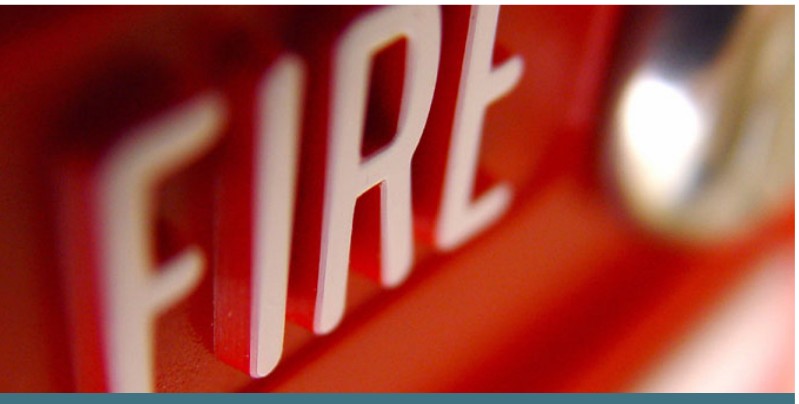
ISCSi • Cyanogenmod en U3/XU/XU3/XU4 • Corregir el Overscan en Ubuntu

ODROID Magazine

Año Tres
Num. #26
Feb 2016

La nueva generación de *Potencia* Portátil

El ODROID-C0 es el nuevo y espectacular mini ordenador



- Sistema de detección de incendios
- Sistema de Control MUNIN
- Juegos Windows 2000
- El Kit Robot OWEN

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.

HARDKERNEL

Pollin ELECTRONIC

Ahora estamos enviando los dispositivos ODROID U3 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

A muchos usuarios ODROID les encantaba el ODROID-W, cuya producción fue cancelada en 2014. Desde entonces, muchas han sido las peticiones para crear un equipo IoT similar que pueda utilizarse para aplicaciones industriales, que pueda llevarse encima y cuyo hardware contara con un tamaño mínimo y un bajo consumo.

Hardkernel ha respondido con la ODROID-C0, que cuenta con el potencial del popular ODROIDC1 y la retirada de algunas conexiones, de modo que los usuarios puedan crear perfectos dispositivos adaptados exactamente a sus necesidades. Un ejemplo de aplicación del ODROID-C0 es el robot OWEN, diseñado por Bo en Ameridroid, que puede caminar, bailar y mostrar expresiones faciales a través de una interfaz web. El ODROID-C0 está disponible en http://bit.ly/1WFYKOL por 25$.

¿Sabías que se puede ejecutar Windows 2000 en un ODROID? Tobias nos muestra cómo instalarlo usando QEMU, para que puedas ejecutar tus programas y juegos favoritos de Windows. Josh nos presenta una alternativa a Android llamada CyanogenMod, Ilham nos ayuda a mantenernos a salvo con un kit de detección de incendios bastante robusto, Nanik desmitifica la pila Bluetooth de Android, Adrian explica cómo utilizar Munin y David nos describe el protocolo iSCSI para la LVM.

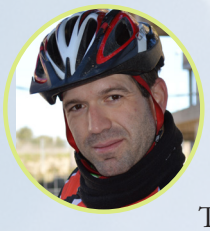
ODROID Magazine, que se publica mensualmente en http://magazine.odroid.com/, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera 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

ODROID Magazine

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`.



Manuel Adamuz, Editor Español

Tengo 31 años y vivo e n 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.

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.

Bruno Doiche, Editor Artístico Senior

Bruno hoy por hoy suele perderse constantemente con David en Magic the Gathering. Necesita adquirir algunas habilidades de juego haciendo un rápido recorrido por Las Vegas.

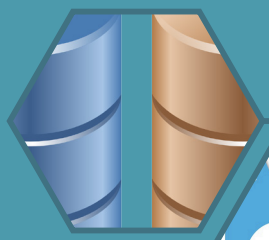
Nicole Scott, Editor Artístico

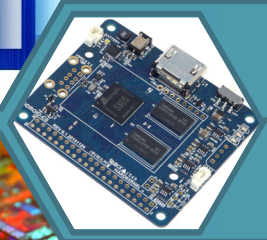
Soy una experta en Producción Transmedia y Estrategia Digital especializa en la optimización online y estrategias de marketing, administración de medios sociales y producción multimedia impresa, web, vídeo y cine. Gestiono múltiples cuentas con agencias y productores de cine, desde Analytics y Adwords 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 enhttp://www.nicolecscott.com. 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. Todavía estoy bastante enamorado de muchos 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.

INDICE





INTERFAZ DE SISTEMA PARA PEQUEÑOS ORDENADORES

ISCSI TE FACILITA LAS COSAS

por David Lima

Si dispones de un gran disco duro y quieres compartir tu espacio disponible con otro ODROID, o si quieres configurar un servidor de archivos y agrupar todo el almacenamiento en un único punto, lo primero en lo que sueles pensar es en usar el sistema de archivos en red (NFS). Sin embargo, éste presenta algunos problemas de seguridad, puesto que no hereda correctamente los permisos y probablemente no necesites compartir los datos con algunos clientes. En este artículo voy a plantear una alternativa, utilizar iSCSI.

iSCSI, acrónimo de Interfaz de Sistema para Pequeños Ordenadores, es un protocolo de Internet que se utiliza para conectar sistemas de almacenamiento de datos entre diferentes recursos. Con iSCSI, puede crear un disco virtual en un servidor y asignarlo a un cliente como un dispositivo en bloque y hacer lo que quieras con él, como crear un sistema de archivos de cualquier clase o utilizarlo como un área de intercambio como si fuera una unidad física local. En lugar de enviar llamadas de procedimiento remoto como hace NFS, iSCSI envía instrucciones SCSI a través de la red. Estas son las mismas instrucciones que tu sistema utiliza para escribir bloques de datos en tu disco.

De esta forma aumentas la seguridad, ya que puedes ajustar los permisos como lo haces normalmente sin tener que preocuparte de "machacar root" y así evitar el acceso a la cuenta root en los sistemas de archivos exportados. Además se mejora el rendimiento ya que el sistema se salta una capa de interacción y simplemente envía una petición de escritura SCSI.

Fundamentos

El disco virtual que vamos a crear se llama LUN (número de unidad lógica). El LUN es en realidad el número usado para identificar el disco virtual, pero también se utiliza para referirse al propio disco. También existe el iniciador SCSI y de destino SCSI. El destino SCSI actuará como un servidor, donde se encuentra el dispositivo de almacenamiento, y el iniciador SCSI es el cliente que es el que recibe los discos virtuales. Puedes pensar en el iniciador como una tarjeta de expansión de bus SCSI, pero en lugar de cables utiliza tu red. En primer lugar, instala la dependencia necesaria en el destino:

```
$ sudo apt-get install tgt
```

Luego, haz lo mismo con el iniciador:

```
$ apt-get install open-iscsi
```

Empezando por el sistema destino, tenemos que crear un destino que sea accesible por los clientes:

```
$ tgtadm --lld iscsi --op new \
  --mode target --tid 1 \
  -T iqn.2016-01.com.server-
iscsi:vdisk.1
```

La opción -T determina la dirección del iniciador, que es como el iniciador verá el LUN que tu creas. El formato de la dirección del iniciador es iqn. yyyy-mm.<reverse_domain_name:LUN name, donde iqn es el correspondiente nombre iSCSI, una dirección con un formato de nombre. La fecha (yyyy-mm) especifica cuándo la autoridad nombre tomó la propiedad del dominio, que es necesario si el iSCSI es utilizado en internet. En redes locales, ésta puede ser cualquier fecha. El nombre de dominio del servidor de destino también se utiliza para configuraciones específicas de Internet, y puede ser cualquier nombre en las redes locales. El nombre LUN puede ser cualquier etiqueta que sea fácil de identificar por el administrador, como el nombre del cliente o la finalidad. Puedes tener múltiples destinos en el mismo servidor cambiando el iqn y la id(tid) de destino.

Comprueba la configuración hasta aquí con el siguiente comando:

```
$ tgtadm --lld iscsi \
  --op show --mode target
```

Verás que LUN ID 0 ya está conectado. Se usa para el control interno de la utilidad y no necesitas cambiarlo. Ahora, tenemos que asignar un dispositivo bloque para crear el LUN de nuestro(s) cliente(s). Tenemos tres opciones:

1. Usar una partición normal de cualquier dispositivo conectado al sistema.
2. Crear un volumen lógico usando LVM para conectarlo al destino
3. Crear un archivo de datos en un sistema de archivos montado para usarlo como si fuera un dispositivo bloque.

Yo te recomendaría la opción 2 y así podrás aprovechar todas las funcionalidades del LVM en tu LUN de destino. La Opción 1 también es buena si no quieres utilizar LVM, y la opción 3 es útil para realiza pruebas cuando quieres configurar un iSCSI y no deseas cambiar las tablas de particiones o volúmenes lógicos, pero no se recomienda para con-

figuraciones a largo plazo.

Para hacer una demostración, voy a elegir la opción 3. Para crear el archivo de almacenamiento, escribe los siguientes comandos que requieren privilegios de root:

```
$ su
# dd if=/dev/zero \
  of=/home/odroid/iscsi-lun \
  bs=4096 count=1024000
```

Esto creará un archivo de 4 GB, que puede ser conectado a un nuevo LUN:

```
# tgtadm --lld iscsi \
  --op new --mode logicalunit \
  --tid 1 --lun 1 \
  -b /home/odroid/iscsi-lun
```

A continuación, comprueba tu LUN recién creado:

```
# tgtadm --lld iscsi \
  --op show --mode target
```

Si vas a crear el LUN con un dispositivo bloque existente, simplemente sustituye el nombre del archivo por la ruta del dispositivo, como /dev/sdb1 o /dev/rootvg/homelv. Estamos listos para que los clientes se unan a este destino:

```
# tgtadm --lld iscsi --op bind \
  --mode target --tid 1 -I
192.168.0.2
```

Esto permitirá que sólo los clientes con IP 192.168.0.2 se conecten y se unan a este LUN. También puedes reemplazar la IP por "ALL" para que cualquier iniciador pueda unirse a este destino, aunque esto no se recomienda por seguridad. Con la opción "ALL", cualquier usuario de la misma red podría leer los datos de este LUN destino, y si te conectas a éste desde dos iniciadores diferentes y empiezan a escribir, todo el sistema podría dañase. Incluso si tienes un clúster y necesitan compartir los LUNs, asegúrate de especificar las direcciones IP en lugar de configurarlo para que sea visible por todos.

Esta configuración sólo se mantendrá hasta que reinicies el servidor. Si quieres mantener los cambios después de reiniciar, vuelca la configuración en /etc/tgt/conf.d ya que cualquier archivo de configuración bajo esta ruta se añade automáticamente:

```
# tgt-admin --dump | \
  grep -v default-driver > \
  /etc/tgt/conf.d/targets.conf
```

Volviendo al iniciador, tenemos que buscar los nuevos LUNs disponibles. Para ello, utiliza el siguiente comando:

```
# iscsiadm --mode discovery \
  --type sendtargets \
  --portal 192.168.0.1
```

Verás el LUN creado en el destino, que ahora puede conectarse en:

```
# iscsiadm --mode node --target-
name \
  iqn.2016-01.com.server-
iscsi:vdisk.1 \
  --portal 192.168.0.1:3260
--login
```

Si deseas que el iniciador se conecte automáticamente tras reiniciar, edita el siguiente archivo:

```
# vi /etc/iscsi/iscsid.conf
```

y cambiar esta línea de manual a automático:

```
node.startup = automatic
```

Ahora, si escribes el comando "fdisk-l", el LUN de destino será visible como un nuevo dispositivo, por ejemplo, /dev/sdb. Todo lo que tienes que hacer es crear un sistema de archivos en él o añadirlo a tu lista de Gestión de Volúmenes Lógicos (LVM).

UN VISTAZO A CYANOGENMOD

EMPECEMOS CON UNA VERSION DE ANDROID LIMPIA Y LIVIANA

por Joshua Sherman

En los últimos nueve años Android ha irrumpido en el mundo, invadiendo dispositivos de todas las formas y tamaños, incluyendo nuestros ODROIDs. Mientras que Android en sí proviene de Google, los sistemas operativos basados en Linux de código abierto tienen un sinfín de versiones diseñadas y modificadas por los fabricantes de teléfonos inteligentes, cada uno aportando su propio enfoque en el sistema operativo. CyanogenMod es una de estas versiones, diseñada por una comunidad de más de 50 millones de personas y está disponible para tu ODROID.

¿Qué es CyanogenMod?

Puesto que Android es un sistema operativo de código abierto, sus versiones están disponibles para cualquiera que quie-ra compilarlas y utilizarlas a través del Android Open Source Project (AOSP). Miles de dispositivos son compatibles, inclu-yendo la mayoría de los ODROIDs con versiones compiladas por Hardkernel. Aunque muchas disfrutan de la experiencia limpia y liviana de AOSP, ésts suelen variar de un dispositivo a otro y pueden carecer de algunas funcionalidades, así que es posible que quieras hacerle una puesta a punto.

Aquí es donde entra CyanogenMod. CyanogenMod es una versión de Android de terceros diseñada y compilada por una comunidad de más de 50 millones de usuarios de todo el mundo. En esencia, CyanogenMod es el mismo que bare bones AOSP disponible para todo el mundo. La comunidad de CyanogenMod lo ha mejorado añadiendo algunos paquetes y características útiles con el fin de conseguir una Interfaz de Usuario (UI) limpia y optimizada.

Lo mejor de todo es que CyanogenMod sigue fiel a la filosofía básica de Android continuando siendo un proyecto de código abierto bajo la licencia Apache. Esto hace que CyanogenMod esté disponible para cualquier persona que quiera compilarlo y desarrollarlo en nuevos dispositivos, como los ODROID, aunque todos los dispositivos ofrecen una experien-cia similar. Esto entra en contraste con la mayoría de las versiones de Android, que a menudo vienen incluidas con software de terceros bajo licencia. Por ejemplo, la interfaz de usuario TouchWiz de Samsung sólo está disponible para dispositivos de Samsung y no puede ser compartida con otros dispositivos sin su permiso.

Algunas de las características que CyanogenMod añade a Android AOSP incluyen una interfaz de usuario totalmente nueva, mejor rendimiento, anclaje USB, conectividad VPN, soporte para audio FLAC, soporte para tema nativo y un sinfín de otras ventajas. CyanogenMod está además pre-rootueado para poder acceder a las funciones avanzada de Android. Echa un vistazo a los detalles en http://bit.ly/1nk2aKz.

¿Cuántas versiones de CyanogenMod existen?

El miembro de la comunidad @voodik ayuda a mantener versiones de CyanogenMod para usarlas en los ODROIDs. Actualmente existen dos versiones de CyanogenMod importantes que estas actualizadas y disponibles para los ODROIDs:

CyanogenMod 13 es la última y mejor versión disponible, basada en Android 6.0.1 Marshmallow, el más reciente sistema operativo Android que hay disponible. Incorpora las últimas características que vienen con Android 6.0.1, tales como la gestión de energía avanzada y el control de permisos. Las versiones CM13 para ODROID todavía no soporta módems 3G USB y adaptadores Bluetooth, de modo que aquellos que necesiten este tipo de conectividad en sus proyectos es mejor que utilicen en su lugar CM12.

CyanogenMod 12.1 es la versión más fiable, basada en Android 5.1 Lollipop. Al igual que Lollipop, tiene un año de antigüedad, lo que significa que ha tenido más tiempo para pasar por correcciones y optimizaciones. Es compatible con la mayoría de los periféricos, incluyendo módems 3G y Bluetooth, es una buena opción para usarla en aplicaciones que presenten problemas de compatibilidad con Android 6.0.1. Sin embargo no contarás con las nuevas funciones de gestión de energía y control de permisos de CM13.

CyanogenMod 11 también existe para algunos ODROIDs, pero es probable que prefieras CM 12 o 13 a menos que haya una razón concreta para mantenerte en CM11. También hay algunas versiones de CyanogenMod diseñada para Android TV, que ofrece al usuario una experiencia ligeramente diferente.

Recuerda que no existe “la versión perfecta”, se trata de

usar la que mejor se adapte a tus necesidades. Al final, todas giran en torno a la misma experiencia básica: ofrecer a los usuarios una versión de Android rápida, limpia y de código abierto.

Empezar con CyanogenMod

Una vez que elijas la versión de CyanogenMod que te interesa, echa un vistazo a los respectivos foros donde está disponible. La versión puede variar de un dispositivo a otro, así que tenlo en cuenta a la hora de comprobar qué versiones hay disponibles para tu dispositivo en concreto:

ODROID-XU3/XU4:

`http://bit.ly/1niFjiA`

ODROID-U3:

`http://bit.ly/1WIQYn8`

ODROID-XU:

`http://bit.ly/1KyX4Q5`

Al igual que Android o Ubuntu, la instalación de CM12 y CM13 siguen el mismo procedimiento. Después, lo que tienes que hacer es explorar el sistema operativo para ver si cubre tus necesidades.

Notas

CyanogenMod es 100% de código abierto, lo que significa que no incluirá las aplicaciones de Google por defecto, consideradas de código cerrado. Entre las que está Google Play Store, que probablemente estés acostumbrado a usar para descargar aplicaciones. No te preocupes, puedes instalar Play Store por tu cuenta a través del paquete GApps disponible en cada foro de CyanogenMod, o utilizar el Universal One-Click Installer de Hardkernel disponible en http://bit.ly/1nL6ymp. Incluye todas las aplicaciones de Google habituales, podrás utilizar Play Store si lo deseas, o seguir trabajando con la experiencia de código abierto básica de CyanogenMod.

