Half-Life en ODROID • Kernel en Tiempo Real • Memoria de sistema con zRAM

ODROID Magazine

Año Tres
Num. #27
Mar 2016

Una nueva generación de ARM:

Procesamiento a *64-Bit*

El innovador ODROID-C2

- Escritorio EXAGEAR para los ODROIDs
- Fundamentos MQTT: Aprende a activar la Mensajería IoT entre dispositivos

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

EDITORIAL

El tan esperado ODROID-C2 de 64-bit por fin está aquí. Con video 4K, 2 GB de RAM, Ethernet Gigabit y un potente procesador S905 Amlogic, ofrece una potencia pionera a un precio muy asequible. Está disponible por 40$ en la tienda de Hardkernel en http://bit.ly/1fbE9Id. Para conocer mejor el C2, y para descargar los sistemas operativos pre-compilados como Android y Ubuntu, visita la nueva página wiki ODROID-C2, en http://bit.ly/1Trq5Ef. Uno de los nuevos periféricos más emocionantes que existen para el ODROID-C2 es la oCAM, una avanzada cámara USB 3.0 que se puede utilizar en proyectos OpenCV. DoYoon Kim describe cómo configurar fácilmente un sencillo sistema de seguimiento y vigilancia usando software de código abierto.

El famoso ODROID-C0 se puede utilizar junto con MQTT para crear dispositivos IoT, como muestra Venkat en su último artículo sobre pequeñas modificaciones de hardware. Tobias continúa su serie de juegos con Half Life, Andrew nos muestra cómo utilizar un kernel en tiempo real para optimizar la eficiencia de las aplicaciones, Christopher presenta un proyecto sobre cómo controlar luz de fondo de la pantalla del ODROID-VU7, Adrian nos ayuda a mejorar nuestro terminal de Linux usando Byobu , y Justin analiza a fondo Exagear para que puedas ejecutar programas de Windows en tu ODROID.

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/1ypImXs.
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

OUR AMAZING ODROIDIAN STAFF:

Rob Roy, Editor Jefe

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

Bruno Doiche, Editor Artístico Senior

Bruno fue a Las Vegas para casarse y disfrutar de sus merecidas vacaciones. Disfrutó bastante, y cuando regresó a nuestra querida revista se pudo muy contento al ver que la gente de Hardkernel había lanzado el ODROID-C2, cosechando muy buenas críticas y finalmente, dándonos la oportunidad de probar este ¡Cohete dispuesto a despegar!

También se puso muy contento cuando vio la lista de invitados para su boda contando con Billy Corgan, Depeche Mode, la modelo de Victoria Secret Candice Swanepoel y muchos otros personajes famosos. Pero al final tan solo se trataba de una broma bien elaborada a manos de su cuñada que lo engaño a él y a su novia (ahora esposa). Al final, resulto ser una fantástica boda en la que disfrutaron muchísimo.

Manuel Adamuz, Editor Español

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

Nicole Scott, Editor Artístico

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

James LeFevour, Editor Artístico

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

Andrew Ruggeri, Editor Adjunto

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

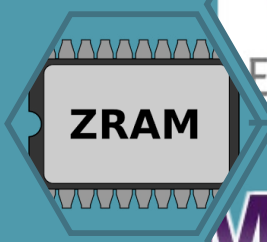
Venkat Bommakanti, Editor Adjunto

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

Josh Sherman, Editor Adjunto

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

INDICE



LUZ DE FONDO ODROID-VU7

AÑADIENDO UN SISTEMA DE CONTROL DIGITAL

por Christopher Dean

Alguna vez has querido controlar la luz de fondo del ODROID-VU7 utilizando la GPIO digital en el ODROID-C1. En este artículo, voy a describir cómo añadir un sistema digital para controlar la luz de fondo utilizando únicamente un transistor y dos resistencias. El interruptor digital puede complementar al interruptor físico ya existente o prescindir de él por completo.

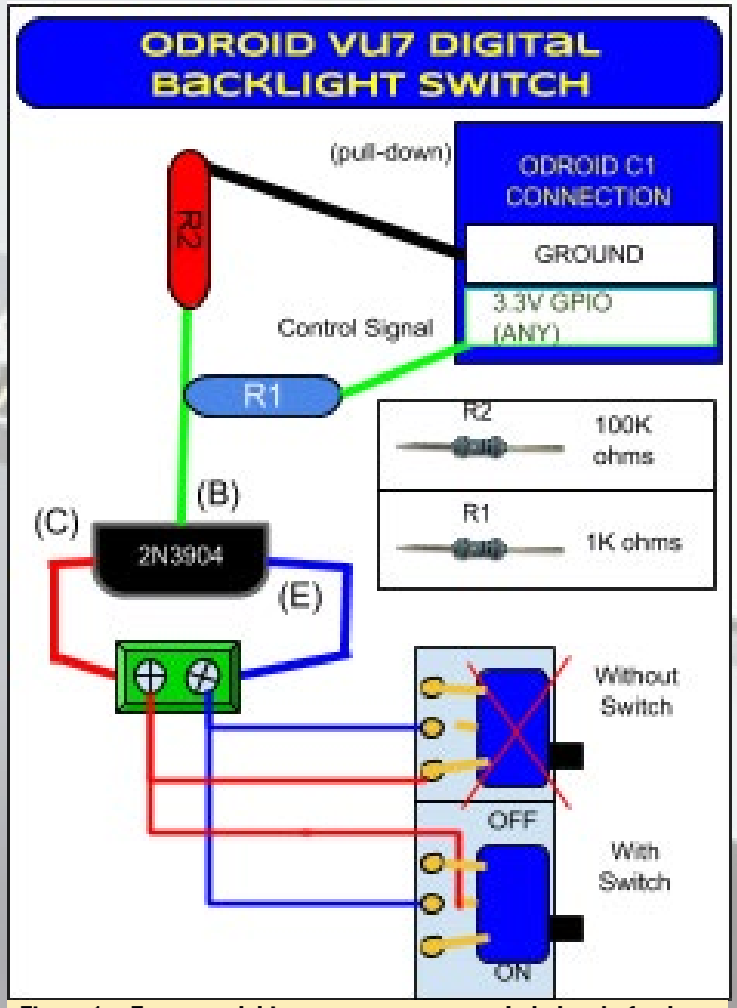


Figura 1 – Esquema del interruptor que controla la luz de fondo en el ODROID-VU7

Ten en cuenta que ésta es una modificación semipermanente que requiere realizar algunas soldaduras. Por favor, ten en cuenta que ni Hardkernel ni yo no nos hacemos responsables de los posibles daños derivados de esta práctica. Además, la garantía Hardkernel sobre el ODROID-VU queda completamente anulada y sin efecto, puesto que estamos modificando el hardware. La Figura 1 muestra una representación esquemática del diagrama del circuito finalizado.

Localiza el interruptor de luz de fondo "On/Off" en la parte trasera de la pantalla en la esquina superior izquierda junto a los 3 puertos. Utiliza unas tijeras para cortar el pin central del

Figura 2 - Pin central del interruptor físico del hardware que debe cortarse,

interruptor. El corte debe realizarse en algún punto intermedio. Realiza una desconexión completa doblando los extremos recién cortados. La figura 2 muestra el pin central que debe cortarse.

Llegados a este punto, tienes que decidir si deseas omitir el interruptor del hardware o seguir utilizarlo. Si decides mantenerlo, la pantalla se apagará cuando éste se encuentre en la posición física "Off", independientemente del estado del pin GPIO. Si por el contrario prescindes del interruptor, el control digital será la única forma de encender y apagar la pantalla. Asegúrate de seguir los procedimientos de seguridad estándar

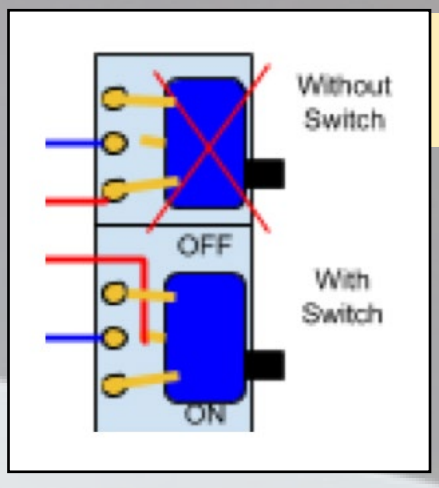


Figura 3 - Primer plano de la modificación del interruptor

para ti y el dispositivo durante la fase de soldadura.

En cualquier caso, suelda un cable en el lado del emisor a la mitad inferior del pin central de la PCB (cable azul). Para conservar el interruptor ya existente, suelda el cable rojo (lado del colector) a la mitad superior del pin central. Por otro lado, para

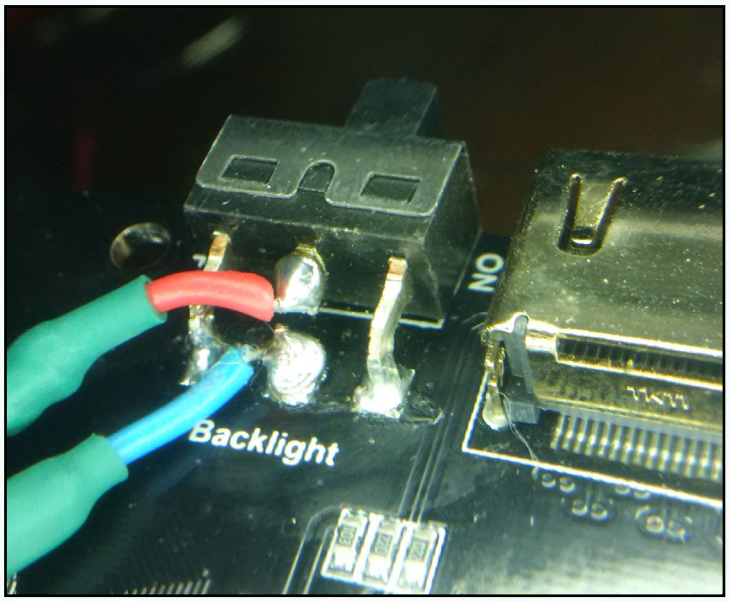


Figura 4 - Soldadura realizada para conservar el interruptor

evitar usar el interruptor, suelda el cable rojo (lado del colector) al pin que va a la posición "On". Analiza bien la Figura 3 antes de continuar.

La Figura 4 muestra el estado del dispositivo en el caso de decidir conservar el interruptor.

Usando las ilustraciones y los componentes indicados, suelda a la vez el transistor (2N3904) y las resistencias (1K y 100K ohmios - ½ vatios) en una placa de pruebas. Puedes utilizar un terminal de tornillo, tal y como se indica en la Figura 1 o soldar directamente los cables desde interruptor del hardware al transistor. A continuación, añade un cabezal hembra o macho para los pines de señal y desplegables, según corresponda en tu caso. Dedica bastante tiempo a este paso para asegurar la precisión del circuito y así evitar posibles cortocircuitos eléctricos. Se recomienda envolver el cable y testear el circuito antes de soldarlo. Además, ten en cuenta que es probable que el transistor se caliente durante su uso, así que asegúrate de colocar un disipador de calor o cualquier otra opción de refrigeración pasiva para evitar que se queme. La Figura 5 muestra el resultado

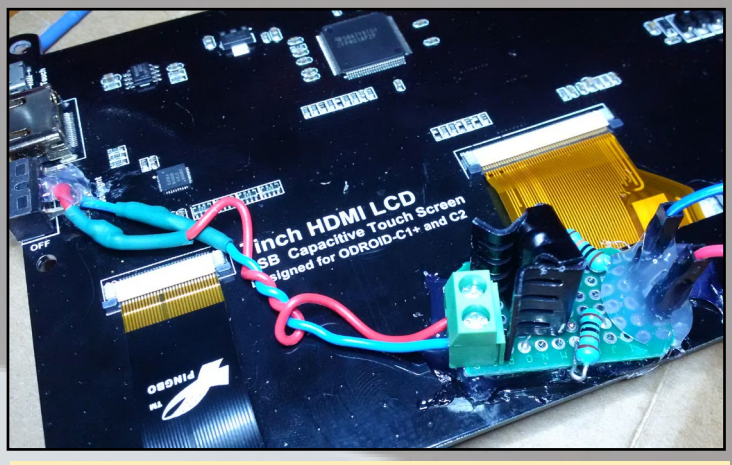


Figura 5 - Añadiendo refrigeración pasiva

de los pasos llevados a cabo hasta el momento.

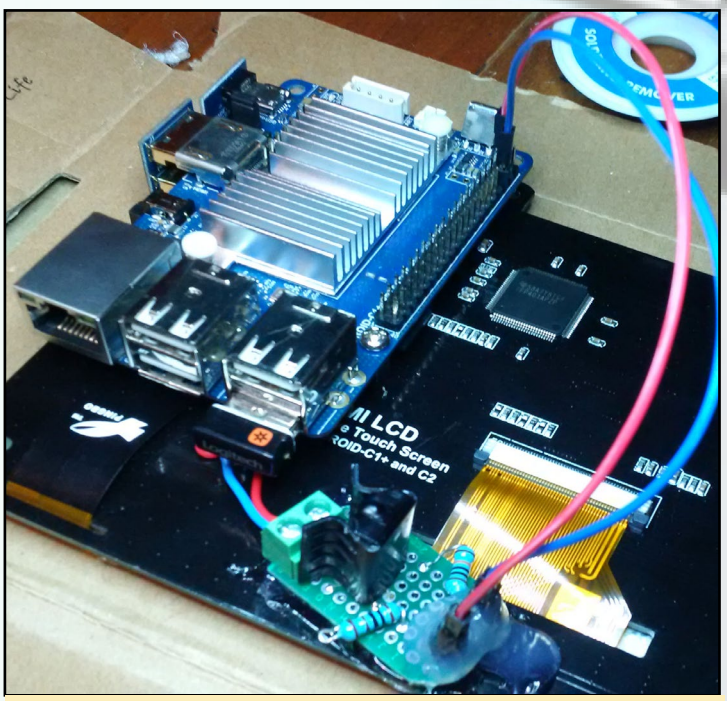


Figura 6 – Montaje de la placa transistor en el panel LCD

Opcionalmente y como último paso, monta la placa de transistores en la parte posterior del panel LCD y conecta los cables de conexión al C1. Para realizar pruebas, en lugar de conmutar el transistor con el GPIO, también puedes usar el pin 1 (alimentación de 3,3 V) para forzar que la luz de fondo se encienda. Elije un pin GPIO y prueba usando ese pin a apagar y encender la luz de fondo. Yo utilice el pin 7 (GPIO # 83) cuando realice mis pruebas.

SUPER OJOS

SUPERVISION Y SEGUIMIENTO DE LAS MANOS CON LA OCAM

por DoYoon Kim

Ya hay disponible un módulo cámara muy económico y al mismo tiempo potente. Cuando buscas una cámara para un ODROID, puedes interesarte por una cámara de visión periférica. Sin embargo, casi seguro que el precio de una cámara de este tipo es mucho mayor que el precio de la propia placa de desarrollo. Naturalmente, la pregunta que se hace mucha gente es: ¿Por qué no usar una barata webcam?

Claro que sí, pero una webcam normal tiene posibilidades muy limitadas y carece de modularidad. En otras palabras, una simple webcam poco flexible y poco configurable. Antes de nada, una cuestión clave de la que suelen carecer las baratas webcams es la posibilidad de cambiar la lente de la cámara para adaptarse a tus necesidades particulares. Esto es una gran desventaja, puesto que la lente determina el campo de visión y la longitud focal. Debido a que las webcams están hechas para video chat, tienen una distancia focal corta y campo de visión muy pequeño. Esto se debe a que la cámara sólo tiene que centrarse en el rostro durante el video chat, que normalmente se encuentra cerca de la cámara. En muchos proyectos, tienes que tomar una imagen de un objeto que está lejos de la cámara, o puede que tengas que ampliar el campo de visión mucho más allá del alcance de una simple webcam. Además, para una aplicación más profesional, es posible que necesites una cámara con un obturador, rara vez encontraras una webcam que soporte esta funcionalidad.

