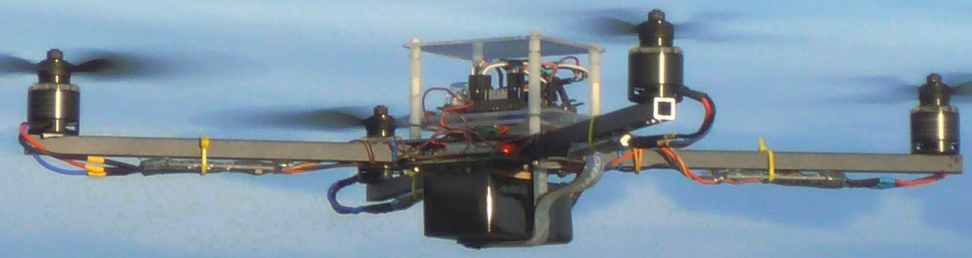
KitKat 4.4.2 en el ODROID-C1 • Juegos Android • Clúster Portatil

ODROID Magazine

Año Dos
Num. #15
Mar 2015

La era del Vuelo



Construye y vuela tu propio QuadCopter con piloto automatico usando el software Navio+

Trucos Docker:
Consigue más de tus contenedores

- Mandos de la Xbox 360 en Android
- Osciloscopio Técnico con ODROID-C1
- Des. Android: Descomponer y Modificar APK
- Cómo Alimentar tu ODROID-C1 por USB

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

Alguna vez has querido volar uno de esos curiosos drones, pero te parecían demasiado caros... Nuestro artículo de portada que muestra cómo coger el **ODROID-C1** y transformarlo en una impresionante máquina de vuelo, usando piezas fáciles de conseguir en tiendas de electrónica on-line. Incluye incluso una cámara, a la que se puede acceder a través de una tablet estándar, y un mando de **PS3** para controlar el vuelo. También puede utilizar Navio+, un popular software de piloto automático para ayudar al QuadCopter a desenvolverse en una ubicación específica.

Este número también ofrece un montón de juegos, Tobias nos muestran cómo jugar a algunos juegos de **NDS** muy singulares, que utilizan curiosas interfaces de usuario para resolver los puzzles, junto con varias análisis de juegos Android. Nuestra serie sobre Docker continúa con algunos consejos sobre su uso en Ubuntu 14.04, y Nanik nos muestra cómo descomponer y reconstruir los APK de Android. Venkat también presenta su proyecto sobre cómo usar el **ODROID-C1** como un osciloscopio, y mostramos un clúster **ODROID** portátil que incluye algunas imágenes pre-compiladas que te pueden ayudar a empezar rápidamente con tus propios proyectos de Informática de alto rendimiento.

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.
Únete a la comunidad ODROID con miembros en más de 135 países en http://forum.odroid.com/ y explora las nuevas tecnologías que te ofrece Hardkernel en http://www.hardkernel.com/.

HARDKERNEL

ODROID Magazine

Rob Roy, Editor Jefe

Soy un programador informático que vive y trabaja en San Francisco, CA, en el diseño y desarrollo de aplicaciones web para clients locales sobre mi cluster de ODROID. Mis principales lenguajes son jQuery, angular JS y HTML5/CSS3. También desarrollo sistemas operativos precompilados, Kernels persona-lizados y aplicaciones optimizadas para la plataforma 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 de SO en `http://bit.ly/1fsaXQs`.

Bo Lechnowsky, Editor

Soy el presidente de Respectech, Inc., Consultoría tecnológica en Ukiah, CA, EE.UU. que fundé en 2001. Con mi experiencia en electrónica y programación dirijo a un equipo de expertos, además de desarrollar soluciones personalizadas a empresas, desde pequeños negocios a compañías internacionales. Los ODROIDs son una de las herramientas de las que dispongo para hacer frente a estos proyectos. Mis lenguajes favoritos son Rebol y Red, ambos se ejecutan en los sistemas ARM como el ODROID-U3. En cuanto a aficiones, si necesitas alguna, yo estaría encantado de ofrecerte alguna de la mías ya que tengo demasiadas. Eso ayudaría a que tuviese más tiempo para estar con mi maravillosa esposa y mis cuatro hijos estupendos.

Bruno Doiche, Editor Artístico Senior

Está recibiendo clases de japonés, pero hasta ahora sólo ha aprendido a comer un libro, beber un periódico, nadar mientras cocina y un puñado de insultos, para disgusto de su profesor. También tiene un puñado de juegos de Android que no aparecen en esta edición de la revista (Sin embargo el crossy road triunfa!)

Nicole Scott, Editor Artistico

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 en http://www.nicolecscott.com.

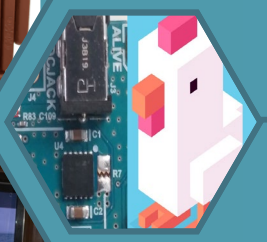
James Editor Artístico

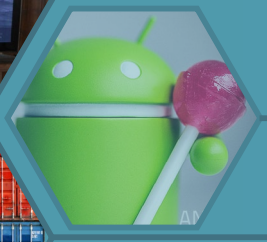
Soy un especialista en medios digitales que disfruta trabajando como freelance en marketing de redes sociales y administración de sitios web. Cuanto más aprendo sobre las posibilidades de ODROID más me ilusiona probar cosas nuevas con él. Me traslade a San Diego desde el Medio Oeste de los EE.UU. Todavía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

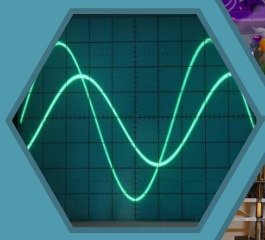
Manuel Adamuz, Editor Español

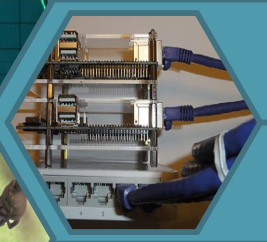
Tengo 31 años y vivo en Sevilla, España, y nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y las tecnologías de la información: ISO 9001, ISO 27001, 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.

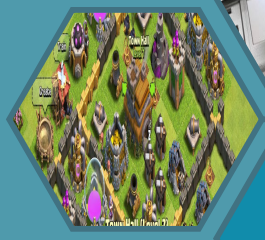
INDICE











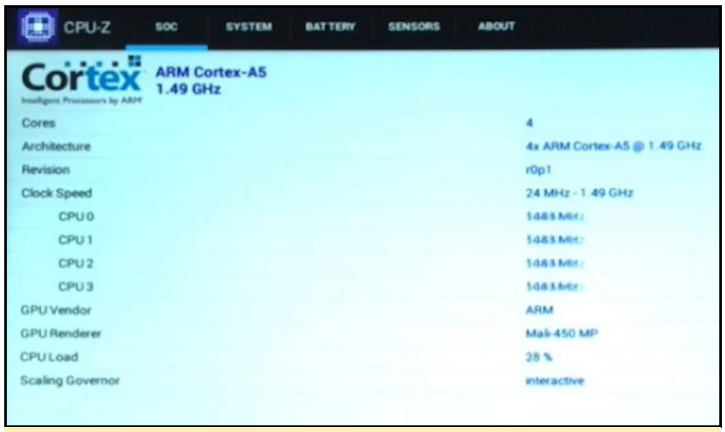
KITKAT 4.4.2

ODROID-C1 A EXAMEN

por Jose Cerrejon Glez

¿Quieres un Android KitKat estable para usarlo como ordenador principal? ¡No busque más!

En este análisis del ODROID-C1, vamos a ver si éste puede ejecutar Android con múltiples aplicaciones, juegos y emuladores mejor que otras placas de la competencia con precios similares .En primer lugar he de mencionar que estoy usando KitKat Android 4.4.2, versión 1.1. La instalación la cual ocupa 219MB, está realizada sobre una tarjeta SD UHS-I de 8GB SanDisk, comprada en la tienda Hardkernel. Todo lo que he leído sobre Android con una tarjeta SD es que es muy lento debido al acceso constante de entrada/salida del sistema operativo, por lo que es preferible usar un módulo eMMC. Sin embargo, he descubierto que es posible utilizar una microSD y además funciona muy bien, como cabe de esperar en cualquier tablet. Estoy seguro que en un módulo eMMC el sistema operativo se ejecuta más rápido, aunque una tarjeta SD es suficiente para conseguir buenos resultados.



CPU-Z mostrando las especificaciones del ODROID-C1

Una vez insertado y arrancado, el sistema redimensionará el tamaño de las particiones disponibles, incluyendo una partición vFAT que actúa como una tarjeta SD externa, en la que podemos copiar archivos auxiliares a través de un PC conectado vía USB. Otro aspecto positivo es que con una tarjeta de 8GB no nos vamos a quedar sin espacio, porque el propio sistema operativo se encargará de instalar las aplicaciones en otra partición si llenamos la partición base de sistema. Realice mis pruebas con resoluciones de 1280x800 y1920x1080.

Las aplicaciones que he instalado en la imagen de Android para hacer pruebas

Aplicaciones preinstaladas

Las apps que acompañan la imagen base de Hardkernel son muy útiles:

ODROID Utility: Lo más importante de esta herramienta es que te permite configurar la resolución de la pantalla para adaptarla a tu pantalla en concreto.

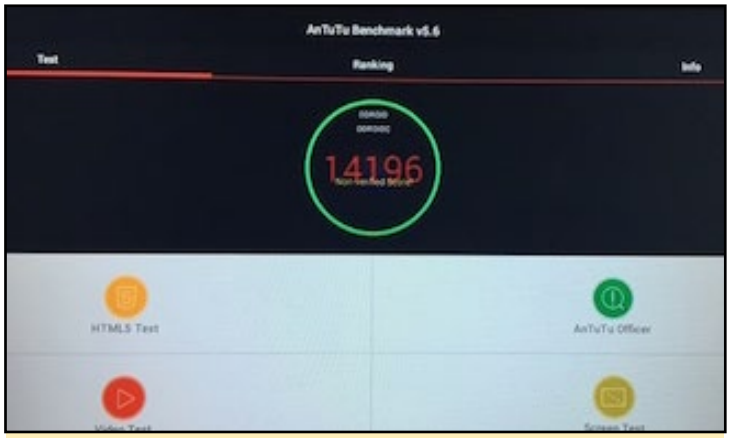
DicePlayer: Un sencillo reproductor de vídeo que reproduce casi cualquier tipo de archivo multimedia sin problema, a excepción de los vídeos 4K. También he probado vídeos con Kodi y el ODROID-C1 es el que mejor funciona de todas las placas que he probado (Banana Pi y Raspberry Pi), la tasa de fotogramas no desciende como en las otras placas.

Aplicaciones recomendadas

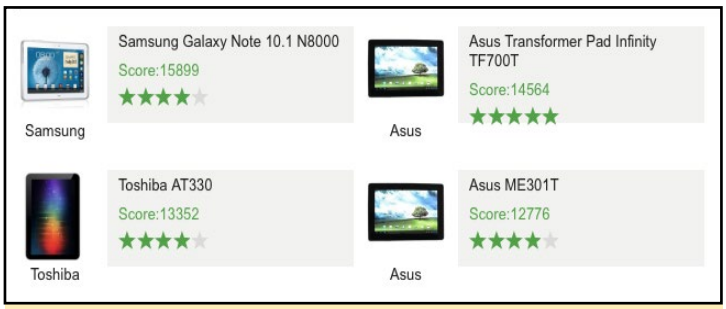
Google Play: La tienda no está incluida por defecto, pero es muy fácil de instalar descargando el archivo instalador desde `http://bit.ly/1wHG45b`.

SuperSU: El sistema operativo viene rooteado, aunque es necesario instalar la aplicación SuperSU desde Google Play para conceder permisos a los programas que acceden a determinados archivos, incluyendo Kodi. Sin aplicaciones como SuperSU, Kodi no puede utilizar la decodificación por hardware, así que esta aplicación es indispensable.

Games: La única limitación que he encontrado está en los juegos 3D - no por su rendimiento, sino porque han sido desarrollados para usarse con pantallas táctiles y el ratón no funciona. Por ejemplo, no puedo presionar un botón para empezar a jugar con el



Prueba de rendimiento AnTuTu en ODROID-C1: 14.196 puntos



Pruebas de rendimiento AnTuTu de otros dispositivos similares al C1

ratón. Tal vez haya alguna manera, pero yo no he encontrado la forma de que funcione correctamente.

RetroArch: He probado SNES, Genesis y MAME, y todos ellos se ejecutan sin problemas.

PPSSPP: A excepción de los juegos que utilizan elevados gráficos en 3D como God of War, PPSSPP funciona muy bien. Un juego especialmente bueno es Kingdom Hearts, que funciona y se ve bastante bien a pesar de sus exigentes gráficos 3D.

Mupen64+ se ejecuta a toda velocidad en un ODROID-C1

Spectaculator (emuladro Spectrum ZX): funciona perfeto.

Mupen64 + (Nintendo 64): Mario 64 va casi perfecto, tanto en sonido como FPS. Hay algunos problemas técnicos con algu-

Kodi reproduciendo un video demo a 1080p

PPSSPP y el juego Kingdom Hearts ejecutándose en un C1

nas sombras y texturas, pero apenas son apreciables.

Control Pad: probé el mando de la Xbox 360 conectado por cable con todos los emuladores y funciona perfectamente.

Navegador Web: He probado el navegador por defecto, además del Chrome. La navegación es fluida y sin ralentizaciones. Youtube móvil a través del navegador pierde calidad de imagen, pero si se utiliza la aplicación nativa de Youtube el rendimiento mejora bastante. Aunque Flash no funciona, ¿quién lo necesita?

Youtube: He instalado Youtube porque es algo "obligatorio" cuando mis amigos vienen a casa, por supuesto funciona como es debido. Además, los videos se cargan muy rápidamente.

Karaoke: No he probado a conectar un micrófono USB, pero instalando la aplicación Karoake desde Google Play puedes reproducir cualquier archivo .kar. Es increíble que algo tan simple no funcione de ningún modo con la Raspberry Pi.

Conclusión

Puedo decir que Android funciona como debe, pudiéndose usarse como sistema operativo principal. Es una pena que el problema con los juegos 3D sólo sea el hecho de que soporten pantallas táctiles, aunque es comprensible. Estoy muy contento con la combinación ODROID C-1 y Android, puedo conectarlo a la TV de mi sala de estar y utilizarlo como centro de entretenimiento. He publicado un video en `http://bit.ly/18hxfqy` para que puedas ver cómo se ejecutan algunas de las aplicaciones. Para preguntas y más información, por favor visita el post original en `http://misapuntesde.com/post.php?id=511`.

ALIMENTAR EL ODROID-CI USANDO EL PUERTO MICROUSB

PODER USAR EL CARGADOR DEL MOVIL GRACIAS A UNA SIMPLE SOLDADURA

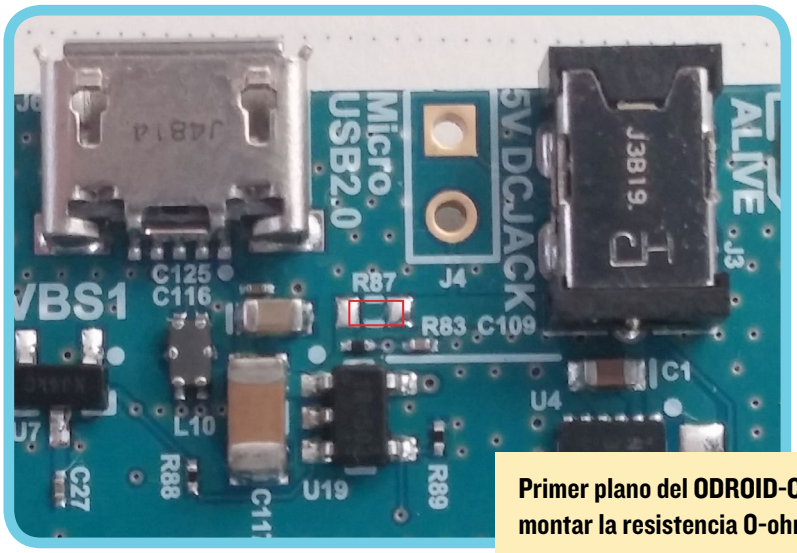
editato por Rob Roy

Una inquietud muy común entre los nuevos usuarios que cuentan con un ODROID-C1, especialmente los que están migrando desde la plataforma Raspberry Pi, es que el dispositivo sólo puede ser alimentado a través de la clavija DC. Por otro lado, mucha gente piensa que el microUSB puede proporcionar energía suficiente, sin tener que usar la fuente de alimentación oficial. Tienden a utilizar fuentes de alimentación de terceros inferiores, lo que provoca ciertos problemas cuando se trabaja con una alta demanda de CPU.

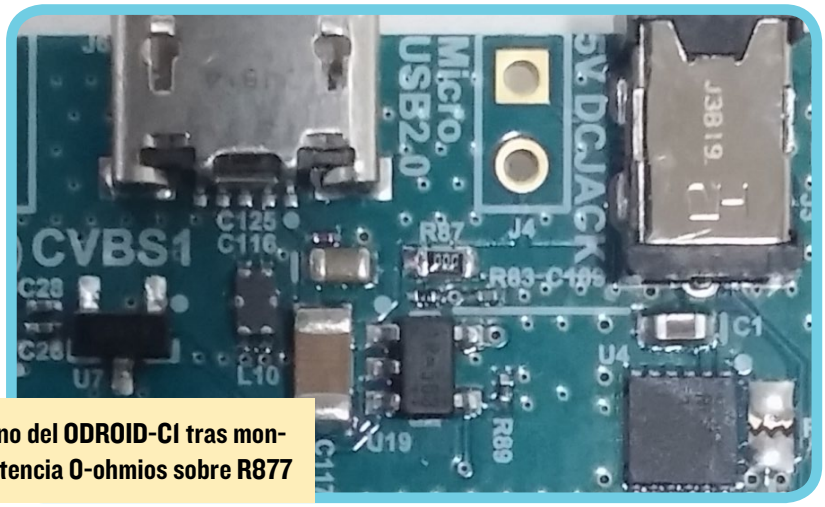
Sin embargo, el ODROID-C1 puede ser alimentado perfectamente por el puerto microUSB, que simplemente no está activado por defecto. Una simple unión en el hardware permite esta opción, la cual se detalla en las siguientes imágenes.

Para realizar la modificación, puede optar por uno de los siguientes métodos:

- Unir las dos yemas R87 (recuadro rojo) mediante una soldadura.
- Montar una resistencia 0-ohmios sobre R87 (tipo 1608).



Primer plano del ODROID-CI antes de montar la resistencia 0-ohmios sobre R87

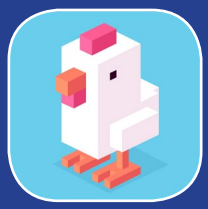


Primer plano del ODROID-CI tras montar la resistencia 0-ohmios sobre R877

CROSSY ROAD

MITAD CLASICO, MITAD MODERNO, ABSOLUTA DIVERSION

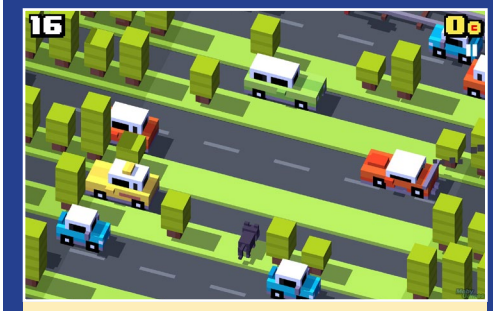
por Bruno Doiche



Los que tenemos una cierta edad podemos recordar fácilmente como a principios de los 80 estaba de moda jugar a freeway sobre la Atari 2600, en el que un pollo tenía que cruzar la carretera. Sin depender de ningún emulador (no es que no nos guste), es un juego increíblemente sencillo y divertido que consiste en ayudar a un gracioso pollo poligonal a cruzar carreteras, ríos, líneas de ferrocarril, evitando multitud de obstáculos. ¡Imprescindible para cualquier ODROID!

https://play.google.com/store/apps/details?id=com.yodo1.crossyroad&hl=en

Un simple juego de ataque y choque, fácil de aprender y muy adictivo.



Con un montón de cosas para desbloquear como recompensa por tus logros, te preguntarás donde fueron las últimas 2 horas.