CORREGIR EL OVERSCAN EN UBUNTU PARA ODROID-C1/C1+

SINCRONIZA TU ODROID CON TU MONITOR

editado por Rob Roy

Si el monitor usado con tu ODROID-C1/C1+ muestra un ligero corte de la imagen visible en pantalla, puede que estés experimentando overscan. Este es un problema muy habitual, especialmente en los monitores de televisión LCD. La solución por lo general es muy simple, y es probable que el problema esté relacionado con los ajustes del monitor LCD. Algunos monitores de PC con entrada HDMI también aplican overscan a la entrada HDMI, suponiendo que usen una señal de emisión de TV. Los monitores que se utilizan para la televisión por lo general tienen el overscan activado por defecto. Esta característica es muy normal y ha estado presente desde los comienzos de la televisión. El Overscan se utiliza para recortar los bordes de los fotogramas de vídeo con el fin de eliminar esos bordes irregulares o distorsionados que a menudo aparecen con la retransmisión de video. Para el espectador tiene como resultado una imagen más limpia y el overscan simplemente no se percibe. Sin embargo para una pantalla de ordenador, esto puede ser un problema. Por esta razón, los monitores LCD para ordenador normalmente no permiten overscan y si cuenta con esta característica, está desactivada por defecto. Para poder usar aquellos monitores que no tienen la opción de overscan integrada en el hardware, puedes seguir estos pasos te que te ayudarán activar overscan por software en Ubuntu para el ODROID-C1/C1 +.

1. Consigue y ajusta los valores overscan de izquierda, superior, derecha e inferior usando el método de ensayo y error, escribiendo los siguientes comandos en una ventana de terminal:

```
$ su
# cat /sys/class/graphics/fb0/window_axis window axis is [100 100 1919 1079]

# echo 100 100 1919 1079 > /sys/class/graphics-/fb0/window_axis
# echo 0x10001 > /sys/class/graphics/fb0/free_scale
```

2. Crea un script BASH llamado overscan.sh, con los valores de izquierda, superior, derecha e inferior:

```
# cat overscan.sh
#!/bin/bash

echo 100 100 1919 1079 > /sys/class/graphics/fb0/window_axis
echo 0x10001 > /sys/class/graphics/fb0/free_scale
```

3. Guarda el script en el directorio /etc/init.d para que se inicie automáticamente durante el arranque:

```
# cp overscan.sh /etc/init.d/
# update-rc.d overscan.sh defaults
```

Si tienes preguntas, comentarios o sugerencias, por favor visita el artículo original en http://bit.ly/236QKum.

DETECCION DE INCENDIOS POR CAMARAS DE SEGURIDAD

DOMINA EL FUEGO CON UNA WEBCAM Y UN ODROID

por Ilham Imaduddin

Hace cientos de miles de años nuestros antepasados llegaron a dominar el fuego. Sin embargo, es una de las herramientas más peligrosas que puede llegar a ser letal cuando está fuera de control. Los incendios de hoy día pueden ser causados por un mal funcionamiento de un aparato, un fallo eléctrico o incluso un rayo. En 2014, hubo casi 500.000 incendios en edificios sólo en los EE.UU., los cuales causaron miles de lesiones y muertes.

Espero que estos datos te hayan hecho "entrar en calor", porque hoy vamos a proteger nuestros hogares de un posible incendio. En este tutorial, vamos a crear una aplicación de vigilancia que detectará un incendio y nos avisará. Imagínate que te encuentras en medio de una aburrida reunión y no hay nadie en casa. En lugar de recibir una llamada de los bomberos diciendo que han ido a tu casa, este programa te avisará desde un principio si se inicia un fuego, ¡así conseguirás más tiempo para salvar tu casa!

Para hacer todo esto, tenemos que escribir un simple script en Python usando el módulo OpenCV. OpenCV es una conocida librería que agrupa varios algoritmos de tratamiento de imágenes en una sencilla interfaz. Este tutorial mostrará cómo una simple placa ODROID puede llegar a ser mucho más que un sencillo y cotidiano dispositivo informático.

El hardware

Necesitarás un ODROID como el XU4 o C1+, una webcam USB y una conexión a Internet para tu placa. En mi caso, he utilizado un ODROID-C1, una cámara Logitech C525 y un módulo WiFi genérico.

El software

En este tutorial repasaremos un programa escrito en Python 2. Este programa requiere tres módulos: el módulo cv2 OpenCV que se utiliza para el reconocimiento de imágenes por ordenador, NumPy que cv2 necesita para manejar grandes matrices y estructuras de datos y el módulo envelopes, que nos ayudará con la gestión del correo electrónico. Hay varios y excelentes tutoriales en los foros ODROID en http://bit.ly/1E66Tm6 que detallan cómo instalar OpenCV. También puedes descargar la versión personalizada de ROS que ya incluye OpenCV en http://bit.ly/1JvIxK1 (XU3 / XU4) y http://bit.ly/1ZLMpID (C1/C1+). Hay una guía de instalación para NumPy en su sitio oficial en http://bit.ly/1KpXV5B. Puedes instalar el módulo envelopes siguiendo con los siguientes pasos:

Primero, si pip no está instalado, se puede instalar con el siguiente comando:

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

A continuación, instala envelopes:

```
$ pip install envelopes
```

Por último, en el código Python importa los tres módulos dentro de la aplicación con las siguientes líneas:

```
import cv2
import numpy as np
from envelopes import Envelope
```

Dominar el fuego

Antes de que podamos detectar un indencio con nuestra cámara, necesitamos conocer las propiedades específicas del fuego que lo hacen diferente del resto de objetos que percibe la cámara. Un fuego tiene al menos dos propiedades que lo hacen único con las que podemos trabajar, la intensidad y color. La estrategia consiste en filtrar la intensidad y el color en cada imagen, luego combinar ambos filtros para formar una nueva "máscara de fuego". Para entender el código, debes estar familiarizado con OpenCV.

Filtro de intensidad

El fuego es brillante y en muchos casos es el objeto más luminoso en la imagen. Esto significa que podemos identificar el fuego seleccionando los píxeles con los niveles de intensidad más altos. Para hacer eso, vamos a convertir la imagen capturada por la webcam en YCrCb, a partir de la gama de colores BGR que OpenCV utiliza por defecto,. Una vez que la imagen está en formato YCbCr,

podemos filtrar por el componente Y, que es el valor de la luminosidad.

```
def filter_intensity(frame,
threshold_y):
    ycrcb = cv2.cvtColor(frame,
cv2.COLOR_BGR2YCrCb)
    threshold_low =
np.array([threshold_y, 0, 0])
    threshold_high =
np.array([255, 240, 240])

    return cv2.inRange(ycrcb,
threshold_low, threshold_high)
```

El proceso cv2.cvtColor se utiliza para convertir una imagen con una gama de colores en otra. En OpenCV, una imagen se representa como una matriz NumPy, de modo que crearemos matrices umbrales utilizando np.array. Tras crear estas matrices, cv2.inRange realiza el proceso de filtrado tomando la imagen como primer parámetro y los umbrales como segundo y tercer parámetro. El proceso devolverá una máscara filtrada basada en estos parámetros.

Filtro de color

El fuego necesariamente no siempre es de color rojo por completo. A veces tiene una mezcla de naranja o incluso blanco. Independientemente de esto, el componente rojo siempre está presente. Esto significa que podemos filtrar la imagen seleccionando los píxeles con un alto contenido de color rojo para identificar las llamas. Filtrar por el rojo obteniendo una imagen con una gama de color BGR es fácil y se puede hacer con este código:

```
def filter_color(frame,
threshold_r):
    threshold_low = np.array([0,
0, threshold_r])
    threshold_high =
np.array([threshold_r-20,
threshold_r-10, 255])

    return cv2.inRange(frame,
threshold_low, threshold_high)
```

El parametro Threshold_high está definido para que los componentes azules y verdes sean más pequeños que el componente rojo. Al igual que ocurre con la intensidad, el filtrado del color se realiza con cv2.inRange que coge la imagen y los umbrales como parámetros y devuelve una máscara filtrada.

Extraer el fuego

Ahora que tenemos creados el filtro de intensidad y de color, podemos extraer el fuego de una imagen. Primero, vamos a crear una máscara de intensidad y una máscara de color activando las funciones filter_intensity y filter_color, respectivamente. A continuación, aplicaremos algún tipo de tratamiento para mejorar la calidad de ambas máscaras. Después, combinaremos las dos máscaras para crear una nueva máscara “fuego”.

```
def extract(frame, threshold_y,
threshold_r):
    intensity_mask = filter_
intensity(frame, threshold_y)
    color_mask = filter_
color(frame, threshold_r)

    kernel = np.ones((5, 5),
np.uint8)
    intensity_mask = cv2.
dilate(intensity_mask, kernel,
iterations=1)

    kernel = np.ones((10, 10),
np.uint8)
    color_mask = cv2.
dilate(color_mask, kernel, itera-
tions=1)


    return cv2.bitwise_and(color_
mask, color_mask, mask=intensity_
mask)
```

El problema con la funciones filter_intensity y filter_color es que a veces pueden devolver una máscara dañada. Podemos aplicar el proceso cv2.dilate para solucionar esto. Dilate aumenta el área de la máscara, rellenando el hueco entre las máscaras dañadas.

Finalizando

Vamos a empezar por activar la cámara en el código. Luego, crearemos un bucle para capturar continuamente nuevas imágenes desde la cámara. Dentro de este bucle, no sólo vamos a obtener nuevas imágenes, sino que también las procesaremos. Lo primero que vamos a hacer tras recibir una nueva imagen es activar la función de extracción que hemos creado anteriormente:

```
cam = cv2.VideoCapture(0)
counter = 0

while (True):
    frame = cam.read()[1]
    fire_mask = extract(frame,
250, 230)

    # Fire handling

    if cv2.waitKey(1) & 0xFF ==
ord('q'):
        break

cam.release()
```

Añadimos la cámara creando un objeto VideoCapture. Su argumento es el índice de la cámara, que será 0 si sólo se conecta una cámara. Puedes usar una segunda cámara pasando a 1 y así sucesivamente para más cámaras. En el bucle, leemos una imagen y luego la filtramos para detectar cualquier fuego. Finalmente, aplicamos un código adicional para minimizar los falsos positivos.

```
# Fire handling
contours = cv2.findContours(fire_
mask.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)[1]

if len(contours) > 0:
    for i, c in
enumerate(contours):
        if cv2.
```

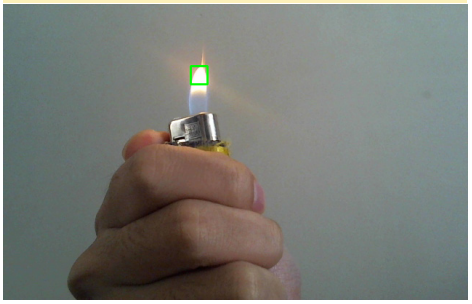
```
contourArea(contours[i]) > 100:
            x, y, w, h = cv2.
boundingRect(contours[i])
            cv2.rectangle(frame,
(x, y), (x+w, y+h), (0, 255, 0),
2)

    counter = counter + 1
    if counter == 100:
        # Send Notification
else:
    counter = 0
```

El primer proceso, cv2.findContours busca cualquier contorno, tal y como sugiere su nombre. Los contornos pueden definirse simplemente como "un esquema que representa o que limita la figura o la forma de algo." El cv2.findContours coge una imagen, además de unos parámetros para realizar ajustes, y devuelve una matriz de todos los contornos encontrado en esa imagen. Cuando pasamos la función cv2.findContrours sobre nuestra máscara fuego, esperamos que la máscara esté vacía, es decir, que no aparezca ninguna forma o contorno. Si resulta que hay una llama o algo de fuego, la máscara no estaría vacía y obtendríamos una matriz resultante con el contorno de una llama.

Si se devuelve un contorno, comprobamos si éste es lo suficientemente grande como para considerarlo un fuego. Esto es importante porque ayuda a minimizar los falsos positivos de la aplicación informando de un incendio cuando en realidad no hay ninguno. Por último, si el contorno detectado es lo suficientemente grande, dibujamos un rectángulo a su alrededor. La figura 1 muestra un ejemplo de una llama detectada.

Figura 1 - Demostración de un fuego que será detectado por la cámara



La función counter nos ayuda a minimizar los falsos positivos. En lugar de enviar una alerta tras la primera imagen de la cámara que detecta fuego, el programa esperará varias imágenes para identificar que verdaderamente se trata de un incendio antes de enviar la alerta. Si el incendio detectado dura lo suficiente, guardaremos esa imagen y la enviaremos junto con el mensaje de alerta.

```
# Send Notification
cv2.imwrite('fire.png', frame)

mail = Envelope(
    from_addr=(u'from@email.com',
u'Name'),
    to_addr=(u'to@email.com',
u'Name'),
    subject=u'Your House is On
Fire!',
    text_body=u"Call emergency
number immediately!"
)
mail.add_attachment('fire.png')
mail.send(
    'your.smtpserver.com',
    login='your@email.com',
    password='your password',
    port=587,
)

print "Email sent"
break
```

La imagen se guarda en el disco con la función cv2.imwrite, que coge el nombre de archivo y la imagen como parámetros. Usando el objeto Envelope, enviamos una notificación por correo electrónico en la cual se avisa de que se ha detectado un incendio que incluye la imagen guardada. Una vez enviado nuestro mensaje de alerta, interrumpimos el bucle para poner fin a la aplicación.

Si envías el e-mail desde una dirección de Gmail, hay una opción que evita los accesos que no cumplen con determinados requisitos de seguridad. Tendrás que desactivar esta opción para poder enviar un email usando Envelope.

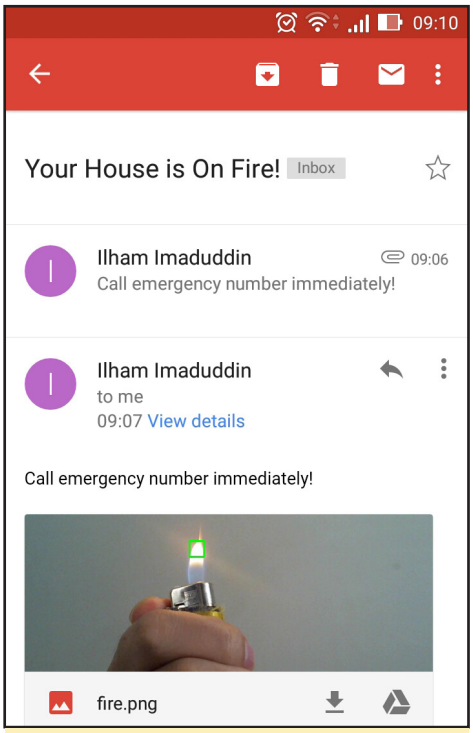


Figura 2 - Notificación por email desde programa de detección de incendios

Prender fuego

Ahora, pondremos nuestro programa a prueba. Como no estoy lo suficientemente loco como para encender fuego en mi casa sólo para hacer pruebas, no verenos como se quema mi casa. En su lugar, usaremos un encendedor que será nuestra fuente de fuego. La Figura 2 muestra el correo electrónico de alerta que recibí cuando probé el programa.

Si decides probarlo, es posible que tengas que cambiar los valores umbrales. Se encuentran definidos en el umbral más adecuado para mi casa. Los valores puede que tengan que cambiarse debido a las diferencias de iluminación en las zonas que están siendo monitoreadas.

Expansión

Existen muchas posibilidades para ampliar esta aplicación básica. Por ejemplo, podemos grabar el fuego, enviar SMS en lugar de un email, o incluso llamar a los servicios de emergencia de forma automática. También podríamos añadir otro algoritmo a nuestra cámara de vigilancia, como la detección de movimiento o la detección de rostros.

JUEGOS LINUX

EJECUTAR JUEGOS DE WINDOWS 2000 EN UN ODROID

por Tobias Schaaf

Actualmente, podemos jugar a miles de juegos en ODROID gracias a la emulación, incluidos juegos de consolas como SNES, PS1, N64 y mucho más. Sin embargo, si queremos ejecutar un juego creado originalmente para PC, contamos con menos opciones. Un gran número de juegos basados en DOS se puede ejecutar en ODROID utilizando el emulador DOSBox. Además de eso, existen versiones nativas para Linux de los juegos más antiguos de Windows y DOS como Arx Fatalis, Planeta natal, Jedi Knight 2 y 3, Quake 3 y muchos otros.

Aún así, si queremos jugar a nuestros viejos juegos de Windows, los que podemos ejecutar no son muchos, ya que sólo unos cuantos han sido exportados realmente a Linux. En este artículo, quiero hablar de otras opciones que hay disponibles y averiguar si realmente podemos ejecutar un Sistema Operativo Windows en nuestro ODROID.

Esta guía mayoritariamente es para "probar cosas", y no te va a proporcionar una rápida y completa máquina de Windows sobre ODROID. El rendimiento de esta configuración, una maquina virtual (VM) con Windows 2000 ejecutándose sbore QEMU es muy baja y determinadas tareas requieren mucho tiempo. Si lees esto con la esperanza de ejecutar Battlefield 3 directamente en un ODROID, esta guía no es para ti. Sólo es un estudio sobre lo qué es posible y hasta dónde se puede llegar.

Requisitos

Puesto que las CPUs ARM no son CPUs x86, ejecutar un sistema operativo x86 en un ODROID necesitará cierta preparación, de modo que necesitamos instalar algunos programas en nuestro sistema y configurarlos antes de intentar instalar Windows y cualquier programa de Windows:

- El emulador QEMU para emular un PC compatible con arquitectura x86 en nuestro ODROID. QEMU tiene muchas versiones y se mejora continuamente. Yo uso la versión de QEMU de mi propio repositorio, actualmente es la 2.5.0.
- El ODROID-XU4 ofrece la CPU más rápida hasta el momento, es necesaria para conseguir la mejor experiencia posible. Un U3 probablemente funcione tambien, pero sin duda irá más lento. Un C1 no se puede usar para esto, simplemente carece de la CPU y la RAM necesarias para emular Windows 2000.
- Al menos 4 GB de espacio libre en la SD, eMMC o disco externo.
- CD/ISO de Instalación de Windows 2000. Por lo general, todas las versiones de Windows deberían funcionar, pero en pruebas anteriores he descubierto que Windows 95 carece de un buen soporte para DirectX, lo cual impide que la ma-yoría de los juegos se puedan ejecutar. Windows 98 que tiene DirectX, carece de soporte para sonido y no podrá funcionar correctamente por esta limitación. En el sitio web del proyecto QEMU se afirma que ya no soporta ningún Windows que sea anterior a Windows 2000.
- Algunos juegos y programas de Windows para hacer pruebas
- Mucho tiempo, el proceso de instalación puede durar varias horas.

