IRQ CPU • Servidor Multimedia • MCI • Gnome • I2C • Punto G

ODROID Magazine

Año Siete
Num. #74
Feb 2020

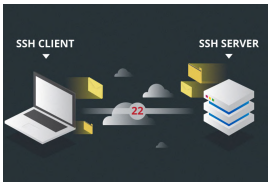
FUNCIONES AVANZADAS QUE PUEDES USAR CON TU ODROID

Virtualización KVM

ODROID-GO: ANALISIS DE RENDIMIENTO USANDO ARM STREAMLINE

SECURE SHELL EN ODROID: A TUTORIAL PARA TI

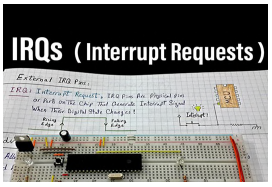
PETITBOOT Y OTG USB: UN GETOR DE ARRANQUE INDEPENDIENTE BASADO EN EL REINICIO DE LINUX KEXEC



Cómo Conectarse de Forma Remota con Secure Shell

February 1, 2020

Utilizar SBCs puede ser una alternativa económica para ejecutar pequeñas aplicaciones de servidor en tu casa o negocio. Así mismo el tema de configurarlos puede llegar a ser tedioso si necesitas compartir un solo cable HDMI entre varios dispositivos. Afortunadamente, la mayoría, si no todas, las distribuciones de Linux abren ▶️



Configurando la Compatibilidad CPU IRQ: Mejorando el Rendimiento de IRQ en el ODROID-XU4

February 1, 2020

Recientemente me he topado con una publicación en la subreddit ODROID que muestra un artículo que ofrece consejo para ajustar el ODROID-XU4. El artículo fue escrito originalmente en alemán y luego fue traducido al inglés y publicado en ODROID Magazine. Puesto tengo un ODROID-XU4 desde hace mucho tiempo, la mayoría de los ▶️

Análisis de Rendimiento Avanzado de ODROID-GO: Usando ARM Streamline

February 1, 2020

Streamline es una herramienta gráfica de análisis de rendimiento que muestra datos a modo de informes tanto en formato visual como estadístico.

Una Carcasa para el ODROID-MC1 Solo: No, no se Trata de una Justificación para Montar tu Propio Clúster SBC; Se Trata de una Carcasa Protectora Transparente por 1$

February 1, 2020

Si está buscando una buena carcasa para proteger tu ODROID-MC1 Solo, no busques más allá del sitio web de Hardkernel

KVM Fun con virtualización en ODROID-H2 - Funciones Avanzadas

February 1, 2020

En mi último artículo, demostré que es bastante fácil instalar y configurar KVM e incluí algunas herramientas para crear y controlar máquinas virtuales directamente en Linux. En esta ocasión, quisiera hablar de algunas características avanzadas que puedes usar con KVM de forma gratuita, que en otros hipervisores solo están disponibles ▶️

Ejecutando GNOME Desktop en el ODROID-N2

February 1, 2020

Este artículo versa sobre cómo ejecutar el Escritorio GNOME en un ODROID-N2 con un kernel Linux v5.4 ascendente. Afortunadamente, el kernel ascendente tiene muchos parches que hacen que el ODROID-N2 funcione sin problemas.

Instalación del Sistema Operativo Utilizando Petitboot y USB OTG

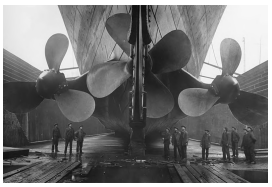
February 1, 2020

Petitboot es un gestor de arranque independiente de plataforma basado en el mecanismo de reinicio en caliente kexec de Linux.

Usando I2C en ODROIDs con Android Things

February 1, 2020

Esta es la continuación del artículo inicial del número de Enero 2020 titulado "Android Things", en el que se detalla cómo usar el nuevo sistema operativo de Google que facilita el uso de los pines GPIO en dispositivos ODROID.



Regulador del Ventilador ODROID-XU4

February 1, 2020

Este artículo trata sobre el regulador del ventilador Hysteresis de ODROID-XU4. Cuando se activa el ventilador, permanece encendido por un tiempo. Enfría la CPU lo suficiente para luego apagarse. A medida que la CPU se calienta, el ventilador gira más rápido.

El Punto G: Tu Destino para Todas las Cuestiones Relacionadas con Juegos Android: Los Juegos de Mesa no son tan Aburridos como Parece

February 1, 2020

¿Piensas que los juegos de mesa son aburridos? Estas actividades de entretenimiento analógico en papel tienen una larga trayectoria, o esta tradición se ha desvanecido y ha sido reemplazada por los juegos Android de ODROID de hoy en día

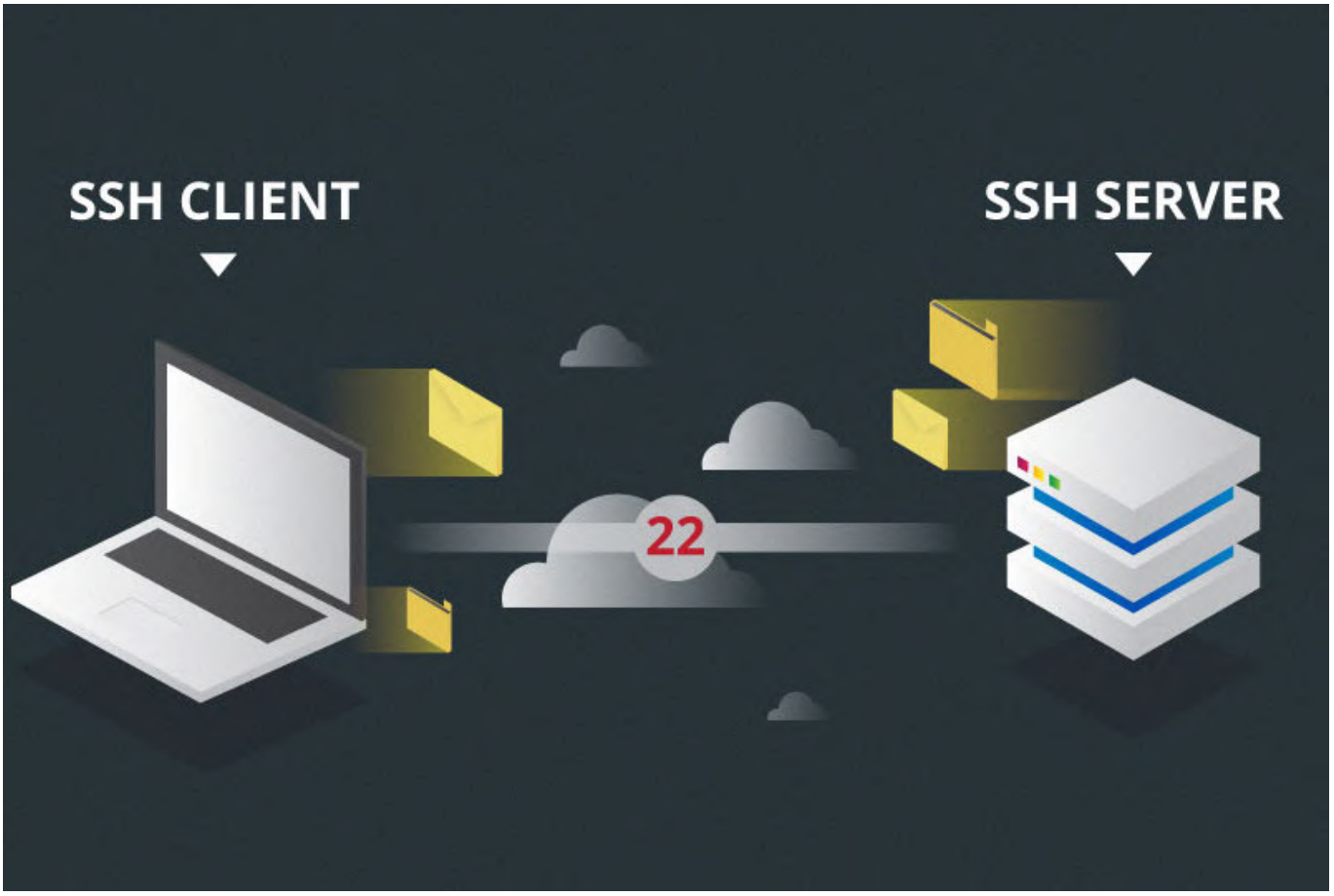
Las Mejores Opciones de Software como Servidor Multimedia ODROID-XU4

February 1, 2020

Aunque la Raspberry Pi es un ordenador de placa reducida (SBC) muy popular, el ODROID-XU4 es un sólido competidor. Manteniendo un tamaño físico pequeño, el ODROID-XU4 tiene un gran rendimiento. Potente y muy eficiente desde el punto de vista energético, con un procesador ARM big.LITTLE, el ODROID-XU4 incluye una CPU ▶️

Cómo Conectarse de Forma Remota con Secure Shell

February 1, 2020 By Miguel Alatorre, www.ameridroid.com Linux, ODROID-C2, Tutoriales



Utilizar SBCs puede ser una alternativa económica para ejecutar pequeñas aplicaciones de servidor en tu casa o negocio. Así mismo el tema de configurarlos puede llegar a ser tedioso si necesitas compartir un solo cable HDMI entre varios dispositivos. Afortunadamente, la mayoría, si no todas, las distribuciones de Linux abren el puerto 22 para conexiones seguras de intérprete de comandos, también conocidas como SSH.

En primer lugar, tanto el PC como el SBC, ya sea un ODROID-C2 u otro SBC, deben estar conectados a la misma red. En segundo lugar, necesitas descargarte un cliente SSH. Una opción muy conocida para el sistema operativo Windows es PuTTY, que puedes descargar desde: https://bit.ly/2TVHZq5. Para usuarios más avanzados, hay disponibles clientes como TeraTerm (https://osdn.net/projects/ttssh2/releases/) Ahora debes localizar la dirección IP del SBC. Esto lo puedes hacer con algún escáner IP como Advanced IP Scanner, disponible en: https://www.advanced-ip-scanner.com/.

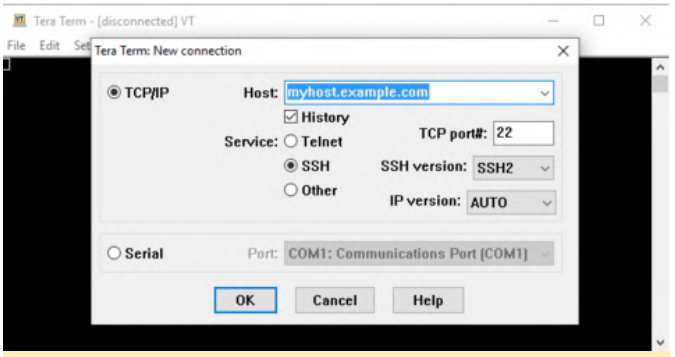


Figura 1 - Después de abrir el cliente, aparecerá una interfaz similar a esta

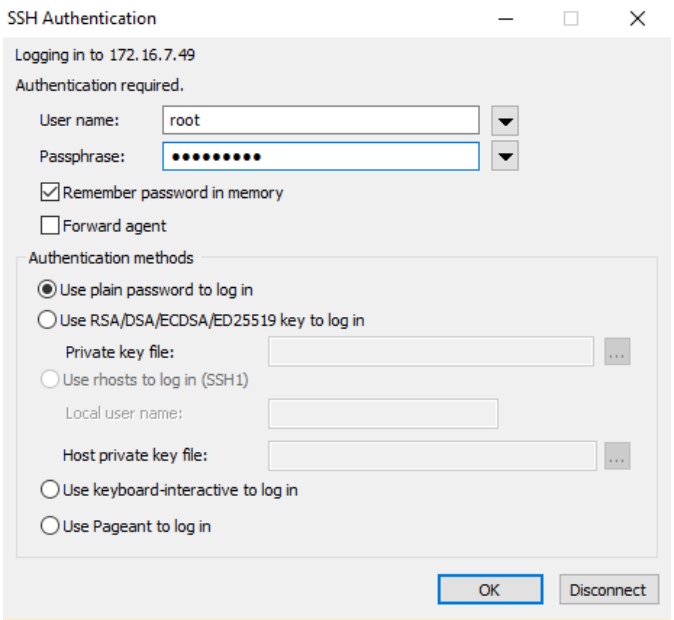


Figura 2 - Introduce la dirección IP del SBC en la sección Host, luego presiona "OK". A continuación, te pedirá un nombre de usuario y contraseña

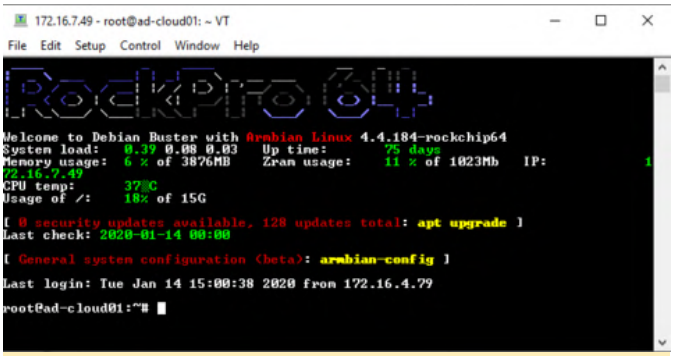
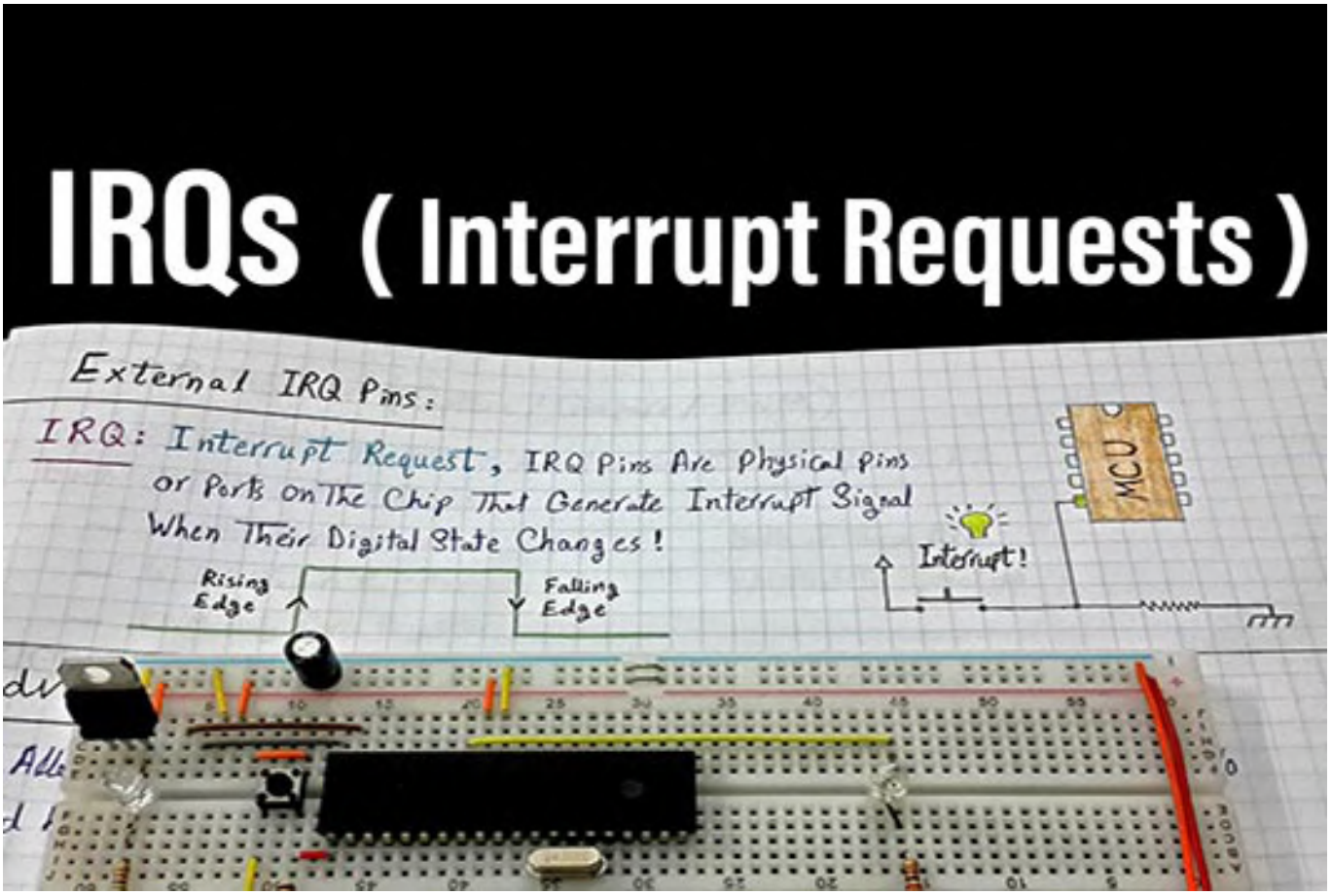


Figura 3 - Tras introducir tus credenciales, se abrirá el terminal del SBC

Ahora, todo lo que queda es interactuar con el terminal. Referencia https://bit.ly/36oWwxo

Configurando la Compatibilidad CPU IRQ: Mejorando el Rendimiento de IRQ en el ODROID-XU4

February 1, 2020 By @Wallace Linux, ODROID-XU4, Mecaniqueo, Tutoriales



Recientemente me he topado con una publicación en la subreddit ODROID que muestra un artículo que ofrece **consejo para ajustar el ODROID-XU4**. El artículo fue escrito originalmente en alemán y luego fue traducido al inglés y publicado en **ODROID Magazine**. Puesto tengo un ODROID-XU4 desde hace mucho tiempo, la mayoría de los consejos ya los conocía, ya que están publicados en los foros de ODROID desde hace bastante tiempo. Sin embargo, aparece un consejo que desconocía, me llamado bastante la atención y no en el buen sentido.

IRQs

Las IRQs (peticiones de interrupción) permiten que el hardware acceda a la CPU incluso cuando está ocupado haciendo otra cosa. De modo que, nuestros teclados, ratones y redes, por ejemplo, no dejan de funcionar si tenemos nuestra CPU funcionando al máximo.

Cualquiera que lleve usando ordenadores bastante tiempo, le será familiar el fenómeno en el que el ratón y el teclado se bloquean momentáneamente, se retrasan o dejan de responder por algún tiempo cuando la CPU lleva a cabo una tarea muy intensa. Esto era bastante común en los primeros ordenadores y se ha vuelto prácticamente inexistente a medida que han ido apareciendo CPUs más potentes, han evolucionado los sistemas operativos y se ha introducido la arquitectura APIC.

Para lograr el mejor rendimiento en términos absolutos de los periféricos de hardware en un sistema de múltiples núcleos, debemos asegurarnos de que estamos dirigiendo las IRQ al núcleo más inactivo, aumentando las posibilidades de que se ejecuten en el acto. En los sistemas de chips Arm big.LITTLE (como el ODROID-XU4) es más probable que obtengamos una mejor capacidad de respuesta

para IRQ con los núcleos "grandes", lo cual tiene bastante sentido.

IRQs en Linux

Para obtener una lista de IRQs y sus compatibilidades de CPU, simplemente debemos mirar dentro de /proc/interrupts. Así es como se aparece en mi ODROID-XU4 con Arch Linux ARM con kernel v4.14.157: Nota: El resultado es bastante largo, así que sólo mostraré la cabeza.

```
$ cat /proc/interrupts | head
CPU0 CPU1 CPU2 CPU3 CPU4 CPU5 CPU6 CPU7
49: 0 0 0 0 0 0 0 0 COMBINER 187 Edge mct_comp_irq
50: 8344372 0 0 0 0 0 0 0 GICv2 152 Edge mct_tick0
51: 0 5765406 0 0 0 0 0 0 GICv2 153 Edge mct_tick1
52: 0 0 4389485 0 0 0 0 0 GICv2 154 Edge mct_tick2
53: 0 0 0 3384898 0 0 0 0 GICv2 155 Edge mct_tick3
54: 0 0 0 0 55211190 0 0 0 GICv2 160 Edge
mct_tick4
55: 0 0 0 0 0 48058391 0 0 GICv2 161 Edge
mct_tick5
56: 0 0 0 0 0 0 33449904 0 GICv2 162 Edge
mct_tick6
57: 0 0 0 0 0 0 0 20020736 GICv2 163 Edge
mct_tick7
```

En el anterior resultado tenemos IRQs # 49-57 que parecen ser las marcas del reloj del sistema. Uno por cada uno de los 8 núcleos. Básicamente, cada IRQ tiene su propia ID y está vinculado a una única CPU.