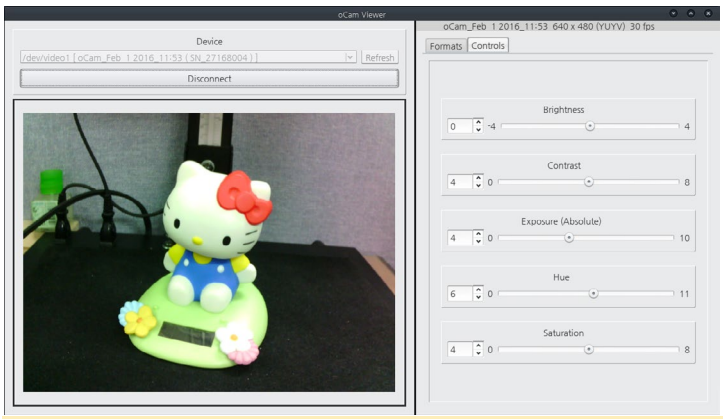
La mayoría de las cámaras están hechas para ser utilizadas con una mínima dificultad y configuración. Esto supone un problema cuando necesitamos controlar determinados parámetros de la cámara, tales como la exposición automática. Para procesar las imágenes correctamente, es necesario mantener constante el tiempo de exposición. Sin embargo, las webcams generalmente incorporan modos automáticos, tales como la auto exposición que impide la manipulación manual y limita bastante la capacidad de controlar la cámara.

Otro problema de la mayoría de webcams es la interfaz entre el módulo de la cámara y la placa. Se podría pensar que para la mayoría de webcams, es suficiente una interfaz USB 2.0 sin que haya necesidad de gastar más dinero en una cámara con una interfaz más rápida. Por desgracia, nos encontramos ante dos limitaciones importantes a causa de esta lenta interfaz de datos. Una limitación obvia la encontramos en el tamaño de la imagen y la velocidad con la que se pueden transmitir los fotogramas. Otro problema con un impacto oculto que incluso es mayor está relacionado con el uso de la CPU en la placa. Al utilizar un canal de comunicación tan lento como el USB 2.0, es inevitable que pueda haber compresión y descompresión de vídeo. Con un tamaño de fotograma y una velocidad de 1280 x 720 a 30 fps, se consume una gran cantidad de CPU.

¡La buena noticia es que por fin tenemos una alternativa! Un módulo cámara para ODROID, llamado "oCam", disponible por 99$ en `http://it.ly/1WsoNbr` y cuenta con las siguientes características:

Soporte para lentes M12: Soporte para lentes M12: El usuario puede elegir una de las lentes M12 que cuenta con 5 distancias focales diferentes para utilizar en la oCam. Pronto habrá disponibles piezas opcionales para adaptar lentes C o CS.

Control total: La oCam viene con un software con el que puedes controlar diversos parámetros de la cámara tales como la resolución y el tiempo de exposición. El código fuente del software y el ejecutable binario para ODROID están disponibles para su descarga.

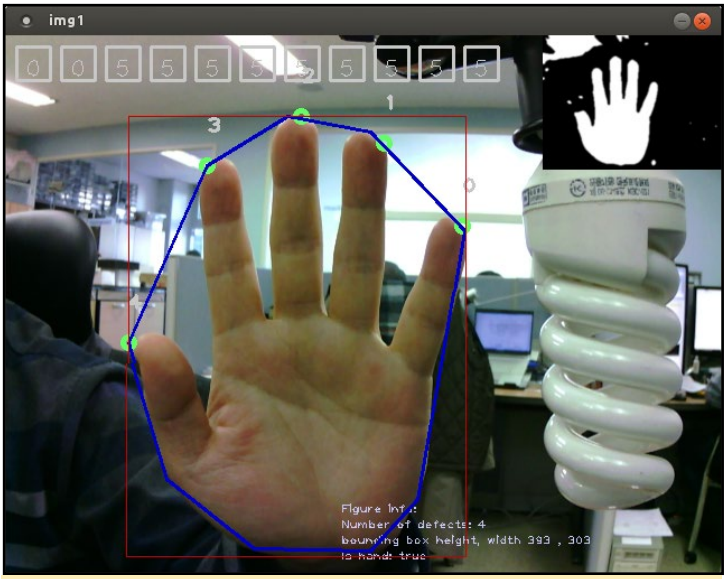


Visor oCam que permite controlar los parámetros de la cámara

Interfaz USB 3.0: La oCam cuenta con una interfaz USB 3.0 de alta velocidad que utiliza DMA para reducir drásticamente el uso de la CPU.

Está disponible a un precio sorprendentemente bajo.

La oCam está totalmente testeada y es compatible con UVC (USB Video Class), que no requiere ningún driver para el dispositivo. Esto permite que la oCam pueda funcionar tanto en Windows como en sistemas Linux. Entre las muchas aplicaciones posibles de la oCam, los siguientes dos ejemplos son los que mejor muestran de lo que es capaz la oCam.



Sistema de detección handDetectionCV de Simen Andresen

Detección de manos

Esta aplicación ha sido desarrollada por Simen Andersen, y utiliza OpenCV para reconocer y rastrear el movimiento de la mano de un usuario. En el post del blog de Simen Andresen " Hand Tracking and Recognition With OpenCV " en `http://it.ly/1mOrrMu` encontraras más información.

Puedes iniciar la detección de manos en un ODROID-XU4 siguiendo estos pasos:

Configura el ODROID-XU4 para acceder a Internet.

Abre un terminal, introduce el siguiente comando y luego, sigue las instrucciones para instalar la librería OpenCV:

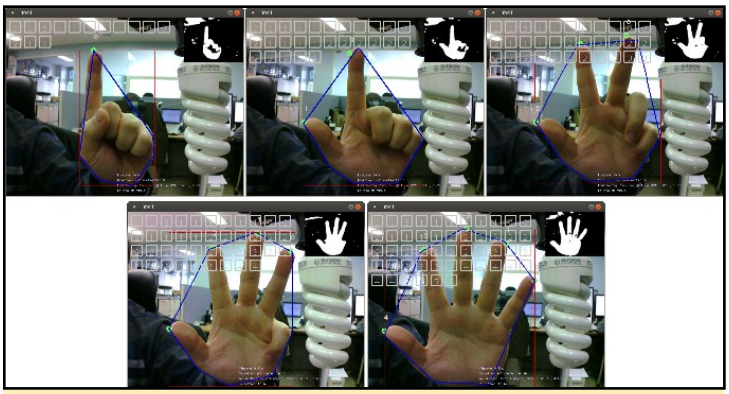
```
$ sudo apt-get install libopencv-dev
```

Consigue el código fuente del sistema de detección, muévete a la carpeta "Linux Version" luego, compila y ejecuta el programa:

```
$ make all
$ opencv
```

Intenta abarcar todos los cuadrados verdes con una mano. Tu mano será detectada cuando los cuadrados verdes se vuelvan blancos como se muestra en las siguientes imágenes. Ahora empieza a mover tu mano y verás cómo el sistema la sigue.

Detección de manos por handDetectionCV

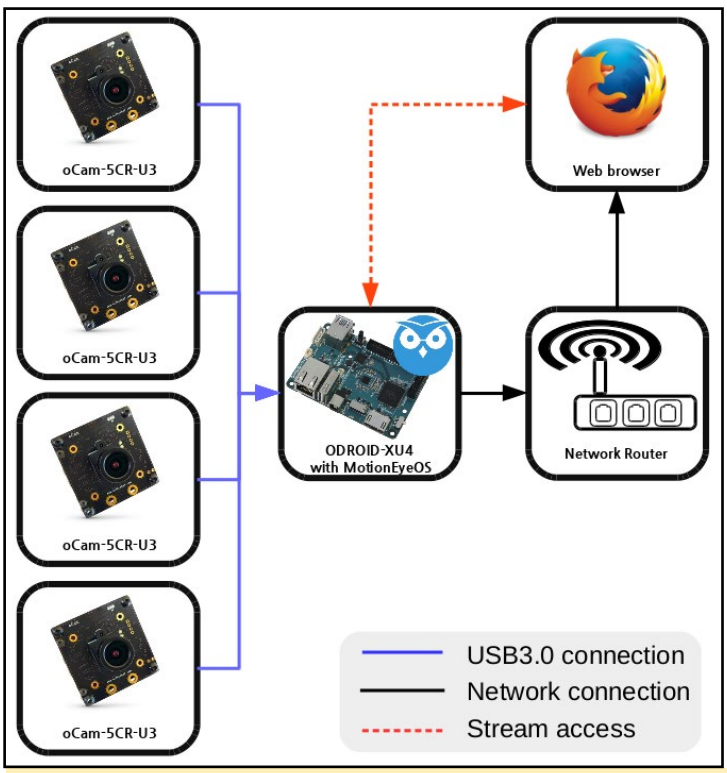


Reconocimiento y seguimiento de la mano por handDetectionCV

Sistema de video vigilancia

Puedes crear tu propio sistema de video vigilancia utilizando un simple ODROID-XU4 y múltiples cámaras oCam. El sistema de vigilancia se basa en MotionEyeOS, desarrollado y publicado por Calin Crisan.

GitHub de MotionEyeOS: `bit.ly/1OlPlVE`
Facebook de MotionEyeOS: `on.fb.me/1ounjlP`



Configuración del sistema de video vigilancia

Para poner en marcha un sistema de vigilancia sigue los siguientes pasos:

• Prepara un módulo eMMC o tarjeta micro SD para ODROID-XU4 y las cámaras. En nuestro ejemplo, nosotros utilizamos 4 cámaras oCam.

• Configura el ODROID-XU4 para acceder a Internet.

• Descarga la última versión de MotionEyeOS para ODROID-XU4 desde la página "Supported Devices" de GitHub (`http://it.ly/214XmuP`) y descomprime el archivo descargado.

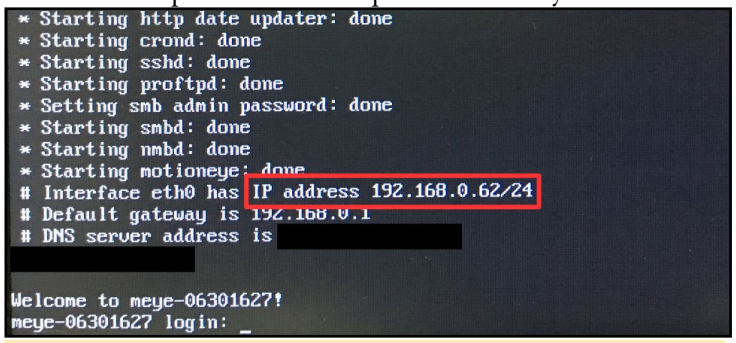
• Conecta el módulo eMMC o tarjeta micro SD a tu ordenador y graba el archivo de imagen descomprimido en éste. Si no está familiarizado con este proceso, puedes echa un vistazo a la siguiente guía de ODROID `http://it.ly/1Vk9u4o`.

• Insertar o fija el módulo eMMC o tarjeta microSD cargada con MotionEyeOS en tu ODROID-XU4.

• Usando el puerto Ethernet, conecta el ODROID-XU4 a una red y enciende el ODROID.

El sistema tardará en arrancar aproximadamente 1 o 2 minutos. Después, podrás iniciar sesión utilizando la cuenta por defecto "admin" sin contraseña.

Puedes podrás acceder al sistema de vigilancia desde cualquier navegador web utilizando la dirección de red que se muestra en la pantalla de arranque de MotionEyeOS.

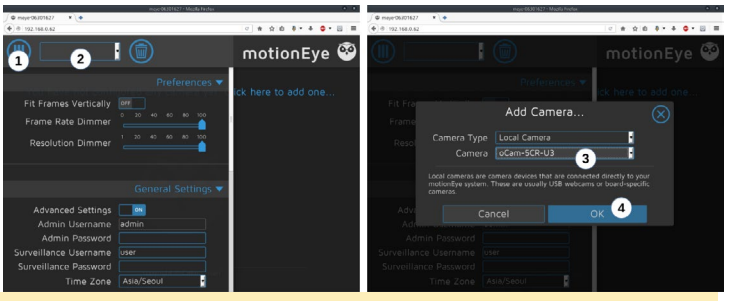


Pantalla de inicio de sesión de MotionEyeOS

Conecta una OCAM al ODROID-XU4.

Desde un navegador web, inicia sesión en MotionEyeOS utilizando la cuenta de administrador. Haga clic en "Add Camera" para añadir la oCam.

Ajusta la resolución a 640 x 480 y la tasa de fotogramas a 30fps, luego finaliza el proceso haciendo clic en "Apply".



Añadiendo oCam al sistema MotionEyeOS

Comprueba que se muestra vídeo procedente de la oCam. Si no ves ninguna imagen procedente de la oCam, intenta alguna de las siguientes soluciones:

Desconecta y vuelve a conectar la oCam. En la consola del ODROID XU4 comprueba la conexión a la cámara usando dmesg.

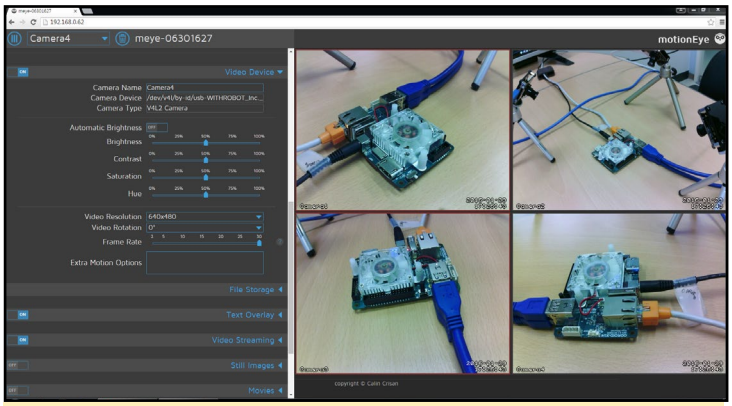
Añade más potencia a la oCam utilizando una fuente de alimentación externa o un hub USB 3,0 autoalimentado.

Conecta la oCam a otro sistema Linux para comprobar si la oCam puede transmitir correctamente imágenes a una resolución de 640 x 480 a 30fps.

Repite el proceso de conexión para el resto de cámaras oCam que no se encuentren conectadas.

Tras completar estos pasos, ¡Dispondrás de tu propio sistema de vigilancia! La pantalla de monitorización MotionEyeOS mostrará algo similar a las siguientes imagenes:

Sistema de video vigilancia MotionEyeOS



Pantalla de video vigilancia del MotionEyeOS

Puesto que a MotionEyeOS se puede acceder a través de cualquier navegador web, puedes visualizar las imágenes de la cámara y controlarlas desde cualquier lugar siempre y cuando el ODROID-XU4 está conectado a Internet. Dispones de más información detallada en el GitHub de MotionEyeOS en `http://bit.ly/1VuPBHS`.

SHOW ME STUFF

UN SISTEMA QUIOSCO INTELIGENTE PARA TU ODROID-SHOW

por @matoking

Hace algún tiempo, desarrolle un script basado en Python llamado SHOWtime (http://bit.ly/1VfzMmW) para poder mostrar alguna información interesante sobre un ODROID-SHOW conectado a un ODROID compatible. La información incluye estadísticas del sistema, precio del Bitcoin y seguimiento del tiempo de actividad de páginas web.

Tras cambiar a un ODROID-XU4, he creado una aplicación web usando Flask y jQuery que tiene esencialmente las mismas características pero además, es capaz de interactuar con el usuario. Puedes ver un vídeo de la aplicación en funcionamiento en http://youtu.be/kVVemfqK-nI.