Instalación

En primer lugar, tenemos que instalar el software necesario y configurarlo. Luego instalar Windows 2000 en un entorno x86 emulado por QEMU.

```
# apt-get install qemu-ODROID
```

Comenzamos por crear una carpeta donde trabajaremos. Decidí crear una carpeta llamada Windows en la carpeta home del usuario ODROID y coloqué todo lo que necesitaba, así que copié la ISO de mi CD de instalación de Windows 2000 a ese directorio. Después, cree una nueva imagen de disco para nuestra instalación de Windows 2000:

```
# qemu-img create -f qcow2 win2k.
img 10G
```

Una imagen qcow2 aumenta lentamente de tamaño y no coge 10 GB desde el principio como lo hace el formato RAW. Además, ofrece funciones avanzadas como la captura de pantalla. Windows sólo necesita 1-2 GB de espacio, así que con 3-4 GB debería bastar. Después, desarrolle un script que me permitía iniciar la máquina virtual QEMU con más facilidad. Lo hice ejecutable y lo coloqué en la carpeta Windows:

```
# /home/ODROID/Windows/start_qemu
#!/bin/bash
```

```
export QEMU_AUDIO_DRV=pa
qemu-system-i386 -hda ./win2k.img
-cdrom ./win2k.iso -boot d -m 512
-localtime -soundhw ac97 -moni-
tor stdio -net nic,model=ne2k_pci
-net user -vga std -cpu pentium
-no-ahci
```

Los parámetros usados son estos:

QEMU_AUDIO_DRV=pa

Este parámetro especifica el uso de PulseAudio como controlador de sonido predeterminado. Funciona con ALSA también, pero PulseAudio parece ser más rápido con QEMU que ALSA. Otras opciones disponibles son “sdl” y “alsa”.

qemu-system-i386

Esto le dice al sistema que estamos ante un sistema compatible con i386 con todo lo que conlleva, BIOS y hardware. También existe un binario qemu-i386, que se puede utilizar para iniciar un único binario i386 en lugar de emular un PC por completo, pero nosotros no lo vamos a utilizar.

-hda ./win2k.img

Esto especifica que la imagen del disco duro es el archivo win2k.img que hemos creado anteriormente.

-cdrom ./win2k.iso

Similar al parámetro de imagen del disco duro, especifica el archivo ISO del CD de instalación de Windows 2000.

-boot d

Esto le dice al sistema que arranque desde la unidad D:, que en este caso es nuestra ISO de Windows 2000. La ISO del Windows 2000 debe ser una imagen de auto-arranque. Con versiones anteriores como Windows 95, a menudo era necesario un disquete para arrancar y cargar un driver de CD-ROM para acceder al CD e instalarlo desde allí. Con las versiones posteriores de Windows, esto ya no era necesario, por lo que podemos arrancar directamente desde el CD. Una vez instalado, se saltará el CD y arrancará desde el disco duro a menos que se pulse una tecla.

-m 512

Este parámetro da a la máquina virtual 512 MB de RAM para su uso. Puedes utilizar 768MB o 1024, pero puede que tenga que activar swap o zram para esto, ya que se utiliza más de 1.024 MB de RAM realmente. Puedes también utilizar sólo 128 o 256 MB, Windows 2000 también funcionará con menos RAM.

-localtime

Esto le indica a la máquina virtual que utilice la hora del sistema host.

-soundhw ac97

Esto activa la tarjeta de sonido Intel 82801AA AC97 virtual. Existen otros modelos, pero AC97 funciona bien.

-monitor stdio

Esto le indica a la máquina virtual que incluya una consola de línea de comandos, que se puede usar para cambiar los parámetros de la máquina virtual, como la instalación de otras imágenes de disco o CD.

-net nic,model=ne2k_pci -net user

Esta es la configuración de la red, que no he llegado a probarla realmente.

-vga std

Utiliza el estándar VGA con extensión Bochs VBE, que es una tarjeta gráfica avanzada que ofrecen más funciones que la opción por defecto. Como alternativa y para una mayor compatibilidad aunque con menos características, puedes utilizar la opción “-vga cirrius”, que utiliza un Adaptador gráfico Cirrus Logic GD5446 VGA virtual.

-cpu pentium

Esto especifica la CPU que se debe utilizar. Hay disponibles diferentes tipos, como el Pentium, Pentium2, pentium3 y 486. Nosotros usaremos Pentium2 para la instalación, aunque más adelante es posible que quieras cambiar a qemu32, que fue desarrollada como una CPU de 32 bits directamente para QEMU, y así poder usar opciones avanzadas tales como “-smp 2”, que crea un PC con 2 CPUs. También puedes ser más específico con opciones como “-smp 2,cores=2,threads=1,soc kets=1”. Utilizando “qemu-system-i386-cpu?”, puedes obtener una lista de todos los modelos de CPU disponibles.

-no-acp

Esta opción desactiva el soporte para ACPI (Interfaz Avanzada de Configuración y Energía), ya que esto puede causar problemas con la detección del hardware

Instalación de Windows 2000

Tras haber copiado tu win2k.iso a la misma carpeta que el script de inicio y creado el win2k.img en la misma carpeta, puedes iniciar QEMU con el siguiente comando despues de navegar hasta su carpeta:

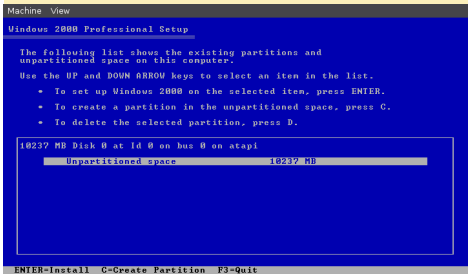
```
$ ./start_qemu
```

Una nueva ventana aparecerá de golpe, y verás el instalador de Windows 2000.

La creación de particiones y la copia de los archivos de instalación de Windows irán relativamente rápido, finalizando el proceso en un par de minutos. Posteriormente, el sistema se reiniciará y empezará a configurarse.

Esta parte se bastante larga, así que siéntate, crúzate de brazos y deja que ODROID haga su trabajo. Es posible

Figura 1 - Seleccionando la partición de nuestra instalación de Windows 2000



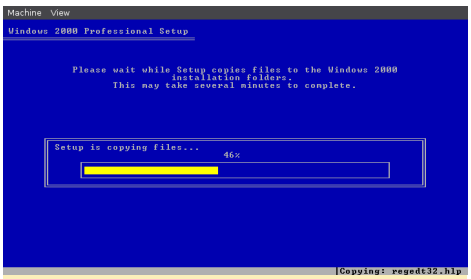


Figura 2 - Copiando archivos de instalación

Figura 3 - La pantalla de inicio de Windows 2000 me trae buenos recuerdos de un sistema operativo muy rápido y estable que he usado durante muchos años

que te dé tiempo a ver una película o un par de episodios de tu serie favorita. La CPU de la ODROID no será utilizada demasiado durante este proceso, así que puedes utilizar tu ODROID para ver una película, escuchar música o navegar por la web. Ten en cuenta que si has elegido mucha RAM para la máquina virtual, puede que te quede poca RAM, de modo que navegar por sitios web como YouTube u otros sitios podría consumir muy rápidamente lo que queda de RAM. Aún no estoy seguro de la causa que provoca esta gran demora, pero he apreciado este mismo comportamiento en todos los emuladores x86 como DOSBox, QEMU, y ExaGear. Todos ellos parecen ser muy lentos en la lectura y escritura, aunque no usan el 100% de la CPU. Después de un tiempo, el sistema iniciará la instalación de los drivers y tratará de detectar automáticamente el hardware del sistema QEMU.

Al rato, la detección de hardware finalizará y el sistema te realizará un par de preguntas sobre la distribución del teclado, la configuración de red, la contraseña de administrador, etc. Sigue las

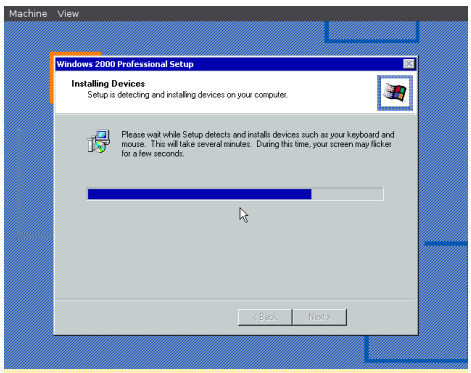


Figura 4 – La Auto detección del hardware en el entorno QEMU puede ser un proceso bastante largo

instrucciones y espera a que se complete el proceso, que puede tardar más de una hora. Se paciente y tras otro reinicio, finalmente aparecerá una pantalla de inicio de sesión seguido del adorable sonido de la intro de Win 2000 y el escritorio.

Llegados a este punto, tenemos un escritorio ejecutando Windows 2000, pero si miras de cerca, notarás que la pantalla actual es de sólo 640x480 y de 16 colo-

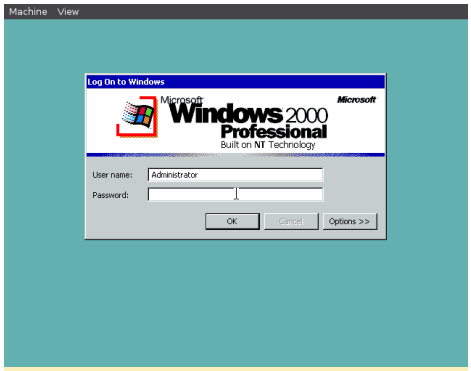


Figura 5 - Pantalla login de Windows 2000

Figura 6 - Escritorio de Windows 2000 tras su primer arranque

res, así que vamos a intentar arreglar esto copiando drivers adicionales en el sistema. Yo uso 7zip y Universal VESA/VBE Video Display Drivers para Windows NT de http://bit.ly/200rAyC. Ambos programas puedes localizarlos dentro de mi repositorio en http://bit.ly/20pYup0. Hay varias formas de introducir estos archivos en el sistema. Quisiera hablar de dos, que funcionan en casi todos los sistemas y pueden ser útiles para otros propósitos. La primera es crear tu propio archivo ISO y poner los archivos que necesitas dentro de éste, luego montar la ISO en QEMU. El segundo es montar directamente el archivo del disco duro QEMU en tu entorno Linux para poder realizar directamente operaciones de archivos sobre él.

Creando una ISO

Si queremos cambiar la ISO que está en la unidad CD de nuestra máquina virtual QEMU, primero tenemos que crear un nueva ISO con los archivos que queremos ver en nuestro sistema. Para esto, instalaremos un programa llamado "genisoimage", que ya podría estar incluido en tu distribución de Linux.

```
# apt-get install genisoimage
```

Crea una nueva carpeta, como "myfiles" en tu disco duro, Después, copia todos los archivos necesarios en esa carpeta. También puedes poner los juegos y programas que quieres instalar más adelante. Usa la siguiente línea de comandos para crear una ISO de esta carpeta:

```
$ Genisoimage -o myiso.iso \
  -V MYISO -r -J myfiles/
```

Este comando creará un archivo llamados nyiso.iso con los ficheros y subcarpetas incluidas en la carpeta "myfiles", asegúrate de sustituir "myfiles" por el nombre de la carpeta que creaste. Luego, necesitamos montar la ISO en nuestra máquina virtual QEMU. Ya sabemos cómo definir el archivo ISO con el script de línea de comandos que hemos creado, sólo necesitamos reemplazar el archivo ISO en la línea de comandos, pero quisiera mostrar un método diferente.

Cuando creamos esta línea de comandos, añadimos el parámetro "monitor stdio", que nos permite introducir comandos desde el terminal para controlar la máquina virtual QEMU.

Si no te has dado cuenta de esto todavía, deberías navegar a la ventana de terminal que usaste para iniciar tu máquina virtual QEMU, debería mostrar un prompt "(QEMU)" indicando que se encuentra a la espera de un comando. Haz clic en esta ventana de terminal, luego pulsa INTRO una o dos veces para ver si está funcionando. Si el puntero del ratón se te engancha en tu máquina virtual Windows, sólo tiene que pulsar "CTRL + ALT + G", y así conseguirás que el ratón vuelva y puedas usar este terminal. Si escribes "help" obtendrá una lista de diferentes opciones para interactuar con la máquina virtual. Qeremos montar una nueva ISO, que puede hacerse con el siguiente comando:

```
(qemu)change ide1-cd0 /home/
ODROID/Windows/myiso.iso
```

Reemplaza la ruta del archivo ISO por el archivo que creaste en los pasos anteriores. Comprueba que el archivo ISO esta exactamente en su lugar con el siguiente comando:

```
(qemu)info block
```

Incluso puedes ser un poco más delicado con el sistema, y en lugar de cambiar los CDs directamente, puede expulsar el CD primero:

```
(qemu)eject ide1-cd0
(qemu)change ide1-cd0 /home/
ODROID/Windows/myiso.iso
(qemu)info block
```

Una vez que hayas hecho esto abre "Mi PC" en tu máquina virtual Windows 2000, debería ver el CD que has creado y pdoras interactuar con la imagen.

Montar el archivo en el disco duro

Los archivos ISO son muy flexibles y permite cambiar rápidamente de programas. Además son bastante rápidos, de modo que por lo general son una buena opción. Sin embargo, una desventaja de esta opción es que el CD siempre estará protegido contra escritura, lo que significa que los archivos que se encuentran en el CD no pueden ser extraídos directamente en éste, siempre tenemos que copiar su contenido al disco duro en primer lugar. Puesto que procede de un CD, todo lo que copiamos de éste automáticamente estará protegido contra escritura, así que hay que cambiar los permisos antes de utilizar los archivos.

Otro método consiste en copiar los archivos directamente sobre la imagen del disco duro de tu sistema QEMU. Aunque esto se puede hacer mientras el sistema está ejecutandose, es mejor hacerlo cuando el sistema está apagado, de modo que apaga el sistema Windows o cierra la ventana directamente, y abre un terminal con privilegios de root.

Si has creado una imagen de disco en formato RAW, es probable que puedas montarla directamente con el comando:

```
# mount -o loop,offset=32256
win2k.img /mnt
```

Después, deberías ser capaz de acceder a la imagen de disco bajo el directorio /mnt directamente y realizar operaciones de archivo sobre él como crear carpetas y copiar archivos. No olvides desmontar la imagen una vez que hayas terminado:

```
# umount /mnt
```

Si creas una imagen qcow2 o el método anterior no te funciona, puedes probar el siguiente método

```
# modprobe nbd max_part=16
# qemu-nbd -c /dev/nbd0 ./win2k.
img
# partprobe /dev/nbd0
# mount /dev/nbd0p1 /mnt
```

Después de esto, también podrás realizar operaciones de disco en /mnt, que almacena los contenidos de tu imagen. Una vez hayas terminado, recuerda que debe desmontar el dispositivo y también cerrar la conexión a la imagen del disco:

```
# mount /mnt
# qemu-nbd -d /dev/nbd0
```

Ajustar los graficos

No importa cómo copies los archivos en tu sistema, pero debes terminar con el instalador de 7zip y el archivo vbempj.zip en tu máquina virtual Windows 2000. El siguiente paso es instalar 7zip y luego, extraer el archivo zip. Una vez completado, haz clic derecho en "Mi PC" y selecciona Propiedades. En la nueva ventana que se abre, selecciona la pestaña Hardware y haz clic en el botón Administrador de dispositivos.

Debería ver un signo de interrogación de un adaptador VGA estándar, como

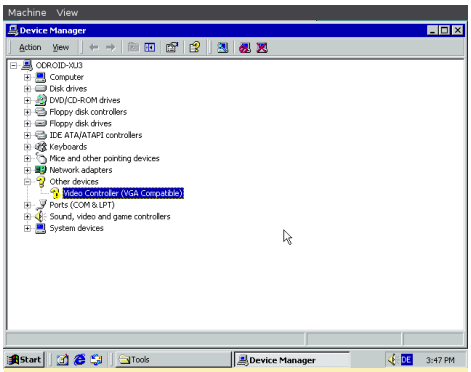


Figura 7 – Drivers VGA de la tarjeta grafica "std" de QEMU no localizados

muestra la Figura 7. Haz doble clic en él para abrir la ventana de propiedades, luego haz clic en el botón Reinstalar controlador. Cuando se te pregunte, selecciona "Buscar un controlador apropiado para mi dispositivo (recomendado)", y pincha en el botón Siguiente. Marca la opción "Especificar una ubicación" y pincha en Siguiente de nuevo. Aparecerá una ventana pop-up, preguntando dónde buscar el driver. Haz clic en el botón Examinar y navega hasta la car-

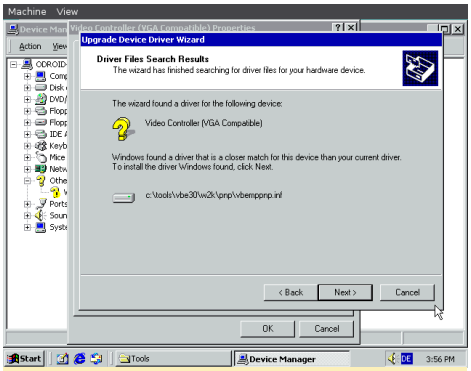


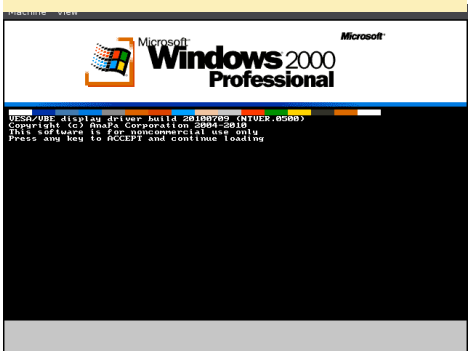
Figura 8 - Windows encontró los drivers VGA para el adaptador de gráficos

peta donde extrajiste el archivo vbempj.zip. Debería ver un par de carpetas, ve a VBE30/W2K/PNP/ y selecciona el archivo vbemppnp.ini. Confirma la selección y deja que Windows busque el driver. Tras una corta espera, deberías ver que se han encontrado los drivers y que están listos para ser instalados.