La última declaración puede ser un tanto difícil de entender, así que echemos un vistazo a cómo se ven las interrupciones del lector de tarjetas SD y MMC:

Nota: Me doy cuenta que dw-mci son interrupciones para los dispositivos de E/S simplemente mirando el código fuente (**https://github.com/hardkernel/linux/blob/odroidxu4-4.14.y/drivers/mmc/host/dw_mmc-exynos.c**).

```
$ cat /proc/interrupts | grep dw-mci
83: 0 0 0 0 0 0 0 0 GICv2 107 Edge dw-mci
84: 103693202 0 0 0 0 0 0 0 GICv2 109 Edge dw-mci
```

IRQ #83 es para eMMC, que no tengo, y IRQ #84 es para la tarjeta MicroSD que obviamente está instalada.

Por defecto, todos las IRQs que no son de reloj de CPU están vinculados a todos los núcleos, aunque en realidad la CPU0 es la que se usará la mayor parte del tiempo, ya que simplemente es la primera. En mi versión de kernel del ODROID-XU4, la CPU0 es uno de los núcleos "pequeños". La forma más fácil de confirmar esto es observando la frecuencia máxima de CPU de cada núcleo, puesto que los "pequeños" se ejecutan a una velocidad menor:

```
$ cat
/sys/devices/system/cpu/cpu*/cpufreq/cpuinfo_max_f
req
1500000
1500000
1500000
1500000
2000000
2000000
2000000
2000000
```

Las primeras 4 CPU (=núcleos) funcionan a 1.5GHz y los últimos a 2GHz, que coincide con las velocidades de CPU Samsung Exynos5422 del ODROID-XU4, mis núcleos "grandes" funcionan a unos 100MHz más lentos.

Cambiar la compatibilidad CPU IRQ

Es bastante fácil cambiar la compatibilidad CPU de los IRQ. Todos los IRQ están enumerados en /proc/irq/ y la compatibilidad de cada uno se escribe correctamente dentro de smp_affinity y smp_affinity_list con el primero que contiene un valor hexadecimal y el último un valor decimal.

De modo que, para cambiar la compatibilidad de la CPU IRQ de nuestra tarjeta MicroSD, todo lo que tenemos que hacer es cambiar el valor de /proc/irq/84/smp_affinity_list a cualquier número de CPU que deseemos, por ejemplo, 5. Por supuesto, no podemos hacerlo como un usuario normal, así que tendremos que usar sudo. La forma más fácil de hacerlo es la siguiente:

```
$ sudo sh -c "echo 5 >
/proc/irq/84/smp_affinity_list"
```

Y podemos confirmar que funciona:

```
$ cat /proc/irq/84/smp_affinity_list
5
$ cat /proc/interrupts | grep dw-mci
```

```
83: 0 0 0 0 0 0 0 0 GICv2 107 Edge dw-mci
84: 103699631 0 0 0 152288 0 0 0 GICv2 109 Edge
dw-mci
```

Nota: Este valor no se mantendrá después de reiniciar, pero esta es la idea en general.

Los "ajustes"

Volviendo a donde empezamos, el artículo sugiere hacer exactamente lo que he dicho antes, entonces, ¿por qué no estoy satisfecho con ello? Por el uso del artículo de irqbalance. Dejando de lado la mala elección de usar /etc/rc.local para aplicar este ajuste, el primer paso fue el que más me llamó la atención:

```
$ systemctl disable irqbalance
```

¿Existe un programa dedicado cuyo objetivo es equilibrar la IRQ y lo vamos a desactivar? Suena extremadamente sospechoso.

Lo que pensé al instante fue que quizás irqbalance no permitía limitar sus asignaciones especificas a la CPU y, por lo tanto, desactivarlo tendría sentido. Sin embargo, no era el caso. Mirando la página del manual de irqbalance, Existe una variable de entorno:

IRQBALANCE_BANNED_CPUS

que puede indicarle al programa que evite asignar IRQ a esas CPU; que es exactamente lo que queremos.

El valor de esta variable de entorno es una máscara hexadecimal. Simplemente necesitamos decir qué CPU queremos activar y cuáles no. Cada CPU está encendida o apagada (= 1 o 0) y en nuestro caso queremos apagar las primeras cuatro y dejar las últimas encendidas. Eso significa que nuestra máscara en binario sería

```
00001111
```

El valor debe ser hexadecimal, que es una conversión bastante fácil de binario en este caso: OF.

Todo lo que tenemos que hacer ahora es ajustar nuestra variable de entorno en 0F (o simplemente F, ya que el 0 inicial no tiene significado). Probemos esto para asegurarnos de que las matemáticas son las correctas:

```
$ sudo su
$ export IRQBALANCE_BANNED_CPUS="f"
$ irqbalance -d
This machine seems not NUMA capable.
Isolated CPUs: 00000000
Adaptive-ticks CPUs: 00000000
Banned CPUs: 0000000f
...
Package 0: numa_node -1 cpu mask is 000000f0 (load
520000000)
Cache domain 0: numa_node is -1 cpu mask is
00000080 (load 90000000)
CPU number 7 numa_node is -1 (load 90000000)
Cache domain 1: numa_node is -1 cpu mask is
00000020 (load 100000000)
CPU number 5 numa_node is -1 (load 100000000)
Cache domain 2: numa_node is -1 cpu mask is
00000040 (load 130000000)
CPU number 6 numa_node is -1 (load 130000000)
Cache domain 3: numa_node is -1 cpu mask is
00000010 (load 200000000)
CPU number 4 numa_node is -1 (load 200000000)
```

- Primero cambiamos el usuario a root para que sea más fácil para nosotros.
- Luego, exporta la variable de entorno para irqbalance.
- Ejecutar irqbalance en modo depuración -d.
- Resultado: La primera parte muestra en CPU prohibidas que aceptaron nuestro valor. Luego, si observamos con detenimiento el resto de los resultados, podemos detectar la asignación a las CPU # 4-7 (5 a 8), que es exactamente lo que queríamos.

Para configurar esta variable de entorno para que la unidad systemd pueda acceder a ella, debemos analizada:

```
$ systemctl show irqbalance
```

Buscamos el valor de EnvironmentFile que podría ser cualquier cosa dependiendo del sistema operativo. En Ubuntu 18.04, es /etc/default/irqbalance y en mi sistema Arch es /etc/irqbalance.env. Probablemente ya exista una plantilla en esta ubicación, y todo lo que tenemos que hacer es asegurarnos de que no esté comentado y configurado el valor correcto.

Usar una ID de IRQ fija

El "tip" indica poner cada línea que corresponde a una ID de IRQ de un controlador de hardware diferente.

Sin embargo, las ID de IRQ no son consistentes y dependen de la versión del kernel. Por ejemplo, en mi sistema, las ID de IRQ # 103-105 se asignan a algunos pines gpio:

```
$ cat /proc/interrupts | awk '$1 ~ /103|104|105/'
103: 0 0 0 0 0 0 0 0 GICv2 110 Edge
13410000.pinctrl
104: 0 0 0 0 0 0 0 0 GICv2 78 Edge
14000000.pinctrl
105: 0 0 0 0 0 0 0 0 GICv2 82 Edge
14010000.pinctrl
```

Vincular cada interrupción a una única CPU

Por último, pero no menos importante, el artículo sugiere vincular cada interrupción a una CPU diferente. La tarjeta de red obtiene CPU4, el adaptador USB3 obtiene CPU5, etc. ¿Por qué molestarse en limitar las CPU puesto que nuestro núcleo puede elegir? ¿Qué sucede si un programa bloquea esa CPU específica durante un largo período de tiempo? El kernel no podría asignar la IRQ a una CPU diferente para evitar ralentizaciones.

Afterword

Aunque me gustan estas compilaciones de ajustes tanto como a ti, siempre tiendo a asegurarme de entender completamente lo que está haciendo cada ajuste y verificar dos veces para ver cómo se aplican a mi sistema y en mi caso en particular. Además, si existe una herramienta dedicada para un determinado propósito (como irqbalance para este asunto), primero deberíamos considerar usarla, de lo contrario, su existencia no estaría justificada.

El propósito de este artículo no es de ninguna manera ofender o condenar a u/blaumedia que escribió el artículo original, está destinado a crear conciencia sobre por qué los usuarios deben considerar su situación y comprender lo que están haciendo. Para más información, consulta el artículo original en **https://my-take-on.tech/2020/01/12/setting-irq-cpu-affinities-to-improve-performance-on-the-odroid-xu4/**.

Análisis de Rendimiento Avanzado de ODROID-GO: Usando ARM Streamline

February 1, 2020 By Joy Cho Desarrollo, ODROID-GO Advance

Streamline es una herramienta gráfica de análisis de rendimiento que muestra datos a modo de informes tanto en formato visual como estadístico. Utiliza contadores de rendimiento de hardware con métricas de kernel para proporcionar una representación precisa de los recursos del sistema objeto de análisis. Esta página wiki describe cómo configurar y ejecutar Streamline y cómo monitorizar el ODROID-GO Advanced.

Resumen

- Instalar DS-5 Community Edition en el PC host
- Compilar el módulo Gator y el Kernel
- Compilar el demonio Gator
- Iniciar Gator en el sistema objeto de análisis y Streamline en el PC host

La Figura 1 muestra una captura de ARM Streamline con ODROID GO Advanced. En unos 48s, se inicia glmark2-es2 y puede ver las transiciones gráficas de los componentes relacionados con Mali.

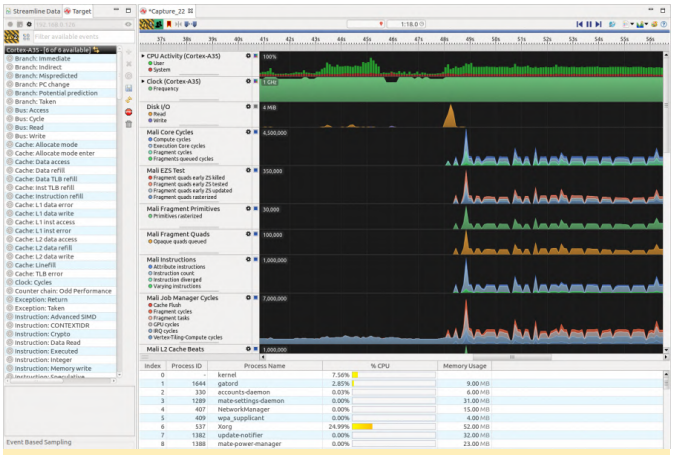


Figura 1 - Muestra de una captura ARM Streamline con ODROID GO Advanced

Usando ARM Streamline, puedes monitorizar los componentes principales de la CPU.

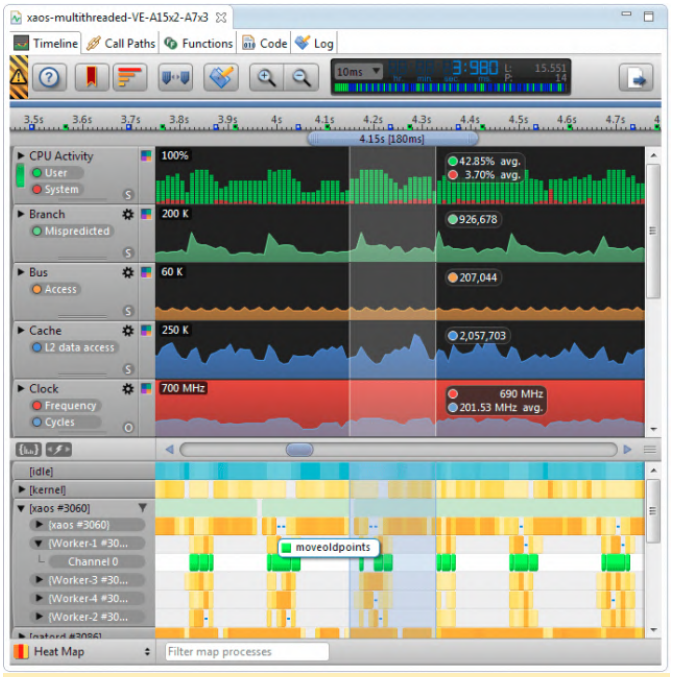


Figura 2: Muestra de captura de ARM streamline con información de CPU

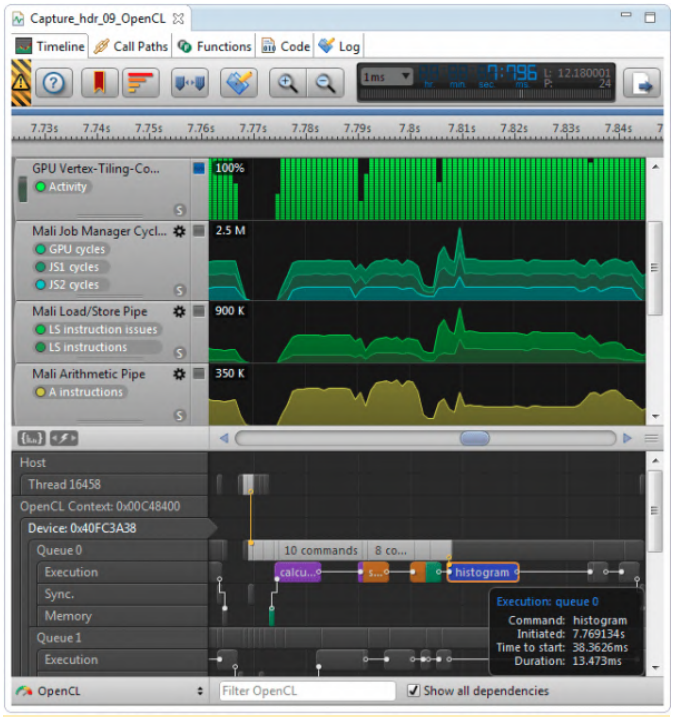


Figura 3 - Muestra de captura de ARM streamline con información de CPU

Instalar DS-5 Development Studio

Primero, descárgate DS-5 e instálalo en tu PC Host. Si no tiene ninguna licencia, puede usar DS-5 Community Edition. El instalador lo puedes encontrar en https://developer.arm.com/products/software-development-tools/ds-5-development-studio/editions/community-edition.

Actualmente, la última versión Streamline de DS-5 Community Edition es la v6.7.1, Ésta no soporta una versión superior de gator, por encima de v6.7.1. Solo DS-5 Ultimate, Professional Edition y DS-5 Development Studio admiten la última versión de Gator. Por lo tanto, ten en cuenta que la versión actual de gator para ODROID-GO Advanced será la v6.5.1, hasta que se actualice DS-5 Community Edition.

Compilar el módulo Gator y el Kernel

En la última versión de la imagen de ODROID-GO Advanced, no existe un módulo gator. Se incluirá en la próxima versión oficial, así que recurre a las siguientes opciones si aún no ha sido publicada una nueva versión del sistema operativo.

[Opción 1] Pre-compilar gator.ko:

```
odroid@odroid:~$ wget
https://dn.odroid.com/ODROID_GO_ADVANCE/gator.ko
```

[Opción 2] Compilar un kernel con el conjunto de peticiones relacionadas:

```
https://github.com/hardkernel/linux/commit/9b681d5
c4f23eac9bad17f75ef41a2c4fe4f698b
https://github.com/hardkernel/linux/commit/fcd6a0b
7a58792917c1cec64fc77ff981243b2ed
```

Compilar el demonio Gator del espacio de usuario

Para comunicarse con el dispositivo de destino, Streamline requiere que el demonio gator (gatord) se ejecute en el dispositivo. Aquí tienes las instrucciones para compilar el demonio gator en tu placa ODROID-GO Advanced.

Debes usar gator daemon v6.5.1 para sincronizar con el controlador gator de ODROID-GO Advanced. Además, necesitarás algunos parches para ejecutar demonio gator en ODROID-GO Advanced porque el nodo sysfs para leer la información de la GPU mali es diferente del existente en el github del software ARM.

```
odroid@odroid:~$ sudo apt-get install git
odroid@odroid:~$ git clone
https://github.com/JeonghwaCho/gator.git -b
odroid-rk3326

odroid@odroid:~$ cd ${path_of_gator}/daemon
odroid@odroid:~$ make
```

Ahora tendrás el binario gatord en el directorio ${path_of_gator}/daemon.

Iniciar Gator en el sistema destino y Streamline en la PC host

1. En el ODROID-GO Advanced Necesitas permiso de root para establecer una conexión ethernet y ejecutar gatord.

(1) Configurar conexión de red

```
odroid@odroid:~$ su
root@odroid:~# dhclient eth0
```

ODROID-GO Advanced conectado a través de Ethernet

(2) Desactiva el tiempo de espera de la tarea bloqueada del kernel

```
root@odroid:~# echo 0 >
/proc/sys/kernel/hung_task_timeout_secs
```

(3) Ejecuta el demonio gator del espacio de usuario y el módulo del kernel

```
root@odroid:~# ${gator_path}/gator/daemon/gatord -
m /lib/modules/$(uname
-r)/kernel/drivers/gator/gator.ko &
```

Puedes determinar si gator se está ejecutando:

```
root@odroid:~# lsmod | grep gator
gator 90112 1
```

La GPU Mali Bifrost de RK3326 se define en gator.ko, de modo que necesitarás obtener los eventos relacionados con ARM_Mali-Bifrost en /dev/gator/events/

```
root@odroid:~# ls /dev/gator/events/
ARM_Mali-Bifrost_Filmstrip_cnt0
ARMv8_Cortex_A35_cnt0
ARM_Mali-Bifrost_MMU_AS_0 ARMv8_Cortex_A35_cnt1
ARM_Mali-Bifrost_MMU_AS_1 ARMv8_Cortex_A35_cnt2
ARM_Mali-Bifrost_MMU_AS_2 ARMv8_Cortex_A35_cnt3
ARM_Mali-Bifrost_MMU_AS_3 ARMv8_Cortex_A35_cnt4
ARM_Mali-Bifrost_MMU_PAGE_FAULT_0
ARMv8_Cortex_A35_cnt5
ARM_Mali-Bifrost_MMU_PAGE_FAULT_1
ARMv8_Cortex_A35_freq
.....
.....
```

2. En el PC host A través de la interfaz Ethernet, puede conectar el dispositivo ODROID-GO Advanced a Streamline en tu PC host. Configura [Conexión] - [Dirección].

Figura 4 - Captura y análisis en Streamline

Si la conexión es reconocida con éxito, todos los contadores disponibles para ODROID-GO Advanced se mostrarán en este menú.

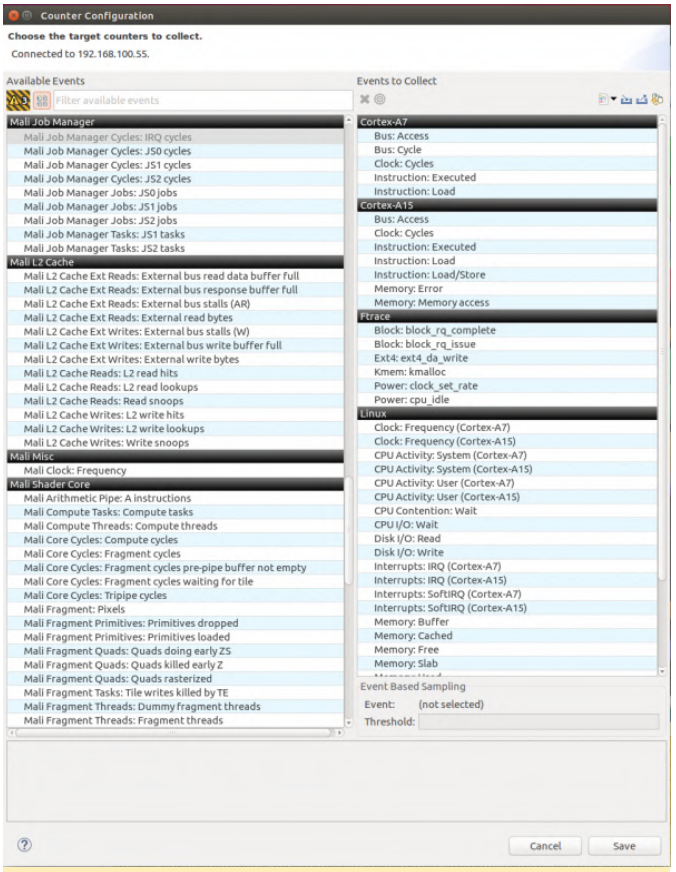


Figura 5 - Configuración del contador en Streamline

Más información

Para obtener información más detallada, consulta el sitio para desarrolladores de ARM y la guía del usuario

en **https://developer.arm.com/products/software-development-tools/ds-5-development-studio/**streamline y **https://static.docs.arm.com/dui0482/w/DUI0482W_streamline_user_guide.pdf**. La entrada wiki original la puede encontrar en la página wiki de Hardkernel en **https://wiki.odroid.com/odroid_go_advance/application_note/arm_streamline_on_goa**.

