerto 1 (M1, M2)

try:
    while True:
        speed = input("Enter stepper speed (rpm) ")
```

```
            stepper.setSpeed(speed)
            steps_forward = input("Steps forward ")
            stepper.step(steps_forward, HAT.FORWARD, HAT.SINGLE)
            steps_forward = input("Steps reverse ")
            stepper.step(steps_forward, HAT.BACKWARD, HAT.SINGLE)


    finally:
        print("cleaning up")
        stepper_hat.getMotor(1).run(HAT.RELEASE)
```

Después de importar el módulo `Adafruit_MotorHAT`, se le asigna el nombre HAT solo para reducir la verbosidad del código en las líneas que le siguen.

El bloque `try/finally` se utiliza para asegurarnos de que la potencia de las bobinas se libera cuando el programa se para utilizando Ctrl-C.

Para saber más

Para una discusión sobre estándar HAT y para saber cómo hacer el suyo propio, vea el Capítulo 9.21.

Para obtener más información sobre el uso de este HAT y su biblioteca adjunta, consulte *https://learn.adafruit.com/adafruit-dc-and-stepper-motor-hat-for-raspberry-pi/*.

Para usar un L293D para controlar un motor paso a paso, vea el Capítulo 11.8; y para una tarjeta RaspiRobot, vea el Capítulo 11.10.

11.10 Utilizar una tarjeta RaspiRobot para impulsar un motor paso a paso bipolar

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea controlar un motor paso a paso bipolar usando una fuente de alimentación externa para el motor y para Raspberry Pi.

Solución

Utilice una tarjeta RaspiRobot versión 3.

La tarjeta RaspiRobot utiliza la fuente de alimentación directamente desde sus terminales de tornillo como suministro al motor y regula esa misma fuente hasta 5 V para accionar Raspberry Pi. Así que, en este caso, la potencia de 12 V se suministrará tanto al motor paso a paso de 12 V como a Raspberry Pi.

Si está utilizando una versión anterior de la tarjeta RasPiRobot (versión 1 o 2), *no* conecte Raspberry Pi a través de la conexión USB al mismo tiempo que la alimenta a través de la tarjeta RasPiRobot.

No hay ningún problema en alimentarlo a través de los dos si utiliza la versión 3 de la tarjeta.

Conecte el motor paso a paso y la fuente de alimentación a la tarjeta RaspiRobot como se muestra en la Figura 11.16. Los colores de los cables para el motor paso a paso de 12 V de Adafruit, en orden de izquierda a derecha, son: amarillo, rojo, gris y verde.

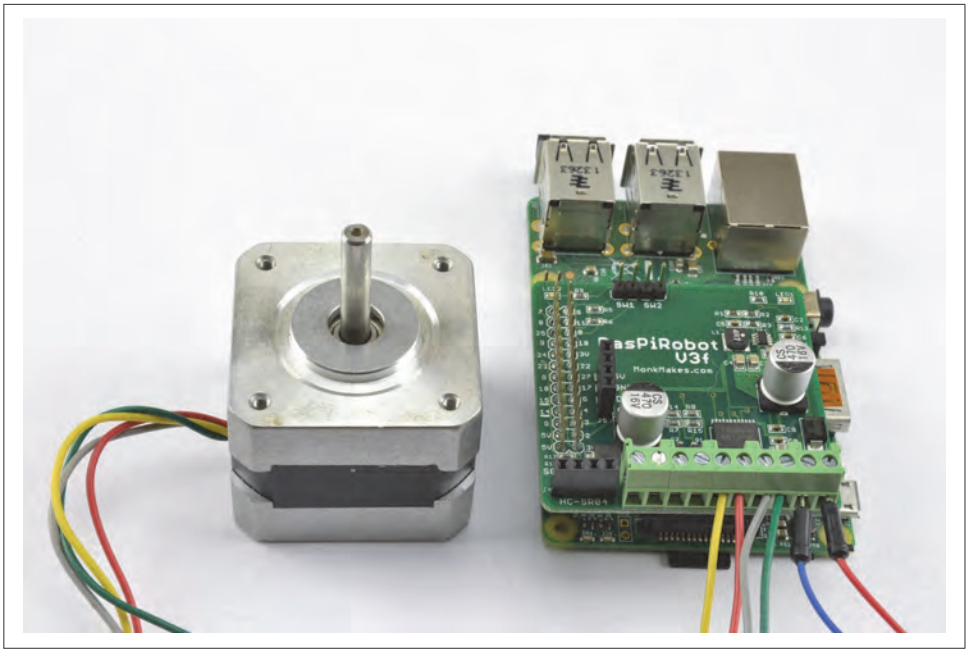


Figura 11.16. *Utilizar una tarjeta RaspiRobot para controlar un motor paso a paso bipolar.*

Antes de poder ejecutar el programa necesitará instalar la biblioteca para RaspiRobot V3 usando los siguientes comandos:

```
$ git clone https://github.com/simonmonk/raspirobotboard3.git
$ cd raspirobotboard3/python
$ sudo python setup.py install
```

Abra un editor (nano o IDLE) y pegue el siguiente código. También puede descargar el programa de la sección Código desde la página web del libro, donde pone *stepper_rrb.py*. Este programa utiliza la línea de comandos, por lo que se puede ejecutar desde SSH.

Si está utilizando Python 3, cambie el comando `raw_input` a `input`:

```
from rrb3 import *
import time

rr = RRB3(12.0, 12.0) # batería, motor

try:
    while True:
        delay = raw_input("Delay between steps (milliseconds)?")
        steps = raw_input("How many steps forward? ")
        rr.step_forward(int(delay) / 1000.0, int(steps))
        steps = raw_input("How many steps backwards? ")
        rr.step_reverse(int(delay) / 1000.0, int(steps))

finally:
    GPIO.cleanup()
```

Observaciones

Verá que hay un valor mínimo de `"Delay between steps"` (retraso entre pasos) por debajo del cual el motor solo vibrará, en vez de girar.

Para saber más

Puede encontrar documentación completa para la tarjeta RaspiRobot y otros proyectos que la utilizan en la página web de RaspiRobot.

Para accionar un motor paso a paso usando un L293D en una placa de pruebas, vea el Capítulo 11.8.

11.11 Crear un robot Rover sencillo

Problema

Desea utilizar una Raspberry Pi como impulsora de un robot Rover simple.

Solución

Utilice una tarjeta RaspiRobotV3 como tarjeta de interfaz para Raspberry Pi para controlar dos motores y un kit de chasis de robot, como Magician Chassis.

Para hacer esto necesitará:

- Tarjeta RaspiRobot V3 (vea "Módulos", en la página 476)
- Chasis del robot, como Magician Chassis (vea "Varios", página 477)
- Soporte de batería de 6 pilas AA (vea "Varios", página 477)
- Adaptador wifi USB

También puede comprar un kit completo de piezas para la construcción de este Rover como, por ejemplo, el kit MonkMakes RaspiRobot Rover.

El primer paso en la construcción del robot es montar el chasis. La mayoría de los chasis de motorreductores de bajo coste se suministrarán con un soporte de batería de cuatro pilas AA, pero para proporcionar energía al Raspberry Pi necesitará un soporte de batería de seis. Cuando llegue a esa parte de las instrucciones suministradas con el chasis, use una caja de 6 pilas AA.

Siga las instrucciones de cableado como se muestra en la Figura 11.17.

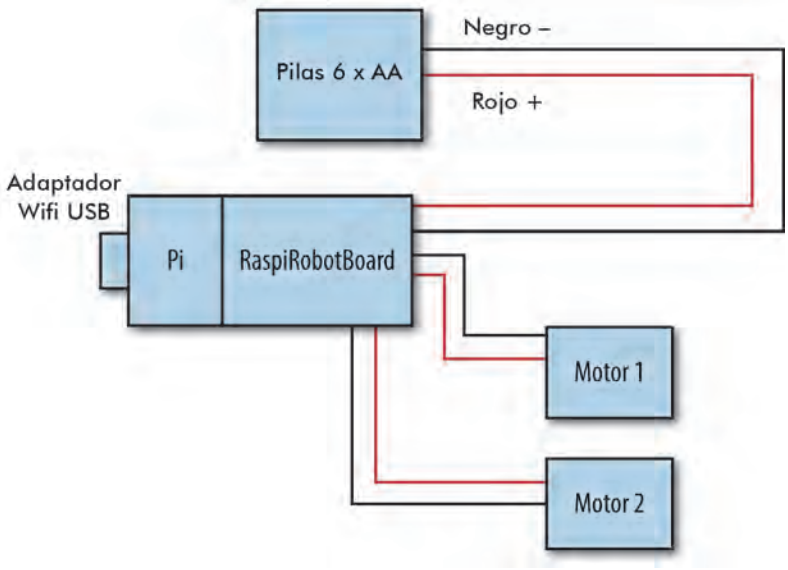


Figura 11.17. *Cableado de un robot Rover.*

El paquete de pilas suministrará energía a la tarjeta RaspiRobot, que a su vez suministrará 5 V a Raspberry Pi. Por lo tanto, solo se necesita una fuente de alimentación.

El Rover terminado debe parecerse al de la Figura 11.18, que también tiene un sensor de distancia por ultrasonidos delante para una buena medición.

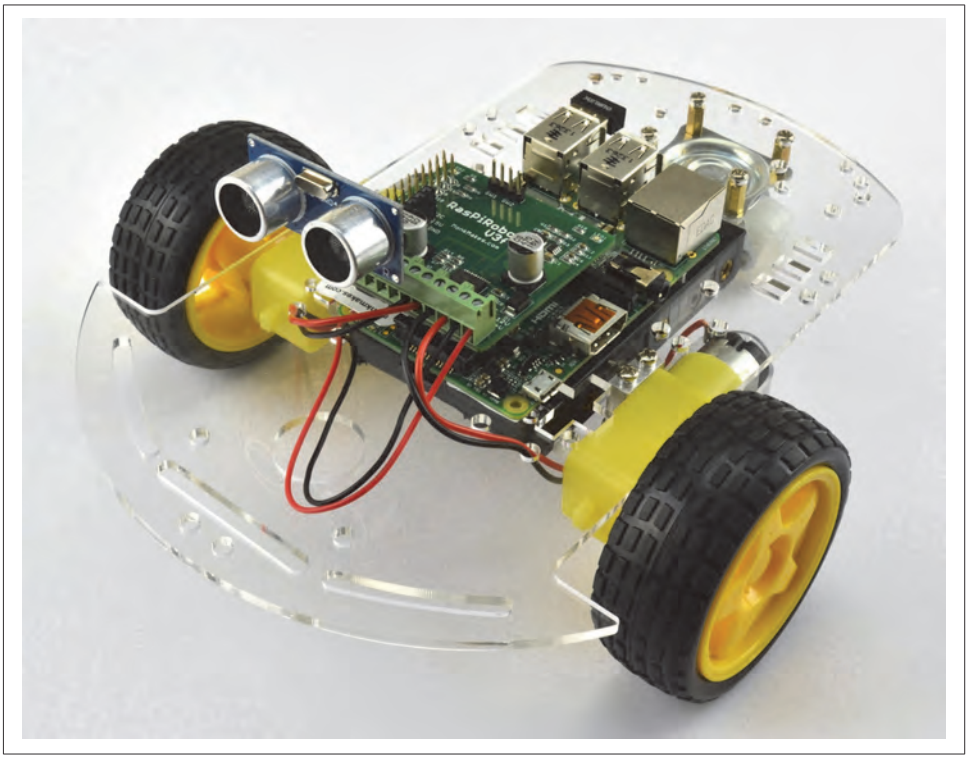


Figura 11.18. *El robot terminado.*

Para controlar el robot utilizará un programa de control que le permitirá dirigir el robot con las teclas de un ordenador portátil u otro ordenador conectado a su Raspberry Pi a través de SSH. Si aún no lo ha hecho, configure su Raspberry Pi para usar wifi y SSH, usando el Capítulo 2.6 y el Capítulo 2.8.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas en este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *rover.py*.

```
from rrb3 import *
import sys
import tty
import termios

rr = RRB3(9.0, 6.0) # batería, motor

UP = 0
DOWN = 1
RIGHT = 2
LEFT = 3
```

```
print("Use the arrow keys to move the robot")
print("Press Ctrl-C to quit the program")

# Estas funciones le permiten al programa leer su teclado
def readchar():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    if ch == '0x03':
        raise KeyboardInterrupt
    return ch

def readkey(getchar_fn=None):
    getchar = getchar_fn or readchar
    c1 = getchar()
    if ord(c1) != 0x1b:
        return c1
    c2 = getchar()
    if ord(c2) != 0x5b:
        return c1
    c3 = getchar()
    return ord(c3) - 65 # 0=Arriba, 1=Abajo, 2=Derecha, 3=Izquierda (flechas)

#Esto controlará el movimiento de su robot y lo mostrará en su pantalla
try:
    while True:
        keyp = readkey()
        if keyp == UP:
            rr.forward(1)
            print 'forward'
        elif keyp == DOWN:
            rr.reverse(1)
            print 'backward'
        elif keyp== RIGHT:
            rr.right(1)
            print 'clockwise'
        elif keyp== LEFT:
            rr.left(1)
            print 'anti clockwise'
        elif keyp== LEFT:
            rr.left(1)
            print 'anti clockwise'
        elif ord(keyp) == 3:
            break

except KeyboardInterrupt:
    GPIO.cleanup()
```

Para poder interceptar las pulsaciones de las teclas, este programa utiliza la biblioteca *termios* y las dos funciones *readchar* y *readkey*.

Después de los comandos de importación, se crea una nueva instancia de RRB3. Los dos parámetros para esto son el voltaje de la batería y el voltaje del motor (en este caso, 9 V y 6 V, respectivamente). Si su chasis tiene diferentes motores, cambie el segundo parámetro.

El bucle principal solo comprueba las pulsaciones de teclas y después envía los comandos apropiados de *avance*, *retroceso*, *izquierda* o *derecha* a la biblioteca RRB3.

Observaciones

Usted puede hacer el Rover más interesante añadiéndole periféricos. Podría, por ejemplo, conectar una webcam y configurarlo para verlo en directo para que su robot se convierta en un espía itinerante (Capítulo 4.6).

La biblioteca RRB3 también admite los sensores de distancia HC-SR04, que se pueden conectar en un zócalo de la tarjeta RaspiRobot V3. Puede utilizarlo para detectar obstáculos. Encontrará un ejemplo de software para esto en la biblioteca RRB3.

Para saber más

Puede obtener más información sobre la tarjeta RaspiRobot y la biblioteca RRB3 en GitHub.

CAPÍTULO 12

Entradas digitales

12.1 Introducción

En este capítulo se hablará sobre el uso de entradas digitales, como interruptores y teclados. Este capítulo también cubre módulos que tienen una salida digital que se puede conectar a una entrada GPIO de Raspberry Pi.

En la mayoría de las secciones necesitará una placa de pruebas y cables puente (revise el Capítulo 9.9).

12.2 Conectar un pulsador

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea conectar un interruptor a su Raspberry Pi para que, cuando lo pulse, se ejecute algún código Python.

Solución

Conecte un pulsador a un pin GPIO y utilice la biblioteca `RPi.GPIO` en su programa de Python para detectar la pulsación del botón.

Para esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Pulsador táctil (vea "Varios" en la página 477)

La Figura 12.1 muestra cómo conectar un pulsador táctil usando una placa de pruebas y cables puente.

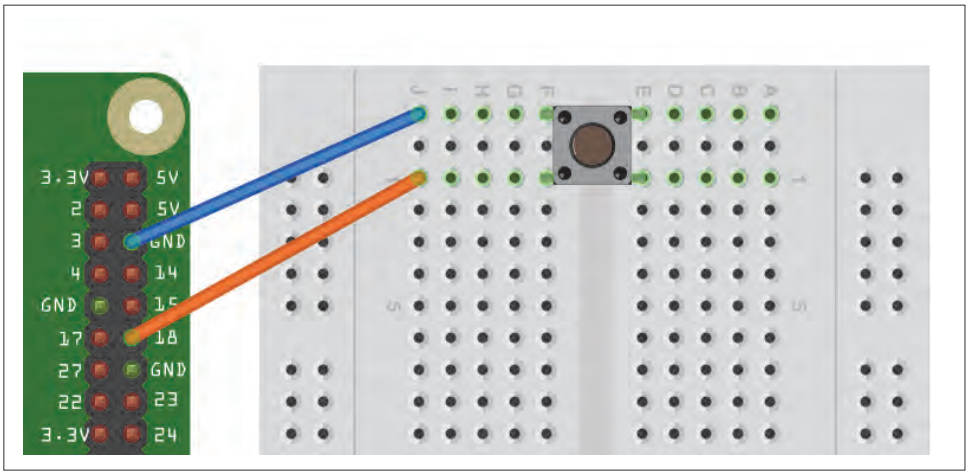


Figura 12.1. *Conexión de un pulsador táctil a Raspberry Pi*

Una alternativa al uso de una placa de pruebas y al pulsador táctil es utilizar el botón Squid (Figura 12.2). Se trata de un pulsador con cabezal hembra soldado en el extremo, que se puede conectar directamente al conector GPIO (Capítulo 9.12).

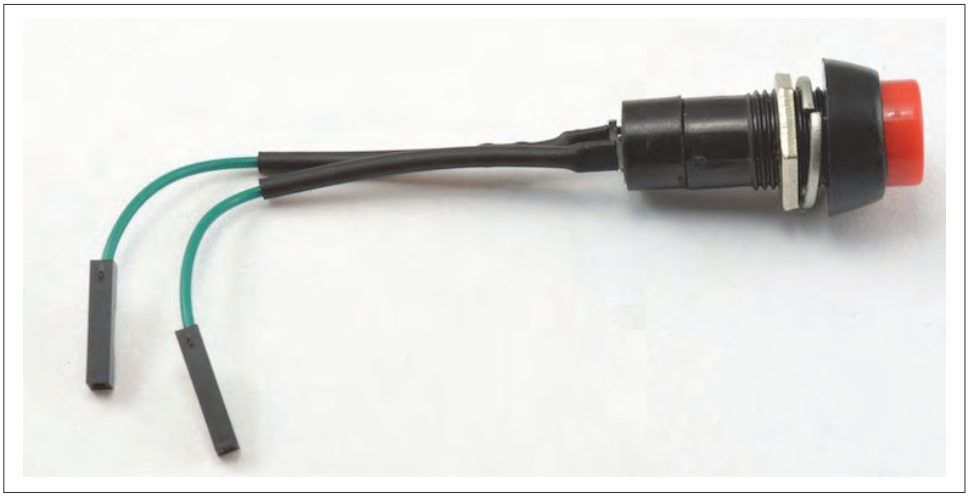


Figura 12.2. *Botón Squid.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas en este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *switch.py*.

Este código de ejemplo muestra un mensaje cuando se pulsa el botón:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)

while True:
    input_state = GPIO.input(18)
    if input_state == False:
        print('Button Pressed')
        time.sleep(0.2)
```

Tendrá que ejecutar el programa como superusuario:

```
pi@raspberrypi ~ $ sudo python switch.py
Button Pressed
Button Pressed
Button Pressed
Button Pressed
```

Observaciones

Verá que el interruptor está cableado de manera que cuando se pulse, conecte el pin 18 configurado como entrada a GND. El pin de entrada tiene 3,3 V por el argumento opcional `pull_up_down=GPIO.PUD_UP` en `GPIO.setup`. Esto significa que cuando lea el valor de entrada utilizando `GPIO.input`, devolverá `False` si se presiona el botón. Esto puede ser poco intuitivo.

Cada pin GPIO tiene resistencias *pull-up* y *pull-down* configurables por *software*. Cuando utilice un pin GPIO como entrada puede configurar estas resistencias para que una, o ninguna, esté habilitada con el parámetro opcional `pull_up_down` de `GPIO.setup`. Si omite este parámetro, ninguna de las resistencias se habilitará. Esto deja la entrada en *flotante*, lo que significa que no se puede fiar de su valor y que está a la deriva entre alto y bajo.

Si está configurado como `GPIO.PUD_UP`, la resistencia *pull-up* estará habilitada; y si está configurado como `GPIO.PUD_DOWN`, la resistencia *pull-down* estará habilitada.

Mientras que algunos pulsadores táctiles tienen solo dos conexiones, la mayoría tienen cuatro. La Figura 12.3 muestra cómo se configuran esas conexiones.

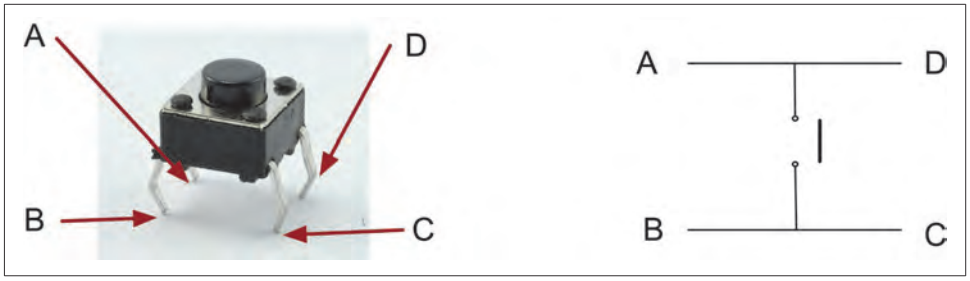


Figura 12.3. *Pulsador táctil.*

En realidad, solo hay dos conexiones eléctricas porque, dentro del interruptor, los pines B y C están conectados entre sí, así como A y D.

Para saber más

Para obtener más información sobre el uso de una placa de pruebas y de cables puente con Raspberry Pi, vea el Capítulo 9.9.

Para utilizar un interruptor para activar una interrupción, vaya al Capítulo 10.14.

Para evitar el rebote de un interruptor, consulte el Capítulo 12.6.

Para utilizar resistencias *pull-up* o *pull-down*, revise el Capítulo 12.7.

12.3 Conmutar con un pulsador

Problema

Desea activar y desactivar algo con un pulsador para que cambie entre encendido y apagado cada vez que lo pulse.

Solución

Guarde el último *estado* del botón e invierta ese valor cada vez que se pulse el botón.

El siguiente ejemplo activa y desactiva un led al pulsar el interruptor.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Pulsador táctil (vea "Varios" en la página 477)

- Led (vea "Optoelectrónica", en la página 476)
- Resistor de 470 Ω (vea "Resistores y condensadores", en la página 474)

La Figura 12.4 muestra cómo conectar un pulsador táctil y un led, usando una placa de pruebas y cables puente.

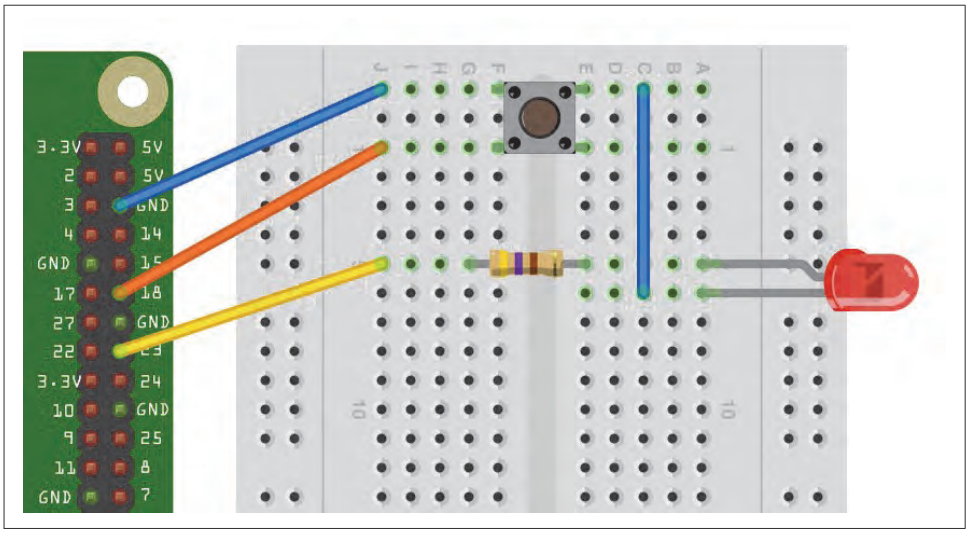


Figura 12.4. *Conexión de un pulsador y un led a Raspberry Pi.*

Además de los cables puente macho a hembra que conectan Raspberry Pi a la placa de pruebas, también necesitará un cable puente macho a macho.

Abra un editor (nano o IDLE) y pegue el siguiente código. También puede descargar el programa de la sección de código desde la página web del libro, donde pone *switch_on_off.py*:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

switch_pin = 18
led_pin = 23

GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(led_pin, GPIO.OUT)

led_state = False
old_input_state = True #pulled-up

while True:
```

```
new_input_state = GPIO.input(switch_pin)
if new_input_state == False and old_input_state == True:
    led_state = not led_state
old_input_state = new_input_state
GPIO.output(led_pin, led_state)
```

Observaciones

La variable `led_state` contiene el estado actual del led (`True` para encendido y `False` para apagado). Cada vez que se pulse el botón se ejecutará la siguiente línea:

```
led_state = not led_state
```

El comando `not` invierte el valor de `led_state`, así que si `led_state` es `True`, se volverá `False` y viceversa.

La variable `old_input_state` se utiliza para recordar la posición del botón, de modo que una pulsación de botón se define como ocurrida cuando el estado de la entrada cambia de `True` (interruptor no pulsado) a `False` (interruptor pulsado).

Para saber más

Verá que a veces cuando pulsa el botón no parece que se alterne el led. Esto ocurre por el rebote del interruptor. Puede encontrar otras técnicas para evitar el rebote en el Capítulo 12.6.

12.4 Usar un conmutador de dos posiciones o un interruptor deslizante

Problema

Desea conectar un conmutador de dos posiciones o un interruptor deslizante a su Raspberry Pi y ser capaz de encontrar la posición del interruptor en su programa Python.

Solución

Utilice el interruptor como lo haría con un pulsador táctil (Capítulo 12.2): simplemente conecte el centro y un contacto final (Figura 12.5).

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea "Equipo para prototipos", página 474)
- Conmutador pequeño o interruptor deslizante (vea "Varios", página 477)

El mismo código que usó en el Capítulo 12.2 funcionará aquí.

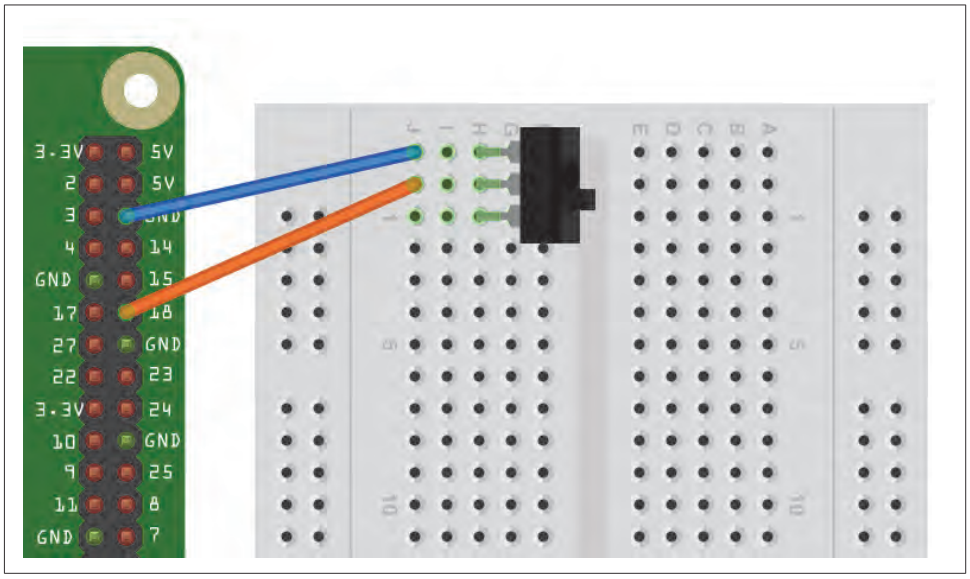


Figura 12.5. *Conexión de un interruptor deslizante a Raspberry Pi.*

Observaciones

Este tipo de conmutadores deslizantes son útiles porque se puede ver la posición sin necesidad de ningún indicador adicional como un led. Sin embargo, son más frágiles y un poco más caros que los pulsadores táctiles, que se utilizan cada vez más en la electrónica de consumo, ya que son más bonitos.

Para saber más

Para utilizar un interruptor de tres posiciones con una posición central de apagado, vea el Capítulo 12.5.

12.5 Usar un conmutador con la posición central de apagado o un interruptor deslizante

Problema

Desea conectar un conmutador de tres posiciones (la central de apagado) a su Raspberry Pi y ser capaz de encontrar la posición del interruptor en su programa Python.

Solución

Conecte el conmutador a dos pines GPIO, como se ve en la Figura 12.6, y use la biblioteca `RPi.GPIO` en su programa Python para detectar la posición del conmutador.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Interruptor de conmutación de tres posiciones (la central de apagado) en miniatura (vea "Varios", página 477)

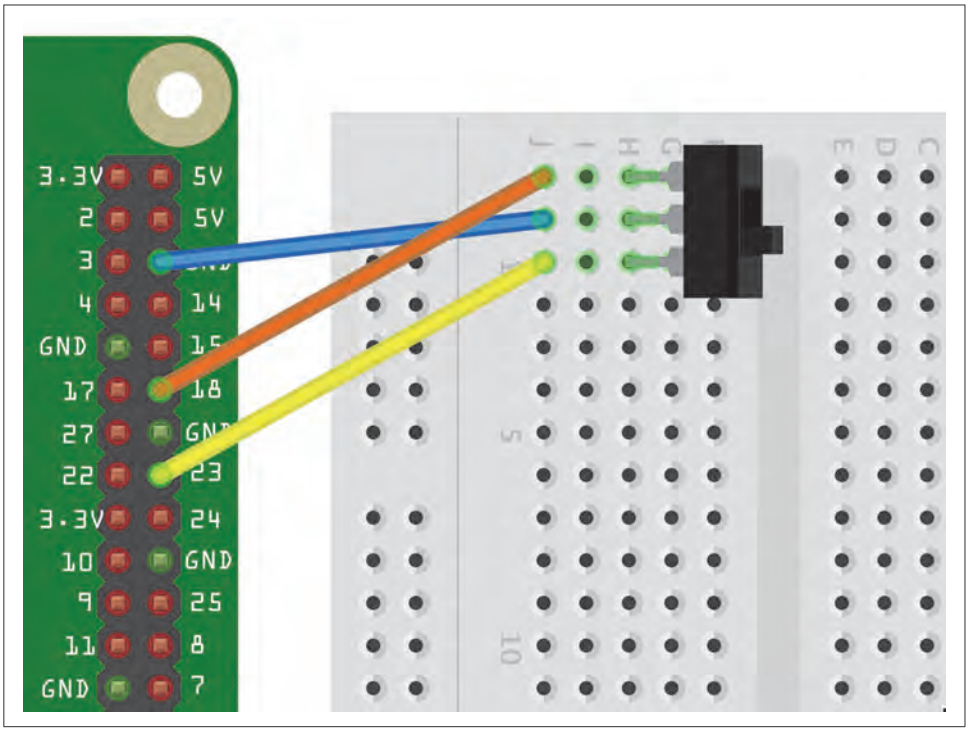


Figura 12.6. *Conectar un interruptor de tres posiciones a Raspberry Pi.*

La conexión común (central) del interruptor está conectada a tierra, y cada uno de los dos extremos del interruptor está conectado a un pin GPIO con la resistencia *pull-up* activada.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la web del libro, donde pone *switch_2_pos.py*:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

top_input = 18
```

```
bottom_input = 23

GPIO.setup(top_input, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(bottom_input, GPIO.IN, pull_up_down=GPIO.PUD_UP)

switch_position = "unknown"

while True:
    top_state = GPIO.input(top_input)
    bottom_state = GPIO.input(bottom_input)
    new_switch_position = "unknown"
    if top_state == False:
        new_switch_position = "up"
    elif bottom_state == False:
        new_switch_position = "down"
    else:
        new_switch_position = "center"
    if new_switch_position != switch_position:
        switch_position = new_switch_position
        print(switch_position)
```

Ejecute el programa y cuando mueva el conmutador de arriba hacia abajo, se informará de la posición cada vez que cambie:

```
$ sudo python switch_3_pos.py
up
center
down
```

Observaciones

El programa configura dos entradas con resistencias *pull-up* habilitadas. La variable `switch_position` se usa para registrar la posición actual del conmutador.

Dentro del bucle se leen ambas entradas GPIO, y las tres condiciones de la estructura `if`, `elif` y `else` determinan la posición del conmutador, asignando el valor a una variable llamada `new_switch_position`. Si esto difiere del valor anterior, se imprimirá la posición del interruptor.

Encontrará una amplia gama de interruptores de conmutación. Algunos serán descritos como DPDT, SPDT, SPST o SPST, momentáneamente encendido, y así sucesivamente. El significado de estas letras es el siguiente:

- D: Doble
- S: Simple
- P: Polo
- T: Vía

Un interruptor DPDT es de doble polo y de doble vía. La palabra *polo* se refiere al número de contactos de conmutador separados que se controlan desde una palanca mecánica. Por lo tanto, un interruptor de dos polos puede encender y apagar dos cosas independientemente. Un interruptor de una simple vía solo puede abrir o cerrar un solo contacto (o dos si es de doble polo). Sin embargo, un interruptor de doble vía puede conectar el contacto común a uno de los otros contactos.

La Figura 12.7 muestra los tipos más comunes de interruptor.

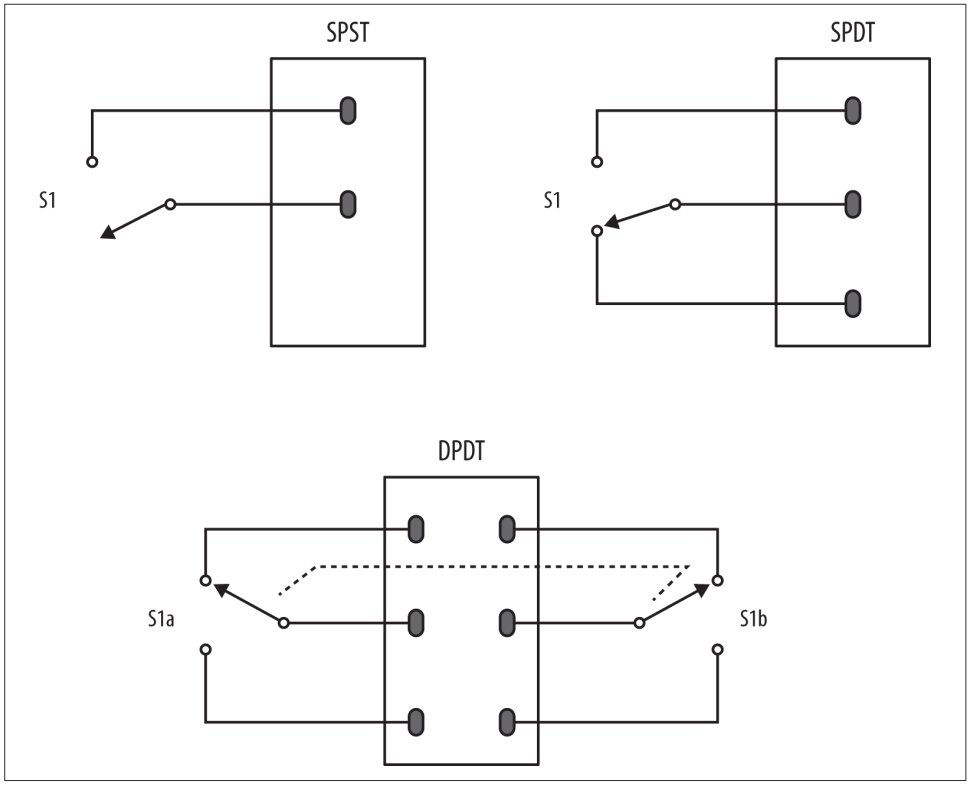


Figura 12.7. *Tipos de interruptores de conmutación.*

Para saber más

Para obtener más información sobre cómo funcionan los enunciados `if`, revise el Capítulo 5.19.

Para leer el capítulo de interruptores más básico, consulte el Capítulo 12.2.

12.6 Eliminar el rebote al pulsar un botón

Problema

A veces, cuando presiona el botón de un interruptor, la acción esperada sucede más de una vez, porque los contactos del interruptor *rebotan*. En ese caso, desea escribir código para eliminar los rebotes del interruptor.

Solución

Hay una serie de soluciones a este problema. Para explorarlas, reproduzca la configuración de la placa de pruebas del Capítulo 12.3.

El código original para este ejemplo, sin ningún intento de eliminar los rebotes, es como:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

switch_pin = 18
led_pin = 23

GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(led_pin, GPIO.OUT)

led_state = False
old_input_state = True # pulled-up

while True:
    new_input_state = GPIO.input(switch_pin)
    if new_input_state == False and old_input_state == True:
        led_state = not led_state
    old_input_state = new_input_state
    GPIO.output(led_pin, led_state)
```

El problema es que, si los contactos del interruptor rebotan, es como si el interruptor se presionara más de una vez en una sucesión muy rápida. Si rebotan un número de veces impar, las cosas parecerán ir bien. Pero si rebotan un número de veces par, parecerá que no funciona.

Debe ignorar cualquier cambio después de que el interruptor se pulse durante un corto período de tiempo, mientras que el interruptor termina de rebotar.

La forma rápida y sencilla de hacerlo es introducir una breve pausa después de que se haya detectado la pulsación del botón, añadiendo el comando `time.sleep`, de, digamos, 0,2 segundos. Este retraso probablemente es más alto de lo necesario y puede reducirlo considerablemente.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

switch_pin = 18
led_pin = 23

GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(led_pin, GPIO.OUT)

led_state = False
old_input_state = True # pulled-up

while True:
    new_input_state = GPIO.input(switch_pin)
    if new_input_state == False and old_input_state == True:
        led_state = not led_state
        time.sleep(0.2)
    old_input_state = new_input_state
    GPIO.output(led_pin, led_state)
```

Observaciones

Esta solución está bien en la mayoría de las situaciones, pero también puede utilizar el pin del interruptor como una interrupción (Capítulo 10.14).

La conmutación de rebote ocurre en la mayoría de los interruptores y puede ser bastante severa en algunos, como muestra el trazado del osciloscopio de la Figura 12.8.

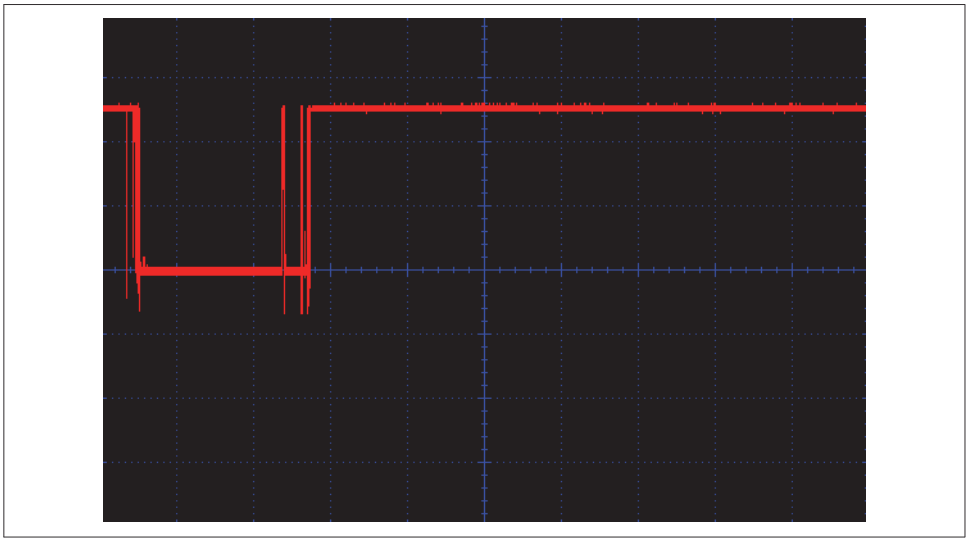


Figura 12.8. *Rebotes de contacto con un interruptor.*

Puede ver que hay rebotes de contacto cuando se cierra el interruptor y cuando se libera.

Para saber más

Para obtener más información básica sobre cómo conectar un botón, consulte el Capítulo 12.2.

La biblioteca del botón Squid quita el rebote de las pulsaciones del botón usando un enfoque similar al descrito aquí. Revise el Capítulo 9.12 para conocer los detalles del botón Squid.

12.7 Usar una resistencia pull-up externa

Problema

Quiere poner un cable largo desde Raspberry Pi al interruptor, pero recibe algunas lecturas falsas en el pin de entrada.

Solución

Las resistencias pull-up internas son bastante débiles (alrededor de 40 kΩ). Si le pone un cable largo al interruptor o funciona en un entorno eléctricamente ruidoso, puede obtener falsos valores en la entrada digital. Para solucionar esto, apague las resistencias internas pull-up y pull-down y utilice una resistencia pull-up externa.

La Figura 12.9 muestra el uso de una resistencia pull-up externa.

Para probar este *hardware*, puede usar el programa *switch.py*. Revise el Capítulo 12.2.

Observaciones

A pulsar el botón, la corriente fluirá desde 3,3 V a través de la resistencia a tierra. Un resistor de 100 Ω extrae una corriente de 3,3 V/100 Ω = 33 mA. Esto está dentro del límite seguro para el suministro de 3,3 V de 50 mA para una Raspberry Pi 1, así que no utilice un valor más bajo que esto si tiene una Raspberry Pi más vieja. Si está utilizando un GPIO de una Raspberry Pi de 40 pines, podría bajar este valor a, probablemente, 47 Ω.

En casi todos los casos, un resistor de 1 kΩ proporcionará un rango adecuado sin problemas.

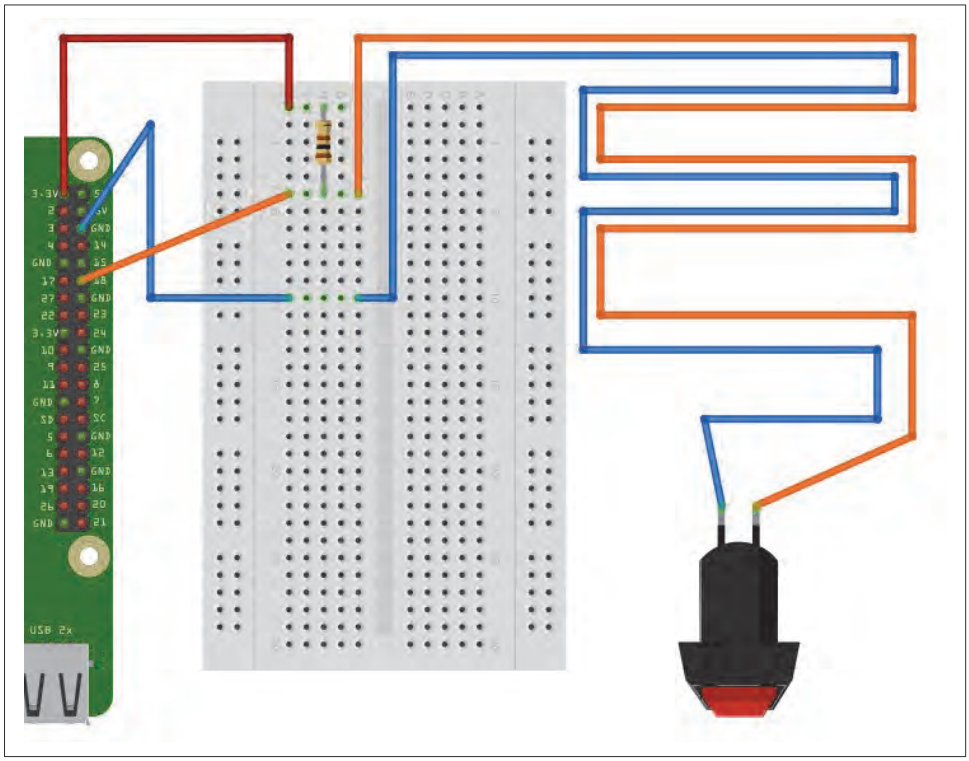


Figura 12.9. *Usar una resistencia externa pull-up.*

Para saber más

Para obtener información básica sobre cómo conectar un botón, vea el Capítulo 12.2.

12.8 Usar un codificador rotatorio (cuadratura)

Problema

Desea detectar la rotación usando un codificador rotatorio.

Solución

Utilice un codificador rotatorio (codificador en cuadratura) conectado a dos pines GPIO, como se muestra en la Figura 12.10.

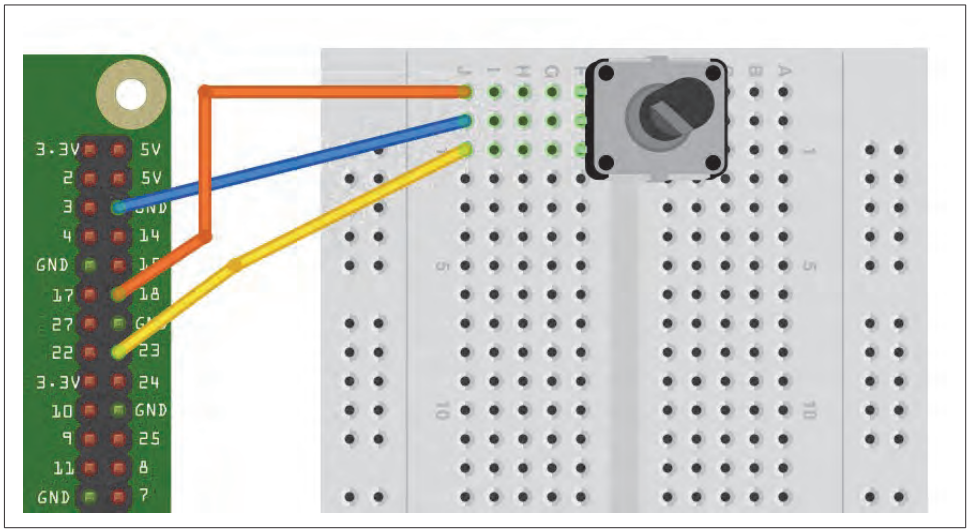


Figura 12.10. *Conexión de un codificador rotatorio.*

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Codificador rotatorio (tipo de cuadratura; vea "Varios" en la página 477)

Este tipo de codificador rotatorio se llama *codificador en cuadratura*, y se comporta como un par de interruptores. La secuencia en la que se abren y cierran, a medida que gira el eje del codificador, determina la dirección de rotación.

El codificador rotatorio mostrado tiene el cable central como el cable *común* y los dos cables en ambos lados como A y B. No todos los codificadores rotatorios utilizan esta disposición, así que compruebe la asignación del patillaje en la hoja de datos para el codificador rotatorio que esté usando. Muchos codificadores rotatorios incluyen un interruptor pulsador, que tiene un par separado de contactos.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la web del libro, donde pone *rotary_encoder.py*.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

input_A = 18
input_B = 23
```

```
GPIO.setup(input_A, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(input_B, GPIO.IN, pull_up_down=GPIO.PUD_UP)


old_a = True
old_b = True

def get_encoder_turn():
    #devolver -1, 0, o +1
    global old_a, old_b
    result = 0
    new_a = GPIO.input(input_A)
    new_b = GPIO.input(input_B)
    if new_a != old_a or new_b != old_b :
        if old_a == 0 and new_a == 1 :
            result = (old_b * 2 - 1)
        elif old_b == 0 and new_b == 1 :
            result = -(old_a * 2 - 1)
    old_a, old_b = new_a, new_b
    time.sleep(0.001)
    return

result x = 0

while True:
    change = get_encoder_turn()
    if change != 0 :
        x = x + change
           print(x)
```

El programa de prueba simplemente cuenta al girar el codificador rotativo en el sentido de las agujas del reloj, y cuenta hacia atrás al girar hacia la izquierda.

