Cliente Bittorrent Deluge • Chatbot • Estación Arcade Portatil • Clipgrab

ODROID Magazine

Año Cuatro
Núm. #37
Ene 2017

ODROIDS Alrededor del Mundo

Celebramos el alcance de nuestros gadgets que recorren el mundo y han revolucionado la informática portátil

- Sistema IoT de notificación y conservación del entorno de bodegas
- Configura una cámara de visión trasera para tu bici con oCAM

Qué defendemos...

Nos esmeramos en presentar una tecnología punta, futura, joven, técnica y para la sociedad de hoy.

Nuestra filosofía se basa en los desarrolladores. Continuamente nos esforzamos por mantener estrechas relaciones con éstos en todo el mundo.

Por eso, siempre podrás confiar en la calidad y experiencia que representa la marca distintiva de nuestros productos.

Simple, moderno y único.
De modo que tienes a tu alcance lo mejor para hacer todo lo que imagines

HARDKERNEL

Pollin ELECTRONIC

Realizamos envios de ODROID-C2 and ODROID-XU4 a los países de la UE! Ven y visita nuestra tienda online!

Dirección: Max-Pollin-Straße 1
85104 Pförring Alemania

Teléfono & Fax
telf : +49 (0) 8403 / 920-920
email : service@pollin.de

Nuestros productos ODROID se pueden encontrar en: http://bit.ly/1tXPXwe

EDITORIAL

ODROID Magazine ya está en su cuarto año!!! Estamos encantados de continuar presentando artículos y proyectos que aportan la comunidad y que destacan por la versatilidad y portabilidad de la fantástica línea de ordenadores de placa reducida de Hardkernel. Algunos de los proyectos que hemos presentado el pasado año han incluido un ODROID-XU4 refrigerado por agua, un sistema Ambilight 4K, una mesa con pantalla táctil de 42 pulgadas y un ordenador portátil. Estamos deseando conocer los proyectos ODROIDians innovadores y únicos que nos deparará este 2017 y en el futuro.

Los proyectos IoT se han vuelto muy populares entre los ODROIDs, y nuestro experto IoT Miltiadis nos presenta un proyecto que combina dos de sus pasatiempos, el vino y los ordenadores, le permite monitorizar su bodega para asegurarse de que su valiosa colección envejece adecuadamente. El sistema también le alerta vía SMS cuando los vinos están listos para ser disfrutardos. Max detalla cómo configurar un chatbot, Brian nos muestra cómo ha montado una cámara de visión trasera en su bicicleta para así pedalear mas seguro, @synportack24 nos da una visión general del cliente BitTorrent Deluge y Tobias nos muestra cómo descargar videos para verlos sin conexión. Nuestros juegos Android destacados de este mes incluyen Pixel Dodgers y Chrome Death para que te diviertas horas y horas.

ODROID Magazine, que se publica mensualmente en http://magazine.odroid.com/, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera arquitectura ARM "big.LITTLE" del mundo basada en una única placa.
Para información sobre cómo enviar articulos, contacta con odroidmagazine@gmail.com, o visita http://bit.ly/1yplmXs.
Únete a la comunidad ODROID con miembros en más de 135 países en http://forum.odroid.com/ y explora las nuevas tecnologías que te ofrece Hardkernel en http://www.hardkernel.com/

HARDKERNEL

NUESTRO MARAVILLOSO PRESONAL ODROIDIAN:

Rob Roy, Editor Jefe

Soy un programador informático que vive y trabaja en San Francisco, CA, en el diseño y desarrollo de aplicaciones web para clients locales sobre mi cluster ODROID. Mis principales lenguajes son jQuery, angular JS y HTML5/CSS3. También desarrollo SO precompilados, Kernels personalizados y aplicaciones optimizadas para ODROID basadas en las versiones oficiales de Hardkernel, por los cuales he ganado varios Premios. Utilizo mi ODROIDs para diversos fines, como centro multimedia, servidor web, desarrollo de aplicaciones, estación de trabajo y como plataforma de juegos. Puedes echar un vistazo a mi colección de 100 GB de software ODROID, kernel precompilados e imágenes en `http://bit.ly/1fsaXQs`.

Bruno Doiche, Editor Artístico Senior

Ha pasado cuatro años, y Bruno todavía sostiene su cerveza en alto y hace que te preguntes: ¿No está esa cerveza ya caliente? ¿No se ha cansado de aguantarla tanto tiempo? ¿O es simplemente una foto?

Manuel Adamuz, Editor Español

Tengo 31 años y vivo en Sevilla, España, aunque nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y las tecnologías de la información: ISO 9001, ISO 27001, ISO 20000 Soy un apasionado de la informática, especialmente de los microordenadores como el ODROID, Raspberry Pi, etc. Me encanta experimentar con estos equipos y traducir ODROID Magazine. Mi esposa dice que estoy loco porque sólo pienso en ODROID. Mi otra afición es la bicicleta de montaña, a veces participo en competiciones semiprofesionales.

Nicole Scott, Editor Artístico

Soy una experta en Producción Transmedia y Estrategia Digital especializa en la optimización online y estrategias de marketing, administración de medios sociales y producción multimedia impresa, web, vídeo y cine. Gestiono múltiples cuentas con agencias y productores de cine, desde Analytics y Adwords a la edición de vídeo y maquetación DVD. Tengo un ODROID-U3 que utilizo para ejecutar un servidor web sandbox. Vivo en el área de la Bahía de California, y disfruta haciendo senderismo, acampada y tocando música. Visita mi web http://www.nicolecscott.com.

James LeFevour, Editor Artístico

Soy un especialista en medios digitales que disfruta trabajando como freelance en marketing de redes sociales y administración de sitios web. Cuanto más aprendo sobre las posibilidades de ODROID más me ilusiona probar cosas nuevas con él. Me traslade a San Diego desde el Medio Oeste de los EE.UU. Continuo muy enamorado de muchos de los aspectos que la mayoría de la gente de la Costa Oeste ya da por sentado. Vivo con mi encantadora esposa y nuestro adorable conejo mascota; el cual mantiene mis libros y material informático en constante peligro.

Andrew Ruggeri, Editor Adjunto

Soy un ingeniero de sistemas Biomédicos anclado en Nueva Inglaterra que actualmente trabaja en la industria aeroespacial. Un microcontrolador 68HC11 de 8 bits y el código ensamblador son todo lo que me interesa de los sistemas embebidos. Hoy en día, la mayoría de los proyectos en los que trabajo están en lenguajes C y C ++, o en lenguajes de alto nivel como C# y Java. Para muchos proyectos, utilizo placas ODROID, pero aún sigo intentando utilizar los controladores de 8 bits cada vez que puedo (soy un fan de ATMEL). Aparte de la electrónica, soy un amante de la fotografía analógica y desarrollo la película friki con la que disfruto intentando hablar en idiomas extranjeros.

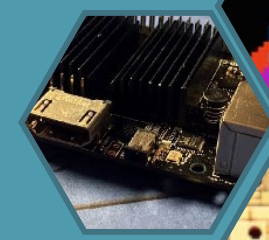
Venkat Bommakanti, Editor Adjunto

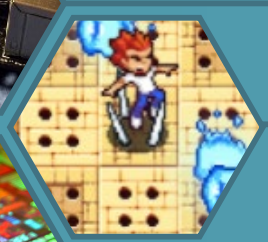
Soy un apasionado de los ordenadores desde la bahía de San Francisco en California. Procuro incorporar muchos de mis intereses en proyectos con ordenadores de placa reducida, tales como pequeños modificaciones de hardware, carpintería, reutilización de materiales, desarrollo de software y creación de grabaciones musicales de aficionados. Me encanta aprender continuamente cosas nuevas, y trato de compartir mi alegría y entusiasmo con la comunidad.

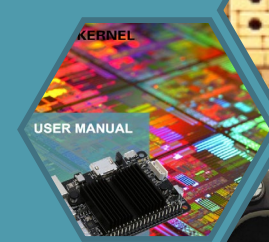
Josh Sherman, Editor Adjunto

Soy de la zona de Nueva York, y ofrezco mi tiempo como escritor y editor para ODROID Magazine. Suelo experimentar con los ordenadores de todas las formas y tamaños: haciendo trizas las tablets, convirtiendo Raspberry Pi en PlayStations y experimentado con los ODROIDs y otros SoCs. Me encanta trabajar con los elementos básicos y así poder aprender más, y disfrutar enseñando a otros escribiendo historias y guías sobre Linux, ARM y otros proyectos experimentales divertidos.

INDICE







SISTEMA IOT DE NOTIFICACION Y CONSERVACION DEL AMBIENTE DE UNA BODEGA DE VINO

por Miltiadis Melissas (@miltos)

Todo buen conservador de vino sabe lo importante que es almacenar el vino en unas condiciones que sean lo más parecidas posibles a las de una bodega tradicional con la finalidad de garantizar el sabor y la calidad del vino. Estas condiciones tienen en cuenta la temperatura, la humedad y la iluminación del espacio utilizado como bodega. En concreto, la temperatura debe mantenerse lo más cerca posible a los 12 grados Celsius, aunque son aceptables variaciones entre los 10 y 14 grados, con fin de garantizar un buen proceso de maduración. Con respecto a la humedad, ésta debe ser al menos del 50% para asegurar un ambiente cálido y húmedo para el vino. Por último, debemos contar con una habitación oscura tan a menudo como sea posible, ya que la luz puede causar oxidación y acelerar el proceso de envejecimiento de un modo poco natural. Todos estos elementos deben estar en un delicado equilibrio para asegurarnos de que el vino envejece correctamente.

Este proyecto es la piedra angular de mis 3 anteriores artículos sobre el Internet de las cosas (IoT) usando un ODROID-C2. Este dispositivo IoT, el sistema de notificación y conservación del ambiente de la bodega, por así decirlo, monitoriza las condiciones de fermentación de la bodega utilizando un sensor de temperatura, humedad e iluminación. La función más importante de este proyecto es la posibilidad de recibir una alerta SMS si no se cumplen las condiciones ideales de fermentación, no obstante los datos de fermentación también se pueden enviar diariamente a la nube para su monitorización remota. En este proyecto, utilizaremos una empresa de comunicación basada en la nube (PaaS) para el correcto envío de esos mensajes SMS (es decir, Twilio). Consulta nuestro anterior artículo publicado en la edición de octubre de Hardkernel (http://bit.ly/2fFXJHQ) para obtener detalles sobre cómo utilizar este servicio. También añadiremos un LED al dispositivo que es controlado mediante programación, y que indicará a un usuario que se encuentre físicamente en la sala si las condiciones ideales son las adecuadas en ese momento.

Figura 1 - Una bodega es el mejor entorno para la fermentación

Montando los circuitos

Utilizaremos una placa de pruebas para alojar nuestros componentes electrónicos tal y como lo hicimos con nuestro anterior proyecto IoT, Controlador de iluminación doméstica y alumbrado público con sistema de notificación por SMS (http://bit.ly/2fFXJHQ), para evitar la molestia de tener que hacer soldaduras y diseñar una PCB para nuestro prototipo. Conectaremos varios circuitos a los pines GPIO del ODROID-C2 utilizando cables puente Dupont, tal y como se muestra en la Figura 2. Aquí tienes una lista con todo el hardware y software que vamos a utilizar en este proyecto:

Hardware:

- ODROID-C2 con Ubunt 16.04
- Fuente de alimentación 5V/2A de HardKernel (http://bit.ly/1X0bgdt)
- Placa de pruebas y cables puentes macho a hembra Dupont
- Un sensor de temperatura y humedad DHT11
- Una foto resistencia
- (2) Resistencias (4.7K y 220Ω)
- Un condensador de 1 µF
- Un LED RGB

Software:

- Ubuntu 16.04 v2.0 de Hardkernel (http://bit.ly/2cBibbk)
- Python 2.7 o 3.3 (preinstalado en Ubuntu)
- Librería WiringPi para controlar los pines GPIO del C2. Puedes aprender a instalarla con ayuda de http://bit.ly/2ba6h8o

Montando el dispositivo IoT

Para nuestras conexiones por cable, usamos los cables macho a hembra Dupont. En el extremo hembra del cable conectaremos al cabezal macho del ODROID-C2, y el extremo macho lo conectaremos a los orificios de la placa de pruebas. Consulta el esquema de distribución de pines de Hardkernel cuando vayas a hacer las conexiones, está disponible en (http://bit.ly/2aXAlmt). El pin físico 1 proporciona la VCC (3,3 V) a nuestro circuito, y lo conectamos en la segunda línea vertical de nuestra placa de pruebas. Puesto que vamos a utilizar el Pin 6 como puesta a tierra, éste lo conectaremos a la primera línea vertical de nuestra placa de pruebas, cerca del borde. El sensor DHT11 tiene tres pines, conectamos el Pin 1 a 3.3V en la placa, el Pin 2 (cable marrón) en el centro al ODROID-C2, el Pin 7 y el último (Pin 3) a la puesta a tierra. En la Figura 2 puedes ver todas las conexiones. A continuación, conectaremos la fotorresistencia/fotocélula al pin físico 18 en uno de sus extremos, el otro va al VCC (3,3 V). Ten en cuenta que este cable/puente rojo Dupont está conectado a la línea vertical de nuestra placa de pruebas. Debes prestar especial atención a la polaridad del condensador (1uF), ya que necesitamos conectar su extremo negativo marcado con el símbolo (-) a la puesta a tierra. El extremo positivo del condensador se conecta a la fotorresistencia a través del cable Dupont amarillo y desde allí al pin físico 18. Finalmente, el LED funcional se conectará al pin físico 16 con su ánodo (+) mientras que el cátodo (-) estará

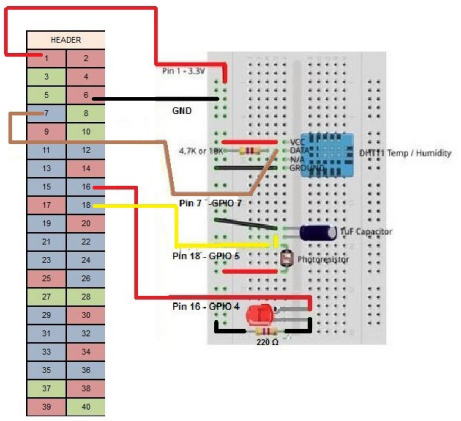


Figura 2 – Observando con más detalle la distribución de la placa de pruebas

conectado a la puesta a tierra a través de una resistencia de 220Ω ¡Ya está! Todo nuestro cableado físico está montado.