PC PARA VER CINE EN CASA

COMPROBAMOS SI ODROID-C1 ESTA A LA ALTURA

por Douglas Roberts

En julio de 2013, monté un par de proyectos divertidos con Raspberry Pi: • Un servidor NFS y Minidlna (http://bit.ly/18HojM9), • un sistema de entretenimiento en casa con XBMC (http://bit.ly/1ASuZl4)

El pasado mes, compre un par de unidades de ODROID-C1, me pareció muy interesante porque por los mismos 35$ que cuesta una Pi, consigo un SBC ligeramente más pequeño y con una potencia aproximadamente 6 veces mayor. La siguiente es una breve comparación:

CPU Pi: ARM 700MHz vs C1: Quad-core ARM 1.5GHz
GPU Pi: 24 GFLOPS vs C1: 54 GFLOPS
USB Pi: 2 puertos USB 2.0 vs C1: 4 puertos USB 2.0
Ethernet Pi: 100MB/s vs C1: Gigabit

Pi y C1 consumen aproximadamente 3-4 vatios en ralentí.

Quería ver que tal funcionaban los C1como sustitutos de las unidades Pi. Estaba deseando probar el mayor ancho de banda que proporciona el ethernet Gigabit del C1, porque me he dado cuenta que el ancho de banda de 100 Mbps de la Pi era una limitación al intentar transmitir video Matroska a 1080p, no era suficiente ancho de banda para hacer streaming correctamente.

Poner en marcha el sistema Ubuntu 14.04 LTS proporcionado por los desarrolladores de ODROID fue facilísimo. Simplemente me descargué la imagen, la descomprimí y utilicé el comando "dd" para escribirla en una tarjeta MicroSD clase 10. Luego inserté la tarjeta en la ranura correspondiente del C1 y arranque el sistema. Las instrucciones para completar la instalación de Ubuntu son muy simples.

Como suelo hacer cuando jugueteo con un SBC, instalé tightvncserver para habilitar el acceso remoto. Los desarrolladores ODROID han hecho un buen trabajo con su distribución de Ubuntu y el gestor de ventanas LXDE que viene instalado por defecto es perfecto.

LXDE es muy rápido y se adapta al ODROID-C1, funciona perfectamente

Instalar el software del servidor NFS sólo me llevó un minuto y la instalación de Minidlna 1.1.4 simplemente era cuestión de descargar la fuente desde http://bit.ly/1FQ5SyK y compilarla.

Una vez hecho esto, el C1 reemplazo mi servidor NFS/Minidlna Pi. No aparecieron problemas, simplemente funcionó. Ahora da servicio a aproximadamente 8 TB de archivos multimedia desde varios discos externos USB.

Tras reemplazar el servidor, centré mi atención en reemplazar mi unidad Pi XBMC por el otro C1. Al igual que antes, la instalación del sistema de Ubuntu 14.04 fue rápida y sencilla y XBMC (ahora llamado "Kodi") viene pre-instalado en las imágenes ODROID-C1. Todo lo que se necesitaba para ponerlo en marcha era configurarlo para montar automáticamente los medios que eran compartidos por el servidor de archivos.

ODROID-C1 con switch Gigabit

Como antes, parecía que iba a ser una sustitución directa y que "simplemente funcionaria". Las Películas Blu Ray Matroska 1080p se reproducían sin problemas. Sin embargo, tuve algunos problemas con algunas de mis viejas películas en MPEG-2, en las que aparecían cortes, como una película de animación con plastilina. El Raspberry Pi reproducia la película sin problemas, por lo que decidí usar la herramienta Gnome MPlayer en el C1 y el archivo se reproducía bien, sin los cortes que se apreciaban con Kodi.

Para obtener más información, por favor visita el post original en http://bit.ly/1Az3ms9.

ANDROID 5.0 LOLLIPOP

LA SIGUIENTE GENERACION DE ANDROID DE GOOGLE

editado por Rob Roy

Android version 5.0 Lollipop, todavía no ha sido lanzado oficialmente para smartphones y tablet, pero puedes conseguir un avance para tu ODROID-C1 y realizar pruebas. Para descargar la versión oficial de Android para una resolución de 1080p visita `http://bit.ly/1DSDs7M`,y para una resolución de 720p visita `http://bit.ly/1AG5crp`.

Para probar tus nuevas aplicaciones con Lollipop, es necesario usar Android Debug Bridge (ADB) para trasladarlas través de un cable USB. En primer lugar, conecta el puerto microUSB del C1 al puerto USB de su equipo de desarrollo. Luego, activa el modo de depuración ejecutando la aplicación Settings en el C1 y entrando en la opción "About", a continuación haz clic varias veces en el área "Build Number".

El módulo 3 WiFi de Hardkernel es la único dongle que reconoce actualmente. El mando a distancia por infrarrojos de Hardkernel también funciona. Si deseas compilar la imagen por ti mismo, sigue las instrucciones de `http://bit.ly/1wHzuPC`.

Para obtener más información, hacer preguntas o sugerencias, visita el post original en `http://bit.ly/1B5Ysqh`. También puede ver un vídeo de Lollipop en acción en `http://bit.ly/18jgco1`.

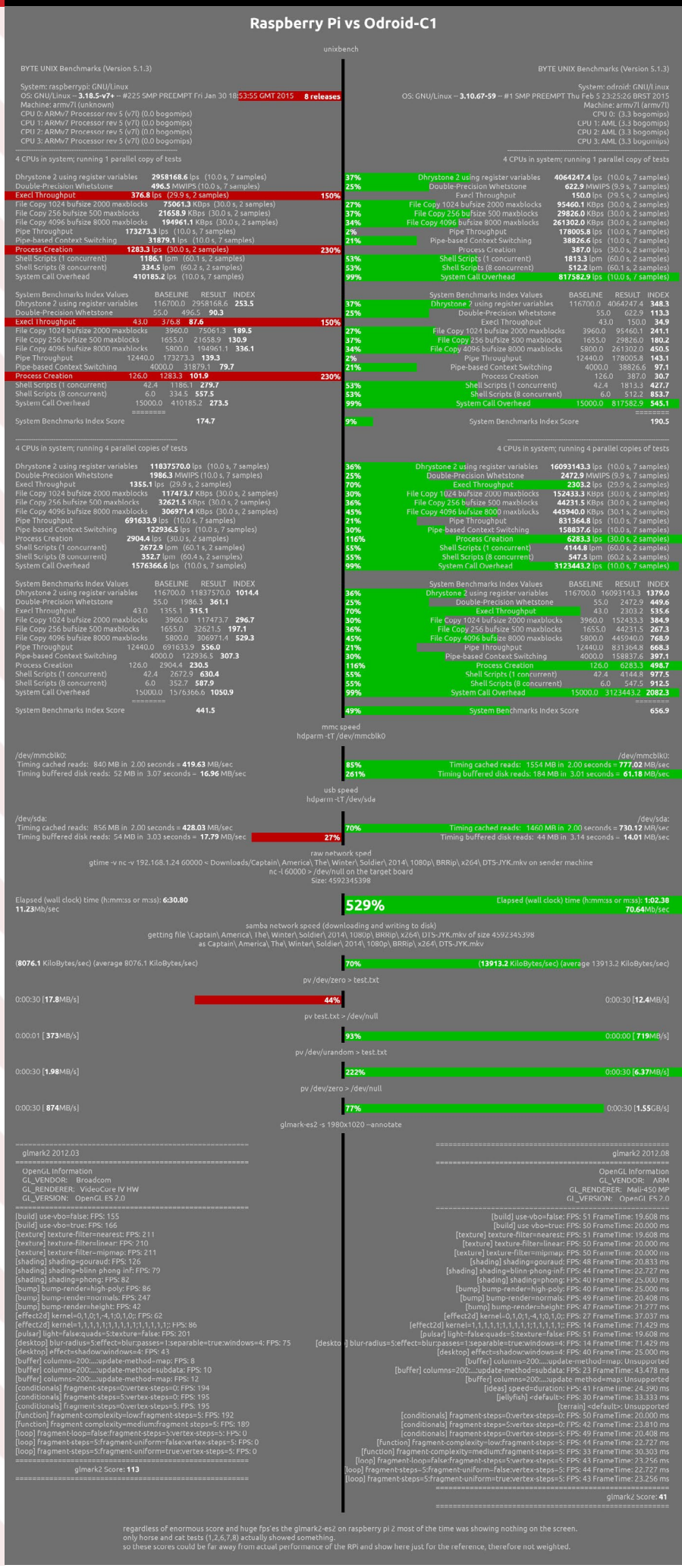
Si prefiere utilizar la versión de CyanogenMod de Android, puede que te interese CM12 Lollipop en http://bit.ly/1w1okVb, disponible para los modelos XU3y XU3 Lite. Sus características incluyen:

Kernel 3.10.9 • OpenGL ES 1.1/2.0/3.0 (Aceleración GPU)• OpenCL 1.1 EP (Aceleración GPU) • Función multiusuario (hasta 8 usuarios) • Ethernet Gigabit USB 3.0 yEthernet en placa • Soporte para dongle USB Ralink Wireless, RTL8188CUS y RTL8191SU • Compatibilidad con exFAT, NTFS(r/w), EXT4• Soporta Bluetooth USB• Soporte para dispositivos 3G USB• Soporte para dongle GPS USB • función moden USB • hotspot Wi-Fi portátil

Para añadir soporte wifi Ralink, escribe en el intérprete de comandos de depuración de Android:

```
add ro.hardkernel.wifi2=true to /system/build.prop
```

A medida que continúe el desarrollo, las actualizaciones estaran disponible `http://bit.ly/1GilbEi`.



CLICK PARA VER MAS

DOCKER EN EL ODROID

CONSEJOS PRACTICOS

por Uli Middelberg

En este artículo se presentan varios consejos útiles para ejecutar Docker en dispositivos ARMv7. No pretendo competir con tutoriales ya existentes, puesto que todos incluyen un gran trabajo. Quiero compartir algunas de mis experiencias con la instalación y puesta en funcionamiento de Docker en un ODROID. No esperes ejecutar todos los ejemplos que se mencionan en este tutorial en tu ODROID, puesto que están hechos expresamente para la arquitectura x86. Sin embargo, deberías ser capaz de ejecutar ciertos ejemplos con algunas pequeñas modificaciones.

Antes de empezar

Uso Ubuntu14.04.1 en mis dispositivos ARMv7, lo que hace más sencillo instalar y ejecutar Docker. Si optas por ejecutar un Linux diferente, es posible que observes pequeñas desviaciones como los nombres de los paquetes. Creé mi espacio de usuario de Ubuntu partiendo de cero y lo utilizo con mis diferentes dispositivos ARMv7. Exportar Ubuntu a un dispositivo diferente significa reutilizar el espacio de usuario, luego compilar y reemplazar el gestor de arranque y el núcleo específicos del proveedor. Por comodidad, regularmente subo mis imágenes a mi cuenta de Dropbox `http://bit.ly/1KoTYCC`.

No debes tener miedo a compilar un kernel Linux personalizado, puesto que la mayoría de los Kernel Linux específicos de proveedores no incluyen soporte para el sistema de ficheros aufs. Aunque Docker se ejecutará en kernels sin aufs, funcionará mejor en plataformas con aufs activado. Además, algunos proveedores pueden no incluir todas las funciones necesarias para que Docker se ejecute correctamente con la configuración del kenel por defecto.

Instalar Docker

Ubuntu 14.04.1 incluye un paquete docker.io (que en realidad es la versión 1.0.1), que se puede instalar desde una ventana de Terminal:

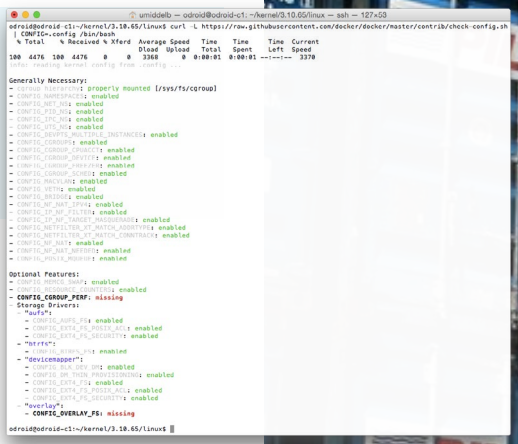
```
$ sudo apt-get install lxc aufs-tools cgroup-lite \
apparmor docker.io
```

Este comando instalará el binario y arrancará los scripts. Pero es muy probable que Docker no se inicie correctamente tras la instalación. Por lo general, hay una función específica del kernel que no está disponible y que Docker utiliza.

Ver lo que falta

Si ejecutas "/usr/bin/docker -D -d", te dará una pista sobre el motivo por el que el demonio Docker se niega a iniciarse. Los desarrolladores de Docker ofrecen un script bash que comprueba si todos los requisitos para Docker se cumplen con una determinara configuración del Kernel. El script check-config.sh (http://bit.ly/1DUaYIh) comprobará la configuración del kenel en ejecución, a la cual se accede via /proc/config.gz, o intentará leer el archivo de configuración de la fuente del kernel especificado sobre la línea de comandos. Usa el siguiente comando para ejecutar el script:

```
$ curl -L https://raw.\
githubusercontent.com/docker/\
docker/master/contrib/\
check-config.sh | /bin/bash \
/dev/stdin /path/to/.config
```



Compilar el kernel

Muchos proveedores publican kernels modificados para sus dispositivos ARM junto con instrucciones sobre cómo instalar la imagen del kernel compilada. Antes de empezar con la compilación, debemos instalar herramientas (make y gcc), así como algunas herramientas de terceros (bc, lzop, u-boot-tools). Ten cuidado que esto sobreescribirá la imagen del kernel por defecto. Haz en primer lugar una backup del kernel. El comando "make menuconfig" te proporciona una interfaz mucho más amigable que tener que editar manualmente la configuración con un editor de texto.

ODROID-C1

El último kernel para ODROID-C1 integra Docker y soporte para aufs, así que no hay necesidad de modificar ninguna de las opciones del kernel mediante make menuconfig.

```
$ git clone https://github.com/hardkernel/linux \
--single-branch -b odroidc-3.10.y
$ cd linux
$ make odroidc_defconfig
$ make clean
$ make -j 4 uImage modules
$ make dtbs
$ sudo make modules_install
$ sudo cp arch/arm/boot/uImage /media/boot
$ sudo cp arch/arm/boot/dts/meson8b_odroidc.dtb /
media/boot
```

ODROID-U3

El código fuente del kernel ODROID para ODROID-U3 ya tiene integrado el soporte para aufs.

```
$ git clone --depth 1 \
https://github.com/hardkernel/linux.git \
-b odroid-3.8.y
$ cd linux
$ make odroidu_defconfig && make menuconfig
$ make clean
$ make -j4
$ sudo make modules_install
$ sudo cp arch/arm/boot/zImage /media/boot
```

ODROID-XU3

```
$ git clone --depth 1 \
https://github.com/hardkernel/linux.git \
-b odroidxu3-3.10.y
$ cd linux
$ make odroidxu3_defconfig && make menuconfig
$ make clean
$ make -j4
$ sudo make modules_install
$ sudo cp arch/arm/boot/zImage arch/arm/boot/dts/\
exynos5422-odroidxu3.dtb /media/boot
```

Integración AUFS

Como he mencionado anteriormente, Docker incrementa su velocidad significativamente si el kernel soporta el sistema de ficheros aufs. Hasta ahora he usado una versión independiente (solo módulo kernel)

```
$ cd <kernel source directory>
$ git clone git://git.code.sf.net/p/aufs/\
aufs3-standalone aufs3-standalone.git
$ cd aufs3-standalone.git
$ git checkout origin/aufs3.10   # 3.10 ..  3.10.25
$ git checkout origin/aufs3.10.x # 3.10.26 and above
$ git checkout origin/aufs3.14   # 3.14
$ rm include/uapi/linux/Kbuild   # this will keep
your kernel sources config management from being
damaged
$ cp -rp *.patch fs include Documentation ../
$ cd ..
$ cat aufs3-kbuild.patch aufs3-base.patch \
aufs3-mmap.patch aufs3-standalone.patch | patch -p1
```

La numeración de la versiones de aufs corresponde a la versión del kernel, por lo que debes dirigirte a origin/aufs 3.14 para el código fuente del kernel Linux 3.14.x. El aufs 3,10 viene con

dos divisiones: 3.10 y 3.10.x. Desafortunadamente, los desarrolladores de aufs decidieron interrumpir el soporte para kernel inferiores al 3,14 desde principios del 2015.

```
$ make oldconfig
...
  Aufs (Advanced multi layered unification filesystem)
support (AUFS_FS) [N/m/y/?] (NEW) m
    Maximum number of branches
    > 1.  127 (AUFS_BRANCH_MAX_127) (NEW)
      2.  511 (AUFS_BRANCH_MAX_511) (NEW)
      3.  1023 (AUFS_BRANCH_MAX_1023) (NEW)
      4.  32767 (AUFS_BRANCH_MAX_32767) (NEW)
    choice[1-4?]: 1
    Detect direct branch access (bypassing aufs)
(AUFS_HNOTIFY) [N/y/?] (NEW) y
      method
      > 1.  fsnotify (AUFS_HFSNOTIFY) (NEW)
      choice[1]: 1
    NFS-exportable aufs (AUFS_EXPORT) [N/y/?] (NEW) y
    support for XATTR/EA (including Security Labels)
(AUFS_XATTR) [N/y/?] (NEW) y
    File-based Hierarchical Storage Management (AUFS_
FHSM) [N/y/?] (NEW) y
    Readdir in userspace (AUFS_RDU) [N/y/?] (NEW) y
    Show whiteouts (AUFS_SHWH) [N/y/?] (NEW) y
    Ramfs (initramfs/rootfs) as an aufs branch (AUFS_
BR_RAMFS) [N/y/?] (NEW) y
    Fuse fs as an aufs branch (AUFS_BR_FUSE) [N/y/?]
(NEW) y
    Hfsplus as an aufs branch (AUFS_BR_HFSPLUS)
[Y/n/?] (NEW) y
    Debug aufs (AUFS_DEBUG) [N/y/?] (NEW) n

$ configuration written to .config
```

Esto agregará los elementos de configuración del kernel relacionados con aufs a tu configuración existente. Asegúrate de elegir “m” para soporte aufs.

OverlayFS

Docker también soporta OverlayFS, que fue incluido con el kernel de linux 3.18. Si has logrado ejecutar Linux 3.18 en tus dispositvos arm, OverlayFS puede reemplazar a aufs.

