Instalación Driver Pantalla Táctil • Imagenes de la comunidad para CI

ODROID Magazine

Año Dos
Num. 18
Jun 2015

Caja Escritorio

El mejor modo para presumir de tu clasico ordenador de juegos ODROID

El original ODROID

- Car PC para ODROID-U3
- Usar un CPLD como conmutador programable
- UltraStar Deluxe Karaoke
- Juegos Nintendo 64: parte 2

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

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

Dongjin, uno de los ingenieros Hardkernel, ha decidido construir una caja para su ODROID que se parece al modelo Macintosh Plus de su juventud. Creemos que hizo un gran trabajo con la creación de una versión moderna y futurista de este clásico equipo. Desde que publicamos un artículo sobre el desarrollo de un Truck PC el año pasado, hemos tenido varias solicitudes de artículos similares para los propietarios de sedán. En respuesta, Belov ha sustituido el equipo electrónico integrado de su Opel Astra por un ODROID-U3 que permite la reproducción de música y la navegación GPS. Venkat también presenta un proyecto muy bueno que te permite obtener información detallada del vehículo a través de Bluetooth, descargarla en un ODROID y trazar los resultados usando Google Earth. Tobias continúa su serie de Nintendo 64 con comentarios de varios juegos muy comunes de N64, Nanik detalla el proceso de creación de un servicio personalizado para Android, Bo nos enseña cómo instalar los drivers de la pantalla táctil en un ODROID-C1, y Carsten presenta su proyecto Guzunty Pi que desarrolla una consola UART barata. También convertiremos un ODROID en una máquina de karaoke, jugaremos al Tekken 6 en alta definición, instalaremos algunas de las imágenes pre-compiladas más relevantes, ¡y mucho más!

ODROID Magazine, que se publica mensualmente en http://magazine.odroid.com/, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera arquitectura ARM "big.LITTLE" del mundo basada en una única placa.
Para obtener información de cómo enviar artículos, contacta a través de odroidmagazine@gmail.com, o visita http://bit.ly/lyplmXs.
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 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 Atistico Senior

Nada conoce mejor a nuestro Editor Artístico Senior que las largas noches, las cuales las pasa haciendo y rehaciendo la portada de la revista. Como un Géminis realmente disfruta, incluso cuando su familia se esfuerza por llamar su atención y averiguar lo que quiere para su cumpleaños. ¿Qué edad tienes Bruno? ¡100.101 años!

Nicole Scott, Editor Artístico

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

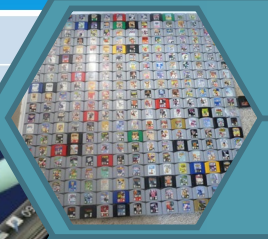
James LeFevour, Editor Artístico

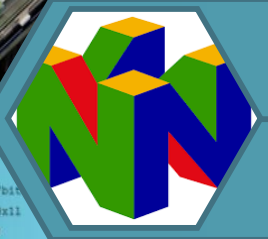
Soy un especialista en medios digitales que disfruta trabajando como freelance en marketing de redes sociales y administración de sitios web. Cuanto más aprendo sobre las posibilidades de ODROID más me ilusiona probar cosas nuevas con él. Me traslade a San Diego desde el Medio Oeste de los EE.UU. Todavía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

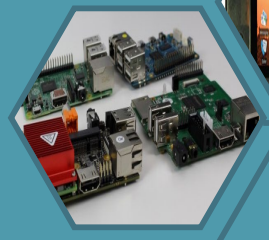
Manuel Adamuz, Spanish Editor

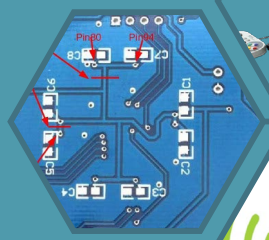
Tengo 31 años y vivo en Sevilla, España, y nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y las tecnologías de la información: ISO 9001, ISO 27001,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









CAR PC CON U3

REEMPLAZA EL EQUIPO DE FABRICA ESTANDAR

por Belov Vitaly

Mi coche, un Opel Astra H viene equipado de serie con una radio, un reproductor CD MP3 y una pantalla de 4 pulgadas en blanco y negro. Echaba en falta entradas USB para conectar pen drive con música MP3 y tenía que usar un smartphone para la navegación GPS, que constantemente necesitaba cargar y que sólo podía colocar en el parabrisas. En un principio, compre un dispositivo prefabricado comercial para reemplazar la unidad principal, pero era muy incómodo porque la pantalla del dispositivo estaba demasiado baja, al nivel de la cintura. Otra posibilidad era montar una tablet de 7 pulgadas como la Nexus 7, pero no encajaba en el hueco, ya que el Astra H solo tiene espacio para una pantalla LCD de 6,5 pulgadas, demasiada pequeña para mí. Así que decidí desarrollar un CarPC con Android e instalarlo en lugar de tener una pantalla en blanco y negro.

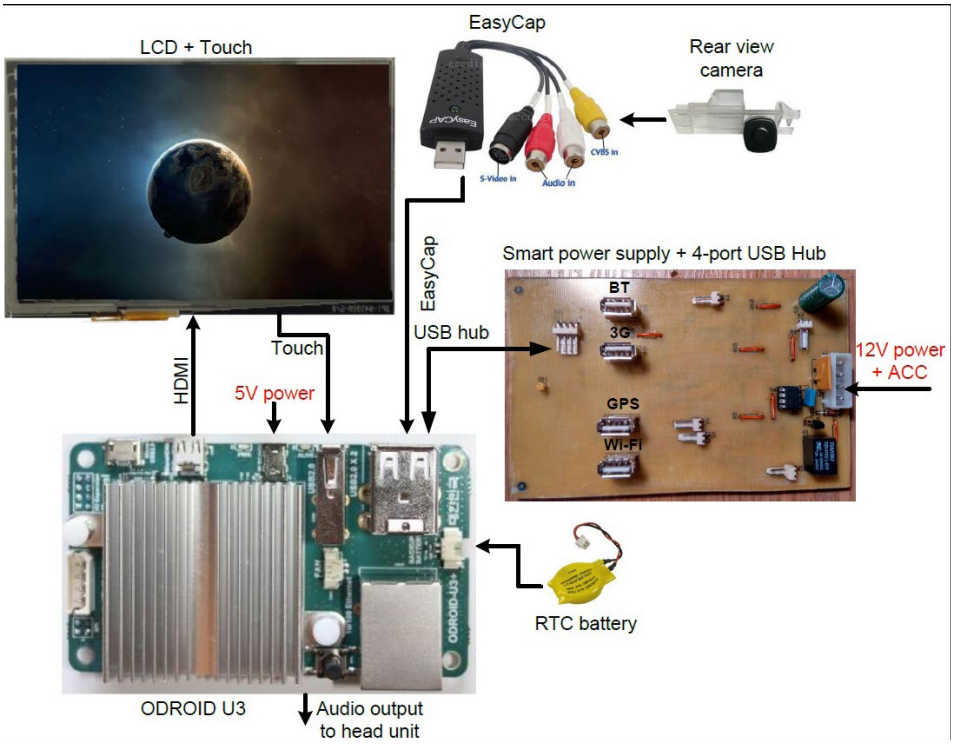


Diagrama de componentes

Instalación Completa del Car PC

Necesitaba que mi CarPC tuviese las siguientes funciones:

- Navegación GPS
- Reproductor Mp3
- Radio por Internet
- Reproductor de vídeo
- Cámara de visión trasera

Opté por un ODROID U3, ya que es un equipo económico pero muy potente con excelente rendimiento y bastante memoria.

Componentes

- ODROID-U3 con eMMC de 8Gb
- Bacteria RTC (Real Time Clock)
- LCD IPS 1280x800 de 7" con sistema multitáctil de 5 puntos
- Marco para LCD de 7" (desarrollo propio, impresión 3D)
- Dongle USB Bluetooth
- Dongle USB WiFi
- Atmel ATTiny
- Modem USB 3G HUAWEI E1550
- Holux M-215+ USB GPS/

GLONASS
- Adaptador EasyCap USB con chip STK1160 para la cámara trasera
- Adaptador Bluetooth OBD2
- Cámara de visión trasera
- Hub USB con 4 puertos de 5v/12v con alimentación
- Adaptador ELM327 OBD2

No logre encontrar un marco prefabricado para la pantalla LCD de 7", así que tuve que hacerlo yo mismo. El diseño de marco fue creado en SolidWorks y fue imprimido con una impresora 3D. Después le di masilla, lo lijé y lo pinté de negro mate. Encajaba perfectamente, dejando visible una zona de 152.5x91.5mm. La pantalla se fija al marco con cinta 0.5mm de doble cara.

Elegí una LCD ChalkElec de 7" con una resolución de 1280x800 píxeles y sistema multitáctil de 5 puntos. La pantalla es muy luminosa, con buen contraste y la imagen es claramente visible a la luz del sol. El ODROID-U3 tiene una entrada micro HDMI, al igual que la pantalla de ChalkElec Puesto que este cable no era fácil de conseguir, tuve que hacer yo mismo. El cable del sistema multitáctil está conectado al ODROID-U3 por el puerto USB, con capacidad para un único punto. Para activar el sistema multitáctil de 5 puntos es necesario compilar el kernel con soporte para ChalkBoard Touch. Para ello recurre al artículo "Tablet Gigante" de la edición de febrero de 2014 de la revista.

Fuente de Alimentación inteligente

El ODROID-U3 necesita 5 voltios, hay muchos convertidores de 12V/5V

Componentes del marco LCD

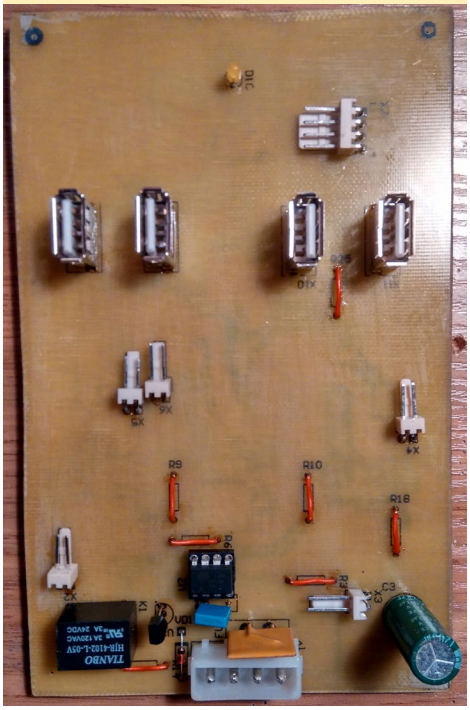
Marco LCD montado

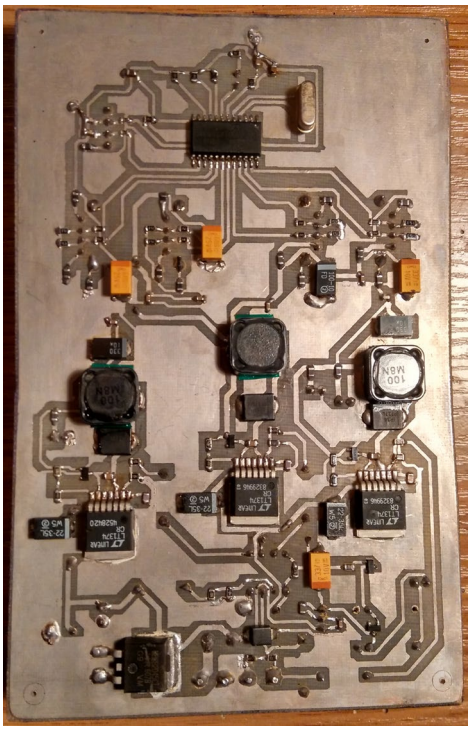
con diversos niveles de potencia disponibles a nivel comercial. No obstante, quería automatizar el encendido y el apagado del CarPC haciendo uso de la posición de la llave de contacto, indicada con la señal ACC del coche, lo cual requería usar una fuente de alimentación adaptada. Elegí el chip LT1374 para el convertidor 12V/5V. Es muy simple y consume muy poca energía en el modo reposo, al mismo tiempo que proporciona corriente de hasta 2A y permite controlar el encendido/apagado. El microcontrolador ATtiny13 fue utilizado para automatizar la fuente de alimentación con la señal de control ACC.

Algoritmo

- En el primer inicio, el microcontrolador lleva y monitoriza la señal ACC.
- Cuando la señal ACC está encendida, los perifericos son alimentados y tras un par de segundos se enciende el

Vista delantera y trasera de la fuente de alimentación inteligente





ODROID-U3.
- Cuando la señal ACC está apagada, se emula la pulsación del botón "Power", ODROID entra en "modo reposo" y se activa el "modo avión".
- Cuando la señal ACC está apagada, tras 1 segundo se emula la pulsación del botón "Power", el ODROID-U3 se despierta y el "modo avión" se apaga.
- cuando la señal ACC esta desconectada durante más de 30 segundos, la batería del coche esta monitorizada de modo que a menos de 11,5V todo se desconecta excepto el microcontrolador.
- Cuando la señal de ACC permanece apagada más de 14 horas, el ODROID-U3 se desconecta.

Hub USB

En la fuente de alimentación también he montado un hub USB. Para poder trabajar a bajas temperaturas durante los inviernos rusos, he elegido el chip industrial AT43301 (ATMEL) que funciona a temperaturas de -40ºC. Para filtrar el ruido en todas las líneas de potencia, he instalado rebordes de ferrita (FB) en el hub USB. Sin ferrita, el chip del hub USB integrado en la placa ODROID-U3 se congelaría de vez en cuando.

Android y el software

La imagen más reciente de Android KitKat 4.4.4 de Hardkernel soporta muchos dispositivos automáticamente, pero no admite ChalkBoard Touch y Easycap, lo que hace necesario tener que compilar un kernel para poder usar estos dispositivos. Sin embargo, hay una solución más simple. Gracias a @voodik, miembro de los foros ODROID, existe la imagen Android 4.4.4 KitKat Cyanogenmod 11.0 que ya soporta ChalkBoard Touch y Easycap y que puede descargarse desde http://bit.ly/1Lc2nWW.

Para que el receptor GPS funcione correctamente, debes especificar la tasa de baudios correcta y el número del puerto USB en el archivo /system/build.prop. En mi ODROID-U3, el puerto es ttyUSB3 y la velocidad es 4800:

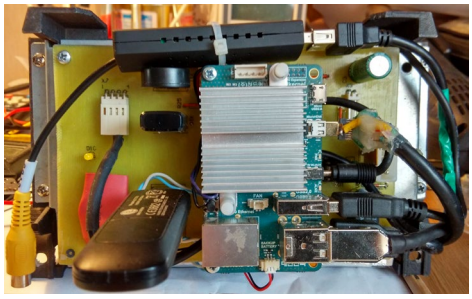
```
ro.kernel.android.gps=ttyUSBx
ro.kernel.android.gps.speed=xxxx
```

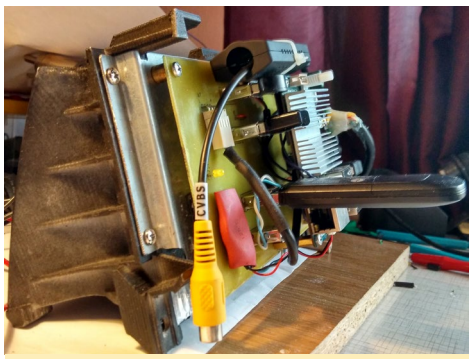
Una vez instalado el sistema, debes instalar la aplicación GoogleApps Installer para utilizar Google Play Store. Para la pantalla de inicio, elegí una aplicación llamada BigLauncher que está diseñado para personas mayores o con dificultades visuales, cuenta con iconos grandes que son muy apropiados para su uso en el coche. Para escuchar música MP3, instalé Power-Amp y para la radio por Internet, PC Radio. Para ver fotos, elegí Quick Pic y para la reproducción de vídeos he usado MX Player. Para conectar el teléfono móvil al CarPC, instalé Tablet Talk y configuré Torque Pro para la comunicación con el ordenador de a bordo del coche utilizando un adaptador OBD2. Para reducir el consumo de energía en modo reposo, también instalé la aplicación Automate It. Tras desactivar la señal ACC, el botón de encendido es emulado por Automate It junto con el cambio a modo avión.

Pantalla de inicio de Android

Instalación

Para montar la pantalla, pegué la pantalla LCD al marco con una fina cinta de doble cara. Luego aseguré la fuente de alimentación y el ODROID-U3 utilizando un soporte de metal adaptado y conecte todos los dispositivos USB. Para la energía, utilice cables de 12V indivi-duales desde la batería del coche a mi CarPC. Después conecté la salida de audio del ODROID-U3 a la entrada AUX de la unidad principal del coche,





Vista del CarPC montado antes de instalarlo en el vehículo

y monte el receptor GPS a un lado del parabrisas.

El CarPC se conecta a Internet a través de un punto de acceso desde mi teléfono móvil. Además, en el caso de que el teléfono se queda sin batería, un dongle USB 3G se conectará al CarPC.

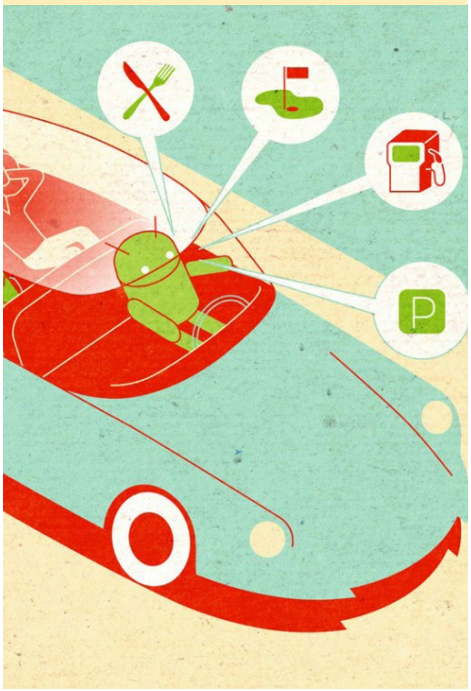
Software

El código para Attiny13 está escrito en CodeVisionAVR y está disponible en http://bit.ly/1RY4G4s. Para montar ATtiny13 con el código del controlador, crea un nuevo proyecto en blanco en CodeVisionAVR. Luego, descarga el código desde http://pastebin.ca/3002845 y cópialo en el nuevo proyecto. Compila el código para producir un archivo firmware HEX, después monta el archivo HEX en Attiny13 utilizando software Arduino.

Usando una placa ODROID-U3, he creado un potente y rápido CarPC, que mejora notablemente las capacidades originales del sistema instalado de fábrica.

La combinación de Android y tu coche es una muy buena y rentable alternativa a los equipos que vienen de serie

Con una car-PC ODROID-U3, serás la envidia de todos tus amigos



OBDGPS LOGGER

COMBINA DATOS GPS Y OBDII PARA CONTAR CON UN COMPLETO SEGUIMIENTO DEL VEHICULO

por Venkat Bommakanti

En el número de mayo de 2015, presenté una potente combinación de un ODROID-C1 con un módulo USB GPS para rastrear vehículos usando OpenGTS. ¿Y si pudiéramos mejorar aún más esa combinación añadiendo un sistema de diagnóstico de vehículos? Pues bien, ¡On-Board Diagnostics (OBDII) hace exactamente eso! En este artículo se describe cómo lograr qué tanto el GPS USB como el adaptador Bluetooth OBDII funciones de forma simultanea con el C1 utilizando el paquete de código abierto OBDGPSLogger, que se puede utilizar con los vehículos fabricados a partir de 1999.

OBDGPSLogger es una solución de línea de comandos basada en C diseñada para ejecutarse sobre Linux con datos de una base de datos SQLite local. No incluye ninguna interfaz nativa de gestión. Sin embargo, para facilitar la gestión de los datos GPS y OBD, usaremos una opción que incluye Nginx como servidor web, PHP como el motor de programación y phpLiteAdmin como interfaz de administración de base de datos SQLite.

Es posible migrar a una solución MySQL o usar utilidades como pyOBD, una utilidad de código abierto basada en Python que permite obtener datos OBD, pero este artículo solo incluye una configuración básica. Consulta el artículo OpenGTS de mayo 2015 para configurar y validar el adaptador USB GPS.

Requisitos

1. Un ODROID-C1, aunque puedes usar un sistema ODROID más potente.
2. Accesorios necesarios para C1: cable HDMI, cable ethernetCAT 5E+ o adaptador wifi 3, adaptador Bluetooth módulo 2 (BT 4.0 +), fuente de alimentación, batería RTC y monitor HDMI, ODROID-VU o pantalla táctil de 3.2”
3. Un módulo eMMC o tarjeta microSD de 16GB+ con la última imagen de escritorio Lubuntu y un lector de tarjetas SD.
4. Un módulo USB GPS de Hardkernel, disponible en http://bit.ly/1EPERhm.
5. Un adaptador Bluetooth OBDII (v1.5 +) con una interfaz ELM327 compatible, como por ejemplo el desarrollado por Panlong.
6. Una red en la que el dispositivo tenga acceso a internet y a los foros ODROID.
7. Software OBDGPSLogger (Versión: 0.16) gpsd y gpsd-cliente.
8. Acceso en red al C1 a través de utilidades como PuTTY, FileZilla, TightVNCViewer (MS Windows 7 +) o Terminal (Mac, Linux), desde un escritorio de pruebas.
9. Pack de baterías LiPo con al menos dos (2) puertos USB que proporcionen 2A+ cada uno: uno para la VU (con la pantalla 3,2” no se utiliza) y otro para el propio C1, junto con los cables de conexión USB-DC compatibles con C1.

Figura I: CI, GPS,OBDII, pantalla táctil y batería montados

Instalar Lubuntu

Instala la última imagen para C1 en el módulo eMMC o tarjeta SD e insértala en el ODROID. Con el monitor conectado arrancar el sistema. Ejecuta ODROID Utility y ajusta la resolución de pantalla, luego reiniciar el sistema.

Expande la partición de instalación para utilizar todo el espacio de la unidad disponible seleccionando la opción “ Resize your root partition “. Reinicia y vuelve a ejecutar ODROID Utility para configurar y actualizar todos los aspectos relevantes del sistema y reinicia de nuevo.