Una Carcasa para el ODROID-MC1 Solo: No, no se Trata de una Justificación para Montar tu Propio Clúster SBC; Se Trata de una Carcasa Protectora Transparente por 1$

February 1, 2020 By Dave Prochnow ODROID-MC1, Mecaniqueo

Si está buscando una buena carcasa para proteger tu ODROID-MC1 Solo, no busques más allá del sitio web de Hardkernel. Dentro de la "Dollar Shop" existe una carcasa transporte para el ODROID-HC1. Con un precio de, sí los has adivinado, 1$, esta carcasa es el compañero ideal para proteger tu clon ODROID-XU4.

Figura 1: Agrega una carcasa transparente a tu ODROID-MC1 Solo hace que destaque como una auténtica estrella informática.

Simplemente desliza la pieza más corta de la carcasa transparente ODROID-HC1 sobre el disipador térmico ODROID-MC1 Solo y dispondrás una funda protectora ajustada para mantener el polvo y los desechos lejos de la placa de circuito impreso (PCB). Además, esta carcasa está suficientemente ventilada para que el PCB permanezca fresco durante las extenuantes sesiones de trabajo.

Figura 2: Únicamente necesitas la mitad de la carcasa transparente ODROID-HC1 para proteger tu ODROID-MC1 Solo.

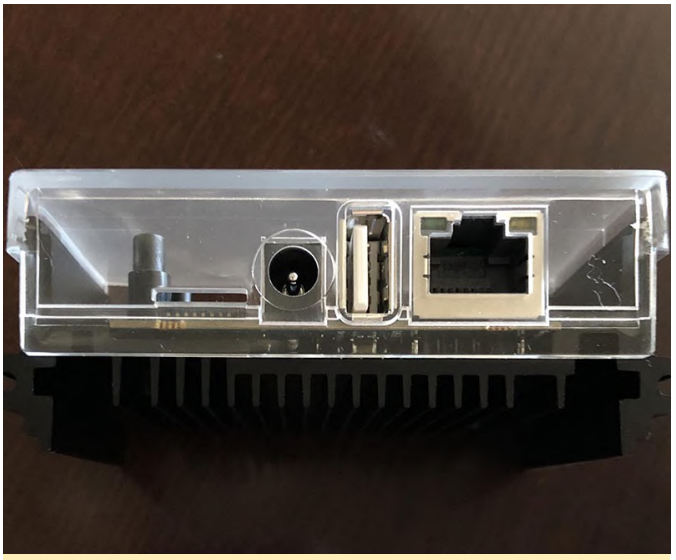


Figura 3: Todas las conexiones que quedan dentro de la carcasa aún son accesibles.

Finalmente, a la hora de comprar tu carcasa Dollar Shop, ten en cuenta el increíble precio de venta del ODROID-MC1 Solo. En el momento de la publicación, este potente nodo de clúster de ordenador de placa reducida de ocho núcleos tiene un precio de 9$. Se trata de un considerable ahorro de 39$ sobre el precio normal. Ah, y mientras llenas tu carrito de compras, no olvide incluir a tu pedido una fuente de alimentación de 5V / 4A, un cable Ethernet y una tarjeta microSD de 8GB. El costo total de este nodo informático completamente desarrollado será inferior a 25$ (excluyendo envío y manipulación). Esta compra

te ayudará a estar un paso más cerca de cumplir ese sueño de tener tu propio sistema de compilación.

Figura 4: Simplemente deja la puerta trasera abierta de tu sistema de compilación para mantener fresco el ODROID-MC1 Solo mientras realizas tus quehaceres rutinarios.

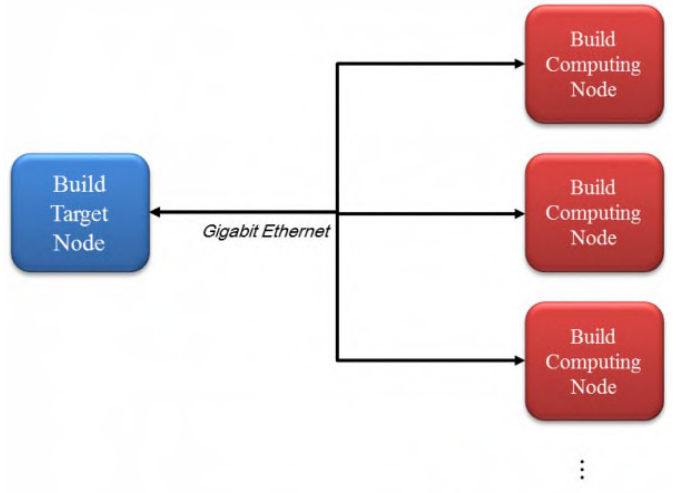


Figura 5 - Diagrama de compilación del sistema de compilación

Hay dos tipos de nodos para el sistema de compilación: nodo de compilación de destino y nodo informático de compilación. Para este ejemplo, utilizaremos el ODROID-HC1 para el nodo de destino de compilación. Este contiene los códigos fuente que se compilarán en este sistema de compilación. Un ODROID-HC1 es perfecto como nodo de destino de compilación porque puede tener una gran cantidad de espacio de almacenamiento con un disco duro conectado. Es muy útil para mejorar el rendimiento de la compilación si el HDD SATA (o SSD) tiene un buen rendimiento de E/S. El nodo informático de compilación recibe el archivo fuente y lo compila. Un ODROID-MC1 es la mejor opción para un nodo informático de compilación, ya que está optimizado para la informática distribuida.

Figura 6: El nodo de compilación de destino de ejemplo es un ODROID-HC1

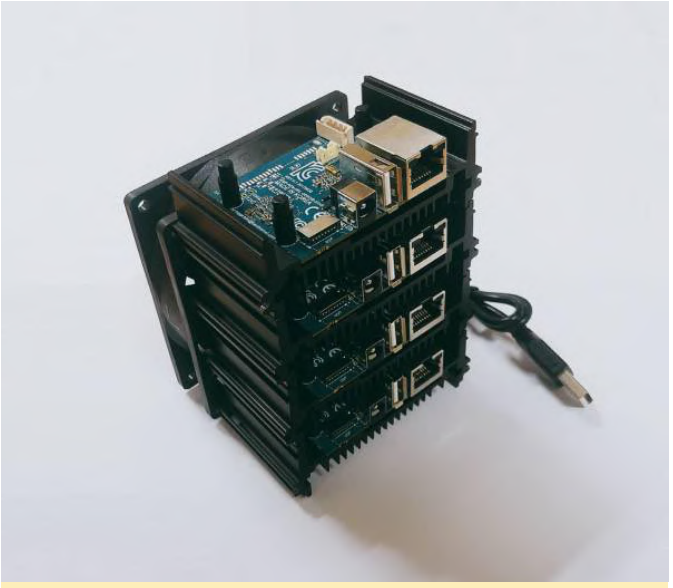


Figura 7: El nodo informático de compilación de ejemplo es un ODROID-MC1

Configuración del nodo informático de compilación

Para acceder a las consolas yjr del ODROID-MC1 y ODROID-HC1, debes hacerte con la dirección IP de los nodos. Consulta la sección sobre como arrancar el ODROID y localizar la dirección IP dentro de la página wiki de configuración de Headless. Todo lo que se necesitas es instalar distcc y configurarlo:

```
$ sudo apt update
$ sudo apt install distcc
$ nano /etc/default/distcc
/etc/default/distcc
STARTDISTCC="true"
ALLOWEDNETS="192.168.100.0/24"
JOBS="8"
ZEROCONF="false"
```

En mi red, el rango de la red IP 192.168.100.0/24 es la dirección IP de mi red local. Debe modificarse para adaptarse a tu entorno de red. A continuación, reinicia el servicio distcc:

```
$ sudo /etc/init.d/distcc restart
```

Configuración del nodo de compilación de destino

Install distcc, distcc-pump and distccmon-gnome. distcc-pump is for running distcc pump mode. Distcc's pump mode accelerates remote compilation with distcc by also distributing preprocessing to the servers. distccmon-gnome is the distcc monitoring application:

```
$ sudo apt update
$ sudo apt install distcc distcc-pump distccmon-
gnome
```

A continuación, configura las direcciones IP de los nodos informáticos de compilación. En este caso, hay 8 nodos informáticos de compilación (ODROID-MC1). Es diferente para cada entorno de red. Escriba las direcciones IP del host de distcc de tus entornos.

```
$ nano ~/.distcc/hosts
~/.distcc/hosts
192.168.100.17,lzo,cpp
192.168.100.19,lzo,cpp
192.168.100.23,lzo,cpp
192.168.100.24,lzo,cpp
192.168.100.26,lzo,cpp
192.168.100.27,lzo,cpp
192.168.100.37,lzo,cpp
```

Los ejemplos agregan las siguientes opciones a la dirección: lzo: habilita la compresión LZO para este host TCP o SSH (esclavo).

cpp: habilita el modo distcc-pump para este host (esclavo). Ten en cuenta que el comando de compilación debe incluirse en el script pump para iniciar el servidor de inclusión.

Puedes encontrar una descripción del modo de bomba dentro del modo bomba de distcc: un nuevo diseño para la compilación distribuida de C/C ++. A continuación, ejecuta distcc-pump:

```
$ distcc-pump make -j64 CC=distcc
```

La Figura 8 muestra el estado de compilación de distcc en distccmon-gnome:

```
$ distccmon-gnome
```

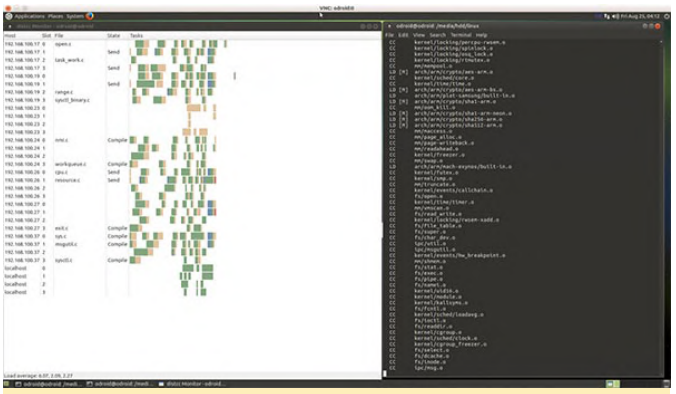


Figura 8: estado de compilación de distcc en distccmon-gnome

Hacer una prueba de medición de rendimiento

Medimos el rendimiento de compilación distribuido según el número de nodos informáticos de compilación. Este es el tiempo de compilación de las fuentes del kernel de Linux para ODROID-XU4. Tiene un mejor rendimiento que una compilación local cuando tiene más de 4 nodos informáticos de compilación, ya que la compilación distribuida tiene una sobrecarga adicional debido a la red. Los comandos de compilación utilizados para las pruebas son:

```
$ sudo apt update
$ sudo apt install git
$ git clone --depth 1
https://github.com/hardkernel/linux.git -b
odroidxu4-4.9.y
$ cd linux
$ make odroidxu4_defconfig
$ time make -j8 # Local build for baseline
$ make clean
$ time distcc-pump make -j64 CC=distcc #
Distributed build for testing
```

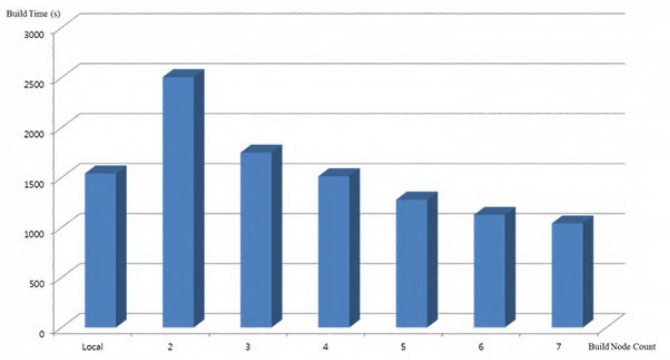


Figura 9 - Tiempo de compilación vs recuento de nodos

KVM Fun con virtualización en ODROID-H2 - Funciones Avanzadas

February 1, 2020 By Tobias Schaaf Desarrollo, ODROID-H2

En mi último artículo, demostré que es bastante fácil instalar y configurar KVM e incluí algunas herramientas para crear y controlar máquinas virtuales directamente en Linux. En esta ocasión, quisiera hablar de algunas características avanzadas que puedes usar con KVM de forma gratuita, que en otros hipervisores solo están disponibles en ediciones empresariales de pago, además de cómo controlar KVM en una instalación "básica" en lugar de un sistema gráfico. Este artículo incluye muchas descripciones técnicas, las cuales están marcadas como tales. Si solo quieres configurar las cosas para que funcionen, simplemente omite estas secciones.

Recapitulando

Recordemos lo que aprendimos la última vez sobre KVM, QEMU y libvirt. KVM: la técnica que utilizamos para virtualizar sistemas directamente en el kernel de Linux (similar a VMWare o Virtualbox). QEMU: se utiliza para emular ciertos tipos de hardware en combinación con la virtualización KVM, permite funciones avanzadas como instantáneas. libvirt: una API que utilizamos para controlar nuestras máquinas virtuales y lo que sucede a su alrededor, con virt-manager como interfaz gráfica.

Segundo Escenario

En este segundo escenario, quiero aprovechar al máximo algunas de las capacidades que te permiten usar KVM en un entorno de producción. Añadiremos un segundo ODROID-H2 a la configuración, experimentaremos con almacenamiento compartido en nuestras imágenes, así como mover un sistema en ejecución de un ODROID a otro sin interrupción. Quiero analizar virsh, el cliente de línea de comandos de libvirt que nos permite controlar máquinas virtuales desde la línea de comandos y nos permite realizar algunas configuraciones avanzadas. Y

controlar nuestras máquinas virtuales desde un PC remoto y no localmente en cada host. Al final, deberías saber bien cómo usar KVM incluso en un entorno de producción en tu lugar de trabajo.

Requisitos

- 2x ODROID-H2
- 1x PC/Servidor para almacenamiento compartido
- 1x PC para control remoto de nuestra configuración (yo usaré mi portátil)
- switches de red (con uno es suficiente, en un entorno de producción deberías contar con al menos dos)
- Conexión a Internet

En este segundo escenario, agregaremos un segundo ODROID-H2 a nuestra configuración para disponer de múltiples "nodos" en los que ejecutar máquinas virtuales. También uso un ODROID-N1 con un SSD conectado a modo de sistema de almacenamiento compartido que podemos usar para ejecutar nuestras máquinas virtuales. En lugar de trabajar directamente en los ODROID, me conectaré a ellos directamente en remoto desde mi portátil usando SSH. Para esto también instalé Debian Buster en el segundo ODROID-H2; esta vez la instalación será de servidor sin un escritorio X11 como MATE (que teníamos en el primer ODROID-H2), y sin el administrador de red o el virt-manager completo. Esto significa que la instalación es mucho más liviana, usa mucha menos RAM y cuenta con menos puntos críticos ante un posible ataque contra el sistema. Es muy similar a lo que se obtiene cuando se instala VMWare u otros hipervisores, que también simplemente instalan un Kernel y las aplicaciones básicas necesarias para ejecutar VM.

Instalación

Asumiré llegados a este punto que ya dispones de una instalación de servidor Debian (o Ubuntu) funcional para continuar con los siguientes pasos. Como he comentado, volví a usar Debian Buster, esta vez como servidor sin administrador de red o entorno de escritorio X11. Recomiendo encarecidamente utilizar el mismo sistema operativo en todos tus nodos y no mezclar Ubuntu y Debian u otros entornos, ya que la versión qemu utilizada probablemente será diferente y puede causar problemas en nuestro escenario. También ejecutaré todos los comandos como "root" en mi sistema, de modo que deberías saber cómo iniciar sesión como root o usar sudo para convertirse en root. Comencemos con la instalación de libvirt y las herramientas necesarias:

```
$ apt install libvirt-daemon-system
$ reboot
```

La instalación del libvirt-daemon-system es suficiente, ya que viene con todas las herramientas necesarias para ejecutar y controlar máquinas virtuales (como virsh, el cliente de línea de comandos).

Configuración de red avanzada

Con el fin de crear un entorno lo más parecido a "producción", analizaremos las configuraciones de red de nuestros ODROID. El ODROID-H2 tiene dos adaptadores de red (NIC) integrados. Sería un desperdicio no aprovechar esta situación. Por lo tanto, vamos a crear algo llamado "enlace", que es la combinación de los dos (o más, si fuera necesario) adaptadores de red en un enlace "virtual" (adaptador de red). Esto nos brinda diferentes escenarios de respaldo, en caso de que falle algo de la infraestructura bajo la cual se sustenta tu entorno de producción. También configuraremos algo llamado "puente" que actuará como un switch virtual para nuestras máquinas virtuales, lo que nos permitirá colocar las máquinas virtuales "lógicamente" en nuestra red, en lugar de utilizar NAT.

Descripción técnica

Si está familiarizado con, por ejemplo, VirtualBox o VMWare, deberías saber que las dos configuraciones de red más utilizadas son Bridge o NAT. NAT significa Traducción de direcciones de red, y es una técnica gracias a la cual la VM (o VMs) subsiste dentro de tu propia red privada, creada por el hipervisor (VirtualBox, VMWare, KVM, etc.), no está directamente conectada a tu red, pero usa la red del sistema host para comunicarse con el mundo exterior. Lo que significa que comparte la conexión a Internet de tu host, y normalmente tiene acceso a la misma red y sistemas a los que tiene acceso tu sistema host, pero

por otro lado no se puede acceder directamente desde otras máquinas, ya que la IP de la VM no está dentro de tu red. De hecho, cada vez que la VM accede a algo en la red, la red lo ve como tráfico entrante desde su sistema host, no desde la VM. Eso es lo que NAT hace por ti. Las redes puenteadas, actúan a modo de switch virtual, que está conectado a su red física. Esto significa que, cuando una VM solicita una IP, ya no le pregunta al host, sino a tu router de red. Obtendrá una IP y la configuración del mismo router del que tu sistema host obtiene su IP. Esto te permite controlar la IP que asigna tu router, y también que otras máquinas de la misma red puedan acceder a ésta.

Comencemos

En la mayoría de los casos dentro de un entorno de producción, desearás tener una red puenteada para tus máquinas virtuales que te permitirá que otros puedan acceder a los servicios que se ejecuten en tus máquinas virtuales. También crearemos un puente de red para KVM y así las nuestras máquinas virtuales podrán conectarse directamente a tu red doméstica o a la red de tu empresa, dependiendo de dónde quieras usar esta configuración. Para ello, necesitaremos instalar algunos paquetes adicionales:

```
# for creating bonds
$ apt install ifenslave
# for creating bridges
$ apt install bridge-utils
```

Necesitamos editar /etc/network/interfaces en ambos ODROID para configurar nuestra red avanzada. Ten en cuenta que la configuración de los adaptadores de red en /etc/network/interfaces deshabilitará el acceso del Administrador de red a estos dispositivos.