Antes de conectar nada a tu ODROID-C2, desconecta la alimentación. Es importante tener en cuenta que puede dejar inservible tu ODROID-C2 a causa de un cortocircuito por una conexión incorrecta. Simplemente ten cuidado y compruébalo todo antes de encender tu C2.

Escribiendo el código

Hemos dividido el código en secciones (trozos de código) para facilitar la lectura. Comenzamos importando los módulos necesarios. Luego definiremos los pines del ODROID-C2 que vamos a utilizar. En el siguiente paso, configuramos el módulo wiringPi de acuerdo con la guía de Hardkernel (http://bit.ly/2aXAlmt). Procedemos a configurar el LED funcional. A continuación, conectaremos la fotorresistencia y el sensor DHT11 al C2. Por supuesto, antes de hacer eso, definiremos las variables necesarias para controlarlos en Python. El siguiente fragmento de código es muy importante ya que lo usaremos para extraer los datos del sensor DHT11. Afortunadamente, existe un trozo de código disponible en GitHub en http://bit.ly/2gAaUfK que hace este trabajo. Puesto que vamos a utilizar el servicio de Twilio, lo configuraremos a continuación. Por último, pero no menos importante, buscaremos las condiciones más adecuadas dentro de la bodega. La temperatura

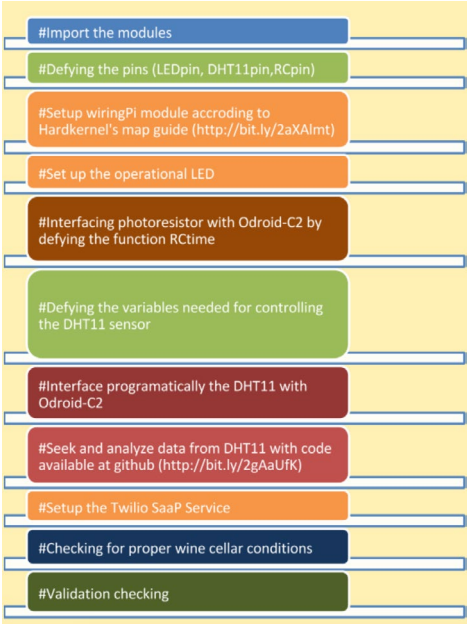


Figura 3 - Un desglose de los diversos trozos de código y hardware usado

debe estar alrededor de los 12 grados Celsius, la humedad al menos al 50% y la ausencia/presencia de iluminación. El último trozo de código únicamente tiene fines de validación. Consulta el diagrama por bloques en la Figura 3.

Utilizando Twilio

Consulta el artículo en la edición de noviembre de ODROID Magazine (http://bit.ly/2fFXJHQ) para obtener información sobre cómo configurar una cuenta en Twilio y cómo utilizar este servicio de comunicación en la nube (PaaS). Lo que realmente necesitamos son las claves API (account_sid, auth_token) para usar este servicio y un script python para enviar mensajes SMS al usuario cuando se cumplan ciertas condiciones. El siguiente trozo de código hace exactamente eso:

```
def sent_SMS():
    from twilio.rest import
TwilioRestClient

    account_sid = "xxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxx" # Your
Account SID from www.twilio.com/
console
    auth_token  = "xxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxx"  # Your Auth
```

```
Token from www.twilio.com/console

    client =
TwilioRestClient(account_sid,
auth_token)
    message = client.messages.
create(body="Alert!!! The condi-
tions on the wine cellar is out
of range!",
    to="+xxxxxxxxxx",    # Re-
place with your phone number
    from_="+xxxxxxxxxx")   # Re-
place with your Twilio number

    print(message.sid)
```

Conectando Twilio con la fotorresistencia

```
def RCtime(RCpin):
    reading = 0
    odroid.pinMode(RCpin,1)
    odroid.digitalWrite(RCpin,0)
    time.sleep(0.1)
    odroid.pinMode(RCpin,0)
    # This takes about 1 milli-
second per loop cycle
    while (odroid.
digitalRead(RCpin) == 0):
          reading += 1
    return reading
   ...
def printData():
   ...
   #Check the lighting conditions
in your wine cellar
   if  (RCtime(5)<2500):
        print ("Alert!!! Light in
the wine cellar!")
        #send_SMS()
```

El proceso de conexión de la fotorresistencia con Twilio está describió en nuestro anterior artículo, "ODROIDC2 como un dispositivo IoT: Controlador de iluminación doméstica y alumbrado público con un sistema de notificación por SMS" en el número de noviembre de ODROID Magazine (http://bit.ly/2fFXJHQ). La parte crítica es definir el umbral de iluminación correcto a partir del cual se activa el envío del mensaje SMS llamando a la función Twilio.

Debes probar y encontrar este nivel de iluminación tan específico de acuerdo a las condiciones de iluminación de tu propia bodega a través del método de ensayo y error. Puesto que no disponiamos de una bodega, nosotros probamos la fotorresistencia bajo las condiciones normales de nuestra habitación, y descubrimos que el envío del mensaje SMS se activa cuando el valor es inferior a 2500:

```
< #Check the lighting conditions
in your wine cellar
   if  (RCtime(5)<2500):
        print ("Alert!!! Light in
the wine cellar!")
        send_SMS()>
```

Conectando Twilio con el sensor DHT11

Aquí tienes el fragmento que usaremos con el sensor de temperatura y humedad:

```
def pullData():
#{{{ Pull data from GPIO.odroid
    global data
    global effectiveData
    global pin

    data = []
    effectiveData = []

    odroid.pinMode(DHT11pin,1)
    odroid.
digitalWrite(DHT11pin,1)
    time.sleep(0.025)
    odroid.
digitalWrite(DHT11pin,0)
    time.sleep(0.14)

    odroid.pinMode(DHT11pin,0)
    odroid.
pullUpDnControl(DHT11pin,2)

    for i in range(0,2900):
        data.append(odroid.
digitalRead(DHT11pin))

    """
    for i in range(0,len(data)):
        print "%d" % data[i],
    print
    """

#}}}

 ...

def printData():
   global Humidity
   global Temperature

   print "H: "+Humidity
   print "T: "+Temperature
   #Checking here if the condi-
tions in the Wine Cellar is met
   if (int(Humidity)<50) or
((int(Temperature)<10) or
(int(Temperature)>14)):
      print ("Alert!!! Tem-
perature and/or Humidity out of
range!")
        #sent_SMS()
  ...

#}}}
```

Conectamos el DHT11 al ODROID-C2 utilizando la librería wiringPi, y la forma en que lo hacemos no era muy diferente al método utilizado con la fotorresistencia. Puesto que ambos elementos son sensores, leemos directamente los datos desde su pin de entrada y los almacenamos en una variable o en una matriz. En el caso del DHT11, almacenando los datos en una matriz. La siguiente línea de código hace exactamente eso:

```
< data.append(odroid.
digitalRead(DHT11pin))>
```

Aquí hay una cuestión muy importante: la velocidad a la que leemos los datos del sensor DHT11 influirá significativamente en el éxito o el fallo del sensor. Después de varias pruebas, encontramos el valor correcto de acuerdo con el reloj de la frecuencia del C2. La siguiente línea de código hace la prueba. El valor de 2900 nos dio una precisión del 100%

en cada lectura del DHT11, pero hay un rango de valores (2900-3300) con los que puedes experimentar si lo deseas:

```
<for i in range(0,2900):>
```

Posteriormente, en algún punto al final del código, extraemos los valores de la humedad con la temperatura, y sólo después probaremos el sistema con las condiciones apropiadas dentro de la bodega antes de activar el mensaje SMS, si fuera necesario:

```
< #Checking here if the condi-
tions in the Wine Cellar is met
   if (int(Humidity)<50) or
((int(Temperature)<10) or
(int(Temperature)>14)):
      print ("Alert!!! Tem-
perature and/or Humidity out of
range!")
       sent_SMS()
       for i in range (0,10):
           #disable LED
         odroid.
digitalWrite(LEDpin, 0)
          # wait 1 second
          time.sleep(1)
          #enable LED
          odroid.
digitalWrite(LEDpin, 1)
            time.sleep(1)
   #cleanup
   odroid.pinMode(LEDpin, 1)
   #Print data
   print RCtime(5)  # Read RC
timing using pin #18
   #Check the lighting conditions
in your wine cellar
   if  (RCtime(5)<2500):
        print ("Alert!!! Light in
the wine cellar!")
        send_SMS()
  for i in range (0,10):
           #disable LED
         odroid.
digitalWrite(LEDpin, 0)
          # wait 1 second
          time.sleep(1)
          #enable LED
          odroid.
```

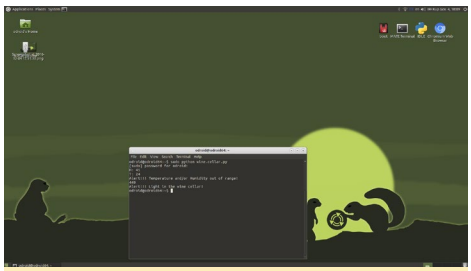


Figura 4 – Vistazo al código de python que se ejecuta en el dispositivo para avisarnos

```
digitalWrite(LEDpin, 1)
            time.sleep(1)
   #cleanup
   odroid.pinMode(LEDpin, 1)
>
```

Observa también cómo parpadea el LED cada vez que se envía un mensaje SMS de alerta, avisando al usuario del correcto funcionamiento del dispositivo IoT. Sin embargo, necesita que el usuario esté presente.

Juntándolo todo

Vamos a montarlo todo en Python. Primero necesitamos importar los módulos:

```
#!/usr/bin/python

import wiringpi2 as odroid
import time
import sys
```

A continuación, definiremos los pines como LEDpin= 4 (pin físico 16) y DHT11pin=7 (pin físico 7):

```
LEDpin=4
DHT11pin=7
```

Despues, configuraremos el módulo WiringPi de acuerdo con la guía de Hardkernel:

```
odroid.wiringPiSetup()
```

Y ahora vamos a configurar el LED :

```
#Set the operational LED
odroid.pinMode(LEDpin,1)
odroid.digitalWrite(LEDpin,1)
```

Interconectamos la fotorresistencia con la función RCtime (RCpin):

```
def RCtime(RCpin):
    reading = 0
    odroid.pinMode(RCpin,1)
    odroid.digitalWrite(RCpin,0)
    time.sleep(0.1)
    odroid.pinMode(RCpin,0)
    # This takes about 1 milli-
second per loop cycle
    while (odroid.
digitalRead(RCpin) == 0):
          reading += 1
    return reading
```

Definimos las variables necesarias para controlar el sensor DHT11:

```
def bin2dec(string_num):
    return str(int(string_num,
2))

data = []
effectiveData = []
bits_min=999;
bits_max=0;
HumidityBit = ""
TemperatureBit = ""
crc = ""
crc_OK = False;
Humidity = 0
Temperature = 0
```

Conectaremos el DHT11 con el C2 a través de GPIO (wiringPi):

```
def pullData():
#{{{ Pull data from GPIO.odroid
    global data
    global effectiveData
    global pin

    data = []
    effectiveData = []

    odroid.pinMode(DHT11pin,1)
    odroid.
digitalWrite(DHT11pin,1)
    time.sleep(0.025)
    odroid.
digitalWrite(DHT11pin,0)
    time.sleep(0.14)
```

```
    odroid.pinMode(DHT11pin,0)
    odroid.
pullUpDnControl(DHT11pin,2)

    for i in range(0,2900):
        data.append(odroid.
digitalRead(DHT11pin))

    """
    for i in range(0,len(data)):
        print "%d" % data[i],
    print
    """

#}}}
```