La aplicación, que incluye el código fuente e instrucciones de instalación y uso, se puede obtener en http://bit.ly/1TuJx2L. Actualmente la aplicación muestra la siguiente información a través de una interfaz a modo de presentación de diapositivas:

- **Uso de memoria RAM y CPU,**
- **Uso del disco,**
- **Cola, que muestra las últimas líneas de un archivo de texto elegido, como por ejemplo el registro log del kernel,**
- **Precio de Bitcoin, acompañado de un elegante gráfico,**
- **Rastreador Bitcoin, que muestra el saldo actual y las transacciones más recientes,**
- **Seguimiento del tiempo de actividad de una Página Web**

Ten en cuenta que la aplicación tiene que ser configurada para utilizar los parámetros VIEW_WIDTH y VIEW_HEIGHT, basándose en las resoluciones soportadas de tu pantalla táctil. La predeterminada debería ser la correcta para ODROID-VU y para las pantallas táctiles HDMI Waveshare de 5” y 7”.

Tras lanzar la aplicación web, se puede acceder a la interfaz de usuario usando un navegador web con el enlace: http://<ODROID-device-ip-address>:5210. Si se accede al dispositivo en local, puede utilizar el enlace: http://localhost:5210 o http://127.0.0.1:5210. Puedes configurar las vistas actualizando la configuración en settings.py, que es el archivo de código fuente en python de la aplicación web.

Si deseas añadir más vistas, puede hacerlo creando los siguientes tres tipos de archivos:

Vista Flask (showmestuff/views/<nombre-de-tu-vista>.py): Contendrá las peticiones HTTP requeridas por la aplicación JavaScript para recuperar la información necesaria, como por ejemplo el uso de CPU.

Vista HTML (showmestuff/templates/views/<nombre-de-tu-vista>.html): Contendrá el código HTML que recoge la configuración específica de la vista y el código fuente JavaScript asociado a la aplicación.

Fichero JS (showmestuff/static/js/views/<nombre-de-tu-vista>.js): Contiene la propia aplicación que recupera la información utilizando las peticiones HTTP proporcionadas por la vista Flask y la muestra en pantalla cuando se activa la vista.

Showmestuff da un toque sofisticado a tu ODROID-SHOW

Por ejemplo, si deseas añadir una vista para visualizar el uso de la CPU, la expresión <nombre-de-tu-vista> indicada anteriormente podría corresponder al uso-CPU. Puedes utilizar la vista de tiempo de actividad como un simple ejemplo para desarrollar y activar tu propia vista. Para comentarios, preguntas y sugerencias, puedes visitar el post original en http://bit.ly/1PGHNB8.

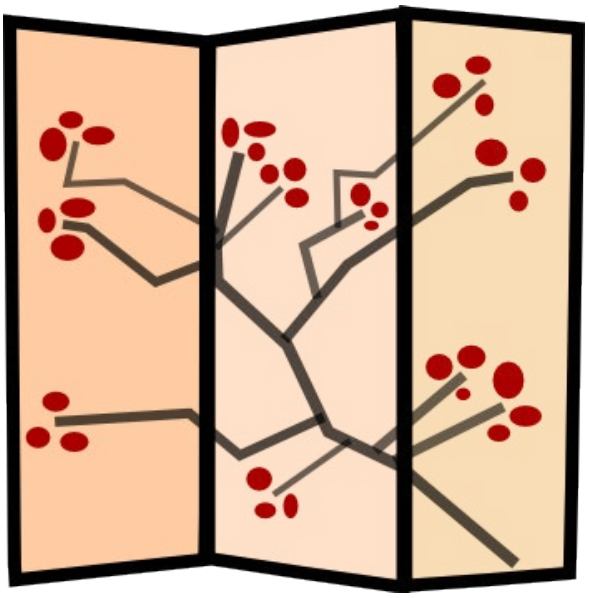
Referencias:

http://bit.ly/1VfzMmW
http://flask.pocoo.org/
https://jquery.com/

BYOBU

TU TERMINAL DE LINUX SOBRE ESTEROIDES

por Adrian Popa



Te pasas mucho tiempo delante de un terminal. La mayoría de tus ventanas consisten en terminales que ejecutan programas de línea de comando medio olvidados. Sufrés la molestia de que se desconecte tu SSH ya sea por tiempo de espera o por fallo en la red, y tienes que volver a conectarte e iniciar sesión de nuevo. Pues bien, creo tener una solución a tus problemas, se llama byobu (http://byobu.co/). Por cierto, si alguna vez olvidas el nombre, puedes localizarlo por mamparas plegables japonesas: (http://bit.ly/1Rm4Mlq)

Byobu es un gestor de ventanas basado en texto y un terminal multiplexor. Está diseñado para mejorar herramientas como GNU Screen o TMUX. Incluye perfiles mejorados, cómodos atajos de teclado y notificaciones de estado del sistema. Me atrajo por la hecho de poder ver en todo momento la temperatura del sistema ODROID mientras lo administraba vía SSH y así poder evitar el sobrecalentamiento.

Para empezar a trabajar con byobu rápida y fácilmente, lo mejor que puedes hacer es ver un vídeo de introducción. (http://bit.ly/1QetGEI) Te mostrará lo que puedes hacer y cómo lo puedes hacer. Este artículo describe algunos de los accesos directos y personalizaciones que puedes usar con la utilidad.

Comencemos

Así que vamos a empezar. Para instalar byobu en Ubuntu:

```
$ sudo apt-get install byobu
```

Para iniciarlo, sólo tiene que ejecutar byobu en tu terminal:

```
$ byobu
```

Y llegarás a tu nueva ventana de terminal. A primera vista parece un entorno shell corriente, pero observarás una barra de estado que muestra varios parámetros del sistema y un prompt ligeramente modificado, como puedes ver en la Figura 1.

La barra de estado muestra la siguiente información por defecto: Sistema operativo y versión, lista de terminales activos, paquetes no actualizados, tiempo de actividad, carga, nivel de CPU, uso de memoria y hora y fecha actuales. Veremos en un minuto cómo podemos cambiar estos parámetros, pero en primer lugar vamos a aprender cómo movernos.

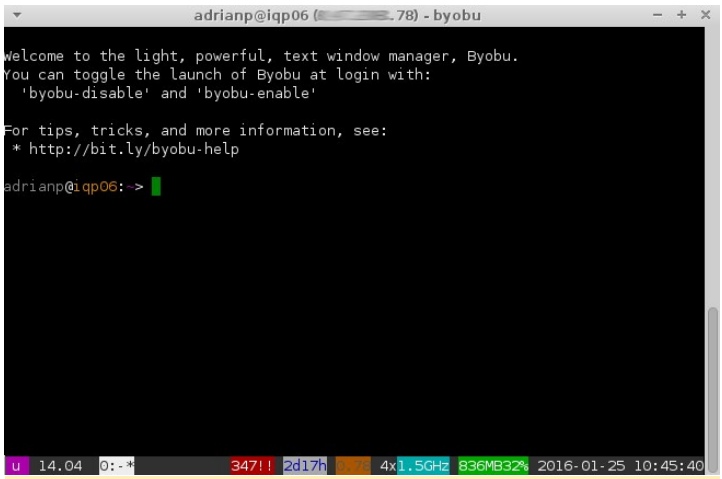


Figura 1 - Pantalla principal de Byobu

Consejos y trucos

Si no estás familiarizado con Screen o tmux, aquí tienes un rápido repaso. Imagina que tienes acceso a un único terminal en modo texto, un terminal muy básico, pero que deseas tener la experiencia multitarea a la que estás acostumbrado en un entorno de escritorio. Con Screen o tmux puedes iniciar tantos terminales como necesites (con CTRL + A C) y cambiar entre ellos (CTRL + A seguido del número de terminal), al mismo tiempo que tienes la ventaja de mantener los terminales activos, incluso si te desconectas de ellos (simplemente sustituye CTRL +A por CTRL + B para tmux). Más adelante, puedes conectarte a cualquier sesión tmux o screen que esté activa y reanudar tu trabajo. Si estás interesado en ellos, dispones de más información en http://bit.ly/1QCPGuU.

Byobu toma como punto de partida las principales caracte-

Figura 2 - Hoja de información de Byobu

rísticas de screen/tmux simplificando su uso en general. Para acceder a la hoja con los atajos de teclado de byobu, simplemente tienes que pulsar Shift + F1 mientras estés ejecutando byobu.

Te sugiero que dediques un tiempo a jugar y practicar creando y cambiando ventanas, además de practicar con ventanas divididas horizontal y verticalmente para entender mejor dónde y cómo utilizar las funciones disponibles. No es necesario tener siempre ventanas divididas, pero a veces puede serte útil para

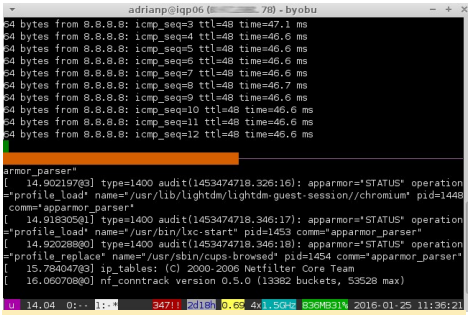


Figura 3 - Multitarea en la misma ventana dividida

echar un ojo a un fichero log mientras editas un archivo o ejecutar un script.

Puesto que byobu hace mucho uso de las teclas de funciónes, puede entrar en conflicto con otras aplicaciones como Midnight Commander. Para activar y desactivar los atajos de teclado de byobu, puedes pulsar Shift + F12. Mientras que los atajos de teclado estén deshabilitadas puedes utilizar los atajos de screen o

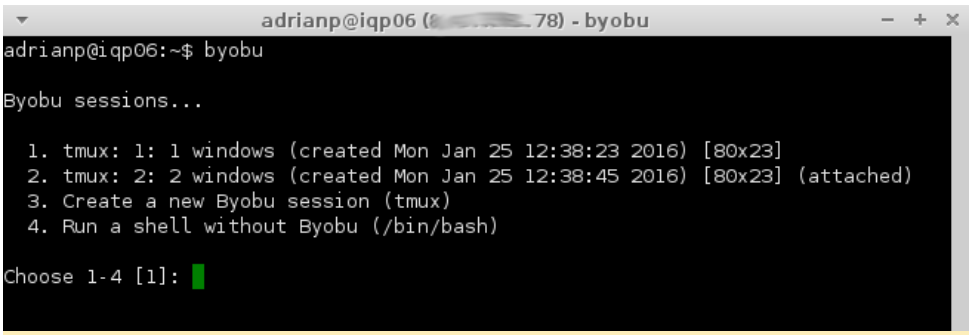


Figura 4 - Seleccionador de sesiones en caso de contar con múltiples sesiones

Figura 5 – Alternando entre las distintas vistas de estado

tmux para crear o navegar entre las ventanas. Al pulsar CTRL + A la primera vez, byobu te pregunta si desea activar los atajos de teclado en "pantalla" o si deseas el editor Emacs por defecto en su lugar. Puede cambiar esto más adelante ejecutando:

```
$ byobu-ctrl-a
```

Ahora que ya has vicheado un poco byobu, tal vez te haya gustado tanto como para tenerlo como shell por defecto al iniciar sesión. Puedes hacerlo activando el menú (con F9) y seleccionando la última opción, "Byobu currently does not launch at logon (toggle on)". Ahora, la próxima vez que te conectes, tus scripts de arranque Bash iniciarán byobu y podrás volver a conectarte automáticamente a tu sesión. Si dispone de varias sesiones, se te preguntará en la que deseas conectarte, como aparece en la Figura 4. Puedes conectar desde varios lugares a la misma sesión y la pantalla cambiará de tamaño para ajustarse a la ventana más pequeña. Aunque esta configuración no hará que todas las nuevas ventanas de terminal bajo X11 inicien byobu, Alt + F1 – Alf + F6 iniciará una nueva sesión byobu al conectarte, lo cual podría serte muy útil.

Personalizar la barra de estado

La barra de estado proporciona información útil sobre el estado y el rendimiento de tu sistema. Por defecto hay algunas "vistas" predefinidas que se pueden alternar con Shift + F5, como puede verse en la figura 5.

Para personalizar lo que se muestra en la barra de estado, dirígete al menú (F9) y selecciona, " Toggle status notifications" (Figura 6). Si activas "custom", serás capaz de añadir tus propios scripts que te pueden facilitar información útil. Puede leer la página de guía para hacerte

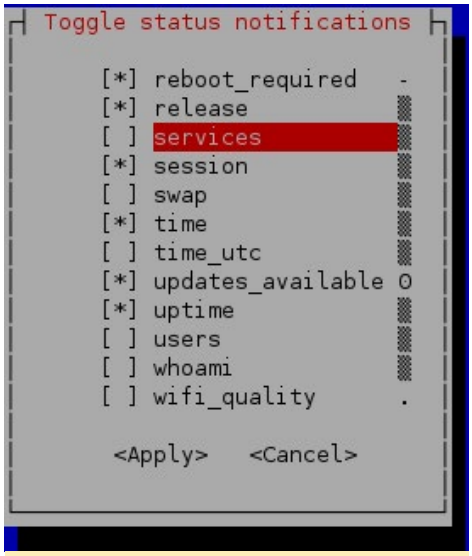


Figura 6 - Seleccionando los plugins deseados.

una idea de lo que tienes que hacer en caso de que quieras desarrollar alguno.

Para los usuarios ODROID he creado unos pequeños scripts a medida que muestran la temperatura del sistema y el porcentaje de potencia del ventilador de nuestros dispositivos. Puede descargarlos desde github (http://bit.ly/1KJh2gH) e instalarlos con estos comandos:

```
$ mkdir -p . byobu/bin
$ sudo apt-get install bc
$ cd . /byobu/bin
$ wget https://raw. githubuser-
content. \ com/mad-ady/odroid-
byobu/master/20_fan
$ wget https://raw. githubuser-
content. \ com/mad-ady/odroid-
byobu/master/20_temperature
$ chmod a+x 10_*
```

El plugin del ventilador debería funcionar en el ODROID-XU3 y ODROID-XU4, mientras que el plugin de temperatura debe trabajar en todos ODROID. El número anterior al nombre del plugin hace referencia a la frecuencia con la que se ejecuta, que en este ejemplo es cada 20 segundos. Tienes total libertad para editar los plugins y configurar los diferentes umbrales para que cambien los colores y así adaptarlos mejor a tus necesidades. Puedes observar un ejemplo en la Figura 7.

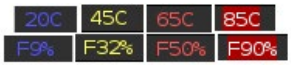


Figura 7: Plugins del ventilador y de la temperatura personalizados

Soluciones y problemas comunes

Estos son algunos de los problemas con lo que me encontré y algunos consejos para solventarlos:

- Al iniciar byobu, con cada actualización de estado (cada segundo) se añade una nueva línea en la parte inferior, apilando varias actualizaciones.

Esto ocurre en los terminales que no soportan UTF-8 (las líneas de estado de byobu utiliza algunos caracteres codificados en UTF-8, que desordenan la longitud de la fila si no son compatibles con el terminal). Para solucionar esto, si te conectas a través de putty intenta esta solución (http://bit.ly/1mWHQ1r) o si te conectar por ssh desde un sistema Linux diferente, intenta ejecutar byobu con la localización "C":

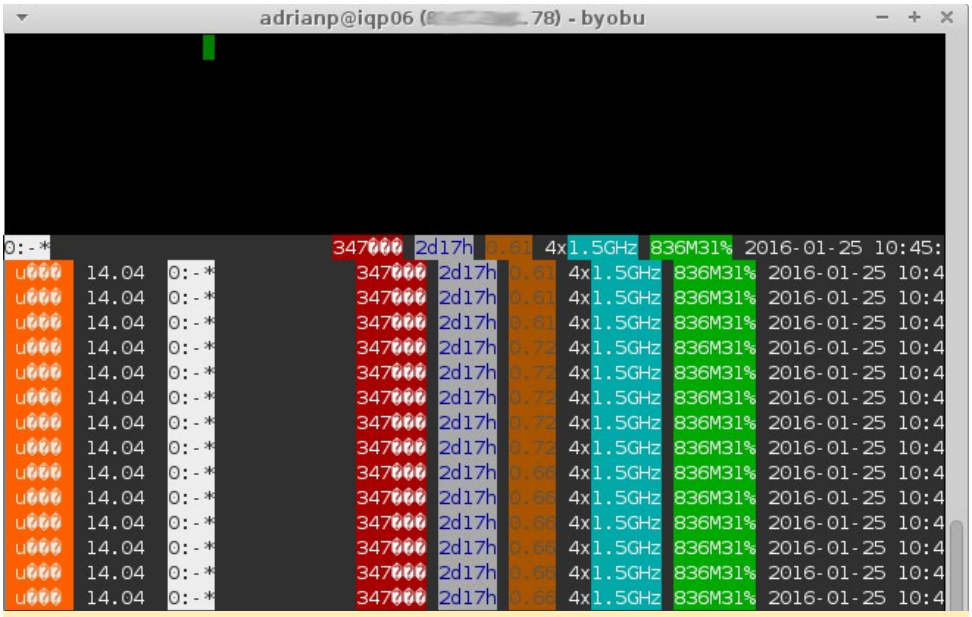


Figura 8. Byobu con soporte UTF-8 irregular

```
$ LC_ALL=C byobu
```

- Observas que el desplazamiento hacia atrás con la rueda del ratón no funciona adecuadamente.