/etc/network/interfaces

```
# This file describes the network interfaces
available on your system
# and how to activate them. For more information,
see interfaces(5).
source /etc/network/interfaces.d/*
auto lo
iface lo inet loopback
auto bond0
iface bond0 inet manual
bond-slaves enp2s0 enp3s0
bond-primary enp2s0
bond_mode balance-alb
auto br0
iface br0 inet static
bridge_ports bond0
address 192.168.0.115
netmask 255.255.255.0
gateway 192.168.0.10
```

Descripción Técnica

El texto anterior es el contenido del archivo /etc/network/interfaces, así que vamos a explicar algunas de sus líneas: bond-slaves enp2s0 enp3s0 – Esta es una lista de NICs que se combina como un enlace. Los dos NIC en mis ODROID se encuentran bajo el nombre enp2s0 y enp3s0, algunos recordareis nombres como eth0 y eth1, enp2s0 es básicamente el nuevo eth0 y así sucesivamente.

ond_mode balanced-alb – Describe cómo el adaptador de red debería funcionar en conjunto. En: **https://www.kernel.org/doc/Documentation/networking/bonding.txt** encontrarás una descripción detallada de cómo funciona la estructura dentro del Kernel y cuáles son los diferentes modos

Un uso muy común es, por ejemplo, que el modo 1 (copia de seguridad activa) en el que un NIC siempre está ON y envía y reciba datos, pero si por alguna razón esto no es posible, cambiará al segundo NIC. Esta es una situación de espera activa para los adaptadores de red.

Hay un par de modos interesantes, por ejemplo, el modo 0 (balance-rr), el modo round-robin tiene la capacidad de acelerar bastante la comunicación. Cuando conecté mis dos ODROID H2 con el modo balance-rr y usé iperf3 para probar la velocidad de conexión, logré una velocidad de conexión de hasta 1.9 Gbit entre ambos ODROID. Lo que significa que realmente podía usar la velocidad completa de los dos adaptadores de red para que se comuncaran entre sí, pero también pude comprobar que la cantidad de errores durante el envío aumentó en consecuencia. Una máquina virtual que se ejecuta en el sistema con un adaptador de red en puente ya no podía comunicarse con el router dentro de mi red, pero si usamos la red NAT en la máquina virtual, no tendremos ese problema. De modo que, aunque

logremos aumentar considerablemente la velocidad entre los diferentes ODROID, el uso está limitado.

Otro modo interesante de conexión es el modo 4 (802.3ad), que es una técnica que ha cambiado su nombre por 802.1ax. De modo qué, si tu router admite el estándar 802.1ax, eso significa que es compatible con 802.3ad, lo cual es algo confuso. Es la supuesta agregación de enlaces, es posible utilizar ambos NIC para comunicarse y, por lo tanto, "técnicamente" duplicar tu ancho de banda. No implica tener una velocidad de conexión de 2 Gbit entre ODROID, pero si que significa que puede tener dos (o más, dependiendo del número de NIC) conexiones con 1 Gbit al mismo tiempo. Entonces, en lugar de una máquina que puede conectarse a un ODROID que tiene dos NIC con 1 Gbit, puede tener DOS dispositivos conectados al mismo tiempo, AMBOS con 1 Gbit. Sin embargo, requiere switches de red que admiten esta configuración y algunos preparativos.

bridge_ports bond0 – este es el adaptador de red para tu switch virtual (el adaptador puente). También podría ser directamente enp2s0 o enp3s0, pero usando un enlace, nos aseguramos de que, si un adaptador no tiene conexión, el otro adaptador pueda mantener nuestra red en funcionamiento.

Yo he utilizado una configuración de IP estática para mi puente, ya que siempre quiero tener la misma IP. Tambien podría haber optado por usar iface br0 inet dhcp para una configuración automática. El enlace está configurado en iface bond0 inet manual ya que no necesita una IP. El adaptador puente mantendrá la IP para el enlace, por así decirlo.

En el mejor de los casos, deberías conectar tus ODROID a dos switches de red diferentes que están conectados a tu router. De esta forma, si uno de los switches muere, los ODROID seguirían funcionando con el otro router. Razón principal para tener varios enlaces y modos como active-backup. También nos permite actualizar el firmware de los switches y reiniciar sin perder la conectividad de las máquinas virtuales que se estén ejecutando en tus hosts KVM

La configuración en su totalidad también funcionará con un switch. No necesita dos switch, ni siquiera necesitas conectar ambos adaptadores LAN del ODROID-H2.

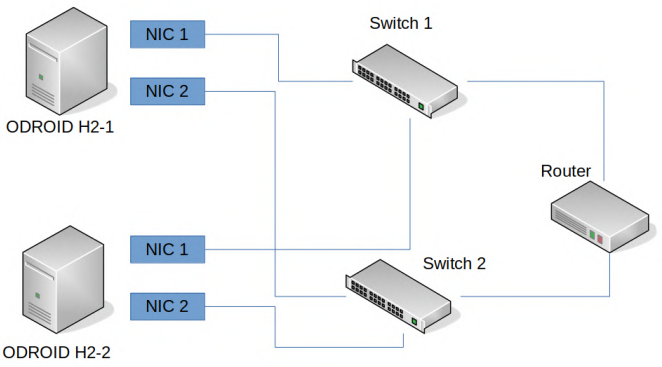


Figura 1 - Diseño de red para configuración de red redundante/ante errores

Acceso remoto

Como he dicho antes, ahora dispongo de más máquinas en la configuración y sería práctico conectar cada sistema a un televisor o monitor con su propio ratón y teclado para configurarlos. De modo que, quiero poder conectarme a cada dispositivo de forma remota desde mi portátil que ejecuta Ubuntu 18.04. Usar SSH me permite conectarme a mis ODROID de forma remota para configurarlos a través de la red. Creé una clave SSH en mi ordeandor portátil (si aún no lo ha hecho, puede usar el siguiente comando para generar una nueva clave ssh:

```
$ ssh-keygen
```

Esta clave se distribuye con el siguiente comando a los dos dispositivos ODROID-H2:

```
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@< server-ip >
```

Esto me permite iniciar sesión como root a través de SSH usando mi clave SSH en lugar de una contraseña. En general, es una forma más segura que usar una contraseña. De hecho, hice esto antes de empezar a configurar la red e instalar aplicaciones en los ODROID. Esto también me permite usar el administrador de máquina virtual de forma remota.

Una vez que hemos instalado los paquetes necesarios para KVM y libvirt y hemos configurado la red como he descrito anteriormente, podemos usar virt-manager para controlar nuestros hosts. Para esto, simplemente inicia virt-manager en el sistema que

desees usar para controlar tus ODROID (en mi caso es mi ordenador portátil). Haz clic en “File” en el menú y selecciona “add connection”. Marca la casilla que deseas conectar a un host remoto, el usuario debe ser root, luego simplemente introduce la IP de tu ODROID H2 en nuestra configuración. Repite el mismo proceso para el otro ODROID-H2. Con esto ya tenemos los ODROID en forma de lista dentro de nuestro administrador de máquina virtual.

Puedes marcar la casilla que debería conectarte automáticamente, o simplemente haz doble clic en los ODROID de la lista para conectarte a los mismos. Debería verse exactamente igual que cuando usaste virt-manager localmente en el primer escenario, solo que ahora tiene dos ODROID para trabajar y crear máquinas virtuales.

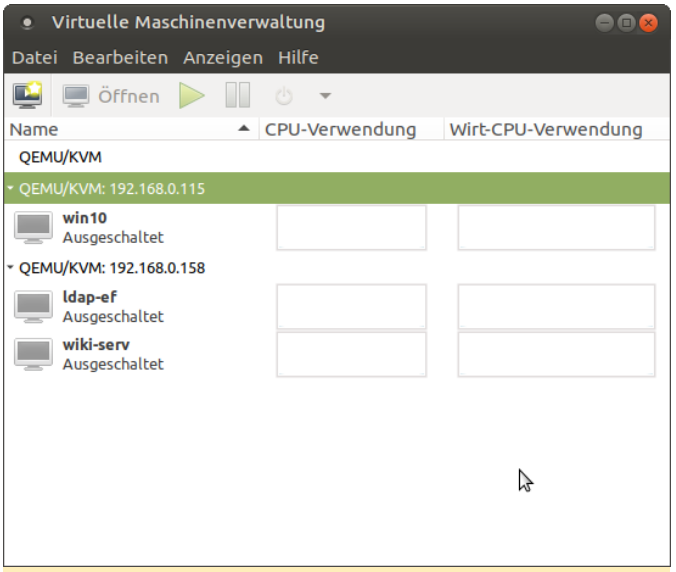


Figura 2: Ejecutando el administrador de máquina virtual para conectarse a hosts de virtualización remotos

Como puede ver, es bastante fácil conectarse a varios hosts, así como controlar y crear multitud de máquinas virtuales en tu red. Esto simplemente te permite ahorrar tiempo, dinero y recursos para usar máquinas virtuales en la red de tu casa o empresa. Teniendo en cuenta que puede ejecutar todo esto en una simple imagen de servidor, sin tener que instalar herramientas gráficas para configurar tus máquinas virtuales en el propio host, se acerca bastante a la forma en que se ejecutan VMware, Xen y otros hipervisores con soluciones "básicas".

Configurar un almacenamiento compartido

Aunque ya podemos realizar bastantes cosas en nuestro actual escenario, hay mucho más que podemos hacer usando KVM y libvirt. Para ello quisiera usar un almacenamiento compartido para los diferentes hosts de virtualización con los que estamos trabajando. Esto nos permitirá tener un almacenamiento centralizado para todos nuestros hosts sobre los que ejecutaremos nuestras máquinas virtuales, y nos brinda la posibilidad de ejecutar la misma máquina virtual en diferentes hosts, simplemente apuntando el host a la imagen de disco duro correcta para una determinada máquina virtual. También nos permite utilizar soluciones de red más rápidas y seguras, tales como SAN, NAS o clústeres de almacenamiento. Esto en sí mismo tiene muchos beneficios a la hora de ejecutar máquinas virtuales fuera del almacenamiento local directamente en los hosts. También reduce los costes generales, ya que no necesitas tener un almacenamiento para cada host, un sistema mínimo es más que suficiente. Puedes ejecutar todo el sistema desde un módulo eMMC de 8GB (incluso siendo más pequeño, también funcionaria) y dejar que las máquinas virtuales se ejecuten en un almacenamiento de red, con RAID, copias de seguridad, etc. Esto reduce el coste y el mantenimiento tanto del sistema de almacenamiento como de los hosts de virtualización (es decir, el ODROID). En mi configuración, utilizo un ODROID-N1 con un SSD conectado. Creé una partición en el SSD para este propósito y monté la partición en /srv/nfs. Como habrás adivinado por el nombre, he configurado un recurso compartido NFS para el almacenamiento compartido.

Instalación y configuración

La instalación es muy simple, así como la configuración. Como he dicho antes, en mi caso usare la partición montada en /srv/nfs, de modo que, si tiene una ubicación diferente, deberías modificar las rutas en consecuencia:

```
$ apt install nfs-kernel-server
$ mkdir /srv/nfs
$ chown nobody:nogroup /srv/nfs
$ chmod 777 /srv/nfs
$ echo "/srv/nfs
192.168.0.0/24(rw,sync,no_root_squash,no_subtree_c
```

```
heck)" >> /etc/exports
$ exportfs -a
$ systemctl restart nfs-kernel-server
```

Ten en cuenta que yo sólo permito conexiones de la subred 192.168.0.0/24. Si tu red es diferente, también necesita ajustar esto, o puedes reemplazar la subred por completo simplemente usando un asterisco (*) en lugar de /srv/nfs:

```
*(rw,sync,no_root_squash,no_subtree_check)
```

Ahora que el servidor está preparado y es totalmente funcional, necesitamos preparar los dos ODROID como cliente. Para esto, necesitamos instalar el paquete nfs-common en ambos clientes y configurar libvirt para montar el soporte NFS. Para esto, usaré virsh, la herramienta de línea de comandos de libvirt. Al igual que el administrador de máquinas virtuales, esta herramienta nos permite manipular todo lo que rodea a nuestras máquinas virtuales. De hecho, es mucho más potente que el administrador de máquinas virtuales en sí.

```
$ apt install nfs-common
$ mkdir -p /var/lib/libvirt/shared-pool
$ echo "< pool type='netfs'>
  < name>shared-pool</ name>
  < source>
      < host name='< server-ip >' />
      < dir path='/srv/nfs' />
     < format type='nfs' />
  < /source>
 < target>
       < path>/var/lib/libvirt/shared-pool</ path>
       < permissions>
              < mode>0755</ mode>
              < owner>-1</ owner>
             < group>-1</ group>
       </ permissions>
    </ target>
</ pool>" > shared-pool.xml
$ virsh pool-define shared-pool.xml
$ virsh pool-autostart shared-pool
```

Obviamente, debes reemplazarlo con la IP de tu servidor donde se está ejecutando el NFS, igual que /srv/nfs en caso de que tu ruta al recurso compartido NFS sea diferente. Lo que estamos haciendo exactamente es crear un archivo .xml llamado shared-pool.xml, que incluirá nuestros parámetros de conexión y la ruta donde queremos montarlo. Con virsh pool-define podemos decirle a libvirt que cree un nuevo grupo de almacenamiento.

Consejos

Se pueden hacer cosas similares con una VM. Puedes, por ejemplo, volcar una configuración xml de una VM e importarla a otra máquina:

```
# dump a VM configuration and redirect into a
file:

$ virsh dumpxml --domain win10 > win10.xml
# import a VM configuration as a new machine:
$ virsh define win10.xml
```

De esta forma, puedes hacer fácilmente copias de la configuración de una máquina sin tener que hacer la misma configuración a través del administrador de máquina virtual una y otra vez. Simplemente descárgate una configuración de VM, edita el archivo con tu editor de texto favorito. Por ejemplo, adapta la dirección MAC de la NIC y el archivo de imagen del disco duro, y tendrás una nueva VM basada en la configuración de una VM ya existente. Naturalmente, también existe una opción para clonar una VM, que también te permitiría iniciar la MISMA VM en un host diferente, siempre que la configuración exista en ambos sistemas y tengan acceso al archivo de disco duro (de ahí el uso compartido del grupo de almacenamiento en el que estamos trabajando).

Administrador de máquina virtual

También puedes hacer lo mismo a través del administrador de máquina virtual. Como puede ver en la Figura 3, existe multitud de formatos de almacenamiento y soluciones compatibles para libvirt que te permiten configurar la solución de almacenamiento que prefieras para tu configuración. NFS es solo una de las muchas opciones que tienes. A modo de comentario, nosotros en el trabajo utilizamos un Ceph Cluster como grupo de almacenamiento para nuestros servidores KVM.

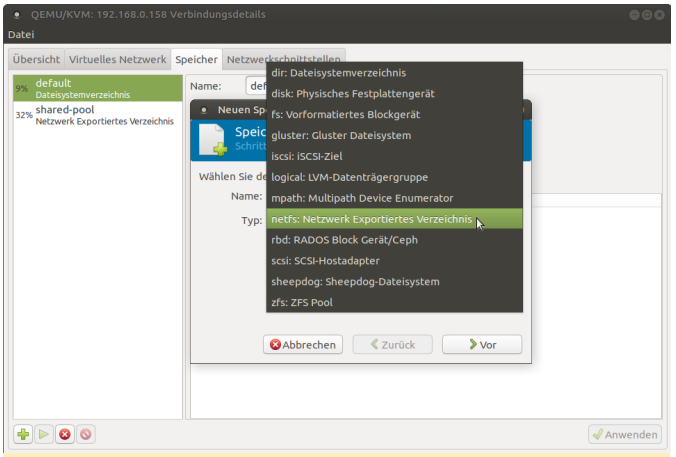


Figura 3: Agrega un grupo de almacenamiento a tu host KVM usando el administrador de máquina virtual

La velocidad de tus máquinas virtuales es tan buena como tu solución de almacenamiento, y si necesita la mayor velocidad posible, entonces, por supuesto, ejecutar un almacenamiento local desde un NVMe podría ser mucho mejor que ejecutar tus máquinas virtuales desde un almacenamiento compartido en red, pero también limita tus opciones a la hora de trabajar con él, y dependiendo de tu solución de almacenamiento, la velocidad puede llegar a no ser tan mala.

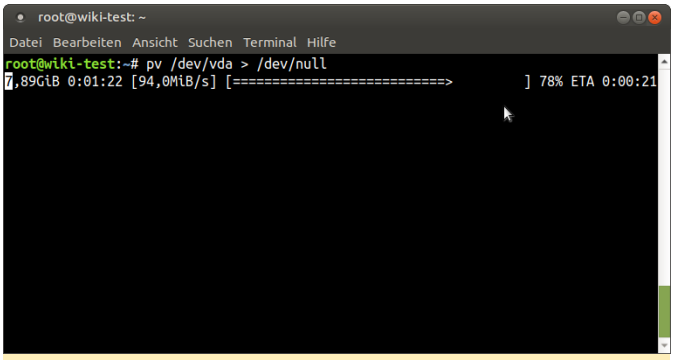


Figura 4: Volcar la imagen de HDD virtual de una máquina virtual en ejecución en /dev/null para verificar el rendimiento de lectura del grupo de almacenamiento NFS

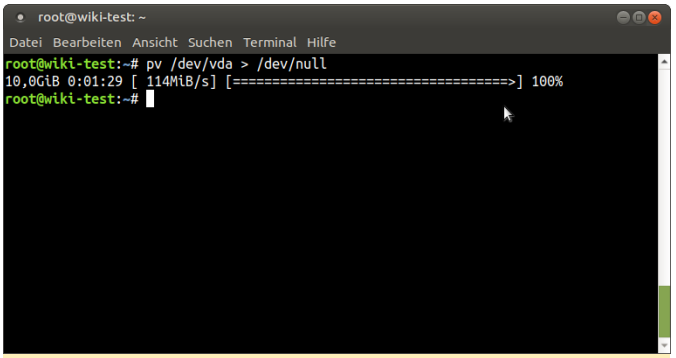


Figura 5: volcar la imagen de HDD virtual de una máquina virtual en ejecución en /dev/null para verificar el rendimiento de lectura del grupo de almacenamiento NFS

Usar el sistema de almacenamiento compartido debería ser lo más obvio. En lugar de crear nuevas imágenes en la ubicación por defecto, las creas en el almacenamiento compartido y estarán disponible para todos los ODROID que acceden al almacenamiento.

Migración en vivo

Para aquellos que se estén preguntando qué es la migración en tiempo real, es una técnica que permite mover una VM de un host a otro mientras el host aún se está ejecutando. Esto permite actualizar o incluso reiniciar el host sin interrumpir los servicios (VM) que estás ejecutando. Por ejemplo, si ejecuta una máquina virtual que aloja WordPress, pero necesita actualizar el host en el que se ejecutan las VM, o si desea distribuir la carga en otro ODROID, puede mover la VM (mientras se ejecuta) a otro host, y durante todo el tiempo el sistema aún está accesible. Las personas que se conecten al servidor de WordPress ni siquiera notarán que la máquina se está moviendo.

Technical description

Este escenario requiere un grupo de almacenamiento compartido, lo cual significa que todos los hosts (ODROID) tienen acceso a la imagen del disco duro. Pero los datos que están en la RAM y procesados por la CPU son únicos y esa es la parte con la que trabaja libvirt. Se copiará la configuración de la máquina (el archivo xml) de una máquina a otra y empezará un proceso de copia del contenido de la RAM de la VM que se está ejecutando actualmente de un host al otro host (un ODROID al otro). Como no necesitas

copiar el sistema operativo en sí (se comparte la imagen del disco duro), incluso se podría mover una máquina virtual que tenga cientos de GB o incluso TB de datos de un host a otro, ya que únicamente debe ser copiado el contenido de la RAM. Libvirt creará una copia idéntica de la configuración de la VM de un host a otro. Esto significa que también creará el mismo hardware adicional, como por ejemplo la tarjeta de red, la tarjeta de sonido, la tarjeta gráfica, etc., para esto, los otros hosts deben tener la capacidad de ejecutar el mismo hardware virtual. Como la máquina en sí misma se está ejecutando y está realizando tareas, el contenido de la RAM puede cambiar durante el proceso de sincronización, verás como la barra de proceso cerca del final "salta hacia atrás" y continuará haciéndolo hasta que complete su tarea por completo. Esto depende de lo rápida que sea tu red entre los hosts y con qué frecuencia cambia la RAM y cuánta RAM tiene una VM. Una máquina virtual que está inactiva la mayor parte del tiempo y se ejecuta en 500MB de RAM será muy rápida de sincronizar y solo tomará unos segundos. Si ejecutamos una base de datos, un servidor de archivos o una máquina con un compilador en ejecución que carga constantemente nuevos datos en la RAM y tienes 64 GB de RAM, este proceso llevará mucho más tiempo y puede tardar varios minutos en completarse. Al final, ambas máquinas se pondrán en "pausa" durante una fracción de segundo para hacer el cambio de una máquina a otra y luego se reactivarán, como he comentato, nadie notará nada en absoluto. Esta característica también está disponible en el hipervisor VMWare, pero no sin invertir una gran cantidad de dinero; mientras que en KVM es una característica gratuita.