Deja que finalice la instalación de los drivers. Windows parpadeara un par de veces al activar los nuevos drivers. Aunque el sistema no te lo indica, reinicia el sistema llegados a este punto para cargar correctamente los nuevos drivers VGA. Una vez completado el reinicio, verás una notificación que indica que este driver es para uso no comercial

Aunque te indica que pulses cualquier botón, simplemente espera un poco y la notificación desaparecerá por sí sola. Inmediatamente tras iniciar sesión, deberías notar la diferencia en la calidad de vídeo. Ahora puedes modificar los ajustes de la pantalla con sólo pulsar el botón derecho del ratón en un espacio vacío del escritorio y seleccionando Propiedades. Cambia a la pestaña Configuración y ajusta tu resolución de pantalla.

Figura 9 – Notificación de los nuevos drivers VGA



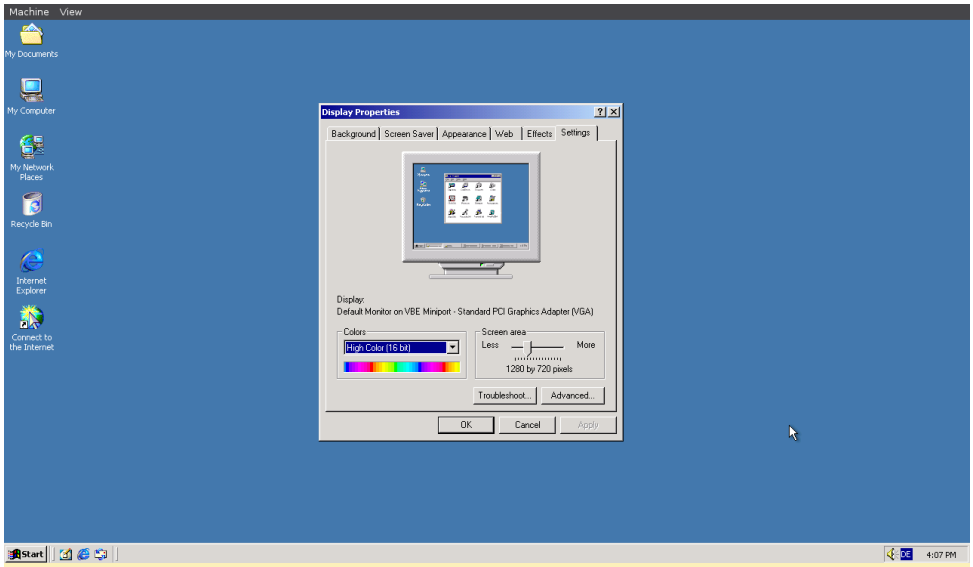


Figura 10 - Gracias al nuevo driver VGA puede ejecutar Windows 2000 en 720p, 1080p o en resoluciones superiores

Aunque nada te impide que elijas 16, 24 o 32 bits de color para tu máquina de Windows 2000, descubrí que todos excepto 256 colores corrompen el menú Inicio y queda inservible. Aunque esto posiblemente no es un problema, ya que puedes ejecutar programas y juegos directamente desde el disco duro, hace que sea casi imposible apagar o reiniciar el equipo correctamente, por lo que tiendes a utilizar 256 colores en su lugar. Esto tiene un inconveniente: la relación de pantalla 4: 3 sólo soporta 256 colores.

Figuras 11 y 12 - Age of Empires I ejecutándose a través de una máquina virtual QEMU con Windows 2000 en un ODROID

Entonces, ¿Qué se puede hacer con esta máquina virtual de Windows 2000? ¡Echa un vistazo a las Figuras 11 y 12!

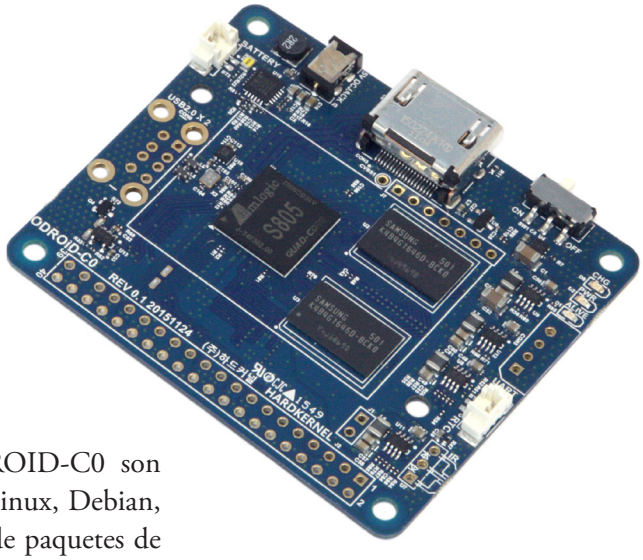
No esperes demasiado rendimiento de tu máquina virtual, porque es bastante lenta. De hecho, en mi anterior prueba, yo ejecute Windows 98 que era más rápido que Windows 2000. Sin embargo, como he mencionado anteriormente, con Windows 98 no fue capaz de activar el audio en absoluto. Es probable que puedas acelerar un poco las cosas usando una CPU diferente o con más memoria RAM. Ahora que sabes lo básico, puedes experimentar con diferentes configuraciones.

Ahora puede instalar otros programas como Microsoft Office, WinAmp o cualquier aplicación de Windows que quieras probar. Si te gusta la aventura, puede utilizar los mismos métodos para instalar Windows 95, Windows 98, Windows ME, o incluso Windows XP, e intentar buscar la opción que mejor se adapte a tus necesidades.

ODROID-C0

UNA COMPACTA PLACA PARA APLICACIONES LIGERAS Y PORTATILES

por Justin Lee



El ODROID-C0 es un equipo pensado para aquellos que desean hacer aplicaciones más flexibles y portátiles. Es una versión minimizada del conocido ODROID-C1, que se adquiere en la tienda de Hardkernel en http://bit.ly/1WFYKOL por 25$. Es ideal para proyectos de Internet de las cosas (IoT), que puedan llevarse encima o para otras aplicaciones que requieran un dispositivo ligero como los drones, como se muestra en la Figura 2.

Todas las imágenes del ODROID-C1/C1+ son totalmente compatibles con el ODROID-C0. Algunos de los modernos sistemas operativos que se pueden ejecutar en el ODROID-C0 son Ubuntu, Android, Arch Linux, Debian, y OpenELEC con miles de paquetes de software de código abierto gratuitos. También existen un montón de sistemas operativos personalizados para utilizar en el ODROID-C0 como un centro multimedia, Kodi, estación de juegos, servidores y mucho más en los foros ODROID en http://bit.ly/1SEUP5p.

Figura 1 - ODROID-C0 con una batería

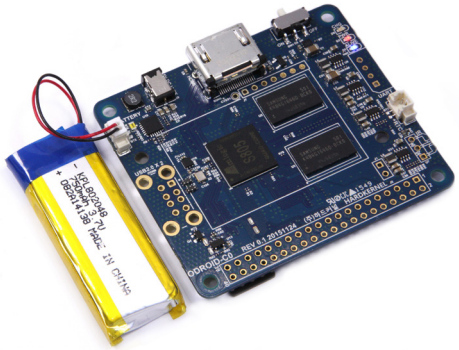


Figura 2 – Ejemplo de proyecto de un drone usando un ODROID-C0

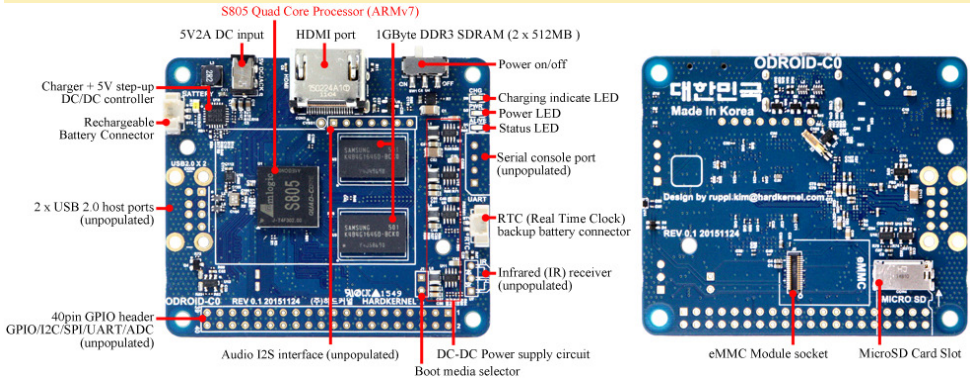
Especificaciones técnicas

Las principales características y mejoras sobre el ODROID-C1 original se detallan a continuación:

- CPUs Amlogic ARM® Cortex®-A5(ARMv7) 1.5Ghz quad core
- GPU Mali™-450 MP2 (OpenGL ES 2.0/1.1 activado para Linux y Android)
- 1Gbyte DDR3 SDRAM
- Ranura para almacenamiento Flash eMMC4.5 HS200/Ranura para tarjetas microSD SDR50 UHS-1
- 40pin + 7-pin GPIOs (deshabilitados)
- 2 x USB 2.0 Host (deshabilitados)
- Receptor de infrarrojos (IR) (deshabilitado)
- Cargador de batería recargable Li+ para aplicaciones portátiles y robots
- El nivel de voltaje de la batería es accesible vía ADC en el SoC
- Transformadores DC con una mayor eficiencia energética
- Transformador DC para raíles de 5 voltios (USB host y HDMI) desde una batería de Li-polimero
- Pack de Conectores C0 de bricolaje para la creación de prototipos a mono (descrito más adelante)
- Circuito de energía de batería totalmente integrado, de modo que la unidad puede ser movil conectando una batería de 3,7 V Li+

Para más información, por favor dirigite a la página de detalles técnicos en http://bit.ly/1RGqrrl.

Figura 3 – Detalles de la placa



Documentación

La wiki del ODROID-C0/C1/C1+ en http://bit.ly/1KRKoGV contiene in-

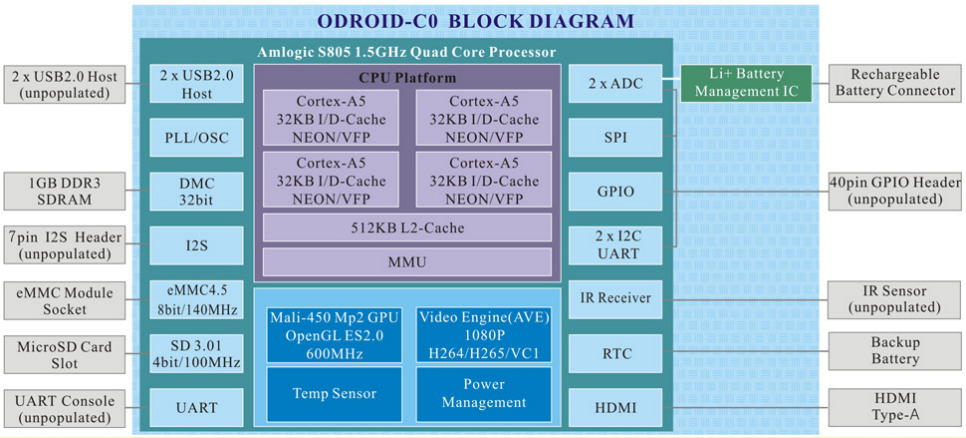


Figura 4 - Diagrama por bloques

formación sobre los sistemas operativos, software y periféricos disponibles para el dispositivo. Los esquemas completos pueden descargarse desde http://bit.ly/1KxguEX, el diseño de la electrónica están en http://bit.ly/1QlLP1F, y la hoja de datos está disponible en http://bit.ly/1dFEHhX.

Datos técnicos

Procesador

Amlogic S805 Quad Core Cortex™-A5 GPU Mali™-450 Dual Core

RAM

1GByte DDR3 de 32bit Samsung K4B4G1646D (2 x 512MByte)

Ranura para módulos eMMC

8GB/64GB: Toshiba eMMC

16GB/32GB: Sandisk iNAND Extreme

El tiempo de acceso de los sistema de almacenamiento eMMC es 2-3 veces más rápido que la tarjeta SD. Existen 4 opciones de tamaño: 8 GB, 16 GB, 32 GB y 64 GB. El uso de un módulo de eMMC aumentará en gran medida la velocidad y capacidad de respuesta, similar a la que se experimenta cuando se cambia a una unidad de estado sólido (SSD) en un típico PC, además mejora el rendimiento con respecto a un disco duro mecánico (HDD).

Ranura para tarjetas Micro Secure Digital (microSD)

El ODROID-C0 puede utilizar los nuevos modelo SD UHS-1, que son aproximadamente 2 veces más rápido que una tarjeta normal de la clase 10.

Entrada DC 5V2A

La entrada DC está diseñado para soportar una potencia de 5V, con un diámetro interior de 0,8 mm y un diámetro exterior de 2,5 mm. El ODROID-C0 consume menos de 0,5A en la mayoría de los casos, pero puede subir hasta 2A si se conectan muchos periféricos USB pasivos directamente a la placa.

Conexión de la batería recargable

Esto es para la conexión de la batería de Li-polímero o Li-ion de 3.7V. Incorpora un indicador LED de carga, que se enciende cuando la batería se está cargando o está suelta. Existe un interruptor deslizante que se utiliza para encender/apagar la placa, y el circuito de carga funciona incluso si la placa está apagada.

Conector de Batería

Cabezal 1,25 mm Molex 53398-0271, montaje en superficie, de tipo vertical (compañero de Molex 51021-0200)

Puertos host USB

Incorpora dos puertos host USB 2.0. Se pueden conectar teclados, ratones, adaptadores WiFi, sistemas almacenamiento y muchos otros dispositivos. ¡También puede cargar tu smartphone con ellos! Si necesita más de 2 puertos, puedes utilizar un hub USB externo auto-alimentado reduciendo además, la carga de potencia en la placa.

Puerto HDMI

Conector HDMI estándar Tipo-A para la salida de vídeo/audio.

LEDs de estado/Potencia

El ODROID-C0 tiene tres indicadores LED que proporcionan información:

1. El LED rojo indica si la placa está conectada a una corriente de 5V.
2. El LED azul, cuando se mantiene fijo, indica que el programa U-boot está ejecutándose. Cuando parpadea como el latido de un corazón, es que el núcleo está funcionando.
3. El LED verde, cuando se mantiene fijo, indica que la batería se está cargando. Cuando la luz se apaga, la batería está completamente cargada. Si se observa un rápido parpadeo, la batería no está conectada o no está funcionando correctamente.

Receptor Infrarrojo (IR)

Este es un módulo receptor de control remoto que puede aceptar datos inalámbricos en el estándar de 37.9Khz en formato NEC.

Puertos Entrada y Salida de Propósito General (GPIO)

Estos puertos GPIO de 40 pines se pueden utilizar como GPIO/I2C/SPI/UART/ADC para electrónica y robótica. Los pines GPIO en un ODROID-C0 son un gran forma de interactuar con dispositivos físicos como botones y LED utilizando un controlador de Linux muy liviano. Si eres desarrollador de C/C++ o de Python, existe una biblioteca muy útil llamada WiringPi que te permite inte-ractuar con los pines. Ya hemos exportado la libreria WiringPi v2 a ODROID-C0. Ten en cuenta que los pines #37, #38 y #40 no son compatibles con el cabezal de 40pin de la Raspberry Pi B+, están destinados para funciones de entrada analógica. Todos los puertos GPIO van a 3.3 voltios, pero las entradas ADC están limitan a 1.8 voltios.

MANUAL DE USUARIO ODROID-XU4

UNA GUIA PARA TODOS LOS NIVELES

editado por Rob Roy

El manual de usuario oficial para el ODROID-XU4 ha sido publicado recientemente en el sitio web de Hardkernel, y está disponible para su descarga directa en `http://bit.ly/1U9Q8yg`, a través de los foros en `http://bit.ly/1RykBrT`, y en Google Play Store en `http://bit.ly/1WrIeSd`. El ODROID-XU4 es uno de los más potentes y económicos ordenadores de placa reducida que existen, además de ser un dispositivo muy versátil. Con un procesador Exynos 5422 big.LITTLE octa-core, una avanzada GPU Malí y una conexión Ethernet Gigabit, se puede utilizar como un sistema de cine en casa, un ordenador para navegar por internet, ejecutar juegos y consultar redes sociales, como herramienta de trabajo para el colegió o la oficina, como prototipo para realizar pequeñas modificaciones de hardware, como controlador para proyectos de domótica, como estación de trabajo para programar y mucho más. Algunos de los modernos sistemas operativos que se pueden ejecutar en el ODROID-XU4 son Ubuntu, Android, Fedora, archlinux, Debian y OpenELEC, con miles de paquetes de software de código abierto totalmente gratis. El ODROID-XU4 es un dispositivo ARM, la arquitectura más utilizada en los dispositivos móviles y en la informática de 32-bit embebida.

Puerto de consola serie

La conexión del puerto de consola serie a un PC permite el acceso a la consola Linux. Puedes ver el registro del arranque o conectarte al C0 para cambiar la configuración de vídeo o de red. El UART serie utiliza una interfaz de 3,3 voltios, por lo que recomendamos usar el kit USB-UART de Hardkernel, disponible en http://bit.ly/1nhQuIm, que es 100% compatible con la interfaz.

RTC (Reloj en Tiempo Real)

Hay un conector de batería de reserva si quieres añadir funciones RTC para registrar o mantener la hora cuando la placa esté desconectada, conectando una batería de reserva de litio tipo botón (CR2032 o equivalente). Todos los circuitos RTC están incluidos en el ODROID-C0 por defecto. Es un Molex 53398-0.271

Cabezal de 1.25mm, Montaje superficial tipo vertical (compañero de Molex 51021-0200)

Controlador VBUS USB

IC de protección NCP380 para la fuente de alimentación USB de OnSemi.

Selector de soporte de arranque

Si éste está abierto, el primer soporte de arranque será siempre eMMC. Si está cerrado, el primer soporte de arranque será la tarjeta SD.

Interruptor de encendido/apagado

Puede activar/desactivar la alimentación del sistema del ODROID-C0 usando este interruptor. El circuito de carga funciona independientemente del estado del interruptor.