Asegúrate de iniciar sesión con el usuario odroid por defecto, a menos que se especifique lo contrario. Escribe los siguientes comandos en una ventana de terminal para actualizar los archivos del sistema operativo, Kernel del sistema y las aplicaciones relacionadas:

```
$ sudo apt-get autoremove && sudo apt-get update
$ sudo apt-get dist-upgrade && sudo apt-get upgrade
$ sudo apt-get install linux-image-c1
```

Apaga el ODROID, conecta todos los accesorios y cables incluyendo los adaptadores GPS y Bluetooth, luego reinicia. Comprueba la versión del sistema desde una ventana de terminal con el siguiente comando para asegurarte de tener la versión más reciente:

```
$ uname -a
Linux odroid 3.10.75-84 #1 SMP PREEMPT Sat Apr 25 18:33:08 BRT 2015 armv7l armv7l armv7l GNU/Linux
```

Configurar Bluetooth

Instala las utilidades complementarias usando los siguientes comandos:

```
$ sudo apt-get install bluez-dbg bluez-hcidump bluez-utils bluez-tools
$ sudo apt-get install bluewho blueman python-bluetooth
```

Luego, comprueba la información USB relacionada con los adaptadores:

```
$ lsusb
…
Bus 001 Device 005: ID 0b05:17cb ASUSTek Computer, Inc.
Bus 001 Device 004: ID 1546:01a6 U-Blox AG
…
```

Ten en cuenta que he usado un adaptador ASUS Bluetooth 4.0 compatible con ODROID en lugar del modelo de Hardkernel. Comprueba la compatibilidad del adaptador Bluetooth para funciones como el protocolo RFCOMM analizando los archivos logs dmesg :

```
$ dmesg | grep Blue
```

```
[    0.851848@0] Bluetooth: Core ver 2.16
[    0.859721@0] Bluetooth: HCI device and connection manager initialized
[    0.866240@0] Bluetooth: HCI socket layer initialized
[    0.871245@0] Bluetooth: L2CAP socket layer initialized
[    0.876447@0] Bluetooth: SCO socket layer initialized
[    1.429422@2] Bluetooth: HCI UART driver ver 2.2
[    1.433876@2] Bluetooth: HCI H4 protocol initialized
[    1.438828@2] Bluetooth: HCI BCSP protocol initialized
[    1.443919@2] Bluetooth: HCILL protocol initialized
[    1.448782@2] Bluetooth: HCI-ATH3K protocol initialized
[    1.453877@2] Bluetooth: HCI Three-wire UART (H5) protocol initialized
[    3.236424@2] Bluetooth: bt-wake_control_init Driver Ver 1.1
[    3.366366@2] Bluetooth: RF-COMM TTY layer initialized
[    3.371156@2] Bluetooth: RF-COMM socket layer initialized
[    3.376392@2] Bluetooth: RF-COMM ver 1.11
[    3.380308@2] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[    3.385744@2] Bluetooth: BNEP filters: protocol multicast
[    3.397895@2] Bluetooth: BNEP socket layer initialized
[    3.402975@2] Bluetooth: HIDP (Human Interface Emulation) ver 1.2
[    3.409060@2] Bluetooth: HIDP socket layer initialized
```

Comprueba los módulos Bluetooth instalados:

```
$ dpkg -l | grep blue
ii  blueman                ... armhf  Graphical bluetooth manager
```

```
ii  bluez                  ... armhf  Bluetooth tools and daemons
ii  bluez-alsa:armhf       ... armhf  Bluetooth ALSA support
ii  bluez-cups             ... armhf  Bluetooth printer driver for CUPS
ii  libbluetooth-dev       ... armhf  Dev.  files for BlueZ Linux Bluetooth lib
ii  libbluetooth3:armhf    ... armhf  Library to use the BlueZ Linux BT stack
ii  libgnome-bluetooth11   ... armhf  GNOME Bluetooth tools - support library
```

Comprueba la presencia del dispositivo Bluetooth, lo cual será muy útil para configurar las conexiones:

```
$ hcitool dev
Devices:
    hci0    00:02:72:CC:F4:CE

$ hciconfig
hci0:    Type: BR/EDR  Bus: USB
    BD Address: 00:02:72:CC:F4:CE ACL MTU: 1021:8  SCO MTU: 64:1
    UP RUNNING PSCAN
    RX bytes:583 acl:0 sco:0 events:33 errors:0
    TX bytes:898 acl:0 sco:0 commands:33 errors:0

$ sudo rfkill list all
0: hci0: Bluetooth
    Soft blocked: no
    Hard blocked: no
```

Configurar Bluetooth

Desde el escritorio de Lubuntu, inicia "Bluetooth Manager" (Figura 2). A continuación, selecciona el elemento de menú "Preferences" para configurar la aplicación (figura 3). Y cambia el nombre del adaptador Bluetooth por algo significativo como "c1-1-0"(figura 4). Haz que el dispositivo sea visible para otros dispositivos Bluetooth.

Guarda la configuración y reinicia.

Figura 2 – Iniciar Bluetooth Manager

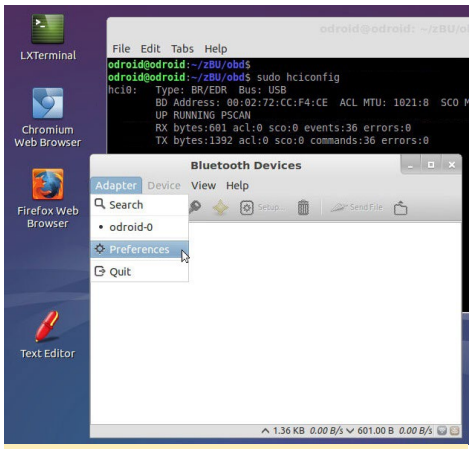


Figura 3 - Configurar Bluetooth Manager

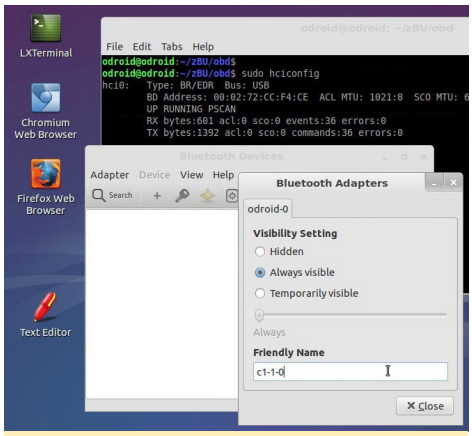


Figura 4 – Actualizar configuración de Bluetooth Manager

Probar Bluetooth

Si fuera necesario, puede usar el analizador de paquetes Bluetooth por línea de comandos para analizar el tráfico del Bluetooth y así depurar posibles errores:

```
$ sudo hcidump
HCI sniffer - Bluetooth packet
analyzer ver 2.5
device: hci0 snap_len: 1500 fil-
ter: 0xffffffff
< HCI Command: Write Class of De-
vice (0x03|0x0024) plen 3
      class 0x700100
> HCI Event: Command Complete
(0x0e) plen 4
      Write Class of Device
(0x03|0x0024) ncmd 1
      status 0x00
< HCI Command: Write Extended
Inquiry Response (0x03|0x0052)
plen 241
      fec 0x00
      Complete local name: 'c1-1-
0'
      TX power level: 0
      Complete service class-
es: 0x112d 0x1112 0x111f 0x111e
0x110c 0x110e 0x1105
> HCI Event: Command Complete
(0x0e) plen 4
      Write Extended Inquiry Re-
sponse (0x03|0x0052) ncmd 1
      status 0x00
< HCI Command: Write Extended
Inquiry Response (0x03|0x0052)
plen 241
      fec 0x00
      Complete local name: 'c1-1-
0'
      TX power level: 0
      Complete service class-
es: 0x112d 0x1112 0x111f 0x111e
0x110c 0x110e ...
> HCI Event: Command Complete
(0x0e) plen 4
      Write Extended Inquiry Re-
sponse (0x03|0x0052) ncmd 1
      status 0x00
...
```

Otra herramienta útil es Wireshark que proporciona una perspectiva gráfica de datos escaneados, se pueden instalar escribiendo el siguiente comando en cualquier ventana de terminal:

```
$ sudo apt-get install wireshark
```

Tras la instalación, puede iniciar Wireshark versión 1.10.6 siguiendo los pasos de la Figura 5. Deberías ver una pantalla de bienvenida como la de la figura 6, que indica que el adaptador Bluetooth ha sido detectado por el C1.

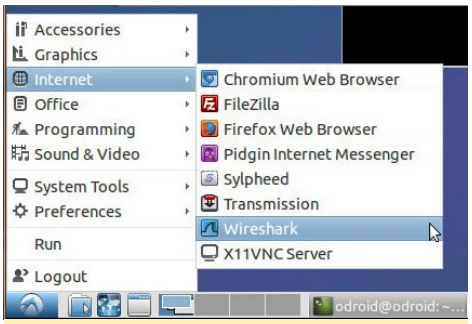
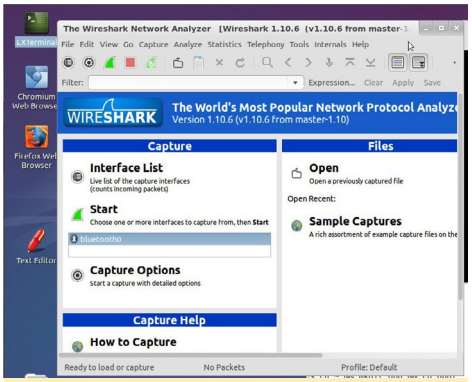


Figura 5 - Iniciar Wireshark



Pantalla de bienvenida de Wireshark con el adaptador Bluetooth detectado

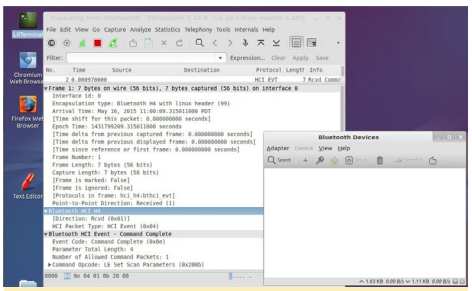


Figura 7 - Wireshark con tráfico Bluetooth escaneado

Haga clic en el icono de aleta de tiburón verde para iniciar una captura. Luego, inicia la aplicación Bluetooth Manager y haz clic en el botón Search una vez que la aplicación haya cargado. Deberías ver al instante todo el tráfico Bluetooth y la información relacionada debería aparece en la a aplicación Wireshark, como muestra la Figura 7.

Instalar gpsd

Instala gpsd y las utilidades pertinentes utilizando el siguiente comando. Si deseas información detallada sobre su configuración, testeo y uso, consulta el artículo OpenGTS de mayo 2015.

```
$ sudo apt-get install gpsd gpsd-clients && sudo reboot
```

Instalar herramientas del servidor web

OBDGPSLogger almacena los datos OBDII y GPS en una base de datos local SQLite3. No incluye una herramienta de gestión de base de datos. Para facilitar la gestión de la base de datos, he incluido la instalación del servidor web nginx y PHP junto con phpLiteAdmin con el fin de disponer de una interfaz gráfica de administración de SQLite3. En primer lugar instala lo siguiente:

```
$ sudo apt-get install nginx-full sqlite3
$ sudo apt-get install autoconf automake autotools-dev libtool curl
$ sudo apt-get install libcurl4-openssl-dev lbzip2
$ sudo apt-get install php5 php5-dev php5-cgi php5-fpm php5-curl php5-gd
$ sudo apt-get install php5-sqlite php5-gmp php5-imagick php5-imap php5-intl
$ sudo apt-get install php5-ldap php5-mcrypt libmcrypt-dev php-xml-parser
$ sudo apt-get install php5-xsl php-apc
```

Luego, actualiza la configuración de nginx para habilitar soporte para PHP5:

```
$ sudo cd /etc/nginx/sites-available
$ sudo cp default default-orig
$ sudo medit default
```

Actualiza la configuración por defecto:

```
...
# our php-handler - add this
upstream php-handler {
    server unix:/var/run/php5-fpm.sock;
}

server {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;

    root /usr/share/nginx/html;

    # try php file execution first
    index index.php index.htm;

    # Make site accessible from http://localhost/
    server_name <your-C1's-ip-address>;

    # set max upload size
    client_max_body_size     10G;
    fastcgi_buffers          64 4K;
    client_body_buffer_size 2M;

    # setup calendar, contact, webdav options
    rewrite ^/caldav(.*)$ /remote.php/caldav$1 redirect;
    rewrite ^/carddav(.*)$ /remote.php/carddav$1 redirect;
    rewrite ^/webdav(.*)$ /remote.php/webdav$1 redirect;

    location = /robots.txt {
         allow all;
         log_not_found off;
         access_log off;
    }

    location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ index.php;

        # The following 2 rules are only needed with webfinger
        rewrite ^/.well-known/host-meta /public.php?service=host-meta last;
        rewrite ^/.well-known/host-meta.json /public.php?service=host-meta-json last;
        rewrite ^/.well-known/carddav /remote.php/carddav/ redirect;
        rewrite ^/.well-known/caldav /remote.php/caldav/ redirect;
        rewrite ^(/core/doc/[^\/]+/)$ $1/index.html;
    }

    # redirect server error pages to the static pages
    error_page 404 /404.html;
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
       root /usr/share/nginx/html;
    }

     # pass the PHP scripts to FastCGI server listening on fpm-socket
     location ~ \.php(?:$|/) {
        fastcgi_split_path_info ^(.+\.php)(/.+)$;
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        # $fastcgi_path_info parse fails in latest php5-fpm. disable it.
        # fastcgi_param PATH_INFO $fastcgi_path_info;
        fastcgi_pass php-handler;
        fastcgi_read_timeout 600;
     }
}
...
```

Actualice la configuración de php5-fpm usando los siguientes comandos:

```
$ cd /etc/php5/fpm/pool.d/
$ sudo cp www.conf www.conf-orig
$ sudo medit www.conf
```

Añade la siguiente configuración para que coincida con la configuración de nginx:

```
...
listen = /var/run/php5-fpm.sock

Enhance file execution security by
setting the following flags in the
php5 config file:
$ sudo medit /etc/php5/fpm/php.
ini

Set these options:
    cgi.fix_pathinfo=0
    display_errors = On
    display_startup_errors = On
    output_buffering = 4096
    default_socket_timeout = 600
...
```

Tras modificar cada archivo guarda los cambios. Luego, descarga phpLiteAdmin versión 1.9.5 desde http://bit.ly/1HHIJAJ y descomprimelo:

```
$ cd ~/obd && mkdir pla && cd pla
$ mv ~/Downloads/phpliteAdmin_v1-
9-5.zip .
$ unzip phpliteAdmin_v1-9-5.zip
```

Consulta el archivo README para ver los consejos de configuración:

```
$ cat README.txt
```

Actualiza config phpLiteAdmin con los datos según tu configuración:

```
$ cp phpliteadmin.config.sample.
php phpliteadmin.config.php
$ medit phpliteadmin.config.php
```

Cambia los siguientes datos:

```
...
    $password = 'odroid';
    $directory = '/home/odroid/
obd/obdgpslogger/data';

    $databases = array(
       array(
                'path'=> 'obdg-
pslogger.db',
                'name'=> 'OBDII-
GPS Logger'
       ),
    )
...
```

A continuación, mueve la aplicación de administración y los archivos de configuración a la ubicación adecuada, ajusta sus privilegios de ejecución. Reinicia tras guardar el archivo.

```
$ sudo cp phpliteadmin.config.php
/usr/share/nginx/html
$ sudo cp phpliteadmin.php /usr/
share/nginx/html
$ chmod 755 phpliteadmin.php

$ cd /usr/share/nginx/html
$ ls -lsa
...
  4 -rw-r--r-- 1 root root   537
Mar  4  2014 50x.html
  4 -rw-r--r-- 1 root root   612
Mar  4  2014 index.html
  4 -rw-r--r-- 1 root root   2691
May 17 10:49 phpliteadmin.config.
php
220 -rwxrwxr-x 1 root root 222859
May 17 10:49 phpliteadmin.php
```

Compilar OBDGPSLogger

Instala los prerrequisitos adicionales para OBDGPSLogger:

```
$ sudo apt-get install libx11-
dev libxft2-dev libgps-dev zlibc
zlib1g zlib1g-dev
$ sudo apt-get install libfltk1.3-
dev fluid libftdi1 libftdi-dev
subversion
```

Descarga el código OBDGPSLogger utilizando los comandos:

```
$ cd ~ && mkdir obd && cd obd/
$ svn co svn://svn.icculus.org/
obdgpslogger/trunk obdgpslogger
$ cd obdgpslogger/obd/obdgpslog-
ger/src/logger
```

El código OBDGPSLogger necesita ser modificado para que pueda trabajar con los últimos servicios gpsd. Aplica el siguiente parche al código fuente:

```
~/obd/obdgpslogger/src/logger/
gpscomm.c

29c29,32
<     struct gps_data_t *g = gps_
open(server,port);
---
>     int rc;
>     struct gps_data_t *g =
NULL;
>
>     g = malloc(sizeof(struct
gps_data_t));
31a35,40
>
>     rc = gps_open(server,port,
g);
>     if(rc != 0) {
>      free(g);
>      return NULL;
>     }
61c70
<             gps_poll(g);
---
>             gps_read(g);
```

Ahora está listo para ser compilado. Reanuda el proceso con los siguientes comandos:

```
$ cd ~/obd/obdgpslogger
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

Verifica la instalación utilizando el comando:

```
$ obdgpslogger -v
Version: 0.16
```

Conectar el C1 y el adaptador OBDII

Apaga el ODROID, aparca el coche en un lugar seguro y apaga el motor. Conecta el adaptador OBDII al puerto correcto de su coche, que normalmente se encuentra bajo el volante, cerca del freno o del acelerador. Déja que se active el proceso de inicialización. A continuación, coloca el C1 en el asiento del pasajero. Se recomienda colocarlo a una distancia máxima de unos 3 metros desde el adaptador OBDII (cuanto más cerca, mejor).

No todos los adaptadores OBDII son compatibles con ODROIDs. Antes de comprar uno, investiga bien el adaptador para asegurarte de que sea compatible con Linux. Me arriesgué con mi

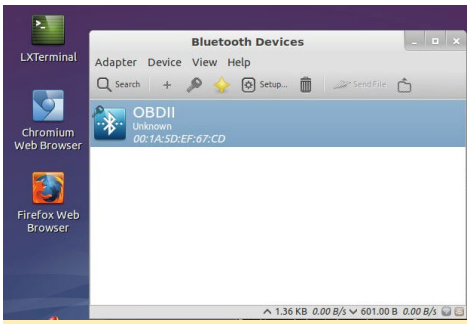


Figura 8 - Adaptador OBDII detectado por el ODROID-C1

adaptador de 10$ y tuve suerte.

Arranca el coche, enciende el C1 e inicia la aplicación Bluetooth Manager. Haz clic en el botón "Search" para buscar el adaptador OBDII. Tras unos momentos, deberías ver el resultado que se muestra en la Figura 8.

Si no se detecta, retira e inserta el adaptador OBDII un par de veces y vuelva a repetir la búsqueda. Si el error persiste, puedes probar la conectividad por Bluetooth con un smartphone. Una vez detectado, selecciona la conexión por el puerto serie como se muestra en

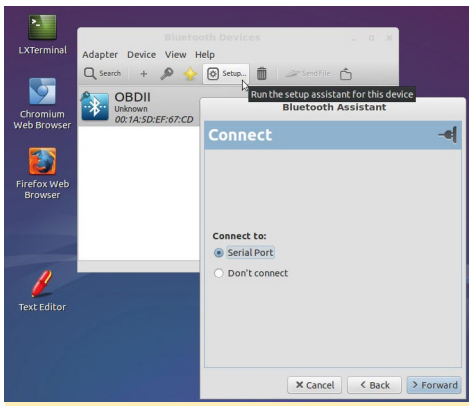


Figura 9: Configuración del puerto serie

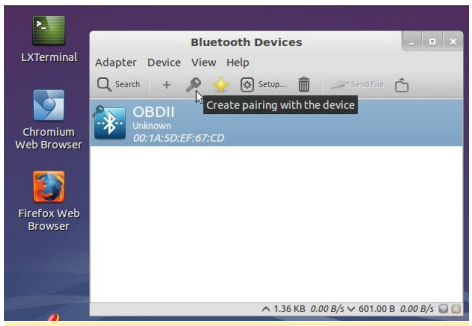


Figura 10 - Conexión del C1 y el adaptador OBDII usando Bluetooth

la Figura 9. Intenta vincular el C1 y el adaptador OBDII haciendo clic en el botón Key - Figura 10.

Se te pedirá una clave de 4 dígitos para completar el proceso de emparejamiento, debería estar disponible desde el proveedor del adaptador OBDII. Podría ser una de las más comunes, tales como 0000 o 1234. Investiga por Internet si el proveedor no te proporciona la clave.

Para garantizar que la aplicación OBDGPSLogger funciona correctamente, comprueba si el perfil del dispositivo USB se ha creado en el C1. La presencia de estas dos entradas indica que el sistema está funcionando correctamente.

```
$ ls -lsa /dev/rf*
0 crw-rw---- 1 root dialout 216,
0  Dec 31 16:22 /dev/rfcomm0
0 crw-r--r-- 1 root root     10,
63 Dec 31 16:00 /dev/rfkill
```

Obtener y trazar los datos

Crea un marcador de posición para capturar los datos. Después, inicia la aplicación OBDGPSLogger asegurándote de especificar el puerto serie. Revisa la salida para garantizar que OBDGPSLogger esta listo para capturar correctamente datos OBD y GPS:

```
$ cd ~/obdgpslogger
$ mkdir data && cd data

$ sudo$  obdgpslogger -s /dev/
rfcomm0
Opening serial port /dev/rfcomm0,
this can take a while
Successfully connected to serial
port.  Will log obd data
Successfully connected to gpsd.
Will log gps data
Creating a new trip
GPS acquisition complete
```

Ahora, coge tu configuración C1 y da una vuelta a la manzana. El OBDGPSLogger debería capturar los datos del vehículo y almacenarlos en la base de datos SQLite. Tras finalizar el recorrido, salte de la aplicación OBDGPSLogger pulsando Control + C, Luego, vuelve a casa para examinar el trazado. Exportar los datos recogidos en formato KML para ser utilizados con Google Maps y Google Earth:

```
$ cd ~/obdgpslogger/data
$ obd2kml -d obdgpslogger-vb.db
-o obdgpslogger-vb.kml
```

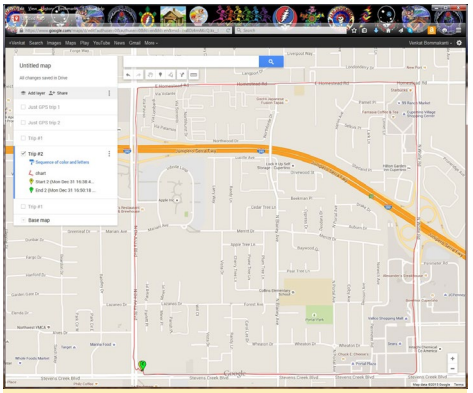


Figura 11 - datos KML en Google Maps

```
$ ls -lsa *.kml
276 -rw-rw-r-- 1 odroid odroid
278707 May 16 22:06 obdgpslogger-
vb.kml
```

Despues, sube el archivo KML a Google Maps (Figura 11).

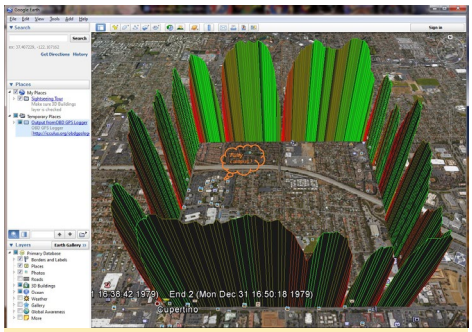


Figura I2 - datos KML en Google Earth

- Inicia sesión en tu cuenta de Google y ve a http://maps.google.com
- Haz clic en Mis mapas
- Haga clic en Crear un nuevo mapa
- Añade un título y una descripción
- Haz clic en Importar
- Haz clic en Seleccionar archivo, selecciona el .kml para cargarlo y luego, haz clic en Cargar archivo

A continuación, instala Google Earth en tu sistema Windows 7+ o Macintosh. Pasa el archivo .kml desde el C1 a tu sistema con Google Earth. Importa el archivo .kml y visualízalo como muestra la Figura 12.

Ten en cuenta que, mientras que los datos GPS se usan para trazar el recorrido, los datos OBD se utilizan para trazar la eficiencia del consumo de combustible del vehículo en millas por galón. Las partes rojas corresponden a la baja eficiencia del consumo de combustible que va desde 0 mph a un máximo de 35 mph. Las partes verdes corresponden a los segmentos eficientes. Los descensos corresponden a las paradas del coche en todas las señales de stop.

¿Reconoces el campus "con olor a fruta" famoso en todo el mundo rodeado con un circulo? ¡Hay una pista en la parte inferior izquierda de la imagen!

Los datos también se pueden exportar a formato .csv e importarlo en MS Excel o a una aplicación compatible para ver los datos numéricos. La Figura 13 muestra el formato de datos .csv.