```
pi@raspberrypi ~ $ sudo python rotary_encoder.py
1
2
3
4
5
6
7
8
9
10
9
8
7
6
5
4
```

Observaciones

La Figura 12.11 muestra la secuencia de pulsos que obtendrá de los dos contactos, A y B. Puede ver que el patrón se repite después de cuatro pasos (de ahí el nombre *codificación en cuadratura*).

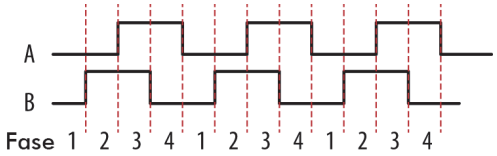


Figura 12.11. *Cómo funcionan los codificadores en cuadratura.*

Al girar en el sentido de las agujas del reloj (de izquierda a derecha en la Figura 12.11), la secuencia será:

Fase	A	B
1	0	0
2	0	1
3	1	1
4	1	0

Al girar en la dirección opuesta, la secuencia de fases se invertirá.

Fase	A	B
4	1	0
3	1	1
2	0	1
1	0	0

El programa Python enumerado anteriormente implementa el algoritmo para determinar la dirección de rotación en la función `get_encoder_turn`. La función devolverá `0` si no ha habido movimiento, `1` para una rotación en sentido horario, o `-1` para una rotación en sentido contrario a las agujas del reloj. Utiliza dos variables, `old_a` y `old_b`, para almacenar los estados anteriores de los conmutadores A y B. Al compararlos con los valores recién leídos, puede determinar (con un poco de lógica) la dirección en la que gira el codificador.

El período de reposo de 1 milisegundo es para asegurar que la siguiente muestra no ocurra demasiado pronto después de la muestra anterior; de lo contrario, las transiciones pueden dar falsas lecturas.

El programa de prueba debe funcionar de manera fiable, no importa cuán rápido gire el botón del codificador rotatorio. Sin embargo, intente evitar hacer algo que consuma mucho tiempo en el bucle, o perderá pasos a su vez.

Para saber más

También puede medir la posición girada de un botón usando una resistencia variable con el método de respuesta de paso (Capítulo 13.2) o utilizando un conversor de analógico a digital (Capítulo 13.6).

12.9 Usar un teclado

Problema

Desea interconectar un teclado con su Raspberry Pi.

Solución

Los teclados están dispuestos en filas y columnas con un interruptor pulsador en la intersección de cada fila o columna. Para averiguar qué tecla es pulsada, primero conecte todas las conexiones de filas y columnas a los pines GPIO de Raspberry Pi. Por lo tanto, para un teclado 4×3, necesitará cuatro + tres pines. Al escanear cada columna a su vez (configurándola en salida alta) y al leer el valor de cada una de las entradas de las filas, puede determinar qué tecla (si la hay) es pulsada.

Tenga en cuenta que los teclados muestran una variación considerable en la asignación de patillaje.

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea “Equipos para prototipos”, página 474)
- Teclado 4 × 3 (vea “Varios” en la página 477)
- Siete pines de encabezado macho (vea “Varios”, página 477)

La Figura 12.12 muestra el diagrama de cableado del proyecto utilizando el teclado SparkFun incluido en “Varios” en la página 477. El teclado se suministra sin los pines de encabezado, que deben soldarse a él.

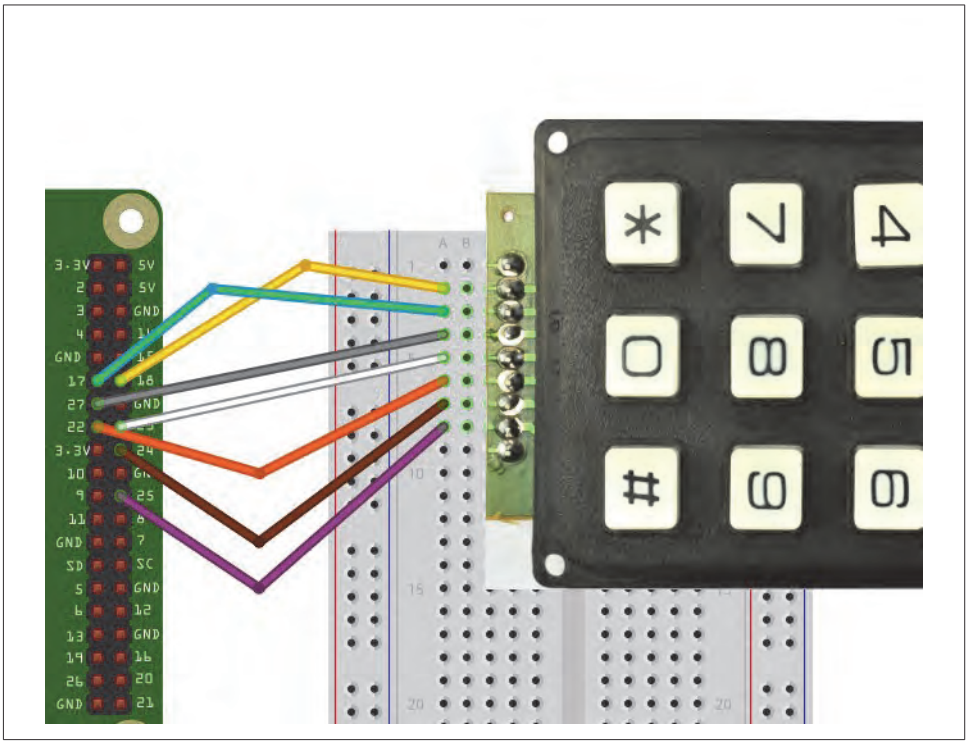


Figura 12.12. *Diagrama de cableado del teclado.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *keypad.py*.

Antes de ejecutar el programa, asegúrese de que los pines de las filas y columnas son correctos para el teclado que está utilizando y, si es necesario, cambie los valores en las variables `rows` y `cols`. Si no lo hace, es posible que al pulsar una tecla se pueda cruzar una salida GPIO a otra, donde una es alta y otra es baja. Esto probablemente dañaría su Raspberry Pi.

La fila y las columnas definidas aquí son correctas para el teclado SparkFun que se muestra en "Varios" en la página 477 en el Apéndice A. La primera fila está conectada al pin GPIO 17, la segunda al 25, y así sucesivamente. El cableado de la fila y la columna al conector del teclado se ilustra en la Figura 12.13.

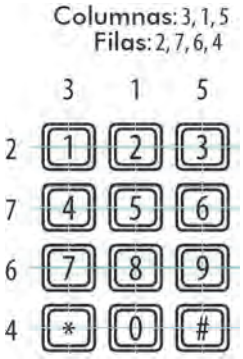


Figura 12.13. *Conexiones de los pines del teclado.*

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

rows = [17, 25, 24, 23]
cols = [27, 18, 22]
keys = [
    ['1', '2', '3'],
    ['4', '5', '6'],
    ['7', '8', '9'],
    ['*', '0', '#']]

for row_pin in rows:
    GPIO.setup(row_pin, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

for col_pin in cols:
    GPIO.setup(col_pin, GPIO.OUT)

def get_key():
    key = 0
    for col_num, col_pin in enumerate(cols):
        GPIO.output(col_pin, 1)
        for row_num, row_pin in enumerate(rows):
            if GPIO.input(row_pin):
                key = keys[row_num][col_num]
        GPIO.output(col_pin, 0)
    return key

while True:
    key = get_key()
    if key:
        print(key)
    time.sleep(0.3)
```

Debe ejecutar el programa con privilegios de superusuario, ya que accede al GPIO. Puede ver el trazado del programa pulsando cada tecla sucesivamente.

```
pi@raspberrypi ~ $ sudo python keypad.py 1
2
3
4
5
6
7
8
9
*
0
#
```

Observaciones

La variable `keys` contiene un esquema del nombre clave para cada posición de fila y columna. Puede personalizarlo para su teclado.

Debido a que hay bastantes pines para inicializar como entradas y salidas, tanto los pines de la fila como de la columna se inicializan en bucles.

Toda la acción real tiene lugar en la función `get_key`. Esto habilita cada columna a su vez, configurándola como alta. Un bucle interno luego prueba cada una de las filas a su vez. Si una de las filas es alta, el nombre clave correspondiente a esa fila y columna se busca en el conjunto `keys`. Si no se detecta ninguna pulsación de tecla, se devuelve el valor por defecto de la `key` (`0`).

El bucle principal `while` solo obtiene el valor clave y lo imprime. El comando `sleep` reduce la velocidad de salida.

Para saber más

Una alternativa a añadir un teclado es simplemente usar un teclado USB. De esa manera solo podrá capturar pulsaciones de teclas como se describe en el Capítulo 12.12.

12.10 Detectar movimiento

Problema

Desea activar alguna acción en Python cuando se detecte movimiento.

Solución

Utilice un módulo de detector de movimiento infrarrojo pasivo (PIR).

Para hacer esto necesitará:

- Cables puente hembra-hembra (vea "Equipos para prototipos", página 474)
- Módulo detector de movimiento PIR (vea "Varios" en la página 476)

La Figura 12.14 muestra cómo se conecta el módulo del sensor. Este módulo exige una fuente de alimentación de 5 V y tiene una salida de 3,3 V, por lo que es ideal para su uso con Raspberry Pi.

Asegúrese de que el módulo PIR que utiliza tenga una salida de 3,3 V. Si tiene una salida de 5 V, necesitará usar un par de resistores para reducirla a 3,3 V (vea el Capítulo 13.7).

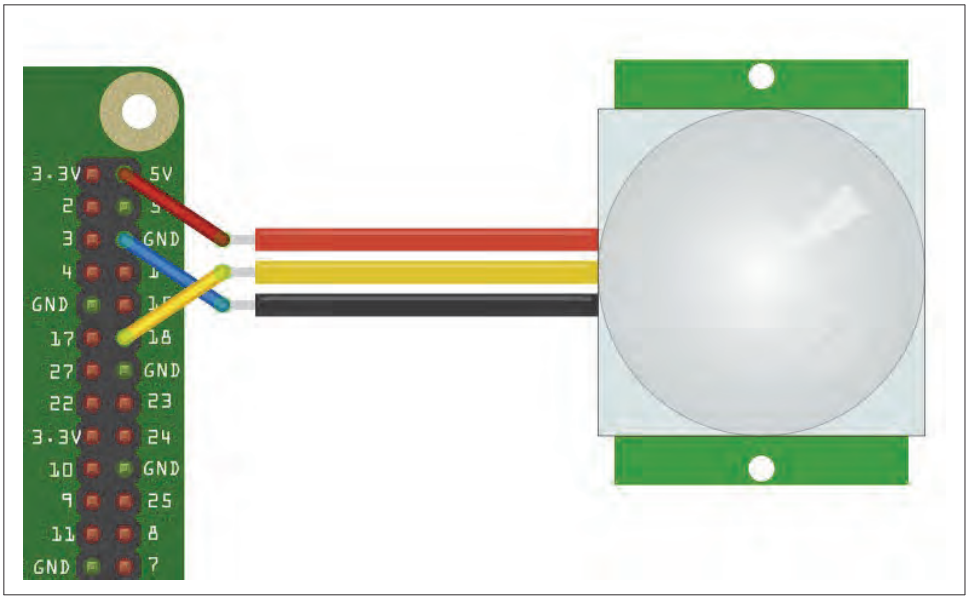


Figura 12.14. *Cableado de un detector de movimiento PIR.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *pir.py*.

```
import RPi.GPIO as
GPIO import time

GPIO.setmode(GPIO.BCM)

GPIO.setup(18, GPIO.IN)
```

```
while True:
    input_state = GPIO.input(18)
    if input_state == True:
        print('Motion Detected')
        time.sleep(1)
```

El programa simplemente imprime el estado de la entrada GPIO 18.

```
$ sudo python pir.py
Motion Detected
Motion Detected
```

Observaciones

Una vez activado, la salida del sensor PIR permanecerá alta durante un tiempo. Puede ajustarlo usando uno de los potenciómetros en su placa de circuito. El segundo potenciómetro (si está presente) establecerá el umbral del nivel de luz que desactivará el sensor. Esto es útil cuando se utiliza el sensor para controlar una luz, encendiéndola al detectar el movimiento, pero solo cuando está oscuro.

Para saber más

Podría combinar este capítulo con el Capítulo 7.17 para enviar un correo electrónico cuando se detecte un intruso o para integrarlo con *If This Then That* (IFTTT), que en castellano significa: 'Si ocurre esto, haz esto otro', para proporcionar posibles formas de notificación (consulte el Capítulo 15.4).

Para detectar movimiento a través de la visión artificial y una webcam, vea el Capítulo 8.7.

12.11 Añadir un GPS a Raspberry Pi

Problema

Desea conectar un módulo GPS en serie a una Raspberry Pi y acceder a los datos utilizando Python.

Solución

Un módulo GPS serie de 3,3 V se puede conectar directamente a la conexión RXD de la Raspberry Pi.

La Figura 12.15 muestra cómo se conecta el módulo. El RXD de Raspberry Pi se conecta al TX del módulo GPS. Las únicas otras conexiones son para GND y 5 V, por lo que podemos simplemente usar tres encabezados hembra a hembra.

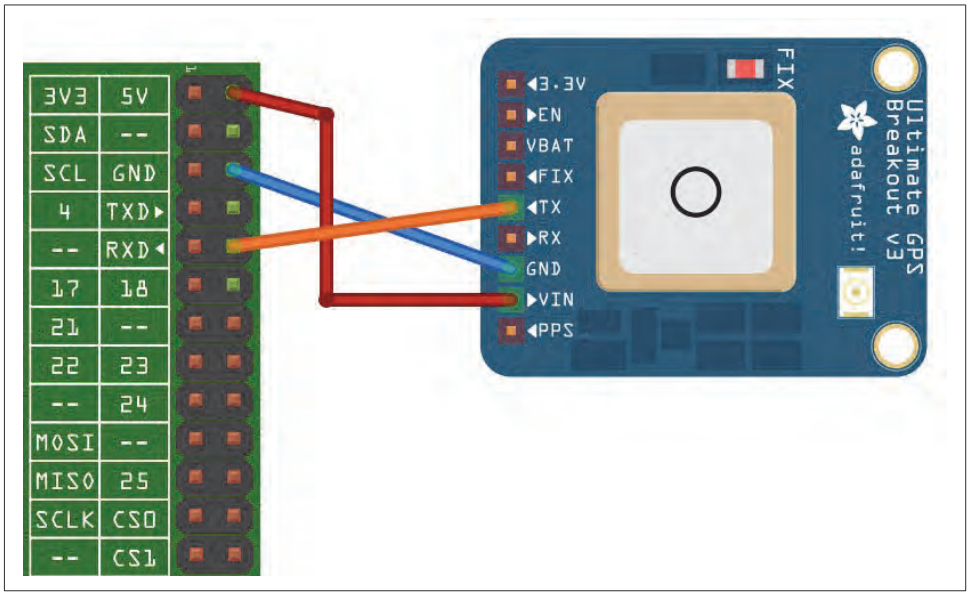


Figura 12.15. *Cableado de un GPS a Raspberry Pi.*

Los mensajes GPS requieren una cierta decodificación. Afortunadamente, hay un conjunto de herramientas para ayudarnos a hacer esto. Instale los siguientes paquetes:

```
$ sudo apt-get install gpsd
$ sudo apt-get install gpsd-clients
```

El más importante de estos es *gpsd*. Esta es una herramienta que lee datos GPS desde una conexión en serie o USB, así como de otras fuentes, y los pone a disposición de los programas cliente para que los utilicen proporcionando un servicio web local en el puerto 2748.

Para iniciar el servicio *gpsd*, ejecute el siguiente comando:

```
$ sudo gpsd /dev/ttyAMA0
```

Puede ver si está funcionando, escriba el comando:

```
$ cgps -s
```

`-s` es opcional; solo suprime la visualización de los datos sin procesar (vea la Figura 12.16).

El tercer paquete que instalamos (*python-gps*) es, como era de esperar, una biblioteca de Python para acceder a los datos GPS de manera agradable y conveniente. Utilizará *python_gps* con un programa de prueba para mostrar la latitud, la longitud y el tiempo.

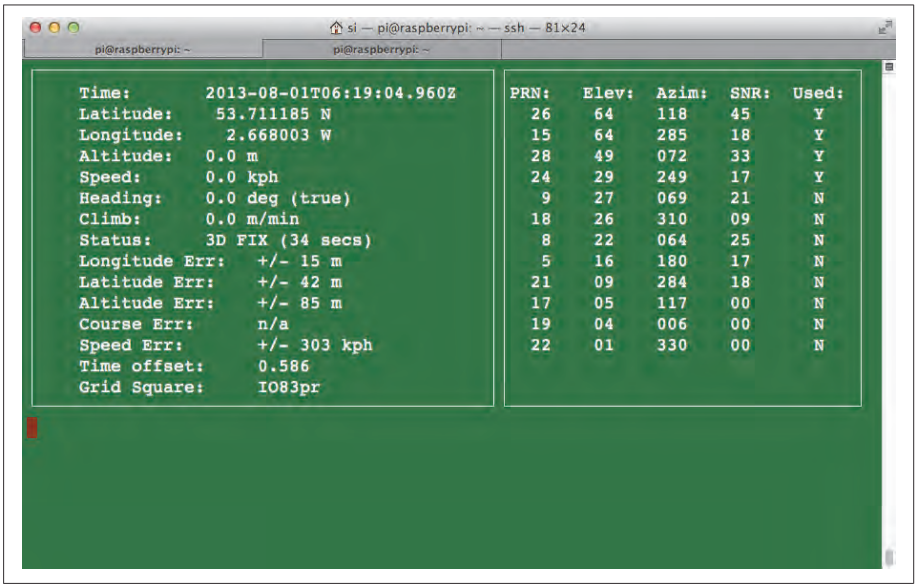


Figura 12.16. *Prueba de GPS con cgps.*

Abra un editor (nano o IDLE) y pegue el siguiente código. No llame al archivo *gps.py* o entrará en conflicto con la biblioteca GPS de Python. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la página web del libro, donde pone *gps_test.py*.

```
from gps import *
session = gps()
session.stream(WATCH_ENABLE|WATCH_NEWSTYLE)

while True:
    report = session.next()
    if report.keys()[0] == 'epx':
        lat = float(report['lat'])
        lon = float(report['lon'])
        print("lat=%f\tlon=%f\ttime=%s" % (lat, lon, report['time']))
        time.sleep(0.5)
```

Ejecute el programa y deberá ver algo como esto:

```
$ python gps_test.py
lat=53.710257   lon=-2.664245          time=2013-08-01T08:06:24.960Z
lat=53.710258   lon=-2.664252          time=2013-08-01T08:06:25.960Z
lat=53.710258   lon=-2.664252          time=2013-08-01T08:06:25.960Z
lat=53.710248   lon=-2.664243          time=2013-08-01T08:06:26.960Z
lat=53.710248   lon=-2.664243          time=2013-08-01T08:06:26.960Z
lat=53.710248   lon=-2.664250          time=2013-08-01T08:06:27.960Z
```

Observaciones

El programa crea una *sesión* y luego establece un flujo de datos que se leerán. El GPS, repetidamente, mandará mensajes en diferentes formatos. El comando `if` selecciona solo los mensajes que desea; aquellos que contienen la información posicional. Las partes del mensaje se almacenan en un diccionario desde el que se puede acceder a los campos y mostrarlos.

Además de utilizar los datos GPS con Python, también puede utilizar las herramientas `xgps` para visualizar los datos GPS (Figura 12.17). Solo tiene que introducir el comando:

```
$ xgps
```

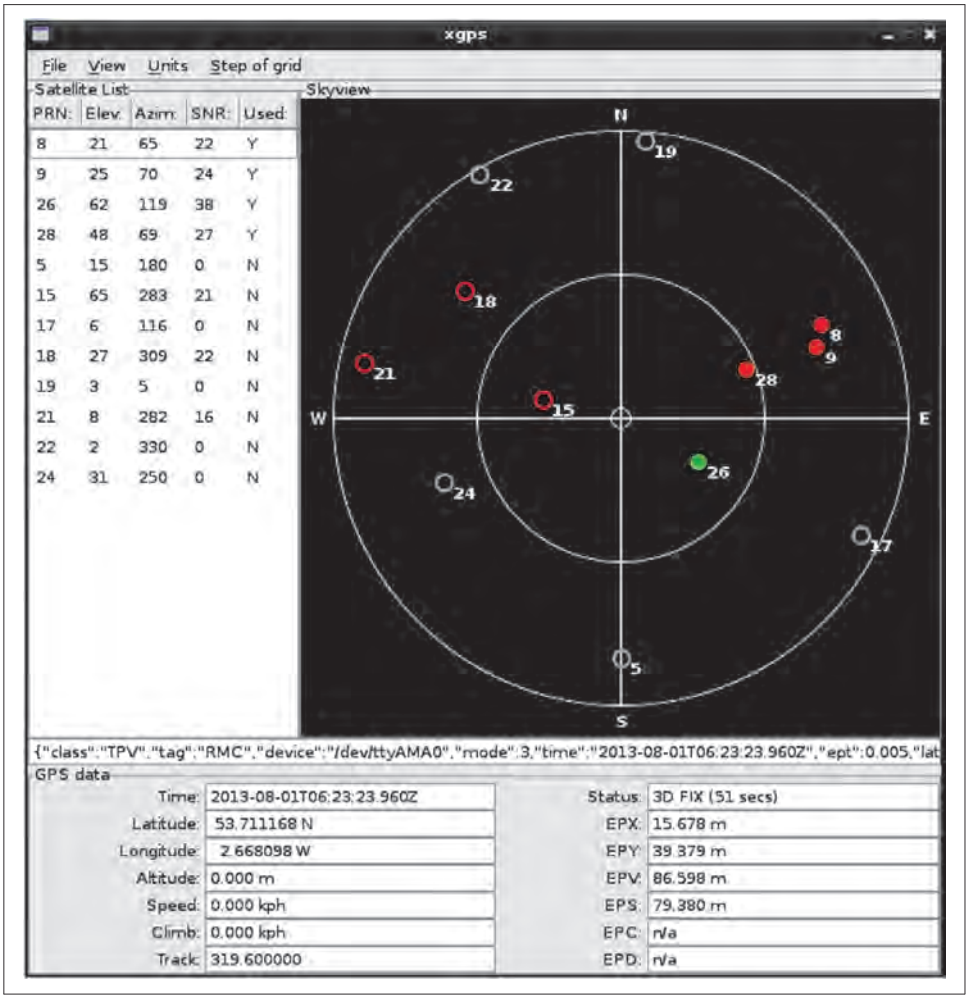


Figura 12.17. *Visualización GPS con xgps.*

Esta utilidad requiere una pantalla, por lo que debe ejecutarlo desde la propia Raspberry Pi o utilizando VNC (Capítulo 2.9) o RDP (Capítulo 2.10).

Para saber más

Puede utilizar el mismo enfoque utilizando un módulo GPS USB. Obtenga más información sobre *gpsd*.

12.12 Interceptar pulsaciones de teclas

Problema

Desea interceptar pulsaciones de teclas individuales en un teclado USB o en un teclado numérico.

Solución

Hay al menos dos maneras de resolver este problema. El más sencillo es utilizar la función `sys.stdin.read`. Esto tiene la ventaja, sobre el otro método, de que no necesita que se ejecute una interfaz gráfica de usuario, por lo que un programa que lo utiliza puede ejecutarlo desde una sesión SSH.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *keys_sys.py*.

```
import sys, tty, termios

def read_ch():
    fd = sys.stdin.fileno()
    old_settings = termios.tcgetattr(fd)
    try:
        tty.setraw(sys.stdin.fileno())
        ch = sys.stdin.read(1)
    finally:
        termios.tcsetattr(fd, termios.TCSADRAIN, old_settings)
    return ch

while True:
    ch = read_ch()
    if ch == 'x':
        break
    print("key is: " + ch)
```

La alternativa a esto es usar Pygame. Pygame es una biblioteca de Python destinada a la escritura de juegos. Se puede utilizar para detectar pulsaciones de teclas. Se debe usar con Python 2 en lugar de Python 3, ya que Pygame no está incluido en Python 3.

El siguiente ejemplo de programa ilustra el uso de Pygame para imprimir un mensaje cada vez que se pulse una tecla. Sin embargo, solo funciona si el programa tiene acceso al sistema de ventanas, por lo que tendrá que ejecutarlo usando VNC (Capítulo 2.9), RDP (Capítulo 2.10) o, directamente, desde Raspberry Pi. El código de ejemplo se encuentra en el archivo *keys_pygame.py* en el sitio web del libro.

```
import pygame
import sys
from pygame.locals import *

pygame.init()
screen = pygame.display.set_mode((640, 480))
pygame.mouse.set_visible(0)

while True:
   for event in pygame.event.get():
        if event.type == QUIT:
            sys.exit()
        if event.type == KEYDOWN:
            print("Code: " + str(event.key) + " Char: " + chr(event.key))
```

Esto abre una ventana en blanco de Pygame y las teclas solo serán interceptadas si se selecciona la ventana Pygame. El programa produce una salida en la ventana de terminal desde la que se ejecuta el programa.

Si pulsa una tecla de flecha o una tecla Shift con el primer foco de lectura `stdin`, el programa lanzará un error porque esas teclas no tienen valor ASCII.

```
$python keys_pygame.py
Code: 97 Char: a
Code: 98 Char: b
Code: 99 Char: c
Code: 120 Char: x
Code: 13 Char:
```

En este caso, Ctrl-C no detendrá este programa de ejecutarse. Para detener el programa haga clic en la *X* de la ventana PyGame.

Observaciones

Cuando se utiliza el método Pygame, las otras teclas tienen valores constantes definidos para ellas, lo que permite utilizar el cursor y otras teclas no ASCII (como la tecla de flecha hacia arriba y la de Inicio) en el teclado. Esto no es posible con el otro enfoque.

Para saber más

Interceptar actividades de teclado también puede ser una alternativa al uso de un teclado matricial (Capítulo 12.9).

12.13 Interceptar movimientos de ratón

Problema

Desea detectar los movimientos del ratón en Python.

Solución

La solución a esto es muy similar a la de usar Pygame para interceptar actividades de teclado (Capítulo 12.12).

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la web del libro, donde pone *mouse_pygame.py*.

```
import pygame
import sys
from pygame.locals import *

pygame.init()
screen = pygame.display.set_mode((640, 480))
pygame.mouse.set_visible(0)

while True:
   for event in pygame.event.get():
        if event.type == QUIT:
            sys.exit()
        if event.type == MOUSEMOTION:
            print("Mouse: (%d, %d)" % event.pos)
```

`MOUSEMOTION` se activará cada vez que el ratón se mueva dentro de la ventana Pygame. Puede encontrar las coordenadas en el valor `pos`. Las coordinadas son coordenadas absolutas relativas a la esquina superior izquierda de la ventana:

```
Mouse: (262, 285)
Mouse: (262, 283)
Mouse: (262, 281)
Mouse: (262, 280)
Mouse: (262, 278)
Mouse: (262, 274)
Mouse: (262, 270)
Mouse: (260, 261)
Mouse: (258, 252)
Mouse: (256, 241)
Mouse: (254, 232)
```

Observaciones

Otros eventos que puede interceptar son `MOUSEBUTTONDOWN` y `MOUSEBUTTONUP`. Estos se pueden utilizar para detectar cuándo el botón izquierdo del ratón ha sido pulsado o soltado.

Para saber más

La documentación de Pygame para `mouse` se puede encontrar en la web de Pygame.

12.14 Usar un módulo de reloj en tiempo real

Problema

Desea que su Raspberry Pi le recuerde la hora, incluso cuando no tenga conexión a la red.

Solución

Utilice un módulo de reloj en tiempo real (RTC).

Un chip RTC muy común es el DS1307. Tiene una interfaz I2C y se puede comprar como un módulo listo para usar que incluye el chip en sí, un cristal de cuarzo para mantener preciso el reloj y un soporte de batería para pilas de litio de 3 V.

Para hacer esto necesitará:

- Un módulo DS1307 o un RTC compatible (vea "Módulos", en la página 476)
- Cables puente hembra-hembra (vea "Equipos para prototipos", página 474)

El módulo RTC que está utilizando debe ser compatible con 3,3 V. Esto significa que su interfaz I2C no debe tener resistencias pull-up en absoluto, o deben llegar hasta los 3,3 V y no 5 V. Cuando utilice el modelo Adafruit aquí, simplemente no incluya las dos resistencias al soldar el módulo. Si tiene un módulo ya preparado, retire cuidadosamente las resistencias pull-up.

Monte el módulo RTC si está en forma de kit, recordando omitir las resistencias pull-up, y luego conecte el módulo a su Raspberry Pi como se muestra en la Figura 12.18.

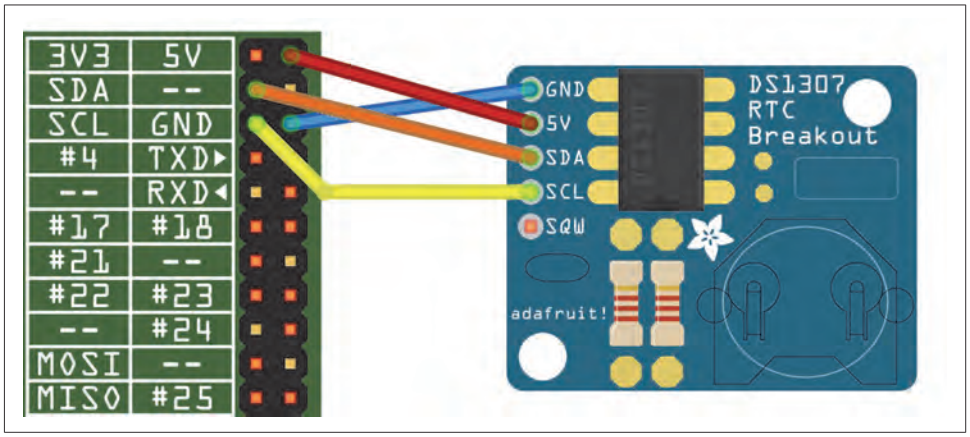


Figura 12.18. *Conexión de un módulo RTC.*

DS1307 es un módulo I2C, por lo que su Raspberry Pi debe configurarse para funcionar con I2C (vea el Capítulo 9.4). Puede comprobar que el dispositivo está visible utilizando herramientas I2C (Capítulo 9.5).

```
$ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

El 68 en la tabla indica que el módulo RTC está conectado al bus I2C en la dirección hexadecimal 68.

Si está utilizando un modelo B de las originales Raspberry Pi, revisión 1, utilice 0 después de la opción y en la línea precedente. Las placas de la revisión 1 eran distintivas al tener una toma de audio negra.

Ahora debe ejecutar los siguientes comandos para que el RTC se pueda usar con un programa llamado *hwclock*.

```
$ sudo modprobe rtc-ds1307
$ sudo bash
$ echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
```

Una vez más, si está utilizando una placa de la revisión 1, cambie `i2c-1` a `i2c-0`. Ahora puede acceder al RTC usando el siguiente comando:

```
$ sudo hwclock -r
Sat 01 Jan 2000 00:08:08 UTC  -0.293998 seconds
```

Como puede ver, el reloj no está configurado.

Para establecer la hora en el módulo RTC, primero debe asegurarse de que su Raspberry Pi tiene la hora adecuada. Si su Pi está conectada a Internet, debería aparecer automáticamente. Puede comprobar esto utilizando el comando `date`:

```
$ date
Tue Aug 20 06:42:47 UTC 2013
```

Si la hora es incorrecta, también puede ajustarla manualmente usando `date` (Capítulo 3.36). Para transferir la hora del sistema Raspberry Pi al módulo RTC, utilice el siguiente comando:

```
$ sudo hwclock -w
```

A continuación, puede leer la hora mediante la opción `-r`:

```
$ sudo hwclock -r
Wed 02 Jan 2013 03:11:43 UTC  -0.179786 seconds
```

El RTC que tiene la hora correcta no tiene sentido a menos que se utilice para establecer la hora correcta del sistema en Linux cuando se reinicie. Para hacer esto debe realizar algunos cambios de configuración.

Primero edite */etc/modules* (usando `sudo nano /etc/modules`) y añada **`rtc-ds1307`** al final de la lista de los módulos. Si ya ha añadido algunos módulos al configurar I2C, SPI y otras opciones, el archivo podría verse así:

```
# /etc/modules: kernel modules to load at boot time.
#
# Este archivo contiene los nombres de los módulos de kernel que se deben
# cargar al arrancar, uno por línea. Las líneas que comienzan por "#" se
# ignoran. Los parámetros pueden ser especificados después del nombre del
# módulo.

snd-bcm2835
i2c-bcm2708
i2c-dev
spidev
rtc-ds1307
```

Después, necesita dos comandos para ejecutarlo automáticamente durante el inicio para que se establezca la hora del sistema. Por lo tanto, edite el archivo */etc/rc.local* usando el comando `sudo nano /etc/rc.local`, y, justo antes de la línea final `exit 0`, inserte las dos siguientes líneas. Recuerde cambiar `i2c-1` a `i2c-0` si está utilizando una Pi de la revisión 1.

```
$ echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
$ sudo hwclock -s
```

Cuando termine el archivo debería ser así:

```
#
# Para habilitar o deshabilitar este script cambie la ejecución de bits.
#
# Por defecto, este script no hace nada.

#Imprime la dirección IP
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi
echo ds1307 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
sudo hwclock -s
exit 0
```

Después, cuando reinicie, su Raspberry Pi debe establecer la hora del sistema desde RTC. Sin embargo, si hay una conexión a Internet disponible, esto tendrá prioridad para configurar la hora.

Observaciones

RTC no es esencial para Raspberry Pi, ya que Raspberry Pi si está conectado a Internet se conectará automáticamente a un servidor de hora de red y establecerá su propia hora. Sin embargo, es posible que no siempre utilice Pi cuando esté conectado a una red, en cuyo caso un RTC de *hardware* será una buena opción.

Para saber más

AB Electronics tiene un RTC que se conecta directamente a la toma GPIO.

Este capítulo se basa en un tutorial de Adafruit.

CAPÍTULO 13

Sensores

13.1 Introducción

En este capítulo usted conocerá diferentes maneras de utilizar sensores de varios tipos que permitirán que Raspberry Pi mida la temperatura, la luz y mucho más.

Comparando con una placa como Arduino, Raspberry Pi carece de entradas analógicas. Esto significa que, en muchos sensores, será necesario utilizar un *hardware* adicional conversor de analógico a digital (ADC). Afortunadamente, esto es relativamente fácil de hacer. También es posible utilizar sensores resistivos con un condensador y un par de resistores.

La mayoría de estos capítulos requerirán el uso de una placa de pruebas sin soldadura y cables puente macho-hembra (consulte el Capítulo 9.9).

13.2 Usar sensores resistivos

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea conectar una resistencia variable a Raspberry Pi y medir su resistencia de modo que usted pueda utilizar la posición de la perilla de la resistencia variable (potenciómetro) en su programa de Python.

Solución

Usted puede medir la resistencia de una Raspberry Pi usando nada más que un condensador, un par de resistores y dos pines GPIO. En este caso, será capaz de estimar la posición de la perilla de una pequeña resistencia variable (potenciómetro) mediante la medición de la resistencia desde su contacto deslizante hasta su extremo.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipo para prototipos", página 474)
- Potenciómetro de 10 kΩ (vea "Resistores y condensadores", página 474)
- Dos resistores de 1 kΩ (vea "Resistores y condensadores", página 474)
- Condensador de 330 nF (vea "Resistores y condensadores", página 474)

La Figura 13.1 muestra la disposición de los componentes de la placa de pruebas.

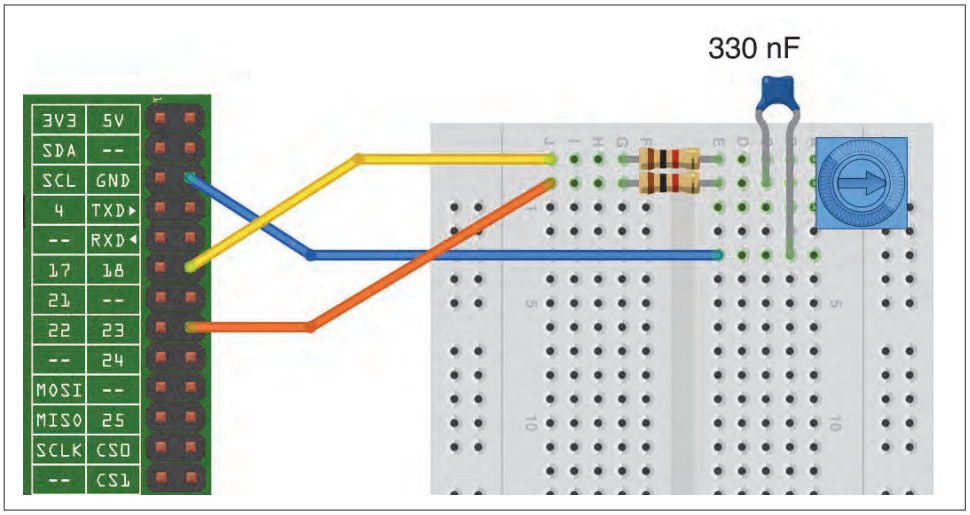


Figura 13.1. *Medición de la resistencia de Raspberry Pi.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *pot_step.py*.

```
import RPi.GPIO as GPIO
import time, math

C = 0.36 # uF
R1 = 1000 # Ohms

GPIO.setmode(GPIO.BCM)
```

```
#El pin a carga el condensador a través de un resistor fijo de 1 k
# y del recipiente en serie
# El pin b descarga el condensador a través de un resistor fijo de 1 k
a_pin = 18
b_pin = 23

# Descarga el condensador, dejándolo listo para comenzar a llenarlo
def discharge():
    GPIO.setup(a_pin, GPIO.IN)
    GPIO.setup(b_pin, GPIO.OUT)
    GPIO.output(b_pin, False)
    time.sleep(0.1)

#Devuelve el tiempo tomado (uS) para que la tensión en el condensador
# cuente como una entrada digital ALTA, que es 1,65 V o más
def charge_time():
    GPIO.setup(b_pin, GPIO.IN)
    GPIO.setup(a_pin, GPIO.OUT)
    GPIO.output(a_pin, True)
    t1 = time.time()
    while not GPIO.input(b_pin):
       pass
    t2 = time.time()
    return (t2 - t1) * 1000000

# Toma una lectura analógica como el tiempo de carga después de
# la primera descarga del condensador
def analog_read():
    discharge()
    t = charge_time()
    discharge()
    return t

#Convierte el tiempo de cargar el condensador en un valor de resistencia
#Para reducir los errores, hágalo 100 veces y tome el promedio
def read_resistance():
    n = 10
    total = 0;
    for i in range(1, n):
        total = total + analog_read()
    t = total / float(n)
    T = t * 0.632 * 3.3
    r = (T / C) - R1
    return r
try:
    while True:
        print(read_resistance())
        time.sleep(0.5)
finally:
    GPIO.cleanup()
```

Al ejecutar el programa, debería ver una salida como esta:

```
$ sudo python pot_step.py
10049.2157936
10105.1559448
10158.6098671
11331.0049375
10162.6154582
10156.8142573
9501.27855937
8216.17444356
```

La lectura variará a medida que gire la perilla del potenciómetro. Idealmente, la lectura de la resistencia variaría entre 0 Ω y 10.000 Ω, pero en la práctica habrá algún error. Es posible que desee modificar el valor de la constante C en la parte superior del programa si desea lecturas más precisas, pero recuerde que el valor de C probablemente tendrá que ser cambiado si pone otro condensador, incluso si el condensador era del mismo valor.

Observaciones

Para explicar cómo funciona este programa, primero necesito explicar cómo se puede usar la técnica *step response* para medir la resistencia de la resistencia variable.

La Figura 13.2 muestra el diagrama esquemático del ejercicio.

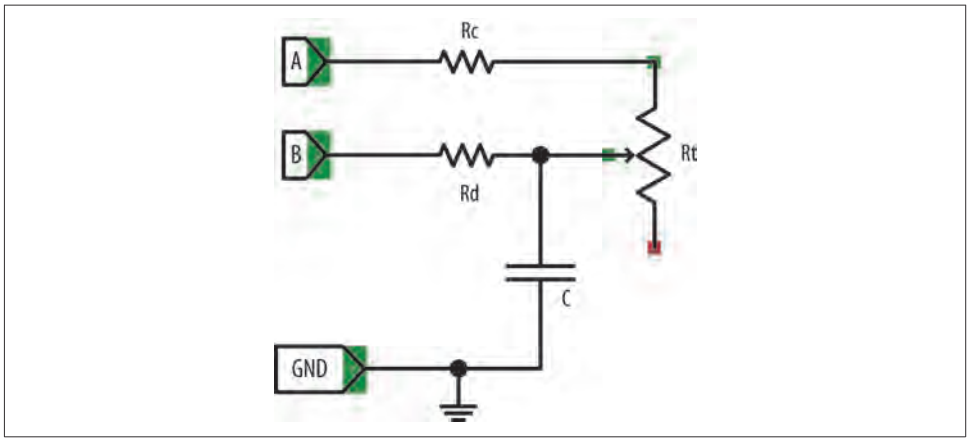


Figura 13.2. *Medición de la resistencia mediante step response.*

Esta forma de hacer las cosas se denomina *step response* porque funciona al ver cómo responde el circuito al cambio de paso (*step*) cuando la salida cambia de bajo a alto.

Se puede pensar en un condensador como en un almacén de electricidad que se llena de carga y con el que el voltaje a través de él aumenta. No puede medir el voltaje directamente porque Raspberry Pi no tiene un conversor analógico-digital (ADC). Sin embargo, puede calcular el tiempo que tarda el condensador en llenarse con carga para que la medida que se consiga supere los 1,65 V aproximadamente y que constituya una entrada digital alta. La velocidad a la que el condensador se llena de carga depende del valor de la resistencia variable (Rt). Cuando más baja es la resistencia, más rápido se llena el condensador de carga y el voltaje aumenta.

Para poder obtener una buena lectura, debe vaciar el condensador cada vez antes de llevar a cabo una lectura. En la Figura 13.2, la conexión A se utiliza para cargar el condensador a través de Rc y Rt, y la conexión B se usa para descargar (vaciar) el condensador a través de Rd. Las resistencias Rc y Rd se utilizan para evitar que fluya una corriente excesiva cuando el condensador se carga y descarga a través de los pines GPIO, relativamente frágiles, de Raspberry Pi.

Los pasos implicados para tomar una lectura son primero descargar el condensador a través de Rd y después dejarlo cargar a través de Rc y Rt.

Para descargarlo, la conexión A (GPIO 18) está configurada para ser una entrada, desconectando efectivamente Rc y Rt del circuito. La conexión B (GPIO 23) se configura entonces como salida y baja. Esto se mantiene así durante 100 milisegundos para vaciar el condensador.

Ahora que el condensador está vacío, puede empezar a permitir que la carga fluya hacia él estableciendo la conexión B para que sea una entrada (efectivamente, desconectándola) y luego permitiendo que la conexión A sea una salida alta a 3,3 V. El condensador C empezará a cargarse a través de Rc y Rt. El bucle `while` no hará nada hasta que el voltaje en la conexión B cambie de bajo a alto en aproximadamente 1,65 V.

En ese punto, se devuelve el tiempo tomado.

La Figura 13.3 muestra cómo una resistencia y un condensador, en este tipo de disposición, cargan y descargan cuando el voltaje se conmuta entre bajo y alto.

Puede ver que el voltaje en el condensador aumenta rápidamente al principio, pero luego se apaga cuando el condensador se llena. Afortunadamente, usted está interesado en el área de la curva hasta que el condensador alcance aproximadamente los 1,65 V, lo que significa que el tiempo tomado para que el voltaje a través del condensador suba hasta este punto es aproximadamente proporcional a la resistencia Rt y, por lo tanto, a la posición de la perilla.

Este enfoque no es muy preciso, pero es muy barato y fácil de usar. La inexactitud se debe en gran parte a que los condensadores son solo exactos al 10 %.

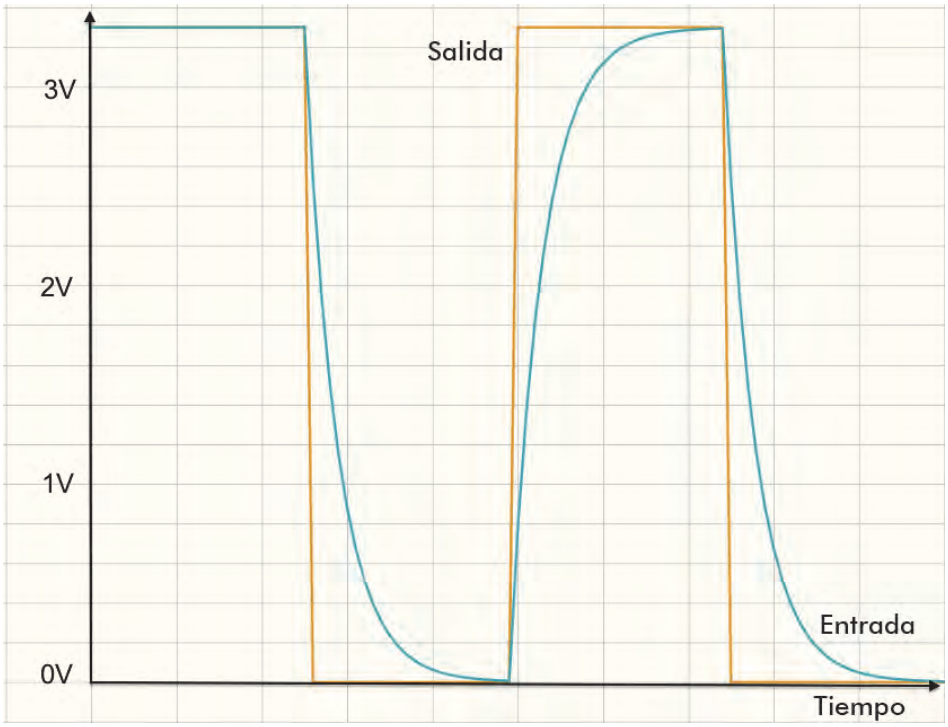


Figura 13.3. *Carga y descarga de un condensador.*

Para saber más

El uso de *step response* funciona bien con todo tipo de sensores resistivos para la luz (Capítulo 13.3), temperatura (Capítulo 13.4) e incluso detección de gas (Capítulo 13.5).

Para mediciones más precisas de la posición del potenciómetro, vaya al Capítulo 13.6, donde se usa con un conversor ADC.

13.3 Medir la luz

Problema

Quiere medir la intensidad de la luz con Raspberry Pi y una fotorresistencia.

Solución

Use lo mismo que en el Capítulo 13.2, pero reemplace el potenciómetro por una fotorresistencia.

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea “Equipos para prototipos”, página 474)
- Fotorresistencia (vea “Resistores y condensadores” en la página 474)
- Dos resistores de 1 kΩ (vea “Resistores y condensadores” en la página 474)
- Condensador de 330 nF (vea “Resistores y condensadores”, página 474)

Todas estas piezas están incluidas en el kit electrónico de inicio de Raspberry Pi de Monk Makes (vea “Equipos para prototipos” en la página 474).

La Figura 13.4 muestra la disposición de los componentes en la placa de pruebas.

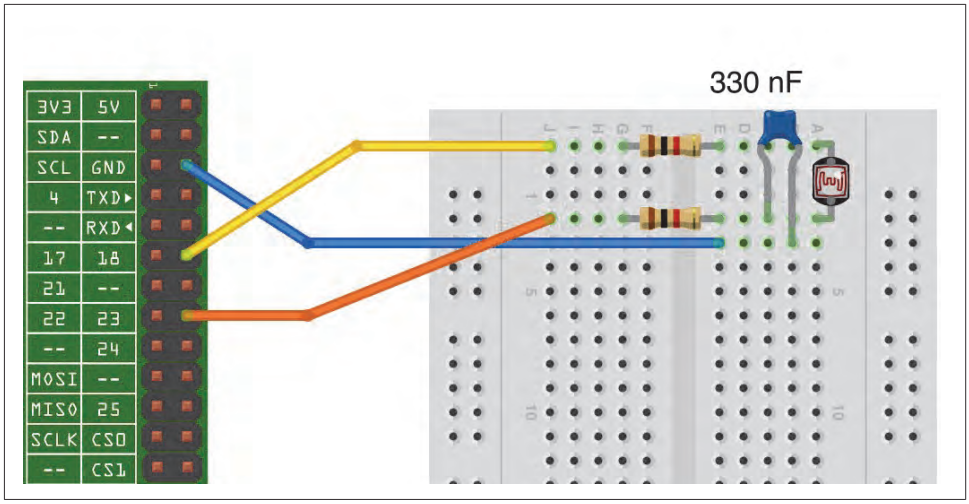


Figura 13.4. *Medir la luz en Raspberry Pi.*

Utilizando el mismo programa que en el Capítulo 13.2 (*pot_step.py*) verá que la salida variará a medida que mueva su mano sobre la fotorresistencia para cortar parte de la luz.

Esta solución proporciona lecturas relativamente fiables de los niveles de luz. Como una adaptación de la solución general para el uso de sensores resistivos (Capítulo 13.2), también se encarga de medir una resistencia de 0 Ω sin ningún riesgo de dañar los pines GPIO de la Raspberry Pi. Puesto que una fotorresistencia nunca va a tener una resistencia cero, puede acabar con ambos resistores de 1 kΩ y con uno de los pines GPIO, y hacer el esquema más simple como se muestra en la Figura 13.5.

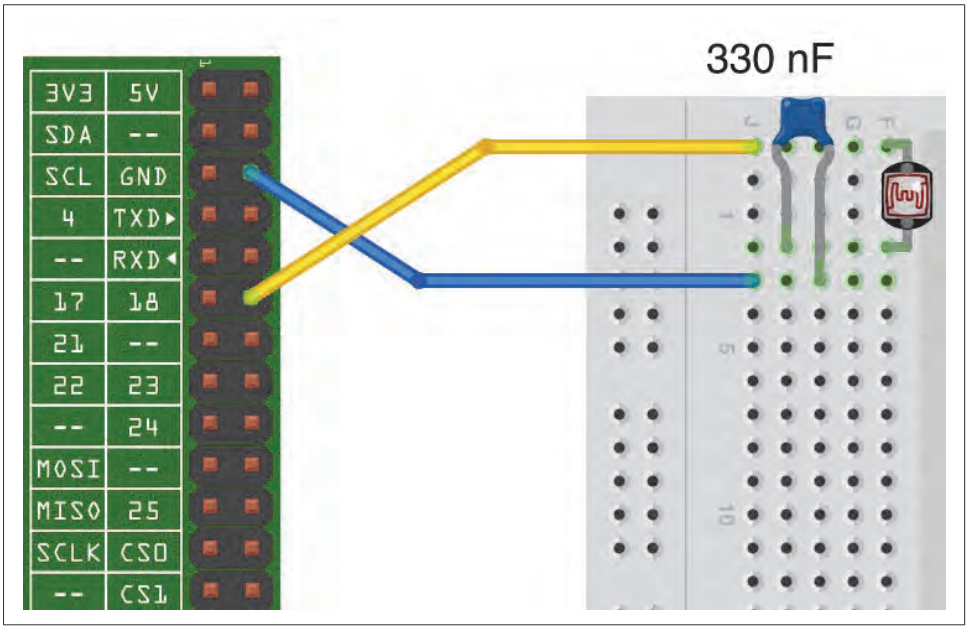


Figura 13.5. *Un sensor de luz simplificado.*

En esta variación, se utilizará el mismo pin GPIO para cargar y descargar el condensador y para comprobar si está por encima del umbral alto (HIGH) de entrada. El programa se puede encontrar en *photoresistor_step.py*.