Circuito de alimentación

Los LDOs y los transformadores DC-DC diferenciados se utilizan para alimentar CPU/DRAM/IO. El IC MP2637 se utiliza para implementar la función de carga y de arranque a 5V.

Pack de conectores

El pack de conectores ODROID-C0 es perfecto para hacer proyectos de bricolaje (DIY). Se necesita algunos conocimientos de soldadura para poder utilizar el pack y su coste es de 1.80$. La mayoría de los conectores son idénticos a los desarrollados para el ODROID-C1+.

Hay dos conectores host USB diferentes en este pack. Si desea crear una placa con un bajo perfil, utilizar el conector de un sólo puerto. Si quieres tener conectividad completa, utiliza el conector de doble puerto.

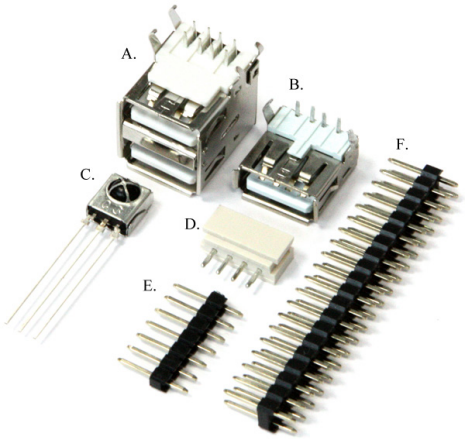


Figura 5 – Detalles del Pack de conectores

A. Conector macho de 8-pin DIP USB Tipo-A de doble puerto

B. Conector macho de 4-pin DIP USB Tipo-A de un sola puerto

C. Sensor receptor de infrarrojos (IR). Este es un módulo receptor de control remoto que puede aceptar datos inalámbricos basados en la frecuencia 37.9Khz estándar.

D. Conector consola UART

Un Molex 5268-04a (cabezal de 2,5 mm) está montado en la PCB. Su compañero es Molex 50-37-5043 Wire-to-Board Crimp Housing.

E. Cabezal macho de 7-pin

Añade pines a la interfaz I2S de audio en el ODROID-C0.

2.54mm (0.1inch) pitch

F. Cabezal macho de 20x2pin

Añade pines al puerto GPIO de 40-pin

SO DESTACADO:

ODROBIAN RETROGAMING ARCADE (ORGA) PARA ODROID-C1+

Una completa solución para emular juegos, ORGA conservará todos tus juegos

He publicado recientemente mi sistema operativo para juegos y entretenimiento llamado Odrobina Retro Gaming Arcade (ORGA). Utiliza los drivers GPU Direct Frame Buffer que he compilado específicamente para la retro-emulación y el sistema multimedia. Esta basado en el centro multimedia Kodi usando EmulationStation como interfaz principal, junto con algunos aspectos de RetroPie. ORGA está desarrollado bajo la siguiente fórmula:

ORGA = (Mali-GPU-Fbdev + SDL2-Fbdev) * (EmulationStation-Fbdev + RetroArch-Fbdev + Kodi-Fbdev) / (Handy X11 GUI)

Las características que incluye ORGA son:

Sistema de entretenimiento completamente temática con EmulationStation V2 preconfigurado

Compilado para Mali-FBdev garantizando el máximo rendimiento de FPS posible con la mejor eficiencia de la CPU

RetroArch acelerado por hardware configurado con drivers de vídeo SDL2 como dispositivo de salida para Mali-FBdev

EmulationStation y RetroArch utilizan SDL2 con una profundidad de color de 32 bpp sin problemas de translucidez ni mezcla de gráficos

Núcleos de emulación RetroArch pre-compilados y preinstalados

Kodi 15,2 Isengard compilado y optimizado para VPU Amlogic con OpenGLES v2, configurado para EmulationStation

Incrementada la RAM lógica pasando de 1 GB a 2 GB

Todas las aplicaciones de entretenimiento se pueden lanzar directamente desde consolas TTY

Puedes ejecutar la interfaz gráfica de usuario X11 desde la consola en cualquier momento con los drivers Mali X Vídeo

Las actualizaciones están disponibles directamente desde el repositorio de Odrobian en http://bit.ly/1OTGdck

Todo el software esta compilado para Odrobian Jessie (Debian 8.2) con parámetros de optimización.

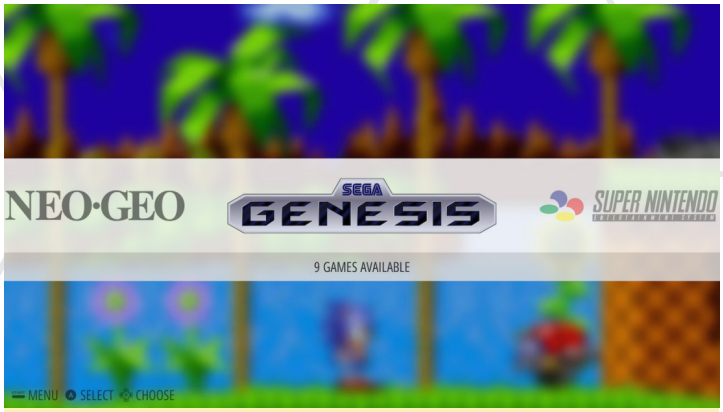
Ten en cuenta que este proyecto sigue en desarrollo, así que se aceptan comentarios, sugerencias y posibles mejoras.

Empecemos

Este software está basado en la distribución Odrobian Jessie MATE, la cual se puede convertir a un sistema de entretenimiento ORGA con un único comando. Para ello, visita el hilo del foro titulado "Odrobian Jessie for ODROID-C1/C1+" en http://bit.ly/1ZVzERa para descargar la última edición de MATE compatible. Después, graba la imagen en tu soporte de arranque y arranca el sistema con ella. Inicia una sesión de terminal (Ctrl + Alt + T) y ejecuta los siguientes comandos para instalar ORGA:

```
$ sudo -s
# apt-get update && apt-get install \
  odrobian-platform-s805
# insta-ORGA
```

Espera hasta que el ODROID-C1+ se reinicie automáticamente y aparezca una consola TTY1. El escritorio X11 se desactivará y zram se activará automáticamente. A continuación, actualiza el kernel con el comando:



Captura de pantalla de Odrobina EmulationStation

```
$ sudo apt-get dist-upgrade
```

Ten en cuenta que es posible que ya tengas el último software y paquetes actualizados al utilizar el comando anterior "insta-ORGA"

Consejos prácticos

Siempre conéctate al sistema con el usuario "odroid" y no como "root". Cuando sea necesario utiliza "sudo" para activar los permisos necesarios. La contraseña de usuario "odroid" también es "odroid".

Siempre inicia EmulationStation desde TTY1 (Ctrl + Alt + F1), que es la consola por defecto en el arranque. Si quieres la interfaz gráfica de usuario X11 es preferible iniciarla a través TTY3 (Ctrl + Alt + F3). Puede utilizar la interfaz gráfica de usuario X11 para navegar por Internet utilizando el navegador Chromium, o para configurar tus ROMs. A los archivos BIOS y Emuladores se puede acceder desde el entorno de escritorio MATE Odrobian con la aplicación File Manager.

Puedes hacer que Odrobina auto-inicie EmulationStation directamente tras el proceso de arranque editando el archivo "/etc/rc.local" e incluyendo el siguiente comando justo por encima de (antes de) la expresion "exit 0":

```
runuser -l odroid -c 'emulationstation'
```

Los drivers del dispositivo Direct FrameBuffer mostrará texto cuando abrimos archivos multimedia y al mismo tiempo ejecutará algunas funciones de vídeo. Si no desea ver el texto, puede iniciar Kodi como aplicación de interfaz gráfica de usuario a través del X-server usando el siguiente comando:

```
$ startx /usr/local/bin/kodi
```

También puedes evitar EmulationStation si va a utilizar el sistema únicamente como un centro multimedia haciendo que Odrobina auto-inicie Kodi directamente una vez que el proceso de arranque haya finalizado. Para ello, incluye el siguiente comando justo por encima (antes) de la expresión "exit 0" del archivo "/etc/rc.local":

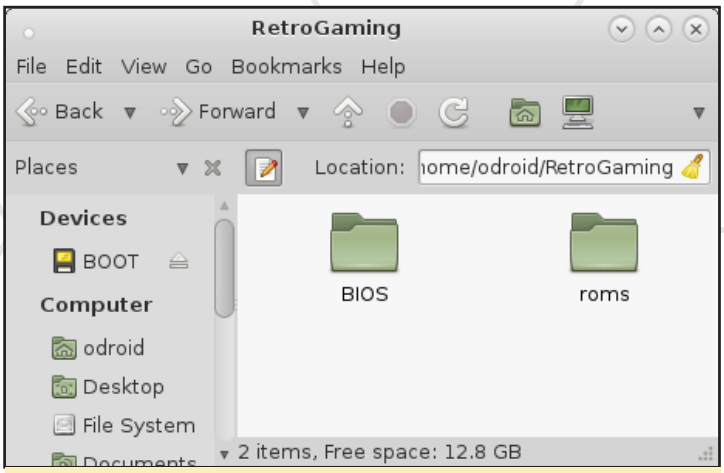
```
runuser -l odroid -c 'kodi'
```

Juegos retro

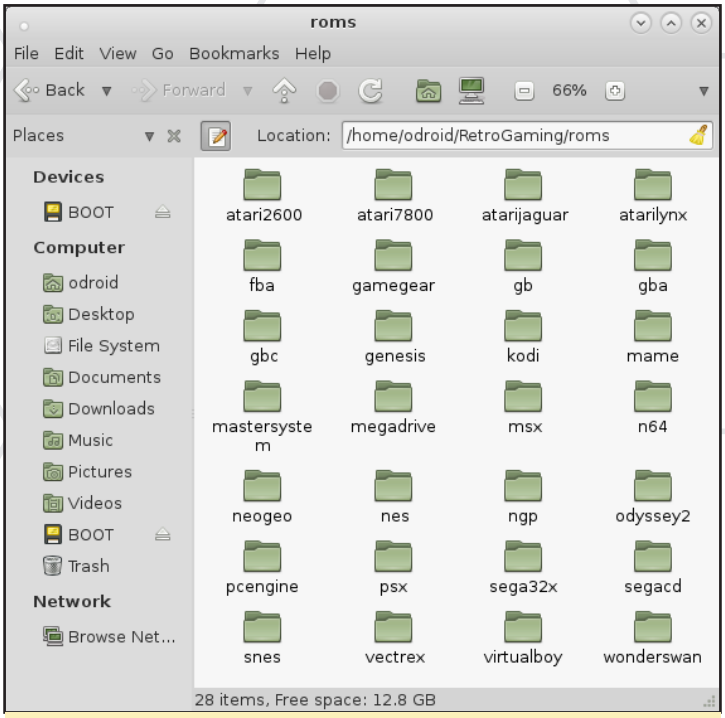
Puede cambiar a TTY3 (Ctrl + Alt + F3) y así podrás utilizar la GUI X11 para configurar el sistema. Para iniciar X11, ejecute el siguiente comando:

```
$ startx
```

Como se muestra en la siguiente imagen, debería ver el directorio del sistema de emulación en la carpeta de inicio principal en /home/ODROID/Retrogaming, que es donde vas a



Carpeta de inicio de las ROMs



ROMs

colocar tus archivos. Por ejemplo, si tienes un archivo BIOS para emular PCSX (PS1), puedes colocarlo directamente dentro de la carpeta ~/Retrogaming/BIOS, y será detectado automáticamente. Las ROMs de juegos actualmente se ejecutan como núcleos RetroArch por defecto. Las ROMs adicionales se irán añadiendo en el futuro a medida que estén disponibles.

Si quieres iniciar tu interfaz EmulationStation, debes volver a la consola TTY1. Pulsa las tecla de acceso directo para TTY1 (Ctrl + Alt + F1), luego ejecuta el siguiente comando:

```
$ emulationstation
```

Emuladores

He probado con éxito los siguientes juegos y emuladores:

Crash Bandicoot (Play Station I) - Máximo (60 FPS)
Super Mario 64 (Nintendo 64) - Máximo (60 FPS)
Sega MasterSystem - Máximo (60 FPS)
Sega MegaDrive - Máximo (60 FPS)
Sega Genesis - Máximo (60 FPS)
GameBoy - Máximo (60 FPS)
Atari2600 - Máximo (60 FPS)
Atari7800 - Máximo (60 FPS)
Atari Jaguar - Máximo (60 FPS)
Atari Lynx - Máximo (60 FPS)
NES - Máximo (60 FPS)
SNES - Máximo (60 FPS)

Tras añadir las ROMs en la carpeta por defecto de tu emulador deseado, deben mostrarse automáticamente en el centro multimedia Kodi. EmulationStation te pedirá que configures tu joypad o teclado conectado para su uso multimedia.

Desarrollo de funciones

Si deseas contribuir en este proyecto con nuevas características y funcionalidades, puedes encontrar el código fuente y los detalles de la configuración en las siguientes ubicaciones:

Sistemas EmulationStation:
`/etc/emulationstation/es_systems.cfg`
Temas EmulationStation:
`/etc/emulationstation/themes/`
Nucleos RetroArch:
`/usr/local/share/retrogaming/cores/`
Configuraciones RetroArch:
`/etc/retroarch/retroarch.cfg`
ROMs de juegos:
`/home/odroid/RetroGaming/roms`
Configuraciones del sistema de juego y BIOS:
`/home/odroid/RetroGaming/BIOS`
Otros:
`/home/odroid/.config/*`

Aquí tienes la lista de los Núcleos RetroArch por defecto proporcionados por el paquete “retroarch-default-cores”:

```
fb_alpha_libretro, fb_alpha_neo_libretro
fceumm_libretro, fmsx_libretro
gambatte_libretro, genesis_plus_gx_libretro
handy_libretro, mednafen_ngp_libretro
mednafen_pce_fast_libretro, mednafen_vb_libretro
mednafen_wswan_libretro, mgba_libretro
mupen64plus_libretro, o2em_libretro
pcsx_rearmed_libretro, picodrive_libretro,
prosystem_libretro
snes9x_next_libretro, stella_libretro
vecx_libretro, virtualjaguar_libretro
```

El árbol de paquetes ORGA está organizado de la siguiente forma:

```
odrobian-retrogaming-arcade
libsdl2-fbdev
libsdl2-2.0-0
libsdl2-dev
es2-fbdev
retroarch-fbdev
retrogaming-default-cores
retrogaming-mame-core
kodi-odrobian-fbdev
mali-fbdev
xboxdrv-odrobian
```

Agradecimientos

Un agradecimiento especial a los siguientes miembros del foro y del equipo HardKernel:

@RetroPie (por tema ES2 y la idea básica)
@memeka (por su ruta SDL2 Mali-Fbdev)
@mdrjr (por repostiorio Odrobian)
@owersun (por el código fuente kodi 15.2 isengard)
@meveric (por soporte técino y referencias)
@robroy (por soporte de la comunidad)

Recomendaciones

Si tiene preguntas, comentarios o sugerencias, por favor visita el hilo original en http://bit.ly/1KjJKif. La imagen original Odrobian Jessie también está disponible en http://bit.ly/1ZVzERa.

MUNIN

UN ANALIZADOR DE RENDIMIENTO DE CODIGO ABIERTO

por Adrian Popa

Munin es una aplicación de código abierto capaz de registrar datos del rendimiento del sistema y de la red. Está escrito en Perl y utiliza bases de datos RRD para crear gráficas que se pueden visualizar a través de una interfaz web. Además de crear gráficas, Munin puede generar alertas por correo electrónico cuando se alcanzan ciertos umbrales. Su objetivo es proporcionar al usuario un punto de partida del estado normal del sistema y así ayudarle a comprender que salió mal, si aparece algún problema.

Munin se puede utilizar en una arquitectura nodo o maestro con múltiples nodos reportando datos de rendimiento hacia el maestro. El nodo maestro almacena los datos en archivos de base de datos RRD y genera las gráficas para facilitar la visualización. Actualmente, Munin tiene más de 500 plugins que están disponibles en bit.ly/1P6YW7T. Los plugins te permite controlar cuestiones básicas como el uso de CPU/memoria, el rendimiento IO, la temperatura del sistema y el rendimiento de interfaz de red, a cosas más complejas como consultas lentas de MySQL o la cola de descarga de Transmisión. La simple API, que se puede descargar desde bit.ly/1OH7COA, se puede usar para crear nuevos plugins y permite a cualquier usuario con conocimientos básicos de Shell crear nuevos tipos de gráficas.

Instalación

Para instalar Munin en un ODROID que ejecute Linux, escribe el siguiente comando:

```
$ sudo apt-get install munin
```

Si lo que desea es configurar el sistema como un nodo, sin una interfaz web, puedes ejecutar esto en su lugar:

```
$ sudo apt-get install munin-node
```

Configuración de la interfaz Web

Una vez que hayas instalado el paquete Munin, necesitas hacer unos pequeños ajustes. La configuración por defecto de Munin se almacena en /etc/Munin. Inicialmente, Munin sólo permite el acceso a la interfaz web desde localhost, así que si quieres acceder a él desde tu LAN, tendrá que hacer algunos cambios.