```
$ obd2csv -d obdgpslogger-vb.db
-o obdgpslogger-vb.csv
$ ls -lsa *.csv
112 -rw-rw-r-- 1 odroid odroid
111969 May 16 22:39 obdgpslogger-
vb.csv
```

Gestionar base de datos SQLite

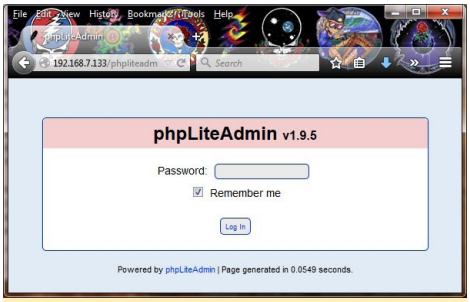


Figura I4 – Conexión a phpLiteAdmin

Siempre es útil tener acceso a los datos en bruto recogidos por OBDGPSLogger, se pueden analizar para depurar errores, o para aprender más sobre su diseño y poder usar el más poderoso sistema de base de datos MySQL. Una vez completado el proceso de instalación descrito anteriormente, puedes dirigir tu navegador a http: // <dirección-IP-C1> para iniciar la aplicación phpLiteAdmin.

Utiliza la contraseña odroid para iniciar sesión en la aplicación. Aparecerá una pantalla de bienvenida como se muestra en la Figura 15. Selecciona las tablas del menú de la izquierda conforme a las figuras 16 y 17. Ahora, tienes todo lo necesario para mejorar las herramientas de código abierto comentadas o desarrollar las tuyas propias.

Figura I3 - Datos CSV

obdgpslogger-vb.csv - OpenOffice Calc

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	obd.temp	obd.rpm	obd.vss	obd.maf	obd.throttlepos	obd.time	obd.trip	obd.ecu	(7.107*obd.vss/obd.maf) as mpg	gps.lon	gps.lat	gps.alt	trip.tripid
2	88	751.5	5	8.18	16.862745	315535124	2	0	4.344132	0	0	0	2
3	88	860.75	5	4.12	16.470589	315535126	2	0	8.625	-122.031818	37.32318	71.3	2
4	88	739	5	4.09	16.078432	315535127	2	0	8.688264	-122.031801	37.32319	70.9	2
5	88	696.75	3	3.29	16.078432	315535128	2	0	6.480547	-122.031789	37.323197	70.8	2
6	88	602.25	3	4.35	16.862745	315535130	2	0	4.901379	-122.031789	37.323214	71	2
7	88	744.5	3	4.32	16.862745	315535131	2	0	4.935416	-122.031789	37.323214	70.9	2
8	88	732	0	4.26	16.862745	315535132	2	0	0	-122.03179	37.32321	70.6	2
9	88	697.5	0	4.25	16.862745	315535134	2	0	0	-122.031793	37.3232	70.5	2
10	88	724	0	3.6	16.078432	315535135	2	0	0	-122.031796	37.323188	70.3	2
11	89	826.5	0	3.95	16.470589	315535136	2	0	0	-122.0318	37.323182	70.7	2
12	89	703	0	5.87	18.039215	315535138	2	0	0	-122.0318	37.32318	70.8	2
13	89	1045.25	0	7.82	19.215687	315535139	2	0	0	-122.031804	37.323184	70.8	2
14	89	1192.75	6	11.51	20.784313	315535140	2	0	3.704778	-122.031811	37.323183	70.7	2
15	89	1434.25	8	6.34	17.647058	315535141	2	0	8.967823	-122.031834	37.323189	70.9	2
16	88	1246	13	3.6	16.078432	315535143	2	0	25.664167	-122.031854	37.323202	71.1	2
17	88	848.25	12	3.54	18.039215	315535144	2	0	24.091526	-122.031885	37.323213	70.8	2
18	88	830.25	7	3.23	16.078432	315535145	2	0	15.402167	-122.031907	37.323227	70.7	2
19	88	749	6	3.32	16.078432	315535147	2	0	12.843976	-122.031938	37.323238	70	2
20	88	830.25	7	4.25	16.862745	315535148	2	0	11.705647	-122.031949	37.323254	69.8	2

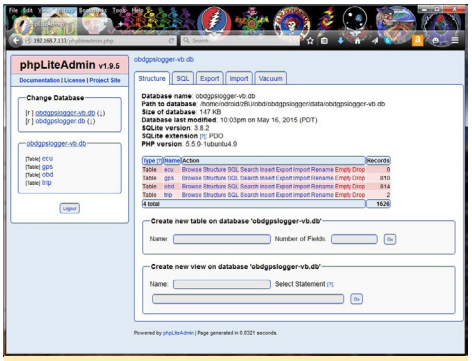


Figura I5 – Pantalla de bienvenida de phpLiteAdmin

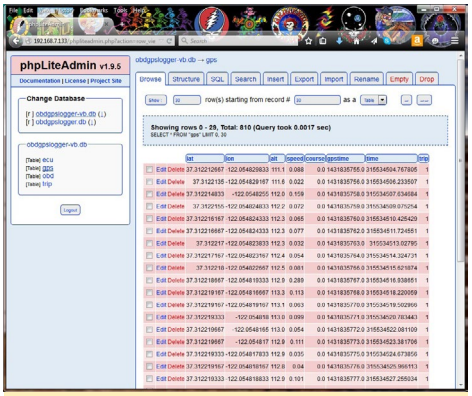


Figura I6 - Datos GPS

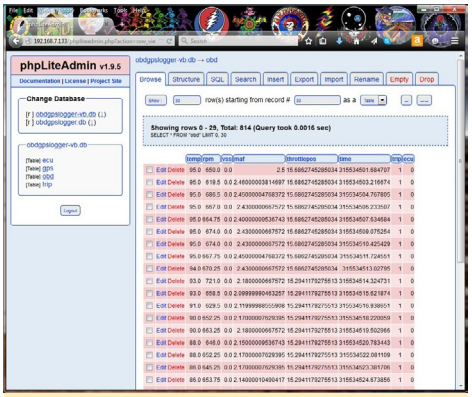


Figura I7 - Datos OBD

Descargo de Responsabilidad

Como siempre, ten cuidado al manipular la configuración en tu vehículo. Garantiza primero la seguridad antes de realizar cualquier actividad de configuración. HardKernel y los colaboradores de estos artículos no se hacen responsables de los posibles accidentes que puedan producirse durante tus experimentos.

Agradecimientos

Gary Briggs (chunky@icculus.org), el autor de OBDGPSLogger, ha consentido amablemente usar su software para publicar este artículo. Gracias a Gary en nombre de la comunidad ODROID.

Recursos Adicionales

- Post del foro ODROID: http://bit.ly/1Eu8HTB
- Pagoma web OBD GPS: http://bit.ly/1AuOe66
- Entorno de trabajo para el analisis de datos ROOT: http://bit.ly/1Sztekz
- Introducción a la programación Bluetooth: http://bit.ly/1AuOfqw
- PHPLiteAdmin: http://bit.ly/1dtZUw0
- Instalar Google Earth en Linux: http://bit.ly/1FMzGz7
- Referecia PyOBD: http://bit.ly/1JSnbnt
- Adaptador Bluetooth: http://amzn.to/1FMzJeq
- Adartador OBDII: http://amzn.to/1AuOh1V

FOROS ODROID

EL LUGAR PERFECTO PARA COMUNICARTE CON DESARROLLADORES HARDKERNEL

por Rob Roy

Los foros ODROID han sido el lugar de encuentro para la creciente comunidad Hardkernel durante varios años, con más de 11.000 miembros a junio de 2015. Se puede discutir sobre ODROIDs con Mauro, el principal desarrollador del kernel Linux y Justin, el CEO de Hardkernel junto con un equipo de desarrolladores cada vez mayor que donan su tiempo para ayudarte a sacarle el máximo partido a tu ODROID. ¡Compruébalo tu mismo en http://forum.odroid.com!

JUEGOS LINUX: EMULACION NINTENDO 64 - PARTE 2

EMBARCATE EN EL MEJOR VIAJE CON LOS JUEGOS DE LOS 90

por Tobias Schaaf

En la Parte 1 de este artículo presenté la última versión del emulador de Nintendo 64 para Linux y comparé su rendimiento en todas las placas ODROID actuales. En esta segunda parte vamos a echar un vistazo a algunos de los más populares juegos de Nintendo 64, como son Mario Kart, Mario Party, Paper Mario, Star Fox, Star Wars, Starcraft, Super Mario, Super Smash Bros y Legend of Zelda.

Mario Kart 64

Mario Kart es una de las franquicias más conocidas de Nintendo como juego de carreras, protagonizado por los más famosos personajes de Nintendo como son Mario, Luigi, Peach, Yoshi, Donkey Kong, Bowser entre otros. Una de las grandes ventajas de este juego es que se puede jugar con hasta 4 jugadores al mismo tiempo.

Realmente no soy un fan de esta serie, especialmente de la versión de Nintendo 64 que en mi opinión es un poco pobre a nivel gráfico. Aunque N64 es conocida por su potencial 3D, Mario Kart 64 utiliza en su mayor parte elementos 2D que no se ven muy bien. Los únicos elementos en 3D del juego son el terreno por el cual se conduce y algunos obstáculos y puentes, lo que hace que el juego no sea muy atractivo.

U3

Cuando ejecute el juego por primera vez sin frameskip, iba muy lento. Ya que el juego utiliza mayoritariamente elementos en 2D, me pregunté realmente por qué este juego necesitaba tanta potencia de CPU. Sin embargo, una vez que activé la opción frame skipping, funcionó muy bien en el U3. Hay una pequeña demora en el sonido mientras se usa el menú, pero nada que realmente moleste. Las carreras funciona bien sin demoras ni retrasos, y el modo multijugador con varios mandos también funciona perfectamente.

CI

Aunque el menú es lento, la experiencia de juego es buena y parece funcionar a toda velocidad usando el plugin Rice. Definitivamente es jugable, aunque se obtiene una mejor experiencia en el U3 o en XU3. Cuando volví a probarlo usando glide64mk2, el juego funcionó muy bien, aunque tenía algunos problemas con las sombras y las texturas del terreno.

XU3

Mario Kart 64 no tiene problemas en el XU3. Se ejecuta a toda velocidad y podía controlarlo fácilmente con un mando Xbox 360.

ODROID MAGAZINE 17

Mario Party

Mario Party es el típico juego de mesa en el que se juega con o contra un máximo de 4 jugadores a diferentes tipos de mini-juegos. El juego es bastante divertido, aunque a veces he tenido dificultades para averiguar los controles de ciertos mini-juegos. Probablemente sea un juego adecuado para cualquier edad, desde niños hasta adultos como juego de mesa o simplemente para pasar un buen rato.

U3

La experiencia con el U3 es perfecta, el juego se ejecuta a toda velocidad. Una vez observe un parpadeo en la pantalla dividida de un mini-juego, pero en el momento en el que empezó la acción, desaparecido. Se puede decir que es un juego totalmente jugable.

CI - plugin rice

El menú es un poco lento al principio, pero cuando se accede al mapa para seleccionar un juego, se percibe un movimiento más fluido, similar al Mario Kart. Sin embargo, cuando intenté iniciar un juego sólo vi una pantalla en blanco. Podía escuchar todo lo que se ejecutaba en segundo plano, y al hacer clic en los botones podía oír las acciones que provocaban, pero no podía ver nada, sólo una pantalla en blanco. Cuando lo intenté de nuevo con un modo de juego diferente, pude ver algo pero faltaban las partes más importantes, y en el momento de iniciar el mini-juego, sólo apareció una pantalla en negro.

CI - glide64mk2

Mientras que el juego no se ejecutaba usando el plugin rice, sí que lo hizo con glide64mk2, aunque un poco lento. La mayoría de las escenas de juego se ejecutan a una velocidad aceptable, de modo que considero que este juego es jugable con glide64mk2 a 16 bits.

XU3

No tuve ningún problema a la hora de ejecutar este juego con XU3. Podía jugar sin problemas, lo cual no me sorprendió teniendo en cuenta que también funcionaba en el U3. En general, la experiencia de juego era bastante buena.

Paper Mario

Paper Mario es una mezcla entre un juego de plataformas como Super Mario y un juego RPG como Final Fantasy. Tiene buenos gráficos y aunque el mundo está en 3D, el propio Mario sólo aparece en 2D. Es en realidad una figura de papel. El modo de juego es muy singular y divertido, es difícil de describir, ¡Sin duda debes probarlo!

U3

La experiencia de Paper Mario sobre U3 es realmente buena. La velocidad en general es muy buena y disfrute bastante con él.

ODROID MAGAZINE 18

C1 - plugin rice

La experiencia en el C1 es difícil de describir. Al principio, el juego no funcionaba para nada. Tras una lenta introducción, el menú principal no aparecía. Después de unos 10 o 15 minutos, parece mostrase otra introducción que básicamente es una imagen de fondo desplazándose. Tras otros 10 o 20 minutos, la imagen cambió de nuevo y de repente apareció el menú de inicio. Guarde el juego y lo volví a iniciar. Una vez más, apareció una sola imagen de fondo. Parece que el juego no funciona para nada en el C1 o podría necesitar horas para iniciarse. El C1 debería ser capaz de ejecutar el juego a una velocidad aceptable pero lamentablemente, el poco soporte gráfico y los drivers defectuosos evitan que el sistema funcione correctamente.

C1 - glide64mk2

Este juego funciona con glide64mk2 a toda velocidad. De forma similar al U3, tiene problemas con las sombras y las texturas del terreno pero aparte de eso, el juego se ejecuta muy bien.

XU3

El núcleo libretro hizo un buen trabajo con este juego. No se apreciaba ninguno de los problemas que aparecerían con el U3 y glide64mk2. Las sombras eran perfectas, los bocadillos estaban bien y se podía leer lo que las estrellas iban diciendo. La velocidad en general era perfecta. Me encanto jugar a este juego en el XU3.

Star Fox 64

Star Fox 64 es un remake del juego Star Fox para Super Famicom/SNES, que fue uno de los primeros shooters espaciales en 3D. La versión de N64 fue famosa por sus buenos gráficos y sobre todo por sus interpretaciones de voz. Las comunicaciones a menudo divertidas de tus compañeros a través de la radio, las intensas batallas y los buenos gráficos hacen que este juego sea muy divertido.

U3

El juego funciona muy bien en el U3. Tiene algunas ralentizaciones en el mapa de la galaxia donde se selecciona la misión, y las sombras son demasiado oscuras. La iluminación no funciona correctamente lo que significa que algunas escenas son muy oscuras. A parte de esto, el juego funciona perfectamente.

C1 - plugin rice

El C1 va bien con este juego. El plugin de vídeo rice hace mejor trabajo cuando representa las sombras que el glide64mk2 en el U3, de modo que las escenas no son tan oscuras. Por otro lado, el rendimiento del C1 es más lento que el U3, y el informe de la misión se demora un poco. Mientras que con el U3 experimentas una ralentización en el mapa de la galaxia donde seleccionas tu misión, el C1 funciona pesimamente. No obstante esto sólo afecta a la selección de la misión y no a todo el juego. Cuando están en plena cacería y continuamente disparando, el juego se ejecuta a toda velocidad y sin problemas, Realmente es posible jugar en el C1

ODROID MAGAZINE 19

C1 - glide64mk2

Como el U3, la experiencia de juego es bastante buena. Se ejecuta más o menos a la misma velocidad que en el U3 y tiene los mismos problemas con la sombra pero aparte de eso, la experiencia de juego es buena y sólo se ralentiza en el mapa de la galaxia.

XU3

Como de costumbre, la experiencia XU3 es el mejor. El juego se ejecuta sin problemas, aunque se ralentiza en el mapa de la galaxia. Los gráficos se ven muy bien y se ejecuta sin problemas.

Star Wars Episode I – Racer

Jugué a este juego hace muchos años en el PC con mi tarjeta gráfica 3DFX Voodoo, que utilizaba el "glide" que está incluido en algunos de los plugins gráficos para Mupen64Plus. El juego se centra en los Pod Racer del Episodio 1 de Star Wars. Es un juego de carreras muy rápido con buenos gráficos y objetos destructibles, puedes modificar tu vaina de carrera para que sea más rápida o más fácil de manejar.

Este juego de hecho utiliza el pak de expansión de memoria de la N64 que mejora los gráficos y el Rumble Pak también es compatible. Sin embargo, la versión para N64 no puede compararse con la versión para PC en términos de gráficos, y también falta el modo multijugador, aun así sigue siendo un buen juego de carreras.

U3

La experiencia en el U3 es muy buena. El juego funciona con fluidez y rapidez, y no parece tener problemas técnicos. Algunas sombras son demasiado oscuras, pero es algo que sólo experimentas en el menú.

C1 - plugin rice

Una vez más, el C1 tiene problemas con este juego, relacionados con el plugin de video rice, los mismos problemas están presente en el U3 cuando se cambia al plugin rice. La imagen se distorsiona y se corta en algunas escenas. El juego funciona perfectamente y sin problemas utilizando glide64mk2.

XU3

El juego funciona muy bien en el XU3. Finalmente averigüe cómo utilizar la inyección, y también vi una opción para dos jugadores. Parece que si el juego encuentra más de un mando conectado, ofrece una opción de multijugador. La experiencia de juego es perfecta.

Star Wars: Rogue Squadron

Está considerado como uno de los mejores juegos de N64 que se han creado, en el que hacer volar un X-Wing contra el malvado Imperio. Jugué al juego en el PC cuando salió y era bastante divertido. Me quedé con ganas de probarlo en el ODROID. He leído que este juego requiere el pack de expansión de memoria para ponerlo en marcha. Sin embargo, no importaba lo que hiciese, no logre que funcionase en ninguna plataforma ni con ningún plugin de gráficos. Tanto los emulador Mupen64Plus como el nucleo libretro se bloqueaban o dejaban de responder.

ODROID MAGAZINE 20

StarCraft 64

StarCraft es un juego de estrategia en tiempo real (RTS) muy famoso. Es uno de los mejores juegos de estrategia jamás creado y todavía se juega en torneos de juegos profesionales. El juego de Nintendo 64 es un remake muy bueno en el que se han reducido los gráficos, los videos y la música. Es un gran juego de estrategia y me pareció muy interesante tener la oportunidad de ejecutarlo en un emulador de Nintendo 64.

U3

El juego funciona sorprendentemente bien en el U3. Hay algunos problemas de velocidad en el menú, que desaparecen tan pronto como entras en el juego, aunque existe un poco de retraso en el sonido, especialmente en las grandes batallas. Se puede oír como mueren las unidades después de que hayan desaparecido de la pantalla.

C1 - plugin rice

StarCraft 64 funciona asombrosamente bien en la C1. Parece que funciona mejor usando el plugin rice. Sin embargo, cuando se utiliza el plugin glide64mk2, el menú se vuelve tan lento que no puedes seleccionar la misión a la que quieres jugar. Por lo tanto, el juego no se puede jugar con glide64mk2. Posiblemente la velocidad del juego sea aceptable, pero como no se puede pasar del menú no hay forma de saberlo.

XU3

Tuve bastantes problemas para lograr que StarCraft 64 se ejecutara en el XU3. El juego iba muy lento al principio y el cambio desde glide64 a Rice o a gln64 mostraba cosas raras. Rice y gln64 iban realmente rápidos con el menú y todo iba a una velocidad aceptable. Sin embargo, tanto Rice como gln64 tenían grandes problemas gráficos, lo cual hacia que el juego fuese impracticable. Tras algunas investigaciones sobre la ralentización del glide64, me enteré que reduciendo la resolución se aumentala velocidad. El juego se muestra a 1080p sin importar la resolución elegida, pero la resolución de los personajes y objetos si se puede cambiar en el XU3. Descubrí que usando una resolución de 800x600 o menor se mejora el rendimiento notablemente.

Super Mario 64

Super Mario 64 fue el título de lanzamiento de la N64, ¡Y menudo título de lanzamiento fue! Este juego coloco a la N64 en la cima de las consolas de su clase, mostrando lo que la consola era capaz de hacer y una vez más, hizo de Mario la estrella de la franquicia de Nintendo.

ODROID MAGAZINE 21

U3

En el U3, Mario 64 tiene algunos problemas con las sombras, las texturas y la iluminación pero aparte de eso, el juego se ejecuta a toda velocidad.

CI - plugin ric

Mario 64 parece ejecutarse algo por debajo de la velocidad máxima en el C1, pero sigue siendo jugable con el plugin rice. La velocidad es un poco mejor con glide64mk2, pero de vez en cuando cae por debajo de la velocidad máxima. También tiene los mismos problemas que el plugin glide64mk2 en el U3 con las texturas del terreno y las sombras.

XU3

El juego funciona muy bien en el XU3, sin problemas ni fallos.

Super Smash Bros

Este juego introdujo un nuevo género. Fue un gran éxito en la N64, y dio lugar a una gran cantidad de segundas partes. Se puede elegir entre los famosos personajes de Nintendo como Mario, Yoshi, la princesa Peach entre otros y luchar contra otros personajes.

U3

La experiencia de juego de Super Smash Bros en el U3 con Mupen64Plus y el plugin glide64mk2 es muy buena. Incluso el menú funciona a una velocidad decente. Hay algunos problemas con la sombra y el texto pero nada serio, sólo la cuestión del texto es apreciable.

CI - plugin rice

El juego es demasiado lento con rice para ser jugable. El menú, la introducción y sistema de juego son muy lentos. Sin embargo, Super Smash Bros funciona mucho mejor con el plugin glide64mk2, y de hecho se puede jugar a toda velocidad, aunque tiene los mismos problemas que el U3.

XU3

Aunque en el menú se pueden apreciar algunas ralentizaciones, el juego funciona perfectamente. Fue realmente divertido jugar a este juego en el XU3.

ODROID MAGAZINE 22

The Legend of Zelda: Majora's Mask

No conozco demasiado los juegos Legend of Zelda en el N64, pero sí sé que en este juego dispones de 72 horas para salvar al mundo, y cuentas con diferentes máscaras para ayudarte en tu causa. Puedes utilizar el "Ocarina of Time" para viajar en el tiempo y empezar las 72 horas una y otra vez hasta que termine el juego.

U3

Aunque la velocidad del juego es muy buena, el plugin glide64mk2 tiene una vez más problemas por ser demasiado oscuro. Dado que no puedes activar el efecto borroso, el juego se ejecuta a toda velocidad todo el tiempo. Sin embargo, al ser demasiado oscuro, a veces es difícil encontrar un camino pero no tanto como cuando se juega en el XU3, donde nada es visible. Considero que es totalmente jugable.

C1 - plugin rice

El juego funciona sorprendentemente bien en la ODROID-C1 con el plugin rice. No hay problemas gráficos, aunque la introducción y algunas escenas son ligeramente lentas. En general, el juego es jugable en el C1 con este plugin.

C1 - glide64mk2

El juego funciona a casi toda velocidad, pero padece del mismo problema de oscuridad que el U3. Rice es probablemente el mejor plugin para ejectuar este juego en el C1.

XU3

La experiencia del juego en general es bastante buena. Cuando hay escenas con efectos borrosos el juego se ralentiza un poco. Aunque, sólo aparece en escenas cortadas. Sin embargo, hay otro problema relacionado con el plugin glide, que hace que los gráficos sean demasiado oscuros, hasta el punto de que resulta difícil averiguar qué camino elegir. Se volvió tan oscuro que decidí cambiar al plugin gln64 que tenía pequeños problemas con el terreno, pero por lo demás funciona a la perfección. No es tan oscuro como para no poder saber a dónde ir, así que el plugin gln64 funciona muy bien en este juego.

The Legend of Zelda: Ocarina of Time

Este es el predecesor de la Máscara de Majora. Me costó bastante trabajo poder disfrutar del juego, aunque supongo que con el tiempo la cosa mejoraría. Tiene que haber una razón por la que muchos lo tienen en su lista de los 10 mejores, así que le di una oportunidad.

U3

En general, el juego funciona muy bien con algunos problemas

ODROID MAGAZINE 23

menores con las sombras y las texturas del terreno. En algunos lugares, es demasiado oscuro, pero sigue siendo jugable.

CI - plugin rice

Al igual que el otro juego de Legend of Zelda, éste funciona muy bien sobre el C1 usando rice. Con el plugin glide64mk2, el juego no se ejecuta del todo bien y muestra los problemas típicos de texturas y sombras.

XU3

La experiencia en el XU3 es magnífica. No he visto ningún tipo de fallo o ralentización hasta el momento, aunque no he llegado muy lejos con el juego. Realmente la experiencia de juego es muy buena.

Testuras de Alta Difinición

Tras experimentar con diferentes juegos, me he dado cuenta que se puede hacer otra cosa con los emuladores. He descubierto que existen un pack de texturas de alta definición que ofrecen mejores gráficos. Decidí probar el pack en algunos juegos para ver cómo se ven y así determinar si funcionan con los ODROIDs. El emulador Mupen64Plus ofrece la posibilidad de utilizar estas texturas para los juegos de N64, mejorando así la experiencia de juego y dándole una nueva apariencia. Esta opción no está disponible para otros emuladores.

Para utilizar las texturas de alta definición, descargarlas desde `http://bit.ly/1Jvpahr` y copiarlas al directorio `~/.local/share/mupen64plus/hires_texture/`. Algunas texturas reescriben completamente los gráficos del juego. Asegúrate de colocar las texturas en una carpeta con el "nombre corto" del

Super Mario 64 con texturas normales y de alta definición

ODROID MAGAZINE 24

juego en mayúsculas. Por ejemplo, Mario 64 es “SUPERMARIO64” y Mario Kart 64 es “MARIOKART64”.

La completa transformación de texturas de Mario 64 le da al juego un aspecto más moderno

Conclusion

La emulación Nintendo 64 en general funciona muy bien en dispositivos ODROID, especialmente en el U3 y XU3. El C1 tiene un montón de problemas que le impiden ofrecer la misma experiencia de juego que los otros dispositivos. El plugin rice, que funciona sin tener que cambiar la profundidad del color en la imagen, tiene importantes problemas con muchos juegos, aunque hace un buen trabajo con otros.

El plugin glide64mk2 sólo funciona a 16 bits, y aunque la mayoría de los juegos funcionan muy bien, los que se ejecutan mejor con rice requieren reiniciar el sistema para poder usarlos, ya que rice no trabaja a 16 bits. Esto es un engorro, ya que siempre tenía que reiniciar el ODROID para cambiar entre los diferentes plugins gráficos en el C1. En el U3 y XU3 puedo hacer esto sin tener que reiniciar el sistema, lo que hace que sea mucho más fácil cambiar entre los plugins.

Además, usar la profundidad del color a 16 bits impide que algunas aplicaciones como el XBMC se ejecuten correctamente, lo que hace que tenas que elegir una interfaz de emulador que realmente soporta el modo 16 bits, o sino estarás obligado a iniciar los juegos de N64 a través de una ventana de terminal.

Todo esto me hace pensar que el C1 no es adecuado para la emulación N64, al menos en Linux. Creo que la mejor forma de disfrutar de los juegos de N64 en el C1 es posiblemente a través de aplicaciones para Android o con una versión modificada utilizando driver fbdev y algunos scripts capaces de cambiar la profundidad de color y las aplicaciones que ejecuta. Esta configuración ya de por sí supone un inconveniente y por supuesto no es adecuada para principiantes.

El U3 y XU3 dan la talla perfectamente a la hora de emular la N64. Ser capaz de cambiar con facilidad entre los núcleos gráficos es una gran ventaja frente al C1. Los Juegos de N64 parecen necesitar algunos ajustes de vez en cuando, y si nos fijamos en las opciones de configuración, ya sea de glide64mk2 o de rice que ofrece el emulador Mupen64Plus, hay un montón de opciones donde elegir.

El XU3 es la única placa que puede utilizar el núcleo libretro del Mupen64Plus con Retroarch por el momento. Integra los controladores muy bien pudiendo fácilmente adaptar a tu gusto la distribución del gamepad y es compatible con diferentes mandos. Además, el XU3 tiene un extra de potencia de CPU , que a menudo marca la diferencia con la velocidad máxima o “casi”. El U3 hace un muy buen trabajo con la emulación N64, el poder utilizar texturas de alta definición es realmente algo genial.

ODROID MAGAZINE 25

INSTALACION DEL DRIVER DE LA PANTALLA TACTIL 3.2” PARA EL ODROID-CI

por Bo Lechnowsky y Owen Browne

La pantalla táctil de 3.2” para el ODROID-C1 es uno de los periféricos más singular disponible en la tienda Hardkernel en http://bit.ly/1KYqWWw. La pantalla táctil viene con un pequeño puntero, y es capaz de visualizar el escritorio de Linux con una nítida resolución de 320x240, ideal para su uso en proyectos de robótica, ordenadores para vehículos y domótica. Los siguientes pasos describen cómo instalar su driver en cualquier distribución Ubuntu 14.04 manualmente. Para instalar el driver de forma automática, consulta la sección “Instalación automática” al final del artículo.

Instalación manual

En primer lugar, instala las dependencias y actualiza el sistema operativo del dispositivo:

```
$ sudo apt-get update && sudo apt-get install fix-w1-blacklist
$ sudo apt-get dist-upgrade
$ sudo apt-get upgrade
```

Después, ejecuta modprobe y cambia rotate=0 si quieres ejecutar la pantalla en modo retrato (vertical):

```
$ sudo modprobe spicc
$ sudo modprobe fbtft_device name=odroidc_tft32 rotate=270 \
  gpios=reset:116,dc:115 speed=32000000 cs=0
```

Para habilitar el escritorio X11, crea un archivo llamado /usr/share/X11/xorg.conf.d/99-odroidc-tftlcd.conf con el siguiente contenido:

```
Section “Device”
   Identifier    “C1 fbdev”
   Driver         “fbdev”
   Option         “fbdev” “/dev/fb2”
EndSection
```

El sistema de ventanas X se puede iniciar ahora:

```
$ sudo startx
```

Desactiva Xorg editando el archivo /etc/init/lightdm.override e incluye lo siguiente, guarda el archivo y reinicie:

```
manual
```

Una vez reiniciado el sistema, ejecuta con2fbmap:

```
$ con2fbmap 1 2
```

A continuación, agrega las siguientes dos líneas a /etc/modules, asegurándote de cambiar rotate=0 si deseas ejecutar la pantalla en modo retrato (vertical):

```
spicc
fbtft_device name=odroidc_tft32 rotate=270
gpios=reset:116,dc:115 speed=32000000 cs=0
```

Agrega la siguiente línea al final del archivo /media/boot/boot.ini:

```
fbcon=map:22
```

Añade esta línea a /etc/rc.local antes del comando exit para finalizar:

```
startx &
```

Reinicie de nuevo y prueba la pantalla táctil creando el archivo /etc/udev/rules.d/95-ads7846.rules con el siguiente contenido en una única línea:

```
SUBSYSTEM==”input”, ATTRS{name}==”ADS7846 Touchscreen”, ENV{DEVNAME}==”*event*”, SYMLINK+=”input/touchscreen”
```

A continuación, aplica el nuevo módulo escribiendo los siguientes comandos en una ventana de terminal:

```
$ sudo modprobe spicc
$ sudo modprobe -r ads7846
$ sudo modprobe ads7846
```

Después, comprueba si existe un nodo de pantalla táctil analizando el resultado del siguiente comando:

```
$ ls /dev/input/touchscreen
```

Luego, instala las librerías de pantalla táctil y eventos:

```
$ sudo apt-get install evtest tslib libts-bin
```

Realiza una prueba para ver si la pantalla táctil responde al tacto. El texto debe finalizar pulsando Control+C:

```
$ sudo evtest /dev/input/touchscreen
```

El dispositivo ha de ser calibrado para mapear correctamente el contacto en la pantalla. Para ello, elimina la calibración anterior:

```
$ sudo rm /etc/X11/xorg.conf.d/99-calibration.conf
```

A continuación, crea un directorio en el directorio X11 si no existe:

```
$ sudo mkdir /etc/X11/xorg.conf.d/
```

Dependiendo de si la pantalla táctil será utilizada en modo retrato (vertical) o paisaje (horizontal), siga los pasos de la sección correspondiente.

Calibración (paisaje)

Añade estas líneas en /etc/X11/xorg.conf.d/99-calibration.conf:

```
Section "InputClass"
        Identifier      "calibration"
        MatchProduct    "ADS7846 Touchscreen"
        Option  "Calibration"   "15 3836 4020 336"
EndSection
```

Calibración (retrato)

Para el modo retrato, la opción rotate=0 debe quedar activada con los siguientes pasos. Añade estas líneas en /etc/X11/xorg.conf.d/99-calibration.conf:

```
Section "InputClass"
Identifier      "calibration"
MatchProduct    "ADS7846 Touchscreen"
Option "Calibration" "37 5543 108 2290"
Option "SwapAxes" ""
EndSection
```

Tras reiniciar, la pantalla táctil debería estar activada y correctamente calibrada para su uso con el lápiz.

Calibración personalizada

Para realizar una calibración personalizada, puedes usar el programa xinput_calibrator:

```
$ sudo apt-get install xinput-calibrator && xinput_
calibrator
```

Instalación automática

La instalación del driver de la pantalla táctil también puede hacerse de forma automática usando el script precompilado de Ameridroid. En primer lugar, desconecta los cables de corriente y HDMI del C1, luego conecta la pantalla táctil al cabezal GPIO. Arranca el dispositivo y expande la partición root usando ODROID utility, luego conecta el C1 a Internet.

A continuación, cambia al usuario root lanzando una ventana de terminal y escribe lo siguiente. Tendrás que escribir la contraseña de root, que es "odroid" en las imágenes oficiales de Hardkernel:

```
$ sudo su
# wget http://respectech.com/odroid/c1-touch.sh
# chmod 755 c1-touch.sh
# ./c1-touch.sh
```

Pulsa "a", luego la tecla Enter para ejecutar todos los pasos de la instalación de forma automática. Si no has actualizado recientemente tu distribución, puedes irte a hacerte una taza de té, ya que llevara un tiempo actualizar todo el sistema operativo. El C1 se reiniciará una vez que la instalación se haya completado, y la pantalla táctil se iniciará automáticamente en el escritorio de Linux.

Conecta la pantalla correctamente para evitar un shock

NUCLEO RETROARCH NINTENDO 64 LINUX

PARA EL ODROID-U3

por Daniel Mehrwald

Retroarch, que es un emulador de juegos multisistema disponible para Android y Linux, siempre está en fase de desarrollo y su código abierto puede ser mejorado por cualquiera que presente una solicitud en GitHub. Recientemente, el núcleo del emulador de Nintendo 64 llamado Mupen64Plus cuenta con varias mejoras haciendo el juego más suave y mejorando el sonido. Para probarlo en Ubuntu 14 o 15, instala primero Retroarch con los siguientes comandos:

```
$ sudo add-apt-repository ppa:libretro/stable
$ sudo apt-get update
$ sudo apt-get install retroarch retroarch-* libretro-*
```

Luego, descarga y compila la última versión de Mupen64Plus:

```
$ git clone https://github.com/libretro/mupen64plus-libretro.git
$ cd mupen64plus-libretro
$ wget -O patch.txt http://pastebin.com/raw.php?i=XWBBFH7d
$ patch -p0 < patch.txt
```

Verifica que el parámetro gcc "-marm" está presente en el archivo Makefile en las líneas 92 y 93 editándolo con tu editor de texto favorito:

```
CFLAGS += -marm -mfloat-abi=hard -mfpu=neon
CXXFLAGS += -marm -mfloat-abi=hard -mfpu=neon
```

A continuación, compila el núcleo Mupen64Plus:

```
$ make -j5 V=1 platform='odroid odroid-u'
```

Copia el archivo resultante .so al directorio principal de Retroarch:

```
$ sudo cp libretro-mupen64plus.so /usr/lib/libretro/
```

Por último, Lanza tu juego de Nintendo 64 usando Retroarch con el nuevo núcleo:

```
$ retroarch -L /usr/lib/libretro/mupen64plus_libretro.so \
  ~/path/to/your-game.n64
```

Los ajustes óptimos de RetroArch son:

Settings->Driver Settings->Audio Driver = alsathread
Settings->Driver Settings->Audio Resample Driver = sinc
Options->Core Options->GFX Plugin = glide64

Si tienes preguntas o comentarios, por favor consulta el post original en http://bit.ly/1d8VR87.

IMAGENES DE LA COMUNIDAD

PARA ODROID-CI

por @robrodrigues

Una de las ventajas de la comunidad ODROID es la amplia variedad de imágenes que ofrece y que han sido subidas por los miembros del foro. Además de las imágenes oficiales que publica Hardkernel, las imágenes de la comunidad ofrecen características especiales que pueden ser útiles para determinados fines, como un centro multimedia, servidor de archivos o un sistema de almacenamiento conectado en red. Esta son algunas de las imágenes más populares que están disponibles para ODROID-C1 desde mayo de 2015:

Triple Boot
http://bit.ly/1EBmLvt

Ubuntu & Debian Minimal
http://bit.ly/1DLugEZ

KitKat Pocket Rocket
http://bit.ly/1e8hLso

GameStation Turbo
http://bit.ly/1JpaccI

OpenElec
http://bit.ly/1FlifFG

OpenMediaVault
http://bit.ly/1FgU0Ho

Tiny Core
http://bit.ly/1Jpah00

DietPi
http://bit.ly/1d9bXP8

Fedora
http://bit.ly/1IFc2XD

Minimal Debian Wheezy
http://bit.ly/1QT8x1l

Nas4Free
http://bit.ly/1d9aPLt

Arch Kali
http://bit.ly/1EcjsKf

Arch Official
http://bit.ly/1IFc6H6

Max2Play
http://bit.ly/1Ecjzpg

NetBsd
http://bit.ly/1e8iuK8

Debian 7.8
http://bit.ly/1d9b6xS

http://bit.ly/1B4PfwC

More Debian images
http://bit.ly/1B4Pgk4

Para sugerir más imágenes y añadirlas a los foros ODROID, consulte el post original en http://bit.ly/1d9blZQ.

EL ORIGINAL ODROID

DONDE TODO COMENZO

Editado por Rob Roy

Cuando Hardkernel creó la primera familia de ordenadores de placa reducida ODROID, estaba pensada para desarrolladores Android que quisieran crear prototipos de aplicaciones en un dispositivo pre-root de bajo coste, que facilitara el desarrollo de las aplicaciones. También era ideal para los juegos, fue uno de los dispositivos más innovadores de su tiempo para juegos Android.

Desde entonces, ODROID se ha convertido en una amplia línea de productos, desde dispositivos que puedes llevar encima a centros multimedia o sustitutos de escritorio, con muchos componentes adicionales para modificar el hardware, para robótica, la domótica, la programación Linux y Android, entre otros. Lo que continúa es el anuncio original del primer dispositivo ODROID desde finales del 2000, que era un dispositivo portátil con una pantalla incorporada, sensores giroscópicos y botones que finalmente se retiró en noviembre de 2009. ¡Disfruta de esta innovación del pasado!

HARDKERNEL

ODROID es el dispositivo con el que sueña un desarrollador. Viene totalmente montado con placa de depuración, código fuente y esquemas. También podías comunicarte con los ingenieros de todo el mundo a través de la comunidad de desarrolladores ODROID. Estaba basado en un Samsung S5PC100 833MHz ARM Cortex-A8 con acelerador multimedia NEON.

Videos de demostración

RockOn demo:

`https://www.youtube.com/watch?v=yM7N3JDnX4k`

SlideMe and SocialDROID:

`https://www.youtube.com/watch?v=-8S-8gCa2bo`

Android 720p Video HDMI Demo:

`https://www.youtube.com/watch?v=zEWrV8LuX04`

Space Megaforce:

`https://www.youtube.com/watch?v=oj5sKT_2-Dg`

Speed Forge 3D HDTV demo:

`https://www.youtube.com/watch?v=M5fKrScVtP8`

Juegos Android

La capacidad 3D OpenGL ES del chip permite ejecutar juegos 3D a alta velocidad como Speed Forge 3D. El ODROID se puede conectar a una TV HD de 42” a través de un cable mini HDMI. La plataforma ODROID está completamente abierta. Puedes desarrollar, editar y ejecutar numerosos juegos.

Reproducción de vídeo TV HD

ODROID soporta archivos de vídeo con una resolución de hasta 720p. Puedes conectar el ODROID a una gran pantalla de televisión de alta definición a través del conector mini HDMI incorporado. Navegar por la web, consultar redes sociales o el correo electrónico resulta muy fácil con una red WiFi.

Música

Elije entre las diversas aplicaciones de música disponibles en el mercado. Echa un vistazo a los flujos de cobertura que te permite mejorar la experiencia final.

Para ver el anuncio original, por favor visita la web de Hardkernel en `http://bit.ly/1Gx5Lr1`.

COMPARATIVA DE LAS 4 MEJORES SBC

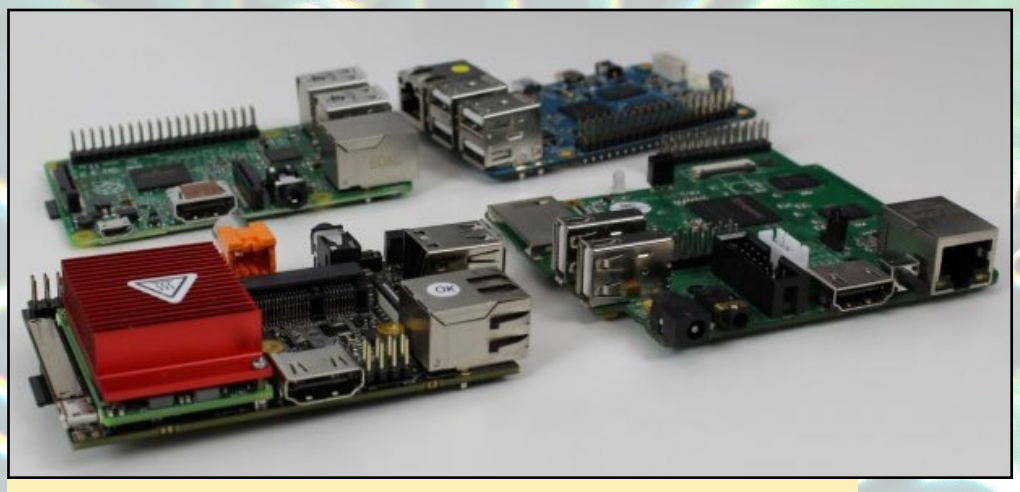
UNA PLACA QUE DOMINA AL RESTO

por Gary Sims

Reproducido con permiso de la Autoridad de Android (www.androidauthority.com)

La versión 2012 de la original Raspberry Pi creó todo un movimiento de aficionados, desarrolladores y educadores, que utilizaron la plataforma basada en ARM para crear, desarrollar y enseñar. Aunque la Raspberry Pi no fue el primer Ordenador de Placa Reducida (SBC) en el mercado, su éxito se debió a tres razones. En primer lugar, se trataba de un completo equipo sobre una pequeña placa, tenía un escritorio y se podía escribir programas de ordenador en él; segundo, tenía un conjunto de pines GPIO programables por el usuario, similar a los encontrados en las plataformas de microcontroladores como Arduino; tercero y probablemente la más importante, sólo costaba 35$.

Desde entonces, el mercado de las SBC ha ido crecido significativamente y la Raspberry Pi ya no es la única opción. Entre los dispositivos más conocidos encontramos el ODROIDC1, el HummingBoard, el MIP Creador CI20 y la Raspberry Pi 2. Por supuesto, la lista de placas disponibles es mucho más larga, pero estas son los que he probado personalmente.



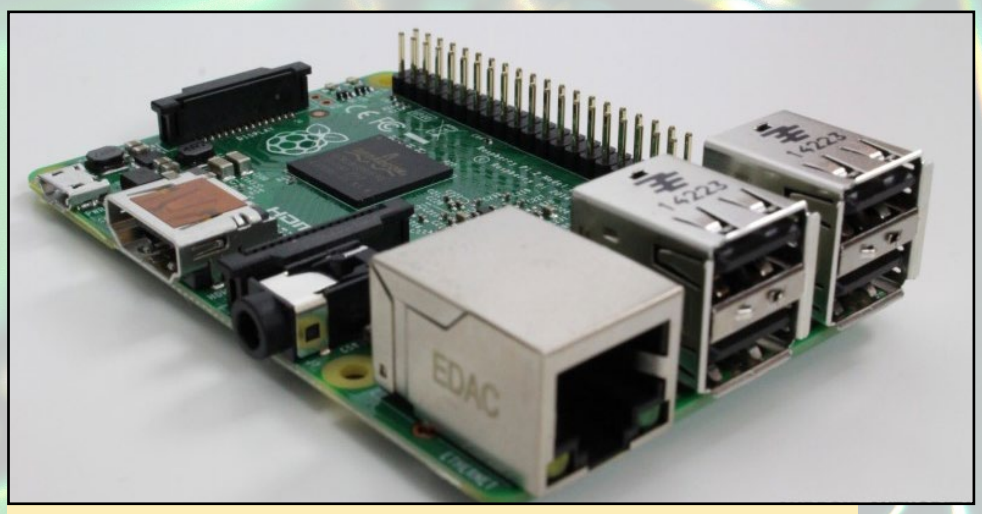
Tamaño de todos los ordenadores de placa reducida probados

El mercado de los SBC está fuertemente dominado por ARM y tres de las cuatro placas que vamos a analizar usan procesadores ARM. La excepción está el Creador CI20 que utiliza un procesador MIPS. Así que antes de empezar a comparar las placas, las voy a presentar formalmente a cada una.

Raspberry Pi 2

Aunque la Raspberry Pi 1 tuvo un gran éxito, tenía un inconveniente: el rendimiento global de la placa era insuficiente, especialmente cuando se ejecutan aplicaciones de escritorio. El rendimiento era menor del deseado, ya que utiliza un único núcleo CPU a sólo 900 MHz. Teniendo en cuenta el coste, el carácter innovador de la placa y su versatilidad, se puede decir que el nivel de rendimiento

es perfectamente comprensible, aunque existía la posibilidad de mejorarlo. Esa mejora se produjo con la llegada de la Raspberry Pi 2, que utiliza un procesador de cuatro núcleos y duplica la cantidad de memoria RAM. A pesar de que el Pi 2 es más potente y tiene más memoria, la fundación Raspberry Pi logró mantener exactamente el mismo precio. Una receta que garantizaría su éxito.

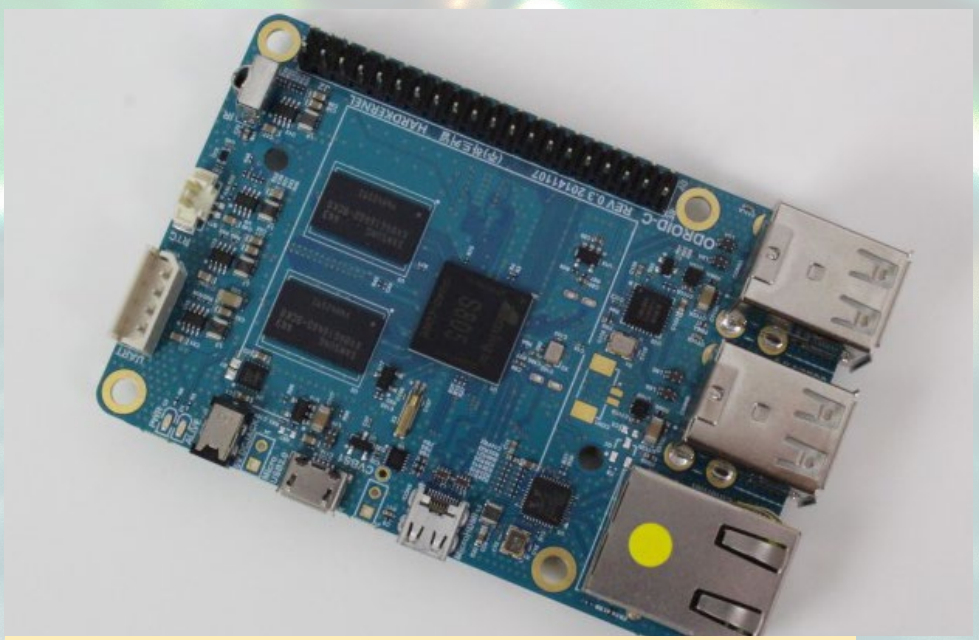


Raspberry Pi 2

ODROID-C1

Una de las razones clave del éxito de la Raspberry Pi fue su precio. Aunque hay muchas otras empresas que hacen SBC, no hay muchas que puedan igualar el precio de la Pi. Por supuesto, algunas de los placas son sólo algo más caras que la pi, y a decir vedad normalmente ofrecen más prestaciones, como veremos con la MIPS Creator CI20.

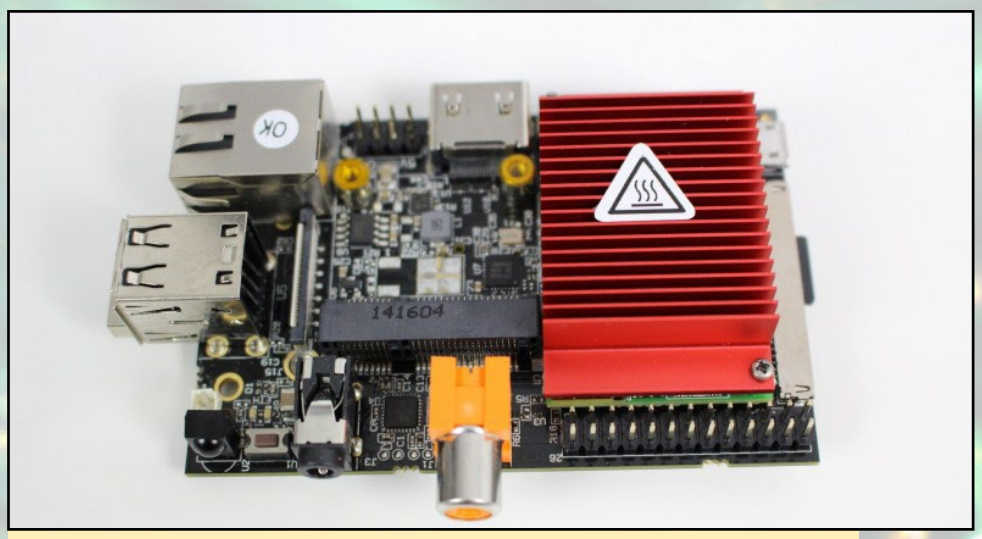
Sin embargo, la empresa que ha logrado desarrollar una placa por el mismo precio base que la Raspberry Pi ha sido HardKernel. Llamada ODROID-C1, también cuesta 35$. Y como la Pi 2, también utiliza un procesador de cuatro núcleos y viene con 1GB de RAM. El ODROID-C1 no es la única SBC que hace HardKernel pero es la más barata.



ODROID-C1

HummingBoard i2eX

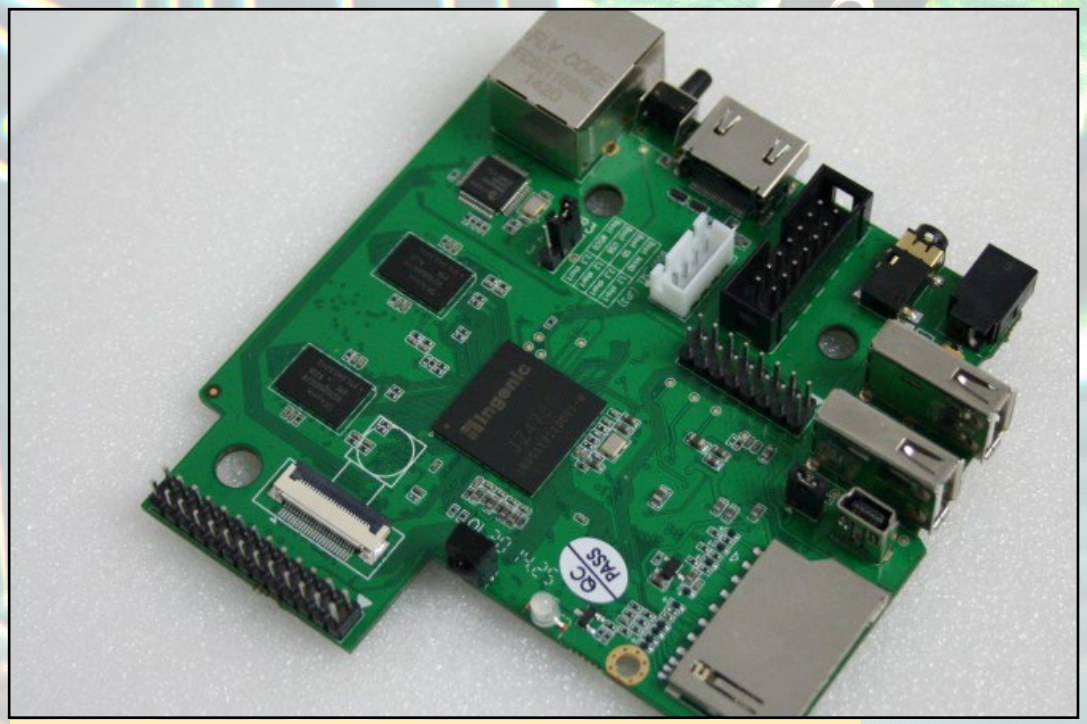
Otra compañía que ofrece varias SBC diferentes es SolidRun. Todos sus plcas están desarrolladas bajo la serie de procesadores i.MX 6 de Freescale. La gama i.MX 6 esta basada en el diseño del Cortex-A9 de ARM y su escala va de uno a cuatro núcleos. El HummingBoard I2EX utiliza un procesador de doble núcleo i.MX 6, viene con 1 GB de RAM, y tiene la misma forma que la Raspberry Pi 1 - incluso encaja en las carcasas diseñadas para la primera generación de la Pi.



Hummingboard

MIPS Creator CI20

La placa de nuestra lista que no utiliza un procesador ARM es la MIPS Creador CI20. En su corazón hay un procesador de doble núcleo MIPS junto con una GPU PowerVR y esta respaldada por 1 GB de RAM. Además es única al incluir su propio sistema de almacenamiento, Wi-Fi y Bluetooth integrados. Su coste es de 65$, más caro que el ODROID-C1 o laRaspberry Pi 2, pero obtienes a cambio más funciones.



Imagination CI20

Comparativa de características

Ahora que hemos presentado nuestras cuatro placas, ¿cómo podemos compararlas sobre el papel? Aquí tienes una lista de las especificaciones de cada una:

Device	ODROID-C1	Raspberry Pi 2	HummingBoard i2eX	Creator CI20
CPU	1.5Ghz quad core ARM Cortex-A5 CPU from Amlogic	900MHz quad-core ARM Cortex-A7 CPU from Broadcom	1GHz i.MX6 dual-core Cortex-A9 CPU	1.2GHz dual-core Imagination MIPS32 CPU
GPU	Mali-450 MP2 GPU	Videocore IV	GC2000	PowerVR SGX540
Memory	1GB	1GB	1GB	1GB
Storage	SD card slot or eMMC module	SD card slot	SD card slot	8GB onboard flash, SD card slot
Connectivity	4 x USB, microHDMI, Gigabit Ethernet, infra red remote control receiver	4 x USB, HDMI, Ethernet, 3.5mm audio jack	2 x USB, HDMI, Ethernet, 3.5mm audio jack, infra red remote control receiver	Ethernet, 802.11 b/g/n Wi-Fi, Bluetooth 4.0, 2 x USB, HDMI, 3.5mm audio jack
OS	Android, Linux	Linux, Windows 10	Linux, Android	Linux, Android
Connectors	GPIO, SPI, I2C, RTC (Real Time Clock) backup battery connector	Camera interface (CSI), GPIO, SPI, I2C, JTAG	Camera interface (CSI-2), GPIO, UART, SPI, I2C, PCI-Express Gen 2, mSATA II, RTC with backup battery	Camera interface (ITU645 controller), 14-pin ETAG connector, 2 x UART, GPIO, SPI, I2C, ADC
Price	$35	$35/£24	$110	$65/£50

Cada una de las placas de nuestro analisis puede ejecutar al menos dos sistemas operativos, todas trabajan con Linux y la mayoría pueden ejecutar Android. La única placa que no puede ejecutar Android es laRaspberry Pi, 1 ó 2. La Fundación Raspberry Pi no ve a Android como una prioridad, y parece ser que hay algunas dificultades para exportarlo debido a que faltan algunos drivers de Broadcom. Por supuesto, todo esto podría cambiar.

Android, sin embargo, se ejecuta en el ODROID-C1, la HummingBoard y en el MIP Creador CI20. Actualmente los tres sólo soportan Android 4.4 KitKat, pero tienen el potencial para ejecutar Android 5.0 Lollipop, sin embargo ninguno de los fabricantes de las placas ha lanzado oficialmente una ROM por ahora. (Nota del editor: El ODROID-C1 cuenta con una imagen beta de la comunidad con Android 5.0 Lollipop disponible en http://bit.ly/1B5Ysqh).

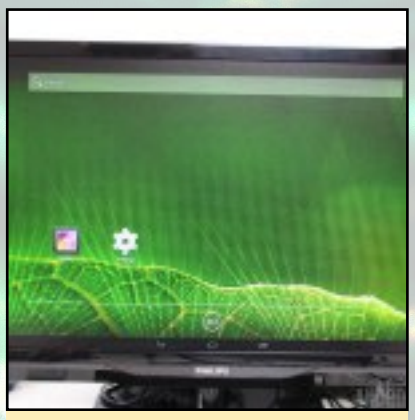
Para evaluar cómo de bien se sustenta Android en cada una de las placas voy a utilizar los siguientes criterios: características, rendimiento y soporte para los servicios de Google. Las dos principales características de Android que diferencias una placa de otra son el soporte para sonido a través de HDMI y el soporte para unidades USB. La mejor placa en estos términos es el ODROID-C1. La HummingBoard y la CI20 no admiten unidades USB bajo Android, y la CI20 no es compatible con el sonido a través de HDMI. Si puntuamos cada placa con un máximo de 4 puntos teniendo en cuentas estas características: el ODROID-C1 obtiene 4, el HummingBoard obtiene 3, y la CI20 2 puntos.

La siguiente comparativa es el rendimiento. Usando AnTuTu como base para el rendimiento, el ODROID-C1 anotó 15.887 y la HummingBoard-I2EX alcanzo 12.198. No he sido capaz de probar la CI20, pero según los comentarios que he visto en Internet, marca menos que las otros dos. Así que, si puntuamos cada placa con máximo de 4 puntos teniendo en cuenta el rendimiento, la ODROID-C1 obtiene 4, el HummingBoard obtiene 3, y la CI20 2 puntos.

Por último, soporte para Google Play y servicios de Google: la HummingBoard viene con Google Play pre-instalado, el ODROID-C1 no incluye los servicios de Google por defecto, pero se puede instalar con una app suministrada por Hardkernel. La CI20 no incluye soporte para los servicio de Google para nada. Por lo tanto, puntuando cada placa con un máximo de 4: la HummingBoard obtiene 4, el ODROID-C1 obtiene 3 y la CI20 2 puntos.

ODROID-CI ejecutando Android

Hummingboard ejecutando Android



MIPS CI20 ejecutando Android

Puesto que la Raspberry Pi no soporta Android, se le puntúa con un 0 en esta sección. Los totales en esta sección son:

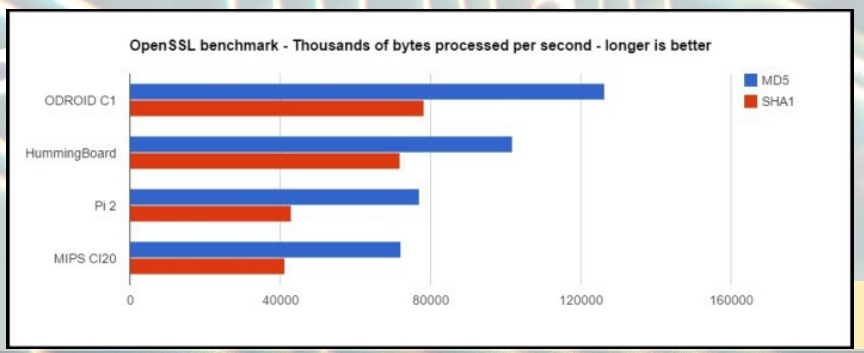
ODROID-CI	**II**
HummingBoard i2eX	**IO**
CI20 Creator	**6**
Raspberry Pi	**0**

Linux

Las cuatro placas soportan Linux y lo hacen muy bien. Para ponerlas a prueba y juzgar que placa soporta mejor Linux, he utilizado los siguientes criterios: el número de distribuciones soportadas, el rendimiento y la cantidad de memoria libre disponible en escritorio tras un arranque limpio.

La placa que soporta la mayoría de las distribuciones Linux es la Raspberry Pi 2. En gran parte debido al gran tamaño de su comunidad, es una plataforma muy popular y por lo tanto, recibe mayor atención en términos de portabilidad. De modo que las puntuaciones relacionadas con las distribuciones quedarían así: Raspberry Pi - 4, ODROID-C1 y HummingBoard – empatan a 3, y CI20 - 1.

En cuanto al rendimiento, la herramienta de línea de comandos OpenSSL cuenta con una opción de velocidad que pone a prueba el rendimiento de diversos algoritmos criptográficos. También permite evaluar el rendimiento de una CPU comparándola con otra, como se muestra en la siguiente imagen.



Prueba de rendimiento OpenSSL SBC

Las puntuaciones de las pruebas de rendimiento SSL son bastante reveladoras. La placa más rápida de las cuatro, en términos de rendimiento de CPU sin ayuda de la GPU, es el ODROID-C1. Luego va la HummingBoard, seguida de la Raspberry Pi 2. En último lugar pero no por mucho, va la CI20. En consecuencia, las puntuaciones de rendimiento son: ODROID-C1 - 4, HummingBoard - 3, Raspberry Pi 2 - 2, y CI20 – 1 punto.

Puesto que todas las placas tienen 1 GB de RAM, es interesante conocer la cantidad de memoria que queda libre una vez que la placa arranca el escritorio. Las interfaces gráficas de usuario necesitan memoria y cada placa usa un gestor de ventanas liviano para conservar la máxima memoria posible . Los resultados perteneces a las distribuciones recomendadas o por defecto, que arrancar el escritorio sin ninguna instalación adicional o configuración por parte del usuario.

La placa más frugal es la Raspberry Pi 2, que tenía libre tras arrancar 816360K. Luego viene la CI20 con737436K libre. El ODROIDC1 tiene 425836K libre y finalmente, el HummingBoard deja libre 313860K. Por lo tanto, las puntuaciones de la prueba de memoria son: - Raspberry Pi 2 - 4, la CI20 - 3, ODROID-C1 - 2, y HummingBoard - 1.

Cotejando todas las puntuaciones de esta sección, los resultados de las pruebas de Linux son los siguientes:

Raspberry Pi	**10**
ODROID-CI	**9**
HummingBoard i2eX	**7**
CI20 Creator	**5**

ODROID-CI ejecutando Linux

Raspberry Pi 2 ejecutando Linux

Kodi/XBMC

Las cuatro placas debería soportar Kodi/XBMC. Para probar el rendimiento de Kodi usé su pantalla de información de codec interna para mostrar la velocidad de los fotogramas y la cantidad de tiempo que usa la CPU para decodificar el vídeo. Luego grave un video Full HD de 50Mbps con mi ZTE Blade S6 Plus y lo ejecute en cada placa.

Tanto el ODROID-C1 y como el HummingBoard I2EX hacían un excelente trabajo pudiendo reproducir el video con una tasa de fotogramas máxima, al tiempo que en ninguno de los dos se forzaba demasiado la CPU. Lo mismo no puedo decir de la Raspberry Pi, que lamentablemente sólo podía magenar 9 fps, en lugar de los 23,97 fps necesarios. Por desgracia no pude encontrar una versión de Kodi que se ejecutara en el CI20, y tampoco pude localizar un reproductor de vídeo en los repositorios online.

Según la Fundación Raspberry Pi para que Kodi funcione en la Pi hay que evitar la GUI, lo que significa que la velocidad de fotogramas mostrada no es la exacta. En cuanto al retraso del ratón es un fenómeno conocido y se logran mejores resultados usando el teclado o un mando a distancia. Las puntuaciones de esta sección son: ODROID-C1 - 4, HummingBoard - 4, Raspberry Pi 2- 2, CI20 - 0.

ODROID-CI ejecutando Kodi

Raspberry Pi ejecutando Kodi

Otros sistemas operativos

La gran noticia que acompañó el lanzamiento de la Raspberry Pi 2 fue que Microsoft lanzará una versión gratuita de Windows 10 para el Pi 2, con la finalidad de crear dispositivos del Internet de las Cosas (IoT). Aunque la idea de que Windows 10 se ejecute en una Raspberry Pi suena intrigante, puede llegar a decepcionarlos. La versión IoT de Windows podría estar muy limitada y de hecho, puede incluso no incluir un escritorio. Además de Windows 10, la Raspberry Pi 2 tiene soporte para RISC OS, NetBSD, FreeBSD y OpenWrt.

En cuanto a las otras tres placas, cada una tiene cierta compatibilidad con diferentes sistemas operativos. Por ejemplo, FreeBSD es conocida por ejecutarse en el HummingBoard, mientras que NetBSD ha sido exportado a ODROID-C1 y a MIPS CI20 Creador. También hay un proyecto en curso para que el CI20 soporte OpenWrt.

En pocas palabras, el Raspberry Pi 2 tiene la más amplia cobertura de sistemas operativo y las otras tres placas tienen similar nivel de soporte. Por lo tanto para puntuar esta sección daré a la Raspberry Pi 2, 4 puntos y a los otras tres, 2 puntos a cada una.

Soporte de la Comunidad

Un factor importante en la selección de una SBC es el tamaño de las diversas comunidades online ¿Cuántas personas escriben en blogs sobre esta placa? ¿Hacen videos al respecto? ¿Escriben libros sobre el tema? ¿Ofrecer ayuda en los foros? No hay duda de que la comunidad Raspberry Pi es el más grande. Esto se debe principalmente al éxito de la Raspberry Pi original, Por otro lado, ya está claro que la comunidad ha aceptado la nueva junta Pi 2 con la misma pasión. Es difícil evaluar las comunidades online del ODROID y del HummingBoard, pero en líneas generales tienen un tamaño muy similar. La CI20 tiene la más pequeña de las comunidades en parte debido a su relativa novedad.

Como resultado, puntuó en esta sección de la siguiente forma: Raspberry Pi 2 - 4, el ODROID-C1 y la HummingBoard - 3 cada una, y la CI20 - 1.

¿Cuál es la placa ganadora?

Las puntuaciones totales son:

Device	ODROID-C1	HummingBoard i2eX	Raspberry Pi 2	MIPS Creator CI20
Android tests	11	10	0	6
Linux tests	9	7	10	5
Other OSes, Kodi/XBMC, community size	9	9	10	3
Totals	29	26	20	14

Los resultados finales muestran que el ODROID-C1 es el ganador de nuestra comparativa de placas. Quizás sea una sorpresa, ya que tal vez esperabas que el Raspberry Pi 2 fuese la ganadora. La razón por la que su puntuación es tan mala es por su falta de soporte para Android. Aparte de la ausencia de Android, la Pi 2 tiene otras debilidades. Fácilmente ha sido derrotada por el ODROID-C1 y el HummingBoard en términos de rendimiento, e incluso el procesador de doble núcleo MIPS se acerca al nivel de rendimiento de la Pi. Además, la versión actual de Kodi para Raspberry Pi no maneja el vídeo muy bien, lo cual podría mejorarse en el futuro, pero por el momento el ODROID-C1 y el HummingBoard realizan un trabajo magnifico en este sentido. Sin embargo, si necesitas soporte para Android el ODROID-C1 es el claro ganador.

Para obtener más información o para enviar preguntas o comentarios, por favor visite en http://bit.ly/1JoW8zT.

CONVERSION TEXTO-VOZ

CON EL ADAPTADOR DE AUDIO USB PARA C1

por Bo Lechnowsky y Brad Wilson

Hemos tenido unas cuantas oportunidades para aprender nuevos trucos mientras trabajabamos en nuestro robot OWEN (ODROID Walking Educational uNit). Uno de los obstáculos fue conseguir que el robot hablase a través del C1 con Ubuntu utilizando software liviano. El primer paso era conectar el adaptador de audio USB para C1, disponible en http://bit.ly/1RW7TS4, y configurar el sistema de conversión texto-voz (TTS) para usarlo.

1. Conecta el adaptador de audio USB

2. Instala Festival TTS:

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

3. Pon en marcha ALSA para utilizar la segunda salida de audio, que corresponde al adaptador de audio USB:

```
$ sudo pico ~.asoundrc

pcm.  !default {
      type hw
      card 1
}
ctl.  !default {
      type hw
      card 1
}
```

4. Configura Festival TTS para usar ALSA y audio de 16 bits. Para ello utiliza el usuario actual, escribe el siguiente comando:

```
$ sudo pico ~/.festivalrc
```

Opcionalmente, puedes tener ALSA por defecto para todos los usuarios, escribe lo siguiente:

```
$ sudo pico /etc/festival.scm
```

Agrega las siguientes líneas al archivo:

```
(Parameter.set 'Audio_Command
"aplay -D plug:dmix -q -c 1 -t
raw -f s16 -r $SR $FILE")
(Parameter.set 'Audio_Method 'Au-
dio_Command)
(Parameter.set 'Audio_Required_
Format 'snd)
(Parameter.set 'Audio_Method
'linux16audio)
```

Guarde el archivo y reinicia:

```
$ sudo reboot
```

¡Ahora puedes usar Festival TTS con el adaptador de audio USB para el C1! Aquí tienes un par de comandos de ejemplo:

Examplo 1: A beautiful day message (echo text)

```
$ echo "It's such a beautiful
day! Why are you in front of the
computer?" | festival --tts
```

Examplo 2: What's today's date? (program output)