```
import RPi.GPIO as GPIO
import time


GPIO.setmode(GPIO.BCM)

pin = 18

def discharge():
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, False)
    time.sleep(0.1)


#Devuelve el tiempo que tarda el condensador en alcanzar ALTO, que es
#1,65V o más
def charge_time():
    t1 = time.time()
    GPIO.setup(pin, GPIO.IN)
    while not GPIO.input(pin):
        #carga por 1 ms
         GPIO.setup(pin, GPIO.OUT)
```

```
            GPIO.output(pin, True)
            time.sleep(0.001)
            # establece la entrada para probarlo de nuevo
            GPIO.setup(pin, GPIO.IN)
            time.sleep(0.001)
        t2 = time.time()
        return (t2 - t1) * 1000000

# Toma una lectura analógica como el tiempo necesario para cargar C
def analog_read():
    discharge()
    return charge_time()


while True:
    print(analog_read())
    time.sleep(0.5)
```

Usando solo un pin, después de descargar el condensador a través de la fotorresistencia, la función `charge_time` carga el condensador a través de la fotorresistencia durante solo 1 ms. Establece el pin para que sea una entrada y comprueba si está sobre el umbral alto, y lo repite hasta que sea alto.

Observaciones

En una fotorresistencia la resistencia varía dependiendo de la cantidad de luz que entre por su ventana transparente. Cuanto más brillante sea la luz, menor será la resistencia. Típicamente, la resistencia varía entre aproximadamente 1 kΩ en luz brillante hasta 100 kΩ en completa oscuridad.

Los sensores solo pueden dar una idea aproximada del nivel de luz.

Para saber más

También puede usar un ADC con la fotorresistencia (Capítulo 13.6).

13.4 Medir la temperatura con un termistor

Problema

Desea medir la temperatura usando un termistor.

Solución

La resistencia del termistor varía con la temperatura. Utilice el método *step response* (Capítulo 13.2) para medir la resistencia del termistor y calcular la temperatura.

Para ello necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Termistor de 1 k (vea "Resistores y condensadores" en la página 474)
- Dos resistores de 1 kΩ (vea "Resistores y condensadores" en la página 474)
- Condensador de 330 nF (vea "Resistores y condensadores", página 474)

Todas estas piezas están incluidas en el kit electrónico de inicio de Raspberry Pi de Monk Makes (vea "Equipos para prototipos" en la página 474). Cuando obtenga su termistor asegúrese de conocer sus valores de Beta y R0 (resistencia a 25 °C) y de que es un dispositivo de Coeficiente de Temperatura Negativa (NTC).

La Figura 13.6 muestra la disposición de la placa de pruebas.

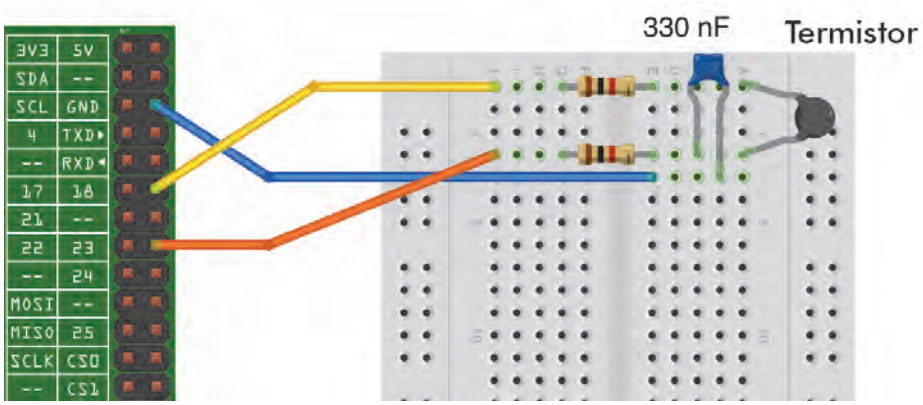


Figura 13.6. *Disposición de la placa de pruebas para el uso de un termistor.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código de la página web del libro, donde pone *thermistor.py*.

```
import RPi.GPIO as GPIO
import time, math

C = 0.36 # uF
R1 = 1000 # Ohms
B = 3800.0 # La constante del tesmistor
R0 = 1100.0 # La resistencia del termistor a 25 °C

GPIO.setmode(GPIO.BCM)

#El pin carga el condensador a través de un resistor fijo de 1 k
# y del termistor en serie
# El pin b descarga el condensador a través de un resistor fijo de 1 k
```

```
a_pin = 18
b_pin = 23

# Descarga el condensador, dejándolo listo para comenzar a llenarlo
def discharge():
    GPIO.setup(a_pin, GPIO.IN)
    GPIO.setup(b_pin, GPIO.OUT)
    GPIO.output(b_pin, False)
    time.sleep(0.1)

# Devuelve el tiempo empleado (uS) para que C se cargue,
# que es 1,65 V o más
def charge_time():
    GPIO.setup(b_pin, GPIO.IN)
    GPIO.setup(a_pin, GPIO.OUT)
    GPIO.output(a_pin, True)
    t1 = time.time()
    while not GPIO.input(b_pin):
       pass
    t2 = time.time()
    return (t2 - t1) * 1000000

# Toma una lectura analógica del tiempo de carga
def analog_read():
    discharge()
    t = charge_time()
    discharge()
    return t

# Convierte el tiempo para cargar el condensador en un valor de resistencia
# Para reducir errores, hágalo 10 veces y tome el promedio.
def read_resistance():
    n = 10
    total = 0;
    for i in range(1, n):
        total = total + analog_read()
    t = total / float(n)
    T = t * 0.632 * 3.3
    r = (T / C) - R1
    return r

def read_temp_c():
    R = read_resistance()
    t0 = 273.15     # 0 grados C en K
    t25 = t0 + 25.0 # 25 grados C en K
    # Búsque en Google la ecuación Steinhart-Hart
    inv_T = 1/t25 + 1/B * math.log(R/R0)
    T = (1/inv_T - t0)
    return T

try:
    while True:
```

```
            print(read_temp_c())
            time.sleep(0.5)
    finally:
        GPIO.cleanup()
```

Al ejecutar el programa verá una serie de medidas de temperatura en grados C. Para convertirlo a grados F use la fórmula *Tf* = *Tc* * 9 / 5 + 32.

```
$ sudo python thermistor.py
18.3040458984
17.8302664759
17.3917856854
17.9286173793
```

Observaciones

La mayor parte del código de este proyecto es el mismo que el del Capítulo 13.2. La parte nueva es la función `read_temp_c`, que convierte el valor de la resistencia en una temperatura en grados C.

Calcular la temperatura de la resistencia de un termistor requiere algunas matemáticas bastante complejas, como la llamada ecuación Steinhart-Hart. Esta ecuación necesita saber dos cosas sobre el termistor: su resistencia a 25 grados C (llamada T0 o a veces T25) y una constante para el termistor llamada Beta, o a veces solo B. Si usa un termistor diferente, necesitará asignar estos valores a las variables B y R0 en la parte superior del programa.

Tenga en cuenta que un condensador normalmente solo tiene una precisión del 10 %, y que los termistores son igualmente inexactos en su valor de R0, por lo que, para obtener lecturas útiles del termistor, tendrá que ajustar las variables C y R0.

Ajuste C primero, usando un resistor de 1 kΩ con un 1 % de precisión, en lugar del recipiente en el Capítulo 13.2. Después, ajuste R0 para obtener lecturas que coincidan con un termómetro preciso.

Para saber más

Para medir la temperatura usando un TMP36, vaya al Capítulo 13.9.

Para medir la temperatura usando un sensor de temperatura digital (DS18B20), consulte el Capítulo 13.12.

Para medir la temperatura usando un Sense HAT, revise el Capítulo 13.11.

13.5 Detectar metano

Problema

Quiere medir los niveles de gas usando un sensor de metano.

Solución

Existen sensores de gas resistivos a bajo coste que pueden conectarse fácilmente a Raspberry Pi para detectar gases como el metano. Puede utilizar el método *step response* que usó en el Capítulo 13.2.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipo para prototipos", página 474)
- Sensor de metano (vea "Módulos" en la página 476)
- Dos resistores de 1 kΩ (vea "Resistores y condensadores" en la página 474)
- Condensador de 330 nF (vea "Resistores y condensadores" en la página 474)

El sensor contiene un elemento de calentamiento que necesita de 5 V hasta 150 mA. Raspberry Pi es capaz de proporcionar esto mientras su fuente de alimentación pueda suministrar 150 mA adicionales.

El módulo del sensor tiene unas patas bastante gruesas, demasiado para que quepan en los orificios de la placa. Una manera de solucionar esto es soldar pequeños trozos de alambre de núcleo sólido a cada cable (Figura 13.7). Otra es comprar un sensor de gas SparkFun.

Figura 13.7. *Cables de soldadura en el sensor de gas.*

Conecte la placa de pruebas como se muestra en la Figura 13.8 si está utilizando SparkFun, o como se muestra en la Figura 13.9 si ha soldado cables más largos al sensor de gas.

Tenga en cuenta que la conexión directa que se muestra en la Figura 13.9 usa el mismo símbolo que para la placa SparkFun. Pero si mira cuidadosamente, las conexiones están en los seis pines del sensor y no en los cuatro de la placa.

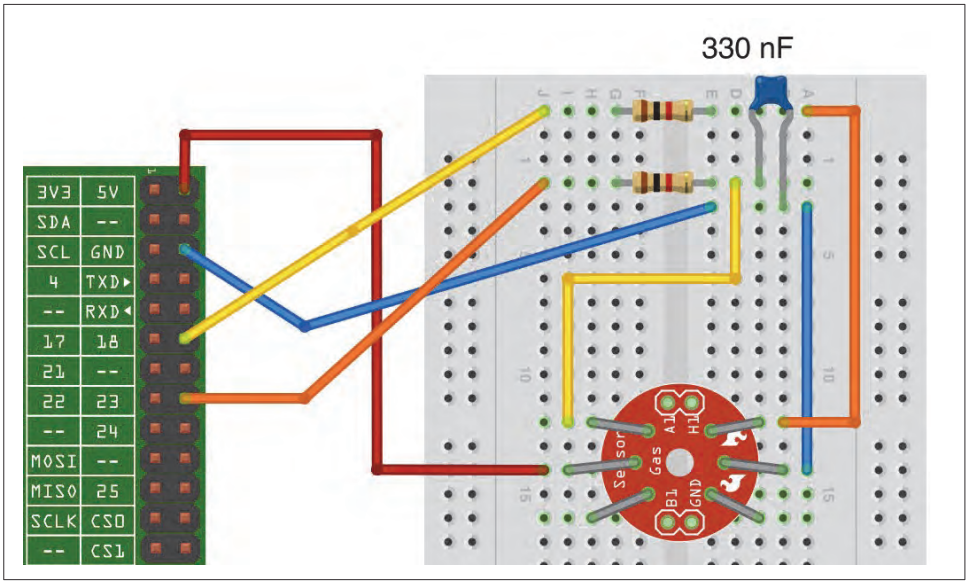


Figura 13.8. *Conexión de un sensor de gas metano a Raspberry Pi (placa).*

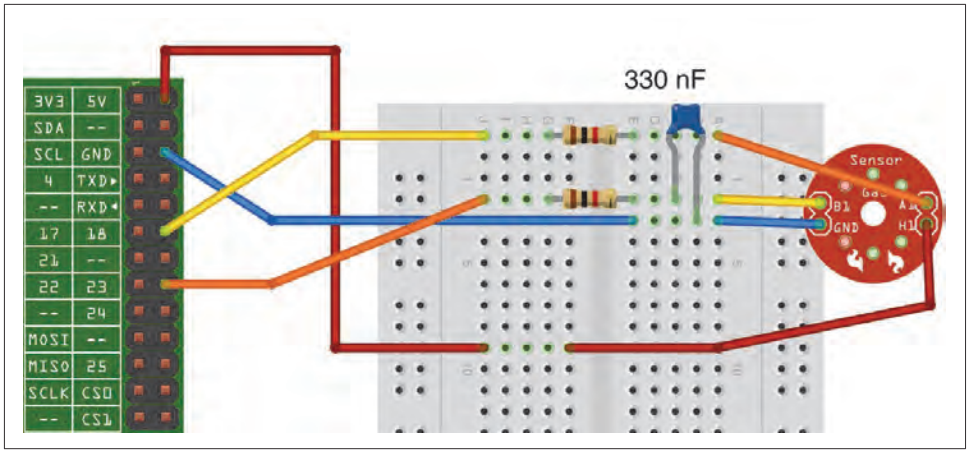


Figura 13.9. *Conexión de un sensor de gas metano a Raspberry Pi (directo).*

Puede utilizar el mismo programa que en el Capítulo 13.2, y puede probar el sensor de metano respirando en él. Verá las lecturas de las caídas del sensor cuando respire.

Observaciones

El uso más obvio de un sensor de gas metano son los proyectos de *detección de pedos*. Un uso más serio sería el de detectar fugas de gas natural. Podría, por ejemplo, imaginar un proyecto de Raspberry Pi con el que controle su casa mediante su monitorización con varios sensores. Podría enviarle un correo electrónico, mientras está de vacaciones, informándole de que su casa está a punto de explotar.

Estos tipos de sensores (Figura 13.10) utilizan un elemento calefactor que calienta una superficie resistiva impregnada con un catalizador sensible a un gas particular. Cuando este gas está presente la resistencia de la capa del catalizador cambia.

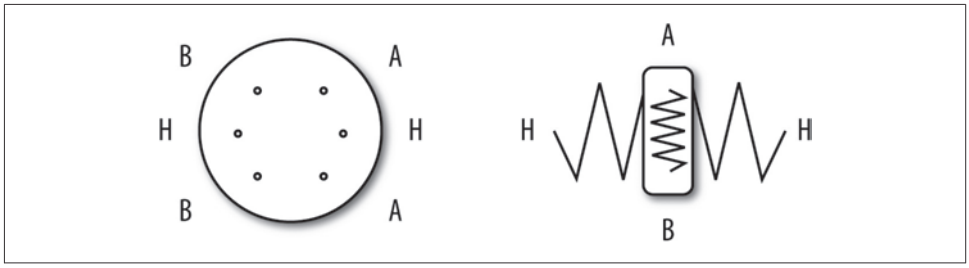


Figura 13.10. *Un sensor de gas metano.*

Tanto el calentador como la superficie de detección son solo resistencias eléctricas. Así que ambas pueden conectarse de cualquier manera.

En particular, este sensor de gas es más sensible al metano, pero también detectará otros gases en menor grado. Es por eso que la respiración en el sensor altera la lectura, ya que los individuos sanos normalmente no respiran metano. El efecto del enfriamiento sobre el elemento también puede tener algún efecto.

Para saber más

La hoja de datos para este sensor se puede encontrar en *http://bit.ly/1gYupsu*. Esto le dará todo tipo de información útil sobre la sensibilidad del sensor a varios gases.

Hay una gama de estos sensores de bajo coste disponibles para detectar diferentes gases. Para obtener una lista de los sensores ofrecidos por SparkFun, consulte la página web de SparkFun.

13.6 Medir el voltaje

Problema

Desea medir el voltaje analógico.

Solución

El conector GPIO de Raspberry Pi tiene solo entradas digitales. Si desea medir un voltaje, debe utilizar un convertidor analógico a digital (ADC) independiente.

Utilice el chip ADC MCP3008 de ocho canales. Este chip tiene en realidad ocho entradas analógicas, por lo que puede conectar hasta ocho sensores a uno de estos y la interfaz al chip usando la interfaz SPI de Raspberry Pi.

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- ADC IC MCP3008 de ocho canales (vea "Circuitos integrados", página 475)
- Potenciómetro de 10 kΩ (vea "Resistores y condensadores", página 474)

La Figura 13.11 muestra el diseño de la placa de pruebas con ese chip. Asegúrese de que coloca el chip de manera correcta. La pequeña marca debe estar hacia arriba de la placa de pruebas.

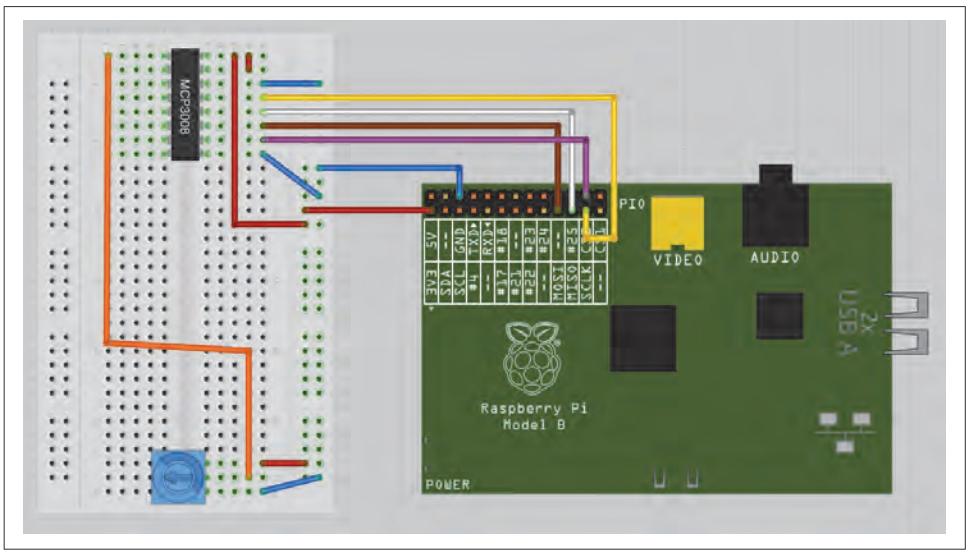


Figura 13.11. *Uso de un chip ADC MCP3008 con Raspberry Pi.*

La resistencia variable tiene un extremo conectado a 3,3 V y el otro a tierra, lo que permite que la conexión media se ajuste a cualquier voltaje entre 0 y 3,3 V.

Antes de probar el programa, asegúrese de tener el SPI habilitado y la biblioteca SPI de Python instalada (Capítulo 9.6).

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *adc_test.py*.

```
import spidev, time

spi = spidev.SpiDev()
spi.open(0, 0)

def analog_read(channel):
    r = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_out = ((r[1]&3) << 8) + r[2]
    return adc_out

while True:
    reading = analog_read(0)
    voltage = reading * 3.3 / 1024
    print("Reading=%d\tVoltage=%f" % (reading, voltage))
    time.sleep(1)
```

La parte interesante del programa está contenida en la función `analog_read`. Esto toma un parámetro que debe estar entre 0 y 7, y especifica cuál de las ocho entradas analógicas del chip debe leerse.

La manipulación de bits configura una solicitud para el canal apropiado y luego envía los bits al MCP3008, que lee los datos resultantes:

```
$ sudo python adc_test.py
Reading=0       Voltage=0.000000
Reading=126     Voltage=0.406055
Reading=221     Voltage=0.712207
Reading=305     Voltage=0.982910
Reading=431     Voltage=1.388965
Reading=527     Voltage=1.698340
Reading=724     Voltage=2.333203
Reading=927     Voltage=2.987402
Reading=1020    Voltage=3.287109
Reading=1022    Voltage=3.293555
```

Observaciones

El MCP3008 tiene un ADC de 10 bits, por lo que, al tomar una lectura, le da un número entre 0 y 1023. El programa de prueba convierte esto en una lectura de voltaje multiplicando la lectura por el rango de voltaje (3,3 V) y dividiéndolo por 1.024.

Puede combinar cualquiera de los siguientes capítulos que usan el MCP3008 para permitir lecturas de hasta ocho sensores.

También puede utilizar sensores resistivos con el MCP3008 combinándolos con un resistor de valor fijo y fijándolos como divisor de voltaje (consulte el Capítulo 13.7 y el Capítulo 13.8).

Para saber más

Si solo está interesado en detectar el giro de una perilla, puede utilizar un codificador rotatorio (Capítulo 12.8).

También puede detectar la posición del aparato sin utilizar un chip ADC mediante el método *step response* (Capítulo 13.2).

Consulte la hoja de datos del MCP3008.

El Explorer HAT Pro de Pimoroni también tiene un ADC (Capítulo 9.18).

13.7 Reducir el voltaje para medir

Problema

Desea medir un voltaje, pero es más alto de los 3,3 V permitido con un MCP3008 (Capítulo 13.6).

Solución

Utilice un par de resistores para que actúen como divisores de voltaje para reducirlo a un rango adecuado.

Para eso necesitará:

- Placa de prueba y cables puente (vea “Equipos para prototipos”, página 474)
- Chip ADC IC MCP3008 de ocho canales (vea “Circuitos integrados”, página 475)
- Resistor de 10 kΩ (vea “Resistores y condensadores” en la página 474)
- Resistor de 3,3 kΩ (vea “Resistores y condensadores” en la página 474)
- Batería de 9 V y un cable de clip

La Figura 13.12 muestra la disposición para esto, usando una placa de pruebas. La configuración medirá el voltaje de la batería.

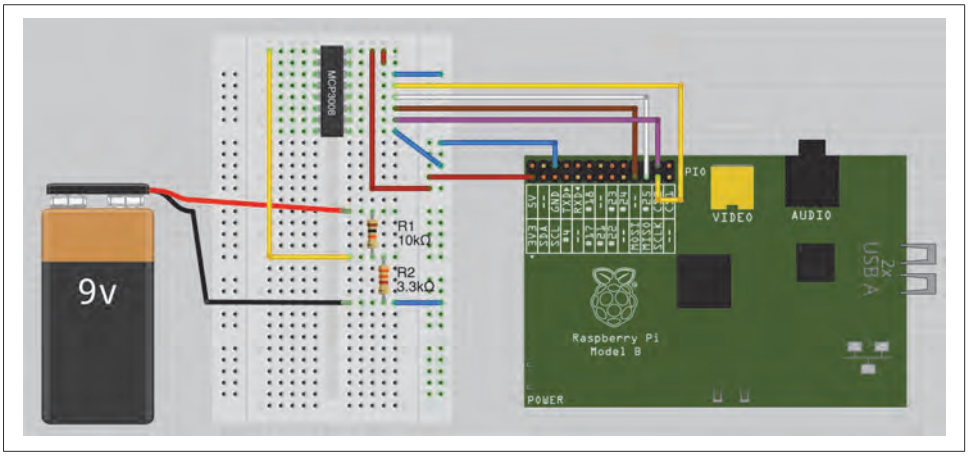


Figura 13.12. *Reducción del voltaje de las entradas analógicas.*

Nunca utilice esto para medir AC de alta tensión o, para esta materia, ningún tipo de AC. Es solo para CC de baja tensión.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *adc_scaled.py*.

```
import spidev

R1 = 10000.0
R2 = 3300.0

spi = spidev.SpiDev()
spi.open(0,0)

def analog_read(channel):
    r = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_out = ((r[1]&3) << 8) + r[2]
    return adc_out

reading = analog_read(0)
voltage_adc = reading * 3.3 / 1024
voltage_actual =  voltage_adc / (R2 / (R1 + R2))
print("Battery Voltage=" + str(voltage_actual))
```

El programa es muy similar al del Capítulo 13.6. La principal diferencia es la escala, usando los valores de las dos resistencias. Los valores de estas dos resistencias se mantienen en las variables *R1* y *R2*.

Al ejecutar el programa se mostrará el voltaje de la batería:

```
$ sudo python adc_scaled.py
Battery Voltage=8.62421875
```

Lea el apartado «Observaciones» cuidadosamente antes de conectar cualquier cosa superior a 9 V o podrá destruir el MCP3008.

Observaciones

Esta disposición de resistencias se denomina *divisor de voltaje* o a veces *divisor de potencial* (Figura 13.13). La fórmula para calcular la tensión de salida, dada la tensión de entrada y los valores de las dos resistencias, es:

Vout = Vin * R2 / (R1 + R2)

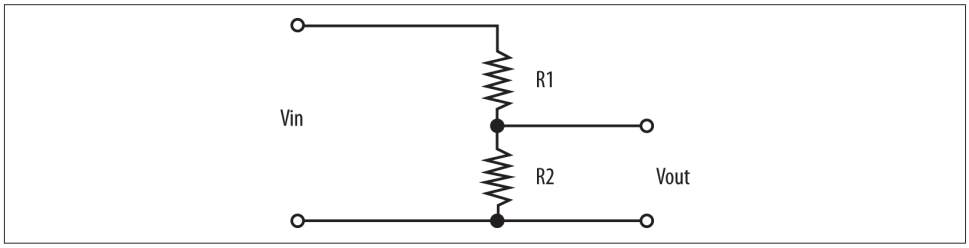


Figura 13.13. *Divisor de voltaje.*

Esto significa que si R1 y R2 tuviesen el mismo valor (es decir, 1 kΩ), *Vout* sería la mitad de *Vin*.

Al elegir R1 y R2, también es necesario considerar la corriente que fluye a través de R1 y R2. Esto será Vin / (R1 + R2). En el ejemplo anterior, R1 es 10 kΩ y R2 es 3,3 kΩ. Así que la corriente que fluye será 9 V / 13,3 kΩ = 0,68 mA. Eso es bajo, pero suficiente para drenar la batería, por lo que no lo deje conectado todo el tiempo.

Para saber más

Para evitar las matemáticas, puede utilizar una calculadora de resistencias en línea.

El divisor de voltaje también se utiliza para convertir la resistencia en voltaje cuando se utiliza un sensor resistivo con un ADC (Capítulo 13.8).

13.8 Usar sensores resistivos con un ADC

Problema

Tiene un sensor resistivo que quiere utilizar con un chip ADC MCP3008.

Solución

Utilice un divisor de potencial con una resistencia fija y el sensor resistivo para convertir la resistencia del sensor en un voltaje que pueda medirse con el ADC.

Como ejemplo, puede rehacer el proyecto del sensor de luz del Capítulo 13.3 y usar el MCP3008 en lugar de la técnica *step response*.

Para hacer eso necesitará:

- Placa de pruebas y cables puente (vea "Equipo para prototipos", página 474)
- Chip ADC IC MCP3008 de ocho canales (vea "Circuitos integrados", página 475)
- Resistor de 10 kΩ (vea "Resistores y condensadores" en la página 474)
- Fotorresistencia (vea "Resistores y condensadores" en la página 474)

La Figura 13.14 muestra la disposición para esto, usando una placa de pruebas.

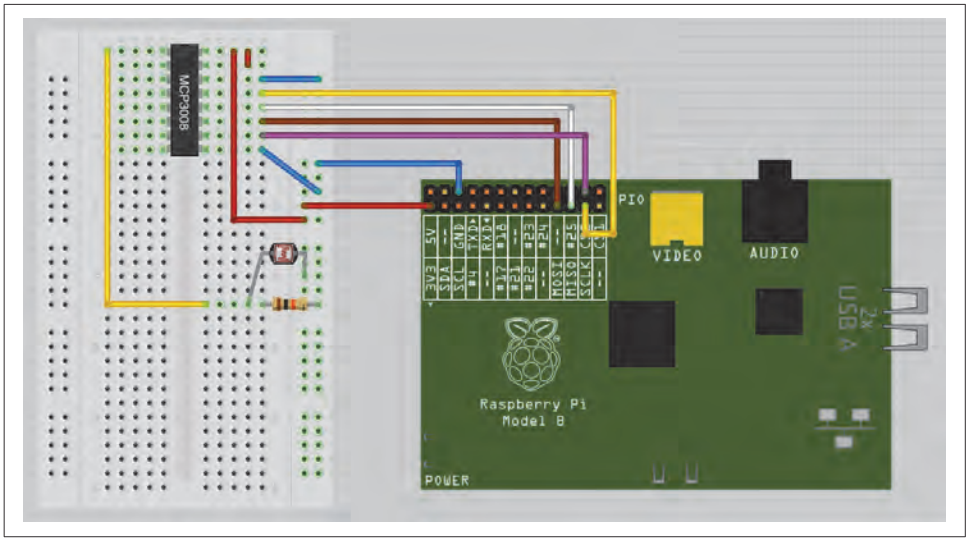


Figura 13.14. *Usar una fotorresistencia con ADC.*

Puede utilizar el mismo programa que en el Capítulo 13.6 (*adc_test.py*). Cubrir el sensor de luz con la mano cambiará las lecturas. También necesitará configurar el SPI en su Raspberry Pi, así que, si aún no lo ha hecho, siga el Capítulo 9.6.

```
$ sudo python adc_test.py

Reading=341    Voltage=1.098926
Reading=342    Voltage=1.102148
Reading=227    Voltage=0.731543
Reading=81     Voltage=0.261035
Reading=86     Voltage=0.277148
```

Estas lecturas pueden ser bastante diferentes dependiendo de su fotorresistencia, pero lo importante es que la salida cambie a medida que cambia el nivel de luz.

Observaciones

La elección de la resistencia de valor fijo no es muy crítica. Si el valor es demasiado alto o bajo, encontrará que el rango de lectura es bastante estrecho. Seleccione un valor de resistencia entre la resistencia mínima y la máxima del sensor. Es posible que necesite experimentar con unos pocos resistores antes de decidir cual se adapta mejor a su sensor en la gama de lecturas que le interese. En caso de duda, comience con uno de 10 kΩ y vea cómo funciona.

Puede intercambiar la fotorresistencia por casi cualquier sensor resistivo. Así que, por ejemplo, podría utilizar el sensor de gas del Capítulo 13.5.

Para saber más

Para medir la intensidad de la luz sin la complicación de un ADC revise el capítulo 13.3.

Para ver un ejemplo de uso de más de un canal del ADC a la vez consulte el Capítulo 13.13.

13.9 Medir la temperatura con un ADC

Problema

Desea medir la temperatura usando un TMP36 y un conversor ADC.

Solución

Use un chip ADC MCP3008.

Sin embargo, a menos que necesite más de un canal analógico, debe considerar utilizar el sensor de temperatura digital DS18B20, que es más preciso y no necesita ningún chip ADC separado (Capítulo 13.12).

Para esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Chip ADC IC MCP3008 de ocho canales (vea "Circuitos integrados", página 475)
- Sensor de temperatura TMP36 (vea "Circuitos integrados", página 475)

La Figura 13.15 muestra la disposición usando una placa de pruebas.

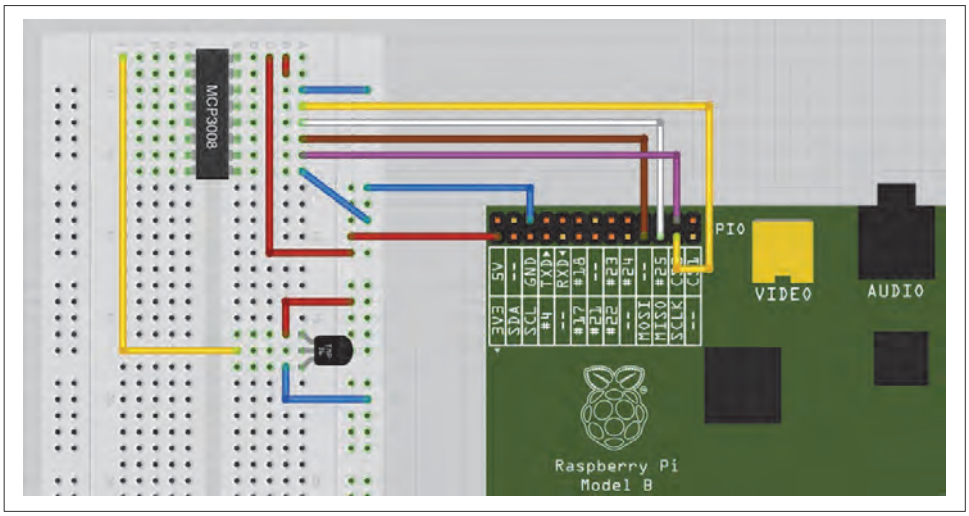


Figura 13.15. *Uso de un TMP36 con un ADC.*

Asegúrese de que el TMP36 está colocado de manera correcta. Un lado del sensor es plano, mientras que el otro es curvo.

Tendrá que configurar el SPI en su Raspberry Pi, así que, si aún no lo ha hecho, siga el Capítulo 9.6.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *adc_tmp36.py*.

```
import spidev, time

spi = spidev.SpiDev()
spi.open(0,0)

def analog_read(channel):
    r = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_out = ((r[1]&3) << 8) + r[2]
    return adc_out
```

```
while True:
    reading = analog_read(0)
    voltage = reading * 3.3 / 1024
    temp_c = voltage * 100 - 50
    temp_f = temp_c * 9.0 / 5.0 + 32
    print("Temp C=%f\t\tTemp f=%f" % (temp_c, temp_f))
    time.sleep(1)
```

El programa se basa en el del Capítulo 13.9. Unas pocas matemáticas adicionales calculan la temperatura en grados Celsius y Fahrenheit:

```
$ sudo python adc_tmp36.py
Temp C=19.287109        Temp f=66.716797
Temp C=18.642578        Temp f=65.556641
Temp C=18.964844        Temp f=66.136719
Temp C=20.253906        Temp f=68.457031
Temp C=20.898438        Temp f=69.617188
Temp C=20.576172        Temp f=69.037109
Temp C=21.865234        Temp f=71.357422
Temp C=23.154297        Temp f=73.677734
Temp C=23.476562        Temp f=74.257812
Temp C=23.476562        Temp f=74.257812
Temp C=24.121094        Temp f=75.417969
Temp C=24.443359        Temp f=75.998047
Temp C=25.087891        Temp f=77.158203
```

Observaciones

El TMP36 emite una tensión que es proporcional a la temperatura. De acuerdo con la hoja de datos del TMP36, la temperatura en grados C se calcula como la tensión (en voltios) 100 veces menos 50.

El TMP36 va bien para medir la temperatura aproximada, pero se especifica que tiene una precisión de solo 2 grados C. Esto solo empeorará si se unen cables largos a la misma. En cierta medida, puede calibrar un dispositivo individual, pero para una mejor precisión utilice DS18B20 (Capítulo 13.12), que tiene una precisión de 0,5 % en un rango de temperatura de −10 a +85 grados C. Un dispositivo digital no debe sufrir ninguna pérdida de precisión cuando se adjunten cables largos.

Para saber más

Eche un vistazo a la hoja de datos del TMP36.

Para medir la temperatura usando un termistor vea el Capítulo 13.4.

Para medir la temperatura usando un sensor de temperatura digital (DS18B20) vea el Capítulo 13.12.

Para medir la temperatura usando un Sense HAT vea el Capítulo 13.11.

13.10 Medir la temperatura de la CPU de Raspberry Pi

Problema

Desea saber lo caliente que se está poniendo la CPU de su Raspberry Pi.

Solución

Use la biblioteca *os* para acceder al sensor de temperatura integrado en el chip Broadcom.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la web del libro, donde pone *cpu_temp.py*.

```
import os, time

while True:
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp = dev.read()
    print(cpu_temp)
    time.sleep(1)
```

Al ejecutar este programa, informará de la temperatura. Tenga en cuenta que el mensaje es en realidad una cadena con `temp=` antes de la temperatura y `'C` después.

```
$ python cpu_temp.py
temp=33.6'C
temp=33.6'C
```

Observaciones

Si desea que la temperatura sea un número en lugar de una cadena, debe cortar el texto extra y convertir el número en flotante. Tiene un ejemplo en *cpu_temp_float.py*.

```
import os, time

while True:
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp_s = dev.read()[5:-3] # principio y final de la cadena
    cpu_temp = float(cpu_temp_s)
    print(cpu_temp)
    time.sleep(1)
```

Para saber más

Para saber cómo cortar cadenas revise el Capítulo 5.16.

Para medir la temperatura con un termistor consulte el Capítulo 13.4.

Para medir la temperatura usando un TMP36 vea el Capítulo 13.9.

Para medir la temperatura usando un sensor de temperatura digital (DS18B20) vea el Capítulo 13.12.

Para medir la temperatura usando un Sense HAT vea el Capítulo 13.11.

13.11 Medir la temperatura, la humedad y la presión con Sense HAT

Problema

Desea medir la temperatura, la humedad y la presión, pero no quiere tener que conectar tres sensores separados.

Solución

Utilice un Sense HAT de Raspberry Pi (Figura 13.16). De esta manera se obtienen todos esos sensores, además de otros extras como una pantalla.

Figura 13.16. *Sense HAT.*

Comience siguiendo el Capítulo 9.17 para instalar las bibliotecas necesarias para el Sense HAT.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la web del libro, donde pone *sense_hat_thp.py*.

```
from sense_hat import SenseHat
import time

hat = SenseHat()

while True:
    t = hat.get_temperature()
    h = hat.get_humidity()
    p = hat.get_pressure()
    print('Temp C:{:.2f} Hum:{:.0f} Pres:{:.0f}'.format(t, h, p))
    time.sleep(1)
```

Al ejecutar el programa el terminal mostrará algo como esto:

```
$ sudo python sense_hat_thp.py
Temp C:27.71 Hum:56 Pres:1005
Temp C:27.60 Hum:55 Pres:1005
```

La temperatura está en grados C, la humedad es el porcentaje de humedad relativa y la presión atmosférica está en milibares.

Observaciones

Encontrará que las lecturas de la temperatura del Sense HAT son más altas de lo común. Esto se debe a que el sensor de temperatura está incorporado en el sensor de humedad y se encuentra en el Sense HAT. El Sense HAT genera muy poco calor (a menos que use la pantalla), pero la Raspberry Pi bajo el Sense HAT se calentará y aumentará la temperatura de HAT. La mejor manera de evitar este problema es usar un cable de cinta de 40 vías para situar el Sense HAT lejos de la Raspberry Pi. También puede ajustar las lecturas usando la lectura de la temperatura de Raspberry Pi; puede seguir la discusión en *http://bit.ly/1OfEEWf*. Personalmente, pienso que estos intentos de compensación son probablemente muy específicos para los usuarios y es poco probable que produzcan resultados fiables.

Además de leer la temperatura desde el sensor de humedad, el sensor de presión tiene un sensor de temperatura incorporado. Puede leerlo así:

```
t = hat.get_temperature_from_pressure()
```

No está claro si esta lectura es más precisa que la de usar el sensor de humedad, pero en mi caso, informó 1 grado C más bajo que el sensor de humedad.

Para saber más

Para comenzar con Sense HAT vea el Capítulo 9.17.

Puede encontrar la referencia de programación para el Sense HAT aquí: *https://pythonhosted.org/ sense-hat/api/*.

El Sense HAT también tiene un acelerómetro, un giroscopio (Capítulo 13.14) y un magnetómetro (Capítulo 13.15) para proyectos de navegación. También tiene una pantalla matriz de ledes de 8 × 8 a todo color (Capítulo 14.4).

13.12 Medir la temperatura con un sensor digital

Problema

Desea medir la temperatura usando un sensor digital preciso.

Solución

Utilice el sensor de temperatura digital DS18B20. Este dispositivo es más preciso que el TMP36 utilizado en el Capítulo 13.9 y utiliza una interfaz digital, por lo que no necesita un chip ADC.

Aunque la interfaz de este chip se llama *one-wire*, esto solo se refiere al pin de datos. Necesita al menos otro cable para conectarlo a un dispositivo de un solo cable.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Sensor de temperatura DS18B20 (vea "Circuitos integrados", página 475)
- Resistor 4,7 kΩ (vea "Resistores y condensadores" en la página 474)

Coloque los componentes en la placa de pruebas como se muestra en la Figura 13.17. Asegúrese de que coloca el DS18B20 de manera correcta.

La última versión de Raspbian tiene soporte para la interfaz de un solo cable (*one-wire*) utilizada por el DS18B20, pero tiene que habilitarlo.

Para habilitar el soporte *one-wire* edite el archivo */boot/config.txt* usando el comando `sudo nano /boot/config.txt` y añada la línea que se muestra a continuación al final del archivo. Después reinicie su Raspberry Pi para que se apliquen los cambios.

```
$ dtoverlay=w1-gpio
```

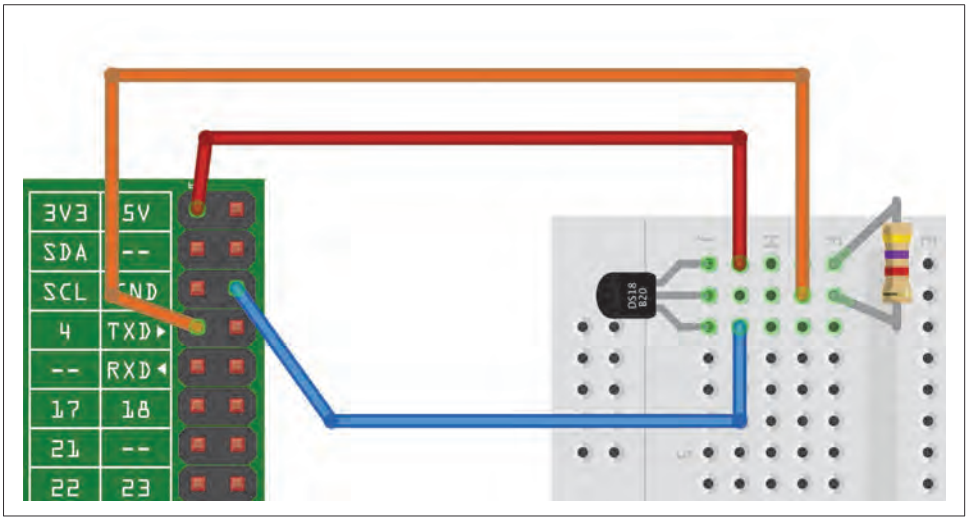


Figura 13.17. *Conexión de un DS18B20 a Raspberry Pi.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la web del libro, donde pone *temp_DS18B20.py*.