Probar Docker

Ahora es el momento de volver a compilar e instalar el nuevo kernel. Si todo ha ido bien, el servicio Docker se ejecutará en tu dispositivo y escuchará las peticiones.

```
$ sudo docker info
Containers: 0
Images: 0
```

```
Storage Driver: aufs
 Root Dir: /var/lib/docker/aufs
 Backing Filesystem: extfs
 Dirs: 0
Execution Driver: native-0.2
Kernel Version: 3.10.66-aufs
Operating System: Ubuntu 14.04.1 LTS
CPUs: 4
Total Memory: 983.4 MiB
Name: odroid-c1
ID: 324D:YXY2:2XQP:CATB:KIQD:AFXA:UZBQ:IEPO:WSB5:3Y2R:
O5QU:FRDU
```

Elegir la imagen

La mayoría de las imágenes Docker están hechas para plataformas x86. Docker en sí mismo no depende de la plataforma, aunque las imágenes Docker contienen un registro de la arquitectura en la que se han creado:

```
$ sudo docker images
REPOSITORY                TAG                 IMAGE ID
CREATED             VIRTUAL SIZE
<none>                    <none>
d8115ff9b785        22 hours ago        301.5 MB
armv7/armhf-ubuntu_core   14.04               c11f-
1521cacf        2 weeks ago         159 MB
$ sudo docker inspect d8115ff9b785 | \
jq '.[] | .Architecture'
"arm"
```

Si ejecutamos un comando usando una imagen x86 en un dispositivo ARMv7, aparecerá el siguiente error:

```
$ sudo docker run ubuntu /bin/echo 'Hello world'
FATA[0205] Error response from daemon: Cannot start
container 9b55520a44ad4c069cc577afa51983713afb8e96ebe-
55a736e0819706b94f10b: exec format error
```

La mayoría de las imágenes Docker para dispositivos ARMv7 tienen en el registro de Docker un nombre que empieza por "armhf-" Puedes buscarlas con el siguiente comando:

```
$ sudo docker search armhf-
NAME                          DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
mazzolino/armhf-debian        Debian Wheezy base image for armhf devices      4
mazzolino/armhf-ubuntu        Ubuntu-Core images for armhf (ARMv7) devices    4
armv7/armhf-archlinux         archlinux arm docker image for the ARMv7(a...   2
armbuild/ubuntu-debootstrap   ARMHF port of ubuntu-debootstrap                1                      [OK]
armv7/armhf-ubuntu_core       ubuntu core docker images for the ARMv7(ar...   1
hominidae/armhf-ubuntu        ubuntu trusty/14.04 image (minbase) for ar...   1
dehy/armhf-couchdb            ARMHF port of klaemo/couchdb                    0
mazzolino/armhf-tiddlywiki    Tiddywiki5 on NodeJS for armhf (ARMv7) dev...   0
```

```
hominidae/armhf-wheezy        armhf image of Debian Wheezy, made with de...   0
dpniel/dekko-armhf            armhf utopic image to build dekko click pa...   0
hominidae/armhf-supervisord   ubuntu trusty/14.04 for armhf architecure ...   0
chanwit/fedora-armhf          Fedora for the armhf architecture               0
mazzolino/armhf-twister       Twister for armhf / armv7 devices               0
jalessio/armhf-ubuntu         Cloned from mazzolino/armhf-ubuntu              0
hominidae/armhf-archlinux     ArchLinux base image for armhf architectur...   0
pshouse/armhf-guacamole       armhf-ubuntu version of hall/guacamole          0
hominidae/armhf-cupsd         armhf image Wheezy, with sshd and cups pri...   0
onlinelabs/armhf-ubuntu                                                       0
mazzolino/armhf-butterfly     Butterfly for armhf (ARMv7) devices             0
moul/armhf-busybox                                                            0                    [OK]
armv7/armhf-ubuntu            'official' ubuntu docker images for the AR...   0
armv7/armhf-baseimage         ubuntu docker images for the ARMv7(armhf) ...   0
zsoltm/ubuntu-armhf           Ubuntu 14.04.1 minimal install, latest upd...   0
armv7/armhf-fedora            minimal fedora docker images for the ARMv7...   0
mazzolino/armhf-prosody       Secured Prosody XMPP server for armhf (ARM...   0
```

Publico mis imágenes Docker utilizando el perfil ARMv7 en Docker hub. Así que, vamos a probar el mismo comando usando la imagen armv7/armhf-ubuntu_core:

```
$ sudo docker run armv7/armhf-ubuntu_core /bin/echo \
'Hello world'
Unable to find image 'armv7/armhf-ubuntu_core:latest'
locally
Pulling repository armv7/armhf-ubuntu_core
c3802ac1b0ad: Download complete
Status: Downloaded newer image for armv7/armhf-ubuntu_
core:latest
Hello world
```

En esta ocasión, he utilizado un dispositivo diferente en el que no se ha descargado la imagen antes. Tras la descarga, formará parte de la memoria caché de la imagen local:

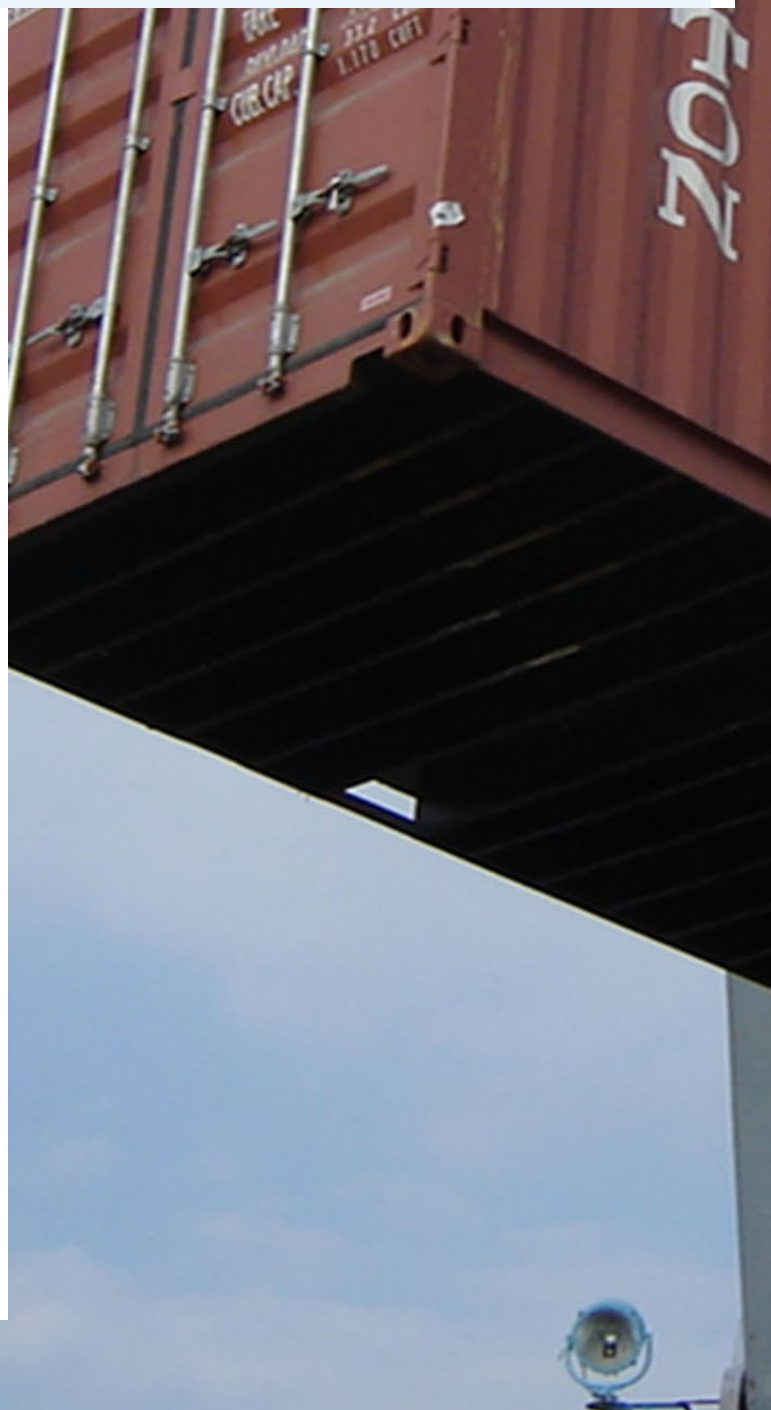
```
$ sudo docker images
REPOSITORY                TAG                 IMAGE ID
CREATED             VIRTUAL SIZE
armv7/armhf-ubuntu_core   latest
c3802ac1b0ad        About an hour ago   163.5 MB
```

Hay muchos y excelentes recursos que te pueden ayudar a comprender mejor el funcionamiento de Docker: `http://bit.ly/1MgEBtz`, `http://bit.ly/1FnLGUY` y `http://bit.ly/1A5PpTQ`.

Actualizar desde la versión 1.0.1

Puede que hayas experimentado un error en Docker 1.0.1, que de vez en cuando impiden que se inicien los contenedores:

```
$ sudo docker run armv7/armhf-ubuntu_core /bin/echo \
```



```
 'Hello world'
2015/01/15 17:57:10 finalize namespace drop
capabilities operation not permitted
```

Este error parece haber sido solucionado en la versión 1.4.0 de Docker. Puedes compilar Docker 1.4.0 desde la fuente con Docker 1.0.1, pero necesitaras las fuentes parcheadas de `http://bit.ly/1BR3mJL`, de lo contrario la fusión resultante impedirá que Docker se inicie.

```
FATA[0000] The Docker runtime currently only supports
amd64 (not arm).  This will change in the future.
Aborting.
```

La wiki Docker en `http://bit.ly/1NseXDB` te puede guiar a la hora de compilar el binario Docker desde la fuente. Desde la versión 1.5.0 los desarrolladores Docker quitaron el "fusible" que necesita expresamente la plataforma AMD64 e integraron la mayoría de los parches, haciendo más seguro el Docker 32-bit. Ahora puedes compilar Docker para ARMv7 con las últimas fuentes originales. Lo único que todavía necesitas es un Dockerfile ligeramente modificado para la plataforma armhf /ARMv7.

```
$ git clone -b 'v1.5.0' --single-branch \
https://github.com/docker/docker.git
$ cd docker
$ curl -L https://github.com/umiddelb/armhf/\
raw/master/Dockerfile.armv7 > Dockerfile
$ make build
$ make binary
$ sudo service docker.io stop
$ sudo cp bundles/1.5.0/binary/docker-1.5.0 /usr/bin
$ (cd /usr/bin; sudo mv docker _docker; sudo ln -s
docker-1.5.0 docker)
$ sudo service docker.io start
```

Es muy probable que el error mencionado anteriormente interrumpa el proceso de compilación. En este caso, tiene que utilizar el comando "sudo make build" más de una vez. Otra posibilidad es que descargues el binario final de Docker desde `http://bit.ly/1aRZu0P`. El binario en sí esta enlazado de forma estática y puede ejecutarse en otras versiones de Linux:

```
$ file /usr/bin/docker-1.5.0
/usr/bin/docker-1.5.0: ELF 32-bit LSB  executable,
ARM, EABI5 version 1 (SYSV), statically linked, for
GNU/Linux 2.6.32, BuildID[sha1]=eef157201c4e1d888d0977
0a8187edf956605176, not stripped
```

Basta con sustituir el binario Docker existente en /usr/bin por el nuevo:

```
$ sudo docker version
[sudo] password for umiddelb:
Client version: 1.5.0
Client API version: 1.17
Go version (client): go1.4.1
Git commit (client): a8a31ef-dirty
OS/Arch (client): linux/arm
Server version: 1.5.0
Server API version: 1.17
Go version (server): go1.4.1
Git commit (server): a8a31ef-dirty
```

Instalar Docker en Fedora 21

Desafortunadamente, no hay ningún paquete rpm disponible para Docker sobre armhfp. Por lo tanto, el proceso de instalación es algo más largo y la mayoría de los pasos derivan del paquete fuente docker-io-1.4.1-7.fc22.src.rpm:

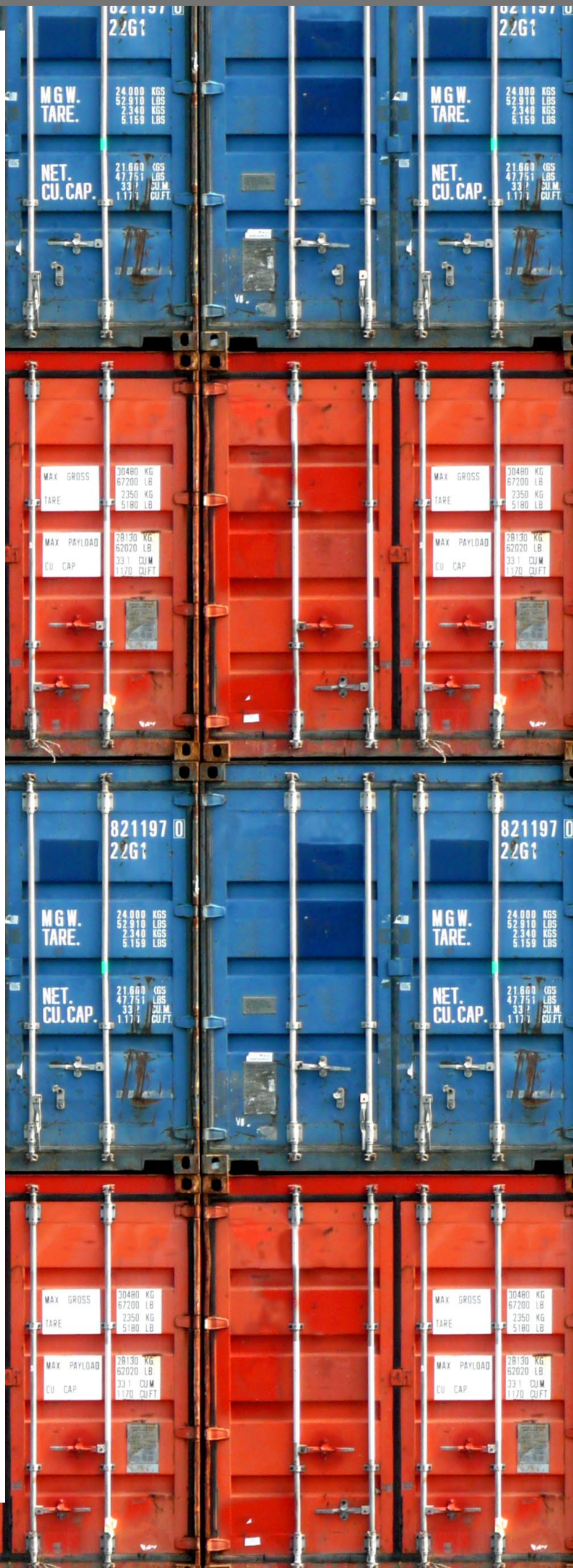
```
$ sudo yum install rpm-build
$ sudo yum install glibc-static
$ sudo rpmbuild --rebuild \
http://copr-be.cloud.fedoraproject.org/results/\
gipawu/kernel-aufs/fedora-21-x86_64/\
aufs-util-3.9-1.fc20/aufs-util-3.9-1.fc21.src.rpm
$ sudo rpm -i /root/rpmbuild/RPMS/armv7hl\
/aufs-util-3.9-1.fc21.armv7hl.rpm
$ sudo yum install lxc bridge-utils device-mapper \
device-mapper-libs libsqlite3x docker-registry \
docker-storage-setup
$ mkdir docker
$ cd docker
$ wget ftp://fr2.rpmfind.net/linux/fedora/linux/\
development/rawhide/source/SRPMS/d/\
docker-io-1.4.1-7.fc22.src.rpm
$ rpm2cpio docker-io-1.4.1-7.fc22.src.rpm | cpio -idmv
$ tar -xzf v1.4.1.tar.gz
$ curl -L https://github.com/umiddelb/armhf/raw/\
master/bin/docker-1.5.0 > docker
```

Este procedimiento de instalación procede de docker-io-1.4.1-7.fc22.src.rpm::docker-io.spec:

```
# install the docker binary
$ sudo install -p -m 755 docker /usr/bin/docker

# install bash completion
$ sudo install -p -m 644 docker-1.4.1/contrib/\
completion/bash/docker /usr/share/bash-completion/\
completions

# install container logrotate cron script
```



```
$ sudo install -p -m 755 docker-logrotate.sh \
/etc/cron.daily/docker-logrotate

# install vim syntax highlighting
$ sudo install -p -m 644 docker-1.4.1/contrib/syntax/\
vim/doc/dockerfile.txt /usr/share/vim/vimfiles/doc
$ sudo install -p -m 644 docker-1.4.1/contrib/syntax/\
vim/ftdetect/dockerfile.vim /usr/share/vim/vimfiles/\
ftdetect
$ sudo install -p -m 644 docker-1.4.1/contrib/syntax/\
vim/syntax/dockerfile.vim /usr/share/vim/vimfiles/syntax

# install udev rules
$ sudo install -p docker-1.4.1/contrib/udev/\
80-docker.rules /etc/udev/rules.d

# install systemd/init scripts
$ sudo install -p -m 644 docker.service \
/usr/lib/systemd/system

# for additional args
$ sudo install -p -m 644 docker.sysconfig \
/etc/sysconfig/docker
$ sudo install -p -m 644 docker-network.sysconfig \
/etc/sysconfig/docker-network
$ sudo install -p -m 644 docker-storage.sysconfig \
/etc/sysconfig/docker-storage

# install docker config directory
$ sudo install -dp /etc/docker

$ getent passwd dockerroot > /dev/null || sudo \
/usr/sbin/useradd -r -d /var/lib/docker -s \
/sbin/nologin -c "Docker User" dockerroot
$ sudo /bin/systemctl enable docker.service
```

Para obtener más información, realizar preguntas o comentarios, por favor visita el post original en `http://bit.ly/1zZxyxP`.

HAZ VOLAR UN ODROID

CONSIGUE QUE VUELE POR TI MISMO

por Gregory Dymarek

¡Vas a ser la envidia de tu barrio cuando todo el mundo vea volar esto!

Ultimamente los Drones están siendo cada vez más populares. Oímos hablar de ellos en los medios de comunicación porque son máquinas muy versátiles. Por un lado, el ejercito los llevan utilizando desde hace algún tiempo y Amazon está estudiando su uso para la entrega de paquetes. Por otro lado, son una gran herramienta como hobby, ya que la gente los usa para hacer fotografías aéreas. También puedes usarlo para participar en competiciones de QuadCopter y para otros fines como aficionado.

Los Quadcopters pueden ser comprados de serie, o ser construidos uno por ti mismo como he hecho yo. El coste de los Quadcopters ya construidos es similar al de un juguete para niños, pero los más profesionales tienen un precio próximo al de un coche pequeño. La base que hay detrás de ellos es la misma, pero la calidad y funcionalidad varían mucho.

Afortunadamente, los quadcopters son relativamente fáciles de construir y con algo de experiencia en bricolaje, puedes ahorrar mucho dinero. Para construir uno, necesitarás tener conocimientos básicos de soldadura o aprenderlos sobre la marcha. Con un poco de tiempo puedes programarlo a tu gusto.

Mi primer Quadcopter fue un Hubsan H107. Hacerlo volar era tan divertido que me hizo sumergirme directamente en el mundo de los quadcopters. Recomendaría este modelo en concreto a cualquiera que se inicie en este hobby. Es barato y resistente para que pongas en práctica tus habilidades de vuelo. Además, tiene un rendimiento muy bueno para su precio.

Después de haber pasado interminables horas leyendo sobre diferentes quadcopters, se hizo evidente que podría intentar construir uno yo mismo partiendo de cero. Así es como nació el proyecto AvrMiniCopter. Mi objetivo era crear un controlador para Quadcopter que se ejecutara en un sistema Linux. De esta forma, podría hacerlo extensible y reutilizar los drivers estándar de Linux. Además, la programación en un sistema Linux con todas las funciones es mucho más rápida y fácil que crear programas para placas integradas. El coste total de todos los componentes fue aproximadamente de 150$, incluye el ODROID, el mando PS3 y la tablet.

Componentes clave

- Estructura base Warthox (brzos de 25cm)
- 4 motores SunnySky X2212 KV980
- 4 ESCs Afro Slim 20A
- ODROID-W
- Arduino Pro Mini 16MHz
- Placa sensor MPU9150
- Placa sensor BMP180
- Dongle usb Bluetooth o WIFI
- Camara Raspberry Pi, o modelo linux h264 compatible
- Bateria Turnigy 2200mAh 3S 25C LiPo, monitor de batería y cargador

Historia del proyecto

En primer lugar recopile todos los componentes básicos que necesitabay luego, lo cableé todo. Por aquel entonces usa-ba una Raspberry Pi como única placa para controlarlo. El resultado fue satisfactorio, puesto que escribí mi propio código de controlador para la RPi, y al instante tuve volando un Quadcopter. Sin embargo, la estructura base y los motores elegidos no ajustaban muy bien y tuve algo de trabajo adicional para conseguir mayor firmeza. En ocasiones, me encontré con algunos problemas técnicos y retrasos en la ejecución, lo cual causaba que el Quadcopter chocara en ciertas ocasiones. Más tarde, me di cuenta que esto se debía a que la tarjeta SD

Vista de pajaro desde el QuadCopter mientras vuela

insertada en la RPI se salía de la ranura, lo que provocaba un "kernel panic". Además, se hizo evidente que el proyecto debía beneficiarse de un sistema en tiempo real. El Raspberry Pi no es un sistema en tiempo real y como tal, no puede garantizar una respuesta en un margen de tiempo preciso, que es necesario para que el vuelo sea suave y fiable. Intente hacer frente a esta cuestión utilizando el famoso entorno de trabajo en tiempo real Xenomai. Sin embargo, la cantidad de trabajo que me suponía, especialmente con la implementación de los driver, me hizo buscar otra solución.

En mi segundo ensayo, decidí utilizar una tarjeta controladora dedicada, para lo cual elegí Arduino Pro Mini junto con un ODROID-W. Arduino es una placa en tiempo real muy conocida y consolidada que puede dar estabilidad al Quadcopter, mientras que el ODROID-W se usaría para otros cálculos, menos sensibles al tiempo como lecturas barométricas, soporte para el mando PS3 y la captura de vídeo. Además, la inclusión del ODROID-W trajo consigo una reducción importante del tamaño físico de todo el sistema. El software del controlador para RPi que originalmente escribí fue exportado a la plataforma Arduino, y desarrollé una imagen para ODROID-W para la cual se configuró un entorno de desarrollo específico para agilizar el proceso. He comprobado que este método funciona perfectamente y no lo he modificado desde que lo implante.

Piezas

Para construir tu AvrMiniCopter necesitarás algunas piezas. En mi caso, he comprado la mayoría en HobbyKing y eBay. Hay muchas web y foros donde puedes encontrar gran cantidad de información sobre los tipos y tamaños de motores, los controles de velocidad electrónicos (ESCs) y las baterías. Algunos ofrecen una estimación de la duración de vuelo. Mi consejo es que no des tanta prioridad al tiempo de vuelo en tu primer Quadcopter, ya que su desarrollo será más complicado.