Si estás ejecutando Ubuntu 14.04, el archivo de configuración Munin para apache es /etc/munin/apache.conf. Por desgracia, su sintaxis está hecha para Apache 2.2, sin embargo Ubuntu 14.04 viene con Apache 2.4. Esto significa que tendrás que añadir la entrada "Require IP 192.168.1.0/24" bajo la directiva "Directory" para que permita el acceso desde 192.168.1.0/24, reemplázalo por tu subred o dirección deseada y también comentar la línea con "Orden allow, deny", o puede ejecutar estos comandos en una sola línea en el terminal:

```
$ sudo sed -i -r 's:^\s+Allow \
  from localhost.*:&\n Require \
  ip 192.168.1.0/24:g' \
  /etc/munin/apache.conf
$ sudo sed -i -e '/Order \
  allow,deny/ s/^#*/#/' \
  /etc/munin/apache.conf
$ sudo service apache2 reload
```

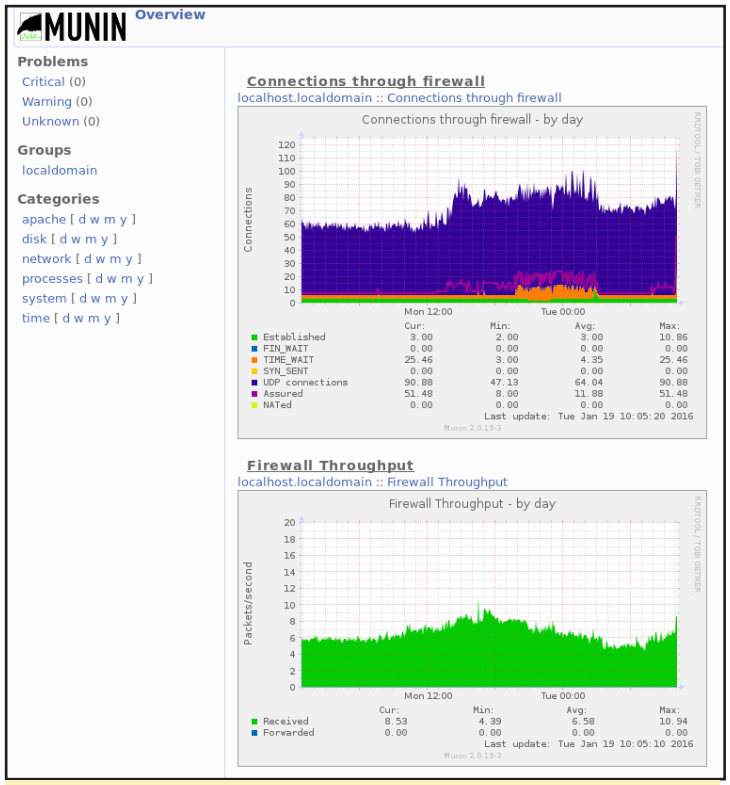
Si en cambio estás en Ubuntu 15.04, edita /etc/monit/apache24.conf y añade tu ip/subred, según sea necesario:

```
$ sudo sed -i 's:Require \
  local:&\n Require ip \
  192.168.1.0/24:g' \
  /etc/munin/apache24.conf
```

Para activar la interfaz web en Apache, puedes crear un enlace simbólico entre la configuración de Munin y la configuración del servidor web:

```
$ sudo ln -s /etc/munin/apache24.conf \
  /etc/apache2/sites-enabled/010-munin.conf
$ sudo service apache2 reload
```

Si navegas a http://<ip-de-tu-odroid> /Munin, debería aparecer una página por defecto donde podrás ver datos de rendimiento.



Esquema general de Munin - Página de la red

Configuración de nodos

Por defecto, Munin inicia la monitorización del sistema local (localhost) y activa una serie de plugins. Deberías tomarte un tiempo para revisar el archivo de configuración y ajustar algunos parámetros como la ruta de acceso a la base de datos RRD o la lista de nodos gestionados. La configuración se encuentra en /etc/munin/munin.conf.

Las entradas dbdir, htmldir y logdir indican rutas dentro de tu sistema de archivos donde se escriben datos con frecuencia, aproximadamente cada 5 minutos. Las lecturas se almacenan en bases de datos Round-Robin (RRD), que es un formato de archivo con una longitud fija donde las entradas más antiguas son sobrescritas por los nuevos datos. Esto significa que las operaciones de escritura guardarán los datos en los mismos sectores del disco, posiblemente reduciendo la vida útil del disco. Si tu sistema se ejecuta desde un módulo eMMC, entonces deberías estar tranquilo, ya que los eMMCs de Hardkernel tienen controladores internos que equilibran el desgaste por su uso. Sin embargo, si tu sistema se está ejecutando desde una tarjeta SD, corres el riesgo de dañarla con el tiempo. Una solución podría ser la de guardar los datos en una unidad USB o en un recurso compartido en red. Además, es posible que desee almacenar los datos en la memoria RAM, utilizando /tmp o una unidad RAM dedicada y escribir en el disco periódicamente, por ejemplo cada hora con el fin de reducir el desgaste. Ten esto en cuenta a la hora de editar la configuración.

Otra parte importante de la configuración es el apartado de los hosts, donde defines qué nodo o nodos deseas supervisar. Por defecto, sólo aparece localhost. Puede cambiar el nombre del host y añadir más equipos ejecutando Munin-node. La comunicación entre el sistema maestro, que es el que ejecuta la interfaz web y el resto de nodos se realiza a través del puerto TCP 4949. Puedes encontrar instrucciones detalladas sobre cómo configurar varios nodos en http://it.ly/1Noa0YJ. Un ejemplo de este apartado podría ser es el siguiente:

```
# a simple host tree
[procyon]
    address 127.0.0.1
    use_node_name yes
```

Una vez que hayas terminado con los cambios, no olvides cargar el servicio Munin-node con el siguiente comando:

```
$ sudo service munin-node restart
```

Por defecto, Munin actualiza las gráficas y los archivos de la interfaz web en intervalos de 5 minutos. Esto garantiza que siempre verás los últimos datos con un mínimo retraso. Sin embargo, a menos que monitorices la interfaz web 24/7, es un desperdicio mantener activadas todas las gráficas. Una mejora que podrías hacer es generar las gráficas sólo cuando accedas a la interfaz web. Esto supondría escribir mucho menos en tu soporte de almacenamiento y un ligero incremento del tiempo de espera cuando naveges por la interfaz. Para ello, sigue las instrucciones de http://it.ly/1Noan5w:

```
$ sudo apt-get install libapache2-mod-fcgid
```

Asegúrate de ajustar las siguientes opciones en el archivo de configuración Munin:

```
graph_strategy cgi
html_strategy cgi
```

Gestión de gráficas

En este punto, deberías ver gráficas de tu sistema o sistemas, sin embargo aún no ha seleccionado los elementos que desea representar gráficamente. Cuando lo instalas por primera vez, Munin analiza tu sistema y activa todos los plugins que parecen ser los más adecuados. Si instalas un nuevo servicio tras tener activado el correspondiente plugin, como el servidor MySQL, Munin no generará automáticamente la gráfica para este servicio. Para corregir esto, debes ejecutar el siguiente comando:

```
$ sudo munin-node-configure --suggest
```

Este comando sugerirá los plugins que puedes activar en función de la configuración actual de tu sistema. También puede ver por qué otros plugins no se han activado, lo cual puede suceder si les faltan dependencias. Realmente para activar los plugins puede ejecutar el siguiente comando:

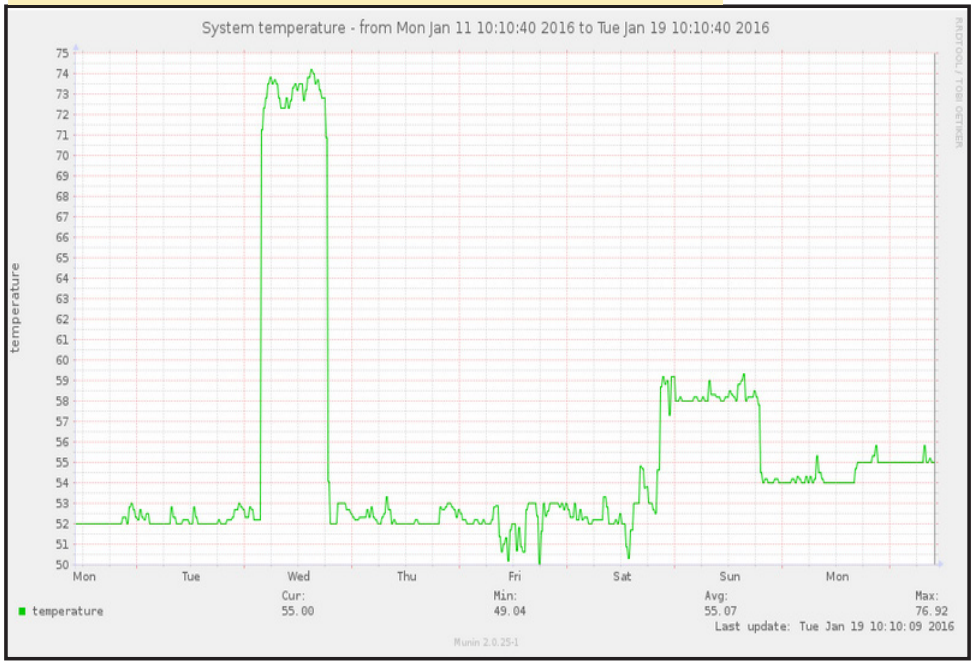
```
$ sudo munin-node-configure --shell
```

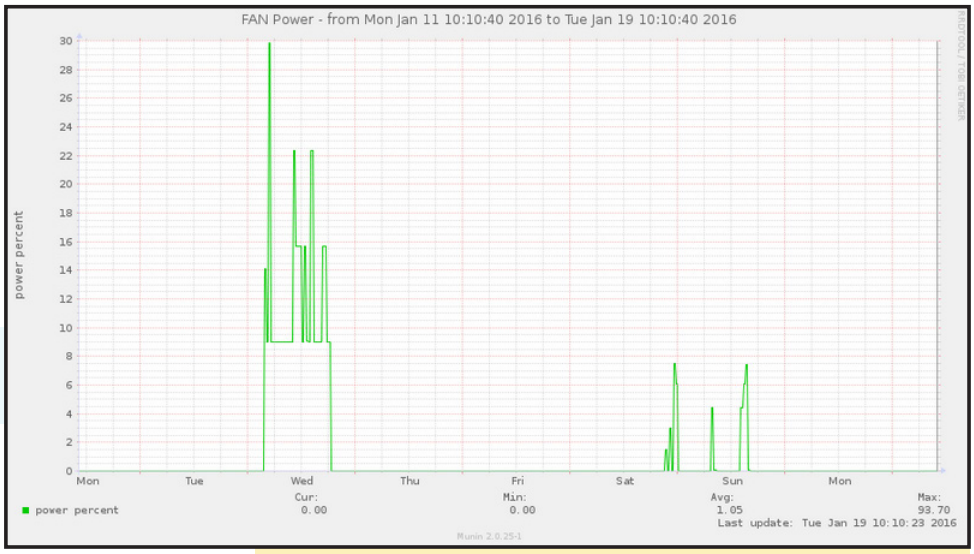
Este comando mostrará algunos comandos que necesitas para intentar activar un plugin específico. Debes copiar y pegar los comandos para crear enlaces simbólicos en /etc/munin/plugins que apuntan a los plugins reales en /usr/share/munin/plugins. Una vez que los enlaces simbólicos estén en su lugar, los nuevos plugins serán utilizados en el siguiente ciclo.

Puesto que estás usando Munin en un ODROID, es posible que quieras representar gráficamente datos específicos de éste, tales como los porcentajes de temperatura del sistema o la fuerza del ventilador. Puedes descargar estos plugins desde mi repositorio GitHub en http://it.ly/1PrTKY8. Para activarlos y fijar sus dependencias escribe los siguientes comandos:

```
$ sudo apt-get install bc
$ sudo wget http://bit.ly/1PI0Gkr
  -O /usr/share/munin/\
  plugins/odroid-temp
$ sudo wget http://bit.ly/1OJWYa6 \
  -O /usr/share/munin/\
```

Gráfica de la temperatura del sistema - Semanalmente





Porcentaje de potencia del ventilador del ODROID - Semanalmente

```
  plugins/odroid-fan
$ sudo chmod a+x \
  /usr/share/munin/\
  plugins/odroid*
$ sudo munin-node-configure --shell
```

El último comando debería sugerir qué plugins son los apropiados para tu sistema, de modo que podrás activar el plugin de temperatura para el C1 + y el XU3/4 y tanto la temperatura como el ventilador para el XU3/4:

```
$ sudo ln -s \
  '/usr/share/munin/plugins/odroid-temp' \
  '/etc/munin/plugins/odroid-temp'
$ sudo ln -s \
  '/usr/share/munin/plugins/odroid-fan' \
  '/etc/munin/plugins/odroid-fan'
```

Por favor, informa de cualquier problema que detectes con estos complementos a través del sistema de seguimiento de incidencias de Github o en el hilo de soporte Munin en los foros ODROID en http://bit.ly/1K0rRu9.

Solución de problemas

De vez en cuando puede que notes que las cosas no van bien, las gráficas no se actualizan o muestran datos incorrectos, o simplemente quieres entender por qué un determinado plugin muestra un valor en particular. Aquí tienes algunos pasos básicos para solucionar problemas:

1. Asegúrate de que funciona cron. El sondeo de datos y la generación de las gráficas dependen de cron. Puedes comprobar que Munin se ejecuta cada 5 minutos con este comando:

```
$ sudo tail -300 /var/log/syslog |\
  grep munin-cron
```

2. Comprueba cuando fueron actualizados por última vez los archivos RRD, qué tipo de datos almacenan y qué permisos tienen. Puedes hacer esto de forma manual o ejecutando Munin-check:

```
$ sudo munin-check
…
# /var/lib/munin/datafile : Wrong permissions (664 !=
644)
# /var/lib/munin/limits : Wrong permissions (664 !=
644)
# /var/lib/munin/munin-graph.stats : Wrong permis-
sions (664 != 644)
# /var/lib/munin/munin-update.stats : Wrong permis-
sions (664 != 644)
# /var/lib/munin-node/plugin-state : Wrong owner
(root != nobody)
# /var/lib/munin-node/plugin-state : Wrong permis-
sions (755 != 775)
# /etc/munin/plugin-conf.d : Wrong permissions (750
!= 755)
```

Comprueba el resultado. Ten en cuenta que este script revisa la mayoría de las cosas, pero no todas.

Tenga en cuenta que munin-check chequea los problemas pero no los corrige, de modo que tendrás que actualizar tú mismo los permisos siempre que se indique. Si estás analizando los ficheros RRD reales, comprueba la fecha de la última modificación y también echa un vistazo dentro para comprobar si se actualizaron por última vez y qué valores se escribieron:

```
$ cd /var/lib/munin/procyon
$ sudo ls -l
procyon-uptime-uptime-g.rrd
-rw-rw-r-- 1 munin munin 50664 Jan 16 17:05 procyon-
uptime-uptime-g.rrd
$ sudo rrdtool info \
 procyon-uptime-uptime-g.rrd | \
  egrep 'last_update|value' | head -2
last_update = 1452956716
ds[42].value = 5.6960000000e+01
```

3. Lee los registros log de /var/log/muni /. Tómate un tiempo para analizarlos y ver si puedes localizar el origen del problema.

4. Si va a solucionar los problemas de un plugin específico, puedes interactuar directamente con el proceso Munin-node y solicitarle que te devuelva información:

```
$ telnet 127.0.0.1 4949
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
# munin node at procyon

nodes
procyon
.

list procyon
cpu df df_inode if_eth0 load memory munin_stats
odroid-fan odroid-temp proc_pri processes uptime us-
ers vmstat

fetch odroid-temp
temperature.value 53
.

quit
Connection closed by foreign host.
```

Si los pasos anteriores no solucionan el problema, puede consultar las Preguntas Frecuentes de Munin para obtener más ayuda en `http://bit.ly/1P9tMwT`.

Conclusión

¿Por qué querrías usar Munin? Una vez que tengas tu sistema funcionando del modo deseado, puedes utilizar la información registrada por Munin para detectar lo que cambia con el tiempo. Por ejemplo, quizás tu cloudshell tenga un bajo rendimiento cuando se reproducen vídeos, en este caso podrías añadir gráficos para echar un ojo a tu red, al estado E/S o NFS para intentar identificar el problema. Además, puede crear tus propias gráficas en base a parámetros personalizados tales como el estado de los puertos GPIO. Ten en cuenta que las bases de datos RRD sólo almacenan una lectura cada 5 minutos, así que si los datos cambian con frecuencia sólo podrás capturar instantáneas del estado en esos intervalos, así que no es adecuado para representar gráficamente datos que cambian de estado varias veces por segundo. La API para añadir nuevas gráficas es fácil de seguir, y contando con datos históricos que puedes relacionar con otros parámetros es muy útil en los procesos de depuración.

EL KIT ROBOT OWEN

UN EJEMPLO DE PROYECTO PARA EL NUEVO ODROID-C0

por Bo Lechnowsky

La rebelión de las máquinas vendrá en forma de graciosos robot, has sido advertido. ¡Hazte con el tuyo ahora!

El proyecto OWEN comenzó hace un año cuando finalmente tuvimos un respiro por el alto volumen de ventas del ODROID C1 que empezaron a mediados de diciembre de 2014. Por aquel entonces, todo el personal de ameriDroid, junto con 6 trabajadores temporales, estuvieron de 12 a 14 horas diarias enviando pedidos del C1, 6 días a la semana, ¡durante más de un mes!

OWEN significa ODROID Walking Educational Unit, fue bautizado casualmente por uno de nuestros técnicos en prácticas por aquel entonces, también se llamaba Owen. La inspiración para el robot fue una plataforma "andante" con un smartphone controlada por Arduino llamado "MobBob". Queríamos ver si podíamos hacer lo mismo con un ODROID-C1, pero sin el requisito de tener que usar un costoso smartphone para echar andar el robot.