```
import glob, time

base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        temp_f = temp_c * 9.0 / 5.0 + 32.0
        return temp_c, temp_f

while True:
        print("temp C=%f\ttemp F=%f" % read_temp())
        time.sleep(1)
```

Cuando el programa se ejecute, informará de la temperatura una vez por segundo en grados Celsius y Fahrenheit:

```
$ python temp_DS18B20.py
temp C=25.187000    temp F=77.336600
temp C=25.125000    temp F=77.225000
temp C=25.062000    temp F=77.111600
temp C=26.312000    temp F=79.361600
temp C=27.875000    temp F=82.175000
temp C=28.875000    temp F=83.975000
```

Observaciones

A primera vista, el programa se ve un poco raro. La interfaz del DS18B20 usa una interfaz de tipo archivo. La interfaz de archivo para el dispositivo siempre estará en la carpeta */sys/bus/w1/devices/* y el nombre de la ruta del archivo comenzará por *28*, pero el resto de la ruta del archivo será diferente para cada sensor.

El código supone que solo habrá un sensor y encontrará la primera carpeta que comience por *28*. Para utilizar múltiples sensores utilice diferentes valores de índice dentro de corchetes.

Dentro de esa carpeta habrá un archivo llamado *w1_slave*, que se abrirá y se leerá para encontrar la temperatura.

El sensor en realidad devuelve cadenas de texto así:

```
81 01 4b 46 7f ff 0f 10 71 : crc=71 YES
81 01 4b 46 7f ff 0f 10 71 t=24062
```

El resto del código extrae la parte de la temperatura de este mensaje. Esto aparece después de `t=` y es la temperatura en milésimas de grado Celsius.

La función `read_temp` calcula la temperatura en grados Celsius y Fahrenheit y devuelve ambos valores.

Además de la versión de chip básico del DS18B20, puede adquirir una versión encapsulada en una sonda robusta e impermeable.

Para saber más

Para obtener información sobre las lecturas de registro revise el Capítulo 13.21. Este capítulo está basado en un tutorial de Adafruit. Eche un vistazo a la hoja de datos del DS18B20.

Para medir la temperatura usando un termistor consulte el Capítulo 13.4.

Para medir la temperatura usando un TMP36 vaya al Capítulo 13.9.

Para medir la temperatura usando un Sense HAT revise el Capítulo 13.11.

13.13 Medir la aceleración con un módulo MCP3008

Problema

Desea conectar un acelerómetro de tripe eje a Raspberry Pi.

Solución

Utilice un acelerómetro analógico con un chip ADC MCP3008 para medir las salidas analógicas X, Y, y Z.

Para ello necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Chip ADC IC MCP3008 de ocho canales (vea "Circuitos integrados", página 475)
- Acelerómetro de triple eje ADXL335 (vea "Módulos" en la página 476)

La Figura 13.18 muestra la disposición para esto usando una placa de pruebas. Utiliza tres canales del ADC para medir las fuerzas de aceleración X, Y y Z.

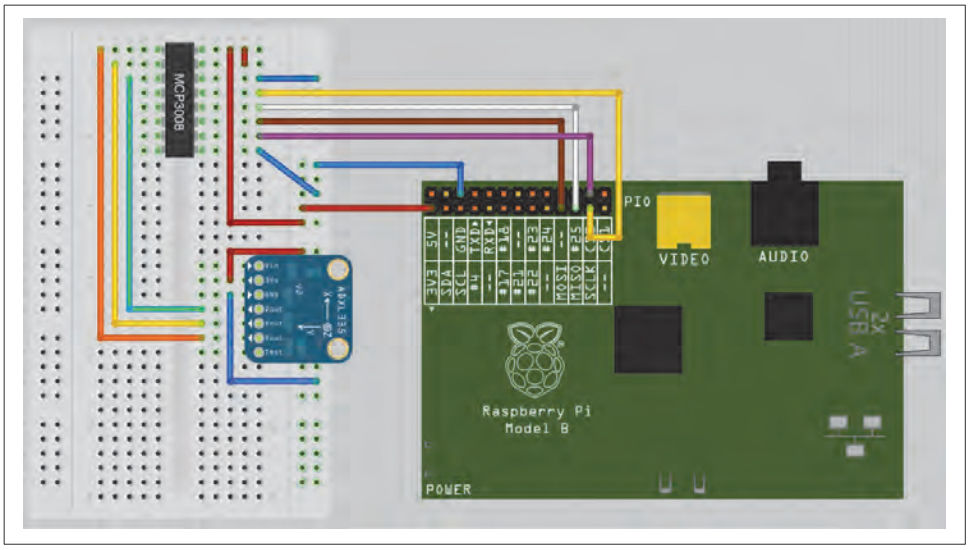


Figura 13.18. *Uso de un acelerómetro de triple eje.*

Tendrá que configurar el SPI en su Raspberry Pi, así que, si aún no lo ha hecho, siga el Capítulo 9.6.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la página web del libro, donde pone *adc_accelerometer.py*.

```
import spidev, time

spi = spidev.SpiDev()
spi.open(0,0)

def analog_read(channel):
    r = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_out = ((r[1]&3) << 8) + r[2]
    return adc_out

while True:
    x = analog_read(0)
    y = analog_read(1)
    z = analog_read(2)
    print("X=%d\tY=%d\tZ=%d" % (x, y, z))
    time.sleep(1)
```

El programa simplemente lee las tres fuerzas y las imprime:

```
$ sudo python adc_accelerometer.py
X=508   Y=503   Z=626
X=508   Y=504   Z=624
X=506   Y=505   Z=627
X=423   Y=517   Z=579
X=411   Y=513   Z=548
X=532   Y=510   Z=623
X=609   Y=518   Z=495
X=607   Y=521   Z=496
X=610   Y=513   Z=499
```

Las primeras tres lecturas tuvieron lugar a nivel del acelerómetro. En las tres siguientes la placa de pruebas entera se inclinó hacia un lado. Puede ver que la lectura X disminuyó. Haciendo girar la placa de pruebas por el otro lado hará que la lectura de X aumente.

Observaciones

El uso más común de un acelerómetro es detectar la inclinación. Esto funciona porque la fuerza del eje Z está dominada por la atracción de la gravedad (Figura 13.19).

Cuando el acelerómetro está inclinado en una dirección parte de esa fuerza vertical de gravedad se activa en otro eje del acelerómetro.

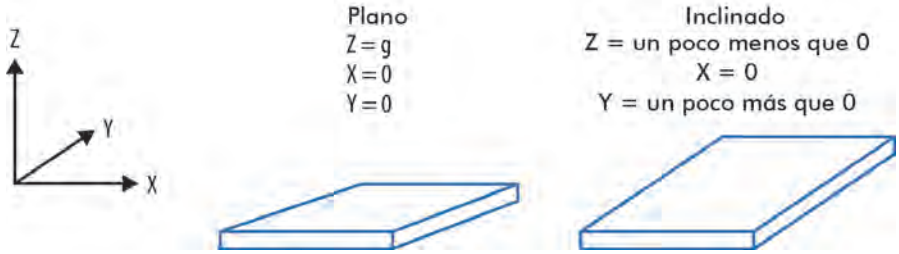


Figura 13.19. *Detección de la inclinación con un acelerómetro.*

Podemos utilizar este principio para detectar cuando la inclinación ha pasado un determinado umbral. El siguiente programa (*tilt.py*) ilustra este punto:

```
import spidev, time

spi = spidev.SpiDev()
spi.open(0,0)

def analog_read(channel):
    r = spi.xfer2([1, (8 + channel) << 4, 0])
    adc_out = ((r[1]&3) << 8) + r[2]
    return adc_out

while True:
    x = analog_read(0)
    y = analog_read(1)
    z = analog_read(2)
    if x <450:
        print("Left")
    elif x > 550:
        print("Right")
    elif y < 450:
        print("Back")
    elif y > 550:
        print("Forward")
    time.sleep(0.2)
```

Cuando ejecute el programa comenzará a ver mensajes de dirección. Podría utilizar esto para controlar un robot o una cabeza monitorizada con una webcam conectada:

```
$ sudo python tilt.py
Left
Left
Right
Forward
Forward
Back
Back
```

Para saber más

Consulte la hoja de datos del chip del acelerómetro utilizado en el módulo.

Hay muchos otros módulos analógicos del acelerómetro disponibles. Puede ver que dan lecturas diferentes. Asegúrese de que las salidas analógicas no superen los 3,3 V.

El Sense HAT incluye un acelerómetro (Capítulo 13.14).

13.14 Usar la Unidad de Medición Inercial (IMU) del Sense HAT

Problema

Desea obtener información de la orientación más precisa desde su Raspberry Pi que la que el acelerómetro del Capítulo 13.13 proporciona.

Solución

Utilice la Unidad de Medición Inercial (IMU) del Sense HAT. Esta unidad incluye un acelerómetro de 3 ejes como el del Capítulo 13.13, pero también tiene un giroscopio y un magnetómetro de 3 ejes. Las lecturas de estos diferentes sensores se combinan para permitirle obtener una orientación más precisa para el Sense HAT expresado como alabeo, cabeceo y guiñada (Figura 13.20).

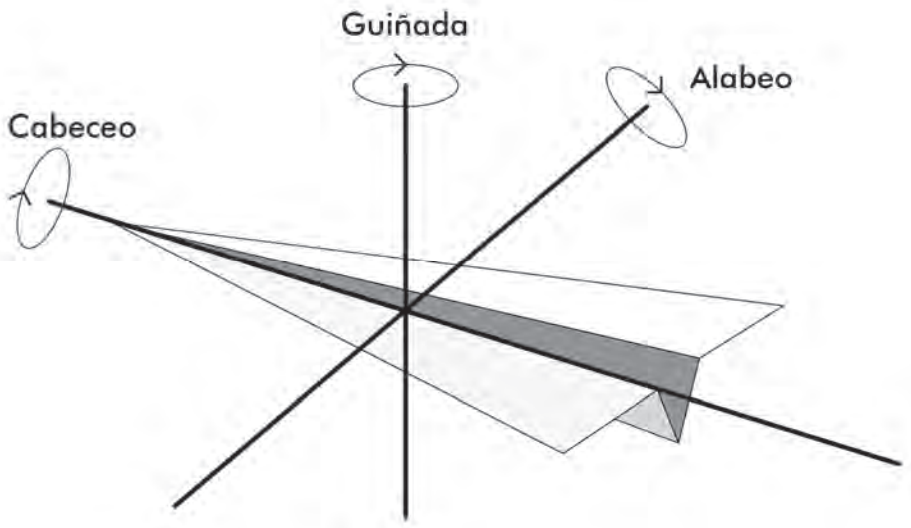


Figura 13.20. *Alabeo, cabeceo y guiñada.*

Alabeo, cabeceo y guiñada son tres términos que vienen de la aviación. Tienen relación con los ejes de vuelo de un avión. El cabeceo es el ángulo a la horizontal. El alabeo es el grado de rotación alrededor del eje de vuelo del avión (imagine un ala que sube y otra que baja) y la guiñada es la rotación del eje vertical.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *sense_hat_orientation.py*.

```
from sense_hat import SenseHat

sense = SenseHat()

sense.set_imu_config(True, True, True)

while True:
    o = sense.get_orientation()
    print("p: {:.0f}, r: {:.0f}, y: {:.0f}".format(o['pitch'], o['roll'],
o['yaw']))
```

La función `set_imu_config` especifica qué brújula, giroscopio y acelerómetro (en ese orden) se deben utilizar para medir la orientación. Poner los tres en `True` significa que todos se usan para realizar la medición.

Al ejecutar el programa verá una salida similar a esta:

```
$ sudo python sense_hat_orientation.py
p: 1, r: 317, y: 168
p: 1, r: 318, y: 169
```

Trate de inclinar el Sense HAT y Raspberry Pi hacia adelante, hacia los puertos USB. Debería ver que aumenta el valor del cabeceo.

Observaciones

Un acelerómetro mide las fuerzas sobre una masa estacionaria y, por lo tanto, puede medir el grado de inclinación calculando cuánta de la fuerza suministrada por la gravedad (eje z) influye en la fuerza medida en los ejes x e y.

Un giroscopio es diferente. Mide la fuerza sobre las masas móviles (vibrando hacia adelante y hacia atrás) cuando esas masas giran en relación con la trayectoria del movimiento, usando un efecto llamado efecto Coriolis.

Para saber más

Para obtener más información sobre la IMU de Sense HAT vea *https://www.raspberrypi.org/learning/astro-pi-guide/sensors/movement.md*.

Para medir la temperatura, la humedad y la presión atmosférica vea el Capítulo 13.11.

La IMU del Sense HAT también se puede usar para hacer una brújula y detectar la presencia de un imán (Capítulo 13.17).

Para obtener más información sobre los giroscopios y el efecto Coriolis vea *https://en.wikipedia.org/wiki/Coriolis_effect*.

13.15 Encontrar el norte magnético con el Sense HAT

Problema

Desea usar un Sense HAT para detectar el norte magnético.

Solución

Use la biblioteca de Python para el magnetómetro de 3 ejes incorporado en el Sense HAT. Primero, siga el Capítulo 9.17 para instalar la biblioteca de Sense HAT.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la web del libro, donde pone *sense_hat_compass.py*.

```
from sense_hat import SenseHat

sense = SenseHat()

while True:
    bearing = sense.get_compass()
    print('Bearing: {:.0f} to North'.format(bearing))
```

Al ejecutar este programa verá una serie de lecturas de los cojinetes:

```
$ sudo python sense_hat_compass.py
Bearing: 138 to North
Bearing: 138 to North
```

Observaciones

La brújula será sensible a cualquier otra fuente cercana de campo magnético, por lo que puede resultar difícil conseguir valores precisos.

Para saber más

Puede encontrar documentación para Sense HAT en *https://github.com/RPi-Distro/python-sense-hat* and *http://pythonhosted.org/sense-hat/api/*.

Para usar el Sense HAT para detectar un imán vea el Capítulo 13.17.

13.16 Detectar un imán con un interruptor de láminas

Problema

Desea detectar la presencia de un imán.

Solución

Utilice un interruptor de láminas (Figura 13.21). Funciona igual que un interruptor normal, excepto que solo se activa cuando hay cerca un imán. La Figura 13.22 muestra cómo funciona un interruptor de láminas.

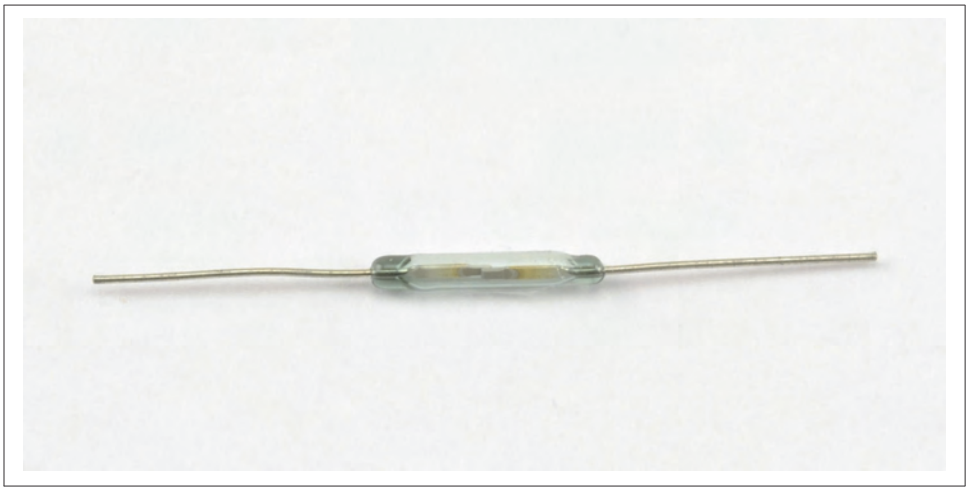


Figura 13.21. *Un interruptor de láminas.*

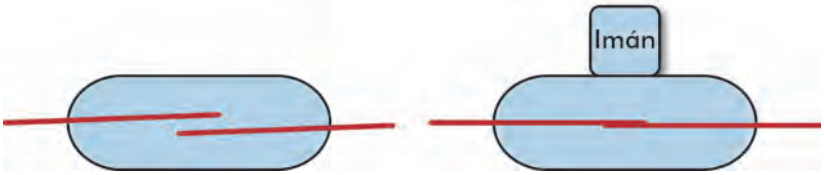


Figura 13.22. *Cómo funciona un interruptor de láminas.*

Los dos contactos de lámina están encerrados en un tubo de vidrio. Cuando se coloca un imán cerca del interruptor de láminas, las láminas se atraen y hacen contacto.

Puede utilizar cualquier punto de interruptores habituales del Capítulo 12 con un interruptor de láminas, comenzando por el Capítulo 12.2.

Observaciones

Los interruptores de láminas son una manera de baja tecnología de detectar un imán. Existen desde los años treinta del siglo pasado y son extremadamente fiables. También se utilizan comúnmente en los sistemas de seguridad, en los que se coloca un interruptor de láminas recubierto de plástico en el marco de una puerta con un imán fijo, también recubierto de plástico en la misma puerta. Cuando la puerta se abre, los contactos del interruptor de láminas se abren, activando la alarma.

Para saber más

Para detectar un imán usando el magnetómetro de Sense HAT vaya al Capítulo 13.17.

13.17 Detectar un imán con un Sense HAT

Problema

Desea detectar la presencia de un imán y tiene un Sense Hat con un magnetómetro integrado.

Solución

Utilice la biblioteca Python de Sense HAT para interactuar con su magnetómetro.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la web del libro, donde pone *sense_hat_magnet.py*.

```
from sense_hat import SenseHat
import time

hat = SenseHat()
fill = (255, 0, 0)

while True:
    reading = int(hat.get_compass_raw()['z'])
    if reading > 200:
        hat.clear(fill)
        time.sleep(0.2)
    else:
        hat.clear()
```

Cuando el imán se acerque al Sense HAT, los ledes se volverán rojos durante 1/5 de segundo.

Observaciones

No importa qué eje de la brújula se utiliza, todos se verán perturbados por la presencia de un imán fijo.

Para saber más

Para detectar un imán usando un interruptor de láminas vaya al Capítulo 13.16.

Para otras formas de usar la pantalla de Sense HAT revise el Capítulo 14.4.

13.18 Medir la distancia

Problema

Quiere medir la distancia usando un telémetro ultrasónico.

Solución

Utilice un telémetro SR-04 de bajo coste. Estos dispositivos necesitan dos pines GPIO: uno para activar el pulso de ultrasonido y el otro para controlar cuánto tiempo tarda el eco en volver.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Telémetro SR-04 (eBay)
- Resistor 470 Ω (vea "Resistores y condensadores" en la página 474)
- Resistor 270 Ω (vea "Resistores y condensadores" en la página 474)

Coloque los componentes en la placa de pruebas como se muestra en la Figura 13.23. Los resistores son necesarios para reducir la salida *echo* del telémetro de 5 V a 3,3 V (vea el Capítulo 9.13).

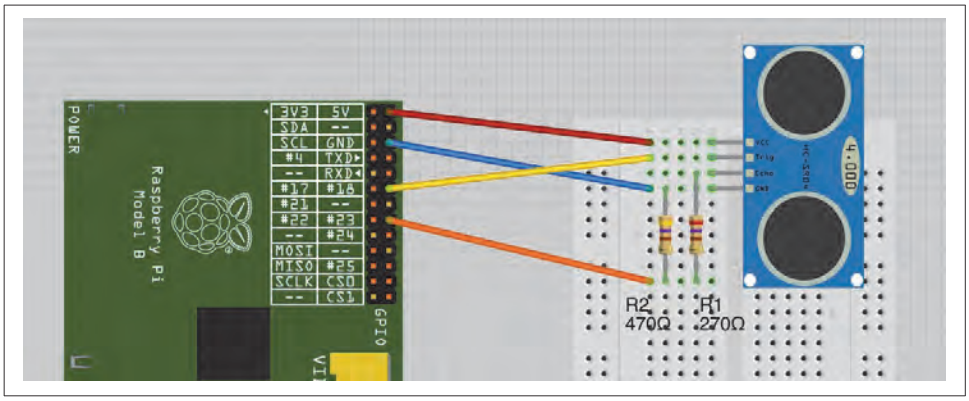


Figura 13.23. *Conexión de un telémetro SR-04 a Raspberry Pi.*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *ranger.py*.

```
import RPi.GPIO as GPIO
import time

trigger_pin = 18
echo_pin = 23

GPIO.setmode(GPIO.BCM)
GPIO.setup(trigger_pin, GPIO.OUT)
GPIO.setup(echo_pin, GPIO.IN)

def send_trigger_pulse():
    GPIO.output(trigger_pin, True)
    time.sleep(0.0001)
    GPIO.output(trigger_pin, False)

def wait_for_echo(value,
    timeout): count = timeout
    while GPIO.input(echo_pin) != value and count > 0:
        count = count - 1

def get_distance():
    send_trigger_pulse()
    wait_for_echo(True, 10000)
    start = time.time()
    wait_for_echo(False, 10000)
    finish = time.time()
    pulse_len = finish - start
    distance_cm = pulse_len / 0.000058
    distance_in = distance_cm / 2.5
    return (distance_cm, distance_in)
```

```
while True:
    print("cm=%f\tinches=%f" % get_distance())
    time.sleep(1)
```

El funcionamiento del programa se describe en el apartado «Observaciones». Cuando se ejecute el programa informará sobre la distancia en centímetros y pulgadas una vez por segundo. Use su mano u otro obstáculo para cambiar la lectura.

```
sudo python ranger.py
cm=154.741879   inches=61.896752
cm=155.670889   inches=62.268356
cm=154.865199   inches=61.946080
cm=12.948595    inches=5.179438
cm=14.087249    inches=5.634900
cm=13.741954    inches=5.496781
cm=20.775302    inches=8.310121
cm=20.224473    inches=8.089789
```

Observaciones

Hay una serie de medidores de ultrasonidos disponibles, el que se utiliza aquí es fácil de usar y barato. Funciona enviando un pulso de ultrasonido y luego midiendo la cantidad de tiempo que se tarda en recibir el eco. Uno de los transductores ultrasónicos de la parte delantera del dispositivo es el transmisor, y el otro es el receptor.

Este proceso se controla desde Raspberry Pi. La diferencia entre este tipo de dispositivos y los modelos más caros es que las versiones más caras incluyen su propio microcontrolador, que realiza todo el cronometraje necesario y proporciona una interfaz I2C o serie para devolver una lectura final.

Cuando se utiliza uno de estos sensores con Raspberry Pi, la entrada *trig* (activador) del telémetro se conecta a una salida GPIO y la salida *echo* (eco) del telémetro se conecta a una entrada GPIO en Raspberry Pi después de tener el voltaje reducido de 5 V a 3,3 V.

La Figura 13.24 muestra el trazo del osciloscopio del sensor en acción. El trazo superior (rojo) está conectado a *trig* y el trazo inferior (amarillo) está conectado a *echo*. Puede ver que primero el pin *trig* sube con un pulso corto. Después, aparece un breve retraso antes de que el pin *echo* suba. Este permanece alto durante un período proporcional a la distancia desde el sensor.

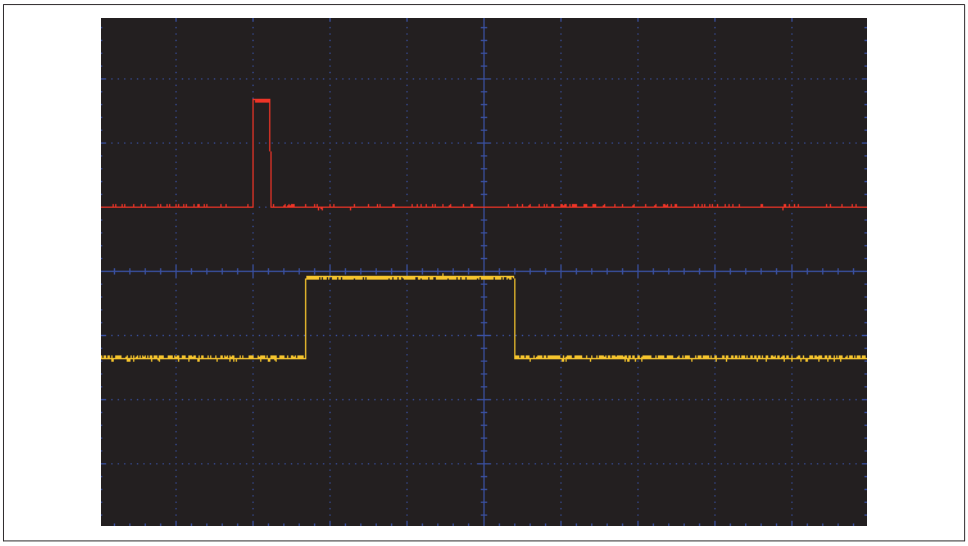


Figura 13.24. *Trazo del osciloscopio: activador y eco.*

El código para este sensor genera primero un pulso de activación (usando la función `send_trigger_pulse`). Debe esperar hasta que el pin *echo* se eleve y después verá el tiempo que el pin *echo* permanece alto.

Podremos entonces calcular la distancia usando el tiempo tomado para el pulso del eco y la velocidad del sonido.

La función `wait_for_echo` espera hasta que el pin *echo* suba o baje, dependiendo de su primer argumento. El segundo argumento se utiliza para proporcionar un tiempo de espera, para que, si por alguna razón el pin *echo* no cambia al estado que se esperaba, el bucle no se bloquee indefinidamente.

Este método de medir la distancia no es muy preciso porque la temperatura, la precisión y la humedad relativa alteran la velocidad del sonido y, por lo tanto, las lecturas de la distancia.

Para saber más

Eche un vistazo a la hoja de datos del telémetro ultrasónico.

13.19 Sensor táctil capacitivo

Problema

Desea proporcionar una interfaz táctil a su Raspberry Pi.

Solución

Use un sensor táctil capacitivo HAT de Adafruit (Figura 13.25).

Los sensores táctiles son muy divertidos y son ideales para el uso educativo. Puede conectar cualquier cosa que conduzca un poco la electricidad, incluyendo la fruta. Un proyecto popular es la construcción de un teclado de frutas utilizando una variedad de frutas y verduras unidas a los terminales sensoriales de la placa mediante pinzas de contacto. Después, al tocar los diferentes elementos de la fruta se producirán diferentes sonidos.

El sensor táctil capacitivo HAT de Adafruit utiliza la interfaz I2C de Raspberry Pi. También necesita herramientas SPI instaladas, así que, si aún no lo ha hecho, siga el Capítulo 9.4.

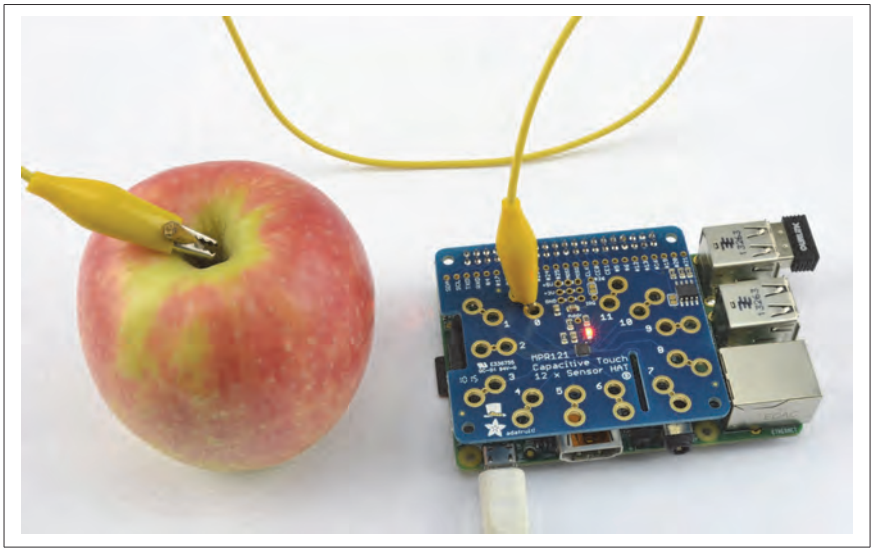


Figura 13.25. *Táctil capacitivo HAT de Adafruit HAT unido a una manzana.*

Para instalar la biblioteca de Python para el HAT ejecute los siguientes comandos:

```
$ cd /home/pi
$ git clone https://github.com/adafruit/Adafruit_Python_MPR121.git
$ cd Adafruit_Python_MPR121/
$ sudo python setup.py install
```

Para probar el HAT táctil ejecute el ejemplo *simpletest.py* suministrado con la biblioteca utilizando los siguientes comandos:

```
$ cd examples
$ sudo python simpletest.py
Adafruit MPR121 Capacitive Touch Sensor Test
Press Ctrl-C to quit.
```

```
0 touched!
0 released!
```

Puede tocar los paneles de conexión o conectar los paneles a una pieza de fruta usando una pinza de contacto como se muestra en la Figura 13.25.

Observaciones

Verá otros ejemplos de programas en la carpeta *examples*, incluyendo el popular teclado musical basado en frutas.

El Touch HAT de Adafruit tiene 12 contactos táctiles. Si solo necesita unos cuantos contactos táctiles, puede utilizar el Pimoroni HAT Explorer Pro, que tiene cuatro contactos compatibles con las pinzas de contacto (Figura 13.26).

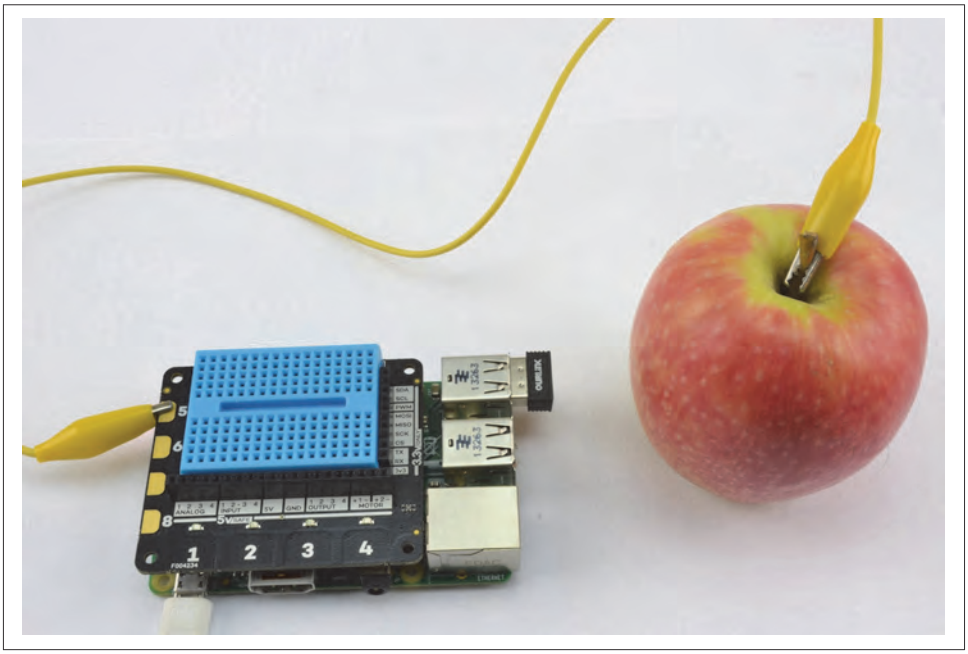


Figura 13.26. *HAT Explorer Pro con fruta.*

Para usar los contactos táctiles del HAT Explorer Pro, primero siga el Capítulo 9.18 para instalar la biblioteca para el HAT.

Además de los cuatro terminales al lado, diseñados para las pinzas de contacto, hay también cuatro interruptores táctiles etiquetados de 1 a 4 que también utilizan la interfaz táctil.

Para probar el funcionamiento en el HAT Explorer Pro, abra una consola de Python y ejecute los siguientes comandos para detectar los toques:

```
>>> import explorerhat
Explorer HAT Pro detected...
>>> explorerhat.touch.five.is_pressed()
True
>>> explorerhat.touch.five.is_pressed()
False
>>>
```

Si ejecuta la función `is_pressed()` mientras sostiene la manzana, el resultado será `True`; de lo contrario, será `False`.

Para saber más

Para obtener la documentación del Touch HAT de Adafruit vea:

https://www.adafruit.com/products/2340

Para el HAT Explorer Pro vea *https://github.com/pimoroni/explorerhat*.

13.20 Visualizar los valores de un sensor

Problema

Tiene un sensor conectado a su Raspberry Pi y quiere una gran pantalla digital para que aparezca la lectura.

Solución

Utilice la biblioteca Tkinter para abrir una ventana y que se escriba la lectura con letra grande (Figura 13.27).

Figura 13.27. *Visualización de una lectura del sensor con Tkinter.*

Este ejemplo utiliza datos del telémetro ultrasónico del Capítulo 13.18. Por lo tanto, sígalo primero si desea probar este ejemplo.

Para probar el telémetro, abra un editor (nano o idle) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la web del libro, donde pone *gui_sensor_reading.py*. El código para medir las distancias con el sensor ultrasónico es el mismo que el del Capítulo 13.18, por lo que solo se repite el trozo del código relacionado con la visualización del valor:

```
class App:

    def init (self, master):
        self.master = master
        frame = Frame(master)
        frame.pack()
        label = Label(frame, text='Distance (inches)', font=("Helvetica", 32))
        label.grid(row=0)
        self.reading_label = Label(frame, text='12.34', font=("Helvetica", 110))
        self.reading_label.grid(row=1)
        self.update_reading()

    def update_reading(self):
        cm, inch = get_distance()
        reading_str = "{:.2f}".format(inch)
        self.reading_label.configure(text=reading_str)
        self.master.after(500, self.update_reading)


root = Tk()
root.wm_title('Range Finder')
app = App(root)
root.geometry("400x300+0+0")
root.mainloop()
```

Observaciones

Aunque este capítulo utiliza un sensor de distancia, funciona igual de bien con los otros capítulos de sensores. Solo tiene que cambiar las etiquetas y el método para obtener una lectura del sensor.

Para saber más

Para obtener información sobre el formato de números decimales vea el Capítulo 7.2.

Para ver un ejemplo de visualización de datos de sensores en un navegador web en lugar de en una ventana de aplicación vea el Capítulo 15.3.

13.21 Registrar en una unidad flash USB

Problema

Desea registrar los datos medidos con un sensor en una unidad flash USB.

Solución

Escriba un programa de Python que escriba los datos en un archivo de una unidad flash USB. Al escribir el archivo en valores separados por comas (CSV) puede importarlo directamente a una hoja de cálculo, incluido Gnumeric en Raspberry Pi (Capítulo 4.3).

El programa de ejemplo registrará las lecturas de temperatura grabadas desde un DS18B20. Así que, si usted desea probarlo, primero siga el Capítulo 13.12.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código de la página web del libro, donde pone *temp_log.py*.

```
import os, glob, time, datetime

log_period = 600 # segundos

logging_folder = glob.glob('/media/*')[0]
dt = datetime.datetime.now()
file_name = "temp_log_{:%Y_%m_%d}.csv".format(dt)
logging_file = logging_folder + '/' + file_name

os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
```

```
            temp_c = float(temp_string) / 1000.0
            temp_f = temp_c * 9.0 / 5.0 + 32.0
            return temp_c, temp_f

    def log_temp():
        temp_c, temp_f = read_temp()
        dt = datetime.datetime.now()
        f = open(logging_file, 'a')
        f.write('\n"{:%H:%M:%S}",'.format(dt))
        f.write(str(temp_c))
        f.close()

    print("Logging to: " + logging_file)
    while True:
        log_temp()
        time.sleep(log_period)
```

El programa está configurado para registrar la temperatura cada 10 minutos (600 segundos). Puede cambiarlo cambiando el valor de `log_period`.

Observaciones

Cuando conecte una unidad flash USB a Raspberry Pi, automáticamente la instalará en */media*. Si hay más de una unidad extraíble conectada a su Raspberry Pi, el programa utilizará la primera carpeta que encuentre dentro de */media*. El nombre del archivo de registro se construye a partir de la fecha actual.

Si abre el archivo en una hoja de cálculo como Open Office, podrá editarlo directamente. Su hoja de cálculo puede pedirle que especifique el separador de los datos, que será una coma.

La Figura 13.28 muestra el conjunto de datos capturados de esta manera. El archivo resultante se ha abierto con la hoja de cálculo Gnumeric que se ejecuta en Raspberry Pi.

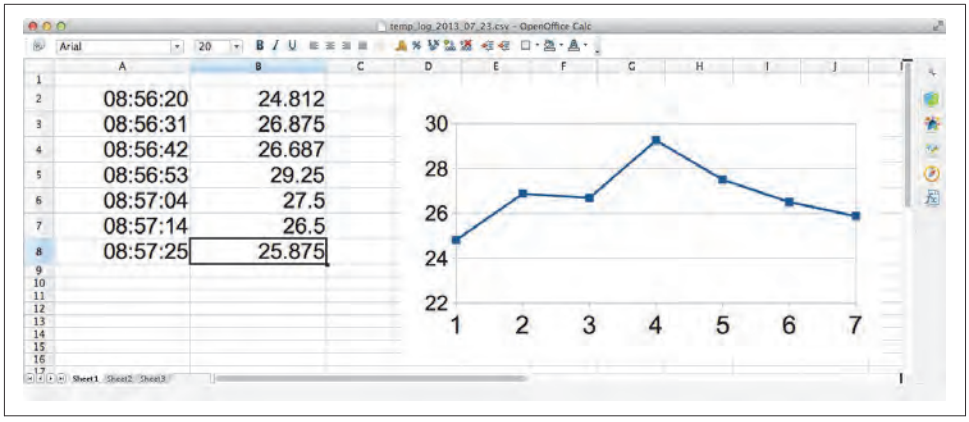


Figura 13.28. *Datos de gráficos.*

Para saber más

Este programa podría ser fácilmente adaptado para su uso con cualquiera de los otros sensores utilizados en este capítulo.

Para obtener un ejemplo de registro de datos de sensores en un servicio web consulte el Capítulo 15.7.

CAPÍTULO 14

Visualización

14.1 Introducción

Aunque Raspberry Pi puede usar un monitor o un televisor como pantalla, a menudo es agradable utilizar una pantalla más pequeña y más especializada. En este capítulo explorará una gama de diversas pantallas que pueden conectarse a Raspberry Pi.

Algunas de las acciones necesitarán el uso de una placa de pruebas sin soldar y cables puente macho-hembra (revise el Capítulo 9.9).

14.2 Usar una pantalla led de cuatro dígitos

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea mostrar un número de cuatro dígitos en una pantalla led de siete segmentos.

Solución

Conecte un módulo led I2C, como el modelo que se muestra en la Figura 14.1, a Raspberry Pi usando cables puente hembra a hembra.

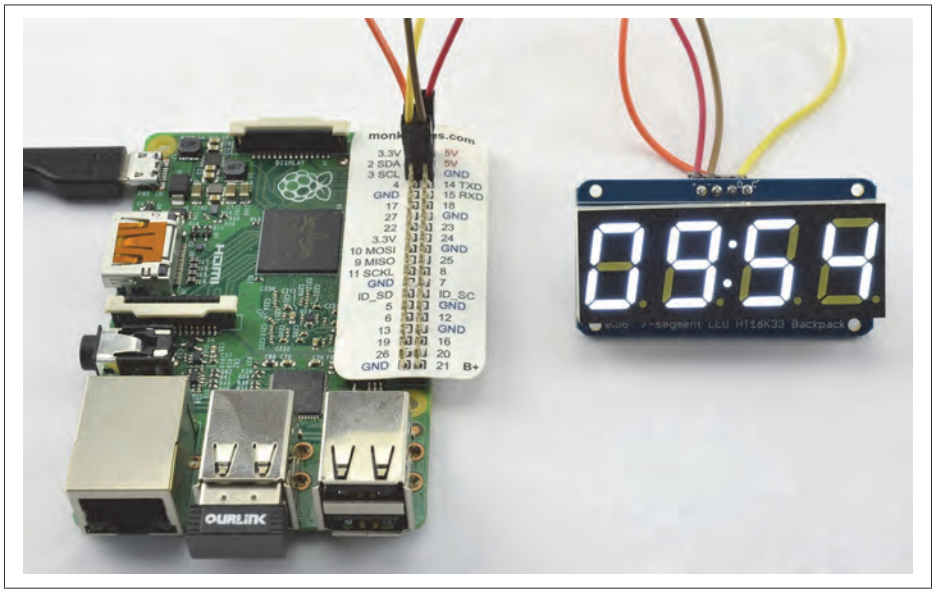


Figura 14.1. *Pantalla led de siete segmentos con Raspberry Pi.*

Para hacer esto necesitará:

- Cuatro cables puente hembra-hembra (vea "Equipos para prototipos", pág. 474)
- Led de 4 × 7 segmentos Adafruit con módulo I2C (vea "Módulos" en la pág. 476)

Las conexiones entre Raspberry Pi y el módulo son las siguientes:

- VCC (+) en la pantalla a 5 V en el conector GPIO de Raspberry Pi
- GND (–) en la pantalla a GND en el conector GPIO de Raspberry Pi
- SDA (D) en la pantalla a GPIO 2 (SDA) en el conector GPIO de Raspberry Pi
- SCL (C) en la pantalla a GPIO 3 (SCL) en el conector GPIO de Raspberry Pi

Tenga en cuenta que Adafruit también suministra una pantalla led de tamaño jumbo. Se puede conectar a Raspberry Pi con las conexiones anteriores, pero la pantalla más grande tiene dos pines de alimentación positivos: uno para la lógica (V_IO) y otro para la pantalla (5 V). Puede utilizar un cable puente extra de hembra a hembra para conectar el pin extra al segundo pin de 5 V en el conector GPIO. Esto se debe a que, al ser una pantalla grande requiere más corriente que una pantalla led. Afortunadamente, Raspberry Pi puede suministrar suficiente energía para ello.

Para que esto funcione tendrá que configurar su Raspberry Pi para I2C, así que primero siga el Capítulo 9.4.

La pantalla tiene una biblioteca de Python escrita por Adafruit. Lo que primero debe descargar es la estructura de carpetas.

```
$ git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Cambie el directorio en el código de Adafruit usando:

```
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_LEDBackpack
```

En esta carpeta encontrará un programa de prueba que mostrará la hora. Ejecútelo utilizando el comando:

```
$ sudo python ex_7segment_clock.py
```

Observaciones

Si abre el archivo de ejemplo *ex_7segment_clock.py* en nano, verá que los comandos clave son:

```
from Adafruit_7Segment import SevenSegment
```

Esto importa el código de la biblioteca a su programa. A continuación, debe crear una instancia de `SevenSegment` usando la siguiente línea de código. La dirección suministrada como argumento es la dirección I2C (vea el Capítulo 9.5).

Cada dispositivo esclavo I2C tiene un número de dirección. La placa led tiene tres pares de conectores soldados en la parte posterior que se pueden puentear con soldadura si desea cambiar la dirección. Esto es esencial si necesita gestionar más de un dispositivo I2C desde una única Raspberry Pi.

```
segment = SevenSegment(address=0x70)
```

Para establecer el contenido de un dígito particular utilice una línea como esta:

```
segment.writeDigit(0, int(hour / 10))
```

El primer argumento (0) es la posición del dígito. Tenga en cuenta que estas posiciones son 0, 1, 3 y 4. La posición 2 está reservada para los dos puntos en el centro de la pantalla.

El segundo argumento es el número que se va a mostrar.

Para saber más

Puede obtener más información sobre la biblioteca de Adafruit en *http://bit.ly/HQBE6W*.

14.3 Visualizar mensajes en una matriz de led I2C

Problema

Desea controlar los píxeles de una pantalla de matriz led multicolor.

Solución

Utilice un módulo led I2C, como el modelo que se muestra en la Figura 14.2, unido a Raspberry Pi mediante cables puente hembra a hembra.

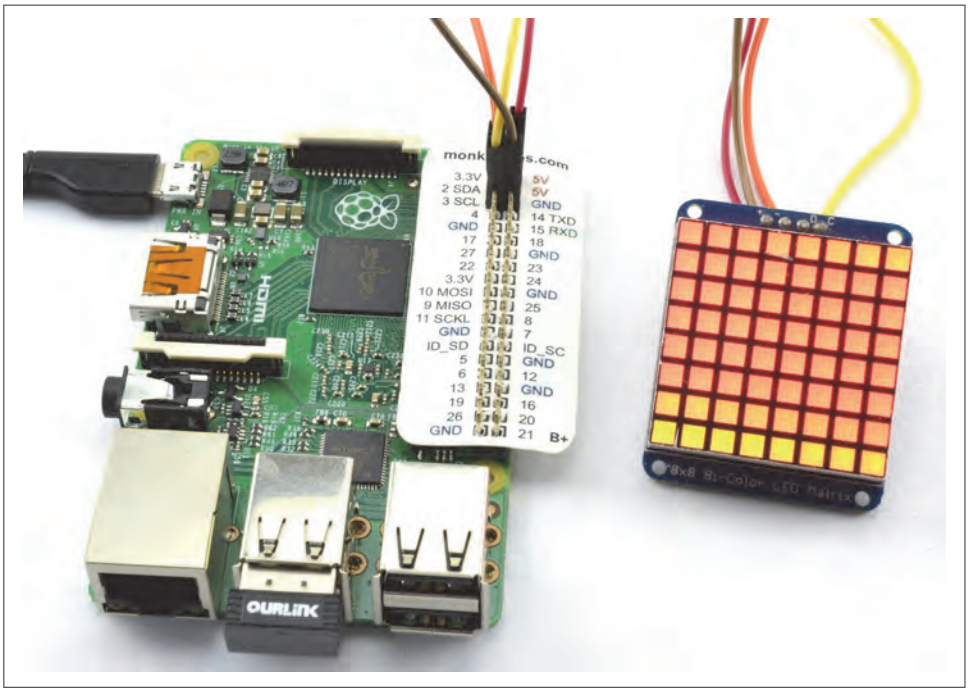


Figura 14.2. *Pantalla de matriz led con Raspberry Pi.*

Para hacer esto necesitará:

- Cuatro cables puente de hembra a hembra (vea "Equipos para prototipos", página 474)
- Pantalla matriz de led Adafruit bicolor de píxeles cuadrados con I2C (vea "Módulos" en la página 476)

Las conexiones entre Raspberry Pi y el módulo son las siguientes:

- VCC (+) en la pantalla a 5 V en el conector GPIO de Raspberry Pi
- GND (–) en la pantalla a GND en el conector GPIO de Raspberry Pi
- SDA (D) en la pantalla a GPIO 2 (SDA) en el conector GPIO de Raspberry Pi
- SCL (C) en la pantalla a GPIO 3 (SCL) en el conector GPIO de Raspberry Pi

Para que esto funcione también deberá configurar su Raspberry Pi para I2C, así que primero siga el Capítulo 9.4.