- Estructura base: Elige una que tendrá suficiente espacio para todo el equipo. Es más fácil construir un Quadcopter grande. Sin embargo, será más caro y más difícil de manejar.

- Motores sin escobillas: Hay demasiados tipos de motores a tener en cuenta. Elije los que mejor se ajusten a la estructura base y cuenten con una potencia más que suficiente. La regla de oro es localizar unos que en conjunto generan suficiente empuje como para levantar dos veces el peso de tu Quadcopter. Los Quadcopters más grandes incorporarán hélices más grandes, de modo que los motores KV más pequeños (revoluciones per-volt) serán aceptables, haciendo el Quadcopter más eficiente.

- Control electrónico de velocidad (ESC): Para el proyecto AvrMiniCopter, necesitarás ESCs que cogan la modulación por ancho de pulsos (PWM) como señal de entrada y que suministre potencia más que suficiente a los motores. Algunos ESCs tienen un Circuito de desconexión de batería (BEC) incorporado de modo que no tendrías que comprar uno por separado.

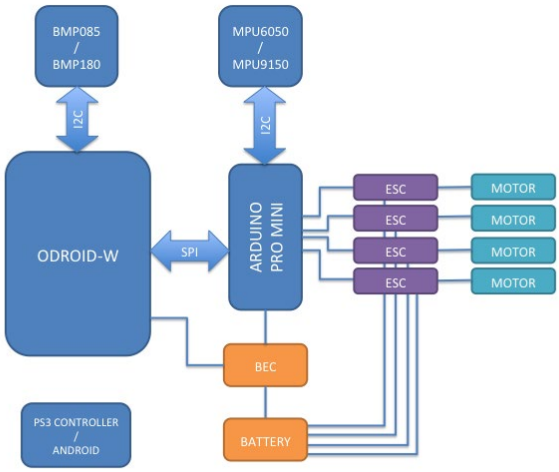


Diagrama por bloques del diseño de hardware del QuadCopter

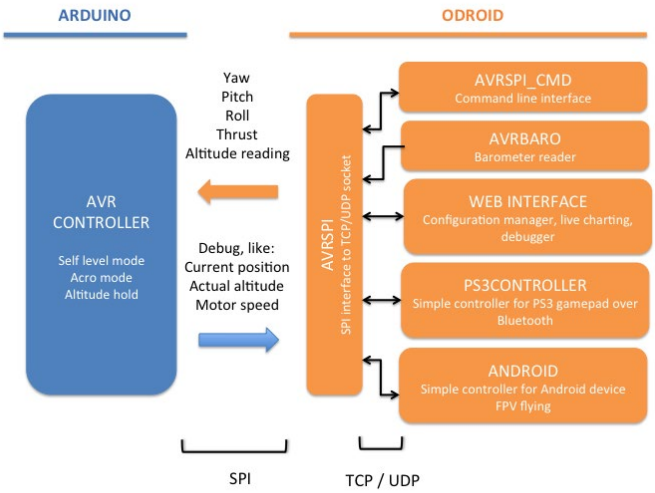
- Batería LiPo: La batería es uno de los componentes individuales más pesados de tu Quadcopter y afecta en gran medida al tiempo de vuelo. Asegúrate de tener una batería con la tensión y el índice de desconexión correctos para tus motores.

Software

El software AvrMiniCopter consta de dos partes: un controlador Arduino y las herramientas de gestión del host (ODROID). El código de Arduino es una implementación de 8 bits de un controlador de vuelo, su función central consiste en leer y calcular la posición actual del Quadcopter, manteniéndolo estable en todo momento. Por otro lado, la actividad principal del host es la de leer el mando controlador PS3 del piloto y enviar las solicitudes a Arduino para su ejecución en cuanto a desviación, inclinación, balanceo y empuje.

La implementación real está un poco más fragmentada y se divide en varios módulos como el componente AVRSPI que traslada la interfaz SPI hacia el socket TCP, la función AVR-BARO que lee datos del barómetro en bucle y pasa los datos a

Diagrama por bloques del diseño de software del QuadCopter



través de Arduino AVRSPI, el módulo AVRCONTROLLER que se encarga de las entradas del gamepad PS3, gestión de la configuración y los datos de registro de vuelo. También existe una interfaz web para la comunicación con el controlador y su configuración, muy útil para ajustar y depurar mientras lo ejecutamos al tiempo que usas un dispositivo móvil.

Todo el software que necesitas también viene en una versión pre-compilada que se puede descargar como una imagen y que puedes trasladar a una tarjeta SD, pudiendo empezar sin tener que compilar nada. La imagen es una distribución Linux minimalista hecha a medida, creada y mantenida utilizando un conjunto de scripts buildroot. Todos los script están disponibles en la página del proyecto de GitHub en http://bit.ly/1NingC0, y las imágenes pre-compiladas de Arduino se pueden descargar desde http://bit.ly/1EOKc8e.

Funcionalidad

Actualmente, el software AvrMiniCopter es un completo sistema capaz de controlar quadcopters de cualquier tamaño y con una configuración X. Ofrece dos modos de vuelo: el modo estabilizado, donde el QuadCopter se auto-nivela por sí sólo, y el modo gradual para un vuelo más ágil. Con un barómetro, el Quadcopter también es capaz de mantener la altitud deseada.

Para el control del vuelo, se puede usar un mando PS3 que funciona a una distancia de alrededor de 50 metros y se comunica vía bluetooth. Una solución Wi-Fi que está en desarrollo proporcionará mayor cobertura, así como la posibilidad de hacer streaming en directo desde un dispositivo móvil. Pronto, serás capaz de utilizar su smartphone o tablet para controlar el vuelo sin la necesidad de usar un mando, ahorrándote así la su compra . También estoy buscando un módulo GPS compatible que me permita hacer vuelos programados, pero en la actualidad esto está en la fase de planificación.

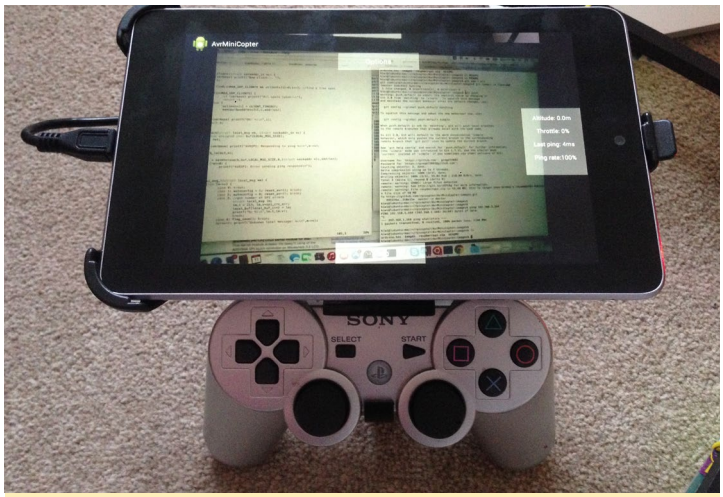
Notas

Te recomiendo construir o comprar una estructura rígida para tu primer prototipo, de un material resistente y ligero como el aluminio. He roto más de 10 estructuras de madera realizando pruebas y aprendiendo antes de pasar a un chasis de aluminio. Asegúrate de mantener tu Quadcopter alejado de cualquier cosa que se pueda romper y no instales las hélices durante las pruebas.

Ten en cuenta que los quadcopters son grandes juguetes para aprender a volar, como aficionados no debemos provocar más restricciones debido a un mal uso o falta de sentido común. Por ejemplo, nunca vueles tu Quadcopter cerca de un aeropuerto o de otras personas. Las hélices giran a gran velocidad y pueden causar lesiones con facilidad

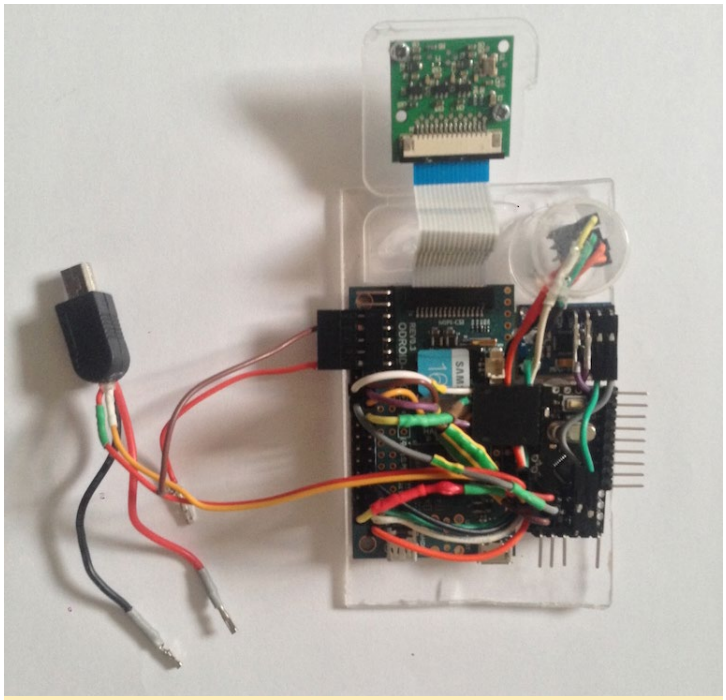
Hay muchos otros aspectos del vuelo del Quadcopter que no han sido tratados en este artículo. Sin embargo, espero realmente que esta información le sea de utilidad y haya conseguido llamar su atención. Por favor, si tienes alguna pregunta házmela saber a través de los foros ODROID en http: // forum.odroid.com. Para obtener más información sobre el proyecto AVR-MiniCopter, visita la wiki del proyecto en http://bit.ly/1DX3OWb. También puede ver un vídeo de la Quadcopter en acción en http://bit.ly/1w5gvhv.

Los componentes del QuadCopter son una tablet y un mando PS3

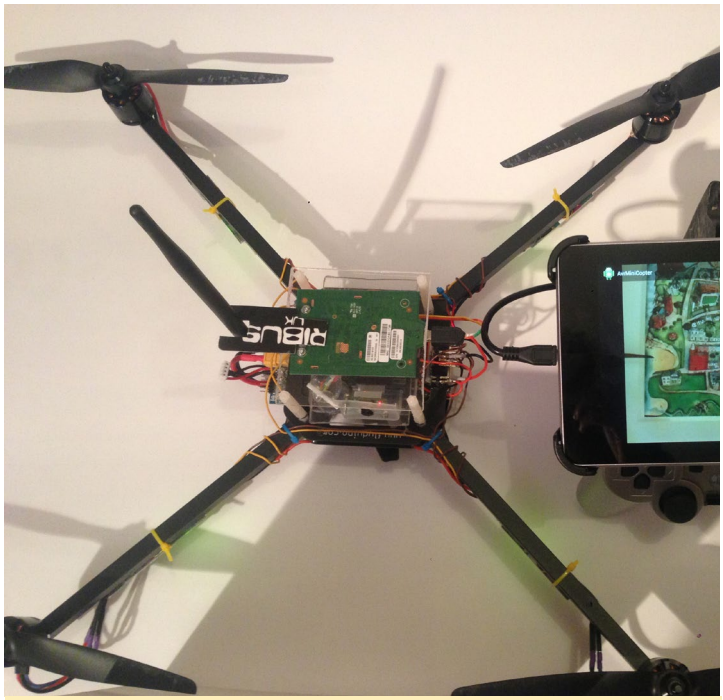


Tablet mostrando la vista de la cámara del QuadCopter

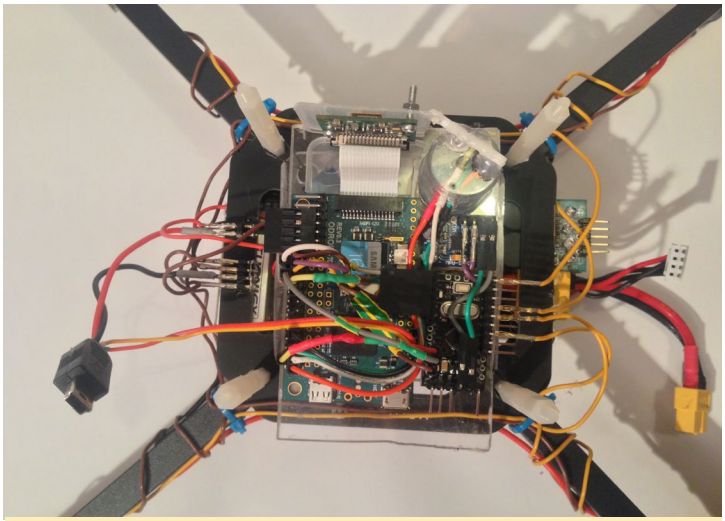
Primer plano del mando PS3 conectado a la tablet



Componentes ODROID-W usados en el Quadcopter



Carcasa protectora del QuadCopter que evita posibles daños



Primer plano de controlador QuadCopter montado con las hélices

QuadCopter completamente montado y listo para su primer vuelo

QuadCopter suspendido en un lugar concreto, sin fluctuar

QuadCopter volando sin problemas sobre los árboles

DESARROLLO ANDROID

DESCOMPONER Y MODIFICAR EL ARCHIVO APK

por Nanik Tolaram

En mi anterior artículo, hablamos sobre el funcionamiento interno del archivo Android Package Kit (APK), la forma en la que se estructura y las herramientas que puedes usar para generarlo. En este artículo, vamos a ver diferentes herramientas para analizar y realizar cambios en un archivo APK. Como ejemplo, vamos a descomponer la app de Android GitHub, que puedes descargar desde http://bit.ly/1Ecc0Tp.

Estructura básica

Antes de descomponer el archivo APK, vamos a echar un vistazo a los contenidos del archivo en la Figura 1.

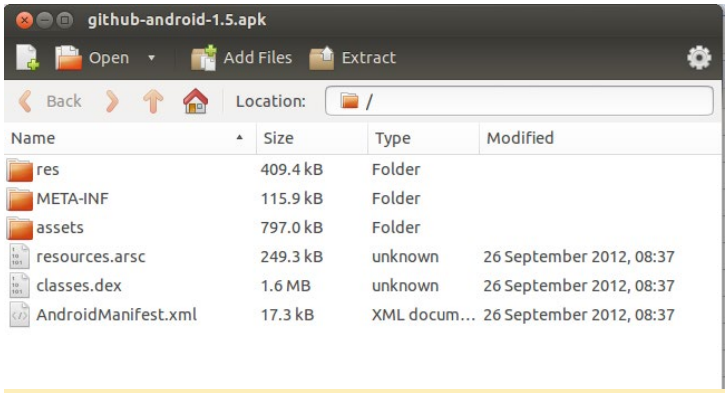


Figura I – Interior de github-android-I.5.APK

Como puedes ver, hay 3 carpetas principales: res/ contiene archivos de recursos (como el diseño y las cadenas), assets/ que normalmente alberga archivos grandes (como vídeos e imágenes) y META-INF/ que contienen archivos de información como la codificación hash de los archivos localizados en las carpetas res/ y assets/. El classes.dex es el archivo principal que contiene el código fuente de la aplicación, también es el archivo que lee y ejecuta la máquina virtual Dalvik de Android.

Desempaquetar

Ahora que hemos visto la estructura básica de los archivos APK, vamos a ver como los desempaquetamos. Una cosa que debes recordar es que no todo el interior del APK está en formato texto. Por ejemplo, si descomprimir el archivo APK y abres el archivo AndroidManifest.xml verás caracteres ilegibles, como se muestra la Figura 2.

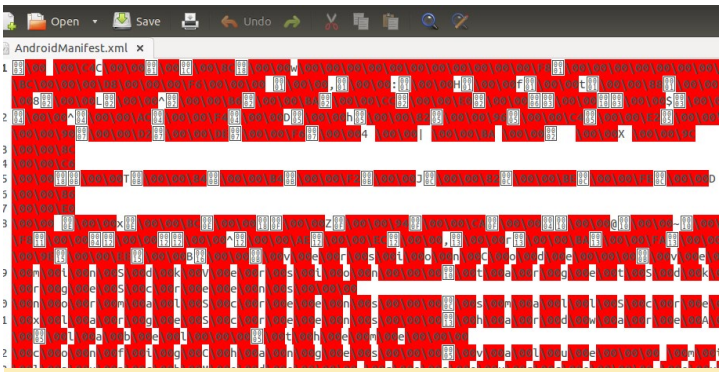


Figura 2 - Galimatías del AndroidManifest.xml

Si queremos ver el contenido del APK en su formato original, es necesario descomprimirlo usando una herramienta llamada APKtool que puedes descargar desde http://bit.ly/1FPDVHo/. En este artículo, vamos a utilizar la versión 1.5.2. Sigue los pasos descritos en la wiki http://bit.ly/1EHQCpy para instalarla en la máquina de desarrollo.

En primer lugar, asegúrate de que tienes tu directorio SDK de Android en tu ruta. Como ejemplo, aquí tienes el contenido de mi variable PATH de mi equipo:

/home/nanik/Downloads/android-sdk-linux/build-tools/20.0.0/:/home/nanik/bin:/usr/lib/lightdm/lightdm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/home/nanik

Ejecuta el siguiente script para descodificar el archivo APK:

```
#!/bin/sh
java -jar <APKtool_directory>/APKtool.jar d ./github-android-1.5.APK
```

Reemplazar <APKtool_directory> por la ubicación de tu directorio local que has descomprimido utilizando APKtool. Una vez que ejecutes el comando, verás el siguiente resultado:

```
I: Baksmaling...
I: Loading resource table...
I: Loaded.
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file:
/home/nanik/APKtool/framework/1.APK
I: Loaded.
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Done.
I: Copying assets and libs...
```

Debes tener muy en cuenta es esta parte del resultado::

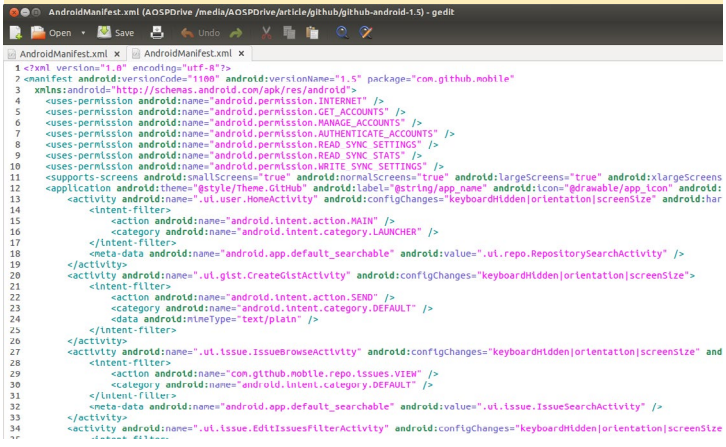
```
I: Loading resource table from file:
/home/nanik/APKtool/framework/1.APK
```

Esta línea es muy importante, ya que muestra el entorno de trabajo que Android está usando para descodificar el archivo. Si estas decodificando en Android 5.0, utilizarás un entorno APK diferente en comparación con Android 4.0 o 2.3. Una vez completado el paso de decodificación, observarás una nueva carpeta con el nombre del archivo APK y su contenido será como la Figura 3. Ten en cuenta que es la misma estructura que en la Figura 1. Si abres AndroidManifest.xml, veras algo similar a la Figura 4.

Figura 3 – Interior de ./github-android-1.5

```
drwxrwxr-x  6 nanik nanik  4096 Feb 17 21:36 ./
drwxrwxr-x  3 nanik nanik  4096 Feb 17 21:36 ../
-rw-rw-r--  1 nanik nanik 12129 Feb 16 22:15 AndroidManifest.xml
-rw-rw-r--  1 nanik nanik   259 Feb 16 22:15 apktool.yml
drwxrwxr-x  4 nanik nanik  4096 Feb 16 22:15 assets/
drwxrwxr-x  3 nanik nanik  4096 Feb 17 21:36 build/
drwxrwxr-x 47 nanik nanik  4096 Feb 16 22:15 res/
drwxrwxr-x  7 nanik nanik  4096 Feb 16 22:15 smali/
```

Figura 4 - AndroidManifest.xml descodificado



Realizar cambios

Vamos a intentar hacer algunos cambios simples en el archivo APK decodificado, luego empaquetarlo y utilizarlo en ODROID o en tu dispositivo Android. La pantalla original antes de la modificación se puede ver en las figuras 5 y 6.