Por defecto tmux se inicia con el desplazamiento del ratón desactivado. Puedes configurarlo ejecutando este comando y reiniciando byobu:

```
$ echo 'set-window-option -g
mode-mouse on' >> . byobu/. tmux.
conf
```

Opcionalmente, puedes utilizar el teclado introduciendo el "copy mode" con CTRL + A [y usando las flechas o RePág/AvPág para desplazarte a través del buffer.

- ¡Socorro! Metí la pata en algo y byobu falla o se cuelga en el arranque y no me puedo conectar.

Si algo estropeo byobu o si deseas conectarte a un intérprete de comandos bash estándar por ejemplo, puedes pasar por alto los scripts de inicio automático:

```
$ ssh -t your-odroid-ip bash
```

- ¿Qué sucede cuando estás dentro byobu e intentas SSH hacia un sistema byobu?

¿Has visto la película Origen? Por defecto byobu detecta que estás trabajando dentro de byobu (por la magia de las variables de entorno) y no te permitirá conectarte a una sesión byobu remota por otro lado. Sólo conseguirás un intérprete remoto plano dentro de tu byobu local. Pero si quieres, puedes arrancar el byobu remoto y experimentar lo mejor de origen - pero no vayas demasiado lejos, puede dolerte la cabeza. Afortunadamente puedes ejecutar screen dentro byobu, siempre y cuando se utilicen diferentes atajos de teclado.

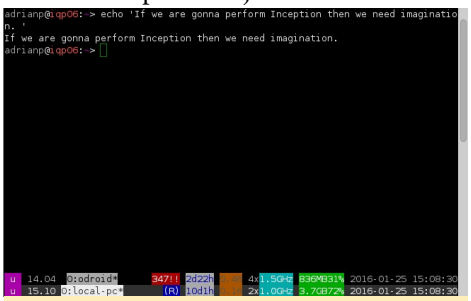


Figura 9 - La parte inferior de la ventana byobu se ejecuta en mi PC mientras que la ventana interior se ejecuta en el ODROID

¿Ver? Con ayuda y algo de esfuerzo, ¡El mundo de los terminales no es un lugar tan espeluznante después de todo!

HALF-LIFE

BLACK MESA HA LLEGADO A LA PLATAFORMA ODROID

por Tobias Schaaf

Bienvenido a Black Mesa! He estado esperando mucho tiempo hasta conseguir que este impresionante juego pueda ejecutarse en la plataforma ODROID. Ahora es posible gracias a @ptitSeb, quien ha exportado Half-Life a ARM utilizando el motor Xash3D y otras librerías. Utilizando su versión de GLshim, ¡Podemos jugar al Half-Life a 1080p en nuestros ODROIDs!

Este juego es único en su género, e hizo que los videojuegos de disparos en primera persona se popularizaran muchísimo. Al igual que la versión de Doom de muchos años atrás, este juego tuvo una gran impacto en la industria. Hay un sinfín de mods para Half-Life, como el Counter Strike y Team Fortress. Estos juegos fueron originalmente creados sólo como complementos para el Half-Life, pero más tarde se convirtieron en juegos independientes.

Figura 1 – Centro de Investigaciones de Black Mesa

Figura 2 - Uno de los robots de la Intro del Half-Life

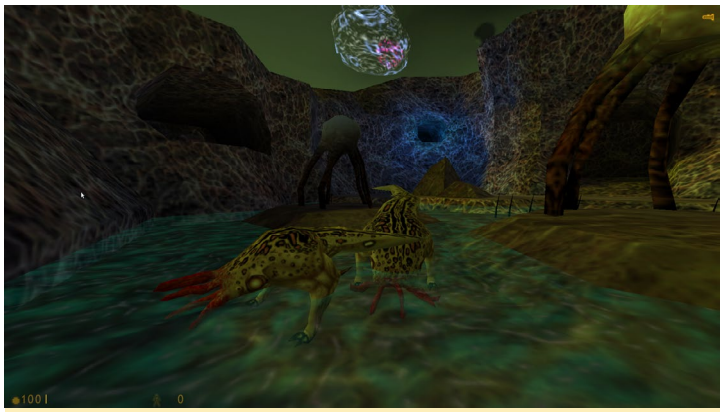


Figura 3 - El planeta alienígena con los monstruos que invaden el Centro de Investigaciones de Black Mesa

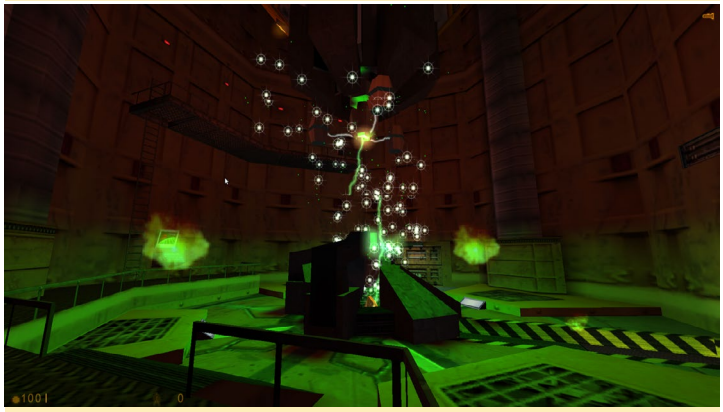


Figura 4 - Aunque es de hace 18 años, el juego cuenta con muy buenos efectos visuales

Con el siguiente comando, puedes instalar el motor Xash3d desde mi repositorio utilizando la lista de paquetes de Debian Jessie:

```
$ apt-get install xash3d-odroid
```

El juego ofrece soporte para los modos individual y multijugador para el Half-Life original, así como el complemento de Blue Shift. También podrían funcionar otros mods, como Counter Strike.

Para jugar, necesitarás la versión 1.1.1.0 del Half-Life original. En el directorio de inicio de Linux, encontrarás una carpeta llamada ".xash3d". Coloca la carpeta "valve" del Half-Life dentro de ésta. También tendrá que colocar en esa carpeta los mods del Half-Life. Echa un vistazo la siguiente lista de directorios a modo de ejemplo:

```
$ ll /home/odroid/.xash3d/
total 12
drwxr-xr-x  9 odroid odroid 1024
Feb  4 21:18 bshift
drwxr-xr-x 12 odroid odroid 1024
Feb  4 21:38 dmc
drwxr-xr-x 14 odroid odroid 3072
Feb  4 21:42 gearbox
drwxr-xr-x 12 odroid odroid 1024
Feb  4 21:42 ricochet
drwxr-xr-x 15 odroid odroid 3072
Feb  4 21:45 tfc
drwxr-xr-x 17 odroid odroid 3072
Feb  4 23:19 valve
```

Para comentarios, preguntas y sugerencias, por favor visita el artículo original en http://bit.ly/1WsqDZF.

EXAGEAR

SACALE MAS PARTIDO A TU ODROID CON TEAMVIEWER, SPOTIFY Y SKYPE

por Gaukhar Kambarbaeva

ELTECHS

ExaGear Desktop es una máquina virtual que permite ejecutar aplicaciones de Linux x86 directamente sobre dispositivos ARM simultáneamente con aplicaciones nativas. También te permite ejecutar aplicaciones de Windows x86 en plataformas ARM utilizando Wine (http://bit.ly/1uHLDzo). La principal ventaja de ExaGear es su excepcional rendimiento. La versión actual de ExaGear Desktop ofrece hasta un 80% de rendimiento sobre las aplicaciones ARM nativas de media, tal y como se describe en t http://bit.ly/20Ks9aK.

Aunque QEMU permite a los usuarios ejecutar aplicaciones x86 en dispositivos ARM, no ofrece un buen rendimiento. Muchos desarrolladores de aplicaciones x86 no se toman la molestia de optimizar el rendimiento de sus aplicaciones, puesto que la desarrollan con una CPU Intel i7. Si su aplicación funciona correctamente en esta máquina, no continúan trabajndo en su rendimiento. Las CPUs ARM presentan un rendimiento menor, pero son más eficientes en cuanto a consumo, más baratas y más novedosas. Con la espectacular sobrecarga que tiene QEMU (5-10 veces), no conseguirás ejecutar aplicaciones x86 con un rendimiento aceptable. ExaGear Desktop es un emulador de última generación que utiliza traducción binaria, que proporciona pequeña sobrecarga en el rendimiento.

Uno de los mejores dispositivos ARM para ejecutar aplicaciones x86 es el ODROID-XU4. Actualmente es el mejor dispositivo del mercado en cuanto a rendimiento. Cuenta con una CPU octacore Samsung Exynos 5422 a 2GHz. Combinando el alto rendimiento del ODROID-XU4 y la eficiencia del ExaGear Desktop, puedes ejecutar una gran cantidad de famosas aplicaciones x86 en un XU4 sin problemas y con una muy buena experiencia de usuario.

Además, ExaGear Desktop te permite interactuar con aplicaciones x86 de la misma manera con la que lo haces con aplicaciones ARM. Tras la instalación simplemente tienes que ir al menú de inicio y ejecutar la aplicación. En este artículo, presentaremos algunos ejemplos reales del funcionamiento de ExaGear a través de la instalación de TeamViewer, Skype y Spotify.

Figura 1 - Acceso directo de ExaGear Desktop en el escritorio y en el terminal x86

Instalar ExaGear

En primer lugar, es necesario instalar ExaGear Desktop en el dispositivo ODROID-XU4. Puede adquirir una licencia ExaGear Desktop en la página web de Eltechs en http://bit.ly/1Q6SxKm. Para ODROID-XU4, tienes que utilizar "ExaGear Desktop for ARMv7". Una licencia es válida por un único dispositivo durante tiempo ilimitado, con futuras actualizaciones gratuitas de ExaGear Desktop.

La versión más reciente de ExaGear Desktop es la 1.4.1, que incluye imágenes de varios sistemas x86: Ubuntu 14.04, Ubuntu 15.04, Debian 7, Debian 8. Existe un script de instalación que instala la imagen x86 en tu sistema ARM, de modo que todo lo que necesita hacer es ejecutar el script en el directorio que contiene los paquetes deb y la licencia.

Puesto que vamos a instalar una versión de Spotify que actualmente sólo funciona en Ubuntu 14.04, instalaremos la imagen de Ubuntu 14.04 x86 ejecutando el script de instalación con las siguientes opciones:

```
$ sudo ./install-exagear.sh
ubuntu-1404
...
ExaGear is activated.
Done!
```

Una vez completada la instalación, haz clic en el acceso directo de ExaGear en tu escritorio, o en la sección de Herramientas de sistema del menú Inicio, y se abrirá una ventana de terminal con sistema X86.

Ahora te encuentras en un entorno x86 que puedes verificar ejecutando el siguiente comando:

```
$ arch
i686
```

Se recomienda actualizar los repositorios en el primer lanzamiento del sistema x86 invitado, después podrás instalar aplicaciones x86 en la ventana de terminal X86:

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

TeamViewer

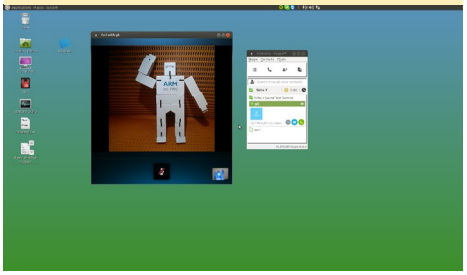
Para instalar TeamViewer, ejecuta los siguientes comandos en el terminal x86 de tu dispositivo ARM, tal como lo harías en un equipo x86:

```
$ sudo apt-get install wget
$ wget http://download.teamview-
er.com/download/teamviewer_i386.
deb
$ sudo dpkg -i teamviewer_i386.
deb
```

Cuando se esté ejecutando el comando, quizás veas el siguiente mensaje:

```
Selecting previously unselected
package teamviewer.
(Reading database ...  7705 files
and directories currently in-
stalled.)
Preparing to unpack teamviewer_
i386.deb ...
Unpacking teamviewer (11.0.53191)
...
dpkg: dependency problems prevent
configuration of teamviewer:
 teamviewer depends on liba-
sound2; however:
  Package libasound2 is not in-
stalled.
 ...
 teamviewer depends on libxfixes3;
however:
  Package libxfixes3 is not in-
stalled.
dpkg: error processing package
teamviewer (--install):
```

Figura 3 - Skype con vídeo



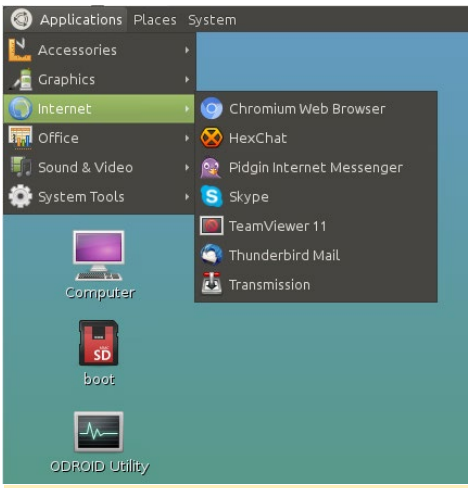


Figura 2 - TeamViewer y Skype en el menú de inicio

```
 dependency problems - leaving
unconfigured
Errors were encountered while
processing:
 teamviewer
```

Este mensaje es sólo un aviso que te informa de que el paquete TeamViewer depende de algunos otros paquetes que deben instalarse. El siguiente comando te ayudará a solventar esta situación instalando los paquetes necesarios y finalizando la instalación de TeamViewer:

```
$ sudo apt-get install -f
```

Ahora puedes ejecutar TeamViewer desde el menú Inicio, y utilizarlo como si lo estuvieras ejecutando en una máquina x86. Tenga en cuenta que después de reiniciar del sistema, TeamViewer se iniciará automáticamente. Si deseas iniciar o detener el demonio TeamViewer manualmente, asegúrate de hacerlo desde el terminal x86.

Instalar Skype

Para instalar Skype, abre el terminal x86 lanzando ExaGear y ejecutando los siguientes comandos:

```
$ sudo apt-get install pulseaudio
$ wget http://download.skype.com/
linux/skype-debian_4.3.0.37-1_
i386.deb
```

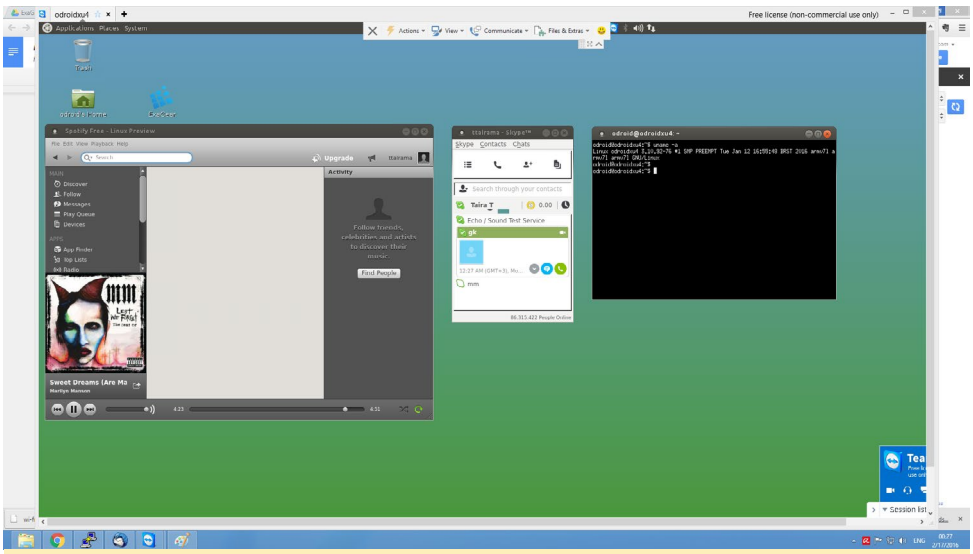


Figura 4 - Skype y Spotify ejecutándose en un ODROID-XU4 conectado a un PC con Windows a través de TeamViewer

```
$ sudo dpkg -i skype-debian_4.3.0.37-1_i386.deb
$ sudo apt-get install -f
```

Una vez completada la instalación, puedes iniciar Skype desde el menú de inicio, con el servicio de mensajería, buzón de voz y videoconferencia funcionando perfectamente.