La pantalla tiene una biblioteca de Python escrita por Adafruit, para usarla primero deberá descargar la estructura de carpetas.

```
$ git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Cambie el directorio al código de Adafruit usando:

```
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_LEDBackpack
```

En esta carpeta encontrará un programa de prueba que muestra la hora como dígitos de desplazamiento. Ejecútelo utilizando el comando:

```
$ sudo python ex_8x8_color_pixels.py
```

Observaciones

El programa recorre todos los colores de cada píxel a su vez. El código aparece aquí, con algunos de los comentarios y una importación innecesaria eliminada:

```
import time
from Adafruit_8x8 import ColorEightByEight

grid = ColorEightByEight(address=0x70)

iter = 0

#Actualizar continuamente la pantalla 8x8 un píxel a la vez
while(True):
  iter += 1

 for x in range(0, 8):
    for y in range(0, 8):
      grid.setPixel(x, y, iter % 4 )
      time.sleep(0.02)
```

La dirección suministrada como argumento a la siguiente línea es la dirección I2C (vea el Capítulo 9.5):

```
grid = ColorEightByEight(address=0x70)
```

Cada dispositivo esclavo I2C tiene un número de dirección. La placa led tiene tres pares de conectores soldados en la parte posterior que se pueden puentear con soldadura si desea cambiar la dirección. Esto es esencial si necesita gestionar más de un dispositivo I2C con la misma dirección base desde una única Raspberry Pi.

La variable `iter` agrega 1 cada vez a través del bucle. El comando `grid.setPixel` toma las coordenadas *x* y *y* como sus dos primeros parámetros. El parámetro final es el color que se fijará al píxel. Será un número entre 0 y 3 (0 es apagado, 1 es verde, 2 es rojo y 3 es naranja).

La variable `iter` se utiliza para generar el número entre 0 y 3 usando el operador %, que es el resto del módulo (es decir, lo que queda cuando se divide `iter` entre 4).

Para saber más

Puede obtener más información sobre este producto en *http://www.adafruit.com/products/902*.

14.4 Usar la pantalla de matriz de led del Sense HAT

Problema

Desea mostrar mensajes y gráficos utilizando Sense HAT.

Solución

Siga el Capítulo 9.17 e instale el *software* que necesita Sense HAT, y luego utilice los comandos de la biblioteca para mostrar el texto.

El programa *sense_hat_clock.py* ilustra esto mostrando repetidamente la fecha y la hora en un mensaje que se desplaza. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro.

```
from sense_hat import SenseHat
from datetime import datetime
import time

hat = SenseHat()
time_color = (0, 255, 0)
date_color = (255, 0, 0)

while True:
    now = datetime.now()
    date_message = '{:%d %B %Y}'.format(now)
    time_message = '{:%H:%M:%S}'.format(now)

    hat.show_message(date_message, text_colour=date_color)
    hat.show_message(time_message, text_colour=time_color)
```

Observaciones

Se definen dos colores para que la fecha y la hora se muestren en diferentes colores. Estos colores se utilizan como un parámetro opcional para `show_message`.

Otros parámetros opcionales para `show_message` son:

- `scroll_speedh`, que es realmente el retardo entre cada paso de desplazamiento en lugar de la velocidad. Así que un valor más alto hará que el desplazamiento sea más lento.
- `back_colour` fija el color de fondo. Observe la ortografía británica de "back_colour" con una "u".

La pantalla se puede utilizar para mucho más que solo mostrar el texto de desplazamiento. Comenzando por lo más básico, puede establecer píxeles individuales utilizando `set_pixel`, establecer la orientación de la pantalla usando `set_rotation` y mostrar una imagen (pequeña) con `load_image`. El siguiente ejemplo, que se encuentra en *sense_hat_taster.py*, ilustra estas llamadas de función. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro.

La imagen debe tener solo 8×8 píxeles. La mayoría de los formatos gráficos más comunes, como *.jpg* y *.png*, se pueden utilizar y se gestionará automáticamente la profundidad de bits.

```
from sense_hat import SenseHat
import time

hat = SenseHat()

red = (255, 0, 0)

hat.load_image('small_image.png')
time.sleep(1)
hat.set_rotation(90)
time.sleep(1)
hat.set_rotation(180)
time.sleep(1)
hat.set_rotation(270)
time.sleep(1)

hat.clear()
hat.set_rotation(0)
for xy in range(0, 8):
    hat.set_pixel(xy, xy, red)
    hat.set_pixel(xy, 7-xy, red)
```

En la Figura 14.3 aparece el Sense HAT mostrando una imagen.

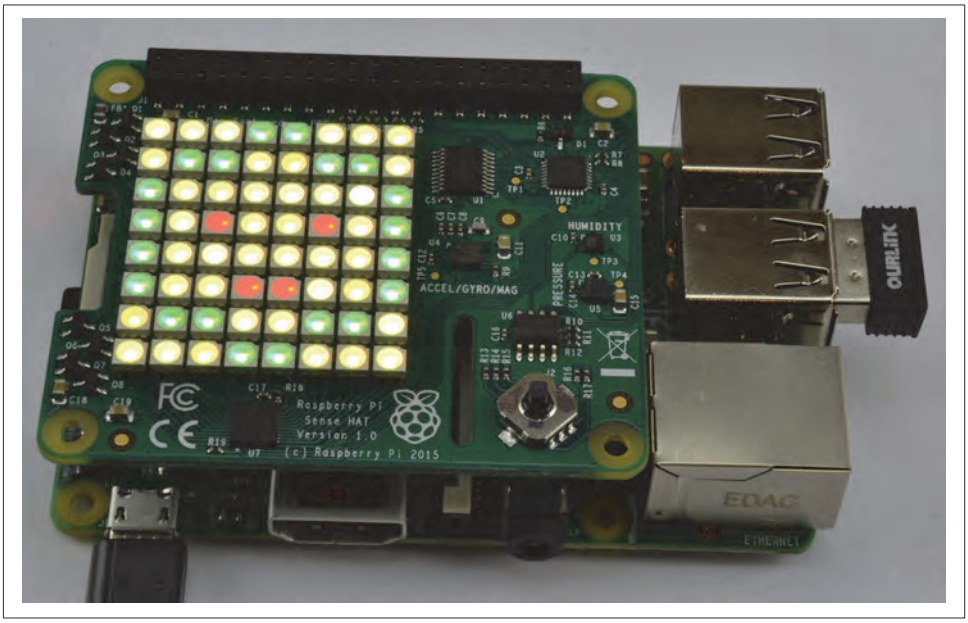


Figura 14.3. *Un Sense HAT mostrando una "imagen".*

Para saber más

Para obtener la documentación completa sobre Sense HAT vea *https://pythonhosted.org/sense-hat/api/*. Para obtener información sobre el formato de fechas y horas vaya al Capítulo 7.3.

Otros capítulos que usan Sense HAT son los Capítulos 9.17, 13.11, 13.14, 13.15 y 13.17.

14.5 Visualizar mensajes en un HAT LCD alfanumérico

Problema

Tiene unas cuantas líneas de texto que desea mostrar con claridad en una pantalla LCD.

Solución

Utilice un HAT Displayotron Pimoroni unido a su Raspberry Pi como se muestra en la Figura 14.4.

Figura 14.4. *Un HAT LCD Displayotron.*

Este HAT necesita que se habilite I2C y SPI, así que, si aún no lo ha hecho, siga el Capítulo 9.4 y el Capítulo 9.6.

Después, busque el código de la biblioteca que lo acompaña desde GitHub e instálelo utilizando los siguientes comandos:

```
$ git clone https://github.com/pimoroni/dot3k.git
$ cd dot3k/python/library
$ sudo python setup.py install
```

A modo de ejemplo, el siguiente programa (*displayotron_ip.py*) encuentra el nombre del equipo y la dirección IP de su Raspberry Pi y los muestra junto con la hora. Si todo está bien, el led retroiluminado será verde, pero si la conexión de red está inactiva, la luz de fondo cambiará a rojo.

Como con todos los ejemplos de programas en este libro, también puede descargar el programa de la sección de código de la página web del libro.

```
import dothat.lcd as lcd
import dothat.backlight as backlight
import time
from datetime import datetime
import subprocess

while True:
    lcd.clear()
    backlight.rgb(0, 255, 0)
```

```
try:
    hostname = subprocess.check_output(['hostname']).split()[0]
    ip = subprocess.check_output(['hostname', '-I']).split()[0]
    t = '{:%H:%M:%S}'.format(datetime.now())
    lcd.write(hostname)
    lcd.set_cursor_position(0, 1)
    lcd.write(ip)
    lcd.set_cursor_position(0, 2)
    lcd.write(t)
except:
    backlight.rgb(255, 0, 0)
time.sleep(1)
```

Observaciones

El programa de prueba importa las bibliotecas necesarias, incluyendo el subproceso (consulte el Capítulo 7.16) que se utilizará para encontrar la dirección IP (consulte el Capítulo 2.3) de Raspberry Pi y su nombre de equipo.

Los principales métodos en la biblioteca son:

- `lcd.clear` borra la visualización de cualquier texto
- `lcd.set_cursor_position` establece la posición para que el nuevo texto se escriba en la columna y en la fila especificada como sus dos parámetros
- `lcd.write` añade el texto suministrado como parámetro en la posición actual del cursor
- `backlight.rgb` establece los valores rojo, verde y azul de la luz de fondo (0 a 255)

Para saber más

Puede obtener más información sobre este HAT en la página de productos de PiMoroni.

Para conectar un módulo LCD de bajo coste directamente a Raspberry Pi sin un HAT vea el Capítulo 14.6.

14.6 Visualizar mensajes en un módulo LCD alfanumérico

Problema

Desea mostrar texto en una pantalla LCD alfanumérica, pero no desea utilizar una pantalla HAT ya hecha.

Solución

Utilice un módulo LCD compatible con HD44780 y conéctelo al conector GPIO.

Eso es mucho más complejo que usar uno ya preparado, pero, si está planeando un proyecto más complejo, en el cual la pantalla es un elemento y necesita acceso a otros pines GPIO, es útil saber cómo usar una pantalla LCD directamente.

Encontrará muchos módulos LCD de bajo coste, como el que se muestra conectado a Raspberry Pi en la Figura 14.5. Estos tienen 16 pines, aunque, afortunadamente, no todos ellos tienen que estar conectados a un pin GPIO.

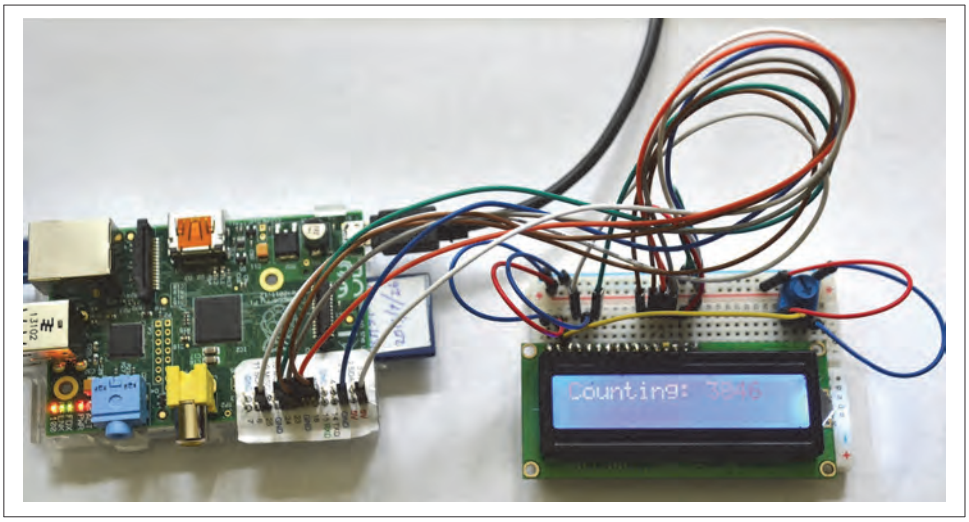


Figura 14.5. *Pantalla LCD 16 × 2 conectada a Raspberry Pi.*

Estos módulos están disponibles en varios tamaños y están especificados por el número de columnas y filas de letras que pueden mostrar. Así, por ejemplo, el módulo usado aquí se describe como 16 × 2 porque puede mostrar dos filas, cada una de 16 caracteres. Otros tamaños comunes son 8 × 1, 16 × 1, 16 × 2, 20 × 2 y 20 × 4.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Módulo LCD 16 × 2 compatible con HD44780 (vea "Módulos", página 476)
- Fila de 16 pines de encabezado (vea "Varios" en la página 477)
- Potenciómetro de 10 kΩ (vea "Resistores y condensadores", en la página 474)

La Figura 14.6 muestra la disposición de la placa de pruebas para conectar la pantalla. Normalmente, los módulos LCD no se suministran con pines de encabezado, por lo que deberá soldarlos.

El potenciómetro se utiliza para controlar el contraste de la pantalla. Tenga en cuenta que, si parece que el proyecto no ha funcionado, mueva la perilla del potenciómetro por todo el rango. Puede ser que el contraste sea tan alto que los caracteres de la pantalla no sean visibles.

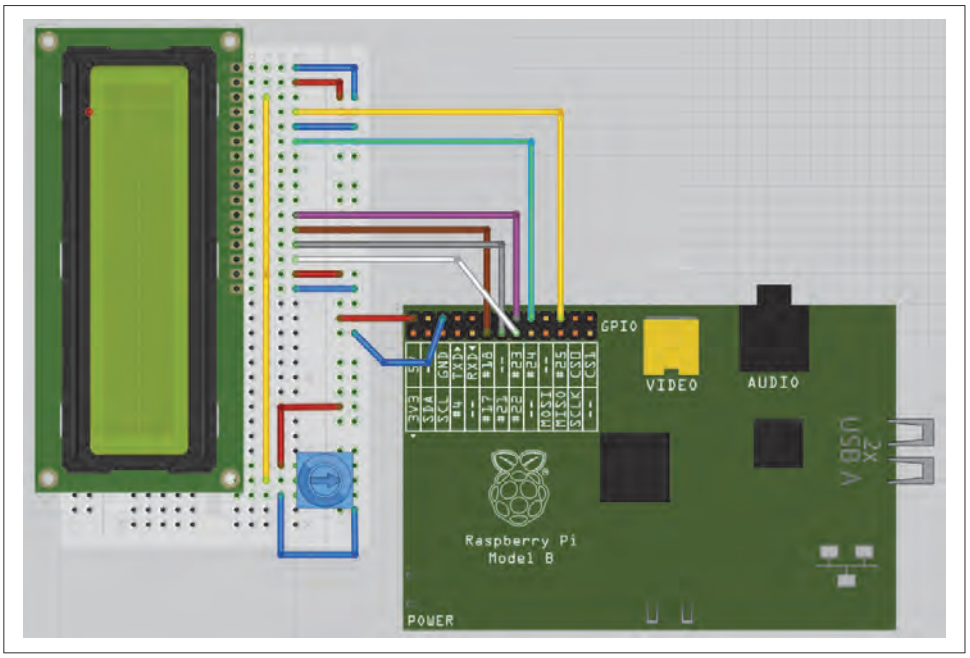


Figura 14.6. *Conexión de una pantalla LCD a Raspberry Pi.*

El código de ejemplo de Adafruit Raspberry Pi, disponible en GitHub, incluye una biblioteca para controlar las pantallas LCD que utilizan el controlador HD44780. Antes de instalarlo siga el Capítulo 9.3 para instalar la biblioteca `RPi.GPIO`.

La biblioteca de Adafruit no está instalada como una biblioteca, por lo que, para usarla, primero debe descargar la estructura de carpetas.

```
$ git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Cambie el directorio al que contiene el código de Adafruit usando:

```
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_CharLCD
```

Hay un programa de prueba que muestra la hora y la dirección IP de Raspberry Pi. Pero primero, si está usando una versión más reciente del modelo B de la Raspberry Pi (revisión 2), necesitará editar el archivo *Adafruit_CharLCD.py*:

```
$ nano Adafruit_CharLCD.py
```

Busque la línea:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 21, 22], GPIO = None):
```

Reemplace el número 21 por 27 para que la línea se vea de la siguiente manera y, después, guarde y salga del archivo:

```
def __init__(self, pin_rs=25, pin_e=24, pins_db=[23, 17, 27, 22], GPIO = None):
```

Ahora puede ejecutar el programa de ejemplo con el comando:

```
$ sudo python Adafruit_CharLCD_IPclock_example.py
```

Observaciones

Estas pantallas pueden funcionar con un bus de datos de cuatro u ocho bits y también necesitan tres pines de control. Los pines están conectados según la Tabla 14.1.

Tabla 14.1. *Raspberry Pi y las conexiones de pin de la pantalla LCD.*

Pin del módulo LCD	Pin GPIO	Notas
1	GND	0 V
2	+5 V	5 V suministro de lógica
3	No conexión	Voltaje de control de contraste
4	25	RS: Selector de registro
5	GND	RW: Lectura/escritura (siempre escritura)
6	24	EN: Habilitar
7-10	No conexión	Solo se usa en modo de ocho bits
11	23	D4: Línea de datos 4
12	17	D5: Línea de datos 5
13	21	D6: Línea de datos 6
14	22	D7: Línea de datos 7
15	+5 V	Retroiluminación led
16	GND	Retroiluminación led

El archivo *Adafruit_CharLCD.py* de la biblioteca Adafruit es responsable de establecer los valores en los pines de datos y después enviarlos al módulo de pantalla. Proporciona las siguientes funciones, que puede utilizar en sus programas:

`home()`
: Mueve a la parte superior izquierda.

`clear()`
: Borra todo el texto de la pantalla.

`setCursor(columna, fila)`
: Define la posición del cursor desde donde se escribirá el texto.

`cursor()`
: Activa la visualización del cursor.

`noCursor()`
: Desactiva la visualización del cursor (predeterminado).

`message(text)`
: Escribe el texto en la posición actual del cursor.

El siguiente programa de ejemplo muestra lo fácil que es mostrar un mensaje simple utilizando la biblioteca:

```
from Adafruit_CharLCD import Adafruit_CharLCD
from time import sleep

lcd = Adafruit_CharLCD()
lcd.begin(16,2)

i = 0

while Tue:
    lcd.clear()
        lcd.message('Counting: ' + str(i))
        sleep(1)
        i = i + 1
```

Para saber más

Este capítulo está basado en el tutorial de Adafruit de la página web de aprendizaje de Adafruit. Adafruit también vende una *placa* con una pantalla LCD adjunta que es compatible con este capítulo.

Para la conveniencia de usar una pantalla LCD alfanumérica ya hecha en un HAT, vea el Capítulo 2.3.

14.7 Utilizar una pantalla gráfica OLED

Problema

Desea adjuntar una pantalla gráfica OLED (led orgánico) a su Raspberry Pi.

Solución

Utilice una pantalla OLED, basada en el chip del controlador SSD1306, utilizando una interfaz I2C (Figura 14.7).

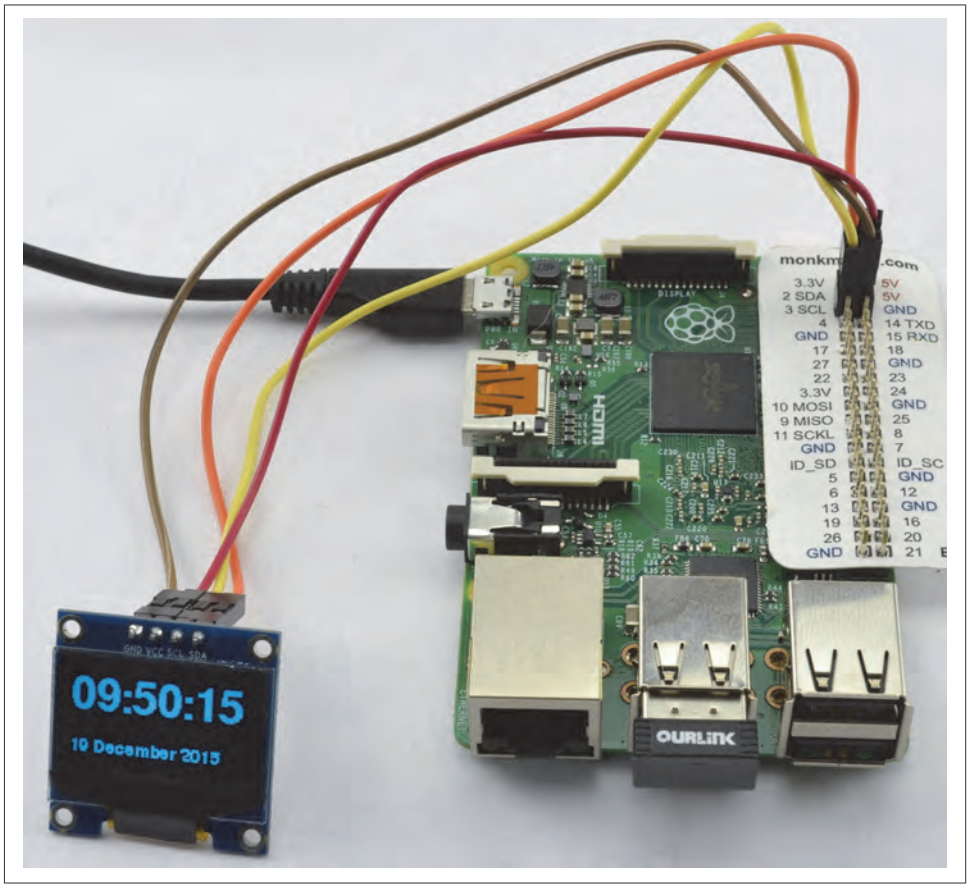


Figura 14.7. *Una pantalla OLED I2C.*

Para hacer esto necesitará:

- Cuatro cables puente de hembra a hembra (vea "Equipos para prototipos" en la página 474)
- Pantalla OLED I2C de 128×64 píxeles (vea "Módulos" en la página 476)

Algunas de estas pantallas tienen solo cuatro pines, mientras que otras tienen ocho. Las variedades de 8 pines (incluyendo las pantallas de Adafruit) tienen más pines porque tienen dos interfaces: I2C y SPI. Las pantallas de 4 pines solo tienen la I2C. La interfaz I2C es más adecuada para ser usada con Raspberry Pi, ya que la pantalla funciona a 5 V y ambos pines de interfaz para I2C (SDA y SCL) tienen salidas de 3,3 V desde Raspberry Pi. Por otro lado, el pin MISO de la interfaz SPI tiene una entrada de 3,3 V que necesita una conversión de nivel antes de ser conectado a Raspberry Pi.

Las conexiones entre Raspberry Pi y el módulo son las siguientes:

- VCC en la pantalla a 5 V en el conector GPIO de Raspberry Pi
- GND en la pantalla a GND en el conector GPIO de Raspberry Pi
- SDA en la pantalla a GPIO 2 (SDA) en el conector GPIO de Raspberry Pi
- SCL en la pantalla a GPIO 3 (SCL) en el conector GPIO de Raspberry Pi

Para que esto funcione también necesitará configurar su Raspberry Pi para I2C, así que primero siga el Capítulo 9.4.

Adafruit tiene una biblioteca para estas pantallas. Instálela usando estos comandos:

```
$ git clone https://github.com/adafruit/Adafruit_Python_SSD1306.git
$ cd Adafruit_Python_SSD1306
$ sudo python setup.py install
```

Esta biblioteca utiliza *Python Image Library* (PIL), que se puede instalar mediante el comando:

```
$ sudo pip install pillow
```

El ejemplo de código *old_clock.py* muestra la hora y la fecha en la pantalla OLED. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro.

```
from oled.device import ssd1306
from oled.render import canvas
from PIL import ImageFont
import time
from datetime import datetime

#Configurar la pantalla
device = ssd1306(port=1, address=0x3C)
font_file = '/usr/share/fonts/truetype/freefont/FreeSansBold.ttf'
small_font = ImageFont.truetype('FreeSans.ttf', 12, filename=font_file)
large_font = ImageFont.truetype('FreeSans.ttf', 33, filename=font_file)

#Muestra un mensaje en 3 líneas, la primera en una fuente grande
def display_message(top_line, line_2):
    global device
    with canvas(device) as draw:
        draw.text((0, 0), top_line, font=large_font, fill=255)
        draw.text((0, 50),  line_2, font=small_font, fill=255)

while True:
    now = datetime.now()
    date_message = '{:%d %B %Y}'.format(now)
    time_message = '{:%H:%M:%S}'.format(now)
    display_message(time_message, date_message)
    time.sleep(0.1)
```

Cada dispositivo esclavo I2C tiene una dirección que se establece en la línea:

```
device = ssd1306(port=1, address=0x3C)
```

Esto se fija en 3C (hexadecimal) para muchos de los módulos I2C de bajo coste. Sin embargo, puede variar, por lo que deberá comprobar cualquier documentación que venga con el dispositivo o usar herramientas I2C (Capítulo 9.5) para listar todos los dispositivos I2C conectados al bus para que pueda ver la dirección de su pantalla.

Observaciones

Las pantallas led orgánicas (OLED) pequeñas son baratas, no utilizan mucha corriente y tienen una alta resolución a pesar de su tamaño diminuto. Están reemplazando las pantallas LCD en muchos productos de consumo.

Para saber más

Estas instrucciones son para interfaces I2C de 4 pines. Si realmente quiere usar la interfaz SPI, eche un vistazo al tutorial de Adafruit sobre esto.

14.8 Usar tiras de ledes RGB direccionables

Problema

Desea conectar una tira de ledes RGB (neopíxeles) a su Raspberry Pi.

Solución

Utilice una tira de ledes basada en los chips WS2812 RGB LED de su Raspberry Pi.

El uso de estas tiras de ledes (Figura 14-8) puede ser muy fácil con una conexión directa a Raspberry Pi y a la alimentación de los ledes suministrada por Raspberry Pi. Este es el escenario que debería funcionar bien, pero vea la Observaciones de este capítulo para ver los posibles problemas de este enfoque y las maneras de garantizar el uso de las tiras de led sin problemas.

No encienda los ledes desde la fuente de 3,3 V de Pi

Puede ser tentador encender los ledes desde el pin de alimentación de 3,3 V en el conector GPIO; no lo haga (consulte el Capítulo 9.3). Usando este pin podría dañar fácilmente su Raspberry Pi.

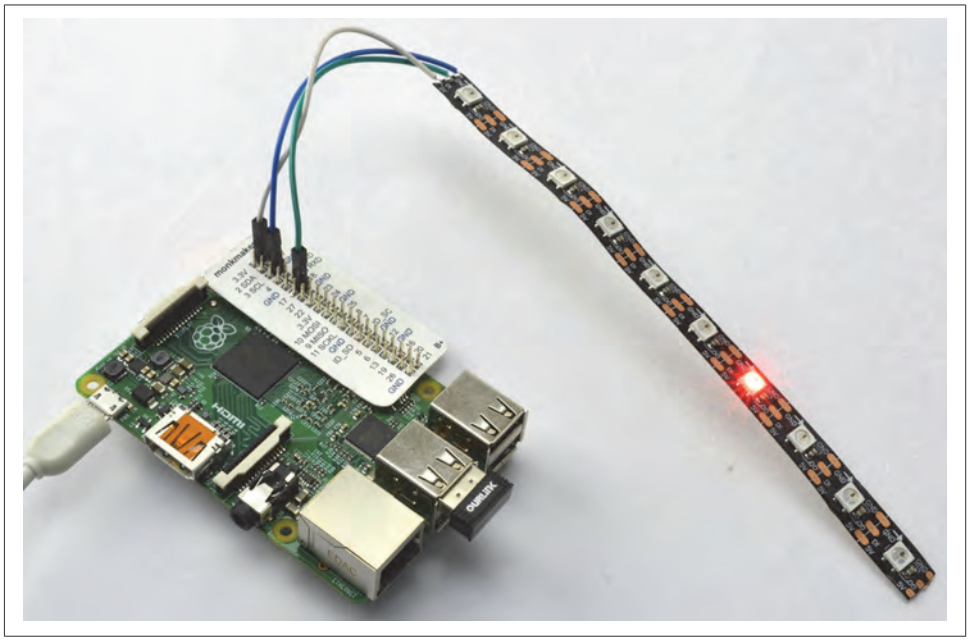


Figura 14.8. *Una tira de 10 ledes.*

La tira de ledes usada en la Figura 14.8 se corta de un carrete. En este caso, hay 10 ledes. Cada led puede utilizar hasta 60 mA, con lo que 10 es probablemente un límite razonable para el número de ledes que se pueden utilizar sin tener que disponer de una fuente de alimentación separada para la tira de led (consulte el apartado «Observaciones»).

Para conectar la tira a Raspberry Pi se cortan por un extremo los cables puente con los conectores hembra y se sueldan los extremos del cable a las tres conexiones en la tira de ledes: GND, DI (*Data In*) y 5 V. Después, estos pueden ser conectados a los pines GPIO GND, GPIO 18 y 5 V, respectivamente.

Observe que la tira de ledes tiene una flecha impresa en ella (Figura 14.9). Asegúrese de que cuando la suelde, comienza desde el extremo del corte hacia la izquierda de la flecha.

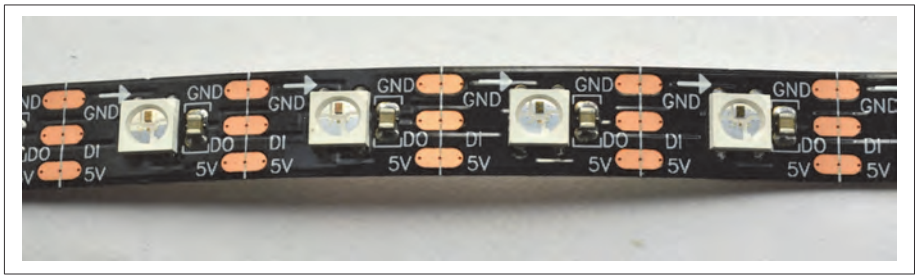


Figura 14.9. *Una tira de ledes de cerca.*

Encontrar una biblioteca para el WS2812 que funcione con todos los modelos de Raspberry Pi es muy complicado. La biblioteca original fue desarrollada por Jeremy Garff, y con el apoyo de Raspberry Pi 2 se añadió la de Richard Hurst. Adafruit también tuvo que ver en la creación de la biblioteca. Sin embargo, el cambio de Raspbian a Linux Kernel 4.1.6 cambió algunas cosas. La HAT Pimoroni Unicorn (que usa una versión modificada de la biblioteca de Richard Hurst) parece ser la más fiable. Estando vinculado a un producto comercial, es muy probable que esta versión se mantenga para cualquier cambio futuro en las placas Raspbian o Raspberry Pi. Para instalar la biblioteca ejecute los siguientes comandos:

```
$ git clone https://github.com/pimoroni/unicorn-hat.git
$ cd unicorn-hat/python/rpi-ws281x
$ make
$ sudo python setup.py install
```

El siguiente programa de ejemplo (*led_strip.py*) establecerá el led rojo en las posiciones sucesivas a lo largo de la tira. Puede descargar el programa de la sección de código desde la web del libro.

```
import time
from neopixel import *

# Configuración de la tira de ledes:
LED_COUNT      = 10      # Número de píxeles de led.
LED_PIN        = 18      # Pin GPIO conectado a los píxeles (debe soportar PWM).
LED_FREQ_HZ    = 800000  # Frecuencia del led en hertz (generalmente 800 khz)
LED_DMA        = 5       # Canal DMA para generar señal (try 5)
LED_BRIGHTNESS = 255     # Se establece 0 para oscuros y 255 para brillantes
LED_INVERT     = False   # True para invertir la señal

RED = Color(255, 0, 0)
NO_COLOR = Color(0, 0, 0)

strip = Adafruit_NeoPixel(LED_COUNT, LED_PIN, LED_FREQ_HZ, LED_DMA,
LED_INVERT, LED_BRIGHTNESS)
```

```
strip.begin()

def clear():
  for i in range(0, LED_COUNT):
    strip.setPixelColor(i, NO_COLOR)
  strip.show()


i = 0
while True:
  clear()
  strip.setPixelColor(i, RED)
  strip.show()
  time.sleep(1)
  i += 1
  if i >= LED_COUNT:
    i = 0
```

Los parámetros en la sección *Configuración de la tira de ledes* pueden ser modificados para las diferentes configuraciones de los ledes. Si tiene un número diferente de ledes en su tira, modifique `LED_COUNT`. El resto de las constantes no tienen que cambiarse.

Cada uno de los ledes pueden tener su color establecido independientemente de los demás usando el método `setPixelColor`. El primer parámetro es la posición (que comienza por 0) del led cuyo color desea fijar. El segundo parámetro es el color. Los cambios en los colores del led no se actualizarán hasta que no se llame al método `show`.

Observaciones

Cada led de la tira puede utilizar un máximo de aproximadamente 60 mA. Solo lo hará si los tres canales de color (rojo, verde y azul) están con el máximo brillo (255). Por lo tanto, si va a utilizar una gran cantidad de ledes, tendrá que utilizar una fuente de alimentación separada de 5 V. La Figura 14.10 muestra cómo conectar una fuente de alimentación separada. Un adaptador de terminal hembra de jack a tornillo de corriente continua (“Equipos para prototipos”, en la página 474) facilitará la conexión de la fuente de alimentación externa a la placa de pruebas.

La hoja de datos para el WS2812 establece que la señal de entrada debe tener una lógica alta de, al menos, 4 V para una fuente de alimentación de 5 V para los ledes. Por lo tanto, cuando se utiliza esto con una salida GPIO de 3,3 V de Raspberry Pi, no hay garantía de que los ledes funcionen de manera fiable. Habiendo dicho esto, nunca he tenido ningún problema.

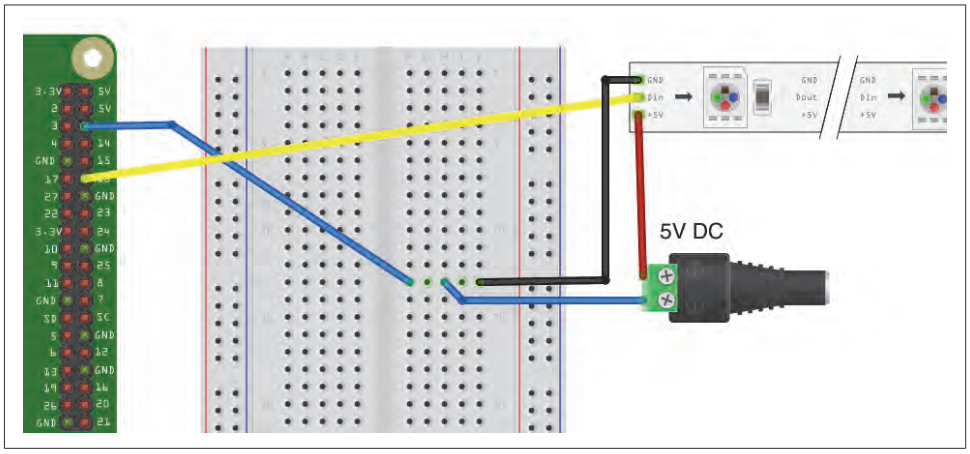


Figura 14.10. *Alimentación de una tira de ledes con una fuente de alimentación externa.*

Si tiene problemas con que la señal no sea suficientemente fuerte, puede usar un transistor conversor de niveles lógicos como se muestra en la Figura 14.11. Un transistor adecuado es el 2N7000 (tenga en cuenta que un 2N3904 no funcionará en esta situación, ya que no puede responder con la suficiente rapidez como para invertir los pulsos).

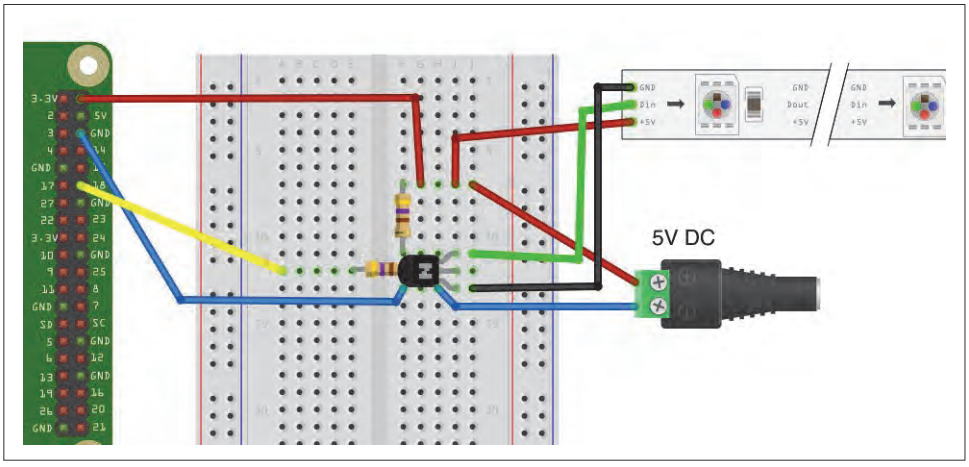


Figura 14.11. *Uso de un conversor de niveles lógicos.*

Este transistor convierte la señal de 3,3 V a 5 V, pero tiene el efecto secundario de invertir la señal, por lo que, si la salida del pin GPIO es ALTA, la señal en la banda led será BAJA y viceversa. Esto significa que la lógica en su programa necesitará ser modificada cambiando la línea:

```
LED_INVERT     = False
```

por:

```
LED_INVERT     = True
```

Para saber más

Una alternativa al uso de un transistor para elevar el nivel de señal de 3,3 V a 5 V es usar un módulo de conversor de niveles (vea el Capítulo 9.14).

El HAT Unicorn, cuya biblioteca se utiliza aquí, tiene 64 ledes WS2812 dispuestos en una cuadrícula de 8×8. Puede obtener más información sobre esta pantalla en *https://shop.pimoroni.com/products/unicorn-hat*.

CAPÍTULO 15

El Internet de las cosas

15.1 Introducción

El *Internet de las cosas* (IoT, por sus siglas en inglés) es la creciente red de dispositivos conectados a Internet. No solo significa más y más ordenadores que usan navegadores, sino aparatos reales y tecnología portátil. Esto incluye todo tipo de domótica desde electrodomésticos inteligentes e iluminación hasta sistemas de seguridad e incluso alimentadores para mascotas que funcionan a través de Internet, así como muchos proyectos menos prácticos pero divertidos.

En este capítulo aprenderá cómo su Raspberry Pi puede participar en el Internet de las cosas de varias maneras.

15.2 Controlar las salidas GPIO mediante una interfaz web

Problema

Desea controlar las salidas GPIO usando una interfaz web para su Raspberry Pi.

Solución

Utilice la biblioteca de servidores web de Python `bottle` (Capítulo 7.18) para crear una interfaz web HTML con el fin de controlar el puerto GPIO.

Para hacer esto necesitará:

- Placa de pruebas y cables puente (vea “Equipos para prototipos”, página 474)
- Tres resistores de 1 kΩ (vea “Resistores y condensadores” en la página 474)
- Tres ledes (vea “Optoelectrónica” en la página 476)
- Interruptor pulsador táctil (vea “Varios” en la página 477)

El diseño de la placa de pruebas para esto se muestra en la Figura 15.1.

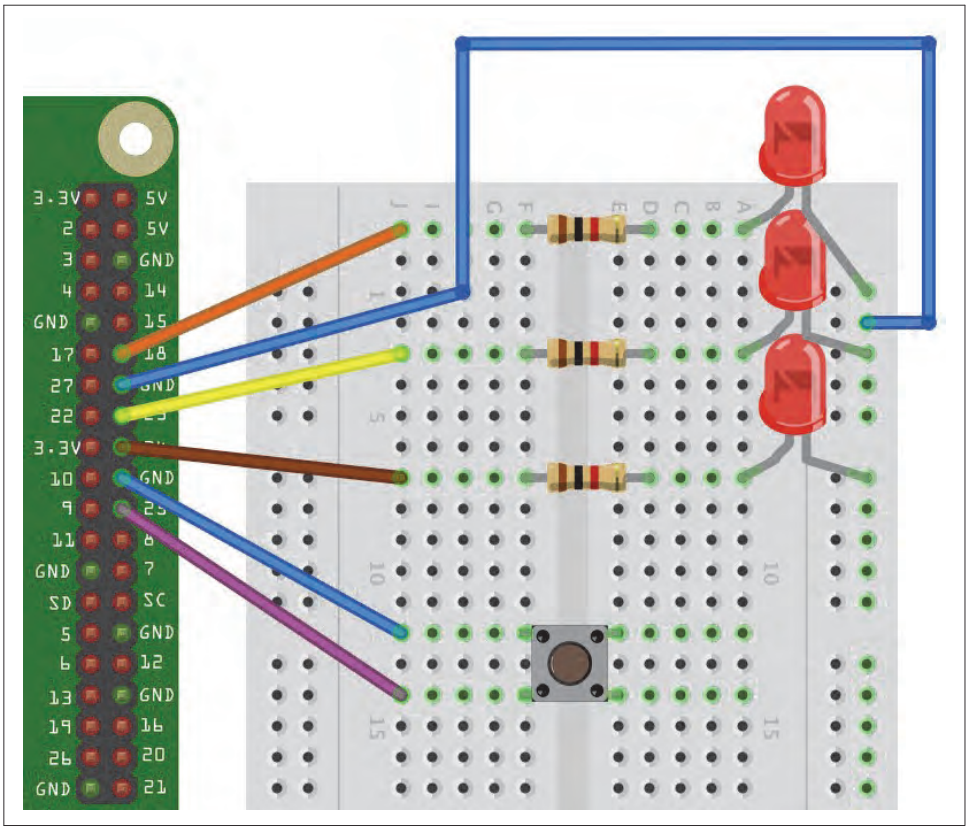


Figura 15.1. *Distribución de la placa de pruebas que controla las salidas GPIO desde una página web.*

Una alternativa al uso de una placa de pruebas es adjuntar una Raspberry Squid y un botón Squid (revise el Capítulo 9.11 y el Capítulo 9.12). Estos pueden estar conectados directamente a los pines GPIO de Raspberry Pi, como se muestra en la Figura 15.2.

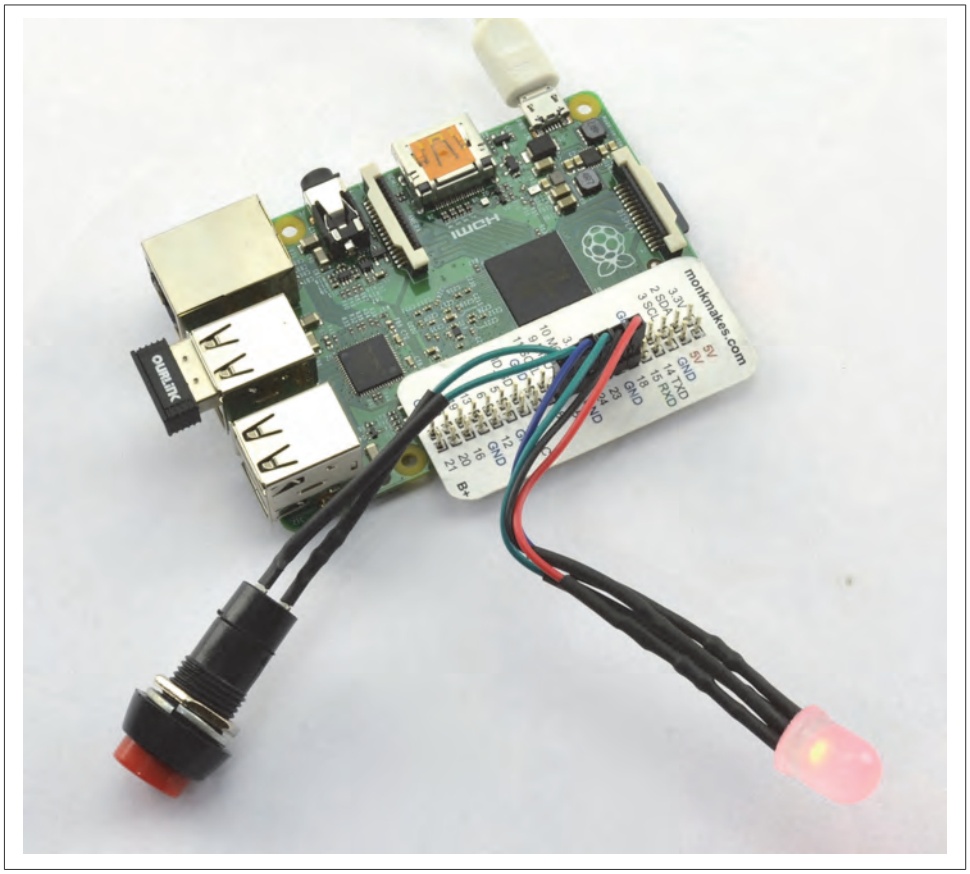


Figura 15.2. *Raspberry Squid y botón Squid.*

Para instalar la biblioteca `bottle` vea el Capítulo 7.18.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *web_control.py*. También encontrará un programa llamado *web_control_test.py*. Este programa simplemente prueba el *hardware* haciendo parpadear los ledes en secuencia y mostrando el estado del interruptor.

Aquí está el programa *web_control.py*:

```
from bottle import route, run
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
led_pins = [18, 23, 24]
```

```
led_states = [0, 0, 0]
switch_pin = 25

GPIO.setup(led_pins[0], GPIO.OUT)
GPIO.setup(led_pins[1], GPIO.OUT)
GPIO.setup(led_pins[2], GPIO.OUT)
GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def switch_status():
    state = GPIO.input(switch_pin)
    if state:
        return 'Up'
    else:
        return 'Down'

def
    html_for_led(led)
    : l = str(led)
    result = " <input type='button' onClick='changed(" + l +
")' value='LED " + l + "'/>"
    return result

def update_leds():
    for i, value in enumerate(led_states):
        GPIO.output(led_pins[i], value)

@route('/')
@route('/<led>')
def index(led):
    if led >= '0' and led <=
        '9': led_num = int(led)
        led_states[led_num] = not led_states[led_num]
        update_leds()
    response = "<script>"
    response += "function changed(led)"
    response += "{"
    response += " window.location.href='/' + led"
    response += "}"
    response += "</script>"

    response += '<h1>GPIO Control</h1>'
    response += '<h2>Button=' + switch_status() + '</h2>'
    response += '<h2>LEDs</h2>'
    response += html_for_led(0)
    response += html_for_led(1)
    response += html_for_led(2)
    return response

try:
    run(host='0.0.0.0', port=80)
finally:
    print('\nCleaning up')
    GPIO.cleanup()
```

Debe ejecutar el programa como superusuario:

```
sudo python web_control.py
```

Si se inicia correctamente, debería ver un mensaje como este:

```
Bottle server starting up (using WSGIRefServer())...
Listening on http://0.0.0.0:80/
Hit Ctrl-C to quit.
```

Abra una ventana del navegador desde cualquier equipo de su red, incluso desde Raspberry Pi, y vaya a la dirección IP de Raspberry Pi. Debería aparecer la interfaz web mostrada en la Figura 15.3.

Figura 15.3. *Una interfaz web para GPIO.*

Si hace clic en uno de los tres botones de led en la parte inferior de la pantalla, verá que el led adecuado se enciende y se apaga.

Además, si mantiene presionado el botón a medida que vuelve a cargar la página web, debería ver que el texto junto a *Button* dice *Down* en lugar de *Up*.

Observaciones

Para entender cómo funciona el programa, primero tenemos que ver cómo funciona una interfaz web. Todas las interfaces web dependen de un servidor en algún lugar (en este caso, un programa en Raspberry Pi) que responde a las solicitudes de un navegador web.

Cuando el servidor recibe una solicitud examina la información que viene con ella y genera un HTML (*HyperText Markup Language*) de respuesta.

Si la solicitud web es solo la página principal (*http://192.168.1.8/*), el `led` recibirá un valor predeterminado de `n`. Sin embargo, si fuéramos a navegar por la URL *http://192.168.1.8/2*, el 2 del final de la URL sería asignado al parámetro `led`.

El parámetro `led` se utiliza para determinar que el led 2 debe ser activado.

Para poder acceder a esta URL de activación de led, debemos arreglar algunas cosas para que cuando se pulse el botón para el led 2, la página se recargue con este parámetro adicional al final de la URL. El truco es incluir una función JavaScript en el HTML. Cuando el navegador ejecute esta función, hará que la página se vuelva a cargar con el parámetro adicional adecuado.

Todo esto significa que tenemos una situación bastante simple en la que el programa de Python está generando código en JavaScript para ser ejecutado más tarde por el navegador. Las líneas que generan esta función JavaScript son:

```
response = "<script>"
response += "function changed(led)"
response += "{"
response += " window.location.href='/' + led"
response += "}"
response += "</script>"
```

En lugar de repetir el HTML para cada uno de los botones, esto se genera mediante la función `html_for_led`:

```
def html_for_led(led):
    l = str(led)
    result = " <input type='button' onClick='changed(" + l + ")'
    value='LED " + l + "'/>"
    return result
```

Este código se utiliza tres veces, una para cada botón, y vincula una pulsación de botón con la función `changed`. La función también se suministra con el número de led como su parámetro.

El código que realmente cambia las salidas GPIO se localiza en la función `update_leds`. A esta función se le llama cada vez que el servidor recibe una petición que incluye un número de ledes que requieren activación:

```
def update_leds():
    for i, value in enumerate(led_states):
        GPIO.output(led_pins[i], value)
```

La función simplemente itera sobre un conjunto de estados, ajustando cada salida a su vez a su valor actual.

La siguiente línea, en la función `index`, cambia el valor en el conjunto de estado por el led indicado en el parámetro `led`:

```
led_states[led_num] = not led_states[led_num]
```

El proceso de informar el estado del botón es mucho más sencillo. Es simplemente el acto de leer el estado de la entrada y de generar el HTML que reporte el estado del botón. Todo esto está contenido en la función `switch_status`:

```
def switch_status():
    state = GPIO.input(switch_pin)
    if state:
        return 'Up'
    else:
        return 'Down'
```

Para saber más

Para obtener más información sobre el uso de `bottle` vea la documentación de `bottle`.

15.3 Visualizar las lecturas de los sensores en una página web

Problema

Desea visualizar las lecturas del sensor de su Raspberry Pi en una página web que se actualice automáticamente.

Solución

Utilice el servidor web `bottle` y algunos JavaScript para actualizar automáticamente su pantalla.

El ejemplo de la Figura 15.4 muestra la temperatura de la CPU de Raspberry Pi usando su sensor incorporado.

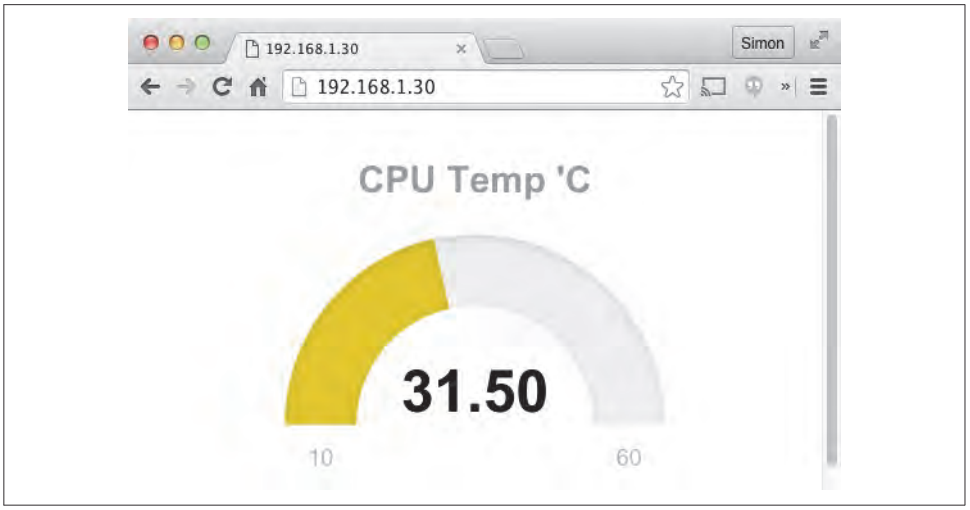


Figura 15.4. *Visualización de la temperatura de la CPU de Raspberry Pi.*

Para instalar la biblioteca `bottle` consulte el Capítulo 7.18.

Hay cuatro archivos para este ejemplo que están en la carpeta *web_sensor*:

web_sensor.py
: Contiene el código de Python para el servidor `bottle`

main.html
: Contiene la página web que se mostrará en su navegador

justgage.1.0.1.min.js
: Una biblioteca de JavaScript de terceros que muestra el medidor de temperatura

raphael.2.1.0.min.js
: Una biblioteca usada por la biblioteca `justgage`

Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la web del libro.

Para ejecutar el programa, cambie el directorio a `web_sensor` y luego ejecute el programa Python usando:

```
$ sudo python web_sensor.py
```

A continuación, abra un navegador, ya sea en la misma Raspberry Pi o en cualquier ordenador de la misma red, e introduzca la dirección IP de Raspberry Pi en la barra de direcciones del navegador. Debería aparecer la página mostrada en la Figura 15.4.

Observaciones

El programa principal (*web_sensor.py*) es en realidad bastante conciso:

```
import os, time
from bottle import route, run, template

def cpu_temp():
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp = dev.read()[5:-3]
    return cpu_temp

@route('/temp')
def temp():
    return cpu_temp()

@route('/')
def index():
    return template('main.html')

@route('/raphael')
def index():
    return  template('raphael.2.1.0.min.js')
```

```
@route('/justgage')
def index():
    return  template('justgage.1.0.1.min.js')

run(host='0.0.0.0', port=80)
```

La función `cpu_temp` lee la temperatura de la CPU de Raspberry Pi (Capítulo 13.10).

Se definen cuatro rutas para el servidor web `bottle`. La primera (`/temp`) devuelve una cadena que contiene la temperatura de la CPU en grados C. La segunda (`/`) devuelve la plantilla HTML principal de la página (`main.html`). Las otras dos dan acceso a copias de las bibliotecas `raphael` y `justgage` de JavaScript.

El archivo `main.html` contiene el JavaScript para representar la interfaz de usuario.

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"
type="text/javascript" charset="utf-8"></script>
<script src="raphael"></script>
<script src="justgage"></script>


<script>
function callback(tempStr, status){
if (status == "success") {
    temp = parseFloat(tempStr).toFixed(2);
    g.refresh(temp);
    setTimeout(getReading, 1000);
}
else {
    alert("There was a problem");
    }
}


function getReading(){
    $.get('/temp', callback);
}
</script>
</head>


<body>
<div id="gauge" class="200x160px"></div>


<script>
var g = new
    JustGage({
    id: "gauge",
```

```
        value: 0,
        min: 10,
        max: 60,
        title: "CPU Temp 'C"
    });
    getReading();
    </script>


    </body>
    </html>
```

Las bibliotecas JQuery, Raphael y JustGage son todas importadas (JQuery de https://developers.google.com/speed/libraries/#jquery y las otras dos de copias locales).