Figura 5 - Pantalla de inicio de sesión original

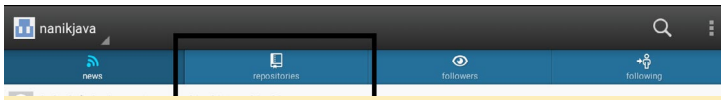


Figura 6 - Pantalla principal original

Vamos a hacer varios cambios en el texto reemplazando las palabras " Login or Email " y " Password ", y sustituir la palabra "repositories ". El código que es necesario cambiar se encuentran dentro de archivo res/values/strings.xml. Modifica la siguiente línea:

```
Antes: <string name="login_or_email">Enter or Email</
string>

Despues: <string name="login_or_email">Please enter
your login or email</string>
```

Luego, cambiar el texto de Password:

```
Antes: <string name="password">Password</string>

Despues: <string name="password">Enter your pass-
word</string>
```

Por último, cambiar la palabra repositories:

```
Antes: <string name="tab_repositories">repositories</
string>
Despues:<string name="tab_repositories">repo</string>
```

Empaquetar

Ahora que hemos completado los cambios, tenemos que empaquetar o codificar los archivos de nuevo en un APK, después firmarlo y usarlo en el dispositivo. Para empaquetar los archivos, utiliza el siguiente script:

```
#!/bin/sh
java -jar <APKtool_directory>/\
APKtool.jar b \
./github-android-1.5/ \
./github.APK
#the following command is to gen-
erate .keystore
#-------------------------------
---------------------------
#keytool -genkey -keystore
./<yourpersonal>.keystore \
-validity 10000 \
-alias <yourkeystorename>

#the following command is to
build and sign the .APK
<your_jdk_directory>/bin/\
jarsigner -keystore ./<yourper-
sonal>.keystore -verbose \
./github.APK <yourkeystorename>
```

La primera línea que ejecuta la herramienta APK es usada para codificar/compilar el archivo APK. Despues, necesitarás utilizar la herramienta jarsigner para firmar el APK con tu propia "keystore". Tras ejecutar la herramienta jarsigner, verás el siguiente resultado:

```
I: Checking whether sources has
changed...
I: Checking whether resources has
changed...
I: Building APK file...
Enter Passphrase for keystore:
   adding: META-INF/MANIFEST.MF
   adding: META-INF/NANIK.SF
   adding: META-INF/NANIK.DSA
  signing: assets/lib/util/\
loadmode.js
  signing: assets/lib/util/\
multiplex.js
  signing: assets/lib/\
codemirror.js
  signing: assets/mode/clike/\
clike.js
  signing: assets/mode/clike/\
index.html
  signing: assets/mode/clike/\
scala.html
  signing: assets/mode/clojure/\
clojure.js
  signing: assets/mode/clojure/\
index.html
  ..
  ..
  ..
  signing:
res/xml/sync_adapter.xml
  signing: AndroidManifest.xml
  signing: classes.dex
  signing: resources.arsc
      pkg: /data/local/tmp/
github.APK
```

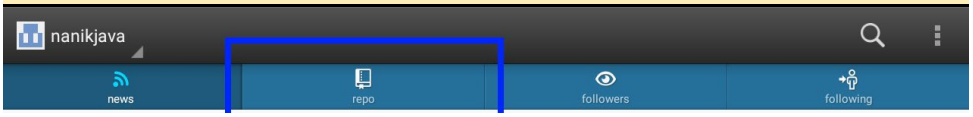
Finalmente, obtendrá un archivo llamado github.APK que se puede instalar con el siguiente comando:

```
$ adb install /path/to/app.apk
```

Una vez instalada la app, puedes ejecutarla para comprobar que el nuevo texto aparece como en las figuras 7 y 8:

Figura 7 - Pantalla de inicio de sesión modificada

Figura 8 - Pantalla principal modificada



ANGRY BIRDS TRANSFORMERS

UNA BUENA MEZCLA DE VIEJOS Y NUEVOS HEROES

por Jeremy Leesmann

Qué podemos obtener si mezclamos Transformers y Angry Birds ... AUTOBIRDS! Es un shooter de acción rápida con el que disfrutaras aniquilando a todos los granujas usando laser y dinamita. Funciona muy bien en el ODROID-U3.

https://play.google.com/store/apps/details?id=com.rovio.angrybirdstransformers

La pantalla principal muestra la isla en la que los granujas han establecido su base de operaciones, ¡Tienes que eliminarlos rápido!

Esta versión de Angry Birds es muy rápida, te mantiene en constante movimiento

Los Transformers siempre ganarán, incluso frente a los retorcidos granujas

OSCILOSCOPIO

USAR EL ODROID-CI COMO UN OSCILOSCOPIO TECNICO

por Venkat Bommakanti

La integración de pines de Entrada/Salida de Propósito General (GPIO) en dispositivos complejos basando en Sistemas de un único Chip (SoC) ha dado lugar a potentes y amigables plataformas de bajo coste como el ODROID-C1. El Kit C-Tinkering junto con la librería wiringPii para C1 ejectuando Lubuntu, cuenta con todos los elementos básicos de código abierto para iniciar el desarrollo de prototipos de circuitos sensoriales inteligentes.

Ahora, ¿Qué ocurre si a esa combinación le añadimos un osciloscopio open source y un analizador lógico con interfaz USB? Nos encontraremos con un completo y moldeable laboratorio de electrónica portátil, capaz de testear señales, realizar mediciones y gestionar y analizar datos. Es cuestión de tiempo que los fabricantes que interactúan con el mundo sensorial, se den cuenta de la necesidad de esta configuración.

Los osciloscopios y analizadores lógicos seleccionados para este artículo son meros ejemplos de los tipos de dispositivos de medición que se podrían usar, con la mirada puesta en el software de código abierto, la buena relación precio-rendimiento y un buen conjunto de funciones. Ten en cuenta que Hardkernel no recomienda ningún dispositivo específico. Se espera que sea el propio usuario el que realice la investigación apropiada para elegir el dispositivo de medición acorde a sus necesidades, teniendo en cuenta que estos productos pueden ser dispositivos que requieran ajustes posteriores. Ten presente que los precios y las posibilidades son muy variadas.

Requisitos

1. Un ODROID-C1. Aunque este artículo se centra en el C1, se puede aplicar a un U3/XU3/ XU3-Lite.
2. Accesorios para C1: cable HDMI, cable ethernet CAT 5E+ o dongle wifi 3, se recomienda PSU, batería RTC o ODROID-VU.
3. Un módulo eMMC 5.0 de 16GB con la última imagen de Lubuntu específica para C1 y/o una MicroSD de 16 GB+ Clase 10 con lector de tarjeta SD.
4. Mali OpenGL-ES SDK v2.4.4
5. Mono runtime 3.2.8
6. Una red en la que el dispositivo tenga acceso a internet y a los foros ODROID.
7. Acceso en red al C1 a través de utilidades como PuTTY, FileZilla, TightVNC Viewer (MS Windows 7+) o Terminal (Mac, Linux) desde un ordenador de pruebas.
8. Un Kit C-Tinkering
9. Un osciloscopio y analizador lógico como el DSLogic (DreamSource Lab) o SmartScope (LabNation), o un dimple analizador lógico como el BeagleLogic. Es bueno utilizar un dispositivo compatible con sigrok, con el fin de aprovechar todas las ventajas de la librería de análisis de señales de código abierto.

Instalar Lubuntu

Instala la imagen C1 más reciente en la eMMC e insértala en el C1. Con la pantalla VU conectada arrancar el sistema. Ejecuta ODROID Utility y configura la resolución de pantalla a 800p y reiniciar el sistema. Después, expande la partición de instalación para utilizar todo el módulo eMMC seleccionando la opción “Resize your root partition”.

Reinicia y vuelve a ejecutar ODROID Utility de nuevo, configura y actualiza el resto de aspectos del sistema, puede que necesites reiniciar nuevamente. Para las imágenes más recientes, tendrás que ejecutar los siguientes comandos en orden para actualizar el sistema:

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

Preparar el sistema

Instala los componentes de software necesarios con el siguiente comando:

```
$ sudo apt-get install \
git-core gcc g++ \
autoconf automake make \
cmake libtool \
pkg-config libglib2.0-dev \
libglib2.0 \
libzip-dev libudev-dev \
libasound2-dev \
libasound2 libusb-1.0 \
python3-dev python3 check \
libqt4-dev libboost-dev \
libboost-all-dev \
libboost-test-dev \
libboost-thread-dev \
libboost-system-dev \
mono-runtime
libmono-system-core4.0-cil \
libmono-system-drawing4.0-cil \
libmono-cairo4.0-cil \
libsdl-mixer1.2 \
libsdl1.2debian \
libmono-system-xml-linq4.0-cil \
libmono-system-windows-\
forms4.0-cil
```

Compilar DSLogic

Prepara un marcador de posición para recibir el último software DSLogic (Ver. 0.4). Crea un directorio usando los siguientes comandos en una ventana de Terminal:

```
$ cd ~ && mkdir dslogic && \
cd dslogic
```

Descarga el software DSLogic (DSLogic-v0.4.tar.gz) desde http://bit.ly/1Fo4Gmk y mueverlo al directorio creado anteriormente. Expande el tarball fuente con el comando:

```
$ tar xvzf DSLogic-v0.4.tar.gz
```

A continuación, compila libusbx:

```
$ cd DSLogic-v0.4/libusbx-1.0.18/
$ ./autogen.sh
$ ./configure
$ make
$ sudo make install
```

Compila la librería libsigrok4DSLogic, que es un plugin compatible con sigrok para el dispositivo analizador lógico/osciloscopio DSLogic proporciona la API básica de hardware DSLogic, utilizando los comandos:

```
$ cd ../libsigrok4DSLogic
$ ./autogen.sh
$ ./configure
     ...
     libsigrok configuration sum-
mary:

      - Package version (major.
minor.micro): 0.2.0
      - Library version
(current:revision:age): 1:2:0
      - Prefix: /usr/local
      - Building on: armv7l-un-
known-linux-gnueabihf
      - Building for: armv7l-un-
known-linux-gnueabihf

    Detected libraries:

      - glib-2.0 >= 2.32.0: yes
(2.40.2)
      - libzip >= 0.10: yes
(0.10.1)
      - libserialport >= 0.1.0:
no
      - libusb-1.0 >= 1.0.9: yes
(1.0.18)
      - libftdi >= 0.16: no
      - libudev >= 151: yes (204)
      - alsa >= 1.0: yes
(1.0.27.2)
      - check >= 0.9.4: yes
(0.9.10)

    Enabled hardware drivers:

      - demo.....................
.......  yes
      - DSLog-
ic.........................  yes

$ make
$ sudo make install
```

Compila la librería libsigrokdecode:

```
$ cd ../
$ git clone git://sigrok.org/\
libsigrokdecode
$ cd libsigrokdecode
$ ./autogen.sh
$ ./configure
    ...
    libsigrokdecode configuration
summary:

      - Package version (major.
minor.micro): 0.3.0
      - Library version
(current:revision:age): 2:0:0
      - Prefix: /usr/local
      - Building on: armv7l-un-
known-linux-gnueabihf
      - Building for: armv7l-un-
known-linux-gnueabihf

    Detected libraries:

      - (REQUIRED) python >= 3.2:
yes (3.4)
      - (REQUIRED) glib-2.0 >=
2.24.0: yes (2.40.2)
      - (OPTIONAL) check >=
0.9.4: yes (0.9.10)

    Enabled features:
      - (OPTIONAL) Library unit
test framework support: yes

$ make
$ sudo make install

    Installing 45 protocol decod-
ers:

    swd pan1321 tca6408a jtag_
stm32 jtag i2c i2cdemux midi pwm
    ir_nec rgb_led_spi usb_sig-
nalling sdcard_spi dcf77 uart
    mx25lxx05d i2s rfm12 ds1307
lm75 spdif am230x onewire_link
    usb_packet ir_rc5 nunchuk mx-
c6225xu can nrf24l01
    guess_bitrate edid onewire_
network avr_isp z80 xfp parallel
    jitter tlc5620 maxim_ds28ea00
eeprom24xx rtc8564 mlx90614
    i2cfilter lpc spi
```

Compila la aplicación DSLogic-gui utilizando los comandos:

```
$ cd ../DSLogic-gui/
$ .  export BOOST_LIBRARYDIR=\
"/usr/lib/arm-linux-gnueabihf/"
$ cmake .
$ make
$ sudo make install
```

Te puedes encontrar algunos errores de código, tales como:

```
...  no matching function for
call to 'min(double, qreal)'...
```

En ese caso, cambia la línea:

```
double delta = min((double)
max(pos - UpMargin, 0), \
get_view_rect().height());
```

por esta otra:

```
double delta = min((double)
max(pos - UpMargin, 0), \ (dou-
ble)get_view_rect().height());
```

Repite el proceso de compilación hasta que lo consigas, realiza los cambios que veas necesarios

Configurar el laboratorio de Electrónica

Consigue el kit C-Tinkering y configúralo usando las instrucciones de http://bit.ly/1NsrlU9. Prepara un directorio para recibir la fuente wiringPi y compílala:

```
$ cd ~ && mkdir tkit && cd tkit
$ git clone https://github.com/hardkernel/wiringPi
$ cd wiringPi
$ sudo ./build
```

Crea un directorio marcador para el código de ejemplo:

```
$ cd ~ && mkdir tkit-example/ && cd tkit-example/
```

Coge example-lcd.c de la wiki y colocalo en el nuevo directorio de trabajo, Después, crea una copia:

```
$ cp myexample-lcd myexample-lcd.c
```

El archivo original se ocupa de los numerosos LEDs, pero para el simple ejemplo de este artículo, es recomendable actualizar la lógica para gestionar un único LED. Aplica el siguiente parche a la copia del archivo (myExample-lcd.c):

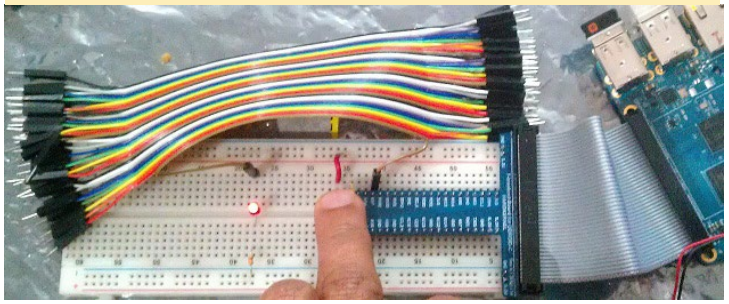
```
94a94,96
> #define OFF 0
> #define ON  1
> static unsigned long int ctr = 1;
97,108c100,123
<
<     // adc value read
<     if((adcValue = analogRead (PORT_ADC1)))       {
<             ledPos = (adcValue * MAX_LED_CNT * 1000)
/ 1024;
<             ledPos = (MAX_LED_CNT - (ledPos /
1000));
<     }
<     else
<             ledPos = 0;
<
<     //  LED Control
<     for(i = 0; i < MAX_LED_CNT; i++)      digi-
talWrite (ledPorts[i], 0); // LED All Clear
<     for(i = 0; i < ledPos;         i++)    digi-
talWrite (ledPorts[i], 1); // LED On
---
>     int tmp;
>
>     // adc value read
>     adcValue = analogRead (PORT_ADC1);
>     if(adcValue)    {
>             ledPos = (adcValue * MAX_LED_CNT *
1000) / 1024;
>     tmp = ledPos;
>             ledPos = (MAX_LED_CNT - (ledPos /
1000));
>             printf("%10lu: adc-value:%10d tmp:%3d
ledPos:%3d\n", ctr++, adcValue, tmp, ledPos);
>     }
>     else
>     {
>             ledPos = 0;
>             printf("%10lu: adc-value:%10d
ledPos:%3d\n", ctr++, adcValue, ledPos);
>     }
>
>     //  LED Control
>     for(i = 0; i < MAX_LED_CNT; i++)      digi-
talWrite (ledPorts[i], 0); // LED All Clear
>     if (adcValue < 15)
>     tmp=ON;
>     else
>     tmp=OFF;
>     for(i = 0; i < ledPos;         i++)    digi-
talWrite (ledPorts[i], tmp); // LED status depends on
light
>     usleep(10000);
141a157
>
```

En una nuevo Terminal, compila el ejemplo y ejecútalo:

```
$ cd ~/tkit-example/
$ gcc -o myexample-led myexample-led.c \
-lwiringPi -lwiringPiDev -lpthread
$ sudo ./myexample-led
```

El kit tinkering modificado (hardware/software) usará el pin GPIOX.BIT0 (marcado con el nº 97) como se ve en la Figura 1, respondiendo al bloqueo del sensor de luz:

Figura I: Instalación del kit Tinkering modificado



Tras terminar con la aplicación ejemplo, se puede decir que estás preparado para usar el osciloscopio o analizador lógico con el fin estudiar el comportamiento eléctrico de los pins (GPIO).

Probar la configuración

Usando tu manual de usuario y notas de la wiki, conecta el osciloscopio DSLogic al LED como se ven en la Figura 2.

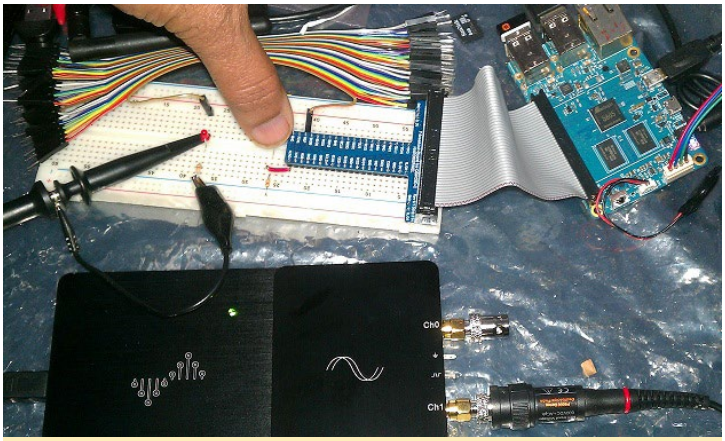


Figura 2: Instalación del osciloscopio DSLogic

En una nueva sesión de Terminal, inicia la aplicación gui del DSLogic mediante los siguientes comandos:

```
$ cd ~/dslogic/DSLogic-v0.4/DSLogic-gui/
$ sudo ./DSLogic
```

En otra sesión de Terminal, inicia la aplicación de ejemplo:

```
$ cd ~/tkit-example/
$ sudo ./myexample-led
```

Con el LED OFF/LOW (0 voltios), El punto de partida de la interfaz gráfica de osciloscopio debe ser como la figura 3.

Consulta el manual de usuario y la wiki para fijar la configuración adecuada. A continuación, mueve lentamente un dedo para bloquear del sensor. El LED debe encenderse. Con el LED ON/HIGH (3,3 voltios), la interfaz del osciloscopio ha de ser similar a la Figura 4.

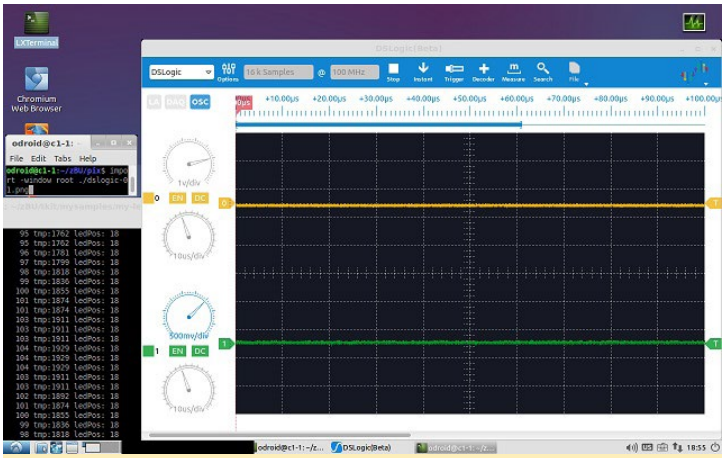


Figura 3: Punto de partida del osciloscopio DSLogic

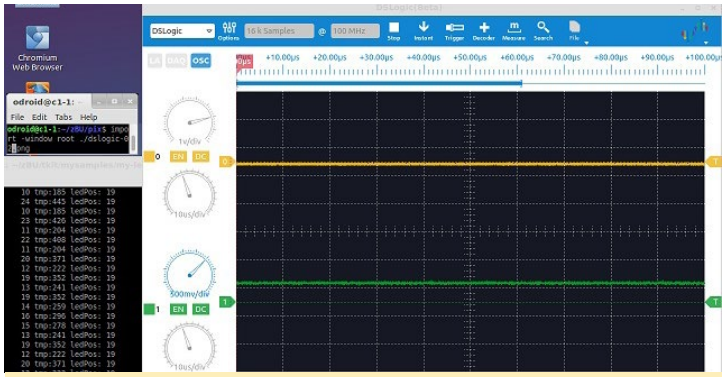


Figura 4: Osciloscopio DSLogic a 3.3 V

La línea del gráfico verde continua representa los 3,3 V y la línea del gráfico verde con puntos representa 0V.