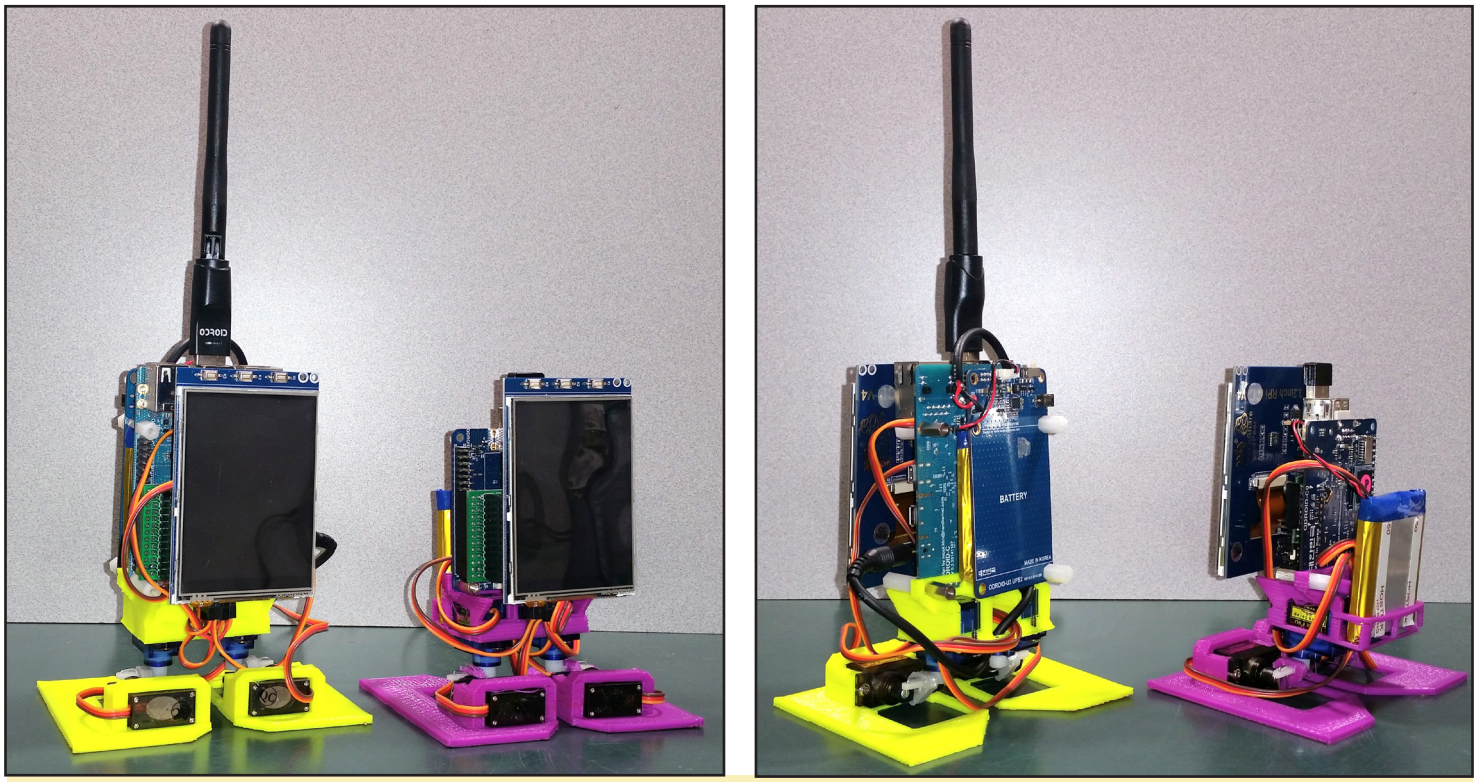
Diseñamos la pelvis, los pies y los tobillos utilizando el magnífico (y gratuito) software 123D Desing de Autodesk, e imprimimos las partes en nuestra impresora 3D Solidoodle con plástico ABS. Fueron necesarios varios intentos hasta perfilar todos los detalles e hacer pequeños ajustes en los diseños. Tras el lanzamiento del C1+ durante el verano del 2015, hicimos algunos cambios y pudimos utilizarlo en lugar del C1.

Puesto que queríamos usar la pantalla táctil de 3.2" del C1, esto descarta Android como sistema operativo. Por lo tanto, utilizamos la distribución Ubuntu estándar de Hardkernel. Para facilitar la instalación de la pantalla, Owen y yo escribimos el instalador ameriDroid automatizado para la pantalla táctil y lo liberamos en http://bit.ly/1NtDk09 para que otros también pudieran utilizarlo en sus proyectos.

Necesitábamos un servo controlador, por lo que buscamos alguno en la web. Conseguimos fijar el módulo Chroma Servo al C1, así que compramos bastantes de éstos y empezamos a ofrecerlos en el sitio web ameriDroid.com. Es un excelente servo controlador de 8 canales, pero presentaba el pequeño problema de que el pin 2x13 pass-through está desplazado ligeramente hacia un lado, haciendo que la pantalla se apagara en el lado izquierdo del robot a unos 6 mm. Esto afecta un poco al equilibrio de OWEN, pero en la mayoría de los casos no tiene consecuencias negativas gracias a los ajustes realizados en el software de control.

El siguiente obstáculo estaba relacionado con la alimentación por batería, pero en este caso no tuvimos ningún problema, Hardkernel ya había lanzado el kit UPS2-C1. Lo bueno del UPS2 es que cuenta con un interruptor de encendido/apagado, por lo que resulta fácil apagar el robot. Un inconveniente relacionado con el UPS2 es que los servos motores podían absorber mucha corriente momentáneamente, y éste está diseñado para proteger al C1 de este tipo de situaciones. Esto hacia que el robot no fuera capaz de arrancar a menos que estuviese conectado a un cargador de pared. La solución de retirar el chip IC de protección de energía nos la facilito rápidamente Justin Lee, y a partir de aquí estábamos listos para echar a andar nuestro robot. Las primeras pruebas nos permitieron ejecutar el robot con UPS2 durante más de una hora.

Queríamos poder controlar el robot incluyendo la programación, usando sólo un navegador web. Uno de los inconvenientes que tiene ejecutar un software de control desde un



El OWEN montado con 2 configuraciones distintas: conectado por cable y de forma inalámbrica.

servidor web son los permisos necesarios para mantener el software web activo pudiendo causar perjuicios al sistema operativo. Tuvimos que escribir un software intermedio para permitir que el software de control pudiera hacer cosas como lanzar y matar procesos. Había utilizado con anterioridad el servidor web Lighttpd en otros proyectos ARM, así que resulto muy fácil usarlo para este proyecto igualmente.

El software de control del robot está escrito en Rebol3, incluyendo los gráficos SVG del rostro - una característica integrada de Rebol/View. Rebol es un gran lenguaje, ya que no diferencia entre datos y código. Esto permite que un código que forma parte de un formulario web pueda ser ejecutado sin problemas. Rebol también cuenta con grades posibilidades para dialectos, conocidos como DSL (lenguaje de dominio específico). Así que escribimos un DSL que permitiera al usuario final programar con facilidad el robot a través de la interfaz web.

Como resultado, el robot puede ser controlado usando cualquier dispositivo que tenga un navegador web, como tablets, ordenadores portátiles, ordenadores de sobremesa, teléfonos inteligentes o cualquier ordenador de placa reducida. Si el puerto correcto está configurado en el router/firewall de la red, el robot puede ser incluso controlado desde cualquier parte del mundo.

Llevamos unos prototipos a la ARM TechCon en Silicon Valley donde compartimos un punto de venta con Hardkernel. Puesto que sabíamos que la red inalámbrica en el centro de convenciones se sobrecargaría, nos llevamos nuestro propio router VoCore OpenWRT para garantizar que los robots, un ordenador portátil, teléfonos inteligentes y algunos ODROID-VU7 pudieran comunicarse entre sí.

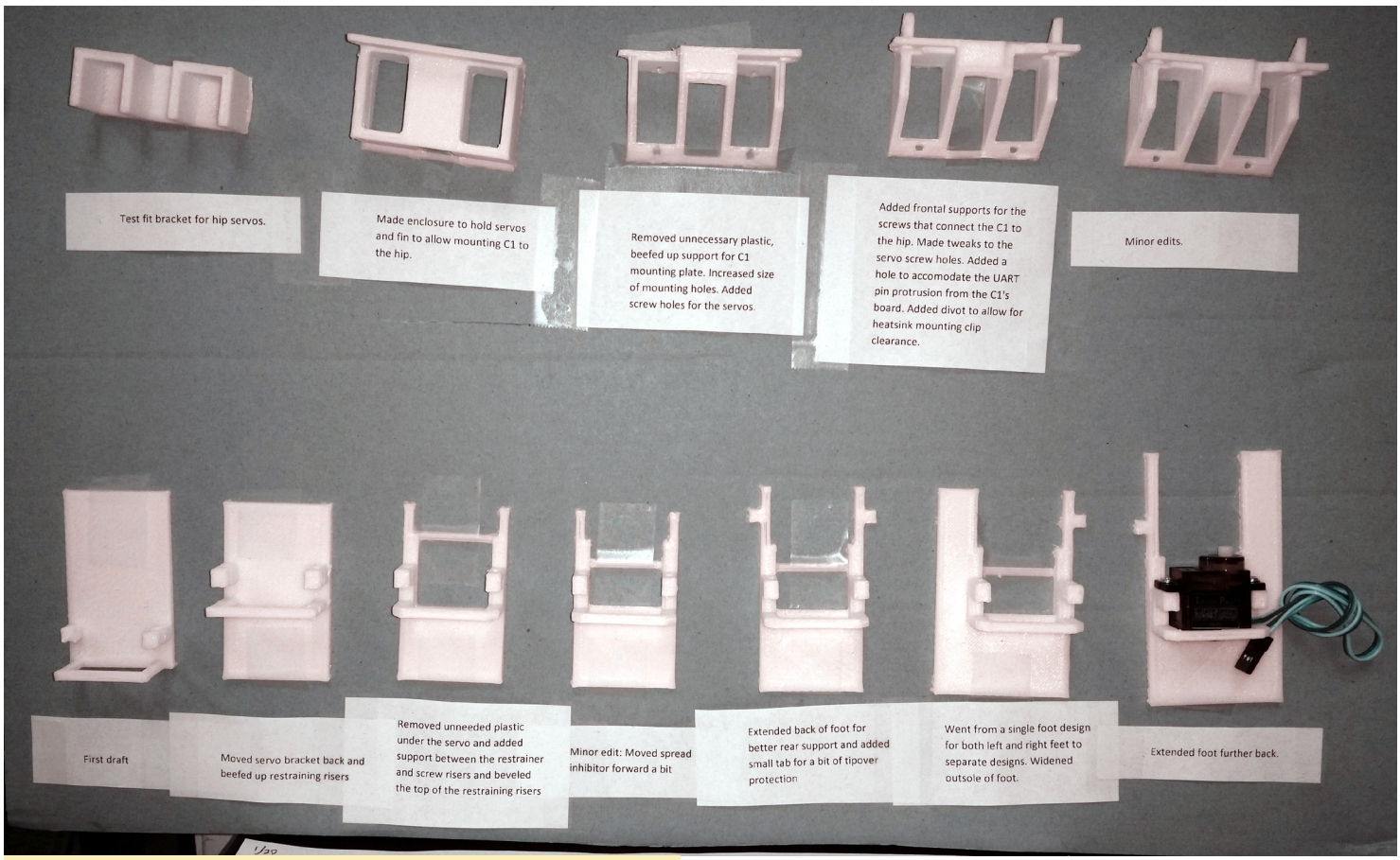
Empezamos a poner en venta el kits OWEN en ameriDroid.com poco después de la Techcon debido al gran número de peticiones por parte de usuarios que querían saber cómo conseguir uno. Si desea ver un vídeo del stand de Hardkernel en la ARM Techcon 2016, echa un vistazo a los siguientes enlaces:

ARMDevices.net http://bit.ly/1OGhQRw
Android Authority http://bit.ly/1PLSxLQ

Poco después de la Techcon, Hardkernel nos informó de que iban a publicar el ODROID-C0, y quisimos modificar OWEN para que funcionase con esta placa. ¡Fue fácil! Tan sólo necesitamos hacer algunos cambios menores en la pelvis para ayudar a OWEN a soportar un poco más de peso y complejidad. Puesto que el C0 tiene un circuito de carga de batería integrado y un interruptor de encendido/apagado, evitamos el hecho de utilizar el UPS2 sin contar con los puertos Ethernet y USB adicionales.

Ahora, estamos trabajando para añadirle unos altavoces a OWEN. Puede hablar con la ayuda de software Festival o Flite TTS para Ubuntu y puede reproducir MP3 con mpg321, así que bailar al ritmo de la música ya no es un problema, pero que disponga de una buena calidad de sonido sigue siendo un reto. También debería ser posible utilizar el módulo 2 Bluetooth USB para transmitir sonido desde OWEN a un altavoz Bluetooth, pero queríamos hacerlo completamente independiente si era posible.

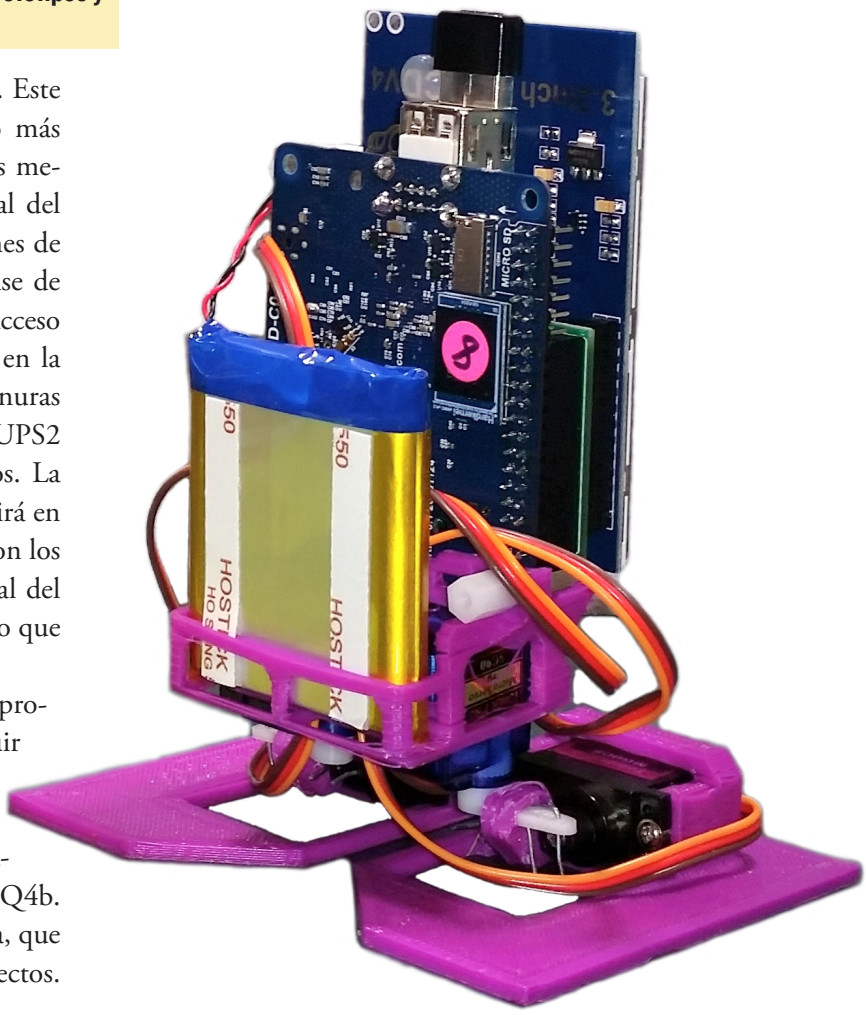
Además, estamos desarrollando un servo controlador interno más pequeño, más ligero y más barato usando la placa



¿El secreto de un buen diseño de robot? Crear muchos prototipos y refinar en cada serie.

ATmega328 Pro Mini con un diseño personalizado. Este debería acoplarse muy bien al tamaño y formato más compacto del C0. Para cuando terminemos con las mejoras inspiradas el C0, esperamos que el coste total del Owen baje al memos un 25%, y que las dimensiones de la parte frontal y trasera del cuerpo de OWEN pase de 60mm a 50mm, al mismo tiempo facilitaremos el acceso a la ranuras eMMC y microSD del C0. El UPS2 en la versión C1 del OWEN bloqueaba el acceso a las ranuras del eMMC y tarjetas microSD, de modo que el UPS2 tenía que ser retirado para acceder a estos módulos. La disponibilidad del módulo Wifi 0 también disminuirá en gran medida la altura de OWEN en comparación con los módulos Wifi 3 y 4 , y el centro de gravedad global del modelo OWEN-C0 será significativamente más bajo que en la versión del OWEN-C1+.

Como Hardkernel continúa haciendo nuevos productos con grandes características, esperamos seguir mejorando Owen y nuestros otros proyectos, como el kit de tablet VU7. Si quieres mantenerse al día de lo que estamos haciendo suscríbete a nuestro canal de YouTube ameriDroid en http://bit.ly/1OGhQ4b. Intentamos publicar al menos un video cada semana, que van desde interesantes tutoriales a videos sobre proyectos.



DESARROLLO ANDROID

ACCEDIENDO A LA PILA BLUETOOTH

por Nanik Tolaram

Android tiene una gran cantidad de sensores y posibilidades inalámbricas con las que desarrolladores de aplicaciones pueden trabajar, y una de la conexión inalámbrica estándar que ha existido durante mucho tiempo, aparte del Wi-Fi, es el Bluetooth. Casi todos los dispositivos Android soportan Bluetooth, y hay muchos productos Bluetooth que se pueden utilizar con dispositivos Android. Android proporciona una API fácil de usar para trabajar con Bluetooth. En este artículo vamos a echar un vistazo a la pila Bluetooth y una aplicación de ejemplo.

Principalmente, hay dos tipos de Bluetooth: el clásico Bluetooth y Bluetooth Low Energy. Estas clasificaciones también varían en la forma en la que la API trabaja en Android.

El clásico Bluetooth

El "viejo" Bluetooth v2.1/3.x. La mayoría de los dispositivos del mercado entran en esta categoría. Esta versión de Bluetooth está orientada a la transferencia de datos con un gran ancho de banda sin preocuparse por el consumo de energía, tales como altavoces, MIDI y auriculares.

El Bluetooth Low Energy

El Low Energy es el "nuevo" Bluetooth v4.x, que se centra principalmente en dispositivos que requieren transferencia de datos con un bajo ancho de banda y un consumo de energía de larga duración, como un faro.