Cómo usar la migración en tiempo real

Para usar la migración en tiempo real, no necesites hacer mucho. Es mejor tener el modo de caché de disco para máquinas virtuales configurado en "ninguno" para la migración, ya que los métodos de almacenamiento en caché pueden causar problemas en caso de fallos. Por ejemplo, cuando migramos una máquina de un host a otro y el host falle, podría ser que los datos que aún estaban en caché se pierden y no se escriben en el disco. Esto generalmente es un problema de los métodos de almacenamiento en caché, no obstante, esta advertencia se puede ignora y forzar la migración entre máquinas. Puedes migrar máquinas a través de la interfaz gráfica desde el administrador de máquinas virtuales. Para esto, debe estar conectado a ambos hosts, hacer clic con el botón derecho en la máquina virtual que deseas migrar de un host a otro y selecciona “migrate”. Luego selecciona el host al que deseas migrar de la lista desplegable y haga clic en “start”. En las opciones avanzadas, puede activar que quieres migrar incluso si se está utilizando un algoritmo de almacenamiento en caché de disco "inseguro". Otra posibilidad es usar nuestra herramienta de línea de comandos virsh para migrar una máquina. Para esto, inicia sesión a través de ssh en el host donde se esté ejecutando la VM y usa el siguiente comando:

```
# Syntax
$ virsh migrate --verbose < VM> qemu+ssh://
/system
# example:
$ virsh migrate --verbose win10
qemu+ssh://192.168.0.115/system
```

Puedes añadir el parámetro --unsafe para permitir la migración con métodos de almacenamiento en caché de disco inseguros.

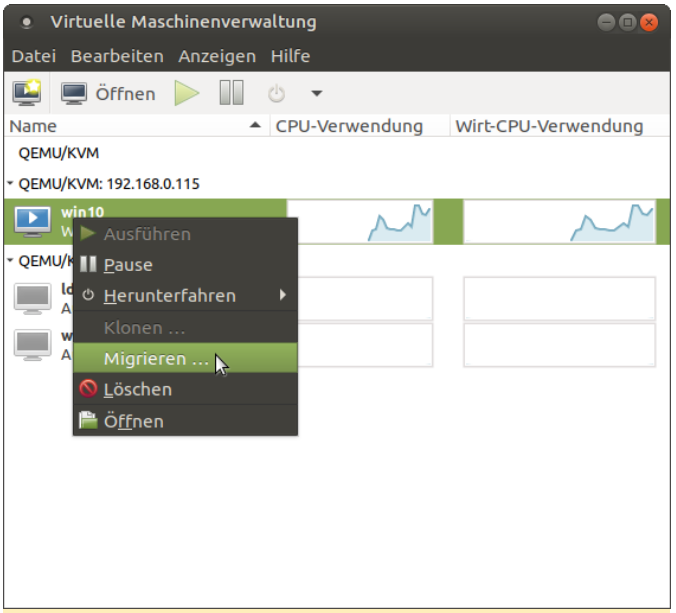


Figura 6: Migración de una VM a través del administrador de máquina virtual

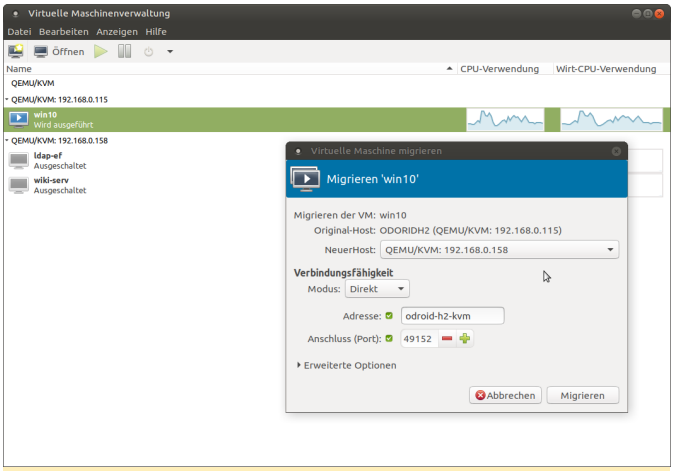


Figura 7: Selecciona el host de destino y haga clic en migrate

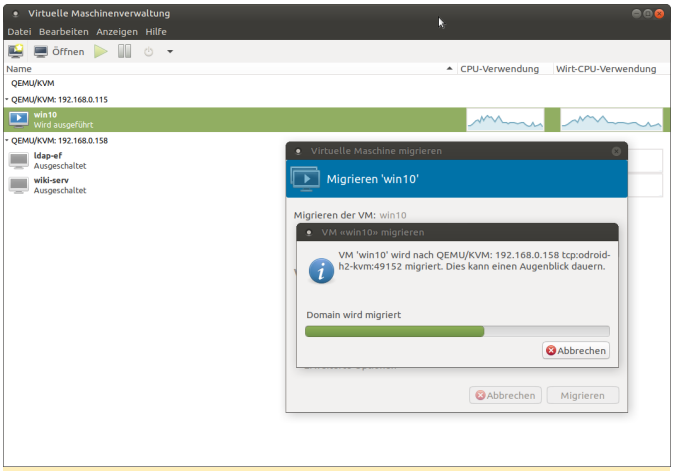


Figura 8: Observa como el proceso de migración hace magia

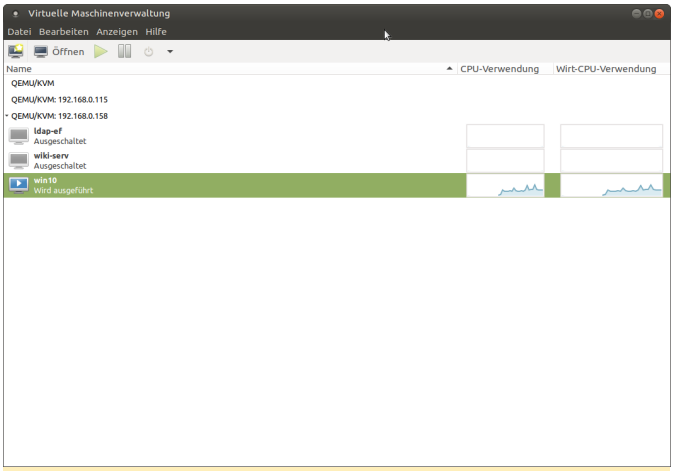


Figura 9: La VM se movió a otro host mientras seguia estaba en funcionamiento

Como puede ver, el proceso es bastante simple y directo. No hay mucho en lo que prestar atención. Una vez que configures el grupo compartido y los dos hosts KVM están en la misma red y pueden mover datos entre sí, no hay nada que impida mover una máquina de un host a otro mientras aún esté en funcionamiento.

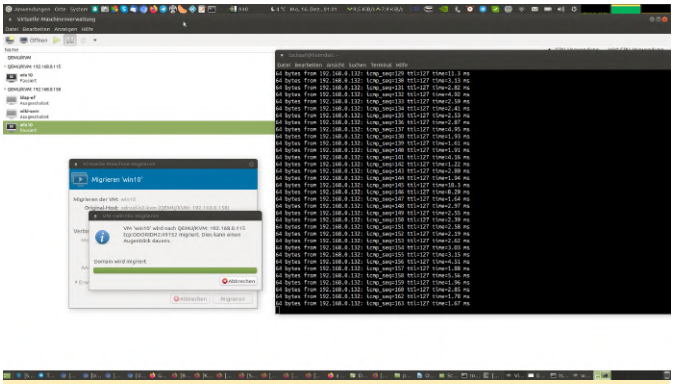


Figura 10: Si observas el ping continuo a una VM en ejecución, notarás que falta el sec. 161, lo que significa que se perdió un solo ping al migrar la VM

La captura de pantalla de la Figura 10 se tomó en el momento en que finalizó el proceso de migración y se cambió la VM. Un único ping no llego a completarse. Imagina que alguien acceda a un servidor de WordPress que se está migrando. Este retraso de un ping ni siquiera se notaría en una carga de trabajo normal y el usuario nunca sabría que se acaba de mover todo el sistema de un host a otro. De hecho, utilizando una conexión de fibra, incluso se podría migrar máquinas virtuales fácilmente entre diferentes ubicaciones o centros de datos.

Conclusión

Con esto, hemos aprendido a configurar un entorno de producción para nuestras máquinas virtuales. Ahora deberías poder compartir máquinas virtuales en tu red para proporcionar servicios a todos tus clientes, o usar esto para alojar un amplio espectro de servicios on line utilizando redes puenteadas. También debe saber cómo configurar grupos de almacenamiento compartido para aprovechar el almacenamiento de red, como un SAN o NAS o un simple recurso compartido NFS. Además, deberías haber aprendido cómo migrar máquinas virtuales entre diferentes hosts, lo cual te permite realizar tareas de mantenimiento o distribuir la carga de máquinas virtuales entre diferentes hosts. Todo esto se puede lograr en tu ODROID-H2 u otros PC/Servidores con la ayuda de KVM como motor de virtualización (hipervisor). Hay un montón de documentación sobre este tema y te recomiendo que leas más sobre ello si esta guía te ha despertado la curiosidad.

Optimización

Hay muchas cosas que puede cambiar en libvirt para aumentar el rendimiento en diferentes escenarios. El algoritmo de almacenamiento en caché para el disco virtual, por ejemplo. Quisiera hablar un poco sobre la optimización de la CPU y sobre la "emulación" de la CPU en KVM o, mejor aún, QEMU. He comentado con anterioridad que QEMU se usa para "emular" el hardware, y KVM se usa para la parte de virtualización. Aquí nos encontramos con algo único de solución KVM. En lugar de pasar por la CPU EXACTA que tu host ofrece, como a menudo se hace con VirtualBox y otra solución de hipervisor, QEMU "emula" una CPU pero pasa a través de las funciones de la CPU que soporta. ¿Por qué emular y no pasar? La respuesta es simple. ¡Portabilidad! Imagine el siguiente escenario: Tienes un servidor de presupuesto pequeño muy reciente para fines de prueba, digamos, por ejemplo, con un procesador Intel® Xeon® E-2226G de 2019 con 64 GB de RAM que utilizas para probar la configuración de un nuevo sistema, por ejemplo, un nuevo servidor de base de datos MSSQL de Windows, y después de las pruebas, desea mover la VM a un servidor más antiguo pero más potente que ejecute un sistema de procesador dual alimentado por dos procesadores Intel® Xeon® Processor E5-2697A v4 de 2016 con 1TB de RAM como entorno de producción. En muchos escenarios, tendría que configurar la VM nuevamente en el Servidor más antiguo, ya que el E-2226G basado en Coffee Lake más nuevo tiene algunas características de CPU que no están disponibles en el E5-2697A basado en Broadwell antiguo. Windows no solo requeriría una reactivación debido al cambio de CPU, en el peor de los casos, el sistema se comporta de manera muy diferente al escenario probado, sino que probablemente no podrá mover la VM mientras aún se ejecute de un servidor a otro. otro, ya que los sistemas son tan diferentes que la transferencia fallará. Aquí es donde QEMU y KVM funcionan de manera diferente. Si revisas la lista de CPU en las opciones del administrador de la máquina virtual, verás que hay una lista bastante extensa de CPU, tal y como se muestra en la Figura 11.

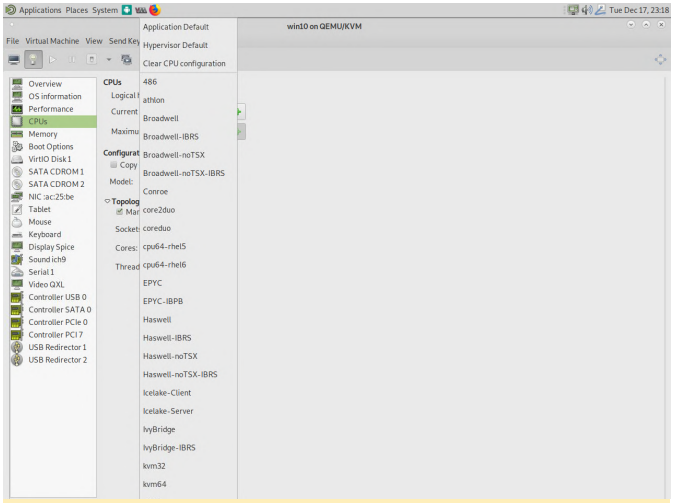


Figura 11 - Lista de CPU emuladas por QEMU

Lo que esto significa es que puedes crear una VM con una CPU emulada que sea compatible con ambos servidores (por ejemplo, Broadwell-IBRS). Este doble soporte permite configuraciones de prueba que son iguales en ambos sistemas y permite la migración en tiempo real de la VM de un host a otro sin interrumpir el servicio (como se ha explicado anteriormente) incluso si los servidores se ejecutan en arquitecturas de CPU completamente diferentes. Siempre que encuentre un denominador común más pequeño en la arquitectura de la CPU, puede configurar una VM con esto y migrar máquinas entre estos servidores de todos modos. ¿Como funciona? Si analizas el resultado del siguiente comando, probablemente habrás notado que esta línea "flags" aparece para cada CPU:

```
$ cat /proc/cpuinfo
```

Estas son las características que admite tu CPU. Incluso las placas ARM tienen esta línea, que muestra qué técnicas entiende y admite la CPU.

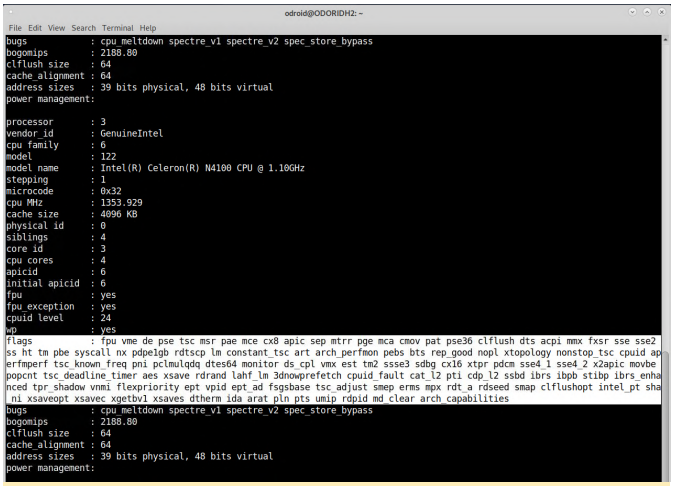


Figura 12 - Características de la CPU ODROID-H2

Ciertas arquitecturas de CPU presentan diferentes tipos de indicadores de CPU. Broadwell tiene menos funciones como, por ejemplo, Skylake, aunque más funciones que una CPU SandyBridge, y éstas dependen QEMU y KVM. Dependiendo de la arquitectura de CPU que selecciones, los indicadores "bien conocidos" se enviarán como características de CPU a la VM. Por lo tanto, reenviar las capacidades de rdseed de tu CPU a la VM puede aumentar el rendimiento de cifrado a medida que añades un generador adicional de números aleatorios a tu sistema para obtener más entropía. Una característica que no estaba presente en una CPU SandyBridge antigua, por ejemplo. ¿Cómo usarlo en el ODROID H2? El problema con el ODROID H2 es que no es una placa de servidor real, en algunos casos ni siquiera es un procesador de escritorio real, por lo que pierdes algunas características de CPU que esperarían presentes en este tipo de dispositivos. La virtualización está optimizada para el entorno del servidor y las placas como ODROID H2 no coinciden con esta descripción, por eso tenemos que solucionar algunas cosas para que funcione correctamente. Por defecto, la CPU debe mostrarse como Westmere o IvyBridge-IBRS, ya que estas son las CPUs que KVM encontrará como compatibles de forma predeterminada, pero eso es solo porque es donde encuentra TODAS las características de la CPU que espera. Inclujso si no encuentra algunas características, no significa que no sea positivo utilizar una arquitectura de CPU más alta

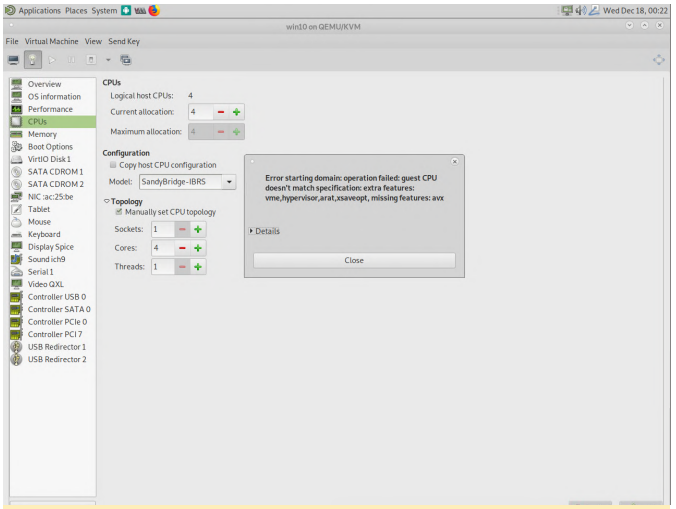


Figura 13: Al intentar iniciar una VM con la configuración SandyBridge-IBRS falla cuando faltan las características de la CPU

Aquí podemos usar virsh nuevamente para configurar opciones específicas como una solución alternativa. Como expliqué antes, los diferentes tipos de CPU son una combinación de diferentes indicadores de CPU. Esto nos dice que nos falta la función avx que, si la comparamos con los indicadores del ODROID-H2, es cierto que falta esta función. Podemos agregar ésta a la configuración de la máquina. También existe una lista de características adicionales que utilicé, pero que no forman parte de la configuración de SandyBridge y, por lo tanto, deben añadirse manualmente. Con el siguiente comando, podemos editar la configuración de la VM directamente:

```
$ virsh edit --domain win10
```

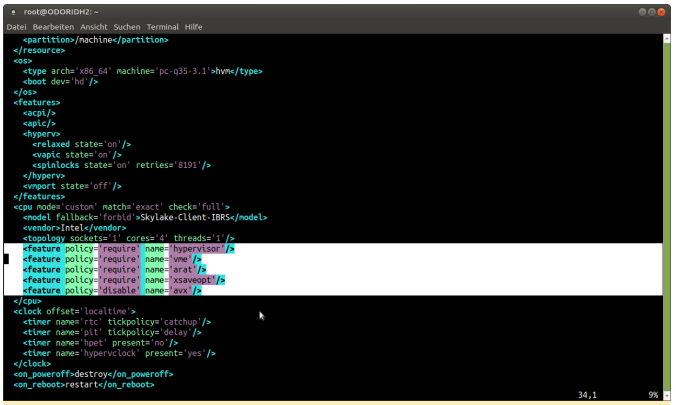


Figura 14: Añadir y deshabilitar funciones de CPU directamente en la configuración de VM con virsh edit

Entre el indicador y , podemos agregar definiciones con respecto a nuestra CPU. Como puede ver en la Figura 14, tenemos dos opciones: requerir o deshabilitar. Requerir reenviará una función de la CPU de la CPU del host a la VM, y Deshabilitar

eliminará una función de la CPU de la lista de funciones reenviadas a una VM. Como SandyBridge, por ejemplo, falla debido a la ausencia de la función avx de la CPU, podemos usar la siguiente opción:

```
< feature policy='disable' name='avx' />
```

Con este ajuste, podemos usar una CPU SandyBridge en nuestra VM incluso si no todas las funciones son compatibles. De hecho, pude ejecutar Skylake-Client-IBRS deshabilitando las siguientes funciones:

```
< feature policy='disable' name='avx' />
< feature policy='disable' name='avx2' />
< feature policy='disable' name='fma' />
< feature policy='disable' name='pcid' />
< feature policy='disable' name='bmi1' />
< feature policy='disable' name='bmi2' />
< feature policy='disable' name='invpcid' />
< feature policy='disable' name='f16c' />
< feature policy='disable' name='hle' />
< feature policy='disable' name='rtm' />
< feature policy='disable' name='adx' />
< feature policy='disable' name='abm' />
```

Sugiero agregar la función "hipervisor", ya que generalmente muestre que la máquina es una VM.