Sal de la aplicación DSLogic-gui. Consulta de nuevo el manual de usuario y la wiki, configura el dispositivo DSLogic en modo analizador lógico. Debería verse como la Figura 5.

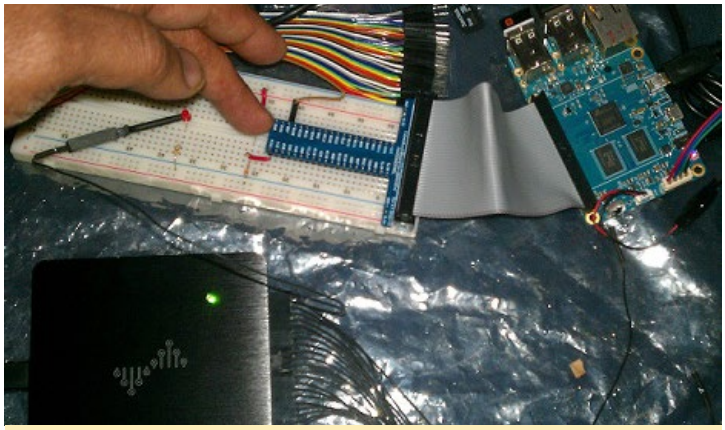
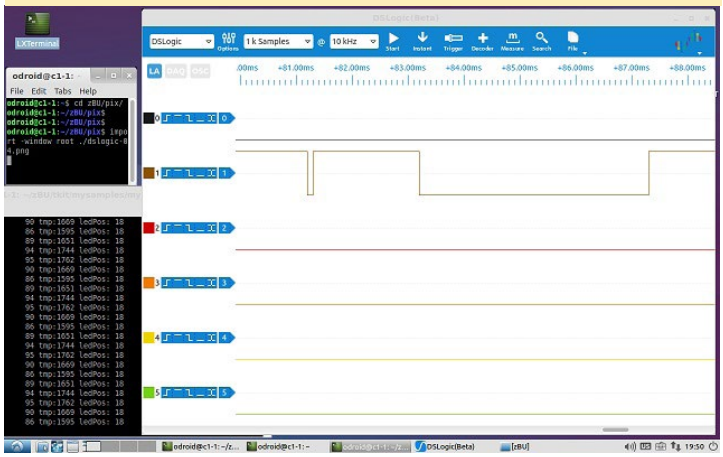


Figura 5: Configuración del analizador lógico DSLogic

Vuelve a iniciar la aplicación DSLogic-gui. Una vez más, mueve un dedo para bloquear y desbloquear el sensor de luz un par de veces. Deberías ver como los niveles lógicos cambian como muestra la Figura 6.

Figura 6: Analizador lógico DSLogic



Compilar Smartscope

Prepara una carpeta para incluir el último software SmartScope (versión 0.0.7.0):

```
$ cd ~ && mkdir smartscope && cd smartscope
```

Descarga el software SmartScope desde http://bit.ly/1BgUwTM y muévelo al directorio creado anteriormente. Instala el paquete debian:

```
$ sudo dpkg -i SmartScope-Linux-0-0-7-0.deb
```

Lanzar SmartScope-gui

Usando el manual de usuario y notas de la wiki, conecta el osciloscopio SmartScope al LED como muestra en la Figura 7.

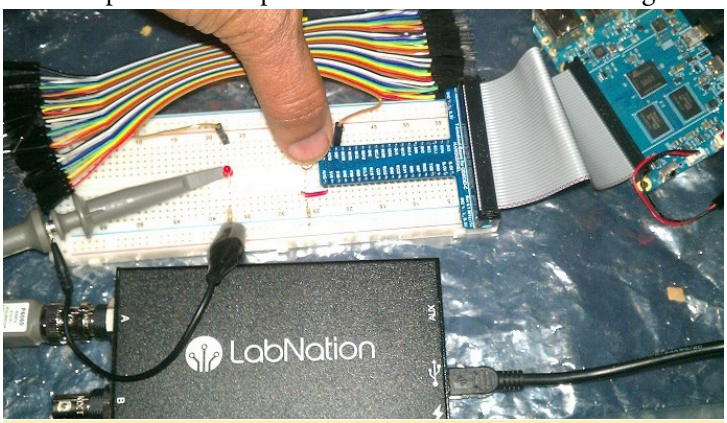


Figura 7: Instalación oscilloscope Smartscope

En un nuevo Terminal, inicia la aplicación de interfaz de usuario del SmartScope mediante el siguiente comando:

```
$ LIBGL_DEBUG=verbose sudo mono \
/opt/smartscope/SmartScope.exe
```

Aquí has de tener en cuenta la librería Mono. En otra sesión de terminal inicia la aplicación de ejemplo:

```
$ cd ~/tkit-example/
$ sudo ./myexample-led
```

Con el LED OFF / LOW (0 voltios), el punto de partida del GUI de osciloscopio debería ser similar a la Figura 8.

Después, mueve lentamente un dedo para bloquear la luz del sensor. El LED debería encenderse de nuevo. Con el LED ON / HIGHv (3.3V), la interfaz gráfica de usuario del osciloscopio debería parecerse a la Figura 9. La captura de pantalla coincide con el descenso de voltaje de 3.3V a 0V.

Dejare el uso final del analizador lógico SmartScope para el usuario. Espero que el lector sepa ver el potencial de todos los dispositivos que se han utilizado en este artículo para la creación de un poderoso laboratorio de electrónica portátil.

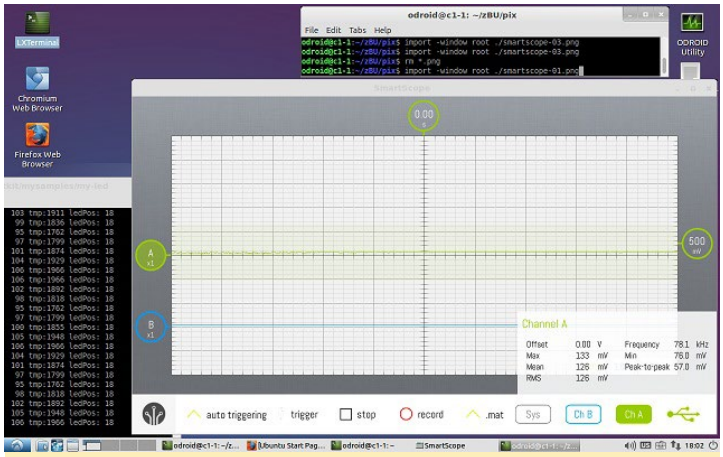


Figura 8: Punto de partida del osciloscopio SmartScope

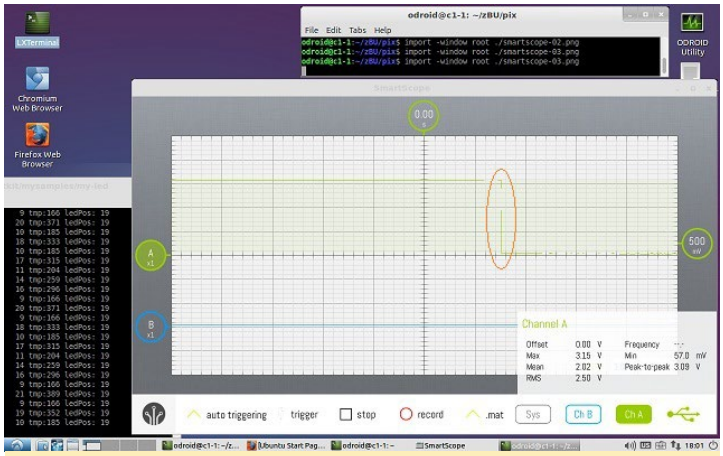


Figura 9: Osciloscopio SmartScope con 3.3 V

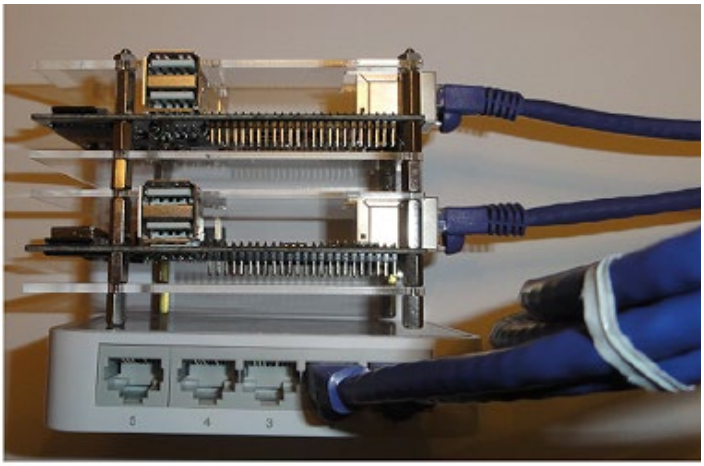
En el próximo artículo, estudiaremos el uso de estos analizadores lógicos y osciloscopios sobre el sistema operativo Android utilizando un ODROID-VU, con su intuitivo control táctil multipunto. Para obtener información adicional o realizar preguntas, por favor visita las siguientes fuentes de información:

- http://bit.ly/1NsrlU9
- http://bit.ly/1BcMRqW
- http://bit.ly/1HapizJ
- http://bit.ly/1zNxyES
- http://bit.ly/1zYhM57
- http://bit.ly/1BXoiR3
- http://bit.ly/1BcN11m

INFORMATICA DE ALTO RENDIMIENTO

DESARROLLA UN ECONOMICO CLUSTER PORTATIL CON CI O U3

por Dave Toth



Como profesor, mi objetivo era el de crear un clúster de informática de alto rendimiento que mis alumnos pudiesen comprar, en lugar de un libro de texto para el curso de informática, lo cual implicaba que éste tuviese que ser barato. Esta configuración permitía a cada estudiante poder hacer su trabajo, donde y cuando quisiera sin echar a perder los esfuerzos realizados por los demás alumanos en el equipo que compartían. También permite a los colegios con un presupuesto reducido tener clases de informática paralela.

Desarrolle mi primer clúster educativo con dos 2 nodos de doble núcleo (con un total de 4 núcleos) por alrededor de 200$ a finales de 2013. Si realizas las compras meticulosamente, podrías bajar el precio a los cerca 175$. Usando nodos DualCore, podemos probar de manera eficiente tanto la Interfaz de Paso de Mensajes (MPI) como OpenMP. Como referencia, un estudio sobre este tema y bajo en nombre de "Un Clúster portátil para cada estudiante", fue presentado en las cuartas jornadas NSF/TCPP sobre Educación informática distribuida y paralela (EduPar-14), en mayo de 2014. Mi siguiente clúster educativo costó un poco más, pero tenía 2 nodos quad-core. Acabo de terminar la tercera versión, que alcanzó un precio de alrededor de 150$ sin dejar de utilizar 2 nodos quad-core. Lo llame "El clúster de la media caja de zapatos", ya que ocupa la mitad de las típicas cajas de zapatos.

Lo mejor de estos clústeres es que yo te proporciono las imágenes, tan sólo tienes que descargarlas y trasladarlas a las tarjetas microSD, insertar éstas en las placas y encenderlas, ¡Tendrás un clúster configurado al instante! No tienes que instalar ni configurar MPI, ni fijar los nombres de host o direcciones IP, ni crear archivos o cualquier otra cosa.

Equipo auxiliar

- Dos cables Ethernet de 0,5 metros (0.49$ cada uno). Gaste unos céntimos más por cable para conseguir un color diferente por simple capricho. Se podría incluso utilizar cables de 6 pulgadas, pero podrían ser demasiados cortos.

- Un switch de red (9,99$). Si quieres sacarle partido al Ethernet Gigabit del clúster ODROID-C1, necesitarás un switch y cables Ethernet diferentes. El switch Ethernet Gigabit tiene un coste de unos 20$.

- Una caja de plástico Sterilite (0.97$)

Es posible que desees algunos otros elementos para que el clúster sea algo más cómodo de usar. Es bueno tener un teclado USB en caso de que no utilices SSH en los nodos, y un cable Ethernet adicional para que puedas conectar el clúster a tu router de casa y usar SSH de esta forma. Ten en cuenta que no he configurado ningún parámetro de seguridad en el clúster, así que tenlo presente.

Hardware

- Dos placas ODROID con adaptadores de corriente, ya sea el U3 o el modelo C1, dependiendo de tu presupuesto. Ambas placas tienen procesadores ARMv7 de cuatro núcleos. Pase por alto comprar las carcasas y en su lugar los he apilado usando separadores. Me gusta más los separadores de metal, pero al final use los de nylon por ser más baratos.

- Dos tarjetas microSD de 16 GB. Use las de clase 10 ya que su precio ha bajado bastante. Recomiendo las que vienen con el adaptador para poder conectarlas a una ranura para tarjetas SD estándar. Se tarda unos 20 minutos por tarjeta en trasladar las imágenes.

- Un cable micro-HDMI a HDMI. Puedes conectar el clúster a cualquier televisor o monitor HDMI compatible con resoluciones de 720p y 1080p.

Software

- Un programa para escribir la imagen en las tarjetas microSD. Yo uso el software gratuito Win32DiskImager para Windows. Si eres hábil con el comando dd en Linux, MacOS o través de Cygwin en Windows también puedes utilizarlo.

- Las imágenes para el nodo principal y el nodo secundario del clúster, descárgalas en la misma máquina que utilizarás para escribirlas en las tarjetas microSD. Si está utilizando dispositivos ODROID U3 para el clúster, descarga la imagen del nodo principal desde http://bit.ly/1wWZL6Z y la imagen del nodo secundario desde http://bit.ly/1BJPktk.

Si usas ODROID-C1, descarga la imagen del nodo principal desde http://bit.ly/18S8X7K y la imagen del nodo secundario de http://bit.ly/1BJQ8hS.

Instrucciones

1. Descomprime los archivos de imagen y escribe (no arrastrar y soltar) las imágenes en tus tarjetas microSD utilizando tu método favorito entre los mencionados anteriormente.
2. Inserta las tarjetas microSD en tus placas ODROID.
3. Conecta las placas ODROID a una regleta de enchufes.
4. Enciende la regleta.
5. Generalmente conecto un teclado y un monitor a cada nodo ODROID, pero tú no tiene que hacerlo.
6. Conéctate a cada placa con el nombre de usuario "odroid" y la contraseña "odroid"
7. En el directorio en el que se inician los nodos hay 3 archivos: machinefile, hellompi.c, y una versión precompilada de hellompi.c llamado hellompi. Puede probar el clúster ejecutando el siguiente comando:

```
$ mpirun -n 8 -f machinefile \
./hellompi
```

Si el resultado es de 8 líneas, cada una diciendo "Hi. I'm processor x, rank y of 8", donde "x" es uno de los dos posibles nombres (odroidtop y odroidbottom para el U3, o c1top y c1bottom para el C1) e "y" es 0, 1, 2, 3, 4, 5, 6 o 7, entonces es que el sistema esta configurado correctamente. Ten en cuenta que los valores de "y" no es probable que aparezcan en orden, lo cual es normal siempre y cuando aparezcan todos.

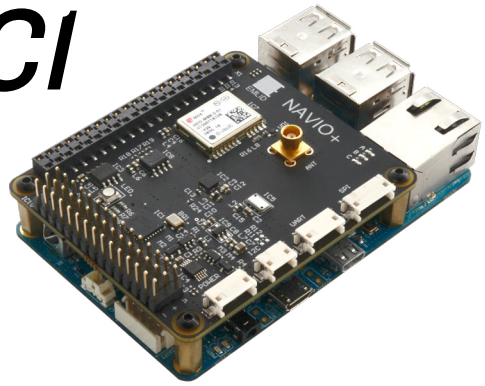
Si hacer un nuevo programa que use MPI (o recompilas uno existente), asegúrate de transferir el nuevo binario/ejecutable al otro nodo con un pen drive o por Secure Copy (SCP) antes de ejecutarlo o no funcionará correctamente.

Para realizar preguntas y obtener más información, por favor consulta el post original en http://bit.ly/1EOOZXi.

NAVIO+ PARA EL ODROID-C1

AUTOPILOTANDO TU DRONE

por Igor Vereninov



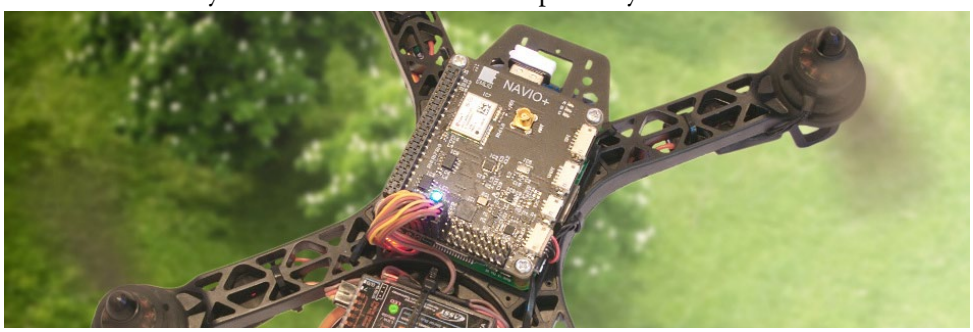
Navio+ es una extensión de piloto automático compatible con HAT. El objetivo principal del proyecto es desarrollar un sistema de piloto automático de nueva generación que se ejecute bajo Linux. Con Navio +, puedes hacer que cualquier vehículo por tierra o por aire sea autónomo. Todos los sensores necesarios se encuentran en la placa, incluyendo 9DOF IMU, sensor de presión barométrica, GPS, ADC y un generador de PWM. Es totalmente compatible con ODROID-C1 y se puede adquirir en http://www.emlid.com.

Crear el hardware adecuado no es suficiente para desarrollar un piloto automático. El software es clave. Hemos añadido soporte para APM, que es el más avanzado software de piloto automático de código abierto que hay disponible, te permite controlar helicópteros, aviones y robots. La estación de control desde tierra tiene muchas características y funciona casi con cualquier dispositivo.

Cuando apareció el ODROID-C1, estábamos muy entusiasmados con su potencial de cálculo e priorizamos la creación de un software compatible. Ya hemos desarrollado soporte para la combinación Navio+ y ODROID-C1 utilizando APM. La portabilidad fue en su mayor parte sencilla, gracias al HAL de APM, puesto que ya contábamos con los drivers para Navio +. Inicialmente, había una par de componentes del sistema no llego a funcionar como se esperaba, pero vemos que el ODROID-C1 está madurando y que el equipo Hardkernel está añadiendo rápidamente nuevas funciones y correcciones.

Otra cuestión importante para que un piloto automático sea fiable es el kernel RT_PREEMPT. Actualmente estamos trabajando en el desarrollo de un kernel en tiempo real, que se pondrá a disposición de todos los usuarios ODROID-C1 tras la pruebas. Estamos deseando ver las posibilidades que pueda ofrecer el increíble potencial de cálculo del C1. Ejecutará APM con facilidad, dejando una gran cantidad de potencia para otras tareas, como la mejora de los algoritmos de posicionamiento y la visión por ordenador entre otras. Hay varias cuestiones pendientes pero una vez solucionadas, ¡Nuestro proyecto ODROID-C1 echará a volar!

Para descargar el software Navio +, visita nuestro repositorio GitHub en http://bit.ly/18hK1oP.



JUEGOS LINUX

DESCUBRE EL MUNDO DE LA EMULACION CON NINTENDO DS (I)

por Tobias Schaaf

Los dispositivos ODROID pueden emular muchos y diferentes sistemas retro. Uno de los sistemas más singulares a emular es la Nintendo DS (i) a través de DeSmuME, que es la precursora de la actual Nintendo 3DS. Cuenta con una gran variedad de juegos disponibles y algunos tienen formas muy particulares de interacción a través del lápiz, DPAD, cámara y micrófono.

La cuestión es, ¿Estos juegos con esas opciones de interacción funcionan sobre ODROID? ¿Qué juegos no funcionan en ODROID? ¿Qué juegos tienen problemas? ¿Se puede jugar a estos juegos con un gamepad o se necesita un teclado y un ratón? Quiero analizar estas cuestiones y ver cómo los ODROIDs realizan la emulación de un sistema NDS.