Instalar Spotify

La instrucción de instalación de Spotify son exactamente las mismas que las aparecen en la página oficial de Spotify en http://spoti.fi/1Wxngk8, excepto que estás ejecutando los comandos sobre un dispositivo ARM en el terminal x86 proporcionado por ExaGear Desktop.

En primer lugar, agrega la clave de firma del repositorio de Spotify y así poder verificar los paquetes descargados, añade el repositorio, actualizar la lista de paquetes disponibles e instala Spotify:

```
$ sudo apt-key adv --keyserver \
  hkp://keyserver.ubuntu.com:80
--recv-keys \
  BBEBDCB318AD50EC6865090613B00F
1FD2C19886
$ echo deb http://repository.
spotify.com \
  stable non-free | sudo tee \
  /etc/apt/sources.list.d/spotify.list
$ sudo apt-get update
$ sudo apt-get install spotify-client
```

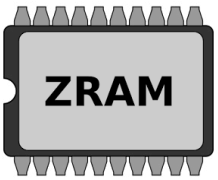
¡Ahora puedes disfrutar de tu música favorita!

Notas

Aunque los sistemas ARM están creciendo rápidamente, todavía hay muchas aplicaciones interesantes que no están disponibles para ARM. ExaGear Desktop hace posible ejecutar estas aplicaciones utilizando un dispositivo ARM, la combinación de ExaGear Desktop y el ODROID-XU4 proporciona una experiencia muy buena.

USANDO ZRAM

EXPANDIR LA MEMORIA MEDIANTE COMPRESION



por Tobias Schaaf

Si comparamos un ODROID con un tradicional PC o portátil, una de las grandes diferencias esta en la cantidad de memoria (RAM) de la que dispone. Mientras que muchos equipos vienen con 4-16GB de RAM o más, los ODROIDs sólamente tiene 2 GB y el C1 y C1+ sólo cuentan con 1 GB de RAM. Esto puede dar lugar rápidamente a problemas, y que los programas que se estén ejecutando se vuelvan muy lentos e incluso se bloqueen por completo. Existe incluso la posibilidad de que todo el sistema llegue a bloquearse. Una forma de evitar este comportamiento es aumentar virtualmente la cantidad de memoria disponible en el ODROID, usando una técnica llamada swap.

Swap

Swap es un archivo o una partición en tu sistema que se utiliza cuando la memoria está llena, con el fin de evitar que un programa o el sistema completo se bloqueen. Al principio, swap se almacena en un disco que, por supuesto, es más lento que la memoria real. Puesto que el disco duro de un ODROID es una tarjeta SD o módulo eMMC, es incluso más lento que un PC convencional que puede utilizar unidades de estado sólido más rápidas (SSD).

zRAM

Es posible que ya han oído hablar de zram, siendo usado en los ODROIDs para mejorar el rendimiento o para aumentar virtualmente la cantidad de RAM disponible. ZRAM es un espacio reservado en la memoria que se utiliza como una partición de intercambio swap que puede comprimir páginas (datos) en la memoria (RAM). Tener la partición swap en memoria es mucho más rápido que almacenarla en una SD o unidad de disco. Lo único que hay que hacer es comprimir los datos, lo cual se puede hacer muy rápido con la CPU.

Clásico swap vs zram

La memoria es un recurso limitado, de modo que la compresión de datos en memoria también está limitada. Con una clásica partición swap, puedes teóricamente tener mucha más memoria RAM virtual (swap) que con zram, pero como he dicho antes, este método es mucho más lento. Por lo tanto, si usamos zRAM, podemos tener una partición swap más rápida con ciertos límites. Incluso es posible combinar zram y el tradicional swap. Zram se utilizaría en primer lugar y cuando te quedas sin espacio de intercambio zram, se usaría la tradicional partición de intercambio swap.

Información general

Usar zram es una muy buena manera de mejorar el intercambio de memoria, aunque debe hacerse con moderación. Si lees artículos en Internet sobre zram, encontraras mucha información, guías y scripts. En todos ellos se recomienda utilizar un máximo del 50% de la memoria disponible para el intercambio zRAM.

Debido a que zRAm comprime las páginas en memoria entre un 30 y un 50% del tamaño original, cuando tienes 2 GB de zRAM, esto equivale alrededor de unos 615 MB de memoria normal. En la práctica, en realidad es más, ya que también hay cierta sobrecarga administrativa. El ODROID C1 tiene un total de 836 MB de memoria RAM disponible, lo cual significa que realmente sólo hay disponible alrededor de 200 MB para que el sistema funcione. El kernel, drivers, registros log, programas en primer plano y las tareas en segundo plano necesitan estos 200 MB de memoria.

Es importante ajustar los valores adecuados para el tamaño del espacio de compresión zRAM. Siempre que un programa necesite datos de zRAM, no los puede utilizar directamente, sino que se primero tiene que descomprimir la información. Si la memoria RAM ya está llena, lo que hay que hacer es eliminar una parte de éstos 200-400 MB de datos activos de la memoria comprimiéndola de nuevo, y luego descomprimir los datos que se necesiten para la tarea actual.

Con más datos en la zRAM, es necesario hacer estos intercambios con mayor frecuencia, lo que da lugar a una mayor carga de la CPU, puesto que está comprimiendo y descomprimiento constantemente. En último termino, todo el sistema estará ocupado con la compresión y descompresión de las páginas en memoria. Cuanta más memoria tengas que comprimir y conforme te acerques al 200% de la memoria física, mayor será la probabilidad de que los datos necesarios estén dentro de las páginas comprimidas.

Uso adecuado

No importa qué tipo de swap uses (Clásico swap, partición zRAM o ambos), el swap es siempre el "último recurso" del sistema para evitar fallos y bloqueos. No pretende ser la configuración por defecto de un sistema. Si tu sistema sólo funciona con intercambio swap, al final tendrás problemas. Si deseas ejecutar un programa que requiera al menos 2 GB de RAM, no es una buena idea ejecutarlo en un dispositivo que disponga de menos de 2 GB, o incluso menos de 1 GB como es el C1. Incluso si tienes el clásico swap o zRAM, ya sabrás que el programa ni siquiera funcionará sin utilizar swap. Este mismo planteamiento se aplica a la ejecución simultánea de muchos programas y apliaciones.

Si Chromium o Firefox necesita 200 MB de RAM para cada sitio web abierto, lanzar 10 a la vez tampoco es una buena idea. Swap sólo debe utilizarse como una medida temporal.

Utilizar zRAM

Puedes descargar la versión más reciente de mis scripts zRAM desde http://bit.ly/1PHq51B e instalarlos con dpkg:

```
$ wget http://bit.ly/1PHq51B \
  -O zram-odroid.deb
$ dpkg -i zram-odroid.deb # or
gdebi zram-odroid.deb
```

Tras su instalación, podrás iniciar y detener zRAM usando los comandos:

```
$ sudo service zram start
$ sudo service zram stop
```

Si estás usando Ubuntu 14.04 o Debian Wheezy, también puedes utilizar el siguiente comando para conseguir información detallada sobre el uso actual de zRAM:

```
$ sudo service zram status
```

Si estás usas un systemd basado en Linux (como Ubuntu 15.04, 15.10, 16.04 o Debian Jessie), usa el comando:

```
$ sudo zramstat
```

Puedes configurar zRAM editando las opciones en el archivo de configuración ubicado en /etc/zram/zram/conf.

Estadísticas

Fraction es la cantidad de RAM en % que se debe utilizar como zRAM. El valor por defecto es 50, lo que significa que el 50% de RAM disponible se podrá utilizar como zRAM. Eso es aproximadamente entre 900 MB a 1 GB en todos los dispositivos excepto en el C1, donde sólo es de unos 418 MB.

Puedes cambiar el valor como quieras: 100 (misma cantidad de zRAM que tu memoria real), 200 (el doble de la cantidad de RAM que tiene el sistema) o incluso 2000 si estás algo loco. Probablemente los mejores valores para zRAM son lo que se encuentran entre 50 y 100% de la memoria disponible.

Threads muestra el número de hilos de ejecución para gestionar zram. El valor predeterminado es 0 y hay tantos hilos de ejecución como núcleos de CPU. Esto creará un número de dispositivos zram más pequeños en /dev/zram* igual a los hilos de ejecución seleccionados. La cantidad de zram disponible se distribuirá de forma uniforme entre los dispositivos zram. Por defecto cuentas con tantos dispositivos zram como núcleos de CPU, pero puedes cambiar esta configuración. Si usas excesivamente zram y utilizas todos los núcleos zram, en el peor de los casos, el sistema no podrá hacer ninguna otra tarea, ya que todos los núcleos están ocupados con la zram. Puedes incluso experimentar con ello, pero no es necesario tocarlo si no quiere.

Notas

Aunque zram se puede comprimir bastante bien al 30-50% del tamaño original, usar grandes cantidades de zram no es recomendable. Cuanto más zram utilices, menos memoria (RAM) tendrás para trabajar, y el sistema no puede funcionar con zram directamente, sino que tienes que descomprimirla primero. Cuanta menor sea la RAM real de la que dispongas, con más frecuencia el sistema necesitará comprimir y descomprimir las páginas en zram. Con grandes cantidades de zram esto ocurre muy a menudo.

Dependiendo del nivel de compresión, tener 2 GB de zram no serviría para nada, causaría bloqueos en aplicaciones o detendría todo el sistema operativo. Por lo tanto, dejar suficiente memoria "real" disponible para que el sistema funcione es crucial, no se debe poner el valor zram demasiado alto. Como he mencionado antes, los valores más seguros son los que están entre el 50 y 100% de tu memoria real, unos valores demasiado altos causan problemas con el tiempo.

Ejemplo

El C1 tiene un total de 836 MB de RAM, cuando utilizas una imagen de escritorio normal. La compresión de la memoria RAM a un 30-50% del tamaño original daría como resultado un tamaño de 1672-2787 MB. Como puedes ver, esto significa que sólo puedes conseguir alrededor de 1,4 GB sobrantes. Dependiendo de los programas que ejecutes, una gran cantidad de zram, como por ejemplo 2 GB no funcionaría y tu memoria se agotaría mucho antes de llegar a ese límite. Además, esto significaría que el 100% de tu RAM tendría que ser comprimida, y el sistema no puede trabajar en datos comprimidos. Es necesario descomprimir los datos en memoria para poder trabajar con ellos, esto se tiene que hacer en la RAM normal.

Suponiendo que dejamos un mínimo de 136 MB, lo cual nos das unos 700 MB de memoria RAM disponible para la compresión. Eso significa que tenemos 1400-2333 MB de zram para usar, y que contamos con un mínimo de 136 MB para trabajar. Todos los programas, drivers, el propio escritorio, y módulos del kernel tienen que compartir 136 MB de memoria para poder trabajar. Cada vez que un programa necesite datos que hay en la zram, una parte de esos 136 MB de memoria disponible necesitan comprimirse y ser quitados de en medio, para que los datos necesarios de zram puedan ser descomprimidos. Con esa pequeña cantidad de memoria RAM, tu sistema estaría constantemente ocupado con la compresión y descompresión de páginas en la memoria, y tu sistema se volvería más y más lento a medida que aumentasen los datos almacenados en zram. Por lo tanto, es importante ajustar la zram a un tamaño adecuado, ya que puede ralentizar el sistema, llegando incluso a crear una situación en la que el sistema no responda, ya que está totalmente ocupado con la compresión y descompresión.

Lecturas

Puedes encontrar más información sobre zRAM en http://bit.ly/1Pt80Gr.

FUNDAMENTOS DE MQTT

EL IOT LO HACE FACIL

por Venkat Bommakanti

MQTT, significa MQ (Messaging Queue) Telemetry Transport, es un invento de IBM, desarrollado como un protocolo de conectividad de máquina a máquina (M2M). Fue diseñado como un sistema de transporte de mensajes publicación/suscripción extremadamente liviano, por lo que es ideal en el ámbito del Internet de las cosas (IoT). Este artículo es una introducción a los conceptos básicos de MQTT utilizando el ODROID-C2, una placa de nueva generación, potente e ideal para el IoT.

Preparar el sistema

Instala la última Imagen oficial del C2 de Hardkernel, luego actualiza el sistema usando los siguientes comandos:

```
$ sudo apt-get update && \
  sudo apt-get upgrade
$ sudo apt-get dist-upgrade && \
  sudo apt-get install linux-
image-c2
$ sudo apt-get autoremove
```

Reinicia el sistema y asegúrate de que el dispositivo esta operativo y tiene acceso a Internet.

Instalar requisitos previos

El ODROID-C2 no incluye soporte para Reloj en Tiempo Real (RTC). Sin embargo, es conveniente disponer de un reloj preciso cuando se trata de eventos relacionados con el IoT. Con ese fin, ejecuta el siguiente comando:

```
$ sudo apt-get install ntp
```

Después, actualiza el sistema para sincronizar el reloj del sistema con un servidor NTP una vez que el arranque y la conectividad de red sean estables. Actualiza el archivo /etc/rc.local con el siguiente fragmento, se requieren privilegios de root. Luego reinicia el sistema y asegúrate de que el reloj del sistema funciona correctamente:

```
( /etc/init.d/ntp stop
until ping -nq -c3 8.8.8.8; do
  echo "Waiting to setup ntp
based clock..."
done
ntpd -gq
/etc/init.d/ntp start )&
```

Hardkernel presentará un complemente RTC para el ODROID-C2 en un futuro cercano. Si se utiliza este complemento, la modificación de software indicada no será necesaria.

Mosquitto

Mosquitto es un bróker de mensajería basado en el proyecto eclipse de código abierto que implementa MQTT. Incluye utilidades de testeo de publicación y suscripción de mensajes que pueden utilizarse para validar la instalación.