Obtener una lectura desde Raspberry Pi hasta la ventana del navegador es un proceso de dos etapas. Primero, se llama a la función `getReading`. Esto envía una solicitud web con la ruta `/temp` a `web_sensor.py` y especifica una función llamada `callback` para que se ejecute cuando se complete la solicitud web. La función `callback` es responsable de actualizar la pantalla antes de establecer un tiempo de espera para llamar a `getReading` de nuevo después de un segundo.

Para saber más

Para obtener un ejemplo de cómo utilizar Tkinter para mostrar los valores de los sensores en una aplicación en lugar de en una página web, vea el Capítulo 13.20.

La biblioteca JustGage tiene todo tipo de opciones útiles para mostrar los valores del sensor. Vea *http://justgage.com/* para más información.

15.4 Enviar correos electrónicos y otras notificaciones con IFTTT

Problema

Quiere que su Raspberrry Pi envíe notificaciones por correo electrónico, estado de Facebook o Twitter de manera flexible.

Solución

Haga que su Raspberry Pi envíe las solicitudes al canal *Maker* de *Si ocurre esto, Haz esto otro* (*If This Then That* - IFTTT) para activar las notificaciones configurables. Por ejemplo, que se le envíe un correo electrónico cuando la temperatura de la CPU de su Raspberry Pi exceda un umbral.

Antes de empezar, cree una cuenta con IFTTT. Visite *www.ifttt.com* y regístrese.

El siguiente paso es crear una nueva *receta* IFTTT. IFTTT utiliza la palabra *recipe* (receta) para referirse a una regla, como *Cuando reciba una solicitud web de una Raspberry Pi, envíe un email*. Haga clic en el botón Create a Recipe. Esto le pedirá que introduzca la parte SI (IF) de la receta y más tarde la parte THAT (ESTO).

En este caso, la parte IF THIS (el activador) va a ser la recepción de una solicitud web desde su Raspberry Pi, así que haga clic en THIS e introduzca *Maker* en el campo de búsqueda para encontrar el canal Maker. Seleccione el canal Maker y cuando se le pida que elija un activador (*trigger*), seleccione la opción de recibir una solicitud web. Esto abrirá el formulario que se muestra en la Figura 15.5.

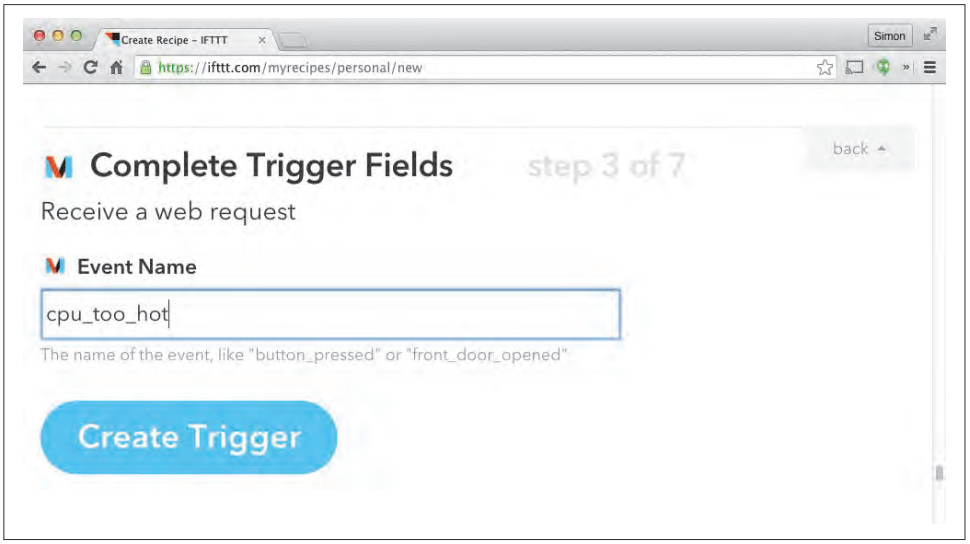


Figura 15.5. *El formulario de activación de recibir una solicitud web.*

Introduzca el texto *cpu_too_hot* en el campo del nombre del evento y haga clic en *Create Trigger* (crear activador).

Esto le llevará a la parte THAT de la receta, la *acción*, y tendrá que seleccionar un canal de acción. Aquí hay muchas opciones, pero para este ejemplo usará el canal *email*, así que escriba email en el campo de búsqueda y luego seleccione el canal de email. Verá varios canales relacionados con el correo electrónico, incluido Gmail. Use el canal *email*, aunque envíe los correos desde una cuenta Gmail.

Una vez seleccionado el canal email seleccione la acción de enviar un correo electrónico y se mostrará el formulario de la Figura 15.6.

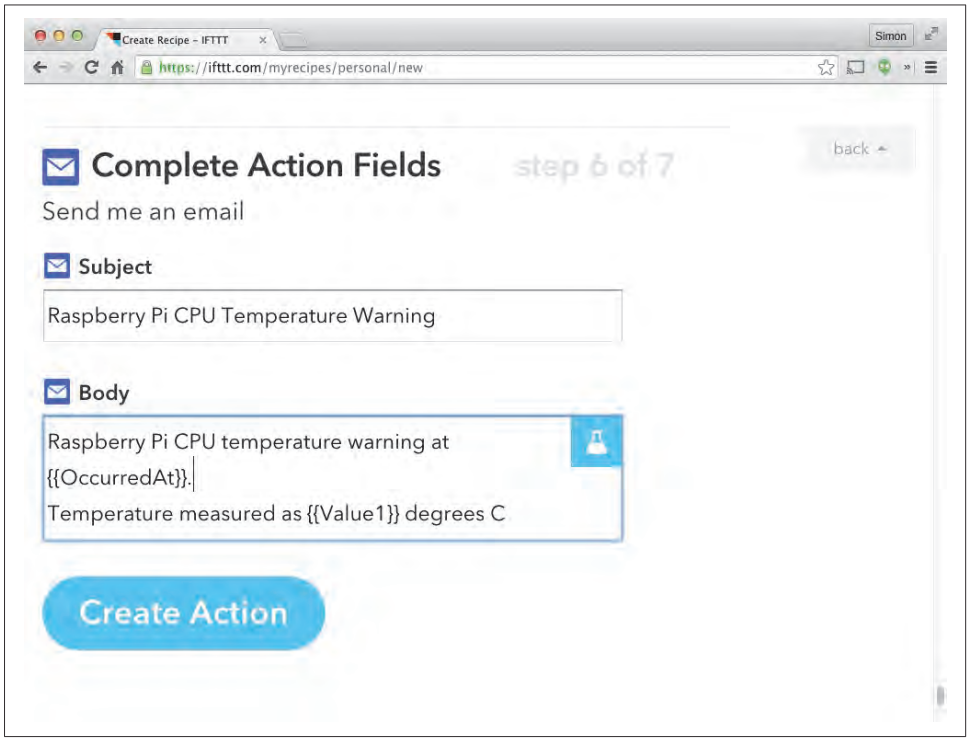


Figura 15.6. *Configuración de la acción de enviar un email.*

Cambie el texto para que parezca como se muestra en la Figura 15.6. Tenga en cuenta que los valores especiales OccurredAt y Value1 estarán rodeados por {{ y }}. Estos valores se denominan ingredientes y son valores variables que se tomarán de la solicitud web y se sustituirán en el asunto y en el cuerpo del correo electrónico.

Haga clic en *Create Action* y *Create recipe* para finalizar la creación de la receta.

Eso es todo; cuando Raspberry Pi esté listo para mandarle peticiones web, comenzará a enviar correos.

El programa de Python para enviar las solicitudes web se puede encontrar en las descargas de la web del libro, donde pone *ifttt_cpu_temp.py*.

```
import time, os, urllib, urllib2

MAX_TEMP = 37.0
MIN_T_BETWEEN_WARNINGS = 60 # Minutos

EVENT = 'cpu_too_hot'
BASE_URL = 'https://maker.ifttt.com/trigger/'
KEY = 'cyR3vPNFlP9K32W4NZB9cd'
```

```
def send_notification(temp):
    data = urllib.urlencode({'value1' : str(temp)})
    url = BASE_URL + EVENT + '/with/key/' + KEY
    response = urllib2.urlopen(url=url, data=data)
    print(response.read())


def cpu_temp():
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp = dev.read()[5:-3]
    return float(cpu_temp)

while True:
    temp = cpu_temp()
    print("CPU Temp (C): " + str(temp))
    if temp >MAX_TEMP:
        print("CPU TOO HOT!")
        send_notification(temp)
        print("No more notifications for: " + str(MIN_T_BETWEEN_WARNINGS) +
" mins")
        time.sleep(MIN_T_BETWEEN_WARNINGS * 60)
    time.sleep(1)
```

Antes de ejecutar el programa, necesitará obtener una clave de acceso para el canal Maker Action seleccionando Channels (en la parte superior de la página web de IFTTT) y luego buscando Maker. Conéctese al canal y, a continuación, aparecerá su clave en la parte inferior de la página web (Figura 15-7).

Pegue la clave en *ifttt_cpu_temp.py* en la línea que pone KEY=, y luego ejecute el programa usando:

```
$ sudo python ifttt_cpu_temp.py
```

Puede aumentar la temperatura de la CPU reproduciendo un vídeo y envolviendo *temporalmente* su Raspberry Pi en papel de burbujas. Una vez que se caliente, deberá recibir un correo electrónico como se ve en la Figura 15.8. Observe cómo se han sustituido los valores en el correo electrónico.

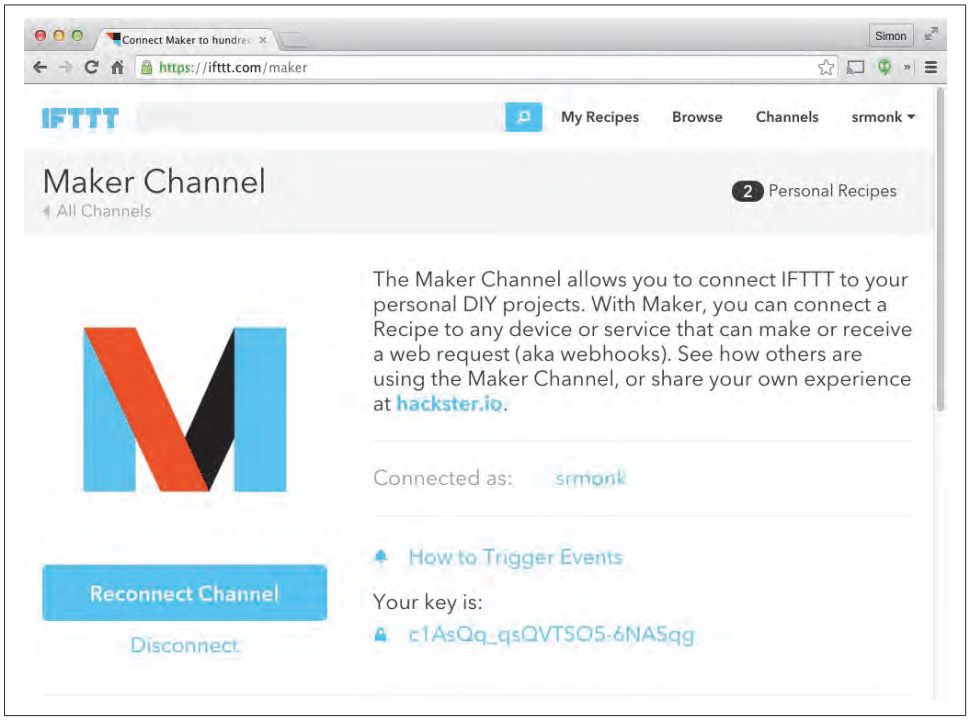


Figura 15.7. *Obtener una clave IFTTT.*

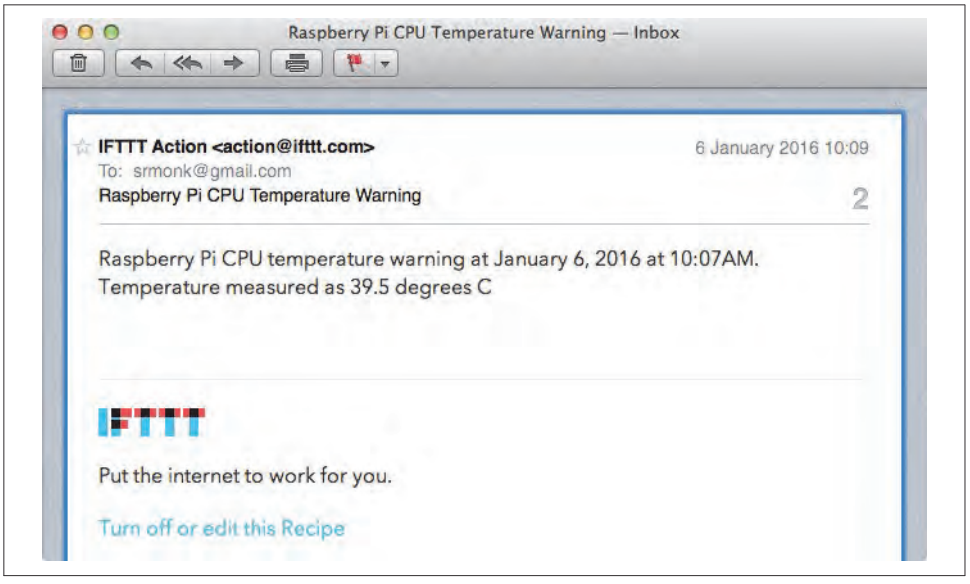


Figura 15.8. *La notificación de correo electrónico.*

Observaciones

La mayor parte de la acción para este programa tiene lugar en la función `send_notification`. En primer lugar, se construye la URL que incluye la clave y el parámetro de solicitud `value1` (que contiene la temperatura), y luego se utiliza el `urllib2` de Python para enviar la solicitud web a IFTTT.

El bucle principal comprueba continuamente la temperatura frente al `MAX_TEMP`, y si la temperatura de la CPU supera `MAX_TEMP`, se envía la solicitud web y se inicia una pausa como se especifica en `MIN_T_BETWEEN_WARNINGS` para evitar que la bandeja de entrada se llene de notificaciones.

Como alternativa al uso de IFTTT, podría enviar un correo electrónico directamente utilizando el Capítulo 7.17. Sin embargo, si utiliza IFTTT, no está limitado a notificaciones de correos; puede utilizar cualquiera de los canales de acción disponibles en IFTTT sin tener que escribir ningún código.

Para saber más

Para enviar un correo electrónico directamente desde Python vea el Capítulo 7.17.

El código para medir la temperatura de la CPU se describe en el Capítulo 13.10.

15.5 Enviar tuits con ThingSpeak

Problema

Desea enviar automáticamente tuits desde su Raspberry Pi, por ejemplo, para irritar a la gente diciéndoles la temperatura de su CPU.

Solución

Puede utilizar el Capítulo 15.4 y cambiar el Canal de Acción para que sea Twitter. Sin embargo, el servicio ThingSpeak es una buena alternativa.

ThingSpeak es similar a IFTTT, pero está dirigido directamente a proyectos de IoT. Le permite crear canales que pueden almacenar y recuperar datos mediante solicitudes web. También tiene varias acciones como ThingTweet, que proporciona un contenedor de servicios web en Twitter. Esto es más fácil que usar la API de Twitter, que requiere que registre su aplicación en Twitter.

Primero visite *https://thingspeak.com* y regístrese. Después, seleccione la acción ThingTweet (Figura 15.9). Se le pedirá que inicie sesión en Twitter.

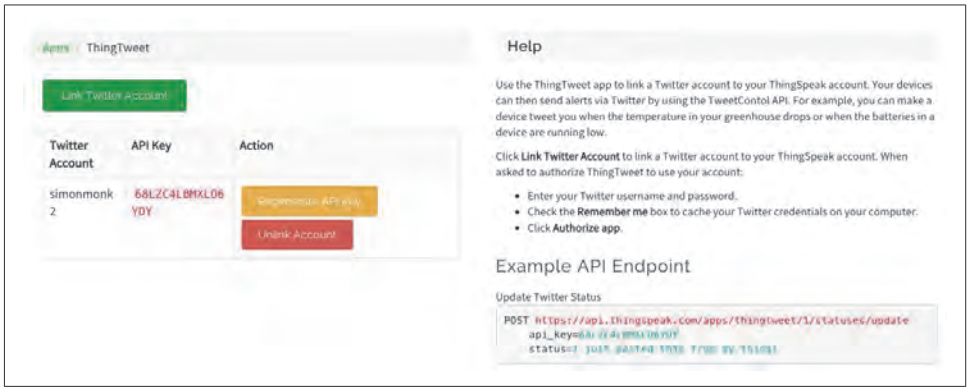


Figura 15.9. *Acción ThingTweet.*

El programa de Python para enviar la solicitud web que desencadena el tuit se puede encontrar en las descargas de la página web del libro, donde pone *send_tweet.py*:

```
import time, os, urllib, urllib2

MAX_TEMP = 37.0
MIN_T_BETWEEN_WARNINGS = 60 # Minutos

BASE_URL = 'https://api.thingspeak.com/apps/thingtweet/1/statuses/update/'
KEY = '68LZC4LBMXLO6YDY'

def send_notification(temp):
    status = 'Raspberry Pi getting hot. CPU temp=' + temp
    data = urllib.urlencode({'api_key' : KEY, 'status': status})
    response = urllib2.urlopen(url=BASE_URL, data=data)
    print(response.read())

def cpu_temp():
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp = dev.read()[5:-3]
    return cpu_temp

while True:
    temp = cpu_temp()
    print("CPU Temp (C): " + str(temp))
    if temp >MAX_TEMP:
        print("CPU TOO HOT!")
        send_notification(temp)
        print("No more notifications for: " + str(MIN_T_BETWEEN_WARNINGS) +
" mins")
        time.sleep(MIN_T_BETWEEN_WARNINGS * 60)
    time.sleep(1)
```

Igual que con el Capítulo 15.4, necesitará pegar la clave de la Figura 15.9 en el código antes de ejecutar el programa. Ejecute y pruebe el programa de la misma manera que en el Capítulo 15.4.

Observaciones

El código es muy similar al del Capítulo 15.4. La principal diferencia está en la función `send_notification`, que construye el tuit y luego envía la solicitud web con el mensaje como estado del parámetro.

Para saber más

Para obtener la documentación completa del servicio ThingSpeak consulte *https://uk.mathworks.com/help/thingspeak/*.

En el Capítulo 15.6 se utiliza el popular Cheerlights, implementado en ThingSpeak. Y, en el Capítulo 15.7, aprenderá a usar ThingSpeak para recopilar datos de sensores.

15.6 CheerLights

Problema

Desea conectar su Raspberry Pi a un led RGB y participar en el popular proyecto Cheerlights.

Cheerlights es un servicio web que hace que, cuando alguien envía un tuit a @cheerlights conteniendo el nombre de un color, se registre ese color como el último. En todo el mundo, muchas personas tienen proyectos de Cheerlight y usan un servicio web para solicitar el último color y establecer su iluminación con ese color. Así que, cuando alguien tuitea, las luces de todo el mundo cambian de color.

Solución

Utilice un led RGB Raspberry Squid conectado a su Raspberry Pi (Figura 15.10) y ejecute el programa de prueba llamado *cheerlights.py*, que se puede encontrar con las descargas del libro en la página web del libro.

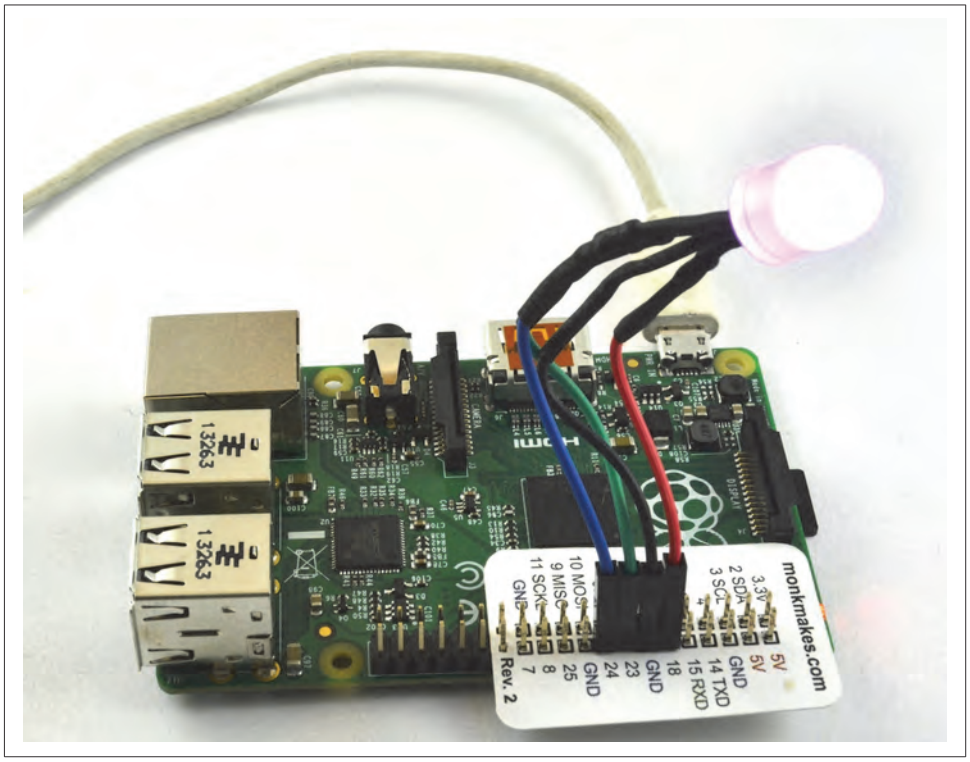


Figura 15.10. *Cheerlight.*

```
from squid import *
import urllib, time

squid = Squid(18, 23, 24)
cheerlights_url = "http://api.thingspeak.com/channels/1417/field/2/last.txt"


try:
    while True:
        try:
            cheerlights = urllib.urlopen(cheerlights_url)
            c = cheerlights.read()
            cheerlights.close()
            print(c)
            squid.set_color_rgb(c)
        except:
            print('Error')
         time.sleep(2)

finally:
    GPIO.cleanup()
```

Antes de ejecutar el programa necesitará instalar la biblioteca Squid (vea el Capítulo 9.11).

Su led debe ajustarse inmediatamente al último color. Es probable que cambie de color después de un rato cuando alguien tuitee; si no, trate de enviar un tuit como "@cheerlights red" y el color de su led y el resto de los ledes del mundo debería cambiar. Los nombres de colores válidos para Cheerlights son: *red*, *blue*, *cyan*, *white*, *oldlace*, *purple*, *magenta*, *yellow*, *orange* y *pink*.

Observaciones

El código solo envía una solicitud web a ThingSpeak, que devuelve una cadena de colores como un número hexadecimal de 6 dígitos. Esto se utiliza para ajustar el color del led.

El código `try/except` se utiliza para garantizar que el programa no se bloquee si hay una interrupción temporal de la red.

Para saber más

Cheerlights usa ThingSpeak para almacenar el último color en un canal. En el Capítulo 15.7 se utiliza un canal para registrar los datos del sensor.

Si no tiene un Squid, puede usar un RGB en una placa de pruebas (vea el Capítulo 10.11), o incluso puede adaptar el Capítulo 14.8 para controlar una tira de led entera.

15.7 Enviar los datos de los sensores a ThingSpeak

Problema

Desea registrar los datos del sensor en ThingSpeak.

Solución

Inicie sesión en ThingSpeak, y, desde el menú desplegable Channels, seleccione My Channels. Después cree un nuevo canal completando la parte superior del formulario como se muestra en la Figura 15.11.

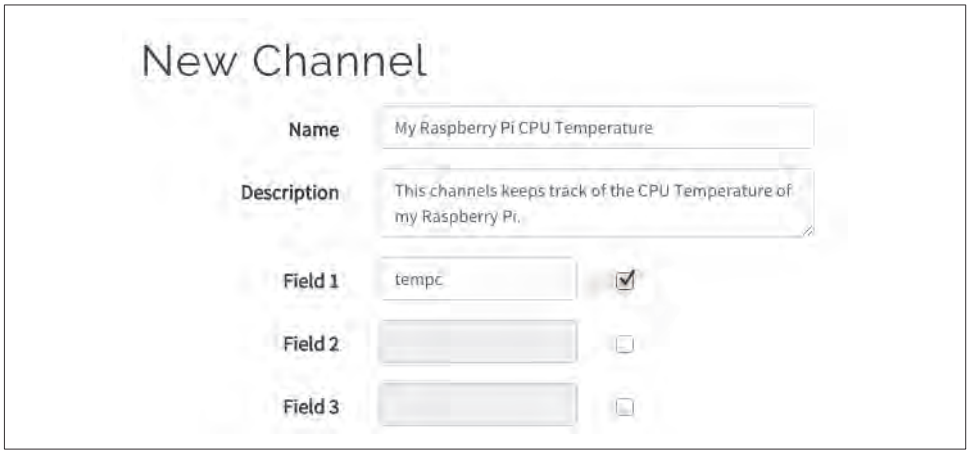


Figura 15.11. *Creación de un canal en ThingSpeak.*

El resto del formulario se puede dejar el blanco. Cuando haya terminado de editar, haga clic en Save Channel al final de la página. Haga clic en la pestaña Data Import/Export para encontrar un resumen de las solicitudes web que puede utilizar, personalizadas para el canal que acaba de crear (Figura 15-12).

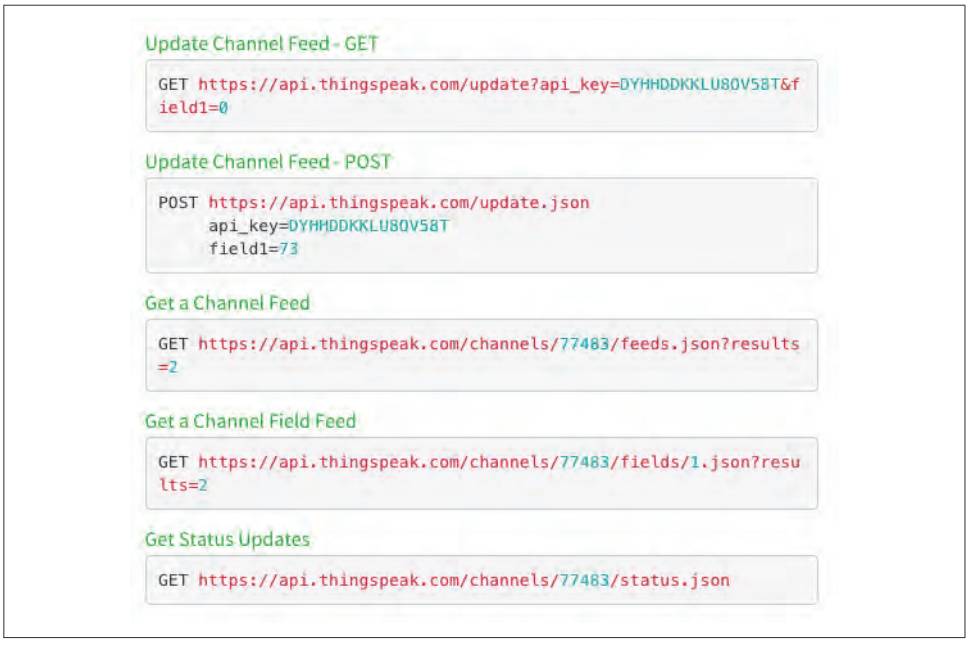


Figura 15.12. *Obtención de datos en un canal ThingSpeak.*

Para enviar datos al canal, se debe enviar una solicitud web. El programa de Python para enviar la solicitud web se puede encontrar con las descargas para el libro en la página web del libro, donde pone *thingspeak_data.py*.

```
import time, os, urllib, urllib2

PERIOD = 60 # Segundos

BASE_URL = 'https://api.thingspeak.com/update.json'
KEY = 'DYHHDDKKLU8OV58T'

def send_data(temp):
    data = urllib.urlencode({'api_key' : KEY, 'field1': temp})
    response = urllib2.urlopen(url=BASE_URL, data=data)
    print(response.read())

def cpu_temp():
    dev = os.popen('/opt/vc/bin/vcgencmd measure_temp')
    cpu_temp = dev.read()[5:-3]
    return cpu_temp

while True:
    temp = cpu_temp()
    print("CPU Temp (C): " + str(temp))
    send_data(temp)
    time.sleep(PERIOD)
```

Ejecute el programa. En la pestaña Private View de la página del canal ThingSpeak debería ver un gráfico como el que se muestra en la Figura 15.13.

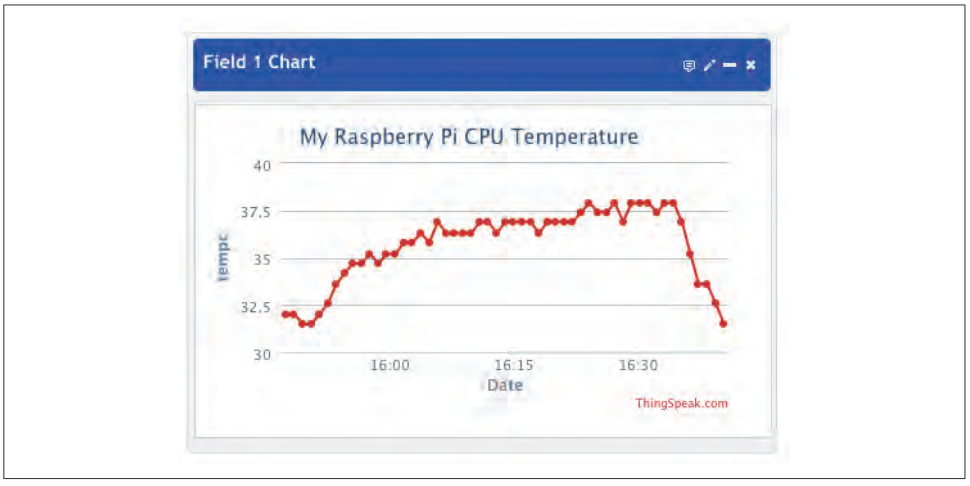


Figura 15.13. *Cartografía de los datos del sensor.*

Esto se actualizará a medida que llegue una nueva lectura cada minuto.

Observaciones

La variable `PERIOD` se utiliza para determinar el intervalo de tiempo entre cada envío de la temperatura. Este periodo es en segundos.

La función `send_data` construye la petición web, suministrando la temperatura en un parámetro llamado `field1`.

Si sus datos son de interés público –por ejemplo, lecturas ambientales precisas– es posible que desee hacer que el canal sea público para que cualquiera pueda hacer uso de él. Seguramente no sea el caso de la temperatura de la CPU de su Pi.

Para saber más

Para obtener un ejemplo de exportación de datos de los sensores a una hoja de datos vea el Capítulo 13.21. Para obtener una explicación del código que lee la temperatura de la CPU consulte el Capítulo 13.10.

15.8 Responder tuits utilizando Dweet y IFTTT

Problema

Quiere que su Raspberry Pi realice alguna acción en respuesta a una etiqueta o a una mención en un tuit.

El Capítulo 15.6 hace exactamente eso, pero de manera ineficaz porque se basa en que continuamente sondea las solicitudes web para ver si el color ha cambiado.

Solución

Un mecanismo eficiente para monitorear tuits es usar IFTTT (vea el Capítulo 15.4) para detectar tuits de interés y luego enviar una solicitud web a un servicio llamado Dweet, que puede enviar notificaciones a un programa de Python que se ejecuta en su Raspberry Pi (Figura 15.14).

Por ejemplo, puede destellar un led durante 10 segundos cada vez que haya una mención de su nombre de usuario en Twitter usando una Raspberry Squid o un led unido a una placa de pruebas.

En cuanto al hardware, este capítulo solo necesita algo de electrónica que haga algo interesante cuando el GPIO 18 esté en alto. Podría ser un canal de una Raspberry Squid (vea el Capítulo 9.11), un simple led unido a una placa de pruebas (vea el Capítulo 10.2) o, para una mayor flexibilidad, un relé (vea el Capítulo 10.7). El uso de un relé le permitiría crear un proyecto como Bubblino.

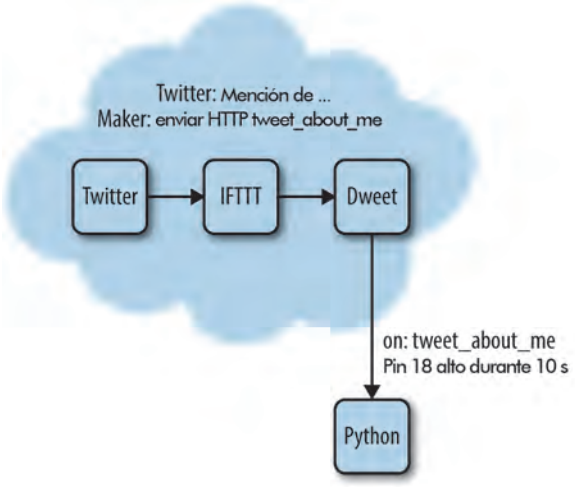


Figura 15.14. *IFTTT, Dweet y Python funcionando juntos.*

El primer paso es iniciar sesión en IFTTT (vea el Capítulo 15.4) y después crear una nueva receta. Elija un canal de acción de New Mention of You y después haga clic en Create Trigger. Para el canal de acción de la receta, seleccione Maker y luego seleccione la acción "Make a Web Request" (hacer una solicitud web) y complete los campos como se muestra en la Figura 15.15.

La URL incluye un parámetro de solicitud con el texto. Esto contendrá el cuerpo del tuit. Aunque solo se utilizará para imprimirlo en la consola, es posible que el mensaje se muestre en una pantalla LCD si se hace un proyecto más sofisticado, por lo que es útil saber cómo pasar los datos del tuit al programa de Python.

Después haga clic en Create Recipe para llevar la receta IFTTT en vivo.

El servicio web *dweet.io* funciona como Twitter para los dispositivos IoT. Cuenta con una interfaz web que le permite tanto enviar como escuchar *dweets*.

Dweet no necesita ninguna cuenta ni ninguna información de acceso para que podamos usarlo. Solo puede tener un elemento (IFTTT en este caso) para enviar un mensaje y tener otro elemento (programa de Python Raspberry Pi) para esperar notificaciones de que algo en lo que está interesado ha ocurrido. En este caso, lo que vincula los dos es `tweet_about_me`. Esto no es muy específico y, si varias personas están probando el ejemplo del libro al mismo tiempo, recibirán mensajes de otros. Para evitar esto, use algo más exclusivo (por ejemplo, añadir una cadena aleatoria de letras y número al mensaje).

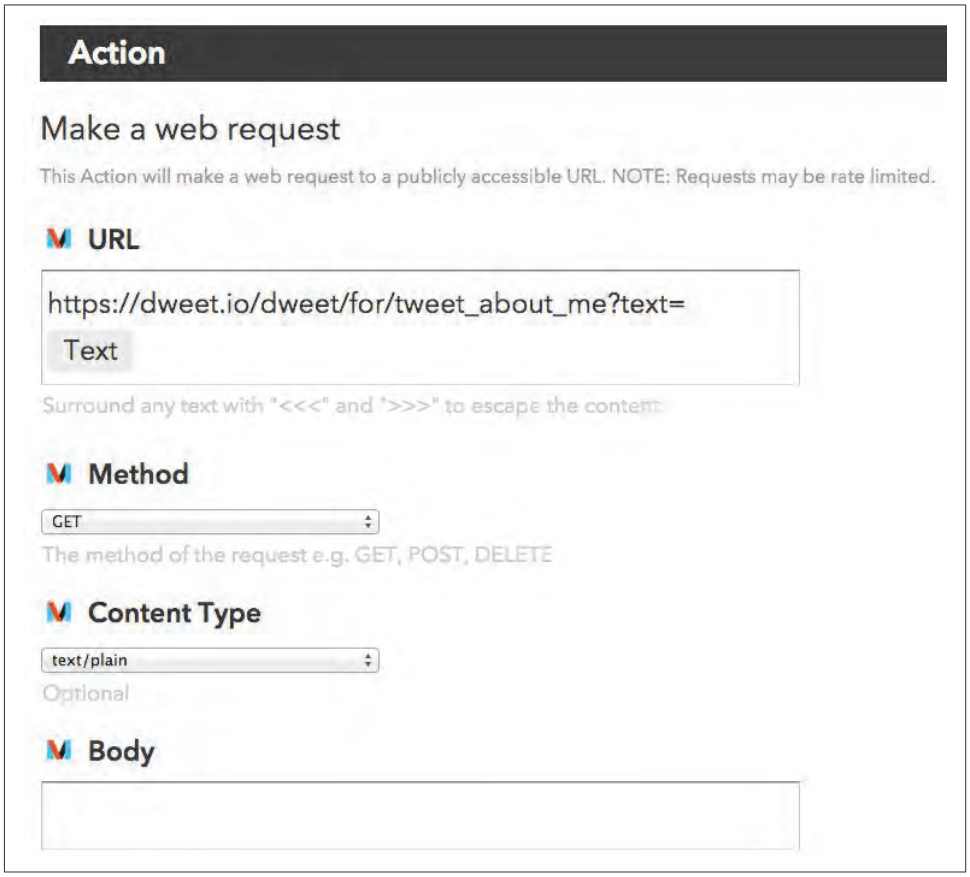


Figura 15.15. *Completar el canal de acción Send HTTP Request en IFTTT.*

Para acceder a Dweet desde su programa de Python es necesario instalar la biblioteca dweepy usando los siguientes comandos:

```
$ git clone git://github.com/paddycarey/dweepy.git
$ cd dweepy
$ sudo python setup.py install
```

El programa para este capítulo se puede encontrar con las descargas para el libro en la página web del libro, donde pone *twitter_trigger.py*.

```
import time
import dweepy
import RPi.GPIO as GPIO

KEY = 'tweet_about_me'
OUTPUT_PIN = 18
OUTPUT_DURATION = 10
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(OUTPUT_PIN, GPIO.OUT)

while True:
    try:
        for dweet in dweepy.listen_for_dweets_from(KEY):
            print('Tweet: ' +dweet['content']['text'])
            GPIO.output(OUTPUT_PIN, True)
            time.sleep(OUTPUT_DURATION)
            GPIO.output(OUTPUT_PIN, False)
    except Exception:
        pass
```

Observaciones

El programa usa el método `listen_for_dweets_from` para dejar la conexión abierta al servidor *dweet.io*, escuchando cualquier mensaje de activación desde el servidor como resultado de un dweet que llegue desde IFTTT en respuesta a un tuit. El bloque `try/except` asegura que, si hay algún corte de comunicación, el programa comenzará el proceso de escucha de nuevo.

Para saber más

Para conocer un proyecto similar con un enfoque diferente vea el Capítulo 15.6.

CAPÍTULO 16

Arduino y Raspberry Pi

16.1 Introducción

Mientras que Raspberry Pi es ideal para proyectos que necesitan una conexión web o una interfaz gráfica de usuario, las salidas GPIO de baja potencia y la falta de entradas analógicas lo ponen en desventaja respecto a las placas de microcontroladores, como Arduino (Figura 16.1). Afortunadamente, es posible tener lo mejor de ambos mundos conectando un Arduino a Raspberry Pi y permitiendo que el Arduino se interconecte con electrónica externa.

Las placas Arduino son aparentemente parecidas a las Raspberry Pi, ya que son pequeñas y esencialmente son ordenadores. Sin embargo, las placas Arduino son muy diferentes en varios aspectos:

- No tienen ninguna interconexión con ningún teclado, ratón o pantalla.
- Solo tienen 2 KB de RAM y 32 KB de flash para almacenar programas.
- Su procesador funciona a solo 16 MHz en comparación con los 700 MHz de Raspberry Pi.

Esto podría llevarle a preguntarse para qué usaría una placa aparentemente pobre en lugar de una Raspberry Pi directamente.

La respuesta es que las placas Arduino, las más comunes son Arduino Uno, son mejores que Raspberry Pi en la interrelación con la electrónica externa de varias maneras. Por ejemplo, las tarjetas Arduino tienen:

- 14 entradas/salidas digitales, como los pines GPIO, pero cada pin puede proporcionar hasta 40 mA en comparación con los 3 mA de Raspberry Pi. Esto les permite alimentar más dispositivos sin necesidad de electrónica extra.
- 6 entradas analógicas. Hace que la conexión de sensores analógicos sea mucho más fácil (vea el Capítulo 16.8).