Esta sección busca y analiza datos del DHT11. Este código ha sido cogido de Github (http://bit.ly/2gAaUfK) como ya hemos comentado:

```
def analyzeData():
#{{{ Analyze data

#{{{ Add HI (2x8)x3 bits to array

    seek=0;
    bits_min=9999;
    bits_max=0;

    global HumidityBit
    global TemperatureBit
    global crc
    global Humidity
    global Temperature

    HumidityBit = ""
    TemperatureBit = ""
    crc = ""

    """
    Snip off the first bit - it
simply says "Hello, I got your
request, will send you tempera-
ture and humidity information
along with checksum shortly"
    """
    while(seek < len(data) and
data[seek] == 0):
        seek+=1;
```

```
    while(seek < len(data) and
data[seek] == 1):
         seek+=1;

    """
    Extract all HIGH bits'
blocks.  Add each block as sepa-
rate item in data[]
    """
    for i in range(0, 40):

        buffer = "";

        while(seek < len(data)
and data[seek] == 0):
            seek+=1;


        while(seek < len(data)
and data[seek] == 1):
            seek+=1;
            buffer += "1";

        """
        Find the longest and the
shortest block of HIGHs.  Aver-
age of those two will distinct
whether block represents '0'
(shorter than avg) or '1' (longer
than avg)
        """

        if (len(buffer) < bits_
min):
            bits_min =
len(buffer)

        if (len(buffer) > bits_
max):
            bits_max =
len(buffer)

        effectiveData.
append(buffer);
        #print "%s " % buffer

#}}}
```

Ahora vamos a configurar el servicio Twilio:

```
def sent_SMS():
    from twilio.rest import
TwilioRestClient

    account_sid = "xxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxx" # Your
Account SID from www.twilio.com/
console
    auth_token  = "xxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxx"  # Your Auth
Token from www.twilio.com/console

    client =
TwilioRestClient(account_sid,
auth_token)
    message = client.messages.
create(body="Alert!!! The condi-
tions on the wine cellar is out
of range!",
    to="+xxxxxxxxxx",    # Re-
place with your phone number
    from_="+xxxxxxxxxx")   # Re-
place with your Twilio number

    print(message.sid)
```

Y ahora la parte más crítica de este proyecto: comprobaremos las condiciones ideales de la bodega, y si esas condiciones no se cumplen, activaremos el envío del SMS:

```
#{{{ Print data
def printData():
   global Humidity
   global Temperature

   print "H: "+Humidity
   print "T: "+Temperature
   #Checking here if the condi-
tions in the Wine Cellar is met
   if (int(Humidity)<50) or
((int(Temperature)<10) or
(int(Temperature)>14)):
      print ("Alert!!! Tem-
perature and/or Humidity out of
range!")
       sent_SMS()
       for i in range (0,10):
           #disable LED
         odroid.
digitalWrite(LEDpin, 0)
```

```
            # wait 1 second
            time.sleep(1)
            #enable LED
            odroid.
digitalWrite(LEDpin, 1)
             time.sleep(1)
    #cleanup
    odroid.pinMode(LEDpin, 1)
    #Print data
    print RCtime(5)   # Read RC
timing using pin #18
    #Check the lighting conditions
in your wine cellar
    if   (RCtime(5)<2500):
          print ("Alert!!! Light in
the wine cellar!")
          send_SMS()
          for i in range (0,10):
             #disable LED
            odroid.
digitalWrite(LEDpin, 0)
            # wait 1 second
            time.sleep(1)
            #enable LED
            odroid.
digitalWrite(LEDpin, 1)
              time.sleep(1)
    #cleanup
    odroid.pinMode(LEDpin, 1)
#}}}
```

Finalmente, hacemos una especie de chequeo de validación:

```
#{{{ Main loop

while (not crc_OK):
     pullData();
     analyzeData();
     if (isDataValid()):
          crc_OK=True;
          print "\r",
          printData();
     else:
          sys.stderr.write(".")
          time.sleep(2);
#}}}
```

Testeando y ejecutando el código

Ejecutamos el código completo como prueba final:

```
$ sudo python wine.cellar.py
```

Debería funcionar a las mil maravillas. El programa muestra en pantalla la temperatura y la humedad, e informa de la presencia o ausencia de iluminación en la bodega. También advierte al usuario a través de un SMS enviado a su teléfono si las condiciones ambientales correctas no se están cumpliendo. Ten en cuenta que cuando se envían mensajes, el LED parpadea como indicación del proceso de "alerta". A continuación, vamos a automatizarlo y haremos que se ejecute cada 3 horas. Para esta tarea, utilizaremos la utilidad cron.

¿Qué es eso de cron? Define trabajos que se utilizan para programar tareas y scripts, como deftags, copias de seguridad y alarmas. Utilizamos la utilidad "cron" anteriormente cuando diseñamos el proyecto IoT del sistema de notificación de Gmail en la edición de octubre de ODROID Magazine (http://bit.ly/2dwqXJ7). Si necesita más información sobre cron, consulta http://bit.ly/2bTmNaN. Para activar cron, debemos ejecutar el comando crontab que nos proporciona una lista de tareas:

```
$ sudo crontab -e
```

Probablemente esté vacío. Elije cualquier editor de texto, como vim y agrega la siguiente línea de código al final de la lista de tareas programadas:

```
* */3 * * * sudo python /home/
odroid/wine.celllar.py
```

Los cinco "asteriscos" ("* * * * *") especifican con qué frecuencia deseas que se ejecute la tarea. El primer asterisco controla los minutos. El segundo asterisco controla las horas, de modo que coloqué el símbolo "/3", ya que quería que esta tarea programada se ejecutara cada 12 horas. El tercero especifica el día del mes, el cuarto indica el mes y el quinto representa el día de la semana. Esos cuatro se dejaron intencionalmente en blanco sin ningún "/número" a parte de los asteriscos. Puedes experimentar con otras posibilidades. Al final de la tarea programada, está el comando que queremos que se ejecute automáticamente:

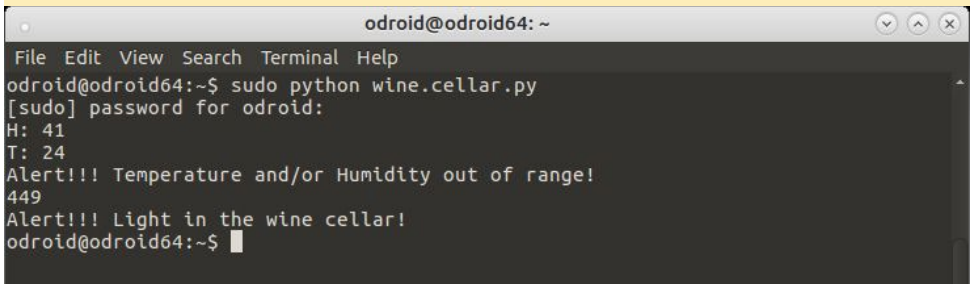
```
$ sudo python /home/odroid/wine.
cellar.py
```

Este comando ejecuta nuestro script y apunta a la ruta donde se encuentra, que en este caso es /home/odroid/. A continuación, guarda y cierra el editor. Ahora, espera y mira como la aplicación hace su trabajo. Recibirás un nuevo mensaje SMS cada vez que la temperatura, la humedad o la iluminación estén fuera de los valores preestablecidos. ¡Eso es todo, lo conseguimos!

Reflexiones finales

El código del sistema de conservación y notificación de la bodega se puede mejorar de muchas formas. Una es añadirle una función que informe de los valores de humedad, la temperatura y la iluminación con cada lectura y los envíe a la nube. Describimos cómo hacer esto usando la API de Twython en el artículo de la edición de septiembre de ODROID Magazine (http://bit.ly/2cIyp36). Consulta este artículo para saber cómo hacerlo. Mediante el uso de Twython, podemos enviar tweets a nuestra cuenta de Twitter con los valores de la humedad, la temperatura y la iluminación.

Figura 5 - Una captura de pantalla del programa que se ejecuta en un terminal



DELUGE

TU NUEVO CLIENTE BITTORRENT FAVORITO

por @synportack24

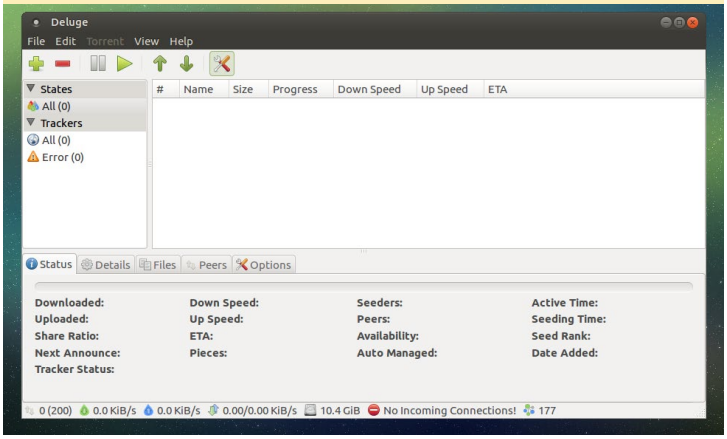
Hay una lista casi interminable de clientes BitTorrent disponibles para Linux, y todos dirán que el que utilizan es el mejor. Sin embargo, uno que seguro que oirás en muchas ocasiones es Deluge. Su gran popularidad no es ninguna sorpresa, ya que no sólo es un software muy liviano sino que al mismo tiempo ofrece una lista muy completa de características, además de ser multiplataforma y de arquitectura cruzada. Deluge cuenta con una interfaz muy agradable y simple de usar que gira en torno a una base libtorrent. Una de las características más útiles para nuestros ODROIDians es la posibilidad de poder tener el demonio torrent ejecutándose sobre un servidor y una conexión remota a éste a través de una interfaz web. Este artículo muestra algunas de las características de Deluge que más me gustan, así como la forma de utilizarlas.

Empecemos

Para juguetear con Deluge cuando escribí este artículo, yo utilice un ODROID-XU4, aunque esta guía debería funcionar con cualquier otra placa ODROID. La instalación es simple y fácil, y se puede hacer rápidamente usando el Gestor de Paquetes de Ubuntu con el siguiente comando:

```
$ sudo apt-get install deluge
```

Figura I - El cliente de escritorio Deluge



Una vez instalado, podemos echar un vistazo al funcionamiento y al uso de Deluge en una máquina local. Esto significa que interactuarás y ejecutaras Deluxe en el mismo ordenador. Al iniciar el programa se iniciará tanto el demonio Deluge como la interfaz gráfica del usuario.

Deluxe en un servidor

Tener Deluge ejecutándose en un servidor es una característica digna de mencionar y que merece tener su propia sección. Deluge es uno de los pocos clientes torrent que tiene la posibilidad de ejecutarse como un demonio e interactuar con él a través de una interfaz web. En realidad son dos los demonios que deben ejecutarse para conseguir esto: deluged y deluge-web. Generalmente los dos demonios no se instalan con el paquete básico de Deluge, pero se pueden instalar con los siguientes comandos:

```
$ sudo apt-get install deluge-webui
$ sudo apt-get install deluged
```

Una vez instalados, necesitamos configurar los scripts sysmd para deluged y deluge-web. Yo recomendaría crear un usuario y un grupo encargados de ejecutar los servicios de Deluxe.

```
$ sudo adduser --system \
  --gecos "Deluge Service" \
  --disabled-password --group \
  --home /var/lib/deluge deluge
```

Esto creará un nuevo grupo y un nuevo usuario llamado "deluxe". Es posible utilizar con el usuario por defecto "odroid", simplemente modificando los siguientes archivos de servicio para usar odroid como usuario. Una vez más, insisto en que por razones de seguridad no es la mejor opción. A continuación tenemos que crear dos archivos de servicio, uno para cada uno de los servicios. Crea un archivo llamado /etc/systemd/system/deluged.service y rellénalo con el siguiente contenido:

```
[Unit]
Description=Deluge Bittorrent
Client Daemon
After=network-online.target

[Service]
Type=simple
User=deluge
Group=deluge
#007 full access to the user and
members of the group.
#022 full access to the user run-
ning, and read access to others.
#000 full access to all accounts.
UMask=007

ExecStart=/usr/bin/deluged -d

Restart=on-failure

# Configures the time to wait
before service is stopped force-
fully.
TimeoutStopSec=300

[Install]
WantedBy=multi-user.target
```

A continuación, necesitamos crear un archivo para webui llamado /etc/systemd/system / delugeweb.service. El contenido del archivo se muestra a continuación:

```
[Unit]
Description=Deluge Bittorrent
Client Web Interface
After=network-online.target

[Service]
Type=simple

User=deluge
Group=deluge
UMask=027

ExecStart=/usr/bin/deluge-web

Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Ahora el servicio puede ser configurado para ejecutarse en el arranque:

```
$ systemctl enable /etc/systemd/
system/deluged.service
$ systemctl start deluged
$ systemctl status deluged