Teclea el siguiente comando para instalar mosquitto y los complementos de software relacionados:

```
$ sudo apt-get install mosquitto
\
  apparmor mosquitto-clients
```

Verifica la instalación utilizando el siguiente comando:

```
$ mosquitto -h
mosquitto version 1.4.8 (build
date Fri, 19 Feb 2016 12:03:16
+0100)
mosquitto is an MQTT v3.1 broker.
Usage: mosquitto [-c config_file]
[-d] [-h] [-p port]
-c : specify the broker config
file.
-d : put the broker into the
background after starting.
-h : display this help.
-p : start the broker listening
on the specified port.
     Not recommended in conjunc-
tion with the -c option.
-v : verbose mode - enable all
logging types. This overrides
     any logging options given
in the config file.
See http://mosquitto.org/ for
more information.
```

Mosquitto 1.4.8 es la última versión disponible. Lanza el bróker en segundo plano utilizando el siguiente comando, que activa un registro log detallado:

```
$ mosquitto -d -v
```

Valida el inicio con el siguiente comando:

```
$ ps aux | grep mos
mosquit+   431  0.0  0.1   6340
2348 ?        S      13:34   0:01
/usr/sbin/mosquitto -c /etc/mos-
quitto/mosquitto.conf
```

Testear MQTT

En primer lugar, crea un archivo de texto para almacenar la carga de mensaje:

```
$ cd ~ && mkdir zBU && \
  cd zBU && mkdir mqtt && cd mqtt
$ touch pub.txt
```

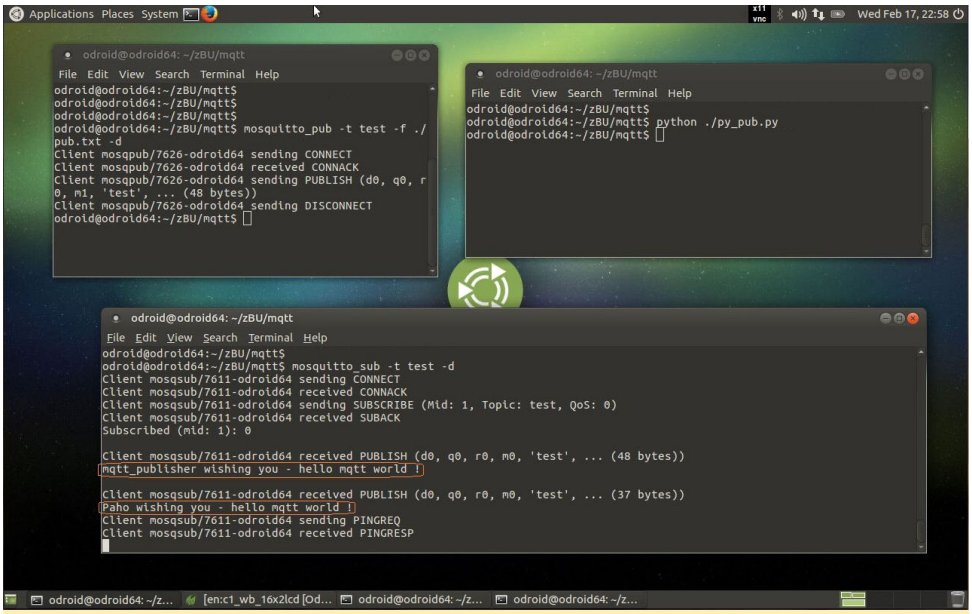


Figura 1: Suscripción y publicación de mensajes usando las herramientas Paho y Mosquitto

Edita el archivo y añade la siguiente línea. Luego, guarda el archivo de carga:

```
mqtt_publisher wishing you -
hello mqtt world !
```

Lanza el suscriptor y publica un mensaje usando el editor:

```
$ mosquitto_sub -t test -d
$ mosquitto_pub -t test -f ./pub.
txt -d
```

Ten en cuenta que el tema de la "conversación" es una prueba. La Figura 1 muestra el resultado en una ventana de terminal donde se ejecuto el suscriptor.

Clientes MQTT Paho

Paho, otro proyecto Eclipse que proporciona clientes MQTT de código abierto para varios lenguajes como Java, C y Python. Instala el cliente Python paho usando los siguientes comandos:

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

Crea un editor pytón de prueba utilizando los siguientes comandos:

```
$ cd ~ && mkdir zBU && cd zBU &&
mkdir mqtt && cd mqtt
$ touch py_pub.py && chmod 755
py_pub.py
```

Edita el archivo y añade este contenido, usando datos específicos del C2:

```
#!/usr/bin/python

# Copyright (c) 2014 Roger Light
<roger@atchoo.org>
#
# All rights reserved. This pro-
gram and the accompanying materi-
als
# are made available under the
terms of the Eclipse Distribution
# License v1.0 which accompanies
this distribution.
#
# The Eclipse Distribution Li-
cense is available at
#    http://www.eclipse.org/org/
documents/edl-v10.php.
#
# Contributors:
#     Roger Light - initial
implementation

# This shows an example of using
the publish.single helper func-
tion.
#
# This is a modified version of
pub-single.py
# By Venkat Bommakanti

import sys
try:
      import paho.mqtt.publish as
publish
except ImportError:
      # This part is only re-
quired to run the example from
within the examples
      # directory when the module
itself is not installed.
      #
      # If you have the module
installed, just use "import paho.
mqtt.publish"
      import os
      import inspect
      cmd_subfolder = os.path.
realpath(os.path.abspath(os.
path.join(os.path.split(inspect.
getfile( inspect.currentframe() ))
[0],"../src")))
      if cmd_subfolder not in
sys.path:
      sys.path.insert(0, cmd_sub-
folder)
      import paho.mqtt.publish as
publish

publish.single("test", "Paho
wishing you - hello mqtt world
!", hostname="odroid64")
```

Ejecuta la aplicación de python:

```
$ python ./py_pub.py
```

En un futuro artículo MQTT abordare la publicación de datos de temperatura, precisión y altitud usando la Placa Meteorológica ODROID con el C2 y una solución Nore-RED

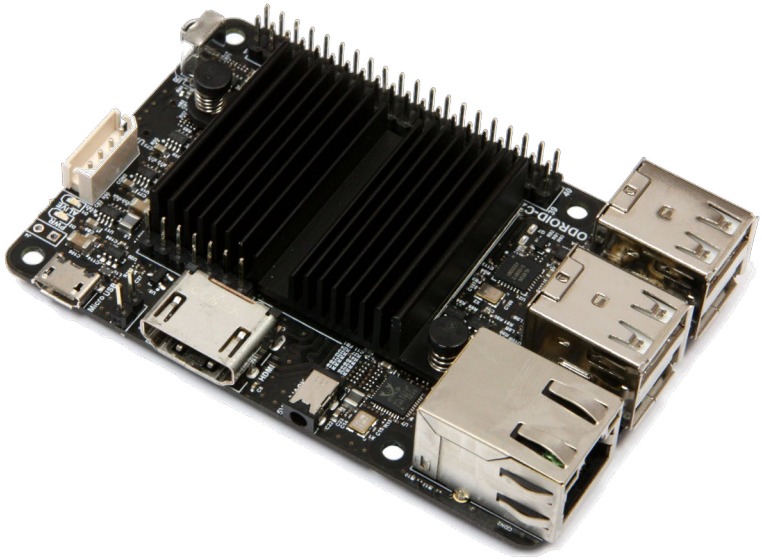
Referencias

http://mqtt.org/
http://mosquitto.org/
http://bit.ly/24lowgd
http://bit.ly/1SLnQwL

ODROID-C2

POTENCIA 64 BIT A BAJO COSTE

por Justin Lee



El ODROID-C2 ha sido valorado como el ordenador de placa reducida más potente y económico que existe, además de ser un dispositivo extremadamente versátil. Con un procesador de cuatro núcleos ARM de 64 bits, una avanzado GPU Malí y un puerto Ethernet Gigabit, puede funcionar como un sistema de cine en casa, un ordenador de uso general para navegar por internet, para ejecutar juegos y consultar redes sociales, como herramienta de trabajo para el colegió o la oficina, como prototipo para realizar pequeñas modificaciones de hardware, como controlador para proyectos de domótica, como estación de trabajo para programar y mucho más. Está disponible en la tienda de Hardkernel (http://bit.ly/1Pp4J7w) por 40$.

Algunos de los modernos sistemas operativos que se pueden ejecutar en el ODROID-C2 son Ubuntu, Android, Arch Linux y Debian, con miles de paquetes de software de código abierto totalmente gratis. El ODROID-C2 es un dispositivo ARM - la arquitectura más utilizada en los dispositivos móviles y en la informática de 64-bit embebida. El pequeño tamaño de su procesador ARM, su reducida complejidad y su bajo consumo de energía hacen que sea perfecto para desarrollar controladores embebidos y pequeños dispositivos que podemos llevar encima.

Especificaciones

- CPUs de cuatro núcleos Amlogic ARM® Cortex®-A53(ARMv8) a 2Ghz
- GPU Mali ™ -450 MP3 (OpenGL ES 2.0 / 1.1 habilitado para Linux y Android)
- VPU capaz de reproducir H.265 4K/60 FPS y H.264 4K/30fps
- 2 GB SDRAM DDR3
- Conexión HDMI 2.0 4K/60Hz
- Puerto Ethernet Gigabit
- Puerto GPIO 40pin + I2S 7pin
- Ranura para almacenamiento flash eMMC5.0 HS400 / Ranura para tarjetas MicroSD UHS-1 SDR50
- 4 puertos USB 2.0 Host, 1 puerto USB OTG (para datos y alimentación)
- Receptor Infrarrojos (IR)
- Ubuntu 16.04 y Android 5.1 Lollipop basado en el Kernel 3.14 LTS
- Las dimensiones de la placa son idénticas a la del ODROID-C1+

Comparación del mercado

Todas las placas comparadas en la Tabla 1 son ordenadores de placa reducida ARM compatibles con Linux cuyo coste es inferior a los 40$.

Table I - ODROID-C2 vs ODROID-CI vs Raspberry Pi2

	ODROID-C2	ODROID-C1	RPi 2 Model B
CPU	Amlogic S905 SoC 4 x ARM® Cortex®-A53 2GHz ARMv8 Architecture @28nm	Amlogic S805 SoC 4 x ARM® Cortex®-A5 1.5GHz ARMv7 Architecture @28nm	Broadcom BCM2836 4 x ARM® Cortex®-A7 900Mhz ARMv7 Architecture @40nm
GPU	3 x ARM® Mali™-450MP 700MHz	2 x ARM® Mali™-450MP 600MHz	1 x VideoCore IV 250MHz
RAM	2GB 32bit DDR3 912MHz	1GB 32bit DDR3 792MHz	1GB 32bit LP-DDR2 400MHz
Flash Storage	Micro-SD UHS-1 @83Mhz/SDR50 or eMMC5.0 storage option	Micro-SD UHS-1 @78Mhz/SDR50 or eMMC4.5 storage option	Micro-SD @50Mhz/SDR25, No eMMC storage option
USB2.0 Host	4 Ports	4 Ports	4 Ports
USB2.0 Device / OTG	1 Port for Linux USB Gadget driver	1 Port for Linux USB Gadget driver	No
Ethernet/LAN	10/100/1000 Mbit/s	10/100/1000 Mbit/s	10/100 Mbit/s
Video Output	HDMI 2.0 4K/60Hz	HDMI	HDMI 1.4 / RCA /DSI
Audio Output	HDMI / I2S	HDMI	HDMI / 3.5mm Jack
Camera Input	USB 720p	USB 720p	MIPI CSI 1080p
Real Time Clock	No(unless using an add-on module)	YES (On-board RTC)	No(unless using an add-on module)
IR Receiver	YES (On-board IR Sensor)	YES (On-board IR Sensor)	No(unless using an add-on module)
IO Expansion	40pin port (GPIO/UART/I2C/ADC) 7pin port (I2S)	40pin port (GPIO/UART/SPI/I2C/ADC) 7pin port (I2S) : ODROID-C1+ only	40pin port (GPIO/UART/SPI/I2C/I2S)
ADC	10bit SAR 2 channels	10bit SAR 2 channels	No (unless using an add-on board)
Size	85 x 56mm (3.35" x 2.2")	85 x 56mm (3.35" x 2.2")	85 x 56mm (3.35" x 2.2")
Weight	40g (1.41 oz)	40g (1.41 oz)	42g (1.48 oz)
Price	$44	$35	$35

Hardware

El C2 tiene muchas ventajas sobre la Raspberry Pi 2. El procesador es un quad-core S905 a 2GHz de Amlogic con 2 GB de RAM DDR3, Ethernet Gigabit y Receptor IR. El tamaño de este ordenador es de tan sólo 85 x 56 mm con un peso de 40g, y su funcionamiento es muy silencioso. Su consumo medio de energía está en torno a los 2-5W y su portabilidad es inmediata, puesto que te lo puedes llevar en el bolsillo de la camisa.

Una de las características más potentes del ODROID-C2 es la fila de pines GPIO (Entrada/Salida de Propósito General) a lo largo del borde del dispositivo. Estos pines son una interfaz física entre la placa y el mundo exterior. El cabezal de 40 pines incluye funciones PWM, I2C, UART, ADC y GPIO

Una tarjeta MicroSD SD UHS-1 compatible con el estándar SD 3.01, así como el módulo eMMC más rápido, se pueden solicitar con el C2, que vendrá con el popular sistema operativo Ubuntu ya instalado. Inserta la tarjeta SD en la ranura, conecta un monitor, un teclado y los cable de Ethernet y de alimentación. ¡Esto es todo lo que necesitas hacer para utilizar el C2! Navega por Internet, ejecuta juegos y programas de ofimática, edita fotos, desarrolla software y visualiza vídeos al instante.

El receptor IR y las características ADC del ODROID-C2 ofrecen muchas posibilidades para la creación de grandes proyectos de bricolaje.

Rendimiento CPU/RAM

Hemos ejecutado varias pruebas de rendimiento para medir la potencia de cálculo del C2. Las mismas pruebas se realizaron también a la Raspberry Pi 2, ODROID-C1, ODROID-U3 y

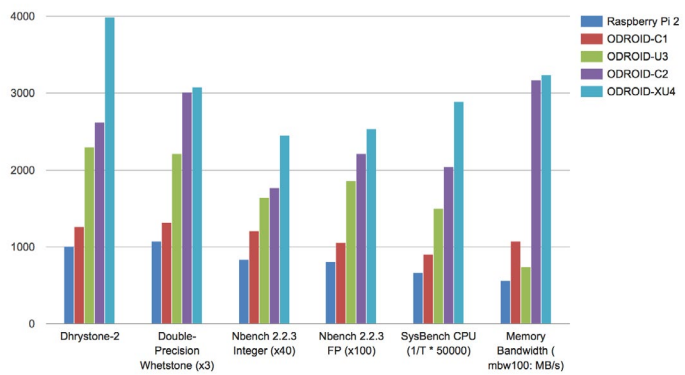


Figura 1 – Gráfica de las pruebas de rendimiento sobre la CPU/RAM

ODROID-XU4. Los valores de los resultados de las pruebas fueron graduados uniformemente de cara a la comparación. La potencia de cálculo del C2 fue cuantificada como 2-3 veces más rápida que la última Raspberry Pi 2 gracias a los núcleos Cortex-A53 a 2 GHz y al mayor ancho de banda de memoria. El excelente rendimiento de los 2 GB de RAM DDR3 es una ventaja adicional que permite que la mayoría de los programas se ejecuten sin problemas en el C2.

Rendimiento almacenamiento

El C2 puede arrancar desde una tarjeta MicroSD o un módulo de eMMC, que se selecciona con un interruptor de fácil acceso. La interfaz MicroSD también soporta el excelente rendimiento del modo UHS-1. El acceso a un archivo de 512 MB (lectura / escritura) en las distintas opciones de almacenamiento muestra diferencias de rendimiento muy claras.

El almacenamiento eMMC 5.0 tiene una tasa de lectura 7 veces más rápida que la tarjeta MicroSD Clase-10. La tarjeta MicroSD UHS-1 es 2 veces más rápida que la tarjeta MicroSD Clase-10. Las tarjetas MicroSD UHS-1 representan una buena opción de bajo coste para muchas aplicaciones. No obstante los nuevos módulos eMMC Black también son muy económicos, gracias a la nueva bajada de precios.

Rendimiento Ethernet

El C2 cuenta con un controlador Ethernet Gigabit.La velocidad de transmisión bidireccional llego a los 900Mbps. Gracias al buffer Tx duplicado en S905, la velocidad de subida es el doble de rápido que en el C1.

Visualización

Como se muestra en la siguiente imagen, la salida HDMI 4K permite mostrar una increible pantalla de escritorio con una resolución de 3840x2160 ultra-alta definición, preparada para la nueva generación de televisores y monitores.