Información general Puedes hacerte una idea de lo que es posible y cuáles son las CPU compatibles más recientes al consultar este enlace: **https://www.berrange.com/posts/2018/06/29/cpu-model-configuration-for-qemu-kvm-on-x86-hosts/**. Es importante habilitar algunas funciones que pueden decirle a un Invitado que no necesita corregir algunos de los errores recientes de Intel, como Spectre. Existe un microcódigo que funciona entorno a este error disponible en el SO. Este microcódigo normalmente reduce el rendimiento de la CPU hasta cierto punto. Si ejecuta una VM, la VM no es consciente del hecho de que este "error" no está presente en la CPU (arreglado por el microcódigo en el host) e intentaría solucionarlo también con su propio microcódigo. Para esto, encontrarás algunos indicadores de CPU en el sitio anterior que le indican a la máquina virtual que este error ya está solucionado, que puedes añadir con el indicador "require". También encontrarás opciones para procesadores AMD en el sitio anterior. Una vez más, esta es una característica en la que debes profundizar si deseas obtener más información al respecto.

Conclusión

Sumergirme en KVM y libvirt ha sido una experiencia muy interesante. La capacidad de migrar máquinas virtuales sobre la marcha sin interrumpir el servicio es una característica muy interesante y muy útil en un entorno de producción. En el trabajo, lo hemos estado usando durante años. En combinación con Ceph u otras soluciones de almacenamiento de bajo coste, es una alternativa buena y barata a VMware, por ejemplo, con muchas características a nivel empresarial. Puesto que se ejecuta directamente en Linux, un servidor Debian o Ubuntu estándar con casi nada instalado ya es suficiente para ejecutar y proporcionar una buena base para administrar tus hosts y máquinas virtuales. Proporciona controladores y kernels actuales con parches de seguridad, no teniendo que pasar por el bloqueado de soluciones como VMWare, donde debes esperar a que proporcionen soporte de hardware y parches de kernel para tu software.

Ejecutando GNOME Desktop en el ODROID-N2

February 1, 2020 By Dongjin Kim Linux, ODROID-N2

Este artículo versa sobre cómo ejecutar el Escritorio GNOME en un ODROID-N2 con un kernel Linux v5.4 ascendente. Afortunadamente, el kernel ascendente tiene muchos parches que hacen que el ODROID-N2 funcione sin problemas y agradezco a Neil Armstrong, ya que contribuye con muchos parches para Amlogic ARM SoC (sin mencionar esa cosa que camina sobre la Luna. Ed.) Y también al usuario de ODROID@Memeka, que realmente logró que GNOME funcionase en ODROID-XU4 y ODROID-N2, con anterioridad. Los cambios en el código central han sido realizados por ellos, yo simplemente he reunido todas las piezas dispersas en mi repositorio de paquetes personal.

He conseguido personalizar el instalador Netboot de Debian/Ubuntu Netboot con mi repositorio de paquetes, http://ppa.linuxfactory.or.kr Recientemente, he pasado mucho tiempo instalando GNOME Desktop en mi ODROID-N2 usando el instalador Netboot de Ubuntu 19.04 que puedes descargar desde http://bit.ly/2NjQSG3. Visita mi otro artículo sobre mi instalador personalizado de Netboot y cómo puede usarlo.

Instalar Ubuntu 19.04 en ODROID-N2

El instalador de Ubuntu 19.04 Netboot se puede descargar desde http://bit.ly/2NjQSG3 y la imagen se puede garbar en la tarjeta MicroSD usando Etcher o la herramienta de línea de comandos de Linux "dd". Si usas Petitboot, la imagen también se puede mostrar en una memoria USB. La ventaja de instalar con Netboot Installer es que el sistema operativo se puede instalar directamente en el almacenamiento USB y puede personalizar la tabla de particiones durante la instalación, si sabes cómo administrar particiones.

Después de grabar la imagen del instalador de Netboot, el contenido se verá así. No es necesario que

toque ninguno, excepto "preseed.ini", si quieres instalar una configuración predefinida.

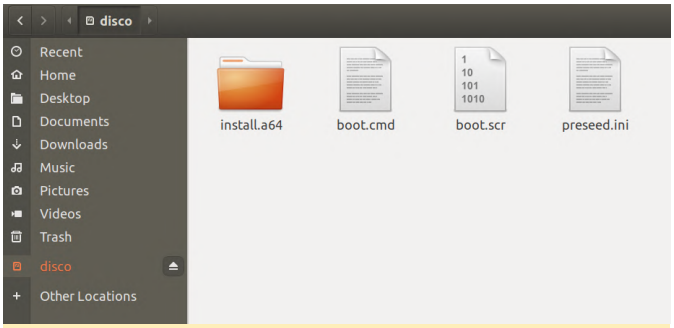


Figura 1 - Contenido del instalador

Por defecto, el instalador de Netboot está configurado para instalar Ubuntu GNOME Desktop. Si está familiarizado con los pasos de instalación de Ubuntu, puede instalar manualmente el sistema operativo cambiando la clave "di_auto" a "false".

preseed.ini

```
;
; Ubuntu 19.04 (Disco) Installer
;

; Set as 'false' if you like to install manually from scratch.
di_auto=true

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; NOTE: the value below only affects when 'di_auto=true'.           ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; Uncomment one to use a installation profile
;
;                'minimal' -> Bare minimal Ubuntu server
;    'ubuntu-gnome-desktop' -> GNOME Desktop
;
;di_profile=minimal
di_profile=ubuntu-gnome-desktop

; The hostname of your computer printed with the command 'hostname'
;
di_hostname=disco

; The domain name of your computer in the network
;
di_domain=disco.your.domain

; Your full name
;
; PLEASE DO NOT USE WHITE SPACE IN THE NAME, YOU CAN CHANGER IT AFTER INSTALLATION
;
di_user_fullname=YourFullName

; The default loggin account name and password
;
di_username=odroid
di_user_password=odroid

; Target disk device to install "Ubuntu 19.04 (Disco)"
;
; The OS will be installed to the disk device specified with Linux device name.
;
;          eMMC -> /dev/mmcblk0
;           uSD -> /dev/mmcblk1
;    USB storage -> /dev/sda, /dev/sdb, ...
;
disk=/dev/mmcblk0
;disk=/dev/mmcblk1
;disk=/dev/sda
;disk=/dev/sdb
;disk=/dev/sdc
;disk=/dev/sdd
```

Figura 2 - preseed.ini

Puedes cambiar los valores predefinidos como mejor te parezca, por ejemplo, la cuenta o la contraseña por defecto. Los valores de configuración se pueden cambiar más tarde tras la instalación; a excepción del dispositivo de disco de destino, donde instalas Ubuntu, ya que el instalador formateará y sobrescribirá el dispositivo de disco con el nuevo sistema operativo.

La instalación lleva tiempo

Ahora estás listo para iniciar el instalador de Netboot. Puedes arrancar desde la ranura de la tarjeta MicroSD, eMMC o, incluso, por Petitboot. El instalador de Ubuntu 19.04 Netboot instala los paquetes a través de la red, por lo tanto, el tiempo de instalación puede variar dependiendo del ancho de banda de su red o del servidor del repositorio de Ubuntu, pero al final, se instalará.

Una vez que todo esté instalado sin fallos, ODROID-N2 se reiniciará y se iniciará GNOME Desktop. Pero observarás que funciona un poco más lento de lo esperado, ya que el instalador actual no puede usar el controlador Mali Bifrost durante la instalación, que he intentado solucionar antes de publicar la imagen del instalador.

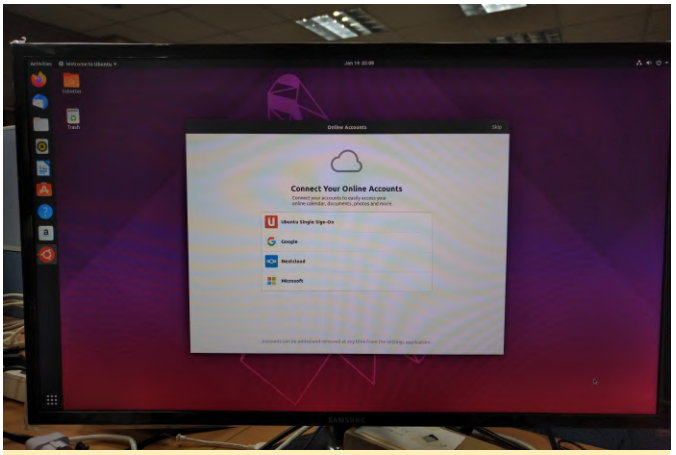


Figura 3 - Gnome Desktop en funcionamiento

Esto se puede resolver fácilmente con una instrucción para instalar el driver tú mismo y hacer que surta efecto después de reiniciar:

```
$ sudo apt install mali-bifrost-wayland-driver
$ sudo reboot
```

En el próximo arranque, debes comprobar si "Ubuntu on Wayland" está seleccionado en la pantalla de inicio de sesión para asegurarte de que el controlador Mali Bifrost Wayland se esté ejecutando.

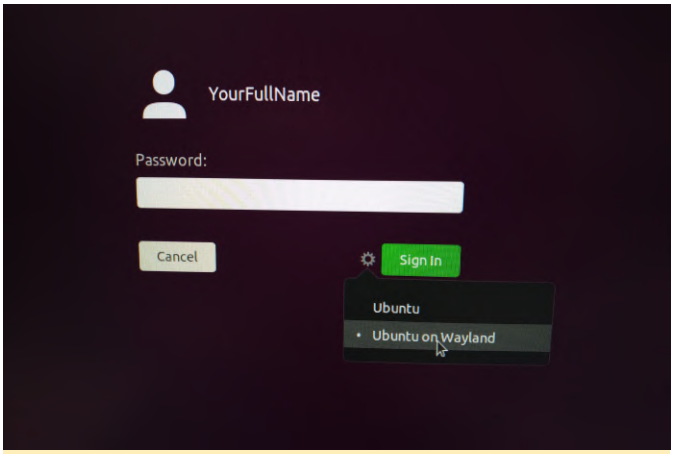


Figura 4 - Opción de selección para "Ubuntu en Wayland"

Probando el driver de Mali Bifrost Wayland

El ejemplo más simple a realizar es con glmark-es2 –wayland, que se puede hacer con el siguiente comando:

```
$ sudo apt install glmark2-es2-wayland
```

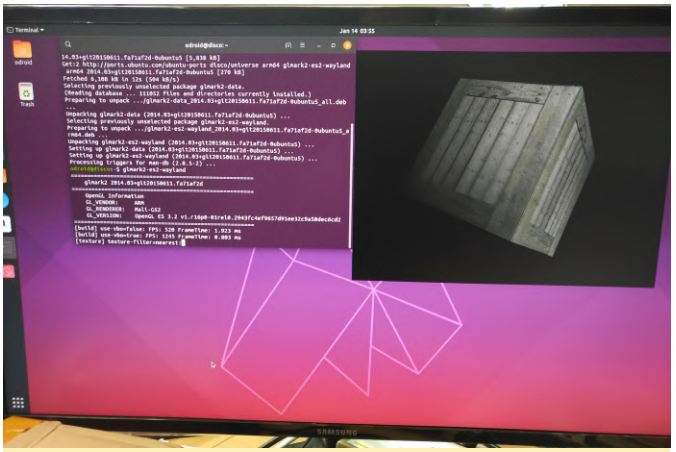


Figura 5 - Demo GLMARK2

También he compilado Qt5 (5.12.2 + dfsg-4ubuntu1.1), para ejecutarlo en GNOME Desktop con Wayland. Por lo tanto, QtWayland5 tiene que estar instalado. He observado que muchos ejemplos de Qt5 no funcionan correctamente con la compilación, pero aún así no está de más probarlo.

```
$ sudo apt install qt5-default qtwayland5
```

Figura 6 - Prueba con QT5 Wayland

Problemas conocidos

Gnome-terminal no se puede iniciar si se instala con un perfil predefinido. Este es un problema que no puedes resolver y solo ocurre si instalas el sistema operativo con el perfil predefinido "ubuntu-gnome-desktop". La solución para este problema es ejecutar las dos instrucciones en la línea de comandos del shell después de conectarse a tu ODROID-N2 o abrir una pantalla de consola y esto surte efecto después de reiniciar.

```
$ sudo locale-gen – purge en_US.UTF-8
$ echo -e ‘LANG=”en_US.UTF-8"
LANGUAGE=”en_US:en”
’ | sudo tee /etc/default/locale
$ sudo reboot
```

Funciones que faltan en comparación con el kernel estándar v4.9 para el ODROID-N2:

Muchos desarrolladores están actualizando el kernel ascendente, especialmente gracias a Neil Armstrong, que contribuye con muchos parches al kernel ascendente para Amlogic SoC y @memeka, que realmente hicieron que GNOME funcione con los blobs de Mali Bifrost. El kernel seguirá actualizándose a menudo y se cargará sin previo aviso, aunque Ubuntu te informará cada vez que aparezca una actualización.

Para obtener más información, consulta la publicación del artículo original en **https://medium.com/@tobetter/running-gnome-desktop-on-odroid-n2-98a187dff055.**

Instalación del Sistema Operativo Utilizando Petitboot y USB OTG

February 1, 2020 By Justin Lee, CEO of Hardkernel ODROID-N2, Mecaniqueo

Petitboot

Petitboot es un gestor de arranque independiente de plataforma basado en el mecanismo de reinicio en caliente kexec de Linux. Petitboot admite la carga de archivos de kernel, initrd y ficheros de árbol de dispositivos desde cualquier sistema de archivos que se monte en Linux, además puede cargar archivos desde la red utilizando los protocolos FTP, SFTP, TFTP, NFS, HTTP y HTTPS.

Verificación de versión

Para verificar tu versión de Petitboot, mueve el interruptor de arranque al modo de arranque SPI y enciende el ODROID-N2.

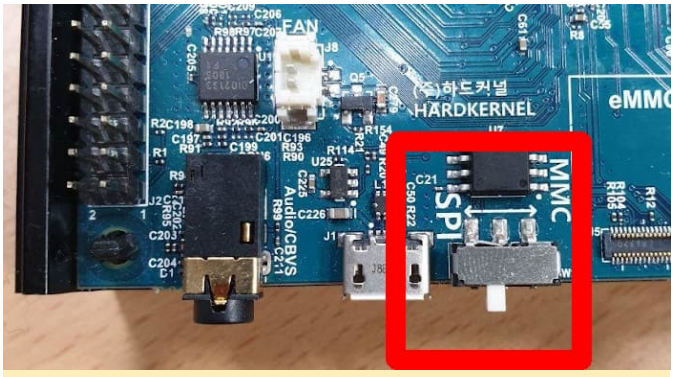


Figura 1 – Cambiando el arranque al modo SPI

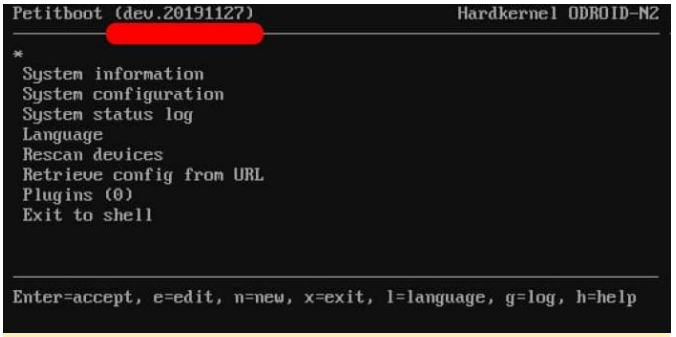


Figura 2 - Información de la versión localizada en la esquina superior derecha.

Si tu versión de Petitboot es inferior a dev.20191127, consulta la página "How to Recover or Upgrade" disponible en https://wiki.odroid.com/getting_started/petitboot/recover_or_upgrade.

Instalación del sistema operativo a través del PC usando OTG del ODROID-N2

Esta configuración permitirá instalar el sistema operativo directamente en la memoria (eMMC o uSD) en el ODROID-N2 desde un PC a través de USB al puerto OTG del N2.

Figure 3 - Block Diagram of the connection

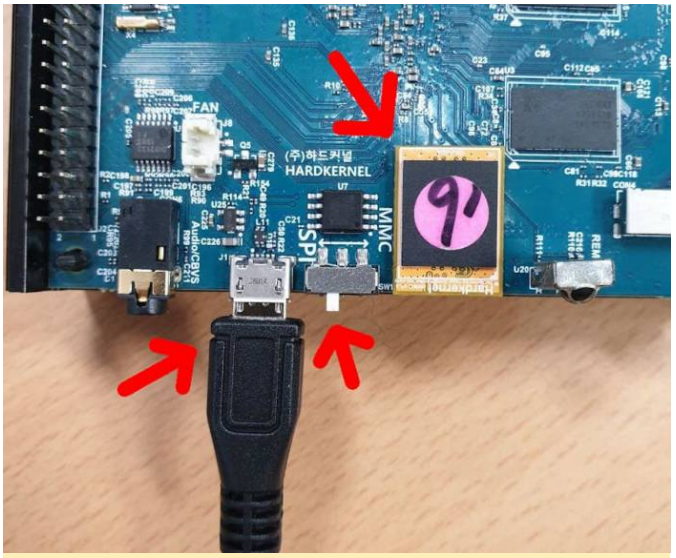


Figura 3 - Diagrama de bloques de la conexión

Configuración de Petitboot

Selecciona "Exit to shell", tal y como se muestra en la Figura 5.

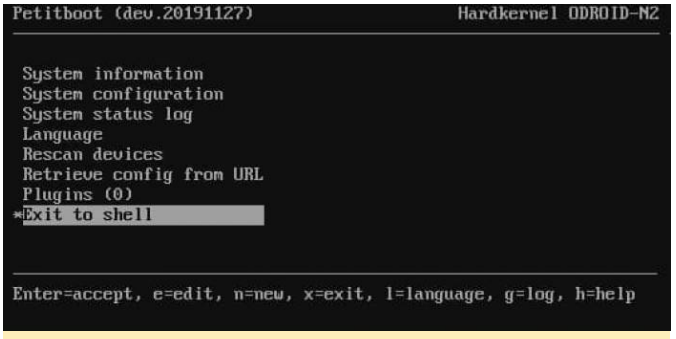


Figura 5: selecciona "Exit a Shell"

Consulta la lista de nodos de dispositivos de almacenamiento:

```
$ ls /dev/mmc*
```

Configura el dispositivo de almacenamiento en el ODROID-N2 como un dispositivo de almacenamiento masivo usando "ums" (modo de almacenamiento masivo USB). Esto permite que ODROID-N2 y OTG actúen como lectores de tarjetas de memoria.

```
$ ums
```

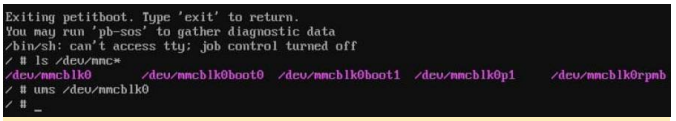


Figura 6 - Ejemplo de configuración de UMS

Espero que tu PC detecte el nuevo dispositivo de almacenamiento masivo.

Imagen del SO Flash a la memoria

Cuando sea detectado el dispositivo, la PC host reconocerá tu N2 como un lector de tarjetas USB. Por consiguiente, para grabar la imagen del sistema operativo en la memoria, consulta la guía de instalación del sistema operativo, a excepción de algunos que usan un lector de tarjetas de memoria, disponible en: **https://wiki.odroid.com/getting_started/os_installation_guide**

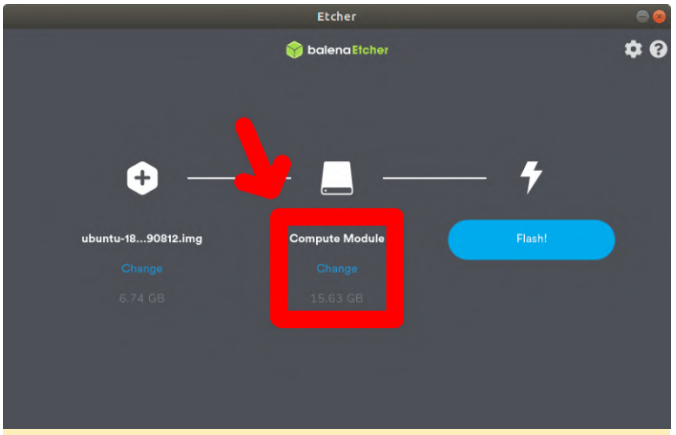


Figura 7 - Usa Etcher para escribir la imagen del sistema operativo en el N2

Cuando termines, presiona Ctrl + Alt + Supr y arranca el sistema operativo instalado. Consulta la sección del sistema operativo que aparece a continuación.