```
$ date '+%A, %B %e, %Y' | festi-
val --tts
```

CAJA ESCRITORIO TODO EN UNO PARA ODROID-C1

por Dongjin Kim

En la película Jobs (2013), Steve Jobs entregó 50 unidades de la placa del ordenador Apple I a una tienda de informática llamada "Byte Shop". El propietario de la tienda, Paul Terrell, se quejaba de no tener los periféricos, como con la caja, la pantalla y el teclado. Tras un pequeño debate, Jobs dijo "todo en uno" para tus colegas cuando salió de la tienda. Mi parte favorita de la película es la escena en la que Jobs dice "¡increíblemente genial!" cuando por primera vez juega con un Macintosh original, que era un verdadero ordenador todo en uno.

Cuando vi por primera vez un Macintosh Plus en una imprenta cerca de mi casa, yo tenía unos 16 años y me encantó su forma. Caí en la cuenta de que se trataba de un Macintosh cuando empecé a estudiar programación. Recientemente, he intentado comprar uno mirando en el mercado de segunda mano y en eBay. Sólo he visto máquinas sucias y agrietadas. Considerando la posibilidad de compra uno y limpiarlo, finalmente, me decante por hacer una imitación a escala 1:1 del original Macintosh Plus utilizando un ODROID-C1.

No logre encontrar un diseño CAD oficial del Macintosh Plus, en su lugar localicé algunos proyectos que colocaban una tablet iPad o placa Micro ATX dentro de una carcasa de Macintosh original. Tenía las dimensiones de uno de esos proyectos, y traté de duplicar el tamaño y la forma. Lo más importante del diseño de la caja era los huecos de la pantalla y del disco flexible, puesto que estos son los elementos característicos de la firma Macintosh. La pantalla tenía que ser de 9" puesto que el Macintosh original utiliza el mismo tamaño de CRT, pero no fue fácil encontrar una pantalla de ese tamaño, así que en su lugar cogí una pantalla LCD IPS de 9,7".

Soy un ingeniero de software con conocimientos básicos de hardware y sin experiencia en CAD para diseño mecánico, así que decidí crear uno con una regla y un bolígrafo al estilo tradicional, utilizando un cartón de 3 mm de espesor. Sin embargo, el cartón era demasiado frágil para soportar las dimensiones reales de un Macintosh Plus, y dudaba de que pudiera sostener el peso de los periféricos como la LCD y el disco duro, de modo que lo cambié por acrílico de 5 mm, que es más resistente.

Puesto que elegí acrílico para la caja, no podía cortar todas las piezas del diseño a mano, y decidí encargar el corte del acrílico de acuerdo con los diseños CAD. Con el tiempo, este proyecto me llevó a aprender a utilizar la herramienta CAD SketchUp que es increíble. Finalice el diseño más rápido de lo que esperaba, con el único problema de que no estaba seguro de si las piezas de acrílico podrían ser montadas correctamente según mi diseño.

Tras una semana, las 20 piezas de acrílico llegaron, y cada pieza parecía estar fabricado exactamente como diseñé. Inmediatamente, comencé a montar todas las piezas y afortunadamente no faltaba nada. Sin embargo, había añadido demasiado espacio en los bordes de conexión, por lo que las piezas podían ensamblarse pero era necesario sujetarlas con fuerza. Menos mal que lo tenía previsto, había preparado conectores con aluminio reforzado.

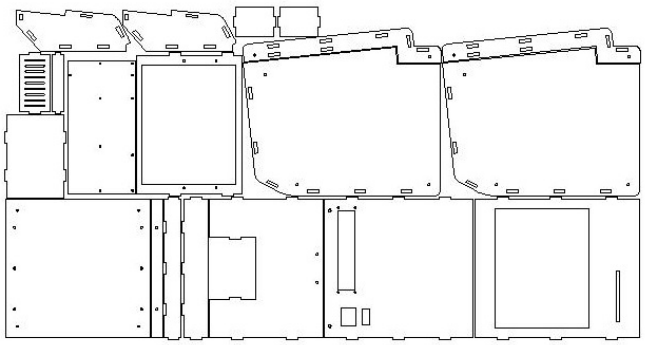


Figura I - Piezas de acrílico diseñadas con el software CAD

Mi hijo quiso ayudarme a completar la caja, y le pregunté si podía hacer la parte de la pantalla LCD. Le fue bien hasta que perdió el interés tras unas horas y salió a jugar con su amigo al parque, de modo que termine de montar la caja yo mismo.

Figura 2 - Caja de acrílico montada

Una vez conectados el teclado, el ratón y los altavoces, la caja sorprendentemente tenía la apariencia de un verdadero PC de escritorio. Tenía previsto usar un dongle de audio USB con altavoces externos, pero al final decidí incorporar altavoces internos en su lugar,

Figura 3 – Mi hijo montando la caja de acrílico

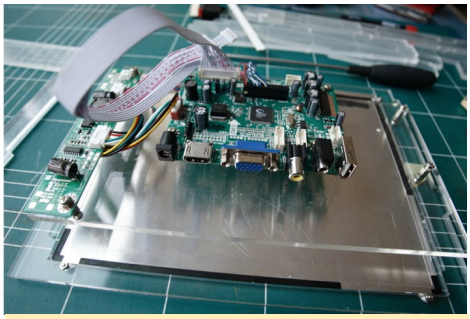


Figura 4 - ODROID-CI montado dentro de la caja de acrílico

Figura 5 - Panel LCD montado en el frontal de la caja

Figura 6 - Caja completamente montada con teclado, ratón y altavoces

que no estaban incluidos en mi diseño original. Por lo tanto, tuve que hacer a mano unos grandes agujeros en los paneles laterales izquierdo y derecho para las unidades de altavoz de 3". Me llevo un par de horas y ensucie bastante mi mesa de trabajo. Una vez que volví a montar la caja con los altavoces internos y empecé a jugar a Angry Birds, me di cuenta de que había tomado la decisión correcta.

Tras montar totalmente la caja descubrí que había cometido un grave error. Al estar completamente cerrada, no podia conectar dispositivos USB al ODROID-

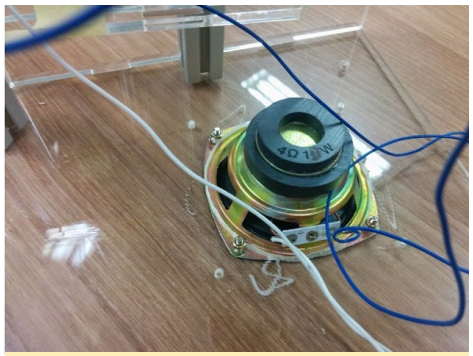


Figura 7 - Altavoz interno montado en el frontal de la tapa

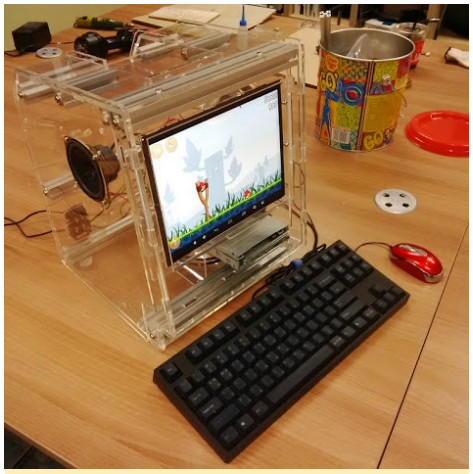


Figura 8 – Primera prueba del equipo montado ejecutando Angry Birds

C1 que se encuentra en el centro de la caja. Por lo tanto, tuve que añadir un panel de expansión E/S a la parte trasera, que había olvidado mientras diseñaba la caja. Dos conexiones UART, un par de puertos USB y conectores RJ45 parecían ser suficiente. Fabrique un panel con estas conexiones, luego tuve que soldar los cables al ODROID-C1. También tuve que añadir una placa de expansión en la parte superior del C1 con el fin de poder adaptar las señales RS-232 para dos UARTs con fines de depuración y GPIO. Mientras tanto, la fuente de alimentación de doble canal montada en el interior de la caja, suministra 5V al ODROID-C1 y 12V a la pantalla LCD.

La placa de visualización y el panel LCD tenian sus propios botones y receptores IR, los cuales representaban otro problema, ya que los botones necesita-

Figura 9 - Area periférica de conexiones

ban ser de fácil acceso y no podían estar montados dentro de la caja. Así que añadí más GPIOs para estos botones en

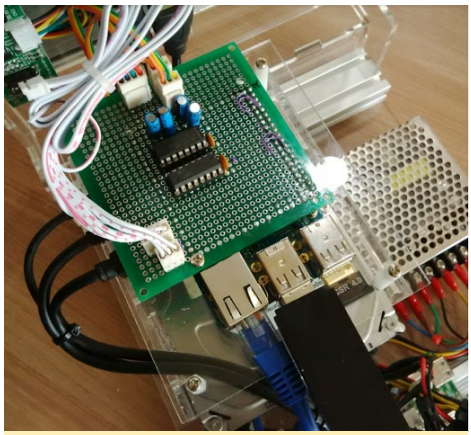


Figura 10 - placa de Extensión

ODROID Magazine esta en Reddit!

ODROID Talk Subreddit
http://www.reddit.com/r/odroid

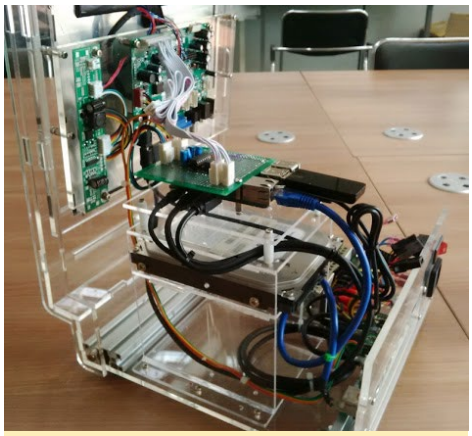


Figura 11 - Vista lateral de la caja montada parcialmente

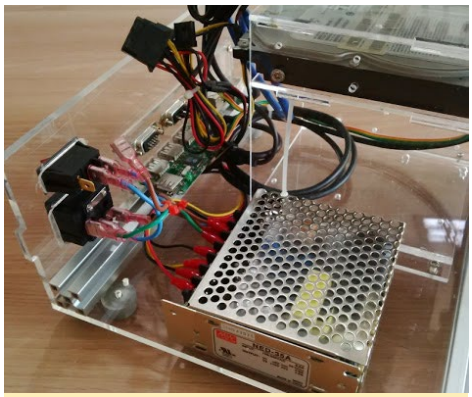


Figura 12 – Primer plano de la fuente de alimentación

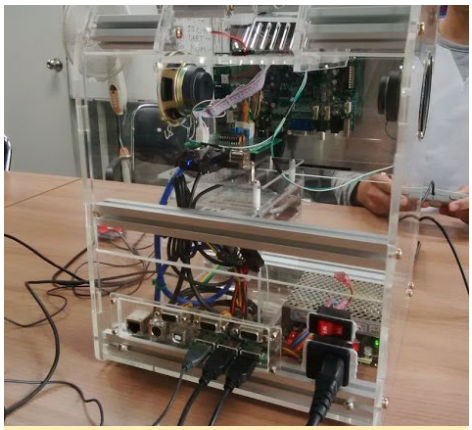


Figura 13 - Vista trasera

Figura 14 - Vista frontal

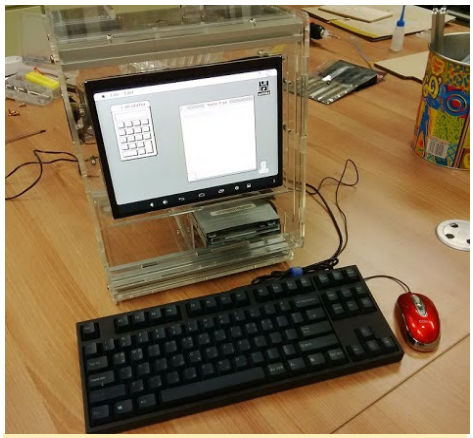


Figura 15 - Ejecución de Mini vMac, un programa emulador de Mac Plus

una placa de expansión y así controlar la visualización de la pantalla utilizando la librería WiringPi que se ejecuta en el C1, en lugar de tener que depender de receptor de infrarrojos de la pantalla.

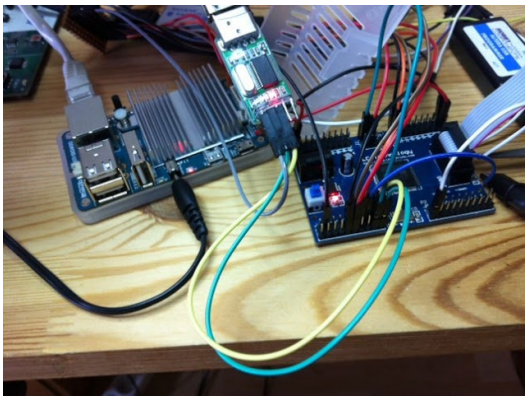
Mi ordenador se parece a un futurista Macintosh Plus transparente cuando ejecuta el software Mini vMac. Tengo la intención de mejorar aún más su funcionalidad, como incluir un lector de tarjetas SD disfrazando de unidad de disquete 3.5" conectado al ODROID-C1. La función más importante de la unidad de 3,5" es que pueda auto-expulsar la tarjeta SD como lo haría un Macintosh original, cuando se envíe un comando de software. La caja en sí pesa casi 4 kg cuando está montada, así que me gustaría actualizar el diseño para que fuese más ligera, con más agujeros de aire y que pueda ser más sencilla de montar.

Si quieres ver más fotos, visite el post original en los foros ODROID en http://bit.ly/1HFbmAE.

¡Ahora puedes jugar al Angry Birds en tu clásico Macintosh Plus!

GUZUNTY PI PARA EL ODROID-U3

USAR UN CPLD COMO UN CONMUTADOR PROGRAMABLE



por Carsten Foss

Tengo un ODROID-U3+ y quería que fuese capaz de conectarse a la consola serie. El procesador del U3 es un Exynos 4412, que utiliza 1.8V para las líneas de comunicación E/S. Las líneas E/S 1.8V representan un autentico desafío, ya que la mayoría de los adaptadores serie a USB en eBay eran de 5V o 3.3V, no siendo compatible con un conector 1.8V.

Mire un buen convertidor de nivel de Texas Instruments, el TXB0104, pero luego recordé que tenía una barata placa Altera EPM240 CPLD (Complex Programmable Logic Device) por ahí. Haciendo uso de un curso online sobre VHDL (VHSIC Hardware Description Language) para principiantes de este verano de PyroElectro (http://bit.ly/1PsRllZ), pensé en tratar de combinar el intercambio de niveles, además de mejorar mis habilidades con VHDL.

Como acababa de ver el anuncio de Hardkernel sobre el nuevo conector SPI (Serial Peripheral Interface) para la placa U3+, quizás podría hacer algo con la expansión del puerto SPI. Al haber192 macroceldas en el CPLD EPM240, es un chip ideal: http://bit.ly/1EdwxD4

Investigación

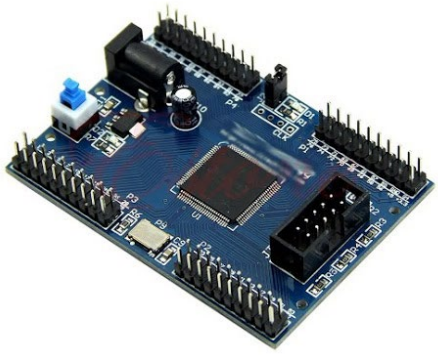
Eche un vistazo a Internet para ver algunos proyectos interesantes que podía utilizar como base, y encontré el proyecto Guzunty Pi en http://bit.ly/1QVEdTN, que era justo lo que estaba buscando. Es una CPLD de expansión E/S SPI para la Raspberry Pi con licencia GPL. También hay una serie de núcleos escritos en VHDL en http://bit.ly/1cJDZ30.

El corazón de Guzunty es un CPLD XC9600XL Xilinx. El diseñador eligió la serie CPLD XC9500XL debido a sus pines tolerantes de 5V sobre el CPLD. No podemos usar la serie CPLD XC9500XL para el U3, ya que no es compatible con 1.8V, necesitamos un CPLD que pueda utilizar 1.8V en la entrada del U3. Además de eso, ya había pensado en usar la placa MAXII EPM240 Altera de 7$ que tenía en el cajón. Ten en cuenta que yo no he creado el Guzunty Pi, sólo he exportado el VHDL a la placa EPM240.

Para la programación del CPLD Altera, he usado un clon Blaster USB Altera de 6$. Ten en cuenta que también hay disponible una placa EPM240 de color rojo, que no se recomienda por dos razones:

1. Las trazas VCCio1/2 están debajo del CPLD, haciendo imposible modificar la PCB. Tienes que levantar 3 pines VCCio1 para hacer que Bank1 funcione a 1.8V, con una separación del pin de 0,5 mm. La versión roja tiene el oscilador de 3.3V sobre Banco2, que por eso se recomienda utilizar Bank1 a 1.8V.
2. Prácticamente carece de todos los condensadores de desconexión, que el diseño EPM240 especifica.

Figura I – Placa EPM240



Placa CPLD azul

He subido el esquema para la placa EPM240 Altera azul a los foros del ODROID U3 en http://bit.ly/1QVEXIw. La placa viene con los siguientes conectores:

U1: Altera MAXII EPM-240 CPLD

U2: regulador de 3.3V

J1: Clavija de alimentación DC para conexiones de 5V

J6: Interruptor de alimentación que alterna los 5V al regulador

P9: Oscilador de 50MHz, conectado a GCLK0 (pin 12)

J2: Conector para conectar el LED de la placa al pin 77 del CPLD

JTAG: Conector JTAG de 10 pines

CLK: Cabezal de 4 pines para tener acceso a los 3 restantes pines de reloj.

P1-P4: Casi todos los otros pines se dirigen a los cabezales pin

Pines dedicados CPLD

El EPM240 tiene dos bloques E/S haciéndolo adecuado por un lado a 1.8V y por otro 3.3V. La mayoría de los pines en un CPLD están delimitados por el usuario, pero hay unos cuantos pines que son fijos, los cuales se detallan a continuación:

1: VCC (alimenta al núcleo CPLD)
2: VCCio (alimenta los bloques E/S)
3: GND (tierra)
4: Relojes universales (GCLK) (la EPM240 tiene 4, dos en cada bloque E/S)
5: Pines JTAG, usados para programar o depurar los contenidos CPLD.

El archivo pinout para el EPM240 puedes localizarlo en http://bit.ly/1F1ttRk. Los dos bloques E/S incluyen los siguientes pines: IoBank1 (pin 2..51) y ioBanco2 (pin 52..1). La figura 2 muestra los pines EPM240, con IoBank1 coloreado en azul claro y IoBank2 en gris claro. Por desgracia, los diferentes colores de IoBank suelen ser difíciles de distinguir, pero así es como se muestra en Altera quartus2.

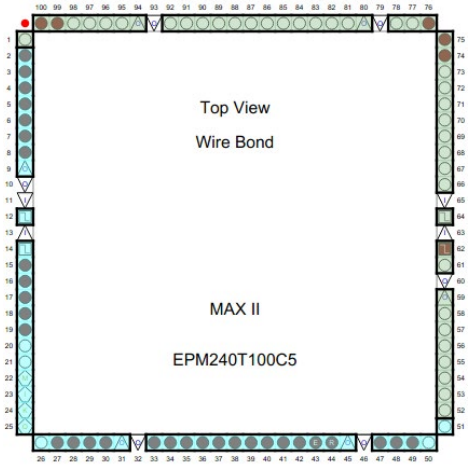


Figura 2 – Pines dedicados (fijos) en el EPM240

He elegido IoBank1 para los 3.3V (pin2..51) por el hecho de que el oscilador 3.3V (OSC) ya está conectado a GCLK0 (PIN12). De este modo, no tenía que desactivar el OSC integrado, pudiéndolo utilizar posteriormente para la modulación PWM o para una finalidad similar.

Debido a la necesidad de un lado a 1.8V y por otro 3.3V, hay aproximadamente 40 Pines E/S en cada bloque. 40 pines en el lado de 1.8V significa que hay una gran cantidad de pines que no se utilizan. Sólo necesitaba 4 para SPI y 2 para UART, o 4 para UART si quería conectar ambos UARTs en el lado de

Dedicated Pin	100-Pin TQFP
IO/GCLK0	12
IO/GCLK1	14
IO/GCLK2	62
IO/GCLK3	64
IO/DEV_OE	43
IO/DEV_CLRn	44
TDI	23
TMS	22
TCK	24
TDO	25
GNDINT	11, 65
GNDIO	10, 32, 46, 60, 79, 93
VCCINT *(1)*	13, 63
VCCIO1 *(2)*	9, 31, 45
VCCIO2 *(2)*	59, 80, 94
No Connect (N.C.)	-
Total User I/O Pins	80

Figura 3 - Diagrama del bloque E/S

3.3V, aunque esa es la forma en que lo hace sólo el EPM240. Los grandes EPM tienen 4 IoBanks, aunque no hay placas baratas disponibles. Puede leer más sobre la serie Altera MAX II en el manual del dispositivo MAX II en http://bit.ly/1GKJfLn.

Modificar la placa

Como la Figura 3, IoBank1 utiliza el pin 9, 31 y 45 para suministrar IoBank con el voltaje deseado y IoBank2 usa el pin 59, 80 y 94. Yo sabía que mi placa CPLD tenía 3.3V conectados a ambos IoBanks, así que necesitaba encontrar la forma de separar IoBank2 de los 3.3V a los que estaba conectado. Esto implicaba cortar algunas pistas de la PCB en los lugares correctos, esperaba que la placa tuviese una doble capa. Si fuera una placa de cuatro capas la modificación no sería posible, a menos que quisiera levantar los pines VCCio de la PCB y soldar cables a los pines separados por 0.5mm.

He encontrado algunas fotos de la PCB sin probar, traté de seguir las pistas de PCB en las imágenes, para ver donde podía cortar las pistas para interrumpir la conexión de los pines VCCio2 con el regulador de 3.3V integrado. Tras pasar un tiempo con las imágenes y un multímetro, modifiqué la placa como se muestra en la figura 4 con mi cuchillo X-acto.

Al principio, parecía no funcionar si sólo medía la resistencia entre el pin C5 y el C7. Parecía que todavía estaban conectados después de haber cortado las pistas de la PCB en los dos puntos indicados. Luego me di cuenta de que los pines VCCio2 estaban conectados dentro del CPLD e indicaban ohmios muy bajos, incluso cuando se separa de la pista de 3.3V. Cuando medí entre C5 / C7 y la pista 3.3V aún conectado a C6, pude ver que la conexión 3.3V se desco-

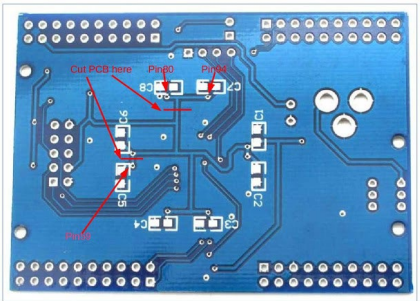


Figura 4 - Modificación de la PCB

nectaba de los pines VCCio2.

No tienes que cortar tan profundo como lo hice yo en la pista C5. Yo lo hice por error con VCCio2. Además, cuando corté la pista C5 arañe la marca de soldadura azul, y deje al descubierto el plano de la base. Esto podía dar lugar a un cortocircuito de C5 con la base, cuando soldara el cable al C5. Por suerte, me di cuenta de mi error antes de aplicar alimentación a la placa.

La Figura 5 muestra la PCB modificada, donde soldé un cable desde C5 a C7 para conectar los pines VCCio2, y soldé la mitad de un cable Dupont blanco al C8. El cable Dupont blanco (alimenta a VCCio2) está conectado al suministro de 1,8 V (pin 2) en el conector de expansión E/S del U3.

A continuación, soldé un cabezal de fila única de 4 pines sobre el cabezal CLK CPLD. Observa el resultado a la derecha, que indica que el pin GND está sobre el cabezal CLK. El cabezal pin GCLK está conectado con el siguiente patrón de izquierda a derecha: GCLK1, GCLK2, GCLK3, GND.

Alimentación

Usé 3 cables para la expansión E/S:

1: VDD_IO (pin 2) está conectado a la CPLD VCCio2 (cable blanco en C8)

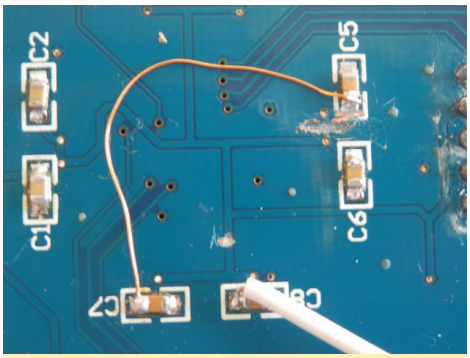


Figure 5 - Modified PCB

2: P5V0 (pin 8), está conectado a la alimentación de 5V de la placa CPLD

3: GND (pin 7) está conectado a GND en la placa CPLD

SPI

Utilicé los 4 cables desde el SPI:

1: SPI_1.CLK debe estar conectado a (VHDL - sclk), que es CLK2 en el cabezal de CLK

2: SPI_1.CSN debe estar conectado a (VHDL - sel), que es el pin 76 en el cabezal P1

3: SPI_1.MOSI debe conectarse a (VHDL - mosi), que es el pin 75 en el cabezal P1

4: SPI_1.MISO debe conectarse a (VHDL - miso), que es el pin 74 en el cabezal P1

Puedo con la misma facilidad tener VHDL mapeado - miso al pin 73 en el CPLD si creo que queda mejor. Todo el mapeo esta hecho en quartus2, ya sea con el planificador de Pin (más fácil), o el Editor de Asignación.

UART

Sólo use dos señales de los cabezales UART, puesto que ya tenía GND a través de la conexión de alimentación de 5V, dejando las otras señales abiertas:

1: TTA_RXD debe estar conectado a (VHDL - to_u3_rx), que es el pin 100 en el cabezal P4

2: TTA_TXD debe estar conectado a (VHDL - from_u3_tx), que es el pin 99 en el cabezal P4

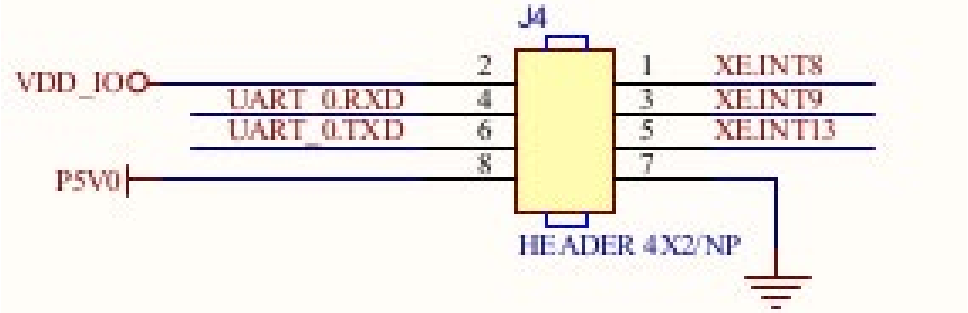


Figura 6 - Conector de Expansión E/S del U3

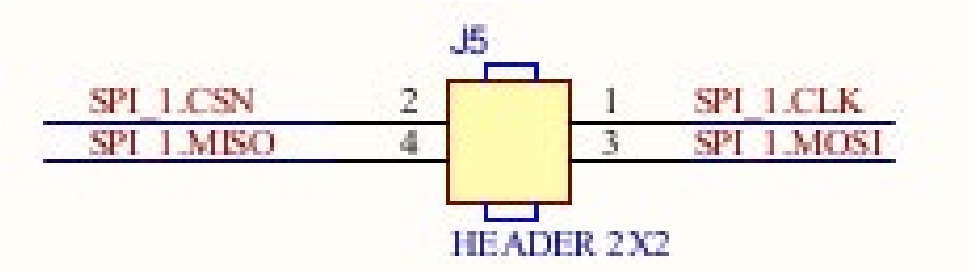


Figura 7 - Conector SPI del U3

Mi adaptador serie a USB de 3.3V utiliza 4 conexiones:

1: GND desde el ODROID

2: 5V desde el ODROID

3: USB TX debe estar conectado a (VHDL - from_usb_tx), que es el pin 28 en el cabezal P3

4: USB RX debe estar conectado a (VHDL - to_usb_rx), que es el pin 29 en el cabezal P3

Sólo se necesita dos líneas VHDL adicionales y cuatro entradas del Planificador de pin para poder conectar el otro UART del U3 en la cabezal de expansión hacia el CPLD y otro adaptador USB-serie, es la ventaja de usa un CPLD.

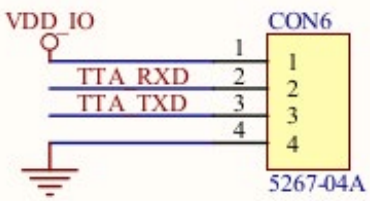
Instalar Quartus2

No voy a tratar la instalación y uso de quartus2, ya que hay un montón de lugares en Internet donde puedes encontrar guías de instalación para ello. Uno de ellos es el curso Pyro que he mencionado al principio del artículo.

Estoy utilizando la edición de Linux de 64 bits de la edición web gratuita de quartus2 en mi maquina con Linux Mint 17, pero recomiendo a la mayoría de la gente utilizar la versión de Windows, ya que la versión Linux generalmente necesita algunas librerías adicionales que deben ser manualmente compiladas, además de tener que modificar algunos archivos de quartus2.

La versión más reciente de quartus2 que cuenta con soporte MAX II es quartus2 v.13.0sp1. No instales una versión más reciente, ya que no funcionará correctamente. Para descargarlo, visita http://bit.ly/1cdWtJg, requiere registrarse. Pincha en la pestaña “Combined

Figura 8 - Cabezal UART U3 UART



files" y descarga el archivo de 4.4GB. Instala Quartus, regístrate en Altera y consigue un archivo de licencia web gratuita.

Desarrollar el diseño

Descarga el archivo de proyecto quartus desde http://bit.ly/1H8iGQs, Luego extrae el proyecto e inicia quartus2. Selecciona File->Open Project, busca el proyecto extraído y selecciona el fichero archivo gz_16o8i.qpf. Compila el diseño seleccionando el botón marcado con el circulo rojo, como se muestra en la Figura 9 e ignora las advertencias.

A continuación, apaga el ODROID y desconecta la alimentación. Conecta el cable Blaster USB Altera de 10 pines a la placa CPLD y conecta los 5V y 1.8V a la placa CPLD. Yo use los 5V y el GND desde la placa ODROID y los conecté al cabezal P1 en la placa CPLD. Después conecte los 1.8V desde el ODROID hasta el punto de modificación en la placa CPLD.

Enciende el ODROID, conecta el programador al puerto

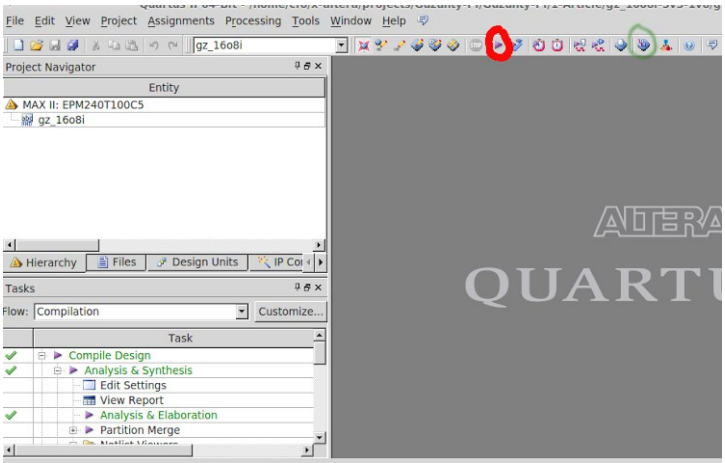
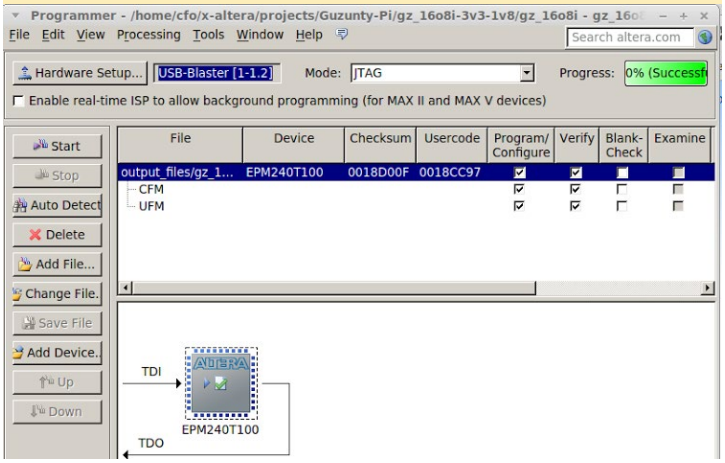


Figure 9 - Quartus 2

USB del PC, e inicia la herramienta de programación pulsando el botón indicado con el circulo verde en la Figura 9. Pulsa el botón de configuración de hardware y selecciona el programador JTAG. Debería ver algo similar a lo que se muestra en la Figura 10. Pulsa "Start" y programa tu archivo de diseño en el

Figure 10 - Quartus Programmer



Preparar el software

El Guzunty Pi utiliza SPI para las comunicaciones y para utilizarlo, tenemos que cargar el módulo del kernel SPI del ODROID-U3. Te recomiendo que utilices la herramienta de las páginas web de ODROID en http://bit.ly/1JmO6HH y http://bit.ly/1RS94lw, ya que describen cómo cargar el módulo del kernel, así como la forma de actualizar el software del U3 si fuese necesario.

Cargar el módulo SPI

Se puede cargar el módulo del kernel SPI manualmente escribiendo lo siguiente en una ventana de terminal:

```
$ sudo modprobe spi-s3c64xx
```

Sin embargo, lo mejor es que estuviera disponible de forma automática en cada arranque. Como root, edita el archivo /etc/modules y añade las siguientes líneas al final del archivo:

```
# SPI for the Guzunty Pi CPLD
spi-s3c64xx
```

Suele ser una buena práctica en programación no solicitar privilegios de root al acceder a dispositivos SPI. Para dar acceso a otros usuarios, crear un nuevo grupo de sistema spidev, luego configura el sistema udev para dar al grupo spidev acceso de lectura y escritura (r/w) a los dispositivos SPI:

```
$ sudo groupadd -f --system spidev
$ sudo usermod -a -G spidev username # replace user-
name with your ODROID username
```

Despues, crea una regla udev para spidev, permitiendo acceso a los miembros del grupo spidev. Como root, crea un archivo llamado /etc/udev/rules.d/99-spidev.rules:

```
# /etc/udev/rules.d/99-spidev.rules
# Allow access for the group members of spidev, in
order to be able to access SPI as a normal user
SUBSYSTEM==”spidev”, MODE=”660”, GROUP=”spidev”
```

Después de guardar el archivo y reiniciar, los miembros del grupo spidev tendrán el acceso adecuado al subsistema spidev. Los módulos SPI pueden comprobarse con el comando:

```
$ lsmod | grep spi
spideV 5641 0
spi_s3c64xx 9849 0
```

Comprueba que el grupo spidev tiene el acceso adecuado al dispositivo SPI:

```
$ ls /dev/spi* -l
crw-rw---T 1 root spideV 153, 0
Jan 1 2000 /dev/spidev1.0
```

Software Guzunty

El software Guzunty tiene dos secciones, cada una en su propio directorio:

1: gzlib, es la librería Linux que interactúa con el subsistema de SPI.

2: El programa de usuario que corresponde al diseño CPLD, que es gz_16o8i en este caso.

Para ejecutar el software Guzunty, es necesario instalar los paquetes git y build-essential de linux. Git es usado para clonar el repositorio de software Guzunty, y build-essential es el sistema de desarrollo de Linux que proporciona el compilador C y otras herramientas:

```
$ sudo apt-get install git \
  build-essential
```

Crear el directorio Guzunty, después clona el repositorio Guzunty en el subdirectorio Pi:

```
$ mkdir Guzunty
$ cd Guzunty
$ git clone https://github.com/\
  Guzunty/Pi.git
```

Es necesario cambiar la siguiente línea en el archivo fuente de la librería Guzunty llamado gz_spi.c:

```
$ cd Pi/src/gzlib/src/
```

Edita el archivo gz_spi.c y cambia "/dev/spidev0.0" por "/dev/spidev1.0" y guarda el archivo:

```
void gz_spi_initialize() {
// spi_open("/dev/spidev0.0");
// Comment out the
Raspberry Pi line by adding // in
front of it
spi_open("/dev/spidev1.0");
// Add the correct odroid SPI
device above
initialized = 1;
}
```

He cambiado la velocidad de SPI de 10MHz a 1 MHz. 10 MHz era demasiado rápido para mi analizador lógico Saleae Basic:

```
//#define SPI_MAX_SPEED 10000000
// 10 MHz
#define SPI_MAX_SPEED 1000000 //
1 MHz
```

No es necesario cambiar la velocidad a 1 MHz si no quieres comprobar las señales con un Saleae, o cuentas con un analizador lógico más rápido. Creo que el ODROIDU3 es capaz de ejecutar 40 MHz en el sistema SPI, pero yo sólo lo he probado a 10MHz. Si optas por 40Mhz, los cables SPI tendran que ser lo más cortos posibles, o experimentarás problemas con la integridad de la señal. Si tiene problemas, intente reducir la velocidad SPI a 1 MHz, y ver si se solucionan. Si lo hace, es probable que estés utilizando cables demasiado largos.

A continuación, compila e instala la librería SPI Guzunty, con el fin de que esté lista para ser vinculada con nuestro programa de diseño CPLD:

```
$ make
$ sudo make install
```

Dado que nuestro diseño está basado en el diseño gz_16o8i de Guzunty con 16 salidas y 8 entradas, cambia de directorio a Pi/src/ gz_16o8i y compila el programa de usuario:

```
$ cd Pi/src/gz_16o8i
$ make
```

Si obtiene el error " fatal error: gz_spi.h: No such file or directory", es que olvidaste ejecutar "sudo make install" del gzlib. De lo contrario, tu programa y CPLD están listos para usarse. Empieza con el siguiente comando.

```
$ ./gz_16o8i
```

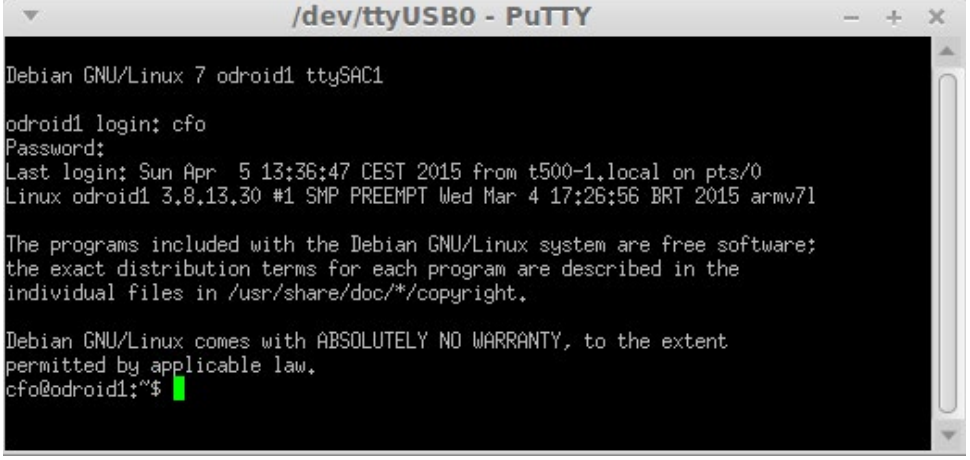
A continuación, prueba la conexión UART desde un ordenador usando PuTTY, debería aparecer una consola como la que se muestra en la Figura 11.

Conclusión

Usando Guzunty Pi, he logrado mi objetivo de crear un convertidor de nivel versátil y barato para usarlo con un convertidor serie a USB de 3.3V. Este método es más complicado si sólo se usa el chip TI, pero también creo que Guzunty para el ODROID-U3 es una buena aportación a una potente placa.

Para preguntas y comentarios, por favor consulta el hilo de mensajes sobre Guzunty en los foros ODROID en http://bit.ly/1d5qqM4. Me gustaría dar las gracias a los miembros @odroid y @robroy por ayudarme durante el desarrollo.

Figura II - Guzunty Pi - consola via UART



DESARROLLO ANDROID

CREAR UN SERVIDOR WEB A MEDIDA

por Nanik Tolaram

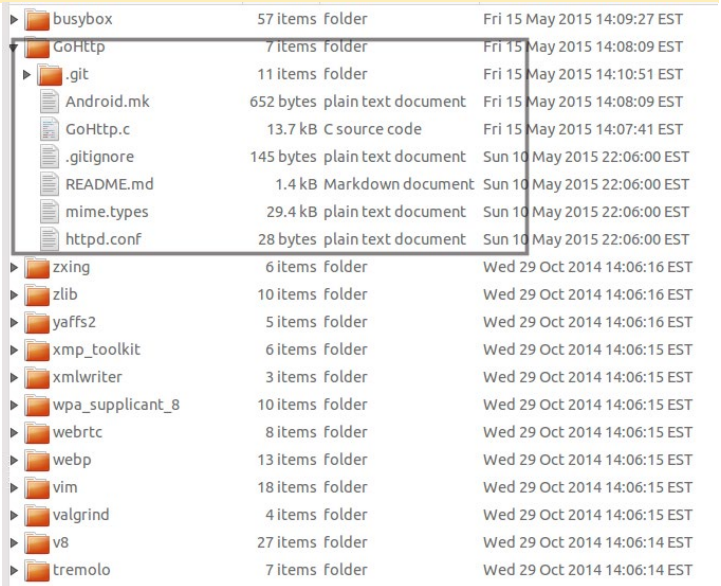
Anteriormente, analice cómo se inicia Android y los diversos servicios que se activan durante el proceso de arranque. También analice el contenido del init.rc que muestra los diferentes servicios que son necesarios para que Android haga su trabajo.

En este artículo, voy a analizar brevemente cómo añadir un servicio personalizado utilizando un servidor web como ejemplo. Esto se consigue en dos pasos: añadiendo una aplicación nativa como parte del proceso init y luego, exportando una aplicación nativa de Linux a Android.

GoHttp

Vamos a utilizar un servidor web de código abierto como parte de este ejercicio. El servidor Web es una aplicación muy básica, no es un servidor web completo. La aplicación exportada se puede descargar en https://github.com/nanikjava/GoHttp desde la sección master-android-parch. El proyecto original también está disponible en https://github.com/fekberg/GoHttp.

Figura I: Gohttp dentro de la carpeta /external



Compilar el archivo

Dado que queremos compilar la aplicación como parte de nuestra imagen de Android, vamos a necesitar integrarla en el código fuente de Android. Tal y como muestra la Figura 1, es necesario colocarla en la carpeta external/.

El primer paso es crear el archivo de desarrollo Android.mk, que es similar al Makefile de Linux. El archivo Android.mk para el proyecto GoHttp usado en este ejemplo debería tener el siguiente aspecto:

```
LOCAL_PATH:= $(call my-dir)

include $(CLEAR_VARS)
LOCAL_SRC_FILES :=  GoHttp.c
LOCAL_LDLIBS += -lrt -ldl -lpthread -llog
LOCAL_CFLAGS := -DDEBUG_ANDROID
LOCAL_SHARED_LIBRARIES := liblog
LOCAL_MODULE := gohttp
include $(BUILD_EXECUTABLE)

include $(CLEAR_VARS)
LOCAL_MODULE := httpd.conf
LOCAL_MODULE_CLASS := ETC
LOCAL_MODULE_PATH := $(TARGET_OUT)/etc
LOCAL_MODULE_TAGS := optional
LOCAL_SRC_FILES := $(LOCAL_MODULE)
include $(BUILD_PREBUILT)

include $(CLEAR_VARS)
LOCAL_MODULE := mime.types
LOCAL_MODULE_CLASS := ETC
LOCAL_MODULE_PATH := $(TARGET_OUT)/etc
LOCAL_MODULE_TAGS := optional
LOCAL_SRC_FILES := $(LOCAL_MODULE)
include $(BUILD_PREBUILT)
```

El archivo Android.mk es similar a otros proyectos dentro del directorio external/. Fíjate que hay 2 archivos que se copian como parte del proceso de compilación:

- httpd.conf → contiene la configuración del servidor web
- mime.types → contiene los tipos de archivo que permite la aplicación de servidor web.

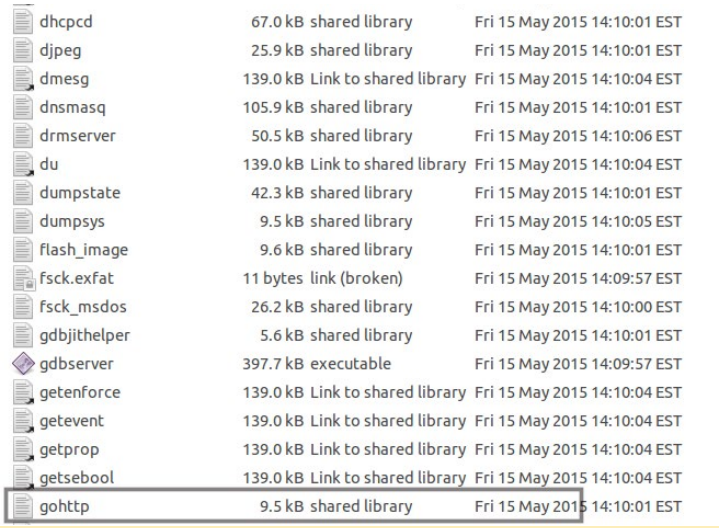


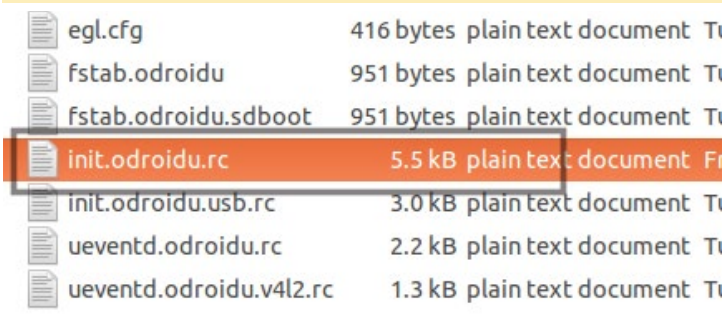
Figura 2: gohttp ejecutable

El servidor web está configurado para escuchar el puerto 8888, que esta especificado en el archivo httpd.conf.

El otro elemento que necesita modificarse dentro de la aplicación es el registro de datos, ya que tiene sentido asegurarnos de que la aplicación envíe cualquier información al servicio Logcat. Este es el nuevo código que se agrega dentro de GoHttp.c para activar el registro de datos:

```
#ifdef DEBUG_ANDROID
#include <android/log.h>
#define LOG_TAG "gohttp"
#define PRINT(...) __android_log_print(ANDROID_LOG_
INFO, LOG_TAG, __VA_ARGS__)
#else
#define PRINT(...) fprintf(stdout, "%s\n", __VA_
ARGS__)
#endif
```

Figura 3: Archivo init para ODROID-U3



Una vez que hayas compilado correctamente el código fuente de Android para ODROID-U3, verás un archivo ejecutable gohttp dentro del directorio out/target/product/odroidu/system/bin, como se muestra en la Figura 2.

Inicialización

El último paso es ejecutar la aplicación gohttp como parte del proceso de inicio de Android. Para ello, tendremos que modificar el archivo init.odroidu.rc dentro del directorio device/hardkernel/odroidu/conf, como se muestra en la Figura 3.

```
on post-fs-data
    mkdir /data/media 0770 media_rw media_rw

    setprop vold.post_fs_data_done 1

    mkdir /data/www  0770 system system
```

Figura 4: mkdir para crear el directorio

También tenemos que añadir un servicio para iniciar gohttp:

```
service gohttp /system/bin/gohttp
    class core
```

Debajo de la sección "on post-fs-data ", es necesario agregar el enunciado que se indica en la figura 4. Esta expresión se utiliza para crear el directorio donde pondrán los archivos .html que quieres que estén disponibles para que acceda el usuario.

Una vez que arranque el ODROID-U3, puedes copiar un archivo index.html a la carpeta /data/www y acceder a él a través del servidor con cualquier navegador introduciendo la URL http: // <IP_ODROID_U3>:8888/index.html.

Si deseas más información sobre programación en Android, contacta con Nanik directamente, sigue sus últimos post, o visita su sitio web en http://naniktolaram.com.

Con la ayuda de Nanik, estás totalmente equipado para crear tu propio servicio de Android personalizado

ULTRASTAR DELUXE KARAOKE

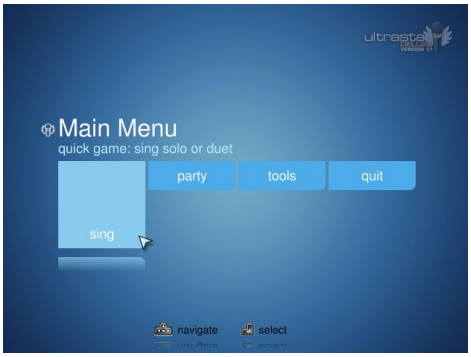
CONVIERTETE EN UNA ESTRELLA DEL ROCK CON ODROID

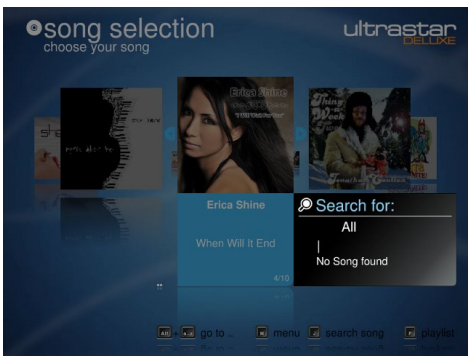
por @vOltumna

Eres un aspirante a estrella de rock, UltraStar Deluxe es un juego de karaoke de código abierto gratuito, similar al juego de Sony PlayStation llamado Sing-Star, que te permite crear una máquina de karaoke portátil. El código fuente esta publicado en http://bit.ly/1e9amZW, y puede ser fácilmente compilado para ODROID, convirtiéndolo en una estupenda maquinaria de fiesta. Yo uso dos micrófonos inalámbricos Sing-Star que funcionan muy bien.

Compilar ffmpeg

Ultrastardx tiene adaptaciones de diferentes versiones de ffmpeg para reproducir archivos multimedia. Sin embargo, detecte durante mis pruebas que no todas las versiones de ffmpeg funcionan correctamente. Probé diferentes versiones obteniendo diversos errores, pero lo logré con la versión 2.1.5 de ffmpeg. Antes de instalarlo, es bueno eliminar la versión antigua y algunos paquetes dev, de lo contrario Ultrastar podría encontrar información de la versión anterior durante la compilación.