$ systemctl enable /etc/systemd/
system/deluge-web.service
$ systemctl start deluge-web
$ systemctl status deluge-web
```

Con ambos servicios en funcionamiento, podemos intentar conectarnos a él desde un navegador web. Por defecto, la interfaz web se encuentra en el puerto 8112. Esto significa que una vez que todo esté ejecutándose, podemos conectarnos desde un navegador web escribiendo http://<IP-servidor>: 8112.

La primera vez que inicie sesión en

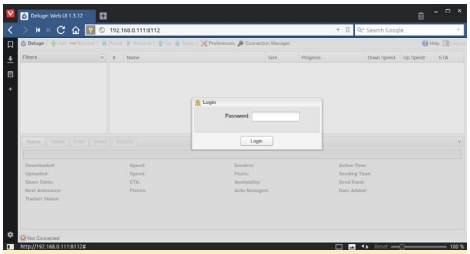


Figura 2 - Acceso a la interfaz de usuario web

el sistema, aparecerá una ventana emergente para introducir una contraseña. La contraseña por defecto es “deluge”, seguida de la opción para cambiarla. Una vez actualizada la contraseña, tendrás el control total sobre tu cliente Deluge.

Plugins

Más allá de las funciones ya incluidas y disponibles, Deluge tiene una amplia lista de plugins adicionales que se pueden instalar. Los plugins de terceros, como los renombradores de lotes y los programadores de tareas más avanzados se pueden descargar desde http://bit.ly/2igFfBC.

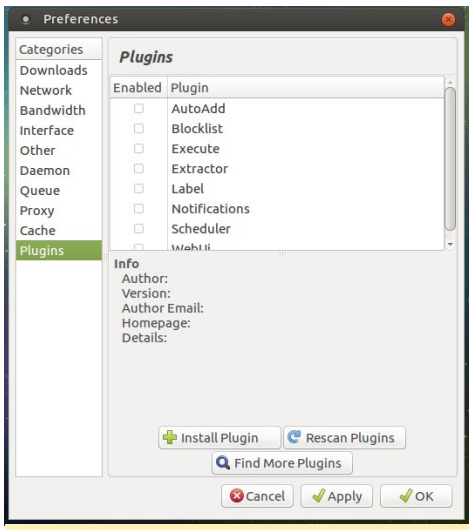


Figura 3 - Menú del plugins

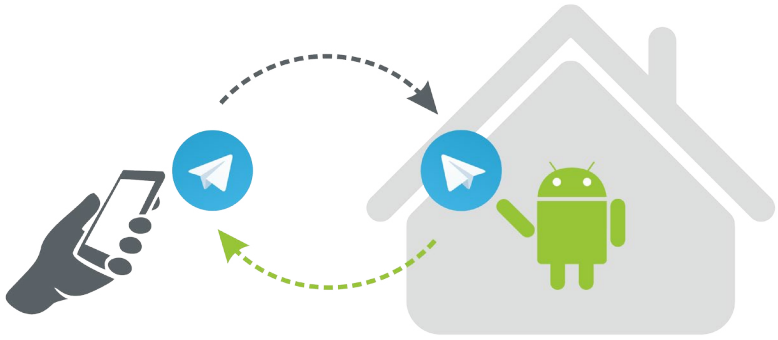
Conclusión

Deluge es un gran cliente de torrents de código abierto que tiene una gran cantidad de características muy útiles. Si tienes un ODROID adicional o estás buscando una buena manera de sacarle partido a tu actual servidor ODROID, la interfaz de usuario web de Deluge es una excelente forma de crear un seedbox rápido y fácil.

TELEGRAM CHATBOT

DOMOTICA AVANZADA

por Max Volkov



Utilizar un Chat bot de mensajería instantánea para la interfaz de usuario de un proyecto de domótica, en algunos casos, tiene sus ventajas frente a un servidor web. Por ejemplo, nuestra casa automatizada tiene acceso a Internet a través de un módem 3G. Si pretendemos utilizar un servidor web visible externamente, necesitaríamos una dirección IP pública. Esto no siempre es posible y podemos incurrir en gastos adicionales. Uno no tendría que enfrentarse a estos problemas cuando utiliza un servicio de mensajería como Telegram Chatbot.

Telegram es una aplicación de mensajería que ha ido ganando popularidad gracias a diversas cualidades como la velocidad, confiabilidad, privacidad y flexibilidad. Sin embargo, lo que nos interesa es otra característica: una API abierta para crear chatbots

Aparte del hecho de que los chatbots son oficialmente compatibles en Telegram, existe otra razón para usar este sistema de mensajería. Puedes crear un teclado virtual personalizado que proporcione una comodidad sin precedentes para que el usuario interactúe con el bot. En lugar de introducir manualmente un comando, el usuario puede tocar un botón de la app cliente Telegram.

En este tutorial, aprenderemos cómo configurar y usar Telegram chatbot en Linux para una aplicación de ejemplo muy simple que imita una casa inteligente. Esta aplicación es capaz de encender dos LED: verde y rojo, de acuerdo con la entrada del usuario.

Configuración inicial

Suponemos que tienes Python 2.7 y pip (Python Package Index) ya instalado en tu dispositivo ODROID. También es necesario instalar un paquete de Python que utiliza esta aplicación - pyTelegramBotAPI, que es una implementación de Python para la API Bot de Telegram.

```
$ pip install pyTelegramBotAPI
```

Ahora necesitas crear una cuenta Telegram, si aún no la tienes. Instala la app cliente de Telegram en tu teléfono inteligente, o descárgala desde telegram.org e instálala en tu PC. Es fácil crear una cuenta desde la aplicación cliente.

A continuación, debes crear tu chatbot. Se hace usando BotFather, que es un bot para crear bots. Para hacer esto en la app cliente, busca BotFather. Tras localizar este contacto, haga clic en el enlace Start debajo de éste. Recibirás un mensaje largo que describe todos los comandos disponibles para administrar bots. Todos los comandos comienzan con una barra ("/"). Utiliza el comando /newbot para crear un nuevo bot. El BotFather te pedirá un nombre y un nombre de usuario y tras proporcionar esta información, se creará un token de autorización para tu nuevo bot. El nombre de tu bot aparecerá en los datos de contacto y en cualquier otro lugar.

Los nombres de usuario son nombres cortos, con una longitud de 5-32 caracteres y no son sensibles a mayúsculas y minúsculas, y sólo pueden incluir caracteres latinos, números y barra baja. El nombre de usuario de tu bot debe terminar en "bot", por ejeplo: "Tetris_bot" o "TetrisBot".

El token es una cadena que es necesaria para autorizar el bot y enviar solicitudes a la API Bot. Una vez que obtengas el token para tu bot, guárdalo en algún lugar para usarlo en tu app. En la app cliente de Telegram, busca el nombre de usuario que especificaste antes y agrégalo a la lista de contactos. Encontrarás más detalles sobre la gestión de bots en Telegram en http://bit.ly/2hlOyQK.

Ejecutar el script Pythont

Ahora que has instalado la aplicación cliente de Telegram y ha creado una cuenta de usuario de Telegram y un bot, estás listo para ejecutar la aplicación bot de Python en tu SBC ODROID y probarla. Guarda los fragmentos de código que aparecen a continuación en un archivo de texto sin formato y nómbralo con, por ejemplo, "chatbot.py". Ejecuta el script usando el comando:

```
$ python chatbot.py
```

A continuación se muestra el script Python, con algunas explicaciones. Primero, declaramos los módulos Python externos usados en el script

```
import telebot
```

```
from telebot import types
```

Para crear un objeto bot de Telegram, debes colocar el token del bot que hemos mencionado en la sección de configuración inicial.

```
bot = telebot.TeleBot("my_Telegram_bot_token_here")
our_chat_id=0
```

El our_chat_id del token es una variable global que contiene el identificador de chat. El significado de ésta se explica a continuación en la descripción de la función send_welcome.

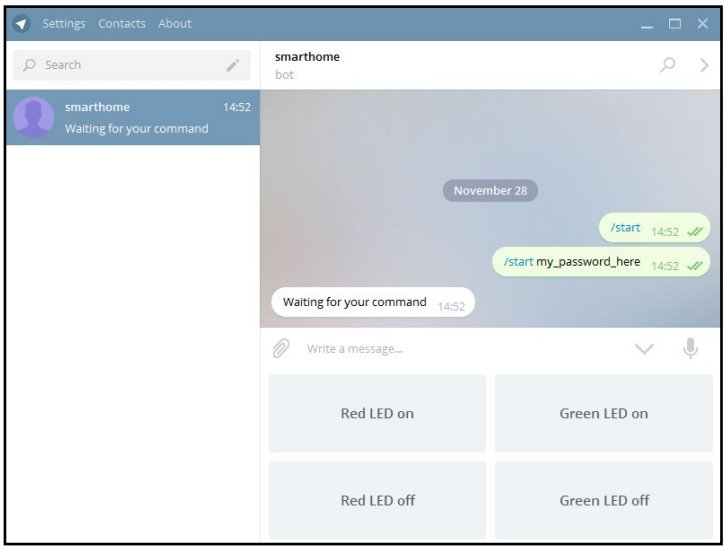
```
def extract_unique_code(text):
    # Extracts the unique_code from the sent /start
command.
    return text.split()[1] if len(text.split()) > 1
else None

@bot.message_handler(commands=['start'])
def send_welcome(message):
    global our_chat_id
    unique_code = extract_unique_code(message.text)
    if unique_code: # if the '/start' command con-
tains a unique_code
        if unique_code=="my_password_here":
            our_chat_id=message.chat.id
            send_LED_ctrl_keyboard(our_chat_id)
        else:
            reply = "Sorry, don't know who are
you..."

def send_LED_ctrl_keyboard(chat_id):
    markup = types.ReplyKeyboardMarkup(row_width=2)
    itembtn1 = types.KeyboardButton('Red LED on')
    itembtn2 = types.KeyboardButton('Green LED on')
    itembtn3 = types.KeyboardButton('Red LED off')
    itembtn4 = types.KeyboardButton('Green LED off')
    markup.add(itembtn1, itembtn2, itembtn3, itemb-
tn4)
    bot.send_message(our_chat_id, "Waiting for your
command", reply_markup=markup)
```

Hemos definido tres funciones. La primera simplemente extrae la contraseña de usuario de la línea de comandos /start. Según la guía oficial de bot de Telegram, todos los desarrolladores de bot deben soportar algunos comandos básicos: /start, /help, /settings.

El comando /start inicia una interacción con el usuario, por ejemplo envía un mensaje de bienvenida. En nuestro script, el comando /start soporta la carga, lo cual significa que después de la palabra clave /start, debes especificar tu contraseña de usuario para que el script pueda identificarte.

Esto se hace al comienzo de la segunda función send_ welcome, que se activa cuando el chatbot recibe el comando /start. Si la contraseña de usuario especificada tras la palabra clave / start es idéntica a la de la cadena especificada en lugar de "my_ password_here" en el script, la función send_welcome hace dos cosas:

1. guarda el id de chat del mensaje actual en la variable global our_chat_id que se usará despues en el script;
2. crea un teclado personalizado con 4 botones y lo muestra al usuario autorizado junto con el mensaje " Waiting for your command" llamando a la función send_ LED_ctrl_keyboard.



Por el contrario, si la contraseña no coincide, la función envía un mensaje de respuesta indicando que el usuario es desconocido. Vamos a analizar con más detalle que es exactamente el ID del chat y como se forma un teclado virtual.

El ID del Chat es un valor numérico que el servicio de mensajería Telegram asignó a la sesión de comunicación con el chatbot. Teóricamente, cualquier persona desde Telegram puede entrar en contacto con nuestro bot. Por lo tanto, tiene sentido utilizarlo como identificador de sesión, en el cual el usuario especificó la contraseña correcta. Este ID de chat se almacena en la variable our_chat_id y será utilizado en comunicaciones posteriores. Es más, este ID de chat puede almacenarse en un archivo o en una base de datos y utilizarse en próximas sesiones. En este caso, no es necesario introducir la contraseña cada vez que se ejecute el script.

En la función send_LED_ctrl_keyboard, montamos un teclado virtual personalizado con 4 botones por comodidad. Tocar un botón tiene el mismo efecto que escribir texto con un teclado y enviarlo al bot.

Un botón es definido llamando al método types.Keyboard-

CHROME DEATH

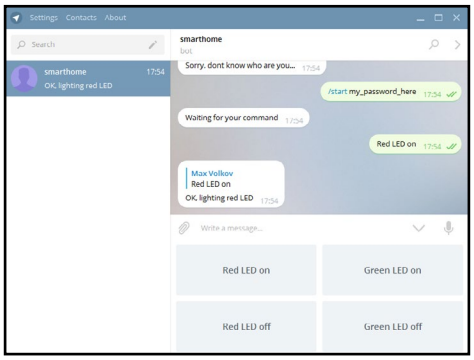
UN JUEGO DE ACCION DE TEMATICA CYBERPUNK CON EL QUE TENDRAS UN CONSTANTE BOMBARDEO DE ADRENALINA

por Bruno Doiche

Sobrevive en una carrera interminable dentro de una ciudad cyberpunk, donde siempre corres por tu vida. Con agentes bloqueando tu camino, tus reflejos son tu mejor herramienta para seguir adelante todo el tiempo que te sea posible. Basado en ese sentimiento guay de las películas VHS de los años 80 y con una banda sonora que es un espectáculo en sí misma, Chrome Death es un thriller que hará que tu colección de juegos Android sea aún más brillante. ¡Con huevos de Pascua ocultos aquí y allá, sin duda perderás un par de vidas disfrutando de este frenético juego!

https://play.google.com/store/apps/details?id=com.newmark.chromedeathandroid&hl=en

Este juego con desafíos muy sutiles pondrá a prueba tus reflejos constantemente



Button. Un argumento es una cadena de texto que aparecerá sobre el botón y se enviará al chatbot tocando ese botón. Otra opción para realizar la misma acción es simplemente escribir un mensaje, por ejemplo, " Green LED off " en tu teclado y hacer clic en "Send". Aunque ¿Por qué molestarse en escribir si existe un botón?"