Información general

Las primeras versiones de DeSmuME sobre ODROID eran muy lentas, sólo fui capaz de jugar a un par de juegos que sólo usaban gráficos en 2D. Aún entonces, dependía del propio juego si éste se ejecutaba a una velocidad aceptable y los juegos en 3D eran impracticables.

Puesto que ahora funciona el compilador JIT para ARM, la velocidad ha subido y el emulador se ejecuta de forma muy estable y rápida en el ODROID-U3 y en los modelos de gama más alta de ODROID. La velocidad de los videos es perfecta en todas las películas que he visto. Los Juegos 2D funcionan a toda velocidad en su gran mayoría, y muchos juegos en 3D se ejecutan a una buena velocidad, pero no todos funcionan. Además, cuanto más rápido sea ODROID, más rápido será el emulador. Si ejecutamos juegos pesados en ODROID-XU3, éste ofrece de 10 a 15 FPS más en comparación con el U3.

Control directo mediante DPad

Algunos juegos pueden ser directamente controlados a través del gamepad. Un juego al que me gusta jugar es “Bleach - The 3rd Phantom”, que es un juego de estrategia RPG en 2D que sólo puede ser controlado por gamepad.

Bleach – The 3rd Phantom (abajo y a la derecha) es un juego de estrategia RPG que sólo puede ser controlado por gamepad

El juego utiliza las teclas de acción y D-pad, no se puede controlar con el lápiz, lo que lo hace perfecto para controlarlo con un gamepad. El mando de la Xbox 360 es perfecto para este juego y puesto que es en 2D, el juego funciona bastante bien en ODROID, aunque hay algunos elementos en segundo plano que hacen que baje el rendimiento. Sin embargo estas escenas son poco comunes y el juego

es bastante divertido.

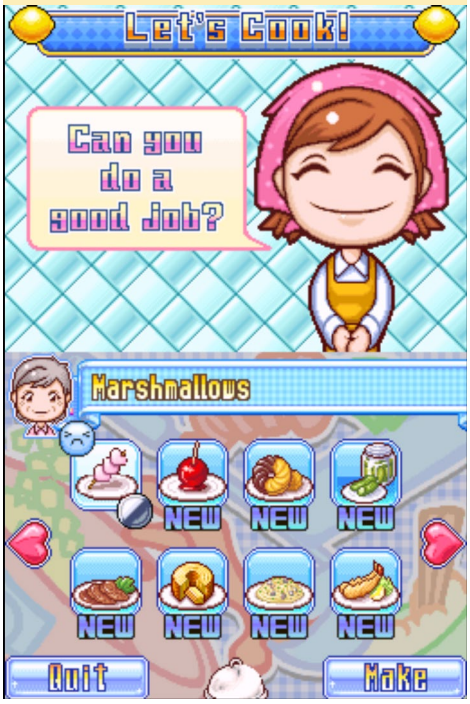
Básicamente, todos los juegos de NDS que he probado y que se pueden controlar con el gamead son bastante divertidos. Hay un montón de juegos de NDS que utilizan el mismo método de entrada, incluyendo Dragon Ball Z - Attack of the Saiyans. Este un poco más exigente, aunque sea en 2D. Tiene unos efectos especiales muy buenos, que en realidad se ven mejor en el verdadera NDS por la pantalla de alta calidad.

Dragon Ball Z – Attack of the Saiyans es otro gran RPG de acción que funciona muy bien con un emulador de NDS

Control táctil

Como decía al principio, todos los juegos que utilizan DPAD como entrada funcionan muy bien en ODROID, pero ¿qué ocurre con los juegos que utilizan la pantalla táctil? ¿Hay alguna manera de controlar la pantalla táctil con el gamepad? Y si es así, ¿La pantalla táctil funciona lo suficientemente bien como para jugar? ¿Qué otras opciones hay? Bueno, vamos a empezar con algunos ejemplos que implican precisión, dibujar círculos o realizar movimientos rápidos de un lado a otro. ¿Se puede hacer esto con un gamepad? Técnicamente, sí que puede. Puedes activar la emulación del puntero en los ajustes básicos de Retroarch y mapéalo de izquierda a derecha con el mando analógico. Esto mueve un pequeño cursor sobre la pantalla y puedes utilizar el botón R2 para simular un toque en la pantalla táctil. Aunque es factible, pondrá a prueba rápidamente tu paciencia, ya que sólo podrás reaccionar tan rápido con muevasn el gamepad o en las diferentes direcciones. Los movimientos complicados, como remolinos en círculos o arrastrar y soltar con precisión, son más difíciles. En algunos juegos, podría ser aceptable cuando sólo interactuas

Cooking Mama 3 (abajo y a la derecha) y Plants vs Zombies (derecha) son dos juegos de NDS que utilizan pantalla táctil



con la pantalla táctil para realizar algunas selecciones, pero en juegos como Plants vs Zombies o Cooking Mama 3 pondrás a prueba los límites del gamepad.

Así que, ¿hay alguna otra forma de interactuar con la pantalla táctil? ¡Existe! Puede utilizar el ratón para mover el pequeño puntero blanco y utilizar el botón izquierdo del ratón para "tocar". Esto funciona muy bien, pero aún así no es tan ideal como usar la pantalla táctil real de la NDS. El ratón parece funcionar buen si ejecutamos juegos como el Plants vs Zombies con interacciones relativamente simples, pero al intentar

jugar con Cooking Mama 3 nos encontramos con limitaciones. El juego en general funciona bien con el ratón y es jugable, pero pronto descubrirás que es realmente difícil hacer movimientos rápidos y precisos, incluso con el ratón. Probablemente, la mayoría de vosotros habéis intentado hacer un dibujo con el Paint o con un programa de dibujo similar usando el ratón. ¿Alguna vez has intentado dibujar un círculo con el ratón sin necesidad de utilizar la herramienta círculo? Se verá muy distinto a lo que es un círculo real. Eso es exactamente lo que ocurre con Cooking Mama 3 cuando se utiliza el ratón para controlarlo.

Aún así, con un poco de práctica es probable que te las apañes. En definitiva, la interacción con la pantalla táctil es posible, pero limitada. En la mayoría de los casos te verás obligado a utilizar el ratón para controlar el pequeño cursor blanco en forma de cruz. Esto funciona, pero tiene sus defectos. La entrada a través del ratón es limitada, especialmente si hablamos de precisión y velocidad. Además, el pequeño cursor blanco a veces es realmente difícil de ver. En una pantalla blanca y muy brillante es muy difícil determinar dónde debemos hacer clic.

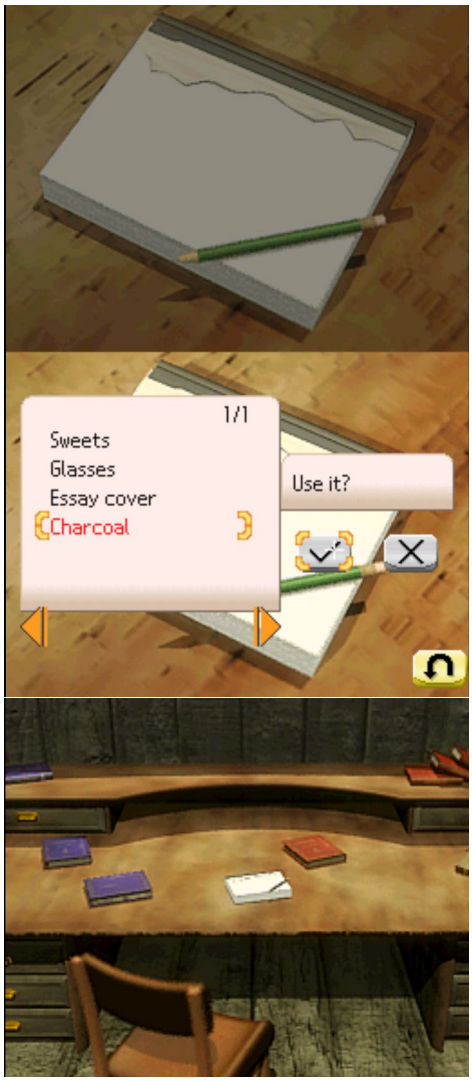
Me gustaría añadir que Cooking Mama 3 y Plants vs Zombies sólo usan gráficos 2D y por ello, son jugables a toda velocidad casi en su totalidad. Sin embargo en Plants vs Zombies notaras ralentizaciones si hay muchos artículos y/o enemigos en pantalla.

Otras entradas

El DPAD y la pantalla táctill no son los únicos métodos de interacción en la NDS. De hecho, está llena de pequeños artilugios como un micrófono y una cámara, y algunos juegos incluso utilizan el hecho de que se pueda plegar y cerrar tu NDS. Estos son métodos de entrada muy singulares, quiero ver si también es posible reproducirlos con el emulador.

Un juego muy interesante que utiliza muchos y diferentes métodos de entrada para resolver los puzzles es Another

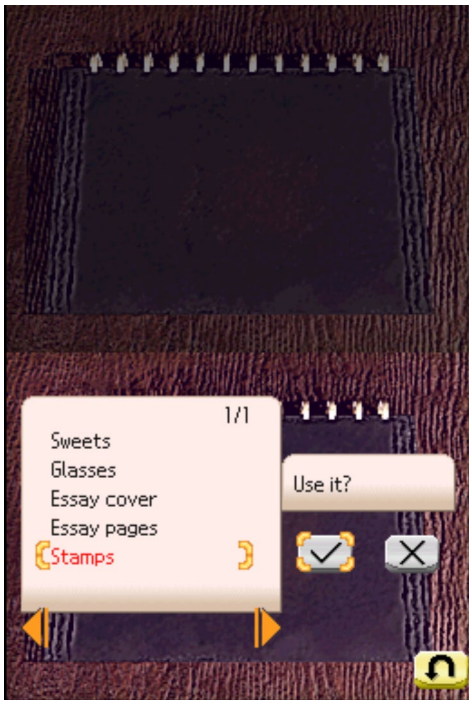
En Another Code - Two Memories (arriba, abajo y arriba a la derecha), tienes que resolver muchos enigmas con los diferentes métodos de interacción de la NDS



Code: Two Memories (Trace Memory en NA). Utiliza diferentes métodos de interacción como soplar en el micrófono para quitar el polvo de un cuadro, y cerrar y abrir la NDS un par de veces para "estampar" una imagen. Tienes que interactuar con la pantalla táctil rascando, girando y apuntando a las cosas. Es muy singular, no hay muchos juegos que hagan uso de estos métodos de entrada alternativos.

La pregunta es, ¿puede ODROID hacer esto también? Bueno, hay dos escenas en este juego que pueden dar respuesta: Estampar un cuaderno y soplar un cuadro polvoriento, que son dos de los métodos de entrada especiales de Another Code - Two Memories. Para estampar la imagen es necesario cerrar y abrir tu NDS, que al parecer es algo

Estampar un cuaderno y soplar un cuadro polvoriento, dos de los métodos de entrada especiales de Another Code - Two Memories (arriba, abajo y en la página siguiente).



difícil de imitar en un emulador. A pesar de todo, había otros juegos que también requerían cerrar y abrir la tapa un par de veces para poder avanzar. Era raro pero se utilizaba, Another Code - Two Memories era realmente bueno para utilizar todas las opciones que ofrecía la NDS.

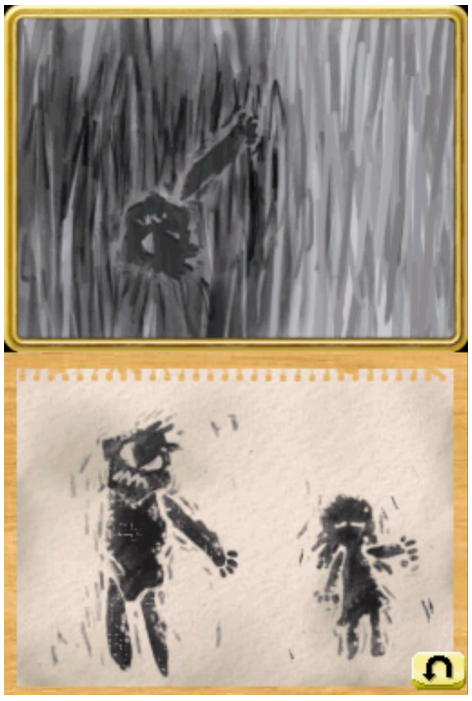
La NDS tenía un micrófono incorporado y había muchos juegos que podían hacer uso de él. Pero la mayoría de los juegos no necesitaban una entrada especial, sino que reaccionaron a cualquier tipo de "ruido", lo que significa que soplar o arañar en el micrófono funcionaba en la mayoría de los juegos que requerían entrada de micrófono. Algunos emuladores usaban esta circunstancia e incluso, emulaban un "ruido de fondo" para simular la entrada de micrófono.

En Retroarch y Libretro, el núcleo DeSmuME mapea el botón L2 para abrir y cerrar la tapa, con sólo pulsar el botón L2 cierras la tapa y presionando de nuevo se vuelve a abrir. De esta forma, puede hacer clic en el botón L2 un par de veces y pasar el puzzle del estampado.

Aunque la acción de estampado es fácil de resolver, ¿qué pasa con los puzles en los que hay que soplar? Bueno, resultó que esto no era tan fácil de abordar. La versión del núcleo DeSmuME libretro que estaba usando no ofrecía esta característica. Miré el proyecto y encontré un informe de errores donde aparentemente fue resuelta esta cuestión a principios de enero. Puesto que mi núcleo era de finales de diciembre, esta revisión no estaba incluida. Analice la nueva versión del núcleo, luego la compilé junto con la versión más reciente de Retroarch y lo intente de nuevo. La experiencia de juego había mejorado bastamte. Comprobé el código y encontré que la simulación de ruido fue asignada al botón L3. Así que también lo probé.

Funcionaba y con ello fui capaz de utilizar todas las características que ofrecía la NDS, a excepción de la cámara, aunque no recuerdo ningún juego que realmente la utilice. Another Code – Two Memories es un juego muy singular que utiliza muchas funciones de la NDS para resolver todo tipo de puzzles.

Resolviendo el puzzle del estampado (abajo ya la derecha), normalmente realizado por el cierre y la apertura de la tapa, que se puede haver pulsando el botón L2



La historia es muy interesante y divertida. El juego utiliza una mezcla de elementos 3D y 2D. Aunque la mayoría de los puzzles y conversaciones con la gente está en 2D, moverse por el mapa se hace en 3D, de modo que el juego varía su rendimiento en función de las escenas. Mientras que los elementos en 2D funcionan a toda velocidad, los elementos 3D a veces bajan a 40 FPS o más, lo que puede llegar a ser un poco molesto ya que el sonido empieza a ralentizarse un poco. Pero, puesto que tienes todo el tiempo que quieras en el juego, y no hay escenas de acción que requieran desplazarse rápido o algo similar, no interfiere en la experiencia global del juego.

Conclusión

La emulación NDS sobre ODROID funciona bien porque hay una equivalencia para todo, aunque yo no he sido capaz de utilizar directamente el micrófono que conecté. Aún así, esta es una increíble pieza de trabajo que te permite ejecutar todos los juegos disponibles para la NDS y DSi. Aunque todos los juegos llegan a ejecutarse, no todos lo hacen al 100%, realmente depende del propio juego el que consigas una buena experiencia.

De cualquier modo y puesto que hay miles de juegos para NDS y DSi,

CLASH OF CLANS

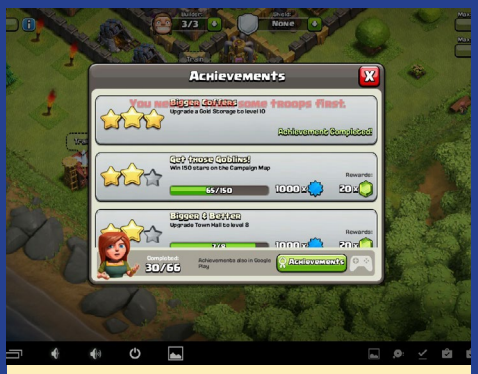
BATALLAS EPICAS EN LA GRAN PANTALLA

por Jeremy Leesmann

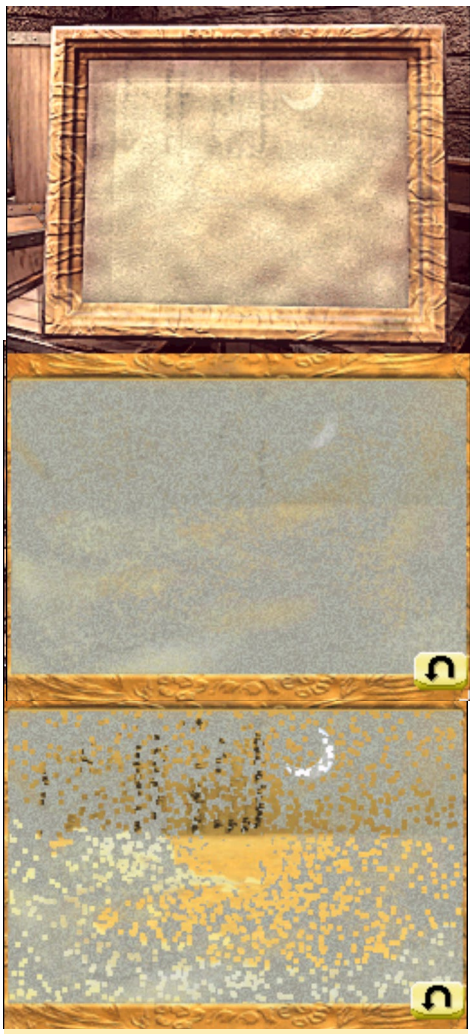
Clash of Clans te sumerge en las mejores batallas épicas. Este juego requiere de grandes dosis de reflexión y estrategia para construir una base bien fortificada, desarrollando al mismo tiempo una fuerte ofensiva para ganar las Guerras de los Clanes. Si quieres perder un montón de tiempo libre (en el buen sentido), entonces disfrutarás con Clash of Clans.

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

Construye tu base, luego defiendela de los invasores que quieren destruirla



Clash of Clans tiene logros que puedes desbloquear conforme avanza el juego, dándole recompensas como experiencia y gemas



Más vistas del puzzle del estampado, que utiliza una singular acción de la tapa que sólo se implementó en el sistema NDS. Los autores del emulador tuvieron que buscar el modo de simular una forma no tradicional de interactuar con el dispositivo.

merece la pena echar un vistazo y probar a jugar. Además, tu ODROID mejorará notablemente la experiencia de juego que va a tener. Esto significa que el ODROID-XU3 supera al U3, que a su vez supera al C1 en términos de fluidez y rendimiento. La NDS sobre ODROID puede resultar bastante divertida, y existen algunos juegos realmente buenos para la NDS, y puesto que actualmente mi DSi XXL tiene algunos problemas, realmente disfruto jugando a mis juegos de DSi con ODROID.

La original Nintendo DSi venía con diversos colores, incluyendo el arco iris.

ODAMEX

JUEGA AL DOOM EN UNA PANTALLA PANORAMICA DE ALTA DEFINICION EN MODO MULTIJUGADOR

por Jeremy Kenney

Doom está con nosotros desde hace muchos años y ha ido acumulando una gran público. Aún se sigue jugando hoy en día. Incluso el galardonado Wolfenstein 3D que ha resurgido recientemente en forma de Total Conversion usa el motor de Doom.

En mi último artículo, presenté un tutorial sobre cómo compilar Doom SDL. Pero esa no era la mejor experiencia con Doom: ¡quieres más daño, más matanza monstruosa y más laberintos para difrutar con tus amigos! Odamex ya está disponible para ODROID, cuenta con una experiencia de juego a toda velocidad, modos de vídeo configurables, incluidos los de pantalla panorámica a 720p y 1080p, la posibilidad de configurar y encontrar fácilmente servidores, reproducción MIDI y mucho más. Esta versión original de Doom de alta calidad te proporcionará acción a destajo y millones de wads (mapas 3D) compatibles.

Los wads creados para otras versiones pueden requerir archivos adicionales, o incluso pueden no funcionar. De cualquier modo, era de esperar ya que algunas versiones han cambiado el código fuente original a algo más "moderno". La versión Odamex te hace sentir verdaderamente este juego, además de ofrecer opciones avanzadas. En unos pocos pasos podrás tener instalado y funcionando rápidamente el Doom multijugador.

Requisitios

Vas a necesitar libsdl 1.2 o superior para que este programa funcione. Si por casualidad conoces una librería que te permita tener un "puerto MIDI" abierto para la reproducción MIDI como freepats, instálalo igualmente. La biblioteca MIDI es opcional pero te permite tener música en el juego.