```
$ sudo apt-get remove ffmpeg
libavutil-dev \
  libswscale-dev libavcodec-dev
libavdevice-dev
```

Luego instala la versión personalizada

```
$ wget http://www.ffmpeg.org/re-
leases/ffmpeg-2.1.5.tar.bz2
$ tar xf ffmpeg-2.1.5.tar.bz2
$ cd ffmpeg-2.1.5
$ export CFLAGS="-O3 -D__ARM_
NEON__ -fPIC \
  -march=armv7-a -mfloat-abi=hard
-mfpu=neon \
  -ftree-vectorize -mvectorize-
with-neon-quad \
  -mcpu=cortex-a9 -mtune=cortex-
a9"
$ export CXXFLAGS="-O3 -D__ARM_
NEON__ -fPIC \
  -march=armv7-a -mfloat-abi=hard
\
  -mfpu=neon -ftree-vectorize
-mvectorize-with-neon-quad \
  -mcpu=cortex-a9 -mtune=cortex-
a9"
$ ./configure --enable-libvorbis
--enable-pthreads \
  --enable-libmp3lame --enable-
nonfree \
  --enable-gpl --enable-libxvid
--enable-libx264 \
  --enable-shared --prefix=/usr
$ make -j5
$ make install
```

En Debian o Ubuntu, es posible que quieras ejecutar checkinstall -D para crear un archivo .deb instalable.

Compilar ultrastardx

Ultrastardx está escrito en Object Pascal, necesitarás Free Pascal Compiler y algunas unidades de dependencia:

```
$ apt-get install fp-compiler fp-units-misc fp-units-base \
  fp-units-math fp-units-fv fp-units-fcl
```

Despues, descarga el último código fuente de ultrastardx y configúralo:

```
$ svn checkout \
  svn://svn.code.sf.net/p/ultrastardx/svn/trunk \
  ultrastardx-svn
$ cd ultrastar-svn
$ ./configure
```

Debido a un error del compilador, la compilación no funcionará hasta que "-O2" sea eliminado de la opción PFLAGS_ RELEASE_DEFAULT en la línea 102 de src/Makefile. Para conseguir que funcione con OpenGL, también hay que quitar o comentar las líneas desde la 4323 a la 4330 en el archivo src/lib/JEDI-SDL/ OpenGL/PAS/glext.pas:

```
//   @glCopyTexSubImage3D := SDL_GL_GetProcAddress('glCopyTexSubImage3D');
//   if not Assigned(glCopyTexSubImage3D) then Exit;
//   @glDrawRangeElements := SDL_GL_GetProcAddress('glDrawRangeElements');
//   if not Assigned(glDrawRangeElements) then Exit;
//   @glTexImage3D := SDL_GL_GetProcAddress('glTexImage3D');
//   if not Assigned(glTexImage3D) then Exit;
//   @glTexSubImage3D := SDL_GL_GetProcAddress('glTexSubImage3D');
//   if not Assigned(glTexSubImage3D) then Exit;
```

Luego, compila el programa:

```
$ make
$ make install
```

Como antes, es posible que quieras ejecutar checkinstall -D para crear un archivo .deb instalable si usas Debian o Ubuntu.

Si recibe errores con las versiones de libavutil, exporta esas versiones utilizando variables de entorno y configura de nuevo. Asegúrate de retirar la optimización en PFLAGS:

```
$ export libavutil_VERSION=52.48.101
$ export libavcodec_VERSION=55.39.101
$ export libavformat_VERSION=55.19.104
$ export libavdevice_VERSION=55.5.100
$ export libavfilter_VERSION=3.90.100
$ export libswscale_VERSION=2.5.101
$ export libswresample_VERSION=0.17.104
$ ./configure
```

Ahora puedes iniciar ultrastardx, notaras que irá un poco lento ya que OpenGL tiene que ser emulado por software. Esto se puede solucionar con ayuda de EGL.

Configuración EGL

Iniciar ultrastardx con un empaquetador a menudo da un error al activar EGL, pero a veces funciona sin hacer cambios en la configuración. Escribe los siguientes comandos para descargar y compilar el empaquetador:

```
$ git clone git://github.com/lunixbochs/glshim
$ cd glshim
$ cmake .
$ make GL
$ git clone git://github.com/lunixbochs/glues
$ cd glues
$ cmake .
$ make
```

Copia las librerías dinámicas a /usr/local/lib. Normalmente, sólo tiene que exportar el LD_LIBRARY_PATH a esta carpeta e iniciar el programa. Sin embargo, esto no siempre funciona, así que aquí tienes mi script que a veces necesita entre 5 y 10 intentos. Cuando funciona correctamente, la experiencia de juego es muy rápida y estable:

```
#!/bin/bash
export LD_LIBRARY_PATH=/usr/local/lib
while true
  do
  ultrastardx
  if [ "$?" -ne 0 ]; then
    break
  fi
done
```

Conguración juego

Aquí tienes la parte gráfica de mi fichero de configuración config.ini que parece funcionar mejor:

```
[Graphics]
Screens=1
FullScreen=On
Visualization=Off
Resolution=640x480
Depth=16 bit
TextureSize=256
SingWindow=Big
Oscilloscope=Off
Spectrum=Off
Spectrograph=Off
MovieSize=Full [BG+Vid]
VideoPreview=On
VideoEnabled=On
```

Notas de compilación

Si la compilación no se completa correctamente, puede que tenga que instalar algunos paquetes de desarrollo:

```
$ apt-get install \
  libsqlite3-dev \
  portaudio19-dev \
  libsdl-image1.2-dev
```

Tras la compilación y la instalación, inicia Ultrastar con el siguiente script:

```
#!/bin/bash
export LD_LIBRARY_PATH=/usr/lo-
cal/lib
ultrastardx
```

Si tiene preguntas, comentarios o sugerencias, puedes consultar el post original en http://bit.ly/1bYvbp7.

TEKKEN 6
EL MEJOR JUEGO DE LUCHA

por Justin Lee

Tekken 6 es un juego de lucha de artes marciales muy popular que funciona muy bien en cualquier ODROID. Puedes comprobarlo en el video de Hardkernel donde PPSSPP emula Tekken 6 en http://bit.ly/1f1BqdX. Para jugar al Tekken 6 en tu ODROID, primero descarga e instale la última imagen de Android para tu dispositivo, luego sigue estos pasos:

1. Cambia la resolución a 1280x720 HD utilizando ODROID-Utility
2. Cambia la configuración de la CPU de Governor a Performance
3. Instala la aplicación PPSSPP, uno de los mejores emuladores de PSP
4. Configura los ajustes PPSSPP:

Graphics menu
Rendering Mode: Non-buffered rendering (Speedhack)
Simulate block transfer (unfinished): Check
Framerate control
Frameskipping: 2
Auto frameskip: Check
Prevent FPS from exceeding 60: Check
Alternative speed: 0

Features
Postprocessing shader: Off
Strech to display: uncheck
Small display: uncheck
Immersive mode: Check

Performance
Rendering resolution: 1xPSP
Display resolution (HW scaler): 2xPSP
Mipmapping: Check
Hardware transform: Check
Software skinning: Check
Vertex cache: Check
Lazy texture caching: Check
Retain changed textrue: Check
Disable slower effects: Check

Hack settings
Disable alpha test: Check
Texture coord speedhack: Check
Show FPC count: Both

Controls menu
On-screen touch controls: Uncheck

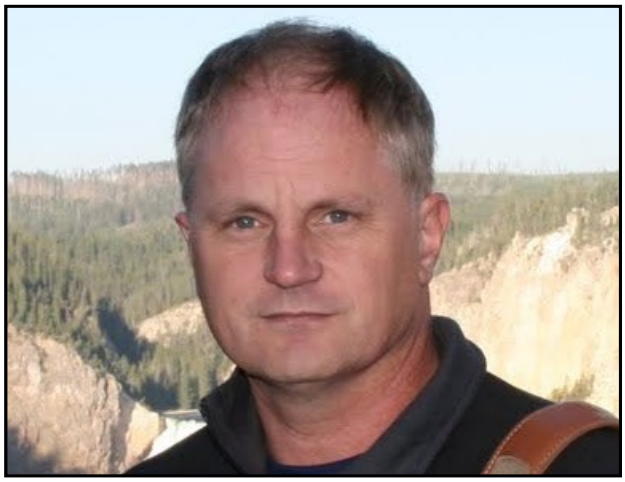
System menu
Multithreaded: Check

Tekken 6 combina la acción rápida con impresionantes movimientos de artes marciales

CONOCIENDO A UN ODROIDIAN

MARKHAM THOMAS (@MLINUXGUY) - UN EXPERTO EN LINUX CON MUCHA EXPERIENCIA QUE VIAJA A MENUDO

editado por Rob Roy



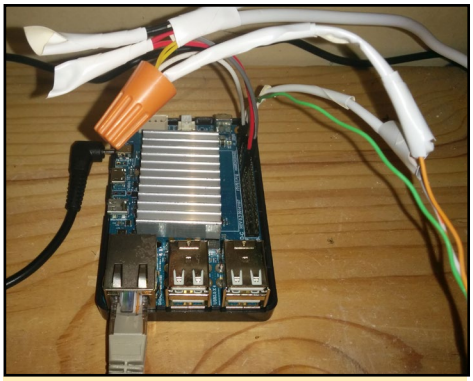
Desde la codificación a resolución de problemas ¡puedes contar con Markham para todo tipo de cuestiones!

Por favor, háblanos un poco sobre ti.

Vivo en el centro de Oklahoma, y trabajo en el campo de la tecnología con gente de todo el mundo. Nos comunicamos a través de salas de chat, salas virtuales y conferencias de voz. Tengo 3 gatos que piensan que los teclados son para pisotearlos y que los ODROIDs son para dormir, además de mi pareja. Vivimos en una casa diseñada a medida con características muy particulares, como un techo de metal, calefacción y refrigeración geotérmica, un diseño de mi padre arquitecto que se asemeja algo que Frank Lloyd Wright había hecho. Tiene un montón de espacios ocultos, rampas y travesaños para que los gatos se suban.

¿Cómo empezarte con los ordenadores?

Me compré mi primer ordenador a los 14 años con el dinero ahorrado de transportar heno en los campos. Fue un TRS-80 Modelo III. Mi madre le dijo a mi abuela que su hijo había desperdiciado 1.400 dólares que debería haberse ahorrado para la universidad en una calculadora. En unos años, equipe el TRS-80 con más memoria RAM (64k) y con unidades de doble disco y logre aumentar la frecuencia del reloj a 5Mhz con un interruptor en el lateral. Todavía recuerdo el primer mensaje que aparecía cuando lo encendia: "¿Cass? No tenía ni idea de lo que significaba (cargador de cinta de cassette), pero con el tiempo fui a la universidad, y empecé a escribir código ensamblador Z80. Las Revistas de la época proporcionaban ejemplos de código y circuitos, que es donde obtenías las especificaciones de cómo hackear los sistemas, puesto que no había Internet.



Un ODROID-CI monitorizando temperaturas geotermales en el sótano de Markham

El dominio del código ensamblador Z80 me consiguió un trabajo en la Universidad, donde desarrollábamos y codificábamos gigantes pruebas hidráulica para la industria pesada aeroespacial. Los códigos hexadecimales se introducían a mano en EEPROMs de 2 KB de tamaño. Todavía puedo recordar los códigos hexadecimales para muchas instrucciones Z80. Más tarde, creamos hardware para conectar el sistema de desarrollo CP/M al programador EEPROM y así acabar con la entrada manual.

Tras la universidad, me fui a trabajar a una empresa de tecnología y continué trabajando con los más modernos sistemas informáticos, aunque sería raro volver a experimentar la emoción de explorar aquellos sistemas, donde se podía rastrear circuitos y conseguir que hicieran cosas para las que nunca fueron creados.

¿Qué te atrajo de la plataforma ODROID?

Durante años, he construido un nuevo PC cada 18 meses, pero cuando llego Windows 7, perdí el interés en actualizar y modificar los PCs, y empecé a buscar sistemas más pequeños en los que pudiera ejecutar Linux. Estuve investigando los procesadores ARM con la idea de desarrollar una placa que ejecutara Linux cuando salió la Raspberry PI. Inmediatamente me hice con cuatro de ellas y empecé a probarlas. Alcance sus límites bastante rápido, así que mientras desarrollaba proyectos con la Raspberry Pi, seguí buscando placas mejores. Una vez que descubrí el ODROID-X, conseguí uno y empecé analizar sus capacidades.

Lo que hace que los ordenadores ODROID sean para mi mucho más interesante que las placas de la competencia es el soporte y su comunidad dinámica. La comunidad de Raspberry Pi es grande, pero a veces carecen de las rápidas respuestas que se obtienen con la comunidad de ODROID. Fue fascinante ver el potencial del ODROID-X tras su lanzamiento, pudiendo dar pie a una revolución como la del PC hace décadas.

Cuál es tu ODROID favorito?

Mi favorito es el ODROIDC1, tengo cinco y una unidad de cada uno del resto de ODROIDs. Me gusta porque existe gran cantidad de código que se le puede fácilmente adaptar desde la Raspberry Pi. Su CPU tiene una geometría más pequeña que sus competidores. Me permite dispone de una maquina Linux rápida, de bajo consumo y sin ventilador, es lo suficientemente barato como para poder enviarlo en una cometa o colocarlo en el tejado como una estación meteorológica, no importa si falla dentro de unos años.

Describa tu configuración ODROID y cómo lo usas.

Resulta difícil elegir un único proyecto. Mi oficina en casa está muy iluminada con luces parpadeantes de todos los ODROIDs y otras plataformas. Tengo una placa de desarrollo Linux para modificar el kernel y sus drivers, y otra dedicada aprobar varias tarjetas de expansión.

También tengo uno en el sótano que mide las temperaturas del circuito geotérmico, y uno en el tejado que aprovecha Arduino para leer los datos de la estación meteorológica. Posiblemente el ODROID más interesante es el que está midiendo los datos geotérmicos. Monitorizo la temperatura tanto de entrada como de salida del circuito, la temperatura exterior, los niveles de luminosidad solar y ultravioleta (UV), y elabora gráficas que me ayudan a interpretar el eficacia de mi circuito cuando varía las condiciones ambientales.

La mayoría de mis ODROIDs ejecutan Ubuntu. Tengo uno en desarrollo con un monitor para poder llegar a él tras destrozar el sistema de red o interrumpir el kernel. Una Raspberry Pi se encarga de ejecutar una consola UART por el puerto serie bajo una pantalla de comandos para poder conectarme por SSH a la Pi y ver que hace que falle el kernel del ODROIDC1

Tengo otro C1 que sustituye a una Pi y que funciona a modo de grabadora de radio FM. Este sistema controla una tarjeta sintonizadora FM de bajo coste a través de SPI y traslada el audio a una tarjeta de sonido USB permitiendo grabar diferentes programas de radio en archivos MP3. Luego los cargo en un reproductor para escucharlos mientras corro. Puedo descargar los MP3 de sistema, pero no consigo que los guarde automáticamente en un recurso de red compartido. El C1 tiene potencia más que suficiente para grabarlos con altas tasas de bits.

Eres muy generoso compartiendo tus conocimientos sobre hardware, ingeniería eléctrica y programación Linux en los foros ODROID ¿Cómo llegaste a ser tan hábil?

He estado haciendo esto durante mucho tiempo, pero esto por sí sólo no me hace ser un experto o necesariamente bueno en ello. Creo que lo que necesitas es que te guste desmontar cosas para ver que puede o no funcionar. Aunque me especialicé en Electrónica en la universidad, yo nunca llegue a usarla. En cambio, pasaba la mayor parte de mi tiempo con el software, depurando problemas de sistemas.

Con esta revolución de ordenadores, quiero profundizar más en la tecnología, de modo que suelo elegir aquellas funciones que me interesa, como son el rendimiento de red, intento comprender cómo funciona y cómo podría mejorarlo. Luego pruebo los cambios realizados.

Hace mucho tiempo, escribí drivers SVGA para DOS en lenguaje ensamblador, donde tenía que contabilizar cada ciclo de instrucción y optimizarlo. Esta experiencia me ha sido muy valiosa a la hora de mejorar el rendimiento del código. No hay nada como la retroalimentación instantánea observando las optimizaciones realizadas en pantalla.

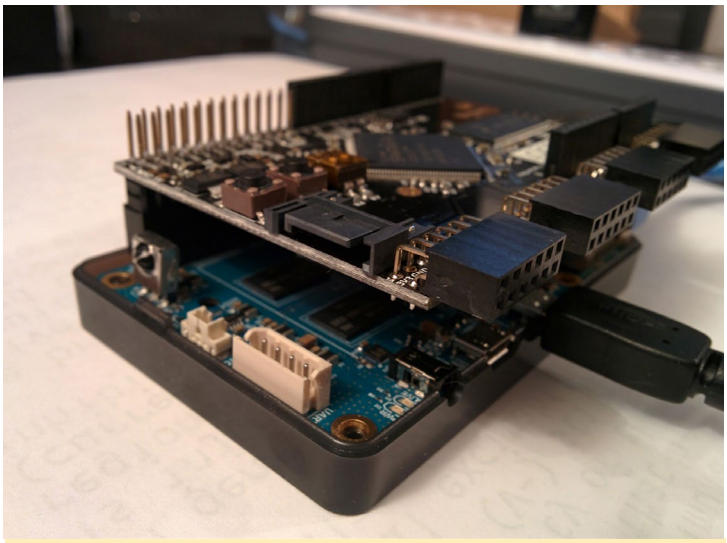
Mis conocimientos en ingeniería están jugando un papel muy importante, ya que, tras varios años analizando los problemas de caja negra en Linux y Unix (problemas de rendimiento o código de alto nivel), he empezado a trabajar con FPGAs, y ahora por fin disponemos de la descripción completa de cómo todos los componentes de hardware interactúan para formar un sistema.

¿Qué aficiones e intereses tienes aparte de los ordenadores?

Pasé años como corredor, pero he reducido esa actividad para jugar al tenis, un deporte que se juega en condiciones extremas en Oklahoma. Rara vez se ve en la televisión los profesionales persiguiendo su propia bola a lo largo del campo de juego como lo hacemos aquí.

He hecho senderismo y he subió a la mayoría de las montañas más altas del suroeste de los Estados Unidos. Me gusta esquiar y tengo un catamarán de 5,5 metros que tengo que restaurar antes de usarlo. Cuando hago senderismo, siempre llevo mi cámara conmigo que pesa más que mi mochila, pero ahora sobre todo hago fotografía aérea desde una cometa o uno de mis drones.

Uno de los proyectos que todavía



Placa LogiPi FPGA conectada aun ODROID-C1, usada pra aprender a programar con FPGA

Markham disfruta de una larga caminata en el Parque de Yellowstone

tengo en mi lista de tareas pendientes es enviar un ODROID en una cometa, y codificar el proceso de estabilización de imagen para un módulo de 9 ejes. Actualmente, sólo puedo disparar imágenes cada 30 segundos y escoger las mejores.

Soy miembro de la Sociedad Planetaria, he ayudado a financiar muchos de sus proyectos, con su Light-Sail siendo uno de los últimos. Tiendo a seguir el desarrollo en áreas como CubeSat y otros proyectos que llevan el acceso al espacio al ámbito de la comunidad.

¿Está involucrado en otros proyectos informáticos a parte de ODROID?

He estado trabajando con varias placas FPGA incluyendo algunas con Linux como mi placa Parallella. Sin embargo, yo no tengo planeado ningún proyecto en particular con ellas más allá de simple aprendizaje de los circuitos digitales.

Tengo todos los componentes en mi taller para hacer nueve placas TRNG Infinite Noise. Realizaré algunas pruebas una vez que consiga uno para compararlo con el generador de números aleatorios LSFR en el ODROID-C1.

Tengo un MakerBot 2 que he modificado con una placa de desarrollo y brazos de aluminio, tapas laterales y la parte superior. Lo uso para imprimir carcasas para mis diversas placas ARM, así como para hacer estructuras para varios proyectos como una plataforma estabilizadora de 3 ejes que incluirá la cámara para la cometa.

También he estado haciendo placas PCB caseras utilizando mi LaserJet y un laminador. Tengo una placa concentradora para limpiar el cableado de mi estación meteorológica del tejado.

El próximo gran proyecto será coger mi tractor diesel de 25HP y reemplazar sus medidores por paneles LCD impulsados por un Arduino. Mi sobrino quiere modificar el turbo así que necesitará sensores adicionales.

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

El próximo gran salto en las placas de desarrollo que estoy ansiosamente esperando es la llegada de las placas de 64 bits. Me encantaría que Hardkernel liberara una, pero hasta que eso ocurra, hay características que faltan y que serían muy útiles.

La incorporación de una interfaz de red de 1GB en el C1 fue una gran mejora, sin embargo, añadir un puerto de alta velocidad como el SATA o USB 3.1 permitiría más proyectos. Una característica innovadora de la Beaglebone es su Unidad en Tiempo Real Programable (PRU), algo similar para una placa Hardkernel podría ser un chip Arduino o la forma de añadir uno tú mismo.

El mercado de la expansión y la arquitectura de plataformas abiertas es una de las cosas que impulsaron la revolución del PC, así que tener la ODROID-C1 con un cabezal de expansión compatible con Raspberry Pi mejoraría considerablemente el valor de la placa Hardkernel. Me gustaría ver futuras placas que tuviesen un cabezal similar, incluso si es sólo el cabezal pre-B+

Una interfaz de expansión de alta velocidad también estaría bien, pero teniendo en cuenta el coste, tal vez la nueva especificación USB 3.1 daría suficiente rendimiento y soporte para periféricos rápidos. Esto sería un complemento ideal para las nuevas placas de Hardkernel.

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

Hay muchos recursos gratuitos en Internet para cualquier persona que quiera aprender más sobre la programación. No dejes que la falta de familiaridad con la codificación o con un lenguaje en particular te frene. Elije un proyecto, selecciona tu idioma y empieza buscar en Internet un poco de código básico para empezar. Una vez que tenga el código, empieza a darle contenido utilizando ejemplos. Yo utilizo un monitor 4K, con 10 o más ventanas abiertas del navegador para modificar código

Te recomiendo que elijas un proyecto de código abierto que te guste y añádele una función, es posible que puedas ser aceptado en el proyecto. GitHub es un gran recurso para encontrar este tipo de proyectos, puedes crear nuevos proyectos o simplemente contribuir con características.

Lo complicado es saber cómo programar sin entender cómo funcionan los microprocesadores, de modo que puede terminar escribiendo código ineficiente. Tómate tu tiempo para perfilar tu aplicación e incluye código ensamblador y mixto cuando lo perfiles, de modo que puedes ver como el compilador implementa tu código.

Si realmente quieres comprender cómo funciona el hardware, como son las pruebas de bits o los cambios de bits, el mejor mandra sería aprender lógica boolena y hacer algunos cursos de FPGA online. Luego, profundiza en algunos de los códigos de http://opencores.org para ver realmente cómo funcionan los circuitos de ordenador.