Arranque directo

1. Apaga el ODROID
2. Cambia el interruptor del modo de inicio al modo de inicio MMC
3. Enciende el ODROID

Arranque via Petitboot

1. Apaga el ODROID
2. Cambia el interruptor del modo de inicio al modo de inicio SPI
3. Enciende ODROID
4. Selecciona 'Rescan devices'
5. Selecciona la partición de arranque

Configurar Autoboot

1. Selecciona 'Rescan devices' 2. Selecciona 'System configuration' 1. Selecciona 'Autoboot' (*) Enabled 2. Selecciona 'Boot Order' 1. Limpiar 2. Añadir dispositivo 3. Ajusta 'Timeout' (recomendables 10 segundoas o mas) 4. OK 3. Presiona Ctrl + Alt + Delete

Figura 8 - Configuración de Petitboot

El documento original de la wiki lo tienes disponible en **https://wiki.odroid.com/getting_started/petitboot/os_installation_using_otg**.

Usando I2C en ODROIDs con Android Things

February 1, 2020 By @Luke.go Android, Tutoriales

Este es la continuación del **artículo inicial "Android Things" del número de Enero 2020 de la revista**, en el que se detalla cómo usar el nuevo sistema operativo de Google que facilita el uso de los pines GPIO en dispositivos ODROID

I2C

I2C-2		I2C-3	
Name	Pin Number	Name	Pin Number
SDA	3	SDA	27
SCL	5	SCL	28

Figura 01 - Tabla I2C disponible

También puedes usar I2C en la placa ODROID con Android. Puedes usar cualquier API I2C proporcionada por Android. Android Things admite varios tamaños de transmisión de datos, bytes, palabras y datos almacenados en búfer.

Exporté el ejemplo de Weather board 2 (**https://www.hardkernel.com/product-category/sensor/**) a Android con Android things. También exporte la pantalla I2C (**https://www.hardkernel.com/product-category/display/**). Al igual que otros dispositivos I2C conocidos, estos dos dispositivos están conectados con 4 cables, voltaje, tierra, I2C SDA e I2C SCL. En los ejemplos, conecté los cables I2C al I2C-2.

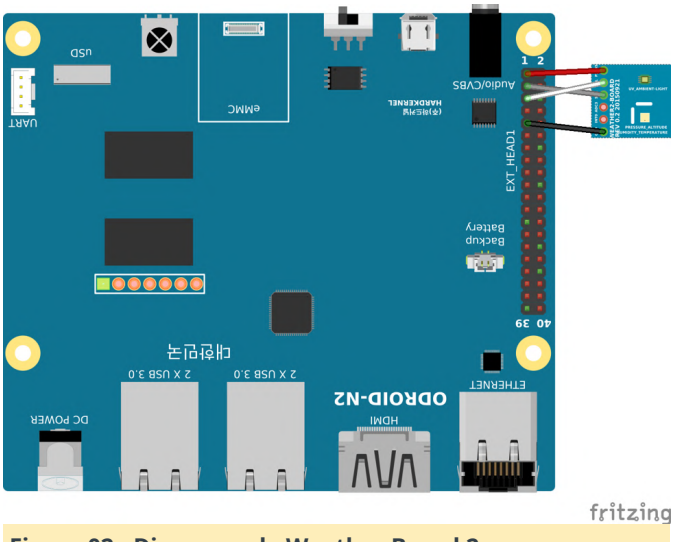


Figura 02 - Diagrama de Weather Board 2

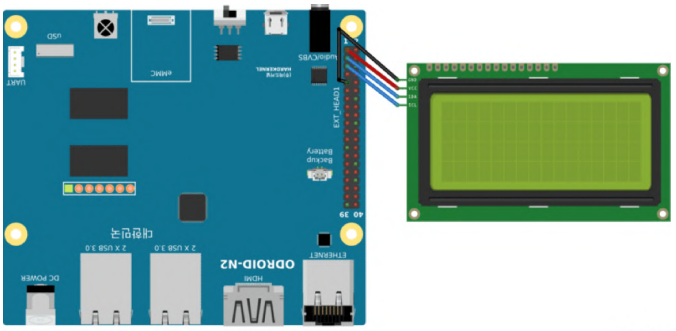


Figura 03 - Diagrama LCD I2C

La mayoría de los procedimientos previos son iguales al procedimiento GPIO. Añadir permiso al manifiesto, importar y llamar la instancia de Peripheral Manager al código fuente del proyecto. Sin embargo, no necesitas coger una instancia de GPIO. Debes llamar al método openI2cDevice para obtener la instancia del dispositivo I2C.

```
...
List i2cBusList = manager.getI2cBusList();

I2cDevice device  =
manager.openI2cDevice(i2cBusList.get(0),
                      I2C_DEVICE_ADDRESS);
// or i2cDevice device =
manager.openI2cDevice("I2C-2",
//                          I2C Device Address);
...
```

Los nombres de la interfaz I2C son I2C-2 e I2C-3. Cada interfaz I2C consta de los pines 3,5 y 27, 28. Cuando obtengas el dispositivo de bus I2C, debe configurar la dirección del dispositivo I2C para cada chip I2C. En este caso, la weather board 2 consta de dos chips I2C. De modo que, creé dos instancias de dispositivo I2C. Una instancia está vinculada por 0x76 al BME280. El chip ofrece valores de temperatura, presión y humedad. Y la otra instancia está vinculada por 0x60 al SI1132. El chip ofrece valores UV, visibles e IR. Y la LCD I2C tiene un chip I2C, así que creé una instancia de I2C. Se vincula por 0x27 para controlar la pantalla LCD. De esta manera, debes crear una instancia de dispositivo I2C para cada dispositivo con su propia dirección

A través de la instancia I2C, puede comunicarte con el dispositivo. Android things proporcionan muchos métodos. Para leer los datos de un dispositivo, proporciona los métodos read, readRegBuffer, readRegByte y readRegWord. También para escribir datos en el dispositivo, proporciona write, writeRegBuffer, writeRegByte y writeRegWord. En el sitio web oficial de Android Things tienes mucha información al respecto.

Referencia del método del dispositivo I2C - https://bit.ly/36yY6N0.

Usando la API I2C, compilé una clase envoltura para weather board2 y I2C LCD. Aquí tienes una parte del código de ejemplo para leer y escribir datos con la API de Android things.

```
…
private void softrst() throws IOException {
    device.writeRegByte(reg.RST,
POWER_MODE.SOFT_RESET_CODE);
}

private byte getPowerMode() throws IOException {
    return (byte)
(device.readRegByte(reg.CTRL_MEAS) & 0b11);
}
...
```

El código es parte de BME280.java. El primer método se llama para reiniciar el chip y el segundo método para obtener el modo de energía del chip. El primer parámetro de cada API es la dirección del registro en el chip. En el método de escritura, el segundo parámetro suele ser la información a transferir. En el método de lectura, el segundo parámetro generalmente no existe. Sin embargo, si desea leer datos por búfer, necesita un búfer para leer y el búfer se pasa como un segundo parámetro.

Puedes probar o usar el proyecto. Aquí tienes el link.

Weather board2 con ejemplo de android things - https://bit.ly/3aKqjnr. Ejemplo de I2C LCD con android things - https://bit.ly/2RUvIzJ.

A continuación, se muestra la conexión del hardware de la Weather board 2:

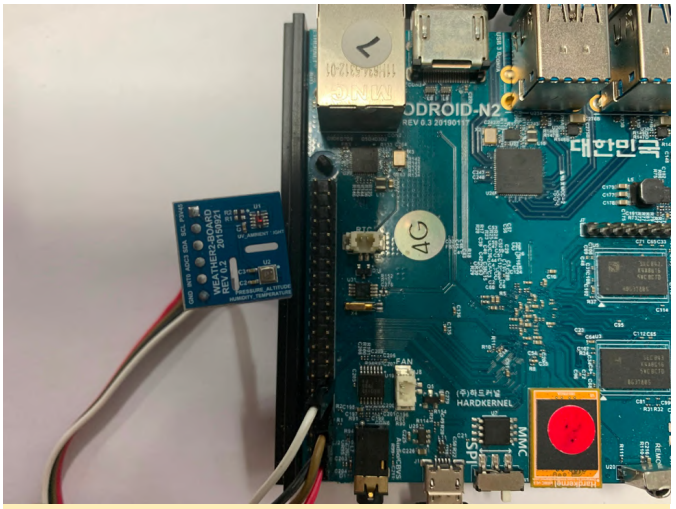


Figura 04

El resultado de la salida Weather Board 2 sería así:

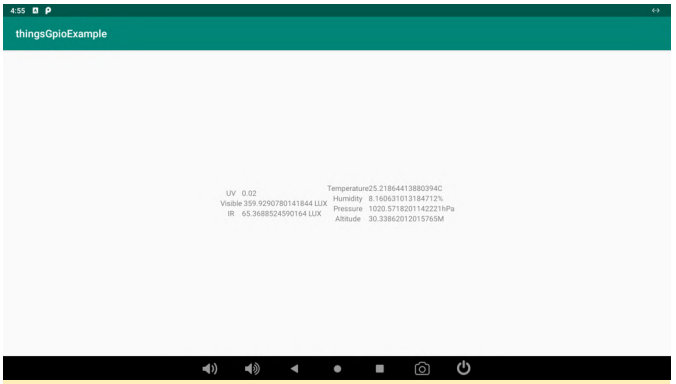


Figura 05

A continuación, se muestra la conexión y el resultado del hardware LCD I2C:

Figura 06

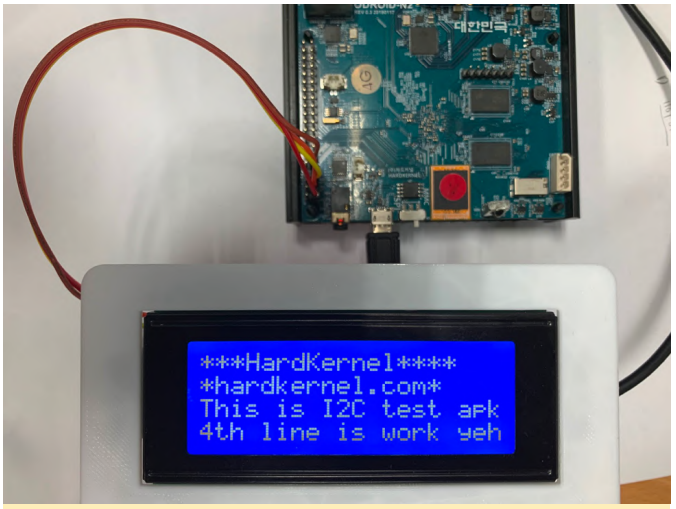


Figura 07

PWM

Name	Pin number
PWM1	12
PWM2	15
PWM3	33
PWM4	35

Figura 08 - Tabla PWM disponible

Las cosas de Android también son compatibles con PWM. Existen muchos métodos para configurar y controlar la interfaz PWM. Puedes configurar la frecuencia PWM a través de setPwmFrequency. Antes de habilitar el pin, debe establecer la frecuencia a través de este método. También puede fijar el ciclo de trabajo PWM estableciendo setPwmDutyCycle entre 0 y 100. Las configuraciones de frecuencia y ciclo de trabajo se pueden ajustar en estado habilitado y deshabilitado y serán recordadas.

Por favor, consulta la referencia. https://developer.android.com/things/sdk/pio/pwm.

Aquí está el proyecto de prueba PWM. En este ejemplo, puede activar y desactivar un pin PWM. y cambiar el ciclo de trabajo a través de la barra de progreso en la Aplicación: https://github.com/xiane/thingsGpioExample/tree/pwm.

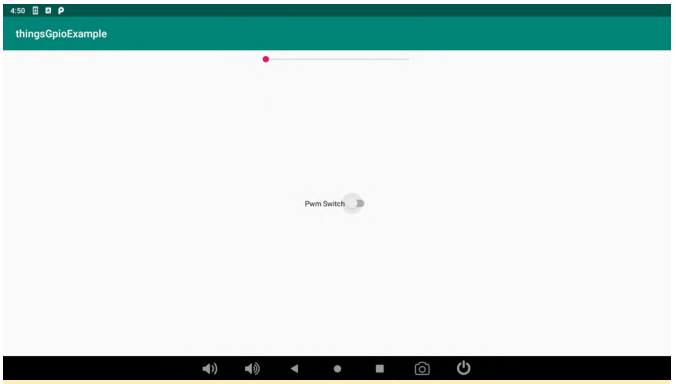


Figura 09 - Estado PWM OFF

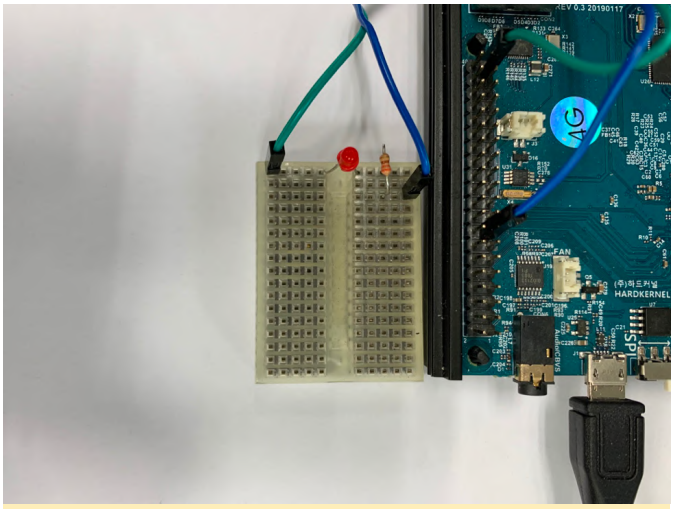


Figura 10 - PWM LED OFF

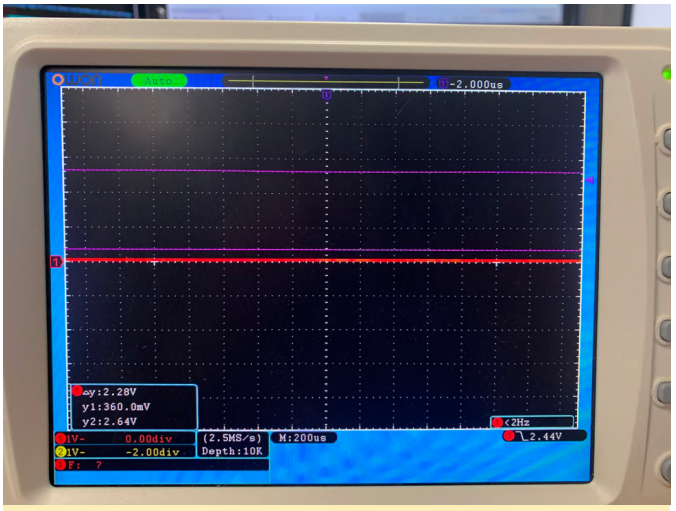


Figura 11 - PWM LED scope OFF

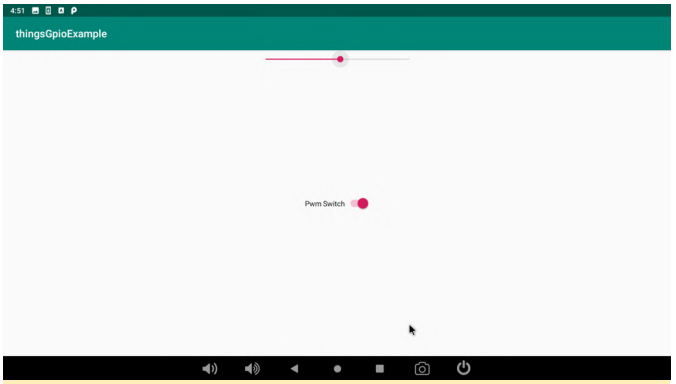


Figura 12 - PWM ON 50%

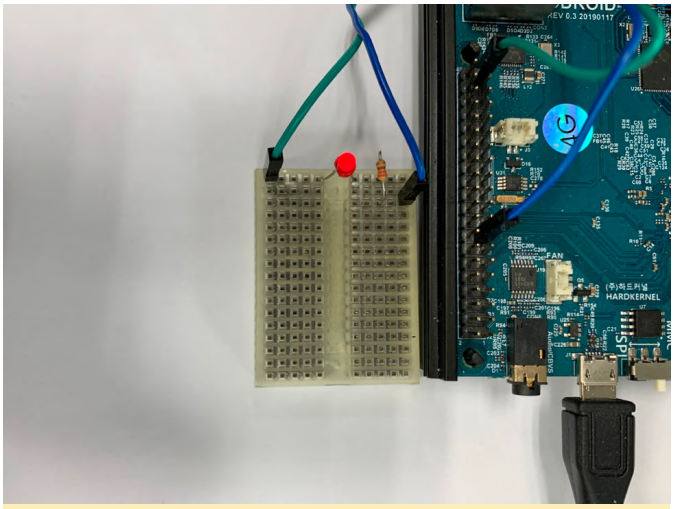
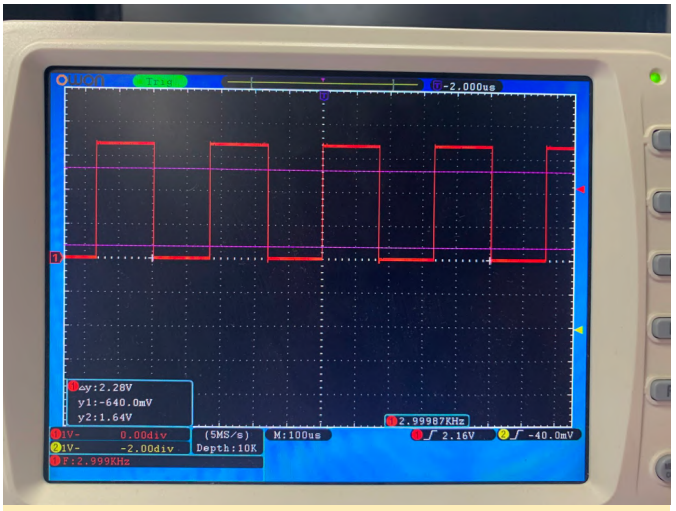


Figura 13 - PWM LED ON



Figurea14 - PWM LED scope ON

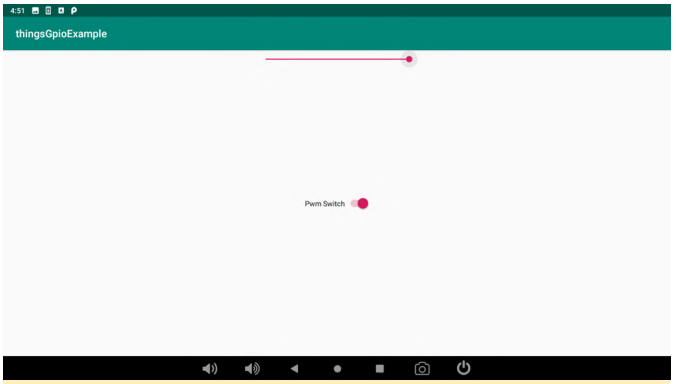


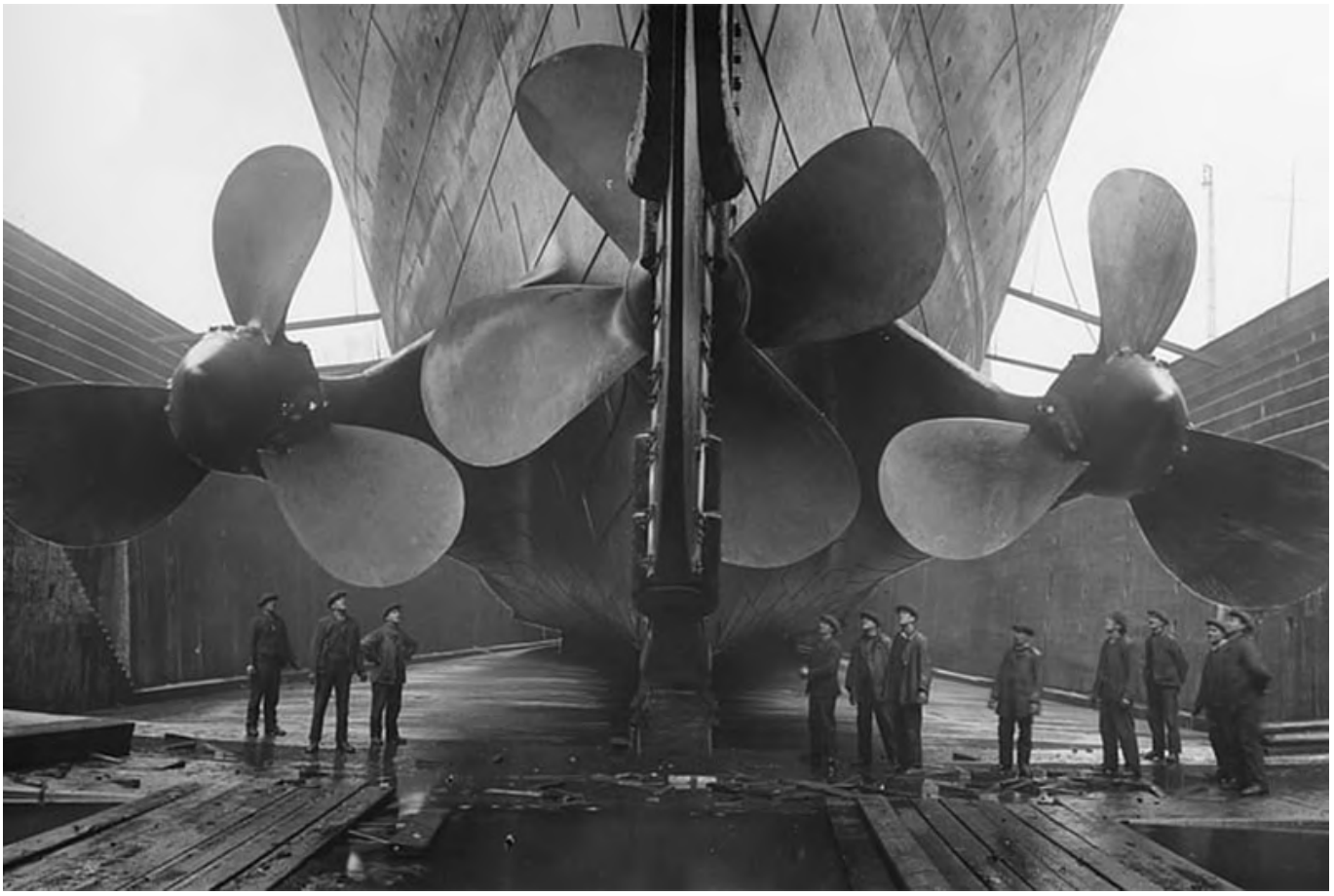
Figura 15 - PWM ON 100%

Ten en cuenta que el voltaje en los pines GPIO en ODROID-N2 es de 3.3V.