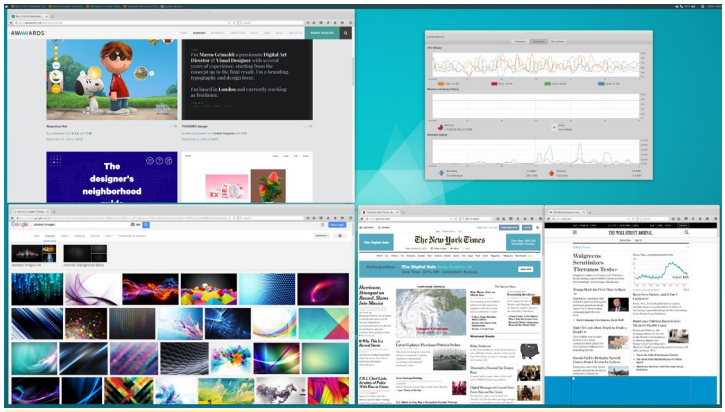


Figura 2 – Captura de pantalla de un escritorio 4K

Notas

En el C2 no existe el bus Serial Peripheral Interface (SPI) o el Real Time Clock (RTC), ya que el SoC S905 no los ofrece. Esto tiene como consecuencia que muchas placas adicionales basadas en SPI no sean compatibles con el C2. Sin embargo, estamos considerando la posibilidad de crear una placa RTC adicional. Además, el problema de la combinación Alfa ha sido resuelta en el S905, por lo que ya no necesitaremos utilizar más el sistema DDX.

Las imágenes de sistema operativo y las guías de desarrollo están disponibles en nuestra wiki en http://bit.ly/1Trq5Ef.

KERNEL LINUX EN TIEMPO REAL

EL MEJOR ALIADO DEL ODROID-CO

por Andrew Ruggeri

Recientemente, el usuario @chlorisdroid de los foros ODROID (http://bit.ly/1PIuRIR) ha publicado un Kernel 3.10 Linux en Tiempo Real (RT) preventivo para el ODROID-C0, C1 y C1+. El hilo del foro, disponible en bits.ly/1QAF0af, incluye enlaces GitHub al código fuente del kernel modificado, además de información adicional. Con el reciente lanzamiento del ODROID-C0, una placa de desarrollo dirigida al Internet de las cosas (IoT) y un uso portátil, un Kernel Linux en tiempo real supone una gran ventaja.

En este artículo describiremos qué es exactamente un Kernel Linux en tiempo real. Espero que cuando termines de leerlo, te encuentres tan entusiasmado como yo, por todas las ventajas que un Kernel RT puede aportar al ODROID-C0, C1 y C1+. Tal vez te estés preguntando, si un Kernel Linux en Tiempo Real es tan fantástico, ¿por qué no lo incluye el kernel Linux por defecto que proporciona Hardkernel? La respuesta es simple, aunque los Kernels estándar y en Tiempo Real son muy similares, se utilizan para diferentes fines.

Qúe hay detrás del nombre

El primer paso para conocer lo que ofrece este kernel en tiempo real modificado es entender su nombre. El título completo usado en el post del foro para describir el Kernel es "Real Time Linux Kernel 3-10-80-rt88 PREEMPT_RT". Lo importante está al final "PREEMPT_RT". Los números del medio hacen referencia a la versión.

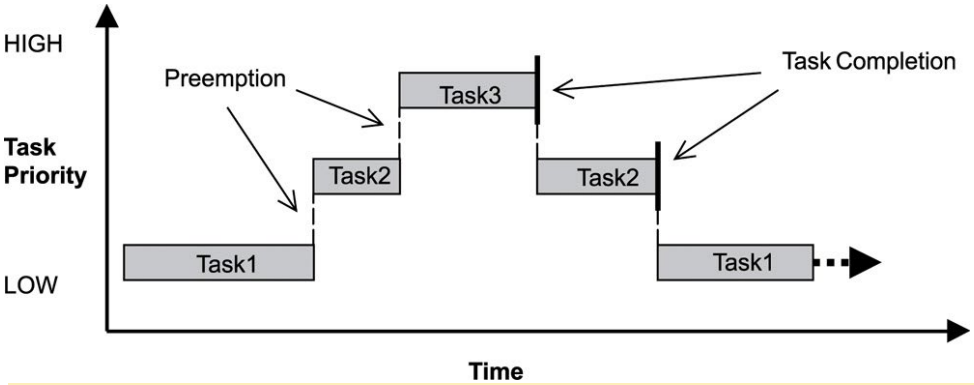


Figura I: Spinlock de una tarea de baja prioridad

"PREEMPT_RT" quiere decir Tiempo Real Preventivo. El kernel Linus estándar tras la versión 2.6 es parcialmente preventivo por defecto. Un Kernel que es preventivo tiene la capacidad de cambiar la tarea en la que está centrada la CPU. La CPU puede estar trabajando en una tarea y luego decirle que se centre temporalmente a otra. Una vez que haya completado esa otra tarea, vuelve a centrarse en la anterior tarea. Una tarea que es sustituida por otra se denomina "cambio de contexto". A las tareas se les da una prioridad para determinar qué tareas son las más importantes y por tanto llamar la atención de la CPU. Una tarea de baja prioridad, como el escaneo en segundo plano de dispositivos wifi será interrumpida con frecuencia por las tareas de mayor prioridad que vayan surgiendo. Por otro lado, una tarea de alta prioridad estará mucho más tiempo en la CPU que una de baja prioridad.

Una cuestión importante es que una tarea puede mantener su posición usando un spinlock. Esto significa que incluso si aparece una tarea de mayor prioridad, la tarea en spinlock permanecerá no siendo interrumpida. La figura 1, cortesía de embeddedlinux.org.cn, muestra un ejemplo de una tarea de baja prioridad (Task 1), que se ve interrumpida por una tarea de alta prioridad (Task 2). La Tarea 2 es luego sustituida a su vez por una tarea de mayor prioridad (Task 3). Una vez que se haya completado la tarea de mayor prioridad (Task 3), la siguiente tarea con más prioridad toma de control (Task 2). La misma situación se puede observar entre la Tarea 2 y la Tarea 3.

El Tiempo real es la parte más importante de la descripción del Kernel, la cual lo diferencia de un Kernel normal. Un sistema operativo en tiempo real (RTOS) es un sistema que utiliza los plazos y las limitaciones de tiempo para completar las tareas. El principal uso de un RTOS es el de ser determinista, que significa que el sistema operativo va a hacer lo que tú quieras en el momento que tú quieras. Sin embargo, ten en cuenta que esto no hace que un RTOS sea más rápido, o que vayas a conseguir

un mejor rendimiento de ejecución de tareas que en un sistema no RTOS. De hecho, es todo lo contrario.

Steven Rostedt, encargado de actualizar el Parche en Tiempo Real Preventivo, es a menudo citado por su artículo " Cuanto más determinista seas, menos rendimiento conseguirás " en relación a un Sistema en Tiempo Real. La Figura 2, cortesía de Altera (http://bit.ly/1Xu3Vl3), muestra la distribución de la fluctuación comparando 2 kernels, uno con el parche RT y otro sin él. La figura 3, proporcionada por Red Hat (http://red.ht/24b4oNN), muestra el tiempo de respuesta de una tarea en un kernel RT frente a otro sin RT. Los resultados de la Figura 2 muestran que el kernel RT presenta una desviación estándar y un sesgo significativamente menor. Una pequeña desviación estándar es lo que hace que el kernel RT sea interesante, ya que esto demuestra que la temporización es muy uniforme.

La figura 3 también muestra la regularidad de la latencia del kernel en Tiempo Real, donde el kernel sin RT (en rojo) alcanza unos 175 microsegundos, y las tareas de RT (en verde) tienen un alcance más limitado de unos 20 microsegundos. Si observamos la típica latencia en la figura 3, los tiempos sin RT son comúnmente menores en comparación con los tiempos con RT. Esto es de esperar ya que, como se ha mencionado anteriormente, se trata de un inconveniente al aumentar el determinismo.

En un RTOS, a una tarea se le da un límite de tiempo y su ejecución debe completarse en ese límite. Existen dos términos comúnmente usados para referirse a cómo un proceso se comporta en un RTOS: Rígido y Flexible. Un Tiempo Real rígido significa que si una tarea no se ejecuta en el tiempo límite, ésta tendrá problemas o fallará por completo. Con un Tiempo Real flexible si una tarea no se completa dentro de su tiempo, surgirán problemas menores o insignificantes. El Kernel Linux, tras la versión 2.6, únicamente está configurado para gestionar límites de tiempo flexibles, El parche PREEMPT_RT añade la posibilidad de usar los límites de tiempo rígidos.

Figura 2 - Distribución de la fluctuación entre un kerenl con RT y un kernel sin RT

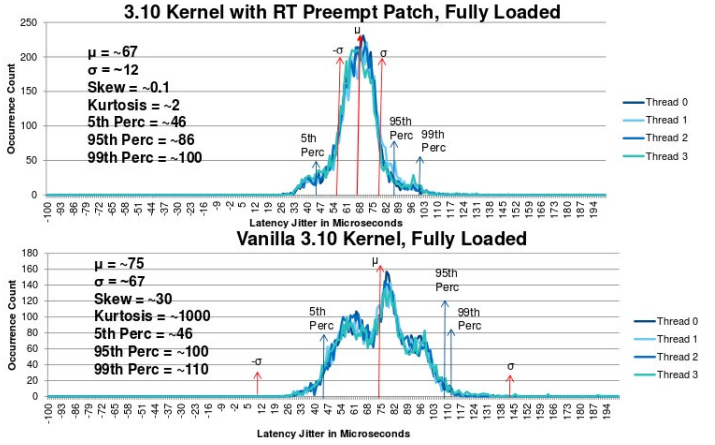
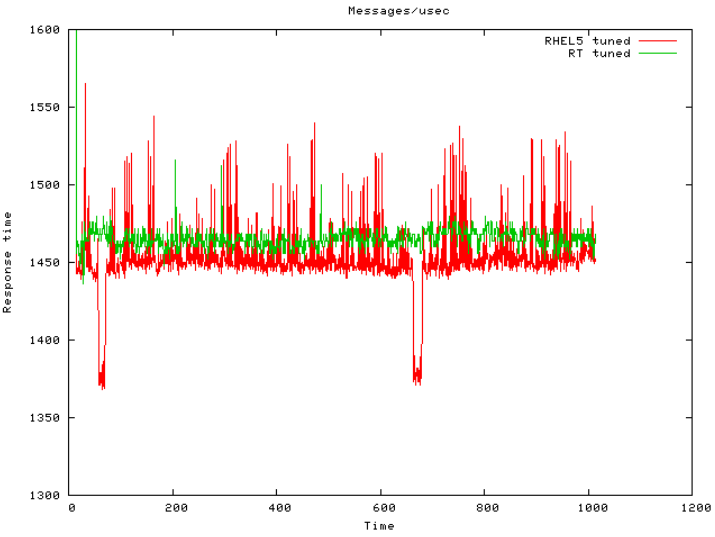


Figura 3 - Latencia entre un kernel con RT y un kernel sin RT



Cómo funciona

El parche PREEMPT_RT funciona cambiando el comportamiento de algunos elementos clave. El parche aspira a ser totalmente preventivo, incluso más allá de lo que un kernel considera preventivo. Esto significa que los spinlocks, que pueden detener un cambio de contexto, no tienen sentido si el objetivo es ser totalmente preventivo. Los Spinlocks en el kernel RT son tratados como mutexes perdiendo su capacidad para bloquear una tarea. Además, las secciones críticas marcadas con rwlock_t y spinlock_t tampoco son preventivas. El cambio más importante es para el comportamiento de las interrupciones, que funcionan como hilos de ejecución cuando se activan. Pueden encontrar más detalles técnicos sobre el funcionamiento del parche RT en los enlaces que aparecen al final de este artículo.

El ejemplo más claro de cuándo utilizar un RTOS viene de la mano de Abraham Silberschatz en su libro Fundamentos de la operación: "Un sistema de tiempo real se utiliza cuando los requisitos de tiempo rígidos han sido colocados sobre la actividad de un procesador o flujo de datos". Con esto en mente, es fácil ver por qué un RTOS se puede encontrar en una gran variedad de lugares y dispositivos, tales como aviones, trenes, reproductores de MP3 y quadcopters. Cualquier aplicación donde se detecte que son necesarias tareas sensibles al tiempo es una posible candidata para usar un RTOS.

Lecturas recomendadas

Presentación del 2013 en el ELC por Steven Rostedt
http://bit.ly/1Lqz8Pl

Wiki Linux en Tiempo Real
http://bit.ly/1OQlDcv

Blog de @chlorisdroid
http://blog.georgmill.de

Post del foro Linux en Tiempo Real de @chlorisdroid
http://bit.ly/1QAF0af

RESETEAR EL TONER

PROLONGA LA VIDA DE LOS CARTUCHOS DE TU IMPRESORA LASER

por @Jojo

He creado un micro-proyecto para resetear el contador de tóner restante de mi impresora láser Samsung CLX-3175N. Algunas impresoras láser Samsung son enviadas con cartuchos de tóner precargados en el arranque. Estos cartuchos no tienen un contador individual para las páginas y el tóner, de modo que la impresora cuenta estos valores por sí misma, siempre y cuando no se estén utilizando cartuchos con chips contadores individuales.

Sin embargo, esta es una práctica comercial muy mala de Samsung: para cada cartucho, presuponen cuanto tóner se utiliza y cuántas páginas se pueden imprimir. Si la impresora cree que no tiene tóner, se niegan a imprimir sin importar la cantidad de tóner que realmente queda.

Por otro lado, muchos modelos de impresora Samsung tienen EPROMS internas para almacenar información, como son los valores de uso de todos los componentes internos. Determinando la dirección de la EEPROM que la impresora utiliza para almacenar cierta información, es posible resetearla. Este es el objetivo de mi proyecto.

Diseño preliminar

Deseaba que el proceso de reseteo fuese lo más cómodo posible. Para lograr esto, decidí quitar la EEPROM SMD interna y reemplazarla con una externa DIP. Retiré la EEPROM de la PCB principal de la impresora y soldé cables a las almohadillas EEPROM de la placa principal. Los cables se encuentran en un lugar dentro de la impresora accesible desde el exterior.

Después, desarrolle una PCB con una ranura para la nueva EEPROM DIP y un microcontrolador, porque quería conectar el microcontrolador a la EEPROM para poder resetear de vez en cuando los valores de tóner. Una vez terminada la PCB, me di cuenta de que era demasiado vago como para escribir el programa del microcontrolador, así que necesitaba buscar otra opción

Segunda estrategia

Tras extraer la EEPROM de la impresora, necesitaba una forma cómoda de leer y escribir en ella. Es cuando pensé en usar mi querido ODROID C1. Tras realizar algunas indagaciones por internet, descubrí que es bastante fácil conectar una memoria EEPROM a un SBC como el C1. He encontrado un artículo escrito por alguien que hizo básicamente lo mismo que lo que yo quería hacer, de modo que hice otra pequeña PCB que me permitiera conectar una memoria EEPROM a mi C1. En ese momento, este proceso no era muy adecuado ya que tenía que conmutar la EEPROM entre mi impresora y el C1, no obstante me llevo menos de un minuto hacerlo.