- 6 salidas PWM. Estas salidas son controladas por hardware y producen una señal PWM mucho más precisa de lo que se puede lograr con Raspberry Pi, haciéndolas mucho mejores para controlar servomotores.
- Una amplia gama de placas de expansión para todo, desde el control del motor hasta las pantallas LCD de diferentes tipos.

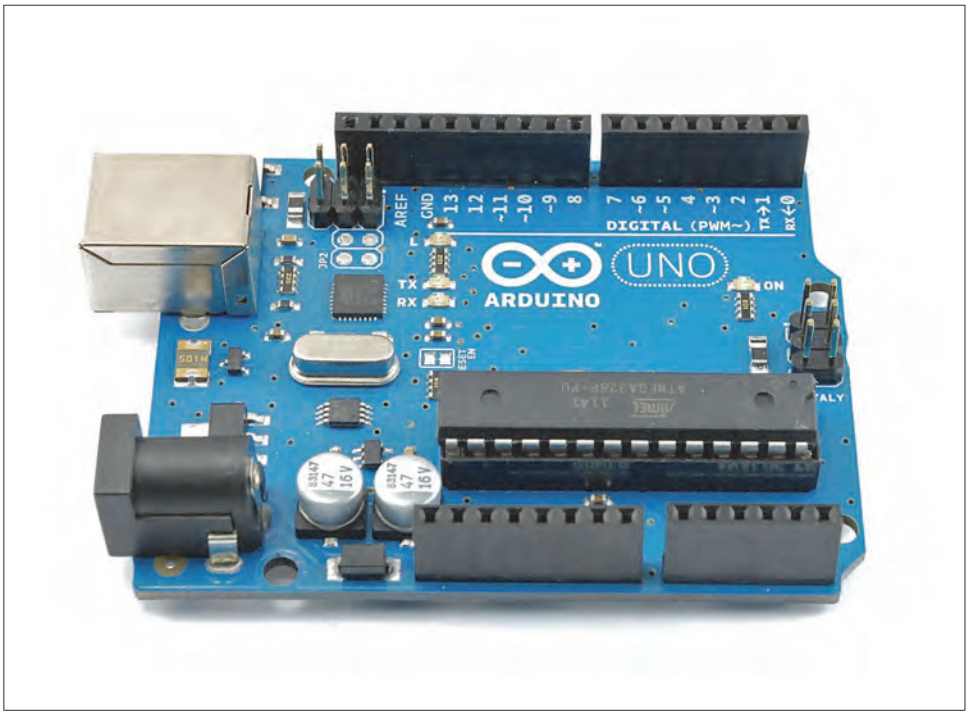


Figura 16.1. *Una placa Arduino Uno.*

Es una buena combinación el uso de Raspberry Pi con Arduino para manejar cosas de bajo nivel, jugando con las capacidades de ambas placas. Esto se puede mejorar con una tarjeta de interfaz, que tiene un conector GPIO e incluye también hardware compatible con Arduino, como aLaMode (Capítulo 16.14).

16.2 Programar un Arduino desde Raspberry Pi

Asegúrese de ver el vídeo que acompaña a este capítulo en *http://razzpisampler.oreilly.com*.

Problema

Desea ejecutar el IDE de Arduino en una Raspberry Pi para que pueda escribir y cargar programas en Arduino.

Solución

El IDE de Arduino está disponible para Raspberry Pi. Es un poco lento, pero se puede usar. Utilice estos comandos para instalar el IDE de Arduino:

```
$ sudo apt-get update
$ sudo apt-get install arduino
```

Después de la instalación, encontrará la opción Board en el menú Tools (Figura 16.2).

Para programar Arduino con el IDE, conéctelo a Raspberry Pi a través de su cable USB. La forma más fácil de configurar su Pi para usar el IDE de Arduino es ejecutar un script creado por Kevin Osborn, que configura los puertos serie y los perfiles de Arduino necesarios para que todo funcione. Tiene la ventaja de que también configura la placa aLaMode (Capítulo 16.14).

Para descargar y ejecutar este script siga estos pasos:

```
$ https://github.com/wyolum/alamode/raw/master/bundles/alamode-setup.tar.gz
$ tar -xvzf alamode-setup.tar.gz
$ cd alamode-setup
$ sudo ./setup
```

Después, conecte su Arduino a su Raspberry Pi. Desde el menú Tools seleccione Board y establezca el tipo de placa en Arduino Uno. Luego, en la opción Serial Port, seleccione */dev/ttyACM0*. Para cargar un programa de prueba que haga que el led en el Arduino parpadee, seleccione el menú File, haga clic en "Examples, Basic," y finalmente haga clic en Blink. Haga clic en la flecha derecha de la barra de herramientas para empezar el proceso de compilación y subida. Si todo va bien, debería ver un mensaje diciendo "Done Uploading" en el área de estado, en la parte inferior de la ventana de IDE.

Si detecta que el dispositivo ttyACM0 no aparece, aunque su Arduino esté enchufado, intente reiniciar el IDE de Arduino. Si esto no funciona, puede que tenga que reiniciar Raspberry Pi. Deje el Arduino conectado mientras reinicia y reanuda el IDE de Arduino.

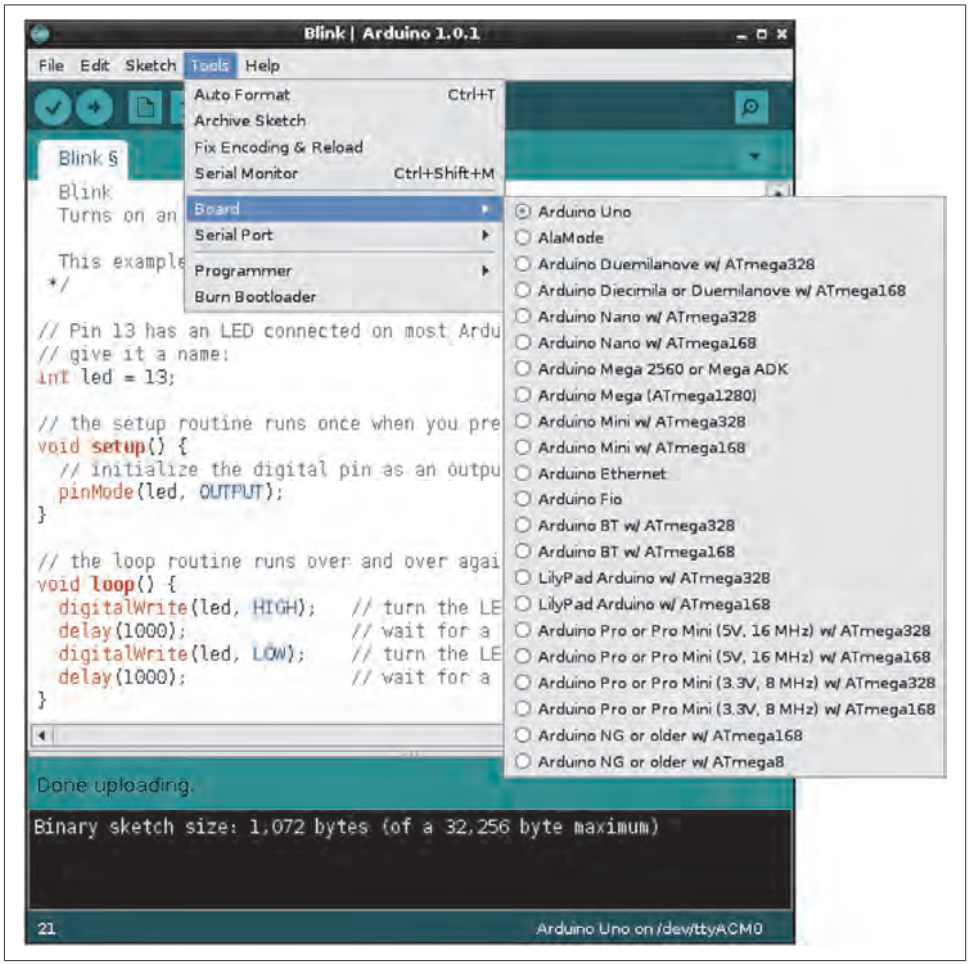


Figura 16.2. *El IDE de Arduino ejecutándose en Raspberry Pi.*

Observaciones

Para sacar el máximo provecho al uso de Arduino con Raspberry Pi, debería aprender un poco de programación de Arduino. El libro *Taller de Arduino: un enfoque práctico para principiantes* (Marcombo) es muy útil.

Sin embargo, puede usar Arduino sin necesidad de escribir ningún código, usando un proyecto llamado PyFirmata. En el Capítulo 16.4 se explica cómo usar PyFirmata.

Para saber más

La parte de la configuración del IDE de Arduino es del blog de Bald Wisdom.

16.3 Comunicarse con el Arduino usando el monitor serial

Problema

Quiere mostrar mensajes enviados desde un Arduino.

Solución

El IDE de Arduino incluye una función llamada *monitor serial*, que le permite enviar mensajes de texto al Arduino y ver los mensajes del Arduino a través de un cable USB.

Para probar esto, primero necesita escribir un programa de Arduino muy corto (los programas se llaman *bocetos* en el mundo de Arduino). Este boceto simplemente repetirá un mensaje, enviándolo cada segundo.

El boceto de Arduino se describe aquí. Como con todos los ejemplos de programas de este libro, también puede descargar este programa desde la web del libro, en la carpeta *ArduinoHello*.

Utilice File→New para crear un nuevo boceto y pegue el siguiente texto antes de cargarlo en el Arduino.

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

void loop()
{
  Serial.println("Hello Raspberry Pi");
  delay(1000);
}
```

Tan pronto como se cargue el boceto en el Arduino, comenzará a enviar el mensaje "Hello Raspberry Pi" en serie. No lo verá hasta que no abra el monitor serie (Figura 16.3) haciendo clic en el icono que parece una lupa, a la derecha de la barra de herramientas.

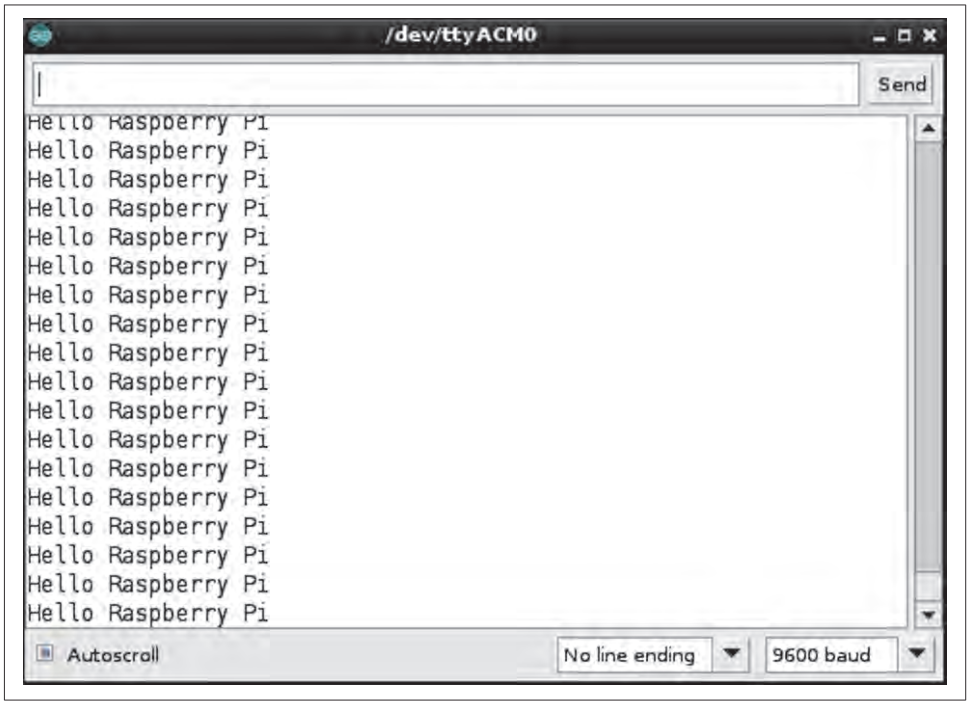


Figura 16.3. *Visualización de mensajes con el monitor serie.*

El monitor serie tiene una lista desplegable en la esquina inferior derecha, donde puede seleccionar la velocidad en baudios (velocidad de comunicación). Si aún no se ha establecido en 9600, cámbielo a ese valor.

Observaciones

En algunos capítulos posteriores (Capítulos 16.11 y 16.12), escribirá su propio código personalizado para comunicarse con los programas de Python que se ejecutan en Raspberry Pi para que no necesite tener el IDE de Arduino en ejecución.

Un enfoque más genérico es utilizar PyFirmata, que evita la necesidad de cualquier programación en Raspberry Pi. Vea el Capítulo 16.4 para conocer los detalles.

Para saber más

Arduino es muy fácil de aprender; aquí tiene algunos enlaces a libros y recursos en línea para empezar:

- Taller de arduino: un enfoque práctico para principiantes (Marcombo) de German Tojeiro
- La guía oficial de inicio de Arduino

- Lecciones de Adafruit Arduino
- Arduino: aplicaciones en robótica, mecatrónica e ingenierías (Marcombo) de Fernando Reyes y Jaime Cid

16.4 Configurar PyFirmata para controlar un Arduino desde Raspberry Pi

Problema

Desea usar Arduino como una tarjeta de interfaz para su Raspberry Pi.

Solución

Conecte el Arduino a la toma USB de Raspberry Pi para que el ordenador pueda comunicarse y enviar corriente a Arduino.

Luego, instale el boceto Firmata en el Arduino y el PyFirmata en su Raspberry Pi. Esto implica instalar el IDE de Arduino, así que, si aún no lo ha hecho, siga el Capítulo 16.2.

El IDE de Arduino incluye Firmata, por lo que todo lo que tiene que hacer para instalar Firmata en su placa Arduino es cargar el boceto. Encontrará el boceto en File→Examples→Firmata→StandardFirmata.

Una vez instalado Firmata, el Arduino espera la comunicación desde Raspberry Pi.

Después necesitará instalar PyFirmata, la otra mitad del enlace. Esto requiere el uso de la biblioteca PySerial, así que siga el Capítulo 9.7 para instalarla.

A continuación, podrá descargar e instalar PyFirmata con estos comandos:

```
$ git clone https://github.com/tino/pyFirmata.git
$ cd pyFirmata
$ sudo python setup.py install
```

Puede probar la biblioteca PyFirmata desde la consola de Python. Introduzca los siguientes comandos para encender el led incorporado en el pin 13 de Arduino (marcado con una *L*) y después apáguelo de nuevo.

```
$ sudo python
Python 2.7.3 (default, Jan 13 2013, 11:20:46)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import pyfirmata
>>> board = pyfirmata.Arduino('/dev/ttyACM0')
>>> pin13 = board.get_pin('d:13:o')
>>> pin13.write(1)
```

```
>>> pin13.write(0)
>>> board.exit()
```

Observaciones

El código precedente importa primero la biblioteca PyFirmata y después hace una instancia de `Arduino` llamada `board`, utilizando la interfaz USB (`/dev/ttyACM0`) como parámetro. A continuación, puede obtener una referencia a uno de los pines Arduino (en este caso, 13) y configurarlo para que sea una salida digital. La `d` es por digital, 13 es el número de pin y la `o` es por *output*.

Para configurar el pin de salida alto use `write(1)`, y para establecerlo bajo use `write(0)`. También puede utilizar `True` y `False` en lugar de 1 y 0.

La Figura 16.4 muestra un Arduino con las filas de conexiones a ambos lados de la placa.

Figura 16.4. *Pines I/O en un Arduino Uno.*

Los pines en la parte superior marcados de 0 a 13 pueden utilizarse como entradas o salidas digitales. Algunos de estos pines también se utilizan para otras cosas. Los pines 0 y 1 se utilizan como interfaz serie; se usan cuando se utiliza el puerto USB y el pin 13 está conectado al led de la placa marcado con una *L*. Los pines I/O digitales 3, 5, 6, 9, 10 y 11 tienen el símbolo ~ al lado, que indica que pueden usarse para la salida PWM (vea el Capítulo 10.4).

En el otro lado de la placa hay un conjunto de conectores que suministran corriente a 5 V y 3,3 V, y seis entradas analógicas marcadas de A0 a A5.

Un Arduino Uno por sí solo utiliza unos 50 mA. La propia Raspberry Pi usa aproximadamente 10 veces más, lo que hace que pueda alimentar perfectamente al Arduino desde la conexión USB de Raspberry Pi. Sin embargo, si empieza a conectar al Arduino una gran cantidad de aparatos electrónicos externos, y aumenta el consumo de corriente, es posible que quiera alimentar el Arduino desde su propio adaptador usando el conector de CC. Este aceptará de 7 V a 12 V de corriente continua.

La única desventaja del uso de Firmata es que, debido a que todas las instrucciones tienen que venir de Raspberry Pi, no hace mucho uso de la capacidad de Arduino para funcionar de manera independiente. Para proyectos avanzados, probablemente acabará escribiendo su propio código de Arduino que recibirá instrucciones de Raspberry Pi y/o enviará mensajes a Raspberry Pi mientras lleva a cabo otras tareas.

Para saber más

Los siguientes capítulos consideran el uso de la gama completa de las características de los pines de Arduino con PyFirmata: 16.5, 16.6, 16.7, 16.8 y 16.9.

Puede encontrar la documentación oficial de pyFirmata en la página web GitHub de pyFirmata.

16.5 Escritura de salidas digitales en un Arduino desde Raspberry Pi

Problema

Desea controlar las salidas digitales de Arduino desde Python en Raspberry Pi.

Solución

En el Capítulo 16.4 encendió el led incorporado (etiquetado con una *L*) en la placa Arduino. Se basará en esto para conectar un led externo y escribirá un programa corto de Python para que parpadee.

Para hacer esto necesitará:

- Arduino Uno (vea “Módulos” en la página 476)
- Placa de pruebas y cables puente (vea “Equipos para prototipos”, página 474)
- Resistor 270 Ω (vea “Resistores y condensadores” en la página 474)
- Led (vea “Optoelectrónica” en la página 476)

Como alternativa al uso de una placa de pruebas y un led, puede conectar un led Squid RGB (Capítulo 9.11).

Conecte la placa de pruebas fijando los componentes al Arduino como se muestra en la Figura 16.5.

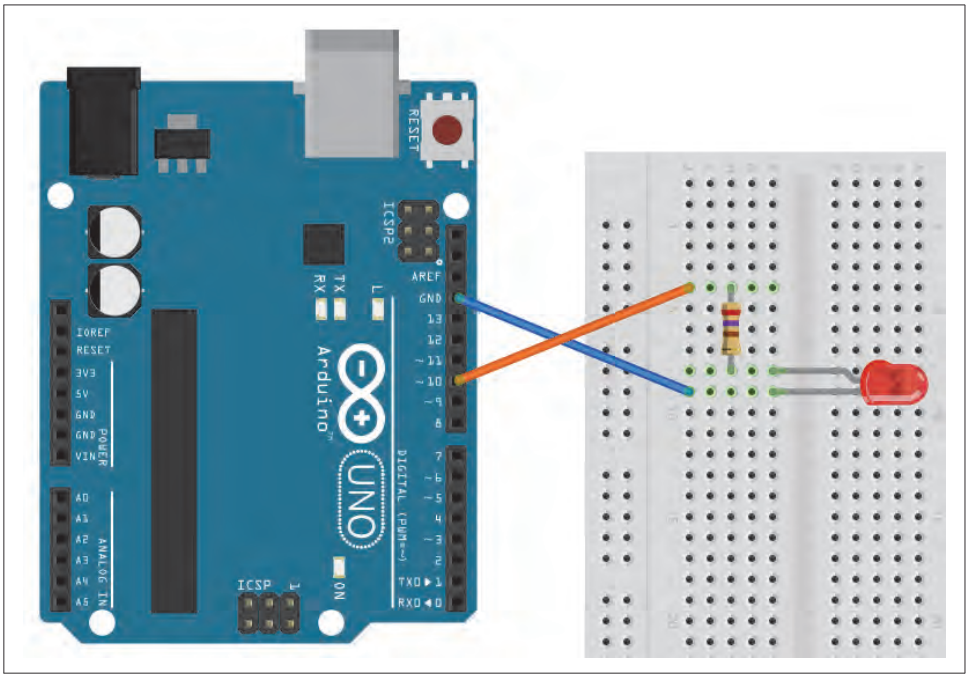


Figura 16.5. *Diagrama de cableado para Arduino y led.*

Si aún no lo ha hecho, siga el Capítulo 16.4 para configurar PyFirmata.

La siguiente secuencia de comandos de Python hace que el led parpadee a una velocidad de aproximadamente 1 Hz. Abra un editor (nano o IDLE) y pegue el siguiente código. Como todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *ardu_flash.py*.

```
import pyfirmata
import time

board = pyfirmata.Arduino('/dev/ttyACM0')
led_pin = board.get_pin('d:10:o')

while True:
    led_pin.write(1)
    time.sleep(0.5)
    led_pin.write(0)
    time.sleep(0.5)
```

Observaciones

Es muy similar a conectar un led a una Raspberry Pi (Capítulo 10.2). Sin embargo, tenga en cuenta que, dado que las salidas de Arduino pueden suministrar mucha más corriente que las salidas de Raspberry, puede utilizar un resistor de menor valor y hacer que el led sea un poco más brillante. Las salidas de Arduino también son de 5 V en lugar de 3,3 V.

Si desea que la interfaz de usuario controle el led como lo hizo en el Capítulo 10.9 (se muestra en la Figura 16.6), modifique el código. Puede encontrar el programa modificado, llamado *ardu_gui_switch.py*, en la web del libro. Recuerde que esto no funcionará desde la línea de comandos SSH. Necesita tener acceso al entorno gráfico de Raspberry Pi para poder ver la interfaz de usuario.

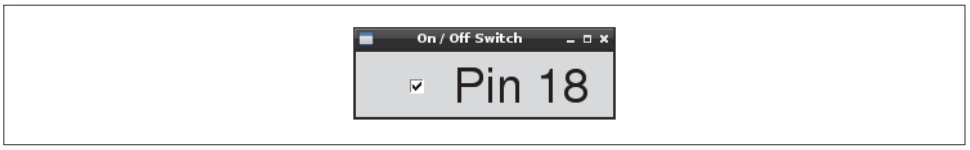


Figura 16.6. *Interfaz de usuario para activar y desactivar cosas.*

Para saber más

Vea el Capítulo 10.2 para controlar un led directamente desde Raspberry Pi.

16.6 Utilizar PyFirmata con TTL en serie

Problema

Desea utilizar PyFirmata sobre la conexión en serie (RXD y TXD en el conector GPIO) en lugar de la conexión USB.

Solución

Utilice un convertidor de nivel para conectar el pin RXD de Raspberry Pi al pin Tx de Arduino, y el pin TXD de Raspberry Pi al pin Rx de Arduino.

Para hacer eso necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistores de 270 Ω y 470 Ω (vea "Resistores y condensadores" en la página 474) o un convertidor de nivel bidireccional de cuatro vías (vea "Módulos" en la página 476)

Si utiliza el módulo convertidor, conecte la placa de pruebas como se muestra en la Figura 16.7.

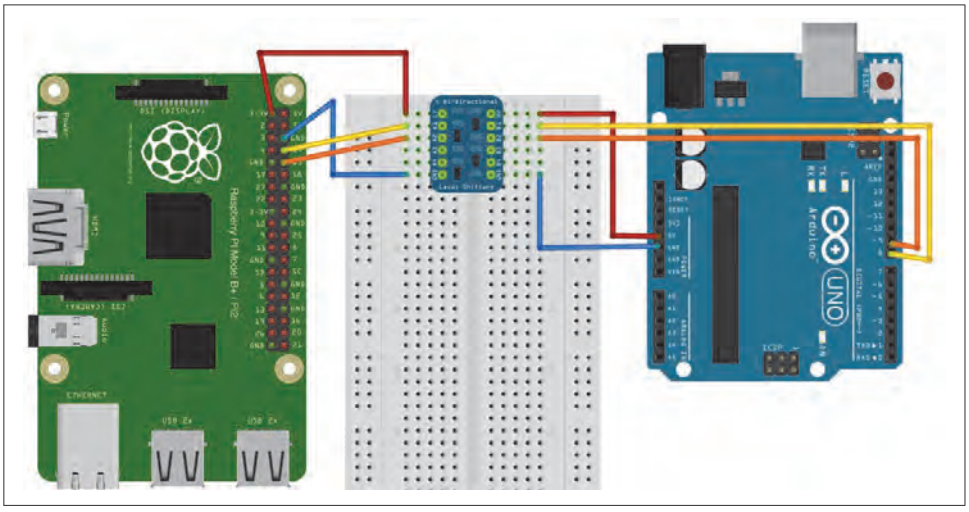


Figura 16.7. *Diagrama de cableado utilizando un módulo convertidor de nivel para una comunicación en serie con un Arduino*

Si por el contrario está utilizando los dos resistores, conecte la placa de pruebas como se muestra en la Figura 16.8.

La entrada de Rx de Arduino va bien con solo 3,3 V desde el pin TXD de Raspberry Pi; sin embargo, los 5 V que vienen del pin Tx de Arduino deben disminuir a los 3 V que espera Raspberry Pi.

Tendrá que configurar PyFirmata; vea el Capítulo 16.4. La parte de Arduino del proyecto permanece exactamente igual que en el Capítulo 16.5, donde se usa USB en lugar de la conexión en serie. Hay un cambio que debe hacer en el programa de Python que se ejecuta en Raspberry Pi; cambie el nombre del dispositivo de */dev/ttyACM0* a */dev/ttyAMA0*, ya que el puerto serie tiene un nombre de dispositivo diferente de la interfaz USB.

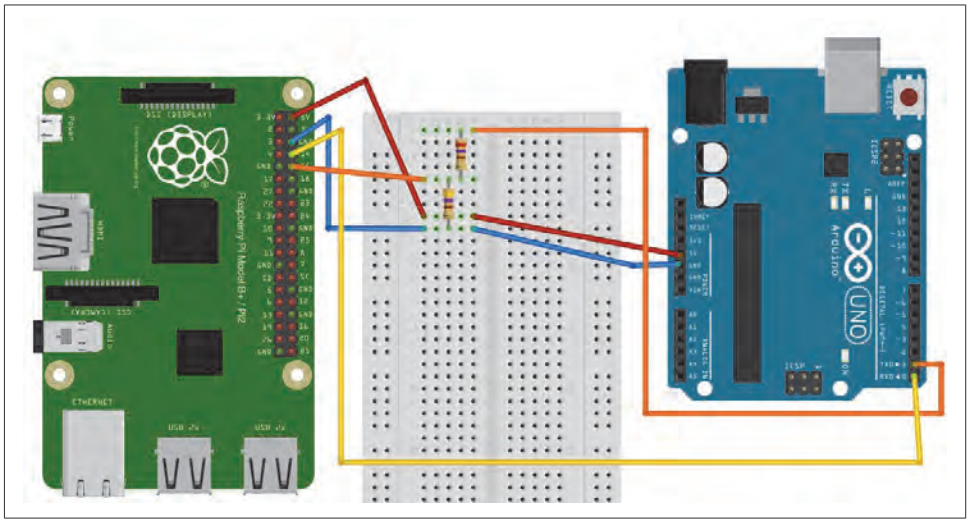


Figura 16.8. *Diagrama de cableado, utilizando un par de resistores para la comunicación en serie con un Arduino.*

La siguiente secuencia de comandos de Python hace que el led parpadee a una velocidad aproximadamente de 1 Hz. Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *ardu_flash_ser.py*.

```
import pyfirmata
import time

board = pyfirmata.Arduino('/dev/ttyAMA0')
led_pin = board.get_pin('d:13:o')

while True:
    led_pin.write(1)
    time.sleep(0.5)
    led_pin.write(0)
    time.sleep(0.5)
```

Observaciones

La conversión de nivel es necesaria porque las conexiones serie de Raspberry Pi, RXD y TXD funcionan a 3,3 V, mientras que el Arduino Uno funciona a 5 V. Mientras que al Arduino de 5 V le va bien usar una señal de 3,3 V, al revés, una señal de 5 V conectada al pin RXD de 3,3 V dañará su Raspberry Pi.

Para saber más

También puede adaptar fácilmente los otros ejemplos que utilizan PyFirmata (Capítulos de 16.6 a 16.9) para usar una conexión en serie en lugar de una conexión USB, simplemente cambiando el nombre del dispositivo en el programa de Python.

16.7 Lectura de entradas digitales de Arduino usando PyFirmata

Problema

Desea leer las entradas digitales de Arduino desde Python en Raspberry Pi.

Solución

Utilice PyFirmata para leer una entrada digital en Arduino.

Para hacer esto necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor 1 kΩ (vea "Resistores y condensadores" en la página 474)
- Pulsador táctil (vea "Varios" en la página 477)

Conecte la placa de pruebas, fijando los componentes al Arduino como se muestra en la Figura 16.9.

Si aún no lo ha hecho, siga el Capítulo 16.4 para configurar PyFirmata.

La siguiente secuencia de comandos de Python imprime un mensaje cada vez que le da al pulsador. Es muy similar al programa en *switch.py*. Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código desde la página web del libro, donde pone *ardu_switch.py*.

```
import pyfirmata
import time

board = pyfirmata.Arduino('/dev/ttyACM0')
switch_pin = board.get_pin('d:4:i')
it = pyfirmata.util.Iterator(board)
it.start()
switch_pin.enable_reporting()

while True:
    input_state = switch_pin.read()
    if input_state == False:
```

```
        print('Button Pressed')
        time.sleep(0.2)
```

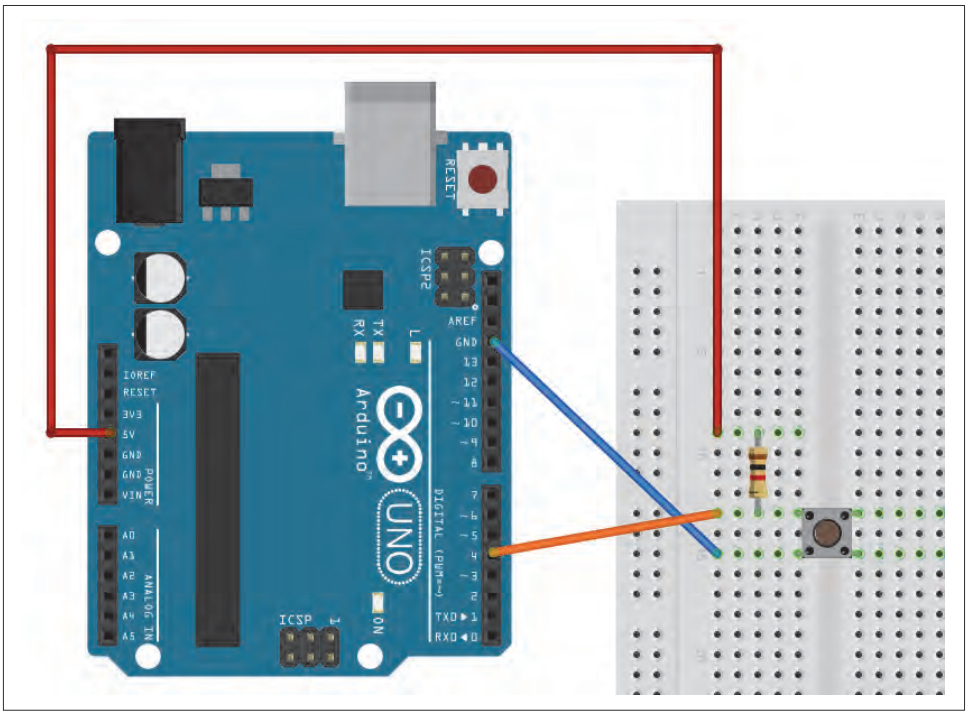


Figura 16.9. *Diagrama de cableado para Arduino y un pulsador.*

Cuando lo ejecute no sucederá nada durante un segundo o dos, hasta que el boceto de Firmata no comience y establezca comunicación con Raspberry Pi. Pero, una vez iniciado, cada vez que pulse el botón aparecerá un mensaje:

```
$ sudo python ardu_switch.py
Button Pressed
Button Pressed
Button Pressed
```

Observaciones

PyFirmata utiliza el concepto de un `Iterator` para supervisar el pin de entrada de Arduino. Las razones de esto están vinculadas a la implementación de Firmata. Esto significa que no puede simplemente *leer* el valor del pin de entrada de Arduino que quiera; para ello deberá crear un subproceso `Iterator` independiente que gestione la lectura del pulsador usando los comandos:

```
it = pyfirmata.util.Iterator(board)
it.start()
```

A continuación, también deberá habilitar el informe del pin que desee mediante el comando:

```
switch_pin.enable_reporting()
```

Un efecto secundario de este mecanismo es que, cuando presione Ctrl-C para salir del programa, no saldrá correctamente. La manera de parar el subproceso `Iterator` es abrir otra ventana de terminal o una sesión SSH (Capítulo 3.27).

Si el único proceso de Python en ejecución es este programa, puede pararlo con este comando:

```
$ sudo killall python
```

Desconectar el Arduino de Raspberry Pi romperá el enlace de comunicación y también hará que salga del programa de Python.

Para saber más

Esto es muy similar a conectar un interruptor directamente a Raspberry Pi (Capítulo 12.2). Si solo tiene un interruptor, no existe ningún beneficio en utilizar un Arduino.

16.8 Lectura de entradas analógicas de Arduino usando PyFirmata

Problema

Desea leer las entradas analógicas de Arduino desde Python en Raspberry Pi.

Solución

Utilice PyFirmata para leer una entrada analógica en Arduino.

Para hacer esto necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Potenciómetro de 10 kΩ (vea "Resistores y condensadores" en la página 474)

Conecte la placa de pruebas fijando los componentes al Arduino como se muestra en la Figura 16.10.

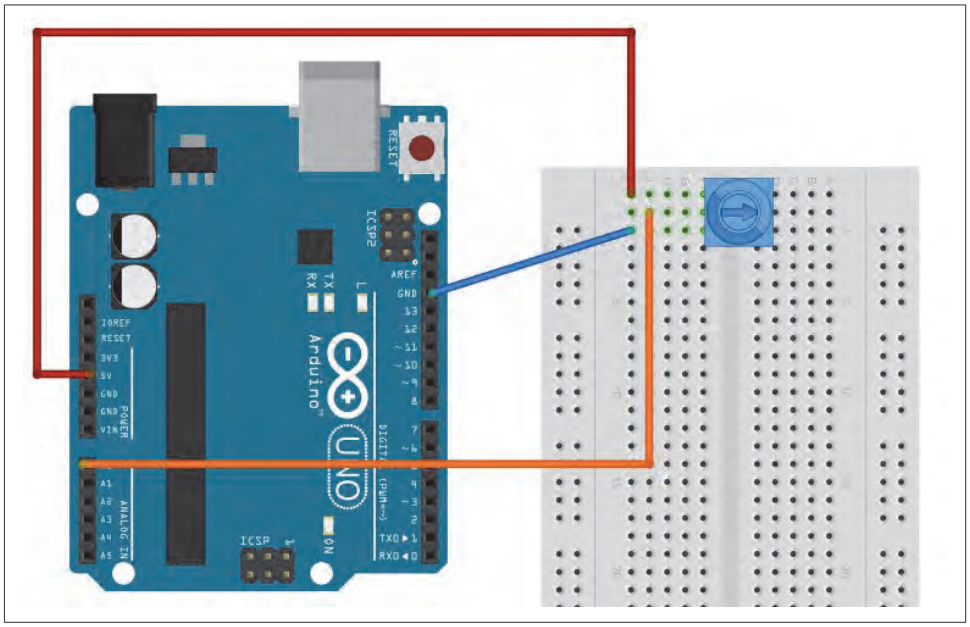


Figura 16.10. *Diagrama de cableado de Arduino y potenciómetro.*

Si aún no lo ha hecho, siga el Capítulo 16.4 para configurar PyFirmata.

La siguiente secuencia de comandos de Python (*ardu_adc.py*) mostrará tanto la lectura sin procesar de la entrada analógica como la tensión. Es muy similar al programa en *adc_test.py*.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *ardu_adc.py*.

```
import pyfirmata
import time

board = pyfirmata.Arduino('/dev/ttyACM0')
analog_pin = board.get_pin('a:0:i')
it = pyfirmata.util.Iterator(board)
it.start()
analog_pin.enable_reporting()

while True:
    reading = analog_pin.read()
    if reading != None:
        voltage = reading * 5.0
        print("Reading=%f\tVoltage=%f" % (reading, voltage))
        time.sleep(1)
```

La lectura analógica será un valor entre 0,0 y 1,0:

```
$ sudo python ardu_adc.py
Reading=0.000000    Voltage=0.000000
Reading=0.165200    Voltage=0.826000
Reading=0.784000    Voltage=3.920000
Reading=1.000000    Voltage=5.000000
```

Observaciones

El programa es muy similar al del Capítulo 16.7. Se debe utilizar un `Iterator`, y se aplican las mismas soluciones para detener el programa.

Se necesita un enunciado `if`, porque si se realiza el primer `read` antes de que se realice la lectura real desde la entrada analógica, `read` devolverá `None` en lugar de un número. El enunciado `if` hará que el programa ignore cualquier lectura nula.

Para saber más

Para utilizar entradas digitales vea el Capítulo 16.7.

16.9 Salidas analógicas (PWM) con PyFirmata

Problema

Desea controlar el brillo de un led usando PWM a través de un Arduino.

Solución

Utilice PyFirmata para enviar comandos a un Arduino y generar una señal PWM en una de sus salidas.

Para hacer esto necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 270 Ω (vea "Resistores y condensadores" en la página 474)
- Led (vea "Optoelectrónica", en la página 476)

Conecte la placa de pruebas, fijando los componentes al Arduino como se muestra en la Figura 16.11.

Si aún no lo ha hecho, siga el Capítulo 16.4 para configurar PyFirmata.

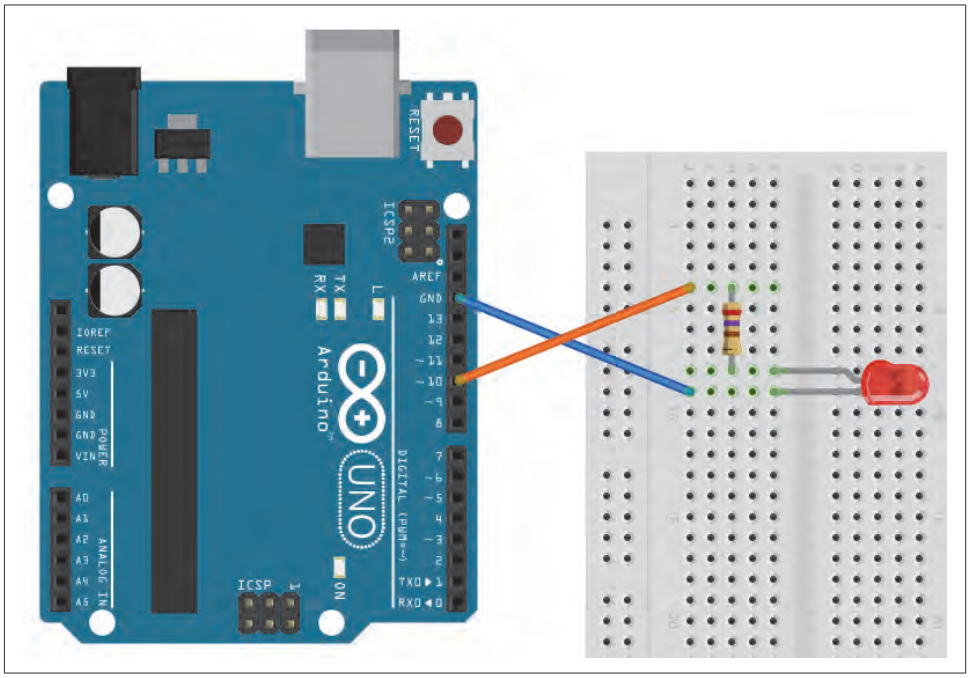


Figura 16.11. *Diagrama de cableado para el control de PWM de Arduino de un led.*

La siguiente secuencia de comandos de Python (*ardu_pwm.py*) le pedirá que introduzca un valor para la potencia de PWM y, después, ajustará el brillo del led en consecuencia. Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa desde la sección de código de la página web del libro, donde pone*ardu_pwm.py*.

```
import pyfirmata

board = pyfirmata.Arduino('/dev/ttyACM0')
led_pin = board.get_pin('d:10:p')

while True:
    duty_s = raw_input("Enter Brightness (0 to 100):")
    duty = int(duty_s)
    led_pin.write(duty / 100.0)
```

Si se introduce el valor 100, el led debería estar con el brillo al máximo. El brillo disminuye a medida que disminuye el número:

```
$ sudo python ardu_pwm.py
Enter Brightness (0 to 100):100
Enter Brightness (0 to 100):50
Enter Brightness (0 to 100):10
Enter Brightness (0 to 100):5
Enter Brightness (0 to 100):
```

Observaciones

El boceto para esto es en realidad muy sencillo. Defina la salida como salida PWM usando el comando:

```
led_pin = board.get_pin('d:10:p')
```

La *p* es por PWM. Pero recuerde, esto solo funciona en los pines de Arduino marcados con el símbolo ~.

También podemos modificar el control deslizante (Figura 16.12) para que funcione a través de PyFirmata. Puede descargar este boceto como *ardu_gui_slider*.

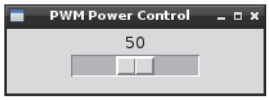


Figura 16.12. *Interfaz de usuario para controlar el brillo.*

Para saber más

Aunque un Arduino puede dar 40 mA a una salida, aproximadamente 10 veces la corriente disponible en un pin GPIO de Raspberry, no es suficiente para accionar directamente un motor o un módulo led de alta potencia. Para ello, necesitará utilizar el circuito descrito en el Capítulo 11.5, modificado para utilizar un pin de salida Arduino en vez de un pin GPIO de Raspberry.

16.10 Controlar un servo con PyFirmata

Problema

Desea controlar la posición de un servomotor usando un Arduino.

Solución

Utilice PyFirmata para enviar comandos a un Arduino y generar los pulsos necesarios para controlar la posición de un servomotor.

Para hacer eso necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Placa de pruebas y cables puente (vea "Equipos para prototipos", página 474)
- Resistor de 1 kΩ (vea "Resistores y condensadores" en la página 474)
- Led (vea "Circuitos integrados" en la página 475)

Conecte la placa de pruebas como se muestra en la Figura 16.13.

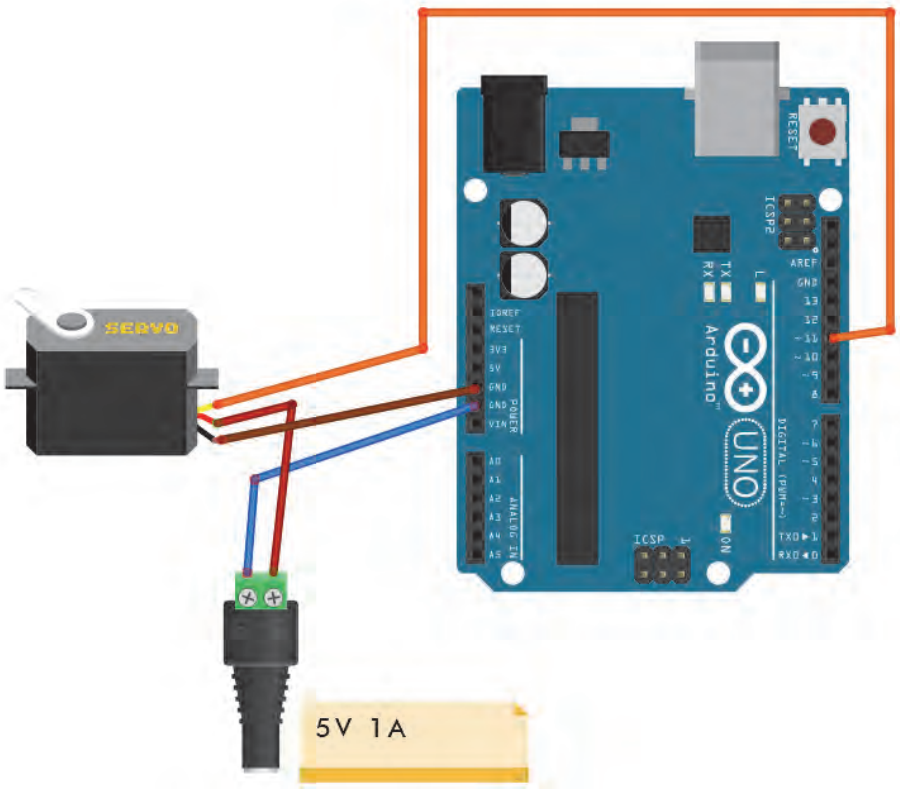


Figura 16.13. *Diagrama de cableado para que Arduino controle un servomotor.*

Si aún no lo ha hecho, siga el Capítulo 16.4 para configurar PyFirmata.

La siguiente secuencia de comandos de Python (*ardu_servo.py*) le pedirá que introduzca un valor para el ángulo del servo y después ajustará el brazo del servomotor en consecuencia.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa desde la sección de código de la página web del libro, donde pone *ardu_servo.py*.