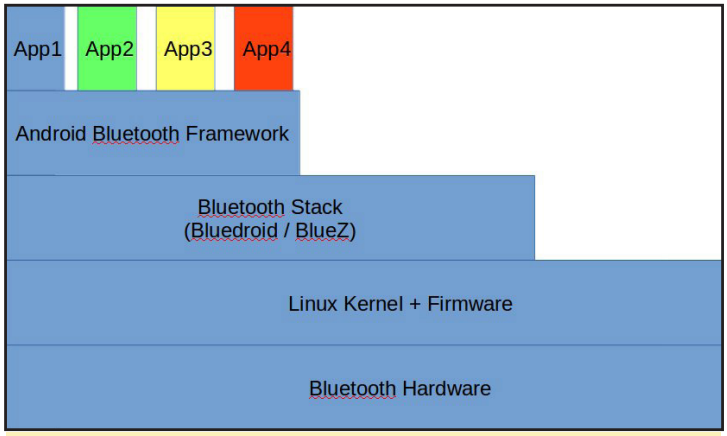


Figura I - Pila de Bluetooth

Pila Bluetooth

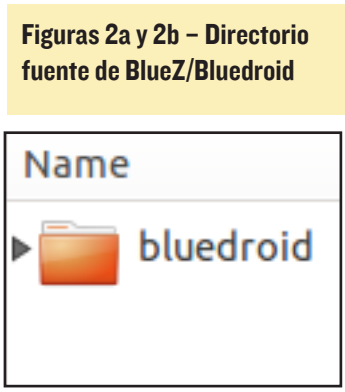
La Figura 1 muestra un diagrama de alto nivel de las diferentes capas de Android que trabajan en conjunto para crear la función de la aplicación Bluetooth.

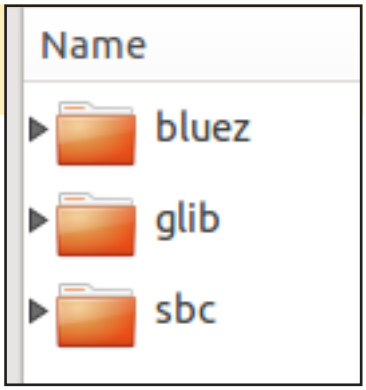
Android Bluetooth Framework – Esta capa es el "puente" que acorta las diferencias entre tu aplicación y todo lo que esté por debajo de ella. La mayoría de las veces, las aplicaciones de Bluetooth va a interactuar con esta capa a través de la API. Por ejemplo, el siguiente es el código más común que utilizarás para conseguir que un adaptador Bluetooth sea activado en tu aplicación.

```
BluetoothAdapter mBluetoothAdapter = BluetoothAdapt-
er.getDefaultAdapter();
```

Bluetooth Stack – Este es el "corazón" de toda la pila, ya que sin esta pila puedes suponer que no habrá Bluetooth en Android. Existen 2 proyectos diferentes en la comunidad de código abierto utilizados ampliamente - BlueZ y Bluedroid. BlueZ es el estándar de facto en el mundo Linux para cualquier tipo de distribución, mientras que el proyecto principal que se utiliza en Android es Bluedroid. BlueZ tiene amplio soporte para Android, y es plug-and-play cuando se utiliza en diferentes versiones de Android. Esta pila de software reside dentro del directorio external/Bluetooth del código fuente de Android. La Figura 3 detalla las diferentes estructuras de las fuentes BlueZ y Bluedroid. La única variante de Android de código abierto que utiliza BlueZ es el proyecto Android-x86.

Figuras 2a y 2b – Directorio fuente de BlueZ/Bluedroid





Linux Kernel + Firmware – Esta es la capa de software final que se encarga de la comunicación entre la pila de software superior y el hardware. La mayoría de los dispositivos Bluetooth tienen firmware propietario que se usa dentro del kernel Linux.

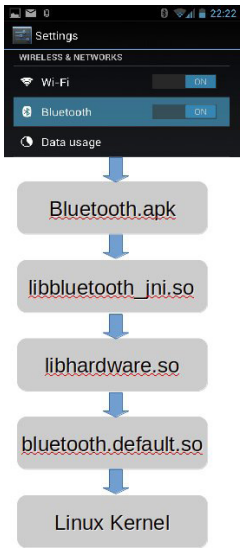
Archivos del sistema Bluetooth

Tanto la Android Bluetooth Framework como la Bluetooth Stack producen una serie de archivos binarios que carga Android durante el tiempo de ejecución. La Tabla 1 muestra algunos de los archivos que se generan y son utilizados por Android.

Binary Filename	Source Code Location
bluetooth.default.so	external/bluetooth/bluedroid
libbt-utils.so	external/bluetooth/bluedroid
libbt-vendor.so	hardware/<vendor>
libbt-hci.so	external/bluedroid/main
libbluetooth_jni.so	packages/app/Bluetooth/jni

Tabla I – Archivos del sistema Bluetooth.

Encendido



La Programación con Bluetooth es divertida, pero ¿alguna vez te has preguntado cómo funcionan las partes internas? ¿Cómo se "comunica" la app con el nivel inferior para que mi aplicación pueda enviar y recibir datos de forma inalámbrica? La Figura 3 muestra lo que ocurre internamente cuando se enciende el Bluetooth en el dispositivo. Bluetooth se activa manualmente a través de la aplicación Ajustes de tu dispositivo Android.

La única aplicación Java que se usa internamente se llama Bluetooth.apk, y esta en el código fuente de Android bajo la carpeta packages/apps/Bluetooth.

Figura 3 – Proceso de encendido del Bluetooth

Programación de Bluetooth

Como se ha mencionado anteriormente, existen dos formas de programar el Bluetooth en Android, dependiendo de los dispositivos con los que quieres que sea compatible. Las Tablas 2 y 3 muestra algunas de las clases que se utilizan normalmente para programar Bluetooth.

Class	Usage
BluetoothAdapter	This class represent the local Bluetooth device adapter in your Android device.
BluetoothDevice	This class represent remote Bluetooth device
BluetoothSocket	This class is used when connecting or connected to a Bluetooth device. This class is similar to TCP Socket class
BluetoothServerSocket	This class is similar to TCP ServerSocket class. Used normally as a listening socket
BluetoothHeadset	This class is the API allowing application to control the headset service and also handset profile
BluetoothClass	Generic class describing the characteristics and capabilities of a Bluetooth device

Tabla 2 - API del Clásico Bluetooth

Tabla 3 – API del Bluetooth Low Energy

Class	Usage
BluetoothAdapter.LeScanCallback	Callback interface that application normally will implement to receive results from scanning operation
BluetoothGatt	Public API for app to use the GATT functionality enabling communication to LE devices
BluetoothGattCallback	Callback interface for GATT events. For example connection change, services discovered, etc
BluetoothGattServer	This class provides Bluetooth GATT server role functionality, allowing applications to create Bluetooth Smart services and characteristics. This means you can make your Bluetooth adapter inside your Android device as a GATT server to broadcast beacon or some other functionality
BluetoothGattCharacteristic	This class represents a Bluetooth GATT Characteristic
BluetoothGattService	This class represents a Bluetooth GATT services

No vamos a entrar en profundidad en cómo se programa usando Bluetooth en Android, pero puedes visitar la web Android de Google, donde encontrarás extensa documentación junto con aplicaciones de ejemplo. Visita http://bit.ly/19NYRE3 sobre Bluetooth Clásico y http://bit.ly/ 1nj9p5Z para ojear tutoriales BLE.

Demo Bluetooth

Para ver una demostración de programación con Bluetooth en Android, descarga la aplicación de ejemplo Bluetooth Chat disponible en http://bit.ly/1SuBe7R, que es una aplicación básica de chat que permite la comunicación en forma de chat entre dos dispositivos Bluetooth usando programación clásica. La aplicación está dividida en 3 secciones principales:

Exploración/Busqueda – La aplicación escanea los dispositivos Bluetooth cercanos y los muestra en forma de lista incluyendo además, los dispositivos ya emparejados. Si el dispositivo no ha sido emparejado, el entorno de trabajo mostrará automáticamente un cuadro de diálogo que permite al usuario introducir el número PIN para emparejar el dispositivo. La forma en la que la interfaz de usuario recibe el resultado desde el entorno de trabajo para un dispositivos Bluetooth escaneados es escuchando a un objetivo BluetoothDevice.ACTION_FOUND, luego utiliza el método getBondedDevices () para obtener una lista de los dispositivos que se han vinculado. Al finalizar la operación de exploración, la aplicación detendrá el proceso de escaneo.

Conexión – Una vez que el usuario seleccione el dispositivo, la aplicación generará un nuevo subproceso para conectarse a éste. La manera de conectarse a un dispositivo clásico es usando el createRfcommSocketToServiceRecord (para co-nexiones seguras) o createInsecureRfcommSocketToServiceRecord (para conexiones no seguras). Una vez establecida la conexión, la aplicación generará otro subproceso para escuchar los paquetes entrantes desde el dispositivo externo. El subproceso generado permanecerá en un estado bloqueado mientras espera la llegada de un paquete entrante desde el dispositivo externo.

Transferencia de datos – Los paquetes y los mensajes entrantes son procesados por la aplicación con un proceso encargado de la gestión de mensajes en la interfaz de usuario.

CONOCIENDO UN ODROIDIAN

DAVID LIMA: ADMINISTRADOR DE SISTEMAS Y EXPERTO EN ALMACENAMIENTO EMPRESARIAL POR EXCELENCIA

editado por Rob Roy

Por favor, háblanos un poco sobre ti.

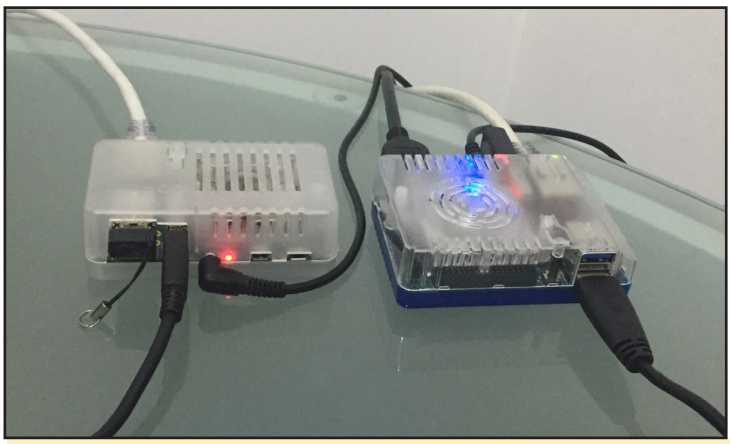
Mi nombre es David Lima y tengo 24 años, vivo en Sao Paulo, Brasil. Me gradué con un grado en Informática de desarrollo de software, y trabajo como especialista en sistemas de almacenamiento SAN Disk gestionando dispositivos de almacenamiento a nivel empresarial y la infraestructura que hay detrás de éstos. No estoy casado, pero tengo la intención de tener dos hijos y luego jubilarme en una pequeña ciudad tranquila.

¿Cómo fueron tus inicios con los ordenadores?

Recuerdo tener un ordenador en casa desde que tenía seis años, pero para cuando me di cuenta de lo que quería hacer ya tenía alrededor de 10 años, y el equipo era tan antiguo que dejaba de funcionar cada dos por tres. Sentía curiosidad por cómo funcionaba y lo que podía hacer para repararlo. ¡Con el tiempo fui montando y desmontando el equipo como si estuviera hecho de LEGO!

¿Qué te atrajo de la plataforma ODROID?

En realidad, soy nuevo en esto de los ODROIDs. Siempre he tenido curiosidad por las placas de desarrollo y simplemente quería una, pero he sido un poco vago a la hora de aprender más sobre ellas. Hace aproximadamente un año, un amigo mío me presentó sus placas ODROID y dije, "tío, esto es impresionante, ¿Por qué no he encontrado esto antes?" He probado muchas cosas con estas placas hasta ahora, pero todavía tengo mucho que aprender y experimentar con todo lo que esta gran plataforma puede ofrecernos.



Todos los artículos LVM han sido probados por la simple configuración U3 y XU4 de David, lo cual muestra lo lejos que se puede llegar con el almacenamiento en ODROIDs

David decidió engatusar a su amigo fotógrafo para conseguir todas sus fotos donde ¡NUNCA MIRA A LA CAMARA!

¿Cómo utilizas tus ODROIDs?

Mis ODROIDs están configurados como clientes torrent, servidores multimedia, servidores Samba, servidores DNS locales, estaciones de juego y utilizo uno para experimentar donde si algo sale mal, simplemente grabo la imagen de nuevo.

¿Cuál es tu ODROID favorito?

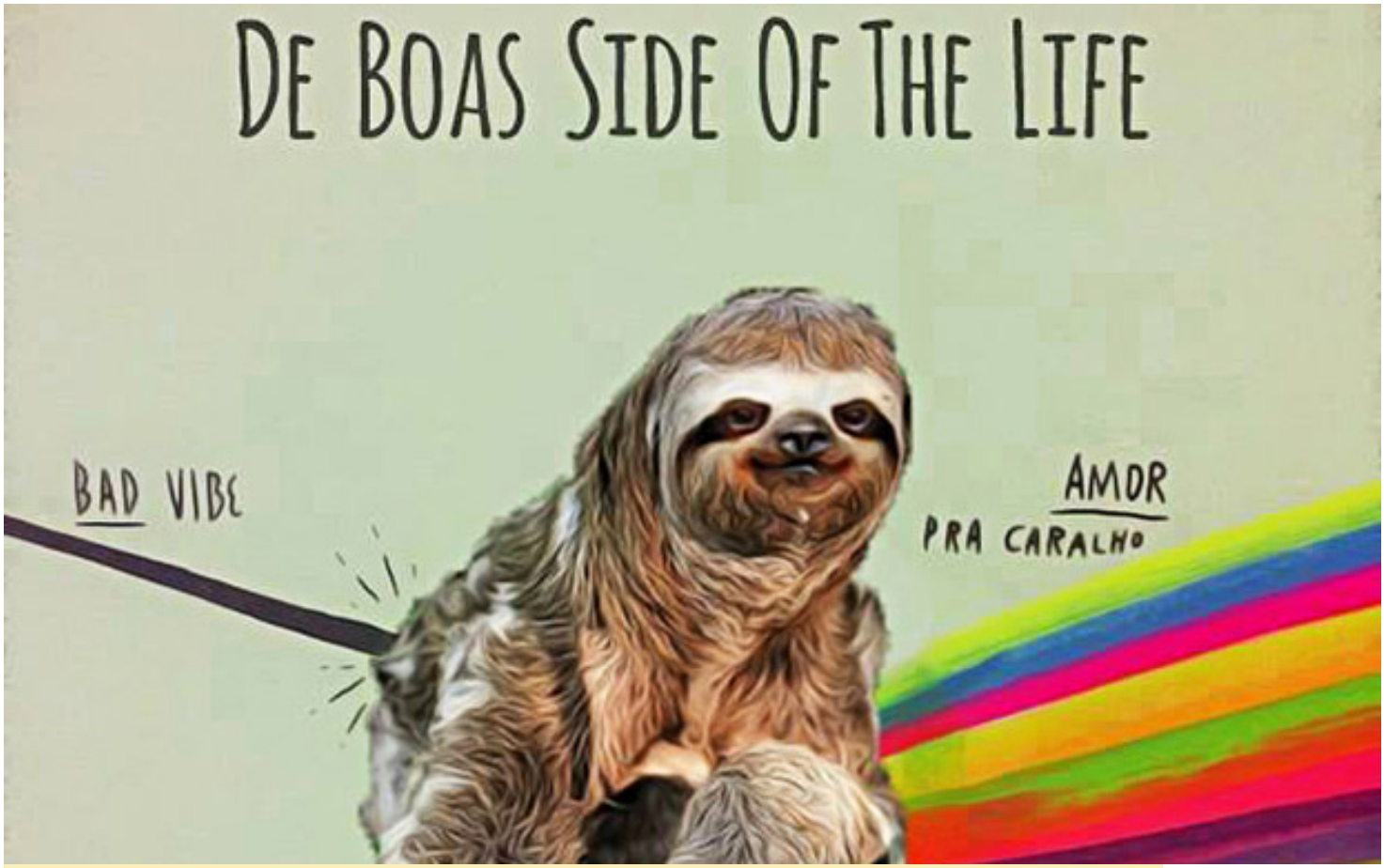
Todavía pienso que el U3 es el más estable y tiene el perfecto equilibrio entre precio y capacidad de procesamiento.

Tus artículos sobre Gestión de Volúmenes Lógicos son realmente buenos. ¿Cómo te has convertido en un experto en administración de servidores?

Pienso en mí mismo como un entusiasta más que un experto. Solía trabajar como administrador de sistemas Linux con soporte para grandes entornos, de aquí es donde procede la mayor parte de mi experiencia. También he tenido sistemas Linux en casa desde hace bastante tiempo, y siento mucha curiosidad a la hora de hablar de sus características.

¿Qué innovaciones te gustaría ver en futuros productos de Hardkernel?

Estoy deseando que los ODROIDs cuenten con más me-



Conocido por todos sus amigos como un maestro zen-life, nos envió esta imagen que representa su filosofía de cómo transformar malas vibraciones en energía positiva. No hay tarea que no se la tome con calma y con buen humor

moria RAM, así como una arquitectura CPU de 64 bits. Entonces, podré empezar a crear máquinas virtuales con calidad de producción y disponer de un centro de datos del tamaño de una tarjeta de crédito. Además, tener Bluetooth y Wi-Fi integrados en las placas ahorraría el tener que usar algunos puertos USB.

¿Qué aficiones e intereses tienes aparte de los ordenadores?

Soy una especie de friki total, así que aparte de los ordenadores, me gusta jugar a un juego de cartas llamado Magic The Gathering. También me gusta ir al cine y leer cómics de superhéroes. Mis deportes favoritos son el voleibol y la natación, pero hace bastante tiempo que no los practico.

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

El secreto para aprender cualquier cosa, que no es ningún secreto en absoluto, es divertirse. Haz lo que te gusta y ni siquiera te darás cuenta de que está aprendiendo. Tener un objetivo también ayuda bastante. Si identificas un problema, vas a aprender mucho resolviéndolo. Cuando lo hagas, a menos que quiera ser Daniel-san de la película Karate Kid y quiera dar cera/pulir cera un y otra vez, escribe como lo hiciste, ya que casi seguro lo olvidarás pasado un tiempo. Esto te ahorrará mucho un tiempo en el futuro.

¿Quieres saber el lado oscuro de nuestro maestro Zen-life? Juega al Magic contra él y descubrirás un jugador extremadamente agresivo, ¡No digas luego que no te lo advertimos!