Escribí un script automático que crea una backup de la memoria EEPROM con fecha de registro, luego reinicie los valores para cuantificar el tóner usado y el restante. Para que el script funcione, instala primero la aplicación dialog:

```
$ sudo apt-get install dialog
```

A continuación, instala una herramienta para escanear el bus I2C:

```
$ sudo apt-get install i2c-tools
```

Opcionalmente, puedes instalar un editor hexadecimal por si quieres indagar aún más:

```
$ sudo apt-get install hexedit
```

A continuación, descarga, compila e instalar la herramienta eeprog. Existe una versión modificada del programa que puedes utilizar, la cual soporta una opción de “retrardo” cuando se realiza una operación de escritura. Descarga el archivo desde http://bit.ly/1OkTf0Z y extraerlo en tu directorio home. A continuación, dirígete al directorio donde ha extraído el programa y compílalo para crear un programa ejecutable:

```
$ cd ~/eeprog-0.7.6-tear12
$ make
```

Lanza el módulo I2C:

```
$ sudo modprobe aml_i2c
$ sudo echo “aml_i2c” >> /etc/
modules
```

Una vez que el módulo se haya cargado, conecta la EEPROM al bus I2C de tu C1, que debe detectarse en la dirección 0x50 ejecutando el siguiente comando:

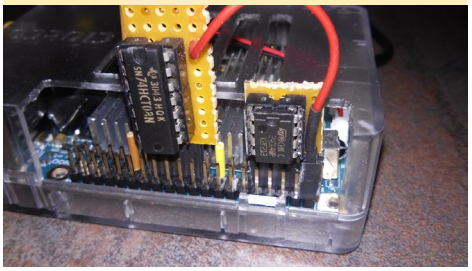
```
$ sudo i2cdetect -y 1
```

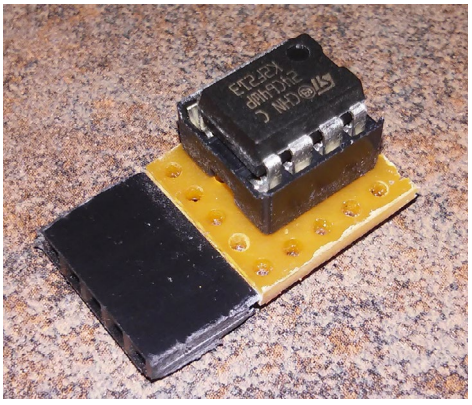
Por último, descarga mi script desde http://bit.ly/1mImpB3, guardalo en el mismo directorio que contiene eeprog y ejecútalo:

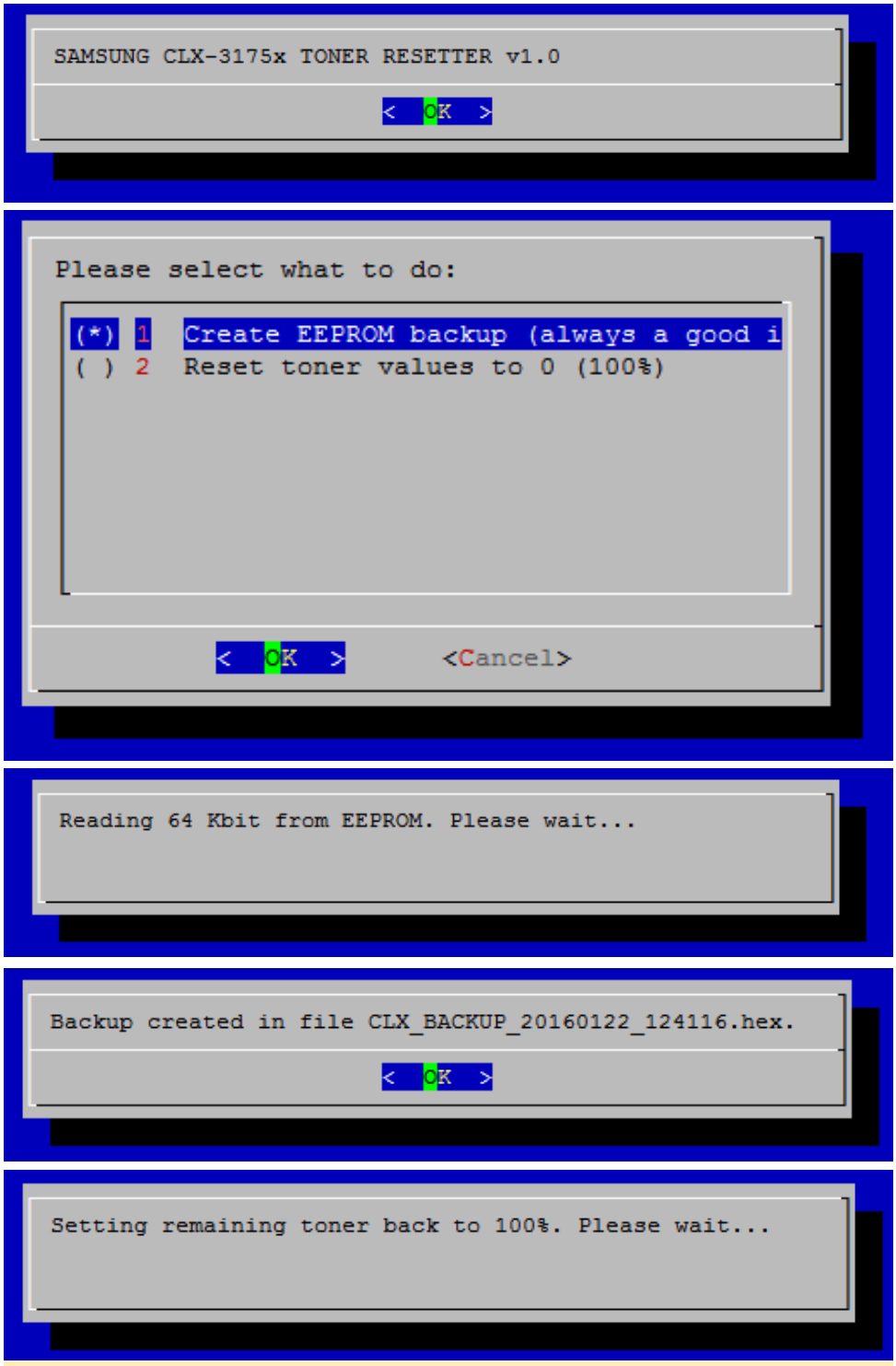
```
$ sudo bash EEPROM_Resetter_v1.sh
```

Selecciona backup y resetea. Luego, conecta la memoria EEPROM en la impresora de nuevo. Todo esto te permite imprimir todo lo quieras y sólo tendrás que rellenar los cartuchos cuando realmente estén vacíos.

Figuras 1 - 3 - La EEPROM conectada al CI







Figuras 4 - 8 – Capturas de pantalla de la aplicación de reseteo de tóner

Mejoras

El script no es muy dinámico, puesto que sólo cuenta con las funciones imprescindibles para mí. En el futuro, podría aplicar un parámetro para diferentes impresoras usando diferentes direcciones dentro de la EEPROM, así como incluir su detección automática. Para comentarios y preguntas, por favor visita el post original en http://bit.ly/1U8R4F2.

Lecturas recomendadas

Proyecto similar:
http://bit.ly/1QPJidO

Programa EEPROM de Raspberry Pi:
http://bit.ly/1L0mkUF

Diseño del hardware:
http://bit.ly/1RdpzHP

CONOCER A UN ODROIDIAN

CHRISTOPHER DEAN (@TPIMP)
DESARROLLADOR QT5 DE EXITO Y EXPERTO EN HARDWARE

editado por Rob Roy

De izquierda a derecha: Storm, Sheldon, Rogue, Nicole, Christopher, Peyton

Por favor, Háblanos un poco sobre ti.

Mi nombre es Christopher Dean. He estado trabajando con placas de desarrollo ODROID durante más de 4 años. Conseguí mi primera ODROID por curiosidad. Tras el lanzamiento del ODROID-U3, empecé a usarlos en proyectos escolares. En los últimos 8 años, he estado estudiado muchos aspectos del Software informático, simulación y diseño de hardware. Actualmente trabajo y soy dueño de protoze, que es una compañía de nueva creación de investigación tecnológica. Me tiene muy ocupado y los días se vuelve largos, pero me gustan los desafíos tan diversos que supone ser dueño de un negocio. Vivo en el magnífico estado de Oregon con mi maravillosa esposa Nicole y nuestros cuatro mascotas, que son como nuestros hijos: Sheldon, Peyton, Rogue, y Storm.

¿Cómo fueron tus inicios con los ordenadores?

Como la mayoría de la gente que nació en la década de los 80: ¡los videojuegos! Mi abuelo nos dio un viejo ordenador con Duke Nukem 2 y otros juegos de DOS que rápidamente aprendí a "ejecutar". Con el tiempo, me cambié a Windows 3.1 con un juego llamado "Raptor: Call of the Shadows" y un joystick Sidewinder. A medida que fui creciendo fui utilizado los ordenadores para hacer trabajos escolares y editar mis secuencias de patinaje. He sido y siempre seré un jugador de PC.

¿Qué te atrajo de la plataforma ODROID?

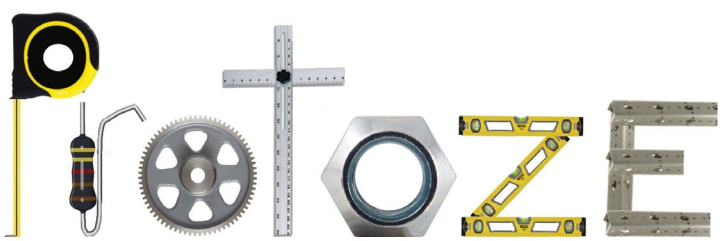
Al principio, me encanto la simple potencia que ofrecía el ODROID-U2 por su bajo coste y pequeño tamaño. Con el tiempo, me atrajo cada vez más la calidad de las placas, combinada con la tenacidad de la comunidad. Sin el trabajo de la comunidad y las imágenes Ubuntu oficiales de Hardkernel, me habría sentido muy perdido.

¿Cómo usan tus ODROIDs?

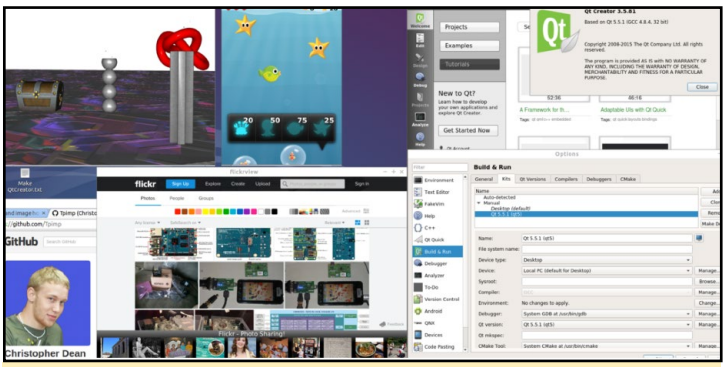
Principalmente utilizo mis ODROIDs para aprender. Con los años, he aprendido mucho sobre Linux en ARM. ¡Ahora uso ODROIDs siempre que puedo! Los pongo en robots, prototipos y cualquier cosa relacionada con el IoT que necesita USB, gran cantidad de datos o intensa carga de trabajo GPU, que está en creciente demanda. Puedo ahorrar un montón de tiempo con mis clientes usando placas de desarrollo ODROID y el ODROID-VU7, convertiendo sus ideas de productos en construcciones tangibles. Utilizando ODROID y QT5 (QML) puedo crear prototipos en semanas en lugar de meses.

¿Cuál es tu ODROID favorito?

El ODROID-U2 es mi favorito porque fue mi primera



Protoze es la emrpesa de Chris, y este es su logotipo.



Escritorio de Qt Developer's Dream

placa. Sin embargo, en cuanto a funcionalidad, el C1 y C2 son muy potentes. ¡Posiblemente con el tiempo el C0 se convierta en mi nuevo favorito, ¡ofrece mucho por muy poco!

Tu imagen SO precompilada Qt5 Developer's Dream es muy famosa. ¿Qué te gusta de QT5 y que te motivó a crear la imagen?

Sin duda, lo mejor de Qt es el soporte multiplataforma. En mi opinión, C++ es, por naturaleza, un lenguaje rápido, multiplataforma y extremadamente potente. Los desarrolladores del proyecto Qt han creado un lenguaje incluso mejor. He trabajado en varios proyectos con Qt como única dependencia, y no creo que exista una plataforma con la que sea más rápido desarrollar prototipos, cuenta con muchas menos dependencias que otras plataformas. Las motivaciones que hay detrás del Qt5 Developer's Dream es ayudar a otros a sacarle el máximo partido a sus ODROIDs. Sé de primera mano que convertirse en un desarrollador QT5 es fácil y divertido, pero desarrollar una gran plataforma de software como QT5 puede resultar muy complicado para muchos principiantes. Una imagen con librerías pre-compiladas directas de la fuente del proyecto Qt permite a los desarrolladores probar la última Qt 5.6 beta hasta el momento, sobre las placas de desarrollo ODROID en tres sencillos pasos:

- **Descarga la imagen Qt5 Developer's Dream**
- **Graba la imagen en la tarjeta SD o eMMC**
- **Con Poco código, puedes crear Mucho con QT5**

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

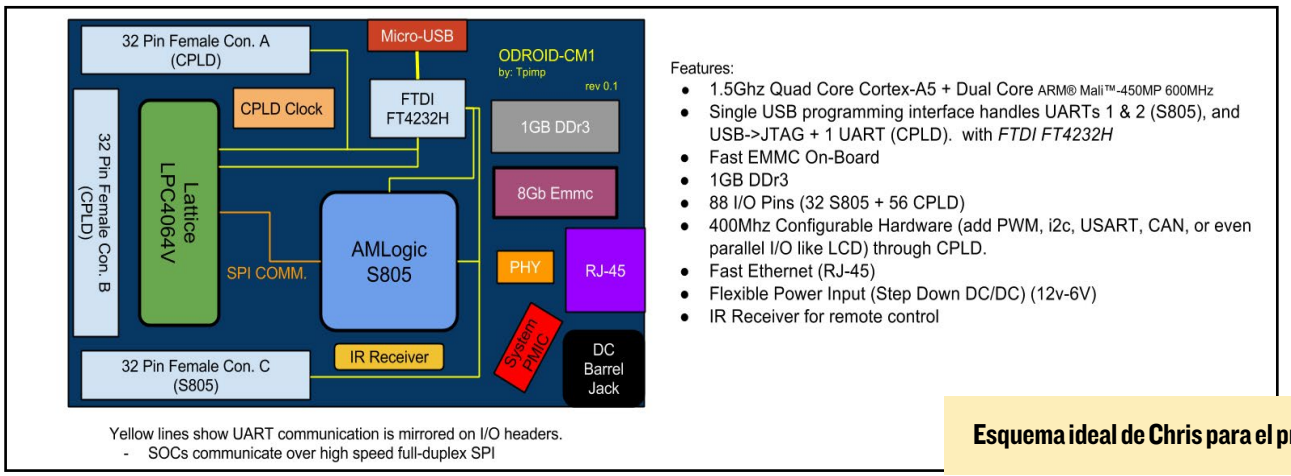
Me gustaría ver más circuitos integrados y circuitos periféricos para poder ampliar la capacidad del ODROID y así interactuar con más widgets de hardware. Actualmente cada ODROID proporciona un conjunto de chips sorprendentes para entornos Linux embebido, pero creo que la plataforma ODROID sería aún más valiosa con más periféricos en hardware, como SPI, PWM, GPIO de alta velocidad. Pienso que cualquier problema IoT que existe actualmente puede resolverse utilizando un ODROID conectado a un microcontrolador o un controlador lógico programable. Personalmente he estudiado la posibilidad de poner la unidad de microcontrolador (MCU) o un Field Programmable Gate Array (FPGA) sobre la misma placa como SOC Linux embebido. Esto es más fácil decirlo que hacerlo, requiere de mucha investigación y desarrollo

¿Qué aficiones e intereses tienes aparte de los ordenadores?

Me encanta montar en monopatín, los videojuegos, los proyectos de código abierto y pasar tiempo con mi familia.

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

Cualquiera puede escribir código, pero no todo el mundo está hecho para desarrollar software. Lo que sea que hagas, no aprendas código porque te haga falta. El desarrollo de software es como el arte, te ayuda si realmente pones el corazón en ello. Si quieres aprender, decide qué tipo de software deseas escribir. Si quieres crear una aplicación, un sitio web, un algoritmo de asignación o un virus, entonces, ¡Haz los deberes primero! Conoce el lenguaje/capa de tu desarrollo, luego, aprende a programar ese lenguaje. Hay muchos recursos online y grandes libros escritos para casi todos los lenguajes de desarrollo. Es importante que entiendas que el hecho de aprender a programar es el paso inicial para escribir software "malo". Tienes mucho que aprender antes de escribir software del "bueno". Estar abierto a la crítica puede perjudicarte, pero en última instancia te ayuda a crecer.



Esquema ideal de Chris para el próximo ODROID