```
import pyfirmata

board = pyfirmata.Arduino('/dev/ttyACM0')
servo_pin = board.get_pin('d:11:s')

while True:
    angle_s = raw_input("Enter Angle (0 to 180):")
```

```
        angle = int(angle_s)
        servo_pin.write(angle)
```

Si introduce el valor 0, el servo se colocará en un extremo de su recorrido. Si lo cambia a 180 lo enviaría al otro extremo, y, si introduce 90, lo colocará en el medio:

```
$ sudo python ardu_servo.py
Enter Angle (0 to 180):0
Enter Angle (0 to 180):180
Enter Angle (0 to 180):90
```

Observaciones

El boceto de esto es en realidad muy sencillo. Defina la salida como una salida servo usando el comando:

```
led_pin = board.get_pin('d:11:s')
```

La `s` es por servo. Se puede utilizar en cualquiera de los pines digitales de Arduino.

Si ha llevado a cabo el Capítulo 11.2, se dará cuenta de que, en comparación, el servo no tiembla cuando se utiliza con un Arduino de esta manera.

Para saber más

Para saber más sobre Raspberry Pi y el uso de servomotores, consulte los Capítulos 11.2, 11.3 y 11.4.

16.11 Comunicación personalizada con un Arduino sobre TTL en serie

Problema

Desea intercambiar datos con un Arduino usando las interfaces serie de ambos dispositivos sin utilizar PyFirmata.

Solución

Utilice un convertidor de nivel o un par de resistores para conectar el pin RXD de Raspberry Pi al pin Tx de Arduino, y el pin TXD de Raspberry Pi al pin Rx de Arduino.

Para hacer esto necesitará:

- Arduino Uno (vea “Módulos” en la página 476)
- Placa de pruebas y cables puente (vea “Equipos para prototipos”, página 474)

- Resistores de 270 Ω y 470 Ω (vea “Resistores y condensadores”, página 474) o un conversor bidireccional de cuatro vías (vea “Módulos”, página 476)
- Potenciómetro de 10 kΩ (vea “Resistores y condensadores”, página 474)

Si está utilizando el módulo conversor de nivel, conecte la placa de pruebas como se muestra en la Figura 16.14.

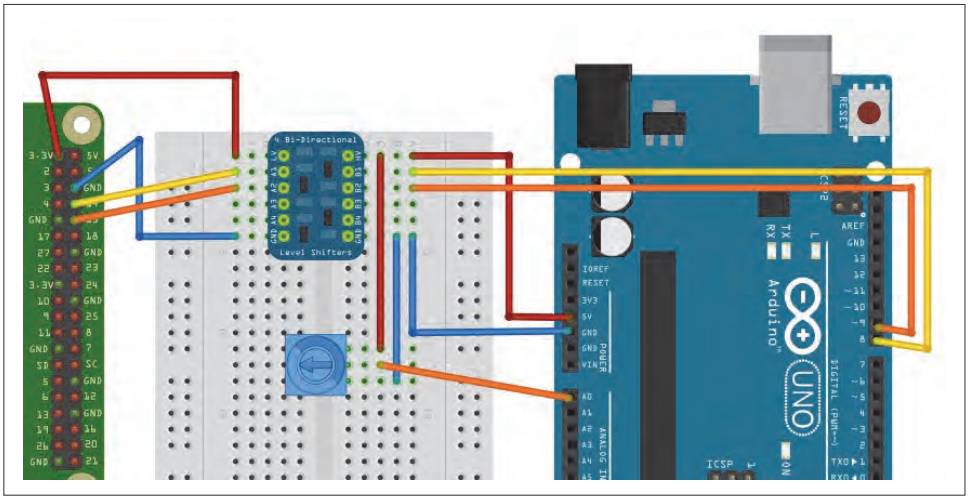


Figura 16.14. *Diagrama de cableado para una comunicación en serie con un Arduino.*

Si, por el contrario, está utilizando los dos resistores, conecte la placa de pruebas como se muestra en la Figura 16.15.

La entrada Rx de Arduino va bien con solo 3,3 V desde el pin TXD de Raspberry Pi; sin embargo, los 5 V que vienen del pin Tx de Arduino deben bajar a los 3,3 V que espera Raspberry Pi.

Tendrá que deshabilitar el registro del puerto serie e instalar PySerial siguiendo el Capítulo 9.7.

La siguiente secuencia de comandos de Python (*ardu_pi_serial.py*) le pedirá que introduzca un comando `l` o `r`. Si introduce `l`, el led incorporado en Arduino se encenderá y se apagará. Si, por el contrario, introduce `r`, el Arduino leerá el valor analógico de la entrada analógica A0 y devolverá un número entre 0 y 1023 como lectura.

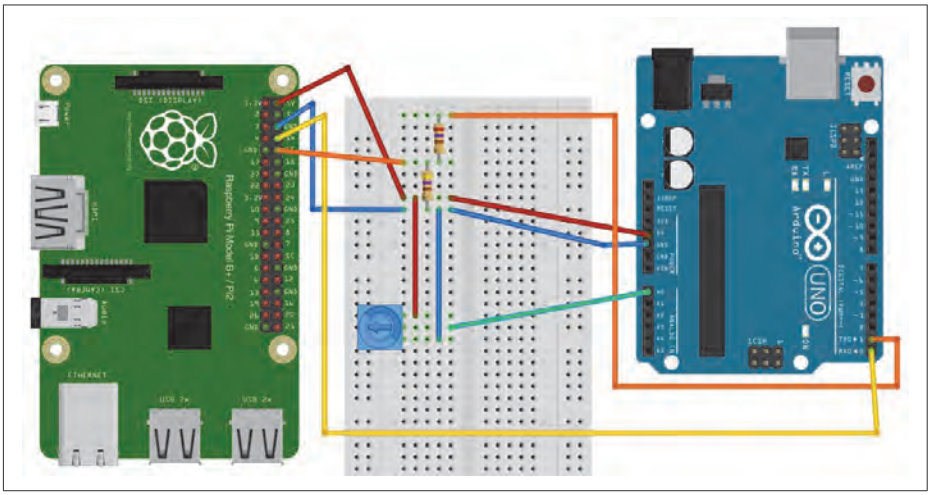


Figura 16.15. *Diagrama de cableado para una comunicación en serie con un Arduino (usando resistores).*

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, también puede descargar el programa de la sección de código desde la página web del libro, donde pone *ardu_pi_serial.py*.

```
import serial

ser = serial.Serial('/dev/ttyAMA0', 9600)

while True:
    command = raw_input("Enter command: l - toggle LED, r - read A0 ")
    if command == 'l':
        ser.write('l')
    elif command == 'r' :
        ser.write('r')
        print(ser.readline())
```

También necesitará subir el siguiente boceto, *ArduinoSerial*, a su Arduino Uno. Puede pegarlo en una ventana nueva de boceto. Está disponible con el resto de los programas para descargar de este libro.

```
#incluye "SoftwareSerial.h"

int ledPin = 13;
int analogPin = A0;

SoftwareSerial ser(8, 9); // RX,  TX

boolean ledOn = false;
```

```
void setup()
{
  ser.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  if (ser.available())
  {
    char ch = ser.read();
    if (ch == 'l')
    {
      toggleLED();
    }
    if (ch == 'r')
    {
      sendAnalogReading();
    }
  }
}

void toggleLED()
{
  ledOn = ! ledOn;
  digitalWrite(ledPin, ledOn);
}

void sendAnalogReading()
{
  int reading = analogRead(analogPin);
  ser.println(reading);
}
```

Ejecute el programa y compruebe que el led *L* incorporado en el Arduino se enciende y se apaga con el comando `l`. Después, intente ajustar el potenciómetro en diferentes posiciones y tome lecturas analógicas enviando el comando `r`:

```
$ sudo python ardu_pi_serial.py
Enter command: l - toggle LED, r - read A0 l
Enter command: l - toggle LED, r - read A0 l
Enter command: l - toggle LED, r - read A0 r
0

Enter command: l - toggle LED, r - read A0 r
540

Enter command: l - toggle LED, r - read A0 r
1023
```

Observaciones

El código en el programa de Python utiliza la biblioteca PySerial para abrir una conexión en el puerto serie. El bucle principal del programa pide repetidamente comandos y los procesa, en ambos casos, escribiendo el comando de un solo carácter al puerto serie. En el caso del comando `r`, también lee una línea desde la conexión serie y luego la imprime.

Tenga en cuenta que, a pesar de las apariencias, el resultado de `serial.readline` es una cadena. Si necesita convertirlo en un número, puede utilizar la función `int` para convertirlo. Por ejemplo:

```
line = ser.readline()
value = int(line)
```

El boceto de Arduino es un poco más complicado. El ejemplo de boceto proporcionado se puedo modificar fácilmente para agregar más entradas o salidas y para responder a más comandos.

Este boceto no utiliza el puerto serie de hardware de Arduino, ya que generalmente se utiliza la conexión USB. En su lugar, se utiliza la biblioteca Arduino llamada *SoftwareSerial* para permitir que los pines 8 y 9 se usen como Rx y Tx, respectivamente.

Como con todos los bocetos de Arduino, debe estar la función `setup` que es llamada cuando Arduino se inicia, y llama a la función `loop` repetidamente.

La función `setup` inicia la comunicación en serie y establece el pin para que el led incorporado (pin 13 de Arduino) sea una salida.

La función `loop` primero comprueba si existen mensajes serie utilizando la función de llamada `ser.available`. Si hay algún mensaje para procesar, lee el carácter y las siguientes cláusulas `if` llaman a la función auxiliar apropiada para procesar el comando.

Para saber más

Para obtener más información acerca de la biblioteca SoftwareSerial, consulte *http://arduino.cc/en/Reference/SoftwareSerial*.

La alternativa a escribir su propio código personalizado para la comunicación es usar PyFirmata (Capítulo 16.4).

Además de comunicarse en serie, puede comunicarse con el Arduino usando I2C (Capítulo 16.12).

16.12 Comunicación personalizada con un Arduino sobre I2C

Problema

Desea intercambiar datos con un Arduino usando las interfaces I2C de ambos dispositivos.

Solución

El bus I2C tiene el concepto de dispositivos maestros y esclavos. El dispositivo maestro controla el bus y se pueden conectar a él una serie de dispositivos esclavos, cada uno en su propia dirección.

Instalará un boceto en el Arduino que lo convierta en un dispositivo esclavo I2C y utilizará la biblioteca *SMBus* de Python para escribir un programa I2C para Raspberry Pi. Las conexiones *SDA*, *SCL* y tierra de ambos dispositivos necesitan estar conectadas entre sí y con el Arduino alimentado por separado o desde el suministro de 5 V de Raspberry Pi.

Para hacer esto necesitará:

- Arduino Uno (vea "Módulos" en la página 476)
- Cables puente macho-hembra (vea "Equipos para prototipos", página 474)

Conecte la placa de pruebas, fijando los componentes al Arduino como se muestra en la Figura 16.16.

La placa Arduino utilizada aquí es la más reciente: Arduino Uno R3. Si tiene una placa más antigua y no tiene pines dedicados a SCL ni SDA, puede conectar los pines SDA y SCL de Raspberry Pi a los pines A4 y A5 de Arduino.

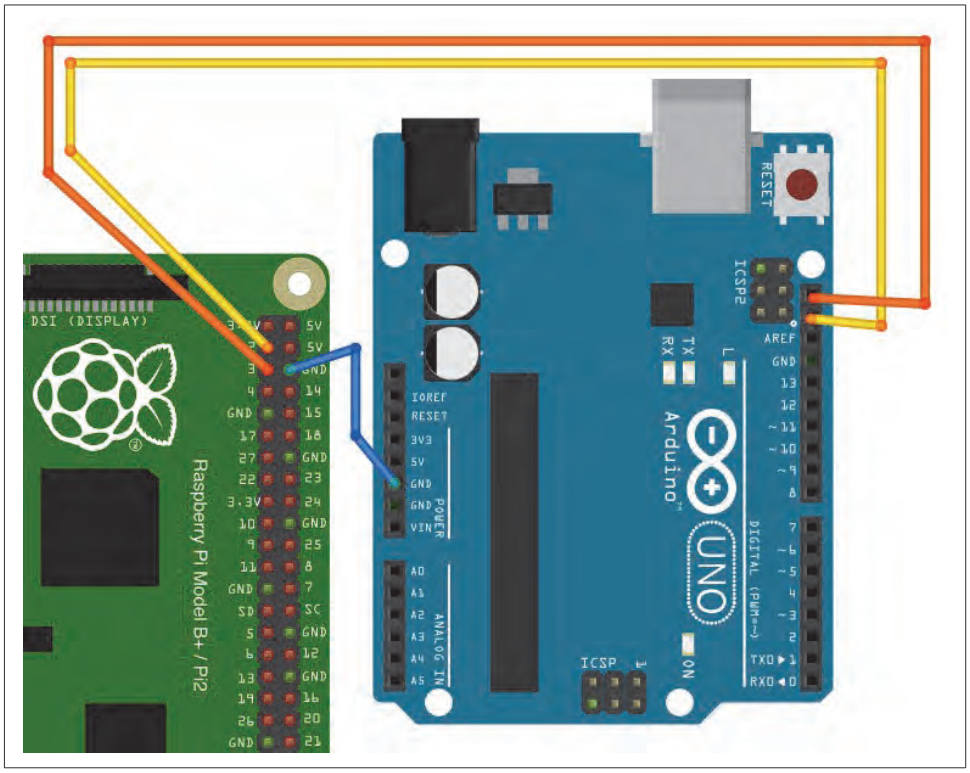


Figura 16.16. *Diagrama de cableado para la comunicación I2C con un Arduino.*

Instale el siguiente boceto en Arduino. Puede encontrarlo, *ArduinoI2C*, en las descargas de programas en la página web del libro.

```
#incluye <Wire.h>

int SLAVE_ADDRESS = 0x04;
int ledPin = 13;
int analogPin = A0;

boolean ledOn = false;

void setup()
{
    pinMode(ledPin, OUTPUT);
    Wire.begin(SLAVE_ADDRESS);
    Wire.onReceive(processMessage);
    Wire.onRequest(sendAnalogReading);
}
```

```
void loop()
{
}

void processMessage(int n)
{
    char ch = Wire.read();
    if (ch =='l')
    {
      toggleLED();
    }
}

void toggleLED()
{
  ledOn = ! ledOn;
  digitalWrite(ledPin, ledOn);
}

void sendAnalogReading()
{
  int reading = analogRead(analogPin);
  Wire.write(reading >> 2);
}
```

Debe seguir el Capítulo 9.4 para configurar Raspberry Pi para la comunicación I2C.

Abra un editor (nano o IDLE) y pegue el siguiente código. Como con todos los ejemplos de programas de este libro, puede descargar el programa de la sección de código en la página web del libro, donde pone *ardu_pi_i2c.py*.

```
import smbus
import time

# para RPI revisión 1, use "bus = smbus.SMBus(0)"
bus = smbus.SMBus(1)

#Esto debe coincidir con lo que está en el boceto de Arduino
SLAVE_ADDRESS = 0x04

def request_reading():
    reading = int(bus.read_byte(SLAVE_ADDRESS))
    print(reading)


while True:
    command = raw_input("Enter command: l - toggle LED, r - read A0 ")
    if command == 'l' :
        bus.write_byte(SLAVE_ADDRESS, ord('l'))
    elif command == 'r' :
        request_reading()
```

El programa de prueba es muy similar al que se utiliza para demostrar la comunicación en serie en el Capítulo 16.11. Está diseñado para mostrar la comunicación en ambas direcciones.

Al introducir el comando `l` se activa y desactiva el led incorporado del Arduino, y mediante el comando `r` se lee la entrada analógica A0 de Arduino. Puede utilizar un cable puente macho-macho para conectar A0 a 3,3 V, 5 V o GND en el Arduino para obtener lecturas diferentes:

```
$ sudo python ardu_pi_i2c.py
Enter command: l - toggle LED, r - read A0 l
Enter command: l - toggle LED, r - read A0 l
Enter command: l - toggle LED, r - read A0 r
184
Enter command: l - toggle LED, r - read A0
```

Observaciones

Arduino actúa como esclavo I2C en esta colocación y, por lo tanto, se le debe dar una *dirección* para que el maestro, ejecutando Raspberry Pi, pueda identificarlo si hay más de un dispositivo esclavo conectado. En este caso, la dirección se establece en 0x04 y se mantiene en la variable `SLAVE_ADDRESS`.

La función `setup` en Arduino configura dos funciones de rellamada. La función `processMessage` se invocará siempre que se produzca `onReceive`. Esto sucederá cada vez que se envíe un comando desde Raspberry Pi. La otra función de rellamada, `sendAnalogReading`, está asociada a `onRequest`. Esto ocurre cuando Raspberry Pi solicita datos, lee el valor analógico, lo divide entre cuatro para que encaje en un solo byte y luego lo envía de vuelta a Raspberry Pi.

La contraparte de Python a este boceto crea primero una instancia de `SMBus` llamada `bus`. Esto toma solo un argumento, el puerto I2C de Raspberry Pi que se utilizará. Será `1` a menos que esté utilizando la primera versión de Raspberry Pi (revisión 1). Estas placas tienen un conector de audio negro en lugar del conector azul claro de las placas de revisión 2. Si tiene una de estas placas más antiguas, use `0` como argumento. El puerto I2C disponible en el conector GPIO se intercambió entre estas dos revisiones.

El programa solicita al usuario un comando (`l` o `r`). Si el comando es `l`, escribe el carácter `l` en el Arduino, lo que a su vez hará que se ejecute el controlador `onReceive` (`process Message`). Esto a su vez llamará a `toggleLED`.

Si, por otro lado, el usuario introduce el comando `r`, llamará a la función `request_reading`. Esto llamará a `read_byte` en la biblioteca `SMBus`, lo que hará que se invoque `onRequest` en el boceto de Arduino.

Es posible que se pregunte por qué es correcto conectar el Arduino de 5 V directamente a las conexiones I2C de Raspberry Pi, sin el módulo convertidor de nivel que tenía que utilizar para las comunicaciones en serie (vea el Capítulo 16.6). La razón es que el estándar del bus I2C funciona a cualquier voltaje que utilicen las resistencias *pull-up* conectadas a SDA y SCL. En este caso, el Arduino Uno no tiene las resistencias *pull-up* conectadas a las líneas I2C. Son proporcionadas por Raspberry Pi, que las arranca a 3,3 V.

Aunque el Arduino no tiene resistencias *pull-up* en las líneas I2C, no ocurre lo mismo en los dispositivos I2C. Si el dispositivo I2C funciona a 5 V, asegúrese de que no tenga resistencias *pull*-up (pero si tiene, por lo general se pueden quitar fácilmente).

Para saber más

Para el equivalente en serie a esta configuración, vea el Capítulo 16.11.

16.13 Utilizar Arduinos pequeños con Raspberry Pi

Problema

Desea utilizar una placa Arduino con Raspberry Pi, pero le gustaría algo más compacto.

Solución

Utilice una de las placas pequeñas de Arduino que van bien con placas de pruebas.

La Figura 16.17 muestra una placa Arduino Pro Mini. Las placas como esta tienen la ventaja de que pueden ser conectadas directamente a una placa de pruebas junto con otros componentes necesarios para el proyecto. La placa Pro Mini está disponible en una versión de 3,3 V, lo que evita tener que utilizar un convertidor de nivel cuando se use con Raspberry Pi.

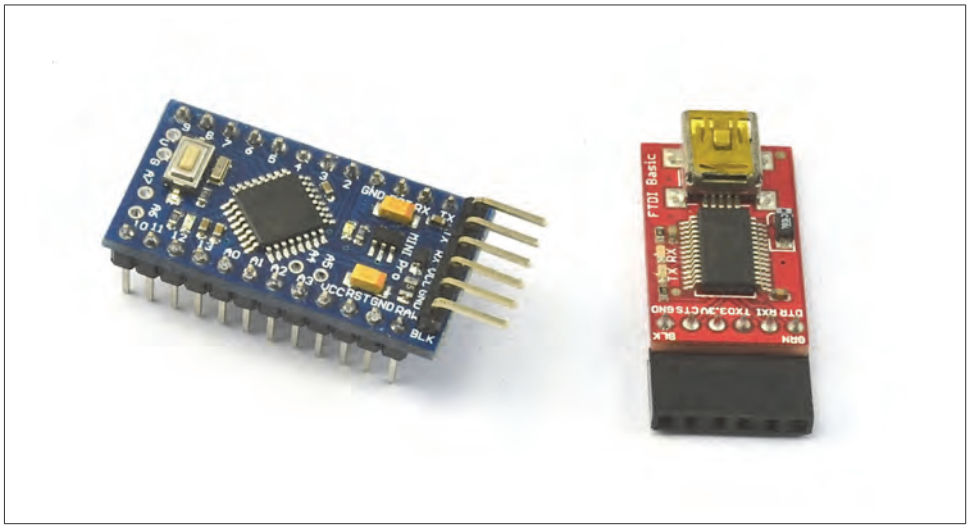


Figura 16.17. *Un Arduino Pro Mini y una interfaz de programación.*

Observaciones

Algunas de estas placas, como la Pro Mini que se muestra en la Figura 16.17, requieren una interfaz de programación USB. Puede programarlas desde Raspberry Pi o desde otro ordenador.

Además de las placas oficiales de Arduino, puede encontrar muchos clones de bajo coste que pueden complementar a Raspberry Pi.

Para saber más

Otras placas a considerar son:

- Teensy
- Arduino Micro
- Arduino Nano

16.14 Comenzar con una tarjeta aLaMode y una Raspberry Pi

Problema

Desea utilizar una placa aLaMode para conectar los aparatos electrónicos externos a Raspberry Pi.

Solución

La placa aLaMode (vea la Tabla A-3), que se muestra en la Figura 16-18, es esencialmente un Arduino Uno con un zócalo GPIO para Raspberry Pi en una esquina. aLaMode se ajusta perfectamente a Raspberry Pi y le permite utilizar Shields de Arduino sin la necesidad de cables adicionales. Aunque está diseñado para la Raspberry Pi original, esta placa también se puede utilizar con Raspberry Pi 2.

Tenga en cuenta que la placa de la Figura 16.18 se muestra sin los conectores de encabezado necesarios para conectar los Shields de Arduino.

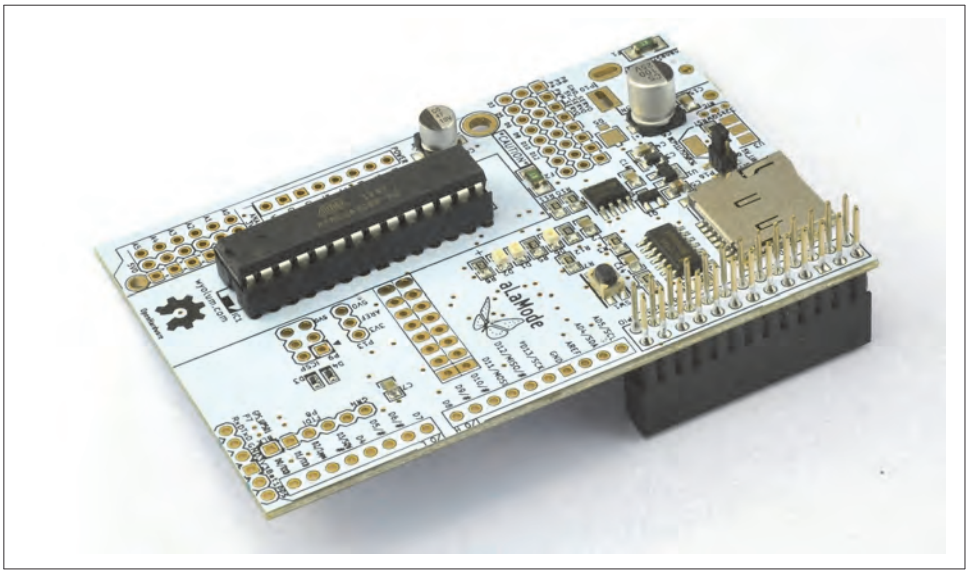


Figura 16.18. *Tarjeta de interfaz aLaMode.*

Las instrucciones de instalación del Capítulo 16.2 también configuran aLaMode en su entorno Arduino. Desde el menú Tools del IDE de Arduino, seleccione Board y luego aLaMode (Figura 16.19).

También es necesario configurar el puerto serie en */devttyS0* antes de programar aLaMode, que se realizará a través de la conexión serie.

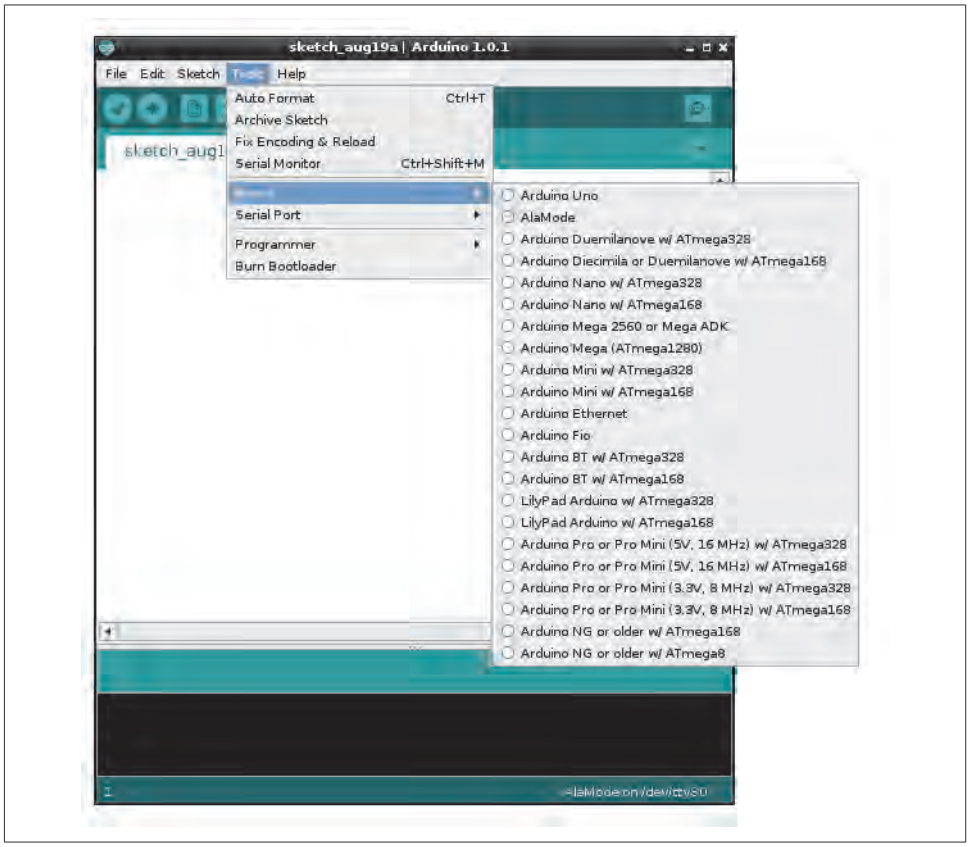


Figura 16.19. *Placa aLaMode en el IDE de Arduino.*

Observaciones

Todos los capítulos de Firmata funcionarán con aLaMode, junto con el ejemplo de comunicación en serie personalizado del Capítulo 16.11. La única modificación necesaria tendrá lugar en los programas de comunicación de Python con aLaMode, ya que deberán utilizar el puerto serie en lugar del puerto USB.

aLaMode está cableado a los pines Tx y Rx de Arduino con conversión de nivel incorporada.

Cuando se utiliza Firmata es necesario cambiar el puerto de */dev/ttyACM0* a */dev/ttyAMA0*:

```
board = pyfirmata.Arduino('/dev/ttyAMA0')
```

Del mismo modo, si está escribiendo su propio código personalizado, tendrá que cambiar la línea que abre la conexión serie para que esté así:

```
ser = serial.Serial('/dev/ttyAMA0', 9600)
```

Para probar el aLaMode cargue el boceto *Standard Firmata* en el aLaMode y ejecute el programa de Python *ardu_flash_ser.py* (Capítulo 16.6).

Algunas de las características interesantes de la placa aLaMode son:

- Puede alimentarse desde la línea de 5 V del conector GPIO de Raspberry Pi o desde un adaptador de corriente de 5 V conectado al zócalo micro-USB.
- El RTC está conectado directamente a Raspberry Pi.
- Los Shields de Arduino pueden unirse con un alto grado de compatibilidad (Capítulo 16.15).
- Puede ser un esclavo de I2C. Además de estar conectados en serie, las conexiones I2C de aLaMode también están conectadas a las de Raspberry Pi (Capítulo 16.12).
- Tiene un lector de tarjetas micro SD.
- Dispone de cabezales para la conexión directa a servomotores (Capítulo 16.10).

aLaMode sufre un pequeño fallo en el diseño; con los shields de Arduino soldados, las entradas analógicas de A0 a A5 pueden tocar fácilmente el metal expuesto de la toma Ethernet RJ45 de Raspberry Pi. Para evitar esto, coloque un par de capas de cinta aislante eléctrica en la superficie superior del conector Ethernet en Raspberry Pi antes de conectar el aLaMode (Figura 16.20).

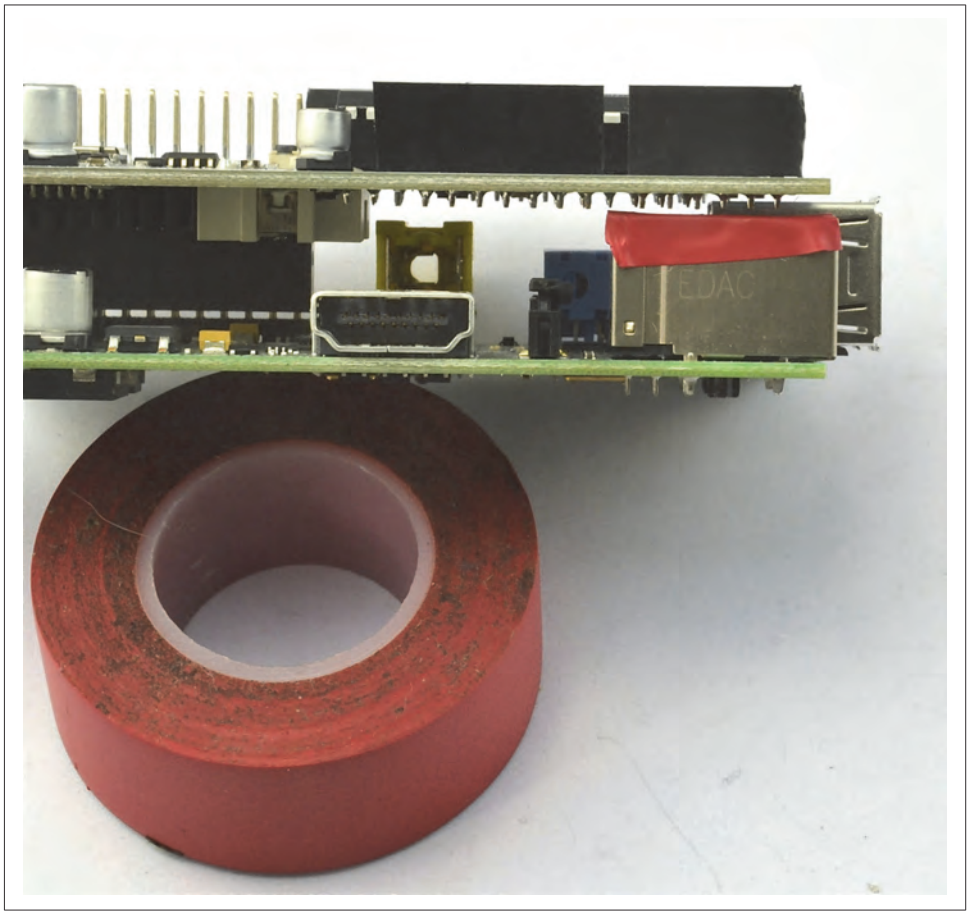


Figura 16.20. *Aislamiento del conector RJ45 en Raspberry Pi.*

Para saber más

Las instrucciones oficiales de aLaMode se pueden encontrar en *http://bit.ly/1d2YMxh*.

16.15 Utilizar un shield de Arduino con aLaMode y Raspberry Pi

Problema

Desea utilizar un shield de Arduino con Raspberry Pi.

Solución

Utilice una tarjeta de interfaz aLaMode. Es posible que la placa necesite los conectores estilo Arduino para soldar los shields.

Por ejemplo, conecte un shield LCD Arduino a aLaMode (Figura 16.21).

Para hacer eso necesitará:

- Placa aLaMode (vea "Módulos" en la página 476)
- Shield LCD Arduino de Freetronic (u otro) (vea "Módulos" en la página 476)

Figura 16.21. *Uso de un shield LCD Arduino con aLaMode.*

Como con todos los ejemplos de programas de este libro, puede descargar este boceto de la sección de código de la página web del libro, donde pone *AlaModeShield*.

```
#incluye <LiquidCrystal.h>

// pines para shields LCD Freetronics

LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
```

```
void setup()
{
  lcd.begin(16, 2);
  lcd.print("Counting!");
}

void loop()
{
  lcd.setCursor(0, 1);
  lcd.print(millis() / s1000);
}
```

Simplemente cargue el boceto en aLaMode y aparecerá el mensaje “Counting” en la fila superior de la pantalla, mientras que en la fila inferior deberá aparecer un número que cuente los segundos.

Observaciones

Este ejemplo utilizó el shield LCD de Freetronics. Hay más shields de pantalla LCD, y algunos tienen diferentes asignaciones de pines. Si utiliza un módulo diferente, deberá cambiar la línea:

```
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
```

Los parámetros son los pines de Arduino que se utilizarán para las siguientes conexiones de LCD: rs, enable, d4, d5, d6 y d7, en ese orden.

Para saber más

Puede encontrar el material de referencia para la biblioteca LiquidCrystal en *http://arduino.cc/en/Reference/LiquidCrystal*.

APÉNDICE A

Componentes y proveedores

Componentes

Las siguientes tablas le ayudarán a encontrar las piezas usadas en este libro. Siempre que sea posible, he puesto el código del producto de algunos proveedores.

Ahora hay muchos proveedores de componentes electrónicos que atienden al fabricante y a los aficionados a la electrónica. Algunos de los más populares se enumeran en la Tabla A-1.

Tabla A-1. *Proveedores de componentes*

Proveedores	Página web	Notas
Adafruit	*http://www.adafruit.com*	Bueno para los módulos
Digikey	*http://www.digikey.com*	Amplia gama de componentes
MakerShed	*http://www.makershed.com*	Bueno para módulos, kits y herramientas
MCM Electronics	*http://www.mcmelectronics.com*	Amplia gama de componentes
Mouser	*http://www.mouser.com*	Amplia gama de componentes
SeeedStudio	*http://www.seeedstudio.com*	Interesantes módulos de bajo coste
SparkFun	*http://www.sparkfun.com*	Bueno para los módulos
MonkMakes	*http://www.monkmakes.com*	Kits electrónicos para Raspberry Pi, etc.
Pimoroni	*https://shop.pimoroni.com*	Fabricante británico de interesantes HAT y vendedores
Polulu	*https://www.pololu.com*	Ideal para controladores de motores y robots
CPC	*http://cpc.farnell.com*	Situado en Reino Unido. Amplia gama de componentes
Ciseco	*http://shop.ciseco.co.uk*	Proveedores de PiLite, Humble Pi, etc.
Farnell	*http://www.farnell.com*	Internacional. Amplia gama de componentes
Maplin	*http://www.maplin.co.uk*	Situado en Reino Unido. Componentes físicos
Pi store Proto-pic	*http://proto-pic.co.uk*	Situado en Reino Unido. Módulos SparkFun y Adafruit

La otra gran fuente de componentes es eBay.

La búsqueda de componentes puede ser difícil y llevar mucho tiempo. El motor de búsqueda Octopart puede ser muy útil para localizar componentes. MonkMakes, Adafruit y Sparkfun tienen paquetes de componentes para empezar.

Equipos para prototipos

Muchos de los proyectos de *hardware* en este libro utilizan cables puente de varios tipos. Son especialmente útiles los cables macho-hembra (para conectar el conector GPIO de la Raspberry Pi GPIO a la placa de pruebas) y macho-macho (para realizar conexiones en la placa de pruebas). Los cables hembra-hembra son útiles, de vez en cuando, para conectar módulos directamente a los pines GPIO. Rara vez se necesitan cables de más de 3 pulgadas (75 mm). La Tabla A-2 enumera algunas especificaciones de cables puente y de placas de pruebas junto con sus proveedores.

Una forma práctica de empezar con una placa de pruebas, cables puente y algunos componentes básicos es comprando un kit de inicio como el Kit de Inicio en Electrónica para Raspberry Pi de Monk Makes.

Tabla A-2. *Equipos para prototipos*

Descripción	Proveedores
Cables puente M-M	SparkFun: PRT-08431; Adafruit: 759
Cables puente M-H	SparkFun: PRT-09140; Adafruit: 825
Cables puente H-H	SparkFun: PRT-08430; Adafruit: 794
Placa de pruebas de tamaño medio	SparkFun: PRT-09567; Adafruit: 64
Pi Cobbler	Adafruit: 1105
Raspberry (26 pines)	Adafruit: 1772
Raspberry (40 pines)	Adafruit: 2196
Kit de Inicio en Electrónica para Raspberry Pi	Amazon; monkmakes.com
Adafruit PermaProto para Pi (media placa de pruebas)	Adafruit: 1148
Adafruit PermaProto para Pi (placa de pruebas entera)	Adafruit: 1135
Adafruit PermaProto HAT	Adafruit: 2314
Adaptador de jack a terminal de tornillo de CC (hembra)	Adafruit: 368

Resistores y condensadores

La Tabla A-3 muestra los resistores y condensadores utilizados en este libro y algunos de sus proveedores.

Tabla A-3. *Resistores y condensadores*

Componente	Proveedor
Resistor de 270 Ω 0,25 W	Mouser: 293-270-RC
Resistor de 470 Ω 0,25 W	Mouser: 293-470-RC
Resistor de 1 kΩ 0,25 W	Mouser: 293-1k-RC
Resistor de 3,3k Ω 0,25 W	Mouser: 293-3.3k-RC
Resistor de 4,7 kΩ 0,25 W	Mouser: 293-4.7k-RC
Potenciómetro de 10 kΩ	Adafruit: 356; SparkFun: COM-09806; Mouser: 652-3362F-1-103LF
Fotorresistor	Adafruit: 161; SparkFun: SEN-09088
Condensador 330 nF	Mouser: 80-C330C334K5R
Termistor T0 de 1 k Beta 3800 NTC	Mouser: 871-B57164K102J (Nota: el Beta es 3730)

Transistores y diodos

La Tabla A-4 muestra los transistores y diodos utilizados en este libro y algunos de sus proveedores.

Tabla A-4. *Transistores y diodos*

Componente	Proveedor
Transistor FQP30N06L N-canal de nivel lógico MOSFET	Mouser: 512-FQP30N06L; Sparkfun: COM-10213
Transistor bipolar 2N3904 NPN	SparkFun: COM-00521; Adafruit: 756
Diodo 1N4001	Mouser: 512-1N4001; SparkFun: COM-08589; Adafruit: 755
TransistorTIP120 Darlington	Adafruit: 976; CPC: SC10999
Transistor 2N7000 MOSFET	Mouser: 512-2N7000; CPC: SC06951

Circuitos integrados

La Tabla A-5 muestra los circuitos integrados utilizados en este libro y algunos de sus proveedores.

Tabla A-5. *Circuitos integrados*

Componente	Proveedor
Regulador de tensión 7805	SparkFun: COM-00107; Adafruit: 2164; Mouser: 511-L7805CV, CPC: SC10586
Propulsor de motor L293D	SparkFun: COM-00315; Adafruit: 807; Mouser: 511-L293D; CPC: SC10241
Controlador IC ULN2803 Darlington	SparkFun: COM-00312; Adafruit: 970; Mouser: 511-ULN2803A; CPC: SC08607
Sensor de temperatura DS18B20	SparkFun: SEN-00245; Adafruit: 374; Mouser: 700-DS18B20; CPC: SC10426
ADC IC de 8 canales MCP3008	Adafruit: 856; Mouser: 579-MCP3008-I/P; CPC: SC12789
Sensor de temperatura TMP36	SparkFun: SEN-10988; Adafruit: 165; Mouser: 584-TMP36GT9Z; CPC: SC10437

Optoelectrónica

La Tabla A-6 enumera los componentes optoelectrónicos utilizados en este libro y algunos de sus proveedores.

Tabla A-6. *Optoelectrónica*

Componente	Proveedor
Led rojo 5 mm	SparkFun: COM-09590; Adafruit: 299
Led de cátodo común RGB	SparkFun: COM-11120; eBay
Sensor IR TSOP38238	SparkFun: SEN-10266; Adafruit: 157

Módulos

La Tabla A-7 enumera los módulos utilizados en este libro y sus proveedores.

Tabla A-7. *Módulos*

Componente	Proveedor
Módulo de cámara Raspberry Pi	Adafruit: 1367; MCM: 28-17733; CPC: SC13023
Arduino Uno	SparkFun: DEV-11021; Adafruit: 50; CPC: A000066
Convertidor de nivel de cuatro vías	SparkFun: BOB-11978; Adafruit: 757
Convertidor de nivel de ocho vías	Adafruit: 395
Convertidor de potencia/cargador LiPo	SparkFun: PRT-11231
PowerSwitch tail	Adafruit: 268
Regulador servo de 16 canales	Adafruit: 815
Propulsor de motor 1A dual	SparkFun: ROB-09457
Placa RasPiRobot V3	Adafruit: 1940; Amazon
Kit de MonkMakes RasPiRobot Rover	Amazon
Placa de prototipado para Pi	Adafruit: 801
Detector de movimiento PIR	Adafruit: 189
Módulo GPS Venuse	SparkFun: GPS-11058
Sensor de metano	SparkFun: SEN-09404
Tarjeta de sensor de fuga de gas	SparkFun: BOB-08891
Acelerómetro de tripe eje ADXL335	Adafruit: 163
Led de 4x7 segmentos con paquete I2C	Adafruit: 878
Matriz de led de píxel cuadrado con paquete I2C	Adafruit: 902
Tarjeta de interfaz aLaMode	Makershed: MKWY1; Seeedstudio: ARD10251P
Matriz LCD Freetronics Arduino	www.freetronics.com
Módulo RTC	Adafruit: 264
Módulo LCD 16x2 HD44780 compatible	SparkFun: LCD-00255; Adafruit: 181

Componente	Proveedor
Sense HAT	Adafruit: 2738
HAT táctil capacitivo de Adafruit	Adafruit: 2340
HAT de motor paso a paso	Adafruit: 2348
HAT de 16 canales PWM	Adafruit: 2327
Pimoroni Explorer HAT Pro	pimoroni.com; Adafruit: 2427
Botón Squid	monkmakes.com; Amazon
Led RGB Raspberry Squid	monkmakes.com; Amazon
Pantalla OLED I2C de píxeles 128x64	eBay

Varios

Varias herramientas y componentes utilizados en este libro y algunos de sus proveedores se enumeran en la Tabla A-8.

Tabla A-8. *Varios*

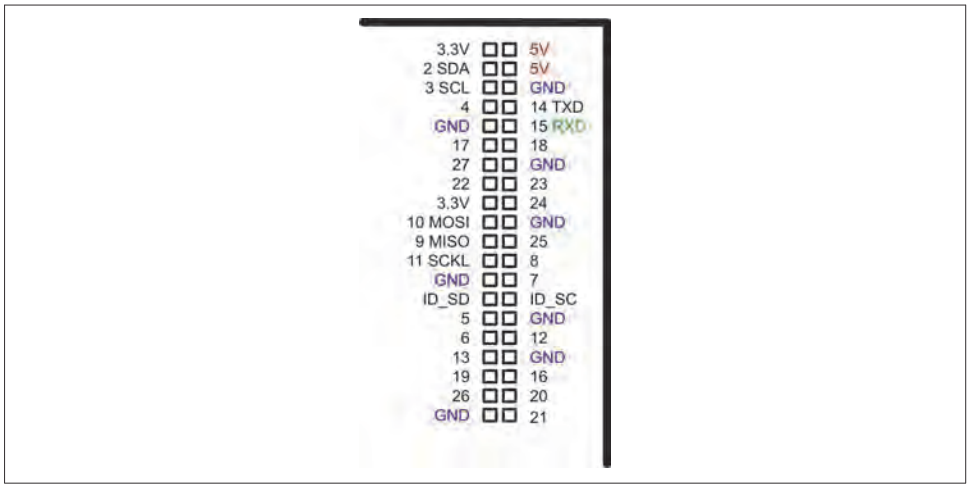
Componente	Proveedor
Batería LiPo 1200 mAh	Adafruit: 258
Relé de 5 V	SparkFun: COM-00100
Medidor de panel 5 V	SparkFun: TOL-10285
Servomotor estándar	SparkFun: ROB-09065; Adafruit: 1449
Servomotor mini 9 g	Adafruit: 169
Fuente de alimentación 5 V 1A	Adafruit: 276
Motor CC 6 V de baja potencia	Adafruit: 711
Pines encabezado 0,1 pulgadas	SparkFun: PRT-00116; Adafruit: 392
Motor paso a paso unipolar 5 V de 5 pines	Adafruit: 858
Motor paso a paso bipolar 12 V de 4 pines	Adafruit: 324
Magician chassis con motorreductores	SparkFun: ROB-10825
Compartimento para pilas 6×AA 10825	Adafruit: 248
Pulsador táctil	SparkFun: COM-00097; Adafruit: 504
Interruptor deslizante en miniatura	SparkFun: COM-09609; Adafruit: 805
Codificador rotatorio	Adafruit: 377
Teclado 4×3	SparkFun: COM-08653
Zumbador piezo	SparkFun: COM-07950; Adafruit: 160
Interruptor de láminas	Adafruit: 375

APÉNDICE B

Asignación de patillaje de Raspberry Pi

Modelo B, B+, A+ y Zero de la Raspberry Pi 3/2

La Figura B-1 muestra la asignación de patillaje del actual GPIO de la Raspberry Pi



de 40 pines.

Figura B-1. *Asignación de patillaje del GPIO de Raspberry Pi de 40 pines*

Modelo B revisión 2 y modelo A de la Raspberry Pi

Si usted tiene una Raspberry Pi, es probable que sea el modelo B revisión 2, como se muestra en la Figura B-2.

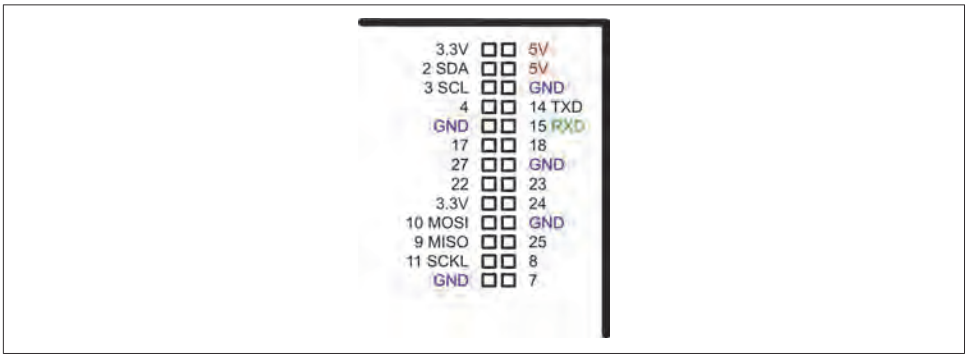


Figura B-2. *Asignación de patillaje del GPIO del modelo B rev. 2 y modelo A*

Modelo B revisión 1 de la Raspberry Pi

La primera versión lanzada de Raspberry Pi modelo B (revisión 1) tiene algunas ligeras diferencias en la asignación de patillaje con la revisión 2 que siguió. Esta es la única versión de Raspberry Pi que no es compatible con las asignaciones posteriores. Los pines incompatibles que han cambiado se resaltan en negrita en la Figura B-3.

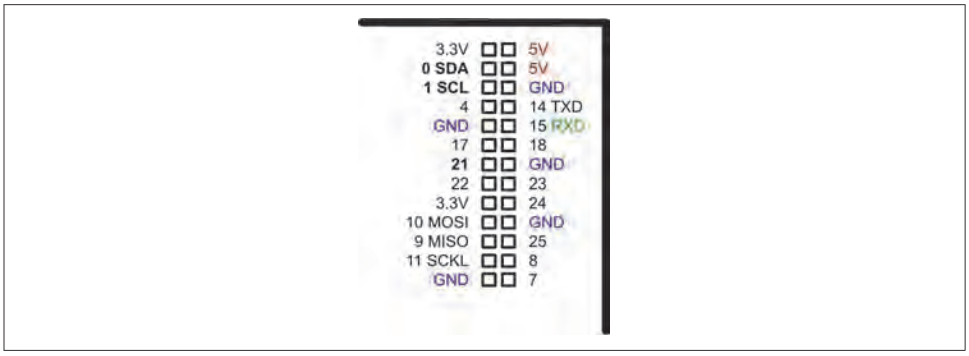


Figura B-3. *Asignación de patillaje del GPIO del modelo B rev. 1*