Ahora que tiene todas las librerías instaladas, descarga el paquete Odamex desde http://bit.ly/1DXz2MX y el paquete del servidor de http://bit.ly/18SdfMl. Instálalo como lo harías con cualquier otro paquete de Linux abriendo Terminal y escribiendo:

```
$ sudo dpkg -i \
packagenamehere.deb
```

Tras instalar la parte cliente y servidor de este software, dispondrás de "odamex" y "odasrv" como comandos ejecutables en Terminal. Si has elegido descargar la interfaz gráfica de usuario, también tendrás "odalaunch". Odamex es el cliente, odasrv es el comando de inicio del servidor y odalaunch es el lanzador. Ahora vamos a pasar a configurar el juego.

Instalar wads

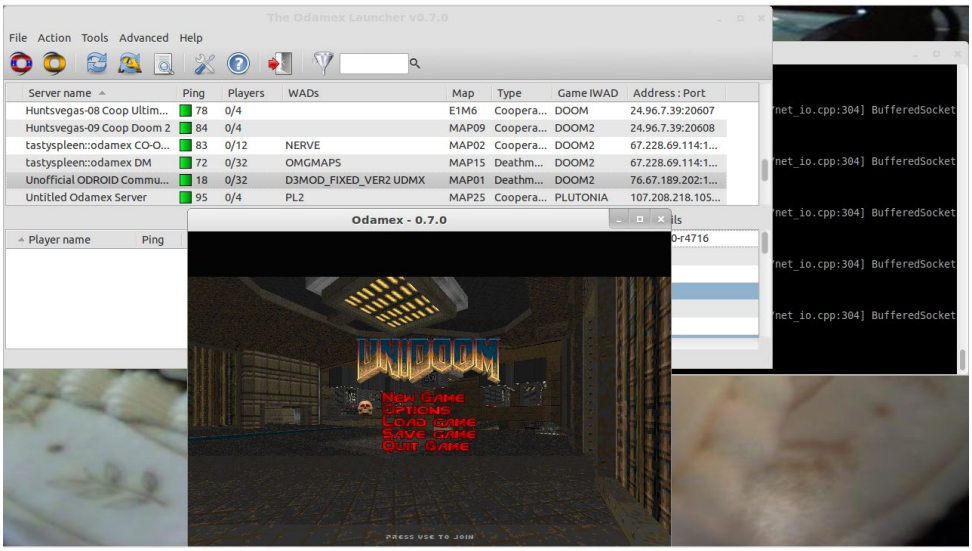
Para ahorrar algo de dinero, descarga el wad shareware de http://bit.ly/17TeidM, ya que las copias originales del juego cuestan 5$ o más. Sin embargo, si has comprado los disquetes o CDs originales del juego, puedes copiar DOOM.WAD y DOOM2.WAD en "/usr/local/share/odamex" para que el juego pueda localizar los archivos wads. Si deseas cambiar el directorio de los wads, has de hacerlo al ejecutar odamex. Escribe esto para ejecutar el juego:

```
-waddir /path/tu/directorio/qui
```

Tras ejecutar el juego, puede que quieras optimizarlo para el modo de vídeo en alta definición y activar un par de opciones más. Ve al menú " Options " y selecciona "Set Video Modes", " Fullscreen on" y "1280 x 720". Ten en cuenta que si utilizas cualquier resolución superior a 720p, necesitarás activar los modos de detalle horizontal y/o vertical con el fin de facilitar la conversión. Puedes poner o quitar los marcos si lo deseas, pero en 720p o superior, no conseguirás quitar los marcos sobre un ODROID-U3.

Acción multijugador

Si quieres jugar en modo multijugador, presiona F8 para ver los mensajes de los otros jugadores y para activar el sonido de notificación de mensajes. Los mensajes también se pueden ver en la consola presionando la tecla ~. A continuación, tendrá que configurar el servidor si desea alojar un juego. Existe una carpeta de muestras para varios tipos de juegos localizada en la carpeta "/usr/local/share/odamex/" llamada "config-samples", que podrás iniciar en algunos servidores básicos.



Coge una de las muestras de servidor y copiarla en el directorio .odamex en tu carpeta home. Renombra el archivo a "odasrv.cfg", luego editarlo utilizando tu editor de texto favorito. Usando el editor puede configurar el correo electrónico, nombre de host, el mensaje del día (MOTD) que aparece en cada primera conexión al servidor por cualquier jugador, el sitio web si lo tienes y la ruta del directorio wad. Si te desplazas hacia abajo un poco más, puedes configurar el modo de juego que desees especificando el límite de tiempo, el límite de fragmentaciones, la gravedad, las opciones de compatibilidad y algunas cosas más.

Ahora tienes configurado correctamente tu servidor para jugar online con tus amigos. Si quieres ver una lista de servidores maestros, puedes escribir "odalaunch" en el Terminal, aparecerá un error al principio. Simplemente desmarca la casilla "show dialog next time" y pulsa continuar. Si el programa no muestra ningún servidor tras uno o dos minutos, haz clic en "Action" y luego en "Get List". Ahora, puede configurar odalaunch haciendo clic en "File", despues en "Settings" en la pestaña identificada como File Locations. Haga clic en "Odamex Path" y selecciona "Other". Busca tu ejecutable ubicado en /usr/local/bin, haz clic en "Open" y selecciona el icono de la carpeta para añadir las carpetas en las que guardas tus wads, por ej. /home/ODROID/ Downloads/Wads/.

Después de elegir un servidor, el cliente se descargará e instalará automáticamente los wads si fuese necesario. Los wads se instalarán uno por uno, por lo que necesitas escribir "Reconnect" tras descargarte e instalar cada wad. Puede que también quieras entrar en "Player Setup" para cambiar tu nick y el color del DoomGuy. Hay un servidor shareware disponible en 74.207.250.98:10668 por si no cuentas con una copia de Doom, y he abierto un servidor público no oficial para la comunidad ODROID en 76.67.189.202:10666. Cuando escribes *connect* e introduces la IP, puedes dejar en blanco el puerto ya que el puerto por defecto es el 10666. No puedo garantizar que mi servidor este abierto 24/7 debido a los cierres de mantenimientos normales, pero si recibes un mensaje de no disponible, el servidor debería estar operativo en menos de 2 horas. Ejecutar Doom onlíne nunca ha sido tan facil como en Odamex ejecutándose en ODROID. ¡La divertida experiencia 3D de los años 90 nunca se abandona!

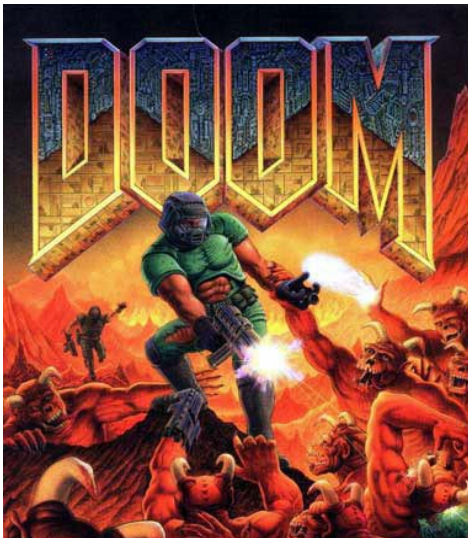
Bono para lectores

Con el de premiar a la gente que lee este articulo, he decidido exportar el juego "Commander Keen Dreams and The Catacombs Series" a ODROID. Este contenido exclusivo al que puedes tener acceso es una forma de dar las gracias a toda esos usuarios que disfrutan leyendo ODROID Magazine.

En primer lugar, descargar los archivos del juego desde http://bit.ly/1DXC2Zr. Si no dispones de una copia de "Commander Keen Dreams and The Catacombs Series", puedes ejecutar la versión shareware del juego. Tras descargar el archivo comprimido, extráelo en una nueva carpeta, coge cualquiera de las copias que poseas (o la versión shareware) y asocia todos los archivos con el ejecutable correcto. La leyenda es la siguiente:

```
refcat3d-100 = Version 1.0 Catacombs 3D
refcat3d-122 = Version 1.22 Catacombs 3D
refcatabyss-113 = Version 1.13 Catacombs Abyss
refcatabyss-124 = Version 1.24 Catacombs Abyss
refcatapoc-101 = Version 1.01 Catacombs Apocalypse
refcatarm-102 = Version 1.02 Catacombs Armageddon
refkdreams-cga105 = Keen Dreams CGA Version 1.05
refkdreams-reg193 = Registered Version Keen Dreams 1.93
refkdreams-shar113 = Shareware Keen Dreams Version 1.13
refkdreams-shar120 = Shareware Keen Dreams Version 1.20
```

Puede encontrar más información sobre Catacombs shareware en el archivo readme incluido, junto con otros excelentes consejos.



MAPEAR DPAD DEL MANDO DE LA XBOX 360 EN ANDROID

USA TU GAMEPAD POR COMPLETO

por @seismograf

Aunque los mandos con cable e inalámbricos de la Xbox 360 funcionan con todos las versiones de Android de Hardkernel, el DPAD no está mapeado por defecto en los archivos del controlador, lo cual hace que algunos emuladores tengas dificultades. Siguiedo estas instrucciones puedes activar los controles del DPAD y habilitarlos para mapearlos en PPSSPP y en muchas otras aplicaciones de emulación Android.

Para empezar, descarga los archivos .kl pre-compilados desde http://bit.ly/1aKX4Rq, colócalos en algún lugar apropiado como la carpeta /sdcard0/download. El archivo Vendor_045e_Product_0719.kl se utiliza para el mando inalámbrico Xbox 360 y el archivo Vendor_045e_Product_0291.kl es para la versión inalámbrica.

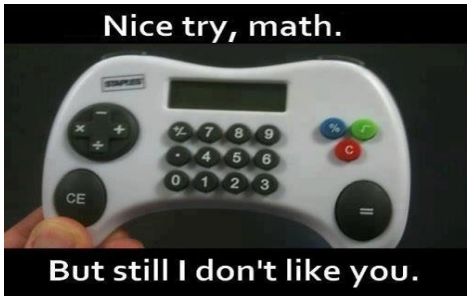
A continuación, abre una sesión terminal usando la aplicación Terminal y escriba lo siguiente:

```
$ su
$ mount -o remount,rw /system
$ cp /storage/sdcard0/download/\
Vendor_045e_Product_0719.kl \
 /system/usr/keylayout/
$ cp /storage/sdcard0/download/\
Vendor_045e_Product_0291.kl \
 /system/usr/keylayout/
$ cd /system/usr/keylayout/
$ chmod 644 \
Vendor_045e_Product_0719.kl
$ chmod 644 \
Vendor_045e_Product_0291.kl
$ exit
```

Alternativamente, puede añadir las siguientes líneas a cualquier de los archivos /system/usr/keylayouts/Vendor_045e_Product_0719.kl o /system/usr/keylayouts/Vendor_045e_Product_0291.kl

```
key 704 DPAD_LEFT
key 705 DPAD_RIGHT
key 706 DPAD_UP
key 707 DPAD_DOWN
```

Gracias a los miembros del foro de XDA-dev, que aportaron la solución en http://bit.ly/1BFRH0v. Para preguntas y comentarios o realizar sugerencias, por favor visita el post original en http://bit.ly/1BFRPwI.



BOOM! TANKS

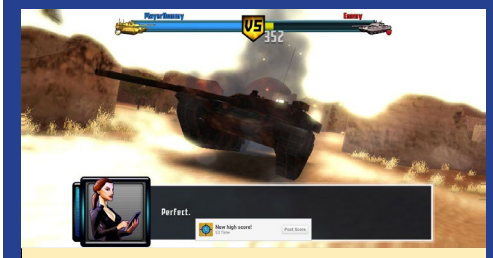
SIMPLEMENTE MATA O MUERE EN EL COMBATE

por Jeremy Leesmann

En BOOM! TANKS sólo tiene que preocuparse de ser el artillero, no conducir el tanque. Es un gran juego de combate que no es difícil de dominar. Construye tu tanque y haz explotar al resto antes de que te destruyan.

```
https://play.google.com/store/
apps/details?id=com.
reliancegames.android.
boomtanks&hl=en
```

Boom! Los Tanques te permiten soñar con destruir todo lo que veas



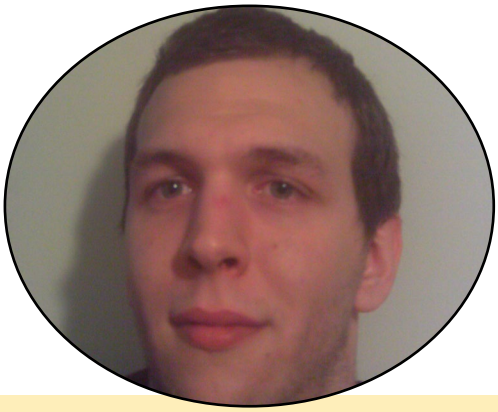
La acción rápida es fácil de aprender, pero difícil de dominar

Los Juegos Android han recorrido un largo camino, y Boom! Tanks ejemplifica lo mejor del género shoot

CONOCIENDO A UN ODROIDIAN

JEREMY KENNEY (@CARTRIDGE) NUESTRO EXPERTO EN JUEGOS RETO

editador por Rob Roy



Jeremy Kinney produce mucho software ODROID

Por favor, Háblanos un poco sobre ti.

Tengo 23 años y me acabo de comprar mi propia casa en Canadá. Mi lengua materna es el francés y también soy un gran fan de Sega. No es que Nintendo no haya hecho grandes juegos, puesto que hay muy buenos juegos para Nintendo, pero simplemente no soy un fan de Nintendo. Soy la típica persona a la que le gusta la forma tradicional de jugar a los juegos multijugador, con amigos sentados uno junto al otro disfrutando de un buen cartucho de juegos. ¡Por eso me suelo llamar Cartridge en los foros!

¿Cómo fueron tus inicios con ordenadores?

Estaba a punto de cumplir los 3 años cuando mi tío le dio un ordenador a mis padres, que almacenaba una copia de Wolfenstein y al que nunca dejé de jugar. Luego recibí una copia demo de algunos software BBS de Internet, también contenía una copia shareware de Doom y un juego llamado Jetpack. Estos dos juegos fueron la puerta de entrada de mi experiencia con los ordenadores hoy día. Estaba tan asombrado que no podía dejar de recibir más y más demos para ver los estilos que se utilizaban y en que consistían los juego. He intentado en múltiples ocasiones crear mis propios juegos, animaciones y música. Todo lo que conozco actualmente es por haber sido autodidacta. Nunca pude conseguir que alguien fuese mi mentor, ya que en el pueblo que vivía era practicamente Amish. Era un gran jugador de demos y jugaba a todo lo que encontraba. Después me topé con algunos libros de Windows, desconocía lo que era Windows por aquel entonces. Los leía constantemente y justo entonces empecé a programar en bash, que es muy básico y fácil de aprender. Los scripts Bash fueron mi puerta de entrada a Linux, ya que nunca he visto Windows como "algo bueno". Con los interminables drivers, pantallas azules y con los muchos errores a problemas simples, Windows era una pesadilla.

¿Qué te llevó a la plataforma ODROID?

Estaba buscando algo que pudiera ejecutar en Linux y que pudiese desarrollar por mí mismo. Nunca había escuchado las palabras "procesadores ARM" por aquel entonces. Seguía gastando dinero en desarrollar algo mejor cada año. Esto me llevó a ciertos problemas financieros provocando que mis desarrollos se parasen. Entonces me topé con las placas Raspberry Pi, pero oí cosas negativas sobre ellas, lo que me llevó a apuntar una dirección diferente. La velocidad del procesador y la capacidad de RAM, junto con la clavija AV/OUT (ya que HDMI no estaba en la RPI en ese momento) me hizo seguir buscando más información. Entonces, alguien en un determinado foro mencionó la familia de ordenadores ODROID. En seguida encontré lo que tengo hoy día: un ODROID-U2 que está a punto de alcanzar los 3 años de edad y su funcionamiento sigue estable como una roca. El procesador quadcore, la GPU Mali y la RAM eran todos ellos recursos muy valiosos, y el U2 hace que los ordenadores sean tan baratos que te permite comprar una nueva placa de vez en cuando, en lugar de tener que adquirir componentes cada año para mantenerte al día con los nuevos juegos y archivos que se van publicando.

¿Qué ODROID es tu favorito?

Mi ODROID favorito es el U2. Me gusta decir que los ODROIDs tienen "procesos explosivos" como un divertido chiste de los 90. También he tenido la oportunidad de probar el U3, pero me causa mejor impresión el formato del U2. Ajusta bien y el disipador facilita el trabajo, no necesitas tener una carcasa.

¡Tus aportaciones de software ODROID son muy populares! ¿Cómo llegaste a ser tan hábil con Linux?

Esto se lo tengo a agradecer al usuario @meve-ric de los foros Hardkernel. Yo tenía una idea sobre programación Bash (DOS), pero aprendí como exportar y modificar código gracias a él. Me mostró al detalle cómo se hace y qué hacer cuan-do aparecen errores durante la compilación. Por supuesto, Google también me dio algunas respuestas, pero la gran comunidad de los foros Hardkernel ha mejorado notablemente mis capacidades. Compilar diferentes programas significa tratar con diferentes librerías, lo que implica siempre más esfuerzo y tiempo. Llege a aprender bastante con el código fuente de otras personas.

¿Qué aficiones e intereses tienes aparte de los ordenadores?

Me gusta jugar al golf de vez en cuando y el ciclismo es una de mis pasiones. Me gusta caminar por los parques. El arte es algo que también me encanta tanto como las ilustraciones en los juegos. No se trata de la forma gráfica en sí, sino de lo que representa y lo que significa.

¿Está involucrado en otros proyectos informáticos ajenos a ODROID?

No necesariamente, sólo numerosos intentos de tener un sitio web para almacenar cosas divertidas de internet. También estoy buscando un software beta para un sitio web de conservación. Tengo un pequeño proyecto con Windows 98 a través de foros en TheIsoZone (www.theisozone.com). El proyecto consiste en la optimización de tu actual Windows 98SE para ejecutar la mayoría de los programas de hoy día, incluye posibilidades para HTML5, Flash, MSVCRT, extensiones del kernel y mucho más. Puedes buscarme en TheIsoZone bajo el nombre de Cartridge.

El Equipo de Jeremy nos muestra que él es un pura raza de la informática de los 90.

¿Qué tipo de novedades de hardware te gustaría ver en futuras placas Hardkernel?

Es difícil pensar en algo mejor cuando ya lo tienes, pero luego está sueños. ¿Por qué el Raspberry Pi es tan popular a pesar de su velocidad y capacidad de RAM? Si lo usas bien, puede conseguir cualquier cosa. Por esta razón, ser demasiado rápido o demasiado lento no es algo determinante para mí. Es más una cuestión de lo que es capaz de hacer y ODROID ya es capaz de mucho, más de lo que algunos piensan. Así que se trata de optimizar lo que ya tenemos y tal vez alguien con un poco de creatividad pueda poner un chip junto a la CPU ARM para procesar código x86. Esto puede ser imposible, por supuesto. Pero podemos soñar con ello.

¿Qué consejo le darías a alguien que quiera aprender programación?

Doom es un gran juego para empezar a programar. ¿Pero si es un juego? Sí, efectivamente es un juego, pero también es sumamente modificable y utiliza el lenguaje ensamblador. Si consigues un Editor de Doom, puedes hojear los ward y ver algo de código. La codificación en Doom imita la misma estructura que el lenguaje C. Es una buena puerta de entrada para aprender la codificación y además, hay un montón de términos que usa Doom y que aparecen en casi todos los lenguajes de programación.

Puedes empezar a programar con Doom descargandote "XWE Doom Editor". Con este programa puede codificar, introducir texturas, parches, sprites, sonido y música. Si descargas un wad modificado (los mods contienen más código que los wads originales) puedes ver cómo están programadas las armas y cómo funcionan. Dirígete a la Wiki ZDoom (http://bit.ly/1EiBSgt) donde encontrarás una chuleta de comandos para programar. Hay muchos mods que sólo utiliza la codificación ASM. Por ejemplo, si descargas el wad de http://bit.ly/1LsvDK8, tienes mucho donde aprender. Usa lo que ya está en el juego para crear una nueva versión. Por supuesto, puedes hacer que esto sea muy fácil, hacer nuevos sprites, nuevos monstruos o nuevas armas con sólo cambia la apariencia. No sólo estás aprendiendo a programar con Doom sino que además estás haciendo mods que es una forma muy divertida de aprender. Recuerda que la wiki ZDoom está siempre disponible para ayudarte.