```
@bot.message_handler(func=lambda
message: True)
def echo_all(message):
    global our_chat_id
    if message.chat.id==our_chat_
id:
        if message.text == 'Red
LED on':
            bot.reply_to(message,
'OK, lighting red LED')
            return
        if message.text == 'Green
LED on':
            bot.reply_to(message,
'OK, lighting green LED')
            return
        if message.text == 'Red
LED off':
            bot.reply_to(message,
'OK, putting out red LED')
            return
        if message.text == 'Green
LED off':
            bot.reply_to(message,
'OK, putting out green LED')
            return
        bot.reply_to(message,
'Not understood')
    else:
        bot.reply_to(message,
"Sorry, don't know who are you…")
```

La función echo_all es un manipulador de mensajes, y se activa cada vez que nuestro chat bot recibe un mensaje excepto con el comando /start, que es gestionado por la función send_welcome. En primer lugar, se comprueba el ID de chat del mensaje entrante. Si es "nuestro" chat (con un usuario autorizado), entonces se continúa, en caso contrario muestra un mensaje de error y se sale. Si pasamos el chequeo, analizamos el contenido del mensaje. El texto cuyo nombre habla por sí mismo. Si es reconocido como uno de los comandos válidos, se lleva a cabo la acción correspondiente. En la función echo_all anterior, sólo está limitada por el mensaje de respuesta al usuario.

```
bot.polling()
```

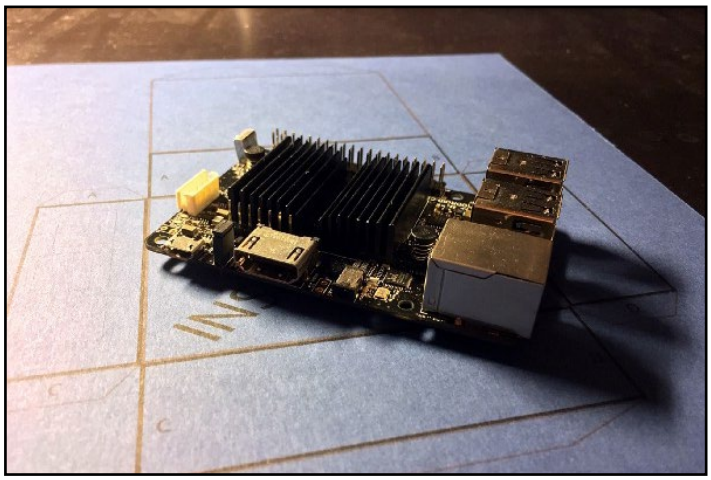
Por último, pero no menos importante, esta línea de código es el método de bloqueo de llamadas llamado bot. polling que hace que nuestro chatbot siempre esté esperando un mensaje entrante.

Notas

En este artículo, hemos visto una versión muy simple de la aplicación Telegram chatbot. Se puede mejorar bastante añadiendo diversas opciones. Por ejemplo, es posible crear un "sistema de teclado" multinivel con una configuración de teclado diferente para cada subdivisión de cada nivel. Además, es posible adjuntar un teclado a un mensaje, denominado teclado en línea. Esta y otras características están explicadas con detalle en la sección correspondiente del sitio web de Telegram en http://telegram.org.

CARCASA DE PAPEL ODROID-C1/C2

por @thekillercarrot



No hace mucho, compré una estupenda placa ODROIDC2. Aunque cuenta con una comunidad más pequeña que mi Raspberry Pi ya retirada, es un producto de muy alta calidad y fiabilidad. Decidí empezar a instalarle todas las cosas que necesitaba para hacerla totalmente funcional. Cuando tenía todo el software completamente configurado, sólo faltaba una cosa: una carcasa. Puesto que aún no dispongo de una impresora 3D, decidí ponerme mano a la obra y hacer mi propia carcasa de papel.

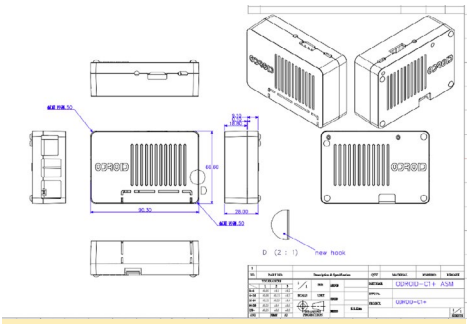


Figura 1 – Las medidas para la carcasa de papel

Tras pasar algún tiempo buscando por internet las medidas exactas del ODROIDC2, descargue los archivos de modelado 3D de otras carcasas para el ODROID-C2 con el fin de usar sus mediciones. Agradezco su trabajo a la persona que hizo el archivo que se muestra en la Figura 1, porque fue el único archivo donde logré encontrar algunos medidas que utilicé para crear mi carcasa de papel.

Tras unas horas con Photoshop, puedo decir que tenía una carcasa bastante decente, la cual se muestra en las figuras 2 y 3. ¡No es un Picasso, pero tiene un buen aspecto y se le puede poner algunas pegatinas!

Te recomiendo imprimir el archivo PDF en el papel de cartulina más grueso que tu impresora pueda soportar.

Otra posibilidad es imprimir en papel normal y pegar éste en una cartulina gruesa

Para hacer tu propia carcasa de papel, descarga el archivo gratuito de http://bit.ly/2ifvFyG. Si tiene comentarios, preguntas o sugerencias, visita el artículo original en http://bit.ly/2ipdVhu.

PIXEL DODGERS

SUJETA TUS DEDOS Y ESQUIVA BOLAS DE FUEGO

por Bruno Doiche

Actúa rápido para esquivar tantas explosiones de píxeles como puedas antes de empezar a resoplar ! En Pixel Dodgers, necesitas realizar un seguimiento de dónde te encuentras y a dónde quieres ir para escapar de las molestas bolas de fuego. Este sistema de juego simple e intuitivo te mantendrá esquivando bolas todo el día. ¡Lo mejor es usarlo con un teclado y sentir que estás jugando algún tipo de juego al estilo MSX!

https://play.google.com/store/apps/details?id=com.bigbluebubble.pixeldodgers

¿Esquivar bolas de fuego en una plataforma suspendida en el antiguo Egipto con trampas en el suelo? ¡Cuenta conmigo!

MANUAL ODROID-C2

UNA GUIA PARA TODOS LOS NIVELES

por Rob Roy (@robroy)

El manual de usuario oficial para el ODROID-C2 ha sido publicado recientemente en el sitio web de ODROID Magazine, y está disponible para su descarga directa en http://bit.ly/2hM1FH6, a través de los foros en http://bit.ly/2i5F7nM y En Google Play Store en http://bit.ly/2iCuupA.

El ODROID-C2 es uno de los más potentes y económicos ordenadores de placa reducida de 64 bit que existen, además de ser un dispositivo extremadamente versátil. Con un rápido procesador Amlogic quad-core, una avanzada GPU Malí y Ethernet Gigabit, se puede utilizar como un sistema de cine en casa, un ordenador de uso general para navegar por internet, para ejecutar juegos y consultar redes sociales, como herramienta de trabajo para el colegió o la oficina, como prototipo para realizar pequeños ajustes y modificaciones de hardware, como controlador para proyectos de domótica, como estación de trabajo para programar, entre mucha otras aplicaciones.

Algunos de los modernos sistemas operativos que se pueden ejecutar en el ODROID-C2 son Ubuntu, Android y ARCH linux, con miles de paquetes de software de código abierto totalmente gratis. El ODROID-C2 es un dispositivo ARM, la arquitectura más utilizada en dispositivos móviles y en la informática integrada. El pequeño tamaño de su procesador ARM, su reducida complejidad y su bajo consumo de energía hacen que sea perfecto para desarrollar pequeños dispositivos que podemos llevar encima.

Portada del Manual de Usuario del ODROID-C2

ESTACION ARCADE PORTATIL

por @LtBenjamin

He creado una estación arcade portátil con una pantalla táctil de 10" utilizando los componentes electróncios que enumero a continuación. La unidad permite ejecutar perfectamente los juegos de Atari, Super Nintendo, Nintendo 64 y Nintendo DS. Los juegos de Dreamcast y PS funcionan realizando algunos ajustes, y es posible ejecutar unos cuantos juegos de PSP decentemente.

Tiene una autonomía de casi 7 horas y necesita entre 10 y 12 horas para recargarse con ayuda de un cargador de 2.5A. Para comentarios, preguntas y sugerencias, visita el post original en http://bit.ly/2ia2eh7.

Un maletín portátil con un aspecto al estilo James Bond

Compoenetes de la estación

ODROID-C2 con carcasa
Módulo Bluetooth 2
Módulo WiFi 0
Altavoz y micrófono Bluetooth Dknight Magicbox 2
Sistema de alimentación RAV power 16750 con un total de 4.5A – puertos 5V/2.1A y 5V/2.4A
Pantalla táctil Waveshare de 10.1 pulgadas 1280x800 (5V 2.5A)
EMMC de 16 GB (Android)
Tarjeta microSDHC Samsung 48mb/seg de 16 GB
Receptor inalámbrico Xbox 360
Soporte multi ángulo Anker
Maletín de fichas de póker

Todo está muy bien organizado dentro del maletín

La pantalla táctil encaja encima del resto de componentes

Los mandos de la Xbox 360 funcionan con todos los emuladores

Turok se ejecuta muy bien en la estación arcade portátil

LOS ODROID ALREDEDOR DEL MUNDO

EL ALCANCE INTERNACIONAL DE LOS POPULARES ORDENADORES DE PLACA REDUCIDA DE HARDKERNEL

por Rob Roy (@robroy)

Hardkernel lanzó su primer ordenador ODROID en 2009 (http://bit.ly/1Gx5Lr1), y desde entonces se ha convertido en un líder en lo que respecta al desarrollo de ordenadores de placa reducida con la reciente introducción de ODROID-C2 y ODROID-XU4. Continuamente lanza placas de desarrollo punteras y su catálogo de productos es muy extenso:

Productos Actuales

ODROID-C2
ODROID-XU4
ODROID-C1+
ODROID-C0

Pantallas

ODROID-VU8C
ODROID-VU5
ODROID-VU7 Plus
ODROID-VU7
3.5inch Touchscreen Sh
C1 3.2inch TFT+Touchsc
16x2 LCD + IO Shield
LED Matrix Shield
ODROID-SHOW2

Kits de Desarrollo

C Tinkering Kit
USB-UART Module Kit
Xprotolab Plain

Placas Adicionales

CloudShell for XU4
Expansion Board
USB IO Board
XU4 Shifter Shield
Universal Motion Joypad
USB3.0 to SATA Bridge
U3 IO Shield
U3 Shield Tinkering Ki

Sensores

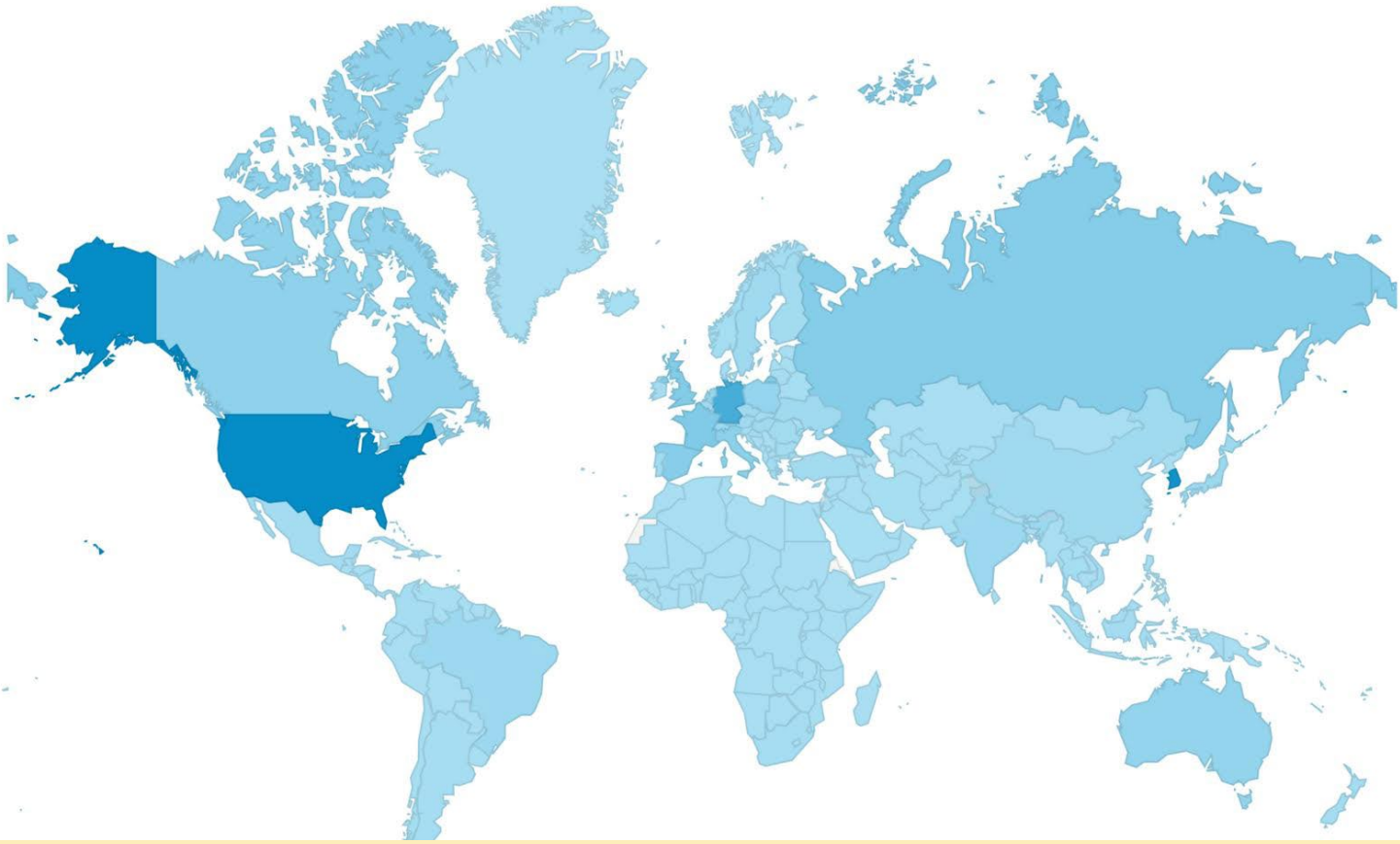
myAHRS+
Weather Board 2

Legado de Productos

ODROID
ODROID-7
ODROID-A4
ODROID-PC
ODROID-S
ODROID-T
ODROID-A
ODROID-U3
ODROID-U2
ODROID-X2
ODROID-E7
ODROID-Q2
ODROID-XU3 (+Lite)
ODROID-XU (+Lite)
ODROID-X
ODROID-C1
ODROID-Q
ODROID-XU+E
Smart Power
HiFi Shield for C2/C1+
ODROID-UPS
ODUINO One
UPS2 for U3
Weather Board
ODROID-VU
ODROID-Show
ODROID-W

Las Noticias de lo útiles y asequibles que son las placas ODROID se ha extendido por todo el mundo. Hardkernel ha recibido pedidos de casi 150 países:

Albania
Algeria
Andorra
Angola
Argentina
Armenia
Aruba
Australia
Austria
Azerbaijan
Bahrain
Bangladesh
Barbados
Belarus
Belgium
Bermuda
Bolivia
Bosnia-Herzegovina
Brazil
British Virgin Islands
Bulgaria
Burkina Faso
Cambodia
Canada
Cape Verde
Chile
China
Colombia



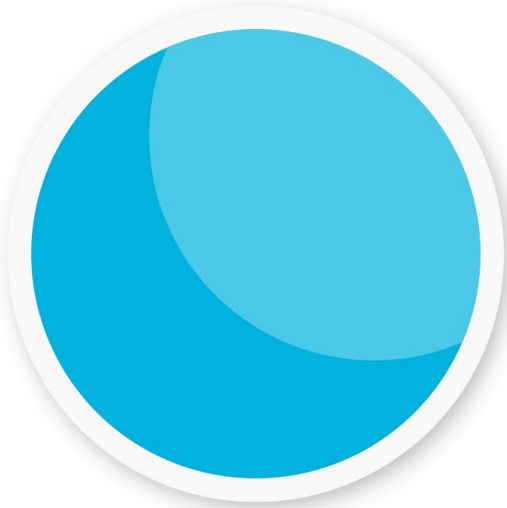
Hardkernel ha vendido sus ordenadores de placa reducida en casi 150 países, y su sitio web recibe visitas procedentes de más de 200 países

Costa Rica
Croatia
Curacao
Cyprus
Czech Republic
Denmark
Dominican Republic
Ecuador
Egypt
El Salvador
Estonia
Faroe Islands
Finland
France
French Guiana
French Polynesia
Gambia
Georgia
Germany
Ghana
Gibraltar
Greece
Greenland
Guadeloupe
Guatemala
Honduras
Hong Kong
Hungary
Iceland
India
Indonesia
Iran
Ireland
Isle of Man
Israel
Italy
Ivory Coast
Jamaica
Japan
Jordan
Kazakhstan
Kenya
Kosovo
Kuwait
Kyrgyzstan
Laos
Latvia
Lebanon
Libya
Liechtenstein
Lithuania
Luxembourg
Macau
Macedonia
Madagascar
Malaysia
Maldives
Mali
Malta
Marshall Islands
Martinique
Mauritius
Mexico
Moldova
Mongolia
Montenegro
Morocco
Namibia
Nepal
Netherlands
Netherlands Antilles
New Caledonia
New Zealand
Nicaragua
Nigeria
Norway
Oman
Pakistan
Palau
Panama
Paraguay
Peru
Philippines
Poland
Portugal
Puerto Rico
Qatar
Reunion
Romania
Russia
Saudi Arabia
Senegal
Serbia
Singapore
Slovakia
Slovenia
South Africa
South Korea
Spain
Sri Lanka
Sweden
Switzerland
Syria
Taiwan
Tajikistan
Tanzania
Thailand
Tunisia
Turkey
U.S. Virgin Islands
U.S.A
Uganda
Ukraine
UAE
United Kingdom
Uruguay
Uzbekistan
Venezuela
Vietnam
Zimbabwe

CLIPGRAB

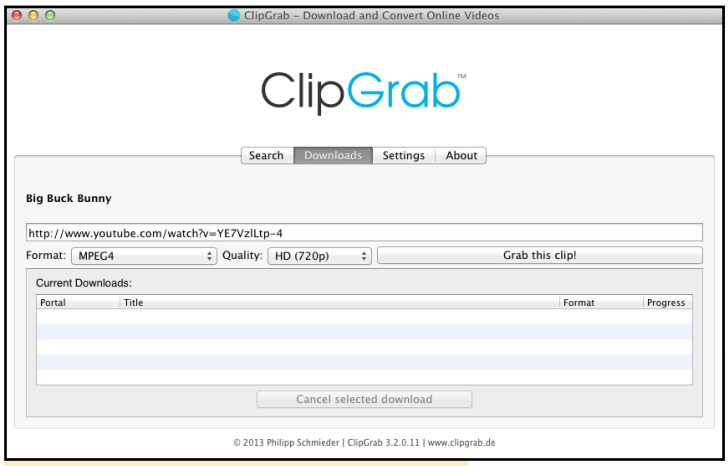
COMO DESCARGAR TUS VIDEOS FAVORITOS PARA VERLOS FUERA DE LINEA

por Tobias Schaaf (@meveric)

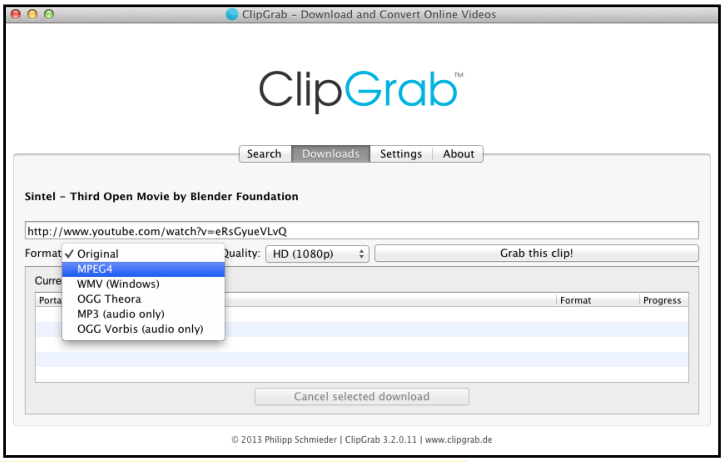


ClibGrab es un software que permite descargar y convertir videos de sitios como YouTube, DailyMotion y otros. Puedes elegir en qué formato deseas convertir los videos, como por ejemplo MPEG4, MP3 u OGG. Utiliza ffmpeg para convertir los archivos.

Navegando por los archivos disponibles



Seleccionando un archivo para descargar



Configurando las opciones de descarga

Sitios admitidos

YouTube
Vimeo
Dailymotion
metacafe.com
youku.com
myspass.de
myvideo.de
clipfish.de
collegehumor.com
Otros sitios deberían funcionar igualmente

Instalación

Puedes descargar el programa desde mi repositorio, el cual se describe en uno de mis anteriores artículos de ODROID Magazine, http://bit.ly/2icmAUQ. La versión armhf puede descargarse de la lista de paquetes principales y la versión arm64 de la lista de paquetes jessie con el siguiente comando:

```
$ apt-get install clipgrab
```

Para comentarios, preguntas y sugerencias, visite el post original en http://bit.ly/2hv9Awo.

PROTECTOR DE PANTALLA DE KODI

CONTROLA EL MONITOR DE TU TV COMPATIBLE CON CEC CON ESTA SENCILLA FUNCION

por @rooted

En este artículo, te voy a mostrar cómo apagar tu televisor cuando el protector de pantalla de Kodi se active y cómo encender el televisor cuando el protector de pantalla de Kodi este desactivado. Antes de empezar, asegúrate de que tu televisor es compatible con CEC.

Instalación

En primer lugar, dirígete a Kodi Settings -> Add-ons -> Install from repository -> Kodi Add-on repository -> Services, e instala el complemento " Kodi Callbacks".

A continuación, monta las particiones / y /system con privilegios de lectura/escritura usando una ventana Terminal:

```
$ su
$ mount -o remount,rw /
$ mount -o remount,rw /system
```

Añade la siguiente línea al archivo /init.odroidc2.rc:

```
chmod 666 /sys/class/amhdmitx/amhdmitx0/cec
```

Luego, descarga los archivos "cecon" y "cecoff" desde https://db.tt/ai1DNnFh, cópialos en / bin y hazlos ejecutables:

```
chmod +x /bin/cecoff
chmod +x /bin/cecon
```

Abre Kodi y entra en Kodi Settings -> Add-ons -> My add-ons -> Services -> Kodi Callbacks. Añade las siguientes tareas bajo la pestaña "Tasks":

```
Task I
Task = script
Script executable file = /bin/cecoff

Task 2
Task = script
Script executable file = /bin/cecon
```

En la pestaña "Events", agrega estos dos eventos:

```
Event I -> Choose event type -> on Screensaver Activated
Task = Task I

Event 2 -> Choose event type -> on Screensaver Deactivated
Task = Task 2
```

Finalmente, reinicia el sistema. Tu TV se apagará cuando se active el protector de pantalla. Puedes modificar el tiempo de espera ajustando el tiempo del protector de pantalla. Si estás utilizando un mando a distancia por infrarrojos, como el mando de Hardkernel, puede presionar un botón del mando para activar el televisor o simplemente volver a encender el televisor utilizando el mando del televisor. Para comentarios, preguntas y sugerencias, visita el post original en `http://bit.ly/2i5OWSz`.:

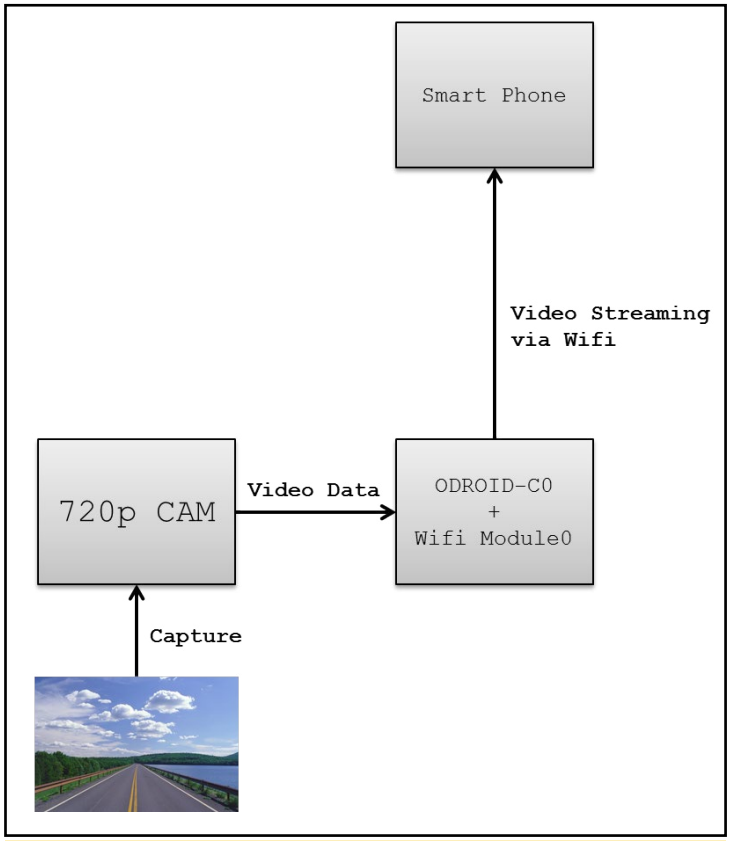
CAMARA DE VISION TRASERA

MANTENERTE SEGURO SOBRE TU BICICLETA

por Brian Kim

Cuando me mudé a mi nuevo apartamento, uno de mis amigos me dejó prestada su bicicleta de carretera. Una bicicleta de carretera es un tipo de bicicleta muy ligera que está diseñada para ir a altas velocidades por las carreteras asfaltadas. Un día, me lastime al caerme de la bicicleta, ya que contaba con poca experiencia en esto de montar en bicicletas de carretera. Sin embargo, llegue a aprender a montar más rápido en una bicicleta de carretera que en cualquier otro tipo de bicicleta, y la sensación de velocidad hacía que me dejara llevar. Como cabría de esperar, una de mis aficiones favoritas llegaría a ser precisamente esta, montar en bicicleta de carretera.

Depende de la legislación de cada país, pero en Corea del Sur, es legal andar en bicicleta por la carretera. Sin embargo, al compartir la carretera con los coches, necesito moverme con fluidez para que el tráfico normal se vea afectado lo menos posible. Es cuando montar en bicicleta a toda velocidad se convierte en algo realmente emocionante para mí. Puesto que comparto el asfalto con los coches y los camiones, siempre me ha preocupado que un coche me alcance por detrás. Me siento muy incómodo por el hecho de no poder ver lo que sucede detrás de mí. Así que decidí montar un sistema de cámara de visión trasera utilizando un ODROID-C0.



Esquema general del concepto de cámara de visión trasera

La cámara de visión trasera casera necesita un elevado ancho de banda para transmitir el video en tiempo real. He decidido que una buena estrategia sería utilizar el wifi como interfaz para este proyecto, puesto que una conexión inalámbrica nos proporciona una mayor flexibilidad en cuanto al espacio. Para evitar tener que hacer un dispositivo de visualización adicional, opté por usar mi smartphone como pantalla de visualización de la cámara. La cámara de visión trasera sería un PA (Punto de acceso) Wifi al cual se conectará mi teléfono. En este proyecto el PA Wifi utiliza un módulo ODROID Wifi 0 y un ODROID-C0. La figura 1 muestra un esquema general del proyecto. Una cámara de 720p captura la imagen trasera y luego el ODROID-C0 codifica la imagen de video y transmite los datos codificados a través del Wifi. Es cuando podemos ver nuestra vista trasera en nuestro teléfono después de conectarnos al servidor de video por streaming alojado en el ODROID-C0.

Estos son los componentes de hardware que necesitas para montar una cámara de visión trasera casera:

ODROID USB-CAM 720p
ODROID-C0
Módulo eMMC de 16GB C1+/C0 con Linux
Módulo Wifi 0
Carcasa ODROID-C2/C1+

Componentes de hardware para la cámara de visión trasera casera

3000mAh Battery
Doble Puerto USB-A hembra

He soldado un doble puerto USB-A hembra al ODROID-C0 como puedes observar en la Figura 2, ya que el ODROID-C0 no viene con puertos USB soldados. También necesitamos una cámara ODROID USB-CAM 720P, una batería de 3000mAh, un módulo WiFi 0, un ODROID-C0 y una carcasa ODROID-C2/C1+. En primer lugar, la cámara de visión trasera necesita tener configurado el software antes de montarla en la bicicleta. Las configuraciones del software las podemos dividir en dos partes: Configuración del PA Wifi y configuración del servidor de video por streaming

Configuración del punto de acceso WiFi

Hostapd es un demonio del espacio de usuario para puntos de acceso y servidores de autentificación. Es posible utilizarlo para crear un hotspot inalámbrico utilizando un ordenador Linux. Son necesarios algunos paquetes adicionales para hostapd. Puedes encontrar información sobre cómo configurar hostapd en la página wiki de Hardkernel para hostapd en http://bit.ly/2fJTr4h. Puedes instalar Hostapd con los siguientes comandos:

```
$ sudo apt-get update
$ sudo apt-get install libnl-3-dev \
  libnl-genl-3-dev libssl-dev hostapd
```

El teléfono se conecta a la cámara de visión trasera, que corresponde al “ssid” en el archivo de configuración tras configurar el modo PA. “Wpa_passphrase” es la contraseña necesaria para conectarse al PA. Los archivos de configuración de hostapd para el módulo Wifi 0 son los siguientes:

```
/etc/hostapd/hostapd.conf
interface=wlan0
driver=nl80211
ssid=ODROID_REARCAM
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=saferiding
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP

/etc/default/hostapd
DAEMON_CONF=”/etc/hostapd/hostapd.conf”
DAEMON_OPTS=”-B”
```

El ODROID-C0 debe asignar la dirección IP a los dispositivos conectados. Yo utilicé dnsmasq para asignar las direcciones IP dinámicas a los nodos conectados. Dnsmasq proporciona infraestructura de red para pequeñas redes tales como: DNS, DHCP, avisos de router y arranque en red.

```
$ sudo apt-get install --reinstall dnsmasq
$ mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig

The conf file for dnsmasq can be configured as shown.
/etc/dnsmasq.conf
domain-needed
bogus-priv
no-resolv
no-poll
server=/example.com/192.168.1.5
server=8.8.8.8
server=4.4.4.4
local=/example.com/
address=/doubleclick.net/127.0.0.1
no-hosts
addn-hosts=/etc/dnsmasq.d/hosts.conf
expand-hosts
domain=example.com
dhcp-range=192.168.1.20,192.168.1.50,72h
dhcp-range=tftp,192.168.1.250,192.168.1.254
dhcp-option=option:router,192.168.1.1
dhcp-option=option:ntp-server,192.168.1.5
dhcp-option=19,0 # ip-forwarding off
dhcp-option=44,192.168.1.5 # set netbios-over-TCP/IP
aka WINS
dhcp-option=45,192.168.1.5 # netbios datagram distri-
bution server
dhcp-option=46,8           # netbios node type
```

El ODROID-C0 debe estar configurado hacia el servidor, o puerta de enlace, dirección IP de "192.168.1.1". El visualizador de la cámara se conectará a esta dirección IP.

```
/etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto wlan0
iface wlan0 inet static
        address 192.168.1.1
        netmask 255.255.255.0
```

Configuración del servidor de streaming de la cámara en tiempo real

El paquete ffmpeg de Linux se suele utilizar para la codificación y transmisión de datos de vídeo, así que podemos utilizarlo para la transmisión de nuestra cámara 720p. Ffserver es capaz de transmitir muchos tipos de formatos de vídeo aunque, nosotros necesitamos reproducir vídeo en tiempo real para ver lo que sucede detrás de nosotros. La cámara ODROID 720p no sólo soporta datos de vídeo raw, sino que también puede codificar cada fotograma de vídeo como jpeg. Por lo tanto, el formato mjpeg (JPEG en varias partes) es una buena opción para reducir la sobrecarga de codificación de vídeo y el uso del ancho de banda de la red.

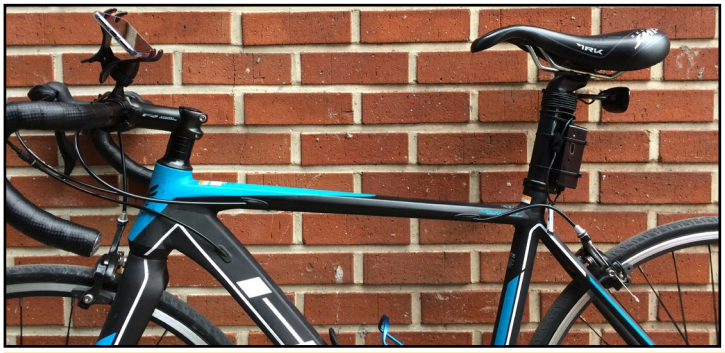
```
$ sudo apt-get install ffmpeg

/etc/ffserver.conf
HTTPPort 8090
HTTPBindAddress 0.0.0.0
MaxHTTPConnections 2000
MaxClients 1000
MaxBandwidth 5000

<Feed cam1.ffm>
File /tmp/cam1.ffm
FileMaxSize 100M
</Feed>

<Stream cam1.mjpg>
Feed cam1.ffm
Format mpjpeg
VideoCodec mjpeg
VideoFrameRate 20
VideoBitRate 4096
VideoSize 640x480
NoAudio
</Stream>

<Stream stat.html>
Format status
ACL allow localhost
ACL allow 192.168.0.0 192.168.255.255
</Stream>
```

Para iniciar el servidor de streaming automáticamente en el arranque, los comandos ffserver y ffmpeg se tiene que añadir al script de arranque /etc/rc.local. Ffmpeg codifica los datos de vídeo de la cámara 720p con la interfaz V4L2 (Video para Linux 2) y a continuación, envía los fotogramas de vídeo al servidor streaming. Ffserver, que es el servidor de streaming de video, recibe los fotogramas de video codificado y transmite el video en formato mjpeg al cliente conectado, que es un navegador web en el smartphone.

```
/etc/rc.local
ffserver -d -f /etc/ffserver.conf&
ffmpeg -f v4l2 -s 640x480 -r 20 -vcodec mjpeg -i /
dev/video0 http://localhost:8090/cam1.ffm

exit 0
```

Instalación de la cámara de visión trasera

Las configuraciones de software para la cámara de visión trasera se realiza tras finalizar la configuración del PA Wifi y la



Instalación de la cámara casera en la bicicleta de carretera

configuración del servidor de video por streaming. El siguiente paso es instalar la cámara en la bicicleta. Conecta la batería 3000mAh al ODROID-C0 e introdúcela dentro de la carcasa

del ODROID-C2/C1. Luego, monta la cámara 720p y la carcasa ODROID en tu bicicleta. Yo utilice una bridas y un soporte para teléfono para sujetar la cámara de visión trasera a mi bicicleta de carretera, tal y como se muestra en la Figura 3. Enchufa la cámara de 720p y el módulo Wifi 0 a los puertos USB de ODROID-C0 antes de encender el ODROID-C0.

El teléfono inteligente, que es el visualizador de de la camara, debe conectarse al punto de acceso “ODROID_REARCAM” a través de Wifi usando la contraseña “saferiding”. Al abrir la página http://192.168.1.1:8080/cam1.mjpg en tu teléfono, podrás tener una vista posterior digital en tiempo real. Pedalea seguro y ser feliz con tu sistema de cámara de visión trasera casero.

Cargando la cámara en casa

EJECUTABLES DE 32 BITS EN UBUNTU DE 64 BITS

CRONICAS DE UN CIENTIFICO LOCO

por Bo Lechnowsky (@respectech)

"¡Maldición!" Dices, cuando aparece un mensaje de error en tu ODROID-C2 ejecutando Ubuntu de 64 bits:

```
$ sudo ./r3
sudo: unable to execute ./r3: No
such file or directory
```

Lo compruebas y observas que realmente el archivo se encuentra en el directorio, y está configurado como ejecutable. Recuerdas haberte encontrado con este problema hace un par de años, pero no recuerdas dónde pusiste tus notas sobre cómo solucionarlo. Murmuras en voz baja: "¡Esta vez, me aseguraré de publicarlo en uno de mis foros favoritos después de que averigüe cómo solucionarlo de nuevo!"

El comando "file r3" de Linux proporciona el siguiente resultado:

```
$ file r3
r3: ELF 32-bit LSB executable,
ARM, EABI5 version 1 (SYSV),
dynamically linked, interpreter /
lib/ld-linux-armhf.so.3, for GNU/
Linux 2.6.27, BuildID[sha1]=96b9
4abdd300ad350ceb0b48b4b0461abd48
1c18, stripped
```

Esto despierta algunas respuestas sinápticas en una región de cierta inactividad de tu cerebro. "¡Aja!, tiene algo que ver con un binario de 32 bits ejecutándose sobre un sistema operativo de 64 bits!" Tras realizar algunas pruebas, das con la siguiente solución:

```
$ sudo dpkg --add-architecture
armhf
$ sudo apt install libc6:armhf \
  libncurses5:armhf
  libstdc++6:armhf
```

Tras instalar estas librerías, intentas ejecutar el ejecutable de nuevo. Esta vez, ¡funciona! "¡La victoria es mía! ¡La tecnología debe arrodillarse ante mi grandeza! La noche transcurre a medida que desarrollas tu entorno de trabajo digital para dominar el mundo. No sin antes de intentar ejecutar un ejecutable gráfico de 32 bits y te encuentran con:

```
error while loading shared li-
braries: libX11.so.6: cannot open
shared object file: No such file or
directory
```

"¡Demonios!" ¡Gritas! De modo que contestas al mensaje con el siguiente comando:

```
$ sudo apt install libx11-6:armhf
```

El sistema entonces devuelve este resultado:

```
error while loading shared
libraries: libXt.so.6
```

Tú y el sistema operativo os enzarzáis en una lucha de poder:

```
$ sudo apt install libxt6:armhf
error while loading shared li-
braries: libXaw.so.7
$ sudo apt install libxaw7:armhf
error while loading shared li-
braries: libfreetype.so.6
$ sudo apt install
libfreetype6:armhf
```

"¡Podemos estar con esto toda la noche!" Le dicen a tu archienemigo digital. Para la mayoría de las otras aplicaciones, esto debería ser suficiente. Pero para tu arma secreta Rebol 2, tienes que ejecutar un par de comandos más:

```
$ sudo apt install xfonts-100dpi
xfonts-75dpi
$ sudo reboot
```

"¡Declárame tu lealtad, Tecnología!" Colocas los brazos detrás de tu cabeza y los pies encima de la mesa mientras disfrutas de ese momento de haber conseguido finalmente que tus vasallo digitales se rindan a tus pies.

CONOCIENDO UN ODROIDIAN

FABIEN THIRIET (@FAB)

editado por Rob Roy (@robroy)

Por favor, háblanos un poco sobre ti

Tengo 51 años y vivo con mi esposa Nadine y mis dos hijos en Orleans, Francia, a unos 100 km al sur de París. Durante los últimos 10 años, he estado dando clases sobre redes y mantenimiento y diseño digital en un instituto francés. Recibí un grado de Sistemas Eléctricos del estado francés para poder dar clase en institutos, y también obtuve un grado de ingeniería en ciencias de la computación. Antes de ser profesor y de finalizar mis estudios en 1988, empecé a trabajar en una compañía internacional que fabricaba tarjetas inteligentes y lectores de tarjetas con chip para aplicaciones bancarias, de salud y telecomunicaciones. Desarrollé un sistema operativo para estos sistemas altamente seguros basado en chips Motorola 6805. También trabajé en los lectores de tarjetas, diseñando PCBs y todo el cableado adyacente. Durante los últimos 3 años con esta compañía, me convertí en su "evangelista" del Java, ya que las tarjetas inteligentes se hicieron cada vez más "inteligentes" incluyendo una Máquina Virtual Java como parte del sistema operativo. Colocamos nuestra JVM en tarjetas y lectores, con el fin de tener soluciones más flexibles y seguras. Estos sistemas eran muy innovadores, porque Android de Google tiene una arquitectura similar con la poderosa JVM de Dalvik sobre Linux. Fue una experiencia muy grata, ya que viajé por todo el mundo, enseñando Java para sistemas embebidos y pasando tiempo en Estados Unidos, China y Sudáfrica.

Tras esta rica experiencia, decidí con otros dos tipos de la compañía dar un gran salto y fundar nuestra propia empresa en 1999 desarrollando soluciones SMS y MMS para operadores de telecomunicaciones. Era la época donde el uso de los SMS/MMS estaba creciendo muy rápido, y los operadores de telecomunicaciones no estaban realmente preparados para manejar cientos de millones de mensajes al día.

Junto a mi equipo, desarrollamos soluciones muy potentes y escalables para gestionar SMS / MMS, basado en servidores Java y Linux, que se ejecutaban bajo Redhat por aquel entonces. Esto era muy diferente a lo que hacía antes, ya que manejábamos auténticas granjas de servidores y clústeres de red, y configuraciones de máquinas de balanceo de carga. El tipo de servidores que utilizamos en este momento eran Dell PowerEdge 8450 (8 CPUs y 512MB RAM), con un peso de alrededor de unos 60kg, que de hecho eran menos potentes que un sólo ODROID-C1. En 2005, decidí vender mis acciones de la compañía que funde conjuntamente para tener más tiempo para mi familia. Dar clases en el instituto de mi localidad cerca de casa era la forma más adecuada para lograr mi meta.

Fabien con su ODROID montado en su coche

Con mi esposa Nadine, tenemos dos hijos: Antoine, que tiene 20 años, y estudia medicina en una universidad francesa y Martin, que tiene 16 años, y se ha convertido en un experto en diseño 3D e impresión. Él fue quien hizo la estructura de Redtop que puede que hayas visto en un número anterior de ODROID Magazine. Va al instituto donde estoy dando clases, y escogió ciencias de la ingeniería como asignatura optativa, y trabaja en su proyecto usando un ODROID-C1. Mi esposa es una educadora especializada en personas con minusvalía. En casa, se dedica a hacer curiosas decoraciones con materiales viejos y olvidados que coge de todas partes.

¿Cómo empezaste con los ordenadores?

Empecé en el mundo de la informática, no con un Commodore 64, un ZX81, o con cualquier otro sistema de este tipo, sino con un mini ordenador Nixdorf de principios de 1980. Los Mini ordenadores por aquel entonces venían a ser pequeños sistemas mainframe. Nixdorf era una empresa alemana y ahora forma parte de Siemens Group.

Mis padres vendían y reparaban máquinas agrícolas, y para gestionar su almacén de piezas de repuesto decidieron comprar un ordenador. El miniordenador Nixdorf fue diseñado específicamente para tiendas profesionales, fue el primer ordenador en el que trabajé. Éste venía con un software diseñado para las empresas de maquinaria agrícola, llegue a realizar muchas modificaciones de código en BASIC, que era el lenguaje de programación primario de esas máquinas, que databan de principios de los 80. La empresa de software que trabajaba para el distribuidor

Una vieja reliquia, el mini ordenador Nixdorf

Fabien y su familia conocedora de la tecnología y con mucho talento muestran sus proyectos.

local de Nixdorf se enteró de lo interesante que eran las modificaciones que había realizado y decidió contratarme durante las vacaciones de verano.

Realmente disfrute con este nuevo trabajo y aprendí mucho, ya que con este miniordenador fui capaz de manejar hasta 10 terminales dentro de una red muy primitiva. Las direcciones de los terminales en la red se configuraban en realidad con microswitches DIP. El miniordenador Nixdorf era capaz de conectarse con otros microordenadores remotos a través de un módem de 1200 baudios. A pesar de esta velocidad tan baja, no era lento, ya que no existían interfaces gráficas para nada.

¿Qué te atrajo a la plataforma ODROID?

En mis clases hasta el 2012, usaba un portátil Asus con Ubuntu. En la primavera de 2012, la fundación Raspberry Pi lanzó un minúsculo ordenador muy inusual. Sé que ODROID había lanzado antes un modelo de ordenador de placa reducida, pero esta información aún no había llegado a los viejos países europeos.

Decidí hacer pruebas con la Raspberry Pi, y reemplacé todas mis portátiles Asus por este nuevo juguete, principalmente por razones de espacio, ya que la Pi es muy eficiente a nivel de espacio en comparación con un portátil tradicional. A los estudiantes les encanto al principio, pero pronto se dieron cuenta que era muy lenta, especialmente cuando navegaban por Internet.

En 2014, Odroid llego por fin a Europa, así que migré todas mis clases a la plataforma ODROID-C1. Este año, he hecho una segunda actualización al ODROID-C2, que es perfecto para lo que estoy haciendo con mis alumnos, es mucho más rápido.

¿Cómo usas tus ODROIDs?

En clase, el ODROID se utiliza para todo: navegar por la web, escribir documentos, programar en Python, testear redes GNS3, diseño FreeCad, sistemas de videovigilancia y robótica. En casa, mi ordenador principal es el Redtop como ya he mencionado antes, basado en un ODROID-C2. Lo hago todo con él: preparar mis clases, poner a prueba mis laboratorios y desarrollar mis propios sistemas. Como puedes ver en las diferentes imágenes de este artículo, he desarrollado muchos proyectos con placas ODROID como con:

- Un mezclador de guitarra basado en Guitarix, gestionado por un ODROID-XU y Jackd. El XU está conectado por cable a un dispositivo MIDI Korg Kontrol2 y a un adaptador USB Behringer Audio.

- Un preciso sistema portátil para marcar las fronteras geográficas de la tierra, basado en un ODROID-C1, una pantalla táctil en color de 3,2 TFT y el dongle USB GPS Hardkernel. Desarrollé el software en Python con la librería ncurses para tener una interfaz semigráfica agradable, pero sin escritorio para una solución liviana.

- Una dashcam para mi coche, para registrar todo lo que sucede delante de mi coche, de cara al seguro, en caso de accidente. El dashcam utiliza un ODROID-W con una cámara Pi.

- Un radar de vigilancia que envía imágenes a través de

El increíble y original proyecto de Fabien, el portátil RedTop

Fabien es un músico de guitarra eléctrica y acústica muy hábil.

Un mezclador de guitarra futurista que usa un ODROID-XU

MMS en el momento en el que algo se mueve frente a dos sensores PIR/microondas. Está creado con un ODROID-C1, un dongle USB Huawei E220 3G y una cámara USB de visión nocturna junto con un doble sensor de movimiento. Este sistema está listo para utilizarse, sólo necesita alimentarse con una toma de corriente de 230V.

¿Cuál es tu ODROID favorito y por qué?

Mi ODROID favorito es el C2, ya que el cabezal de pines GPIO sigue siendo más o menos compatible con Raspberry Pi, que realmente es una gran locomotora en el mundo de los SBC como placa adicional. La familia ODROID-XU es muy buena, pero desafortunadamente no cuenta con los GPIOs de Raspberry Pi.

¿Cuál fue su motivación para desarrollar tu portátil ODROID "RedTop"?

Mi RedTop realmente es mi dispositivo preferido. En primer lugar, mi hijo me ayudo en su desarrollo, el cual realizó un trabajo muy bueno en la parte CAD, y en segundo lugar, el RedTop está sustentado por un C2 que es el ordenador que estoy usando en mis clases. Por lo tanto, todas las cosas que estoy haciendo se pueden probar y preparar muy bien con el RedTop. El RedTop tiene una pantalla LCD de color de 13 pulgadas, de modo que es muy cómodo trabajar con él.

¿Qué innovaciones te gustaría ver en futuros productos Hardkernel?

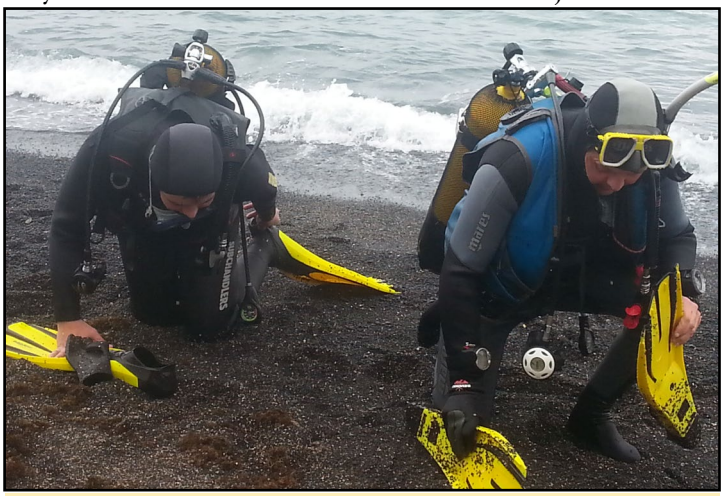
Como la mayoría de la gente, me gustaría tener algo más rápido. Un mejor núcleo ARM con una velocidad de reloj real de 2GHz estaría muy bien. Lo que es muy importante, en mi opinión, es que continúe la compatibilidad GPIO con la Raspberry Pi. Además, el C2 no cuenta con SPI y RTC, sería bueno tenerlos de nuevo y así evitar el tener que usar placas adicionales para esta finalidad.

¿Qué aficiones e intereses tienes aparte de los ordenadores?

Desde abril a octubre, cuando la temperatura del océano Atlántico es agradable, practico buceo con mi hijo Martin en la costa de Bretaña. Soy un instructor de buceo CMAS de 3 estrellas. He practicado buceo alrededor de todo el mundo en diversos lugares como Islas Maldivas, Hurghada en Egipto, Cuba y las islas francesas del Caribe. También toco la guitarra acústica y eléctrica. La última canción que toqué fue "Hello" de Adele al estilo fingerpicking.

¿Qué consejo le darías a alguien que quiere aprender más sobre programación?

Después de más de 30 años, trabajando con sistemas digitales, puedo decir que aprender algo y especialmente la programación, es algo que uno no sólo puedes hacer en la escuela. No te valgas solo de los profesores. Prueba cosas por ti mismo, comete errores y descubre soluciones por ti mismo. Los mejores programadores que contraté durante el tiempo que trabajé en mi propia compañía eran personas muy curiosas, con un sentido muy desarrollado de la autonomía mientras trabajaban.



Fabien ha practicado buceo por todo el mundo