Referencia **https://bit.ly/36qpFrW**

Regulador del Ventilador ODROID-XU4

February 1, 2020 By @lbseale Linux, ODROID-XU4



Este artículo trata sobre el regulador del ventilador Hysteresis de ODROID-XU4. Cuando se activa el ventilador, permanece encendido por un tiempo. Enfría la CPU lo suficiente para luego apagarse. A medida que la CPU se calienta, el ventilador gira más rápido. Por ejemplo, digamos que el punto de arranque del ventilado está en los 60 grados C y la histéresis es de 8 grados C. El ventilador se activará cuando la temperatura alcance los 60 grados C, pero no se apagará hasta que la temperatura alcance (60-8) = 52 grados C. Para obtener más información sobre este tipo de controlador, consulta el artículo de Wikipedia en **https://bit.ly/2NWCowb**. Ten en cuenta que este controlador solo funciona con Ubuntu en el ODROID-XU4, y no está diseñado para ningún otro ordenador de placa reducida o sistema operativo.

Instalación

Descargate el instador **https://bit.ly/2RKqRB4** desde GitHub. Para instalarlo usando el instalador GUI, haz doble clic en el archivo xu4fan-installer.deb. Luego haga clic en el botón “Install Package” en la esquina superior derecha de la ventana. Para instalarlo usando la línea de comando, utiliza:

```
$ sudo apt install xu4fan-installer.deb
```

Para desinstalar, usa:

```
$ sudo apt remove xu4fan
```

El controlador del ventilador se iniciará automáticamente después de tu instalación.

Configuración

El archivo de configuración para el regulador del ventilador se encuentra en /etc/xu4fan/xu4fan.conf Puedes usar este archivo para cambiar la configuración del regulador del ventilador, como los puntos de disparo y la histéresis Debes reiniciar el regulador del ventilador después de cambiar el archivo de configuración. Para hacer esto usa:

```
$ sudo systemctl restart xu4fan.service
```

Opciones de Configuración

trip_temps: Lista de temperaturas correspondientes a los valores PWM. Cuando la temperatura aumenta y se alcanza una de estas temperaturas, se fijará el valor PWM correspondiente. Las unidades son Deg C * 1000. Ejemplo: [60000, 70000, 80000]

trip_speeds: Lista de valores de PWM correspondientes a las temperaturas. Las unidades son valores PWM en el rango (0-255). Un valor de 120 es (120/255) = 47% de la potencia máxima del ventilador. A medida que la CPU se calienta, el ventilador gira más rápdio. Ten en cuenta que los valores inferiores a 120 no son lo suficientemente potentes como para hacer girar el ventilador de serie. Ejemplo: [120, 200, 240]

hysteresis: Número de grados C más allá del punto de disparo que debe alcanzar la temperatura para caer al punto de disparo previo. Si el punto de disparo es de 60 ° C y la histéresis es de 8 ° C, entonces la temperatura debe caer por debajo de (60-8) = 52 ° C para que el ventilador se apague. Las unidades son Deg C * 1000. Ejemplo: 8000

poll_interval: Número de segundos para que el controlador del ventilador espere entre comprobaciones de temperatura. Ejemplo: 0.25

verbose: Si es True, aparece un mensaje en syslog cada vez que el ventilador cambia de velocidad. No es necesario cambiar las opciones para las secciones [Thermometer] y [Fan]. Estas son específicas del ODROID-XU4.

Servicio Systemd

El regulador del ventilador lo ejecuta un servicio systemd. Que se inicia automáticamente cuando se instala. También se iniciará automáticamente cuando se inicie tu ODROID-XU4.

Para ver su estado, usa:

```
$ sudo systemctl status xu4fan.service
```

Para pararlo, usa:

```
$ sudo systemctl stop xu4fan.service
```

Para deshabilitarlo y no permitir que se inicie automáticamente, usa:

```
$ sudo systemctl disable xu4fan.service
```

Para habilitarlo, usa:

```
$ sudo systemctl enable xu4fan.service
```

Archivos de muestra

Tienes un archivo de configuración de muestra junto con un archivo .service de systemd en /usr/share/xu4fan/.

Referencía

https://github.com/lbseale/odroid-fan

El Punto G: Tu Destino para Todas las Cuestiones Relacionadas con Juegos Android: Los Juegos de Mesa no son tan Aburridos como Parece

February 1, 2020 By Dave Prochnow Juegos

¿Piensas que los juegos de mesa son aburridos? Estas actividades de entretenimiento analógico en papel tienen una larga trayectoria, o esta tradición se ha desvanecido y ha sido reemplazada por los juegos Android de ODROID de hoy en día. Pues bien, Dire Wolf Digital es un tanto diferente. Esta empresa de desarrollo de juegos con sede en Denver, Colorado, se ha forjado un nicho bastante importante por sí misma, al coger juegos de mesa y convertirlos en obras maestras de juegos digitales.

Figura 1. Se escucha el aullido del gigante del juego de mesa Dire Wolf Digital.

Este alocado y arriesgado cambio comenzó hace varios años cuando Renegade Game Studios buscó a Dire Wolf Digital para llevar dos de sus juegos de mesa al mundo digital. "Lanterns: The Harvest Festival" y "Lotus" ofrecen una "experiencia de usuario única" (UX) y un sistema de "juego táctil intuitivo" que se convertiría en el sello distintivo de los futuros esfuerzos de remasterización de Dire Wolf Digital.

"Pass GO and Collect 200$"- Actualmente, Dire Wolf Digital ha anunciado que todo 2020 nos deleitará con una enorme lista de juegos de mesa de su red digital. Estas ofertas representan una increíble mezcla de títulos que será la envidia de TODO desarrollador de Android.

No todos los títulos anunciados son compensaciones "únicas" en las que un juego de mesa equivale a un título digital. Algunos, como "Mage Knight" de WizKids, son solo un título de un catálogo completo

de juegos que recibirán un tratamiento digital. Si bien es posible que no hayas oído hablar de algunos (o ninguno) de estos juegos de mesa, son increíblemente populares entre los jugadores de mesa. Por ejemplo, Dire Wolf Digital nos ha proporcionado un fragmento del análisis de "Rock, Paper, Shotgun:"

Figura 2. "Mage Knight" es el lanzamiento inicial de la alianza entre WizKids y Dire Wolf Digital.

'" Mage Knight "es un juego asombroso. Es uno de esos juegos que solo puedes retroceder y admirarlo como algo increible, una obra de arte. Es una obra maestra. No puedo creer lo bueno que es. Gosh, gush, ¿verdad Hay muchos juegos de Android que anhelarían ese tipo de crítica.

Aunque WizKids es un creador de juegos de mesa tradicional, se ha forjado otra alianza con Dire Wolf Digital con el juego Kickstarter 2018, "Root". Este "juego de guerra" entre criaturas de bosque y un gato es la creación de Cole Wehrle y Leder Games. Una vez más, nos han proporcionado una crítica brillante de este juego que parece empapar a sus jugadores con el afecto antropomórfico de los protagonistas del juego.

Figura 3. Tus ojos también serían muy grandes, si tuvieras que luchar contra un gato monstruo asesino con colmillos afilados.

Continuando con la lista de éxitos de Dire Wolf Digital 2020, llegamos a “Sagrada”, un juego de dados, vidrieras y maravillas arquitectónicas. Sí, lo has leído bien; tres de los compañeros de cama más extraños que jamás verás en un juego de Android. Considerado como un juego de "dibujo de dados", "Sagrada" es un título de 2017 de Floodgate Games que obtuvo un puñado de nominaciones a los premios, además de ser etiquetado como el juego más innovador del año.

Figura 4. Combinar en un juego dados con vidrieras y la arquitectura de Antoni Gaudí es innegablemente algo muy innovador.

Si te encantan los juegos basados en fichas, entonces deberías probar el género de los juegos de mesa. Es decir, hasta que Dire Wolf Digital traiga "Yellow & Yangtze" de Reiner Knizia a tu ODROID. Este es un juego de estrategia/combate salvaje similar al

aclamado juego "Tigris and Éufrates" de Knizia. En cada título, haces crecer tu imperio; pero cuidado. El ganador del juego no solo es juzgado por la prosperidad de su reino, sino también por la fortaleza de su activo más débil. Así que haz las cosas bien manteniendo un equilibrio durante toda tu dinastía.

Figura 5. El lanzamiento más reciente de la colección de juegos de estrategia de Reiner Knizia.

Buenas noticias: Justo cuando esta columna se iba a imprimir, "Reiner Knizia's Yellow & Yangtze" apareció en Google Play. Increíblemente, ya hay también una actualización (es decir, Actualización 1.17). Lo mejor de esta actualización es poder: "ralentizar la animación de las revueltas de los campesinos, lo cual es obvio cuando un oponente hace esta jugada". ¡Hurra!

Figura 6. Juego ACTIVADO; pero asegúrese de vigilar tus recursos.

Los fanáticos del antiguo juego de mesa American Heritage de 1964, "Dogfight", definitivamente querrán ver los títulos de Dire Wolf Digital, "Wings of Glory" y "Tripods and Triplanes". Sentados exactamente en los extremos opuestos de la fiabilidad histórica, estos títulos de Ares Games te llevarán a la cabina de mimbre de un avión de 1917 para enfrentarte al "rayo de la muerte" de una nave espacial extraterrestre en una adaptación posterior de la "Guerra de los Mundos". Del mismo modo, como sugiere el título de este último, combatirás a Martian con los aviones de la época de 1917.

Figura 7. Invasores marcianos de un nuevo enfrentamiento de H. G. Wells con los aviones de la Primera Guerra Mundial.

Puedes ver un tutorial de juego en https://www.youtube.com/watch?v=IVJi-54hb5c.

Finalmente, volvemos a donde empezaron todos estos juegos de mesa con otro título de Renegade Game Studios. "Raiders of the North Sea", que no es un juego de guerra submarina durante las guerras mundiales, sino más bien un juego de saqueo de la conquista vikinga. Es muy simple; construye un bote (es decir, un bote vikingo) vendrán y lo saquearán. Lo mejor de todo, es que este título ya está disponible para jugar en Google Play.

Figura 8. Remar, remar, remar su bote, suavemente para saquear todo el Mar del Norte; felizmente, felizmente disfrutando del credo de los vikingos.

Puedes ver el avance en https://bit.ly/2Rp7q1C.

Así que mantente atento a Dire Wolf Digital en la tienda Google Play. Hasta entonces, desempolva tu copia de "Monopoly" del armario del pasillo y dale el reconcomiendo que se merece a los venerables juegos de mesa. Puedes obtener más información sobre todos estos juegos en el sitio web de Dire Wolf Digital en http://www.direwolfdigital.com.

Las Mejores Opciones de Software como Servidor Multimedia ODROID-XU4

February 1, 2020 By Moe Long, www.cupofmoe.com ODROID-XU4, Tutoriales

Aunque la Raspberry Pi es un ordenador de placa reducida (SBC) muy popular, el ODROID-XU4 es un sólido competidor. Manteniendo un tamaño físico pequeño, el ODROID-XU4 tiene un gran rendimiento. Potente y muy eficiente desde el punto de vista energético, con un procesador ARM big.LITTLE, el ODROID-XU4 incluye una CPU de ocho núcleos Samsung Exynos Cortex-A15 2GHz y Cortex-A7. Cuenta con una GPU Mali-T628 con capacidades OpenGL ES 3.1 y OpenCL 1.2. Integrados los 2 GB de RAM LPDDR3 hacen que la multitarea sea muy fácil.

Entre los mejores usos para un ODROID-XU4 está el de Servidor multimedia. Debido a su pequeño tamaño, pero sus especificaciones robustas, el ODROID-XU4 consume una mínima energía mientras maneja los procesos de almacenamiento conectado a la red (NAS) como un campeón. ¡Desde Plex y Emby hasta Owncloud y OpenMediaVault, estas son las mejores opciones de software de servidor de medios para el ODROID-XU4!

Montar un servidor ODROID-XU4 con Ubuntu, Debian, imágenes independientes, etc.

La mayoría de los usuarios probablemente instalarán un sistema operativo Linux como Ubuntu o Debian. Luego, dentro del sistema operativo host, puede instalar servicios NAS. DietPi es una de las mejores opciones, además de necesitar poco espacio en disco y contar con una instalación modular que te permite descargar fácilmente un servidor multimedia y un servidor de archivos, programas. Otra posibilidad es recurrir a una imagen de servidor independiente. En cualquier caso, el ODROID-XU4 ofrece la posibilidad de crear un NAS potente y de bajo consumo por muy poco. El kit NAS CloudShell 2 incluso permite instalar

hasta dos discos duros de 3.5", y ejecuta SPAN, RAID0, RAID1 y JBOD.

Si optas por una carcasa sin conexiones de disco duro integradas, deberás decidir cómo quieres almacenar tus archivos multimedia. Afortunadamente, con sus puertos USB 3.0, el ODROID-XU4 puede acceder rápidamente a los archivos multimedia de las unidades conectadas. Si tienes una unidad multimedia en red, puedes instalar el software del servidor en el ODROID-XU4 y simplemente conectarte a un recurso compartido Samba u otro dispositivo de almacenamiento de archivos en red. Para conectar unidades directamente al ODROID-XU4, es posible que necesites una fuente de alimentación más potente, como una de 5V/6A.

Consideraciones sobre el servidor de medios Odroid XU4:

- Carcasa
- Almacenamiento multimedia (unidades conectadas directamente frente a unidades en red)
- Imagen de servidor independiente frente a un sistema operativo host con software de servidor multimedia
- Sistema operativo host (Debian, Ubuntu, DietPi, etc.)
- PSU

Servidor Multimedia Plex

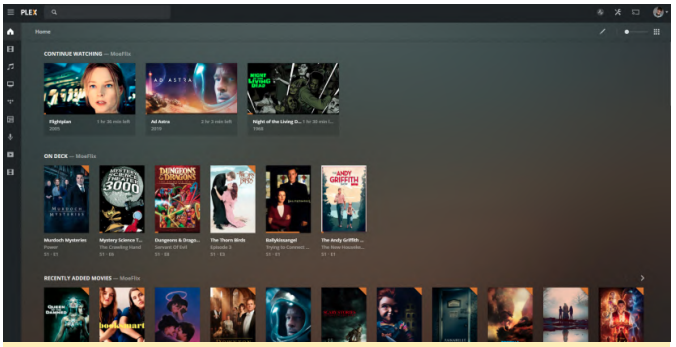


Figura 1 - Plex

Ideal para: Usuarios principiantes, intermedios y avanzados de servidores multimedia

Plex sigue siendo una de las mejores aplicaciones de servidor multimedia disponibles, y un servidor multimedia Plex ODROID-XU4 funciona extremadamente bien. Algo así como Netflix, pero hecho por ti mismo (DIY), Plex te permite alojar tu colección personal de medios, como películas, programas de televisión, música e incluso fotos en un servidor. Luego, puede acceder al repositorio completo de películas, TV y archivos de audio desde clientes Plex compatibles. Una vez que hayas cargado tus bibliotecas, Plex escaneará tus archivos y descargará metadatos y carátulas si están disponibles.

Cuando comenzó Plex, se centraba principalmente en organizar archivos multimedia personales. Y aunque ese sigue siendo su objetivo principal, Plex ha ido incorporando una serie de características adicionales, como películas y programas de TV gratuitos con publicidad, podcast e integración Tidal, además de la funcionalidad de TV en vivo y DVR. Puesto que es increíblemente fácil de instalar y configurar, Plex es un excelente programa de servidor para principiantes. Aun así, los usuarios avanzados pueden profundizar en su amplio abanico de configuraciones para personalizarlo. Una instalación del servidor ODROID-XU4 Plex es increíblemente fácil de improvisar. He encontrado un servidor ODROID-XU4 Plex simple de crear y funcional que permite hacer streaming en casa con más de cinco clientes conectados.

Emby

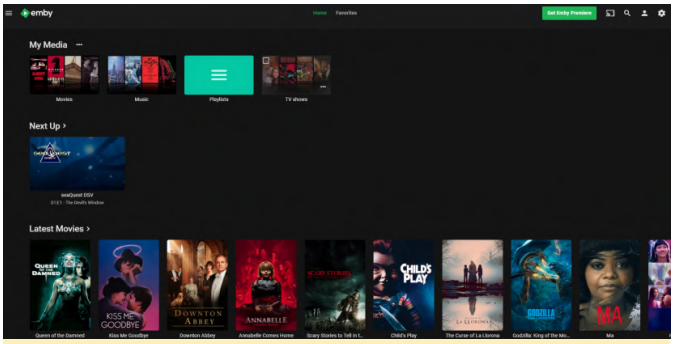


Figura 2 - Emby

Ideal para: Usuarios intermedios y avanzados de Servidor multimedia.

Del mismo modo, Emby funciona sin problemas como un servidor multimedia ODROID-XU4. Similar a Plex, Emby es una aplicación de software de servidor todo en uno que organiza tu colección personal de medios en una interfaz exuberante con metadatos y diseño de caja. Sin embargo, Emby es el más adecuado para usuarios avanzados. Aunque los principiantes pueden y deberían probar Emby, su gran lista de opciones de

personalización lo convierte en una opción sólida para usuarios avanzados.

Es cierto que el proceso de configuración de Emby no es complicado, pero las opciones adicionales, como activar la extracción de imágenes de los capítulos, lo hacen un poco más complejo que la sencilla configuración de Plex. Además, la amplia configuración para ajustar incluso la capacidad de implementar tu propio CSS en su aplicación web hacen de Emby el verdadero sueño de los entusiastas de los servidores domésticos. Al usar el ODROID-XU4 como servidor, Emby clasifica rápidamente tus archivos multimedia en varias categorías, permitiéndote hacer streaming sobre tus dispositivos cliente Emby.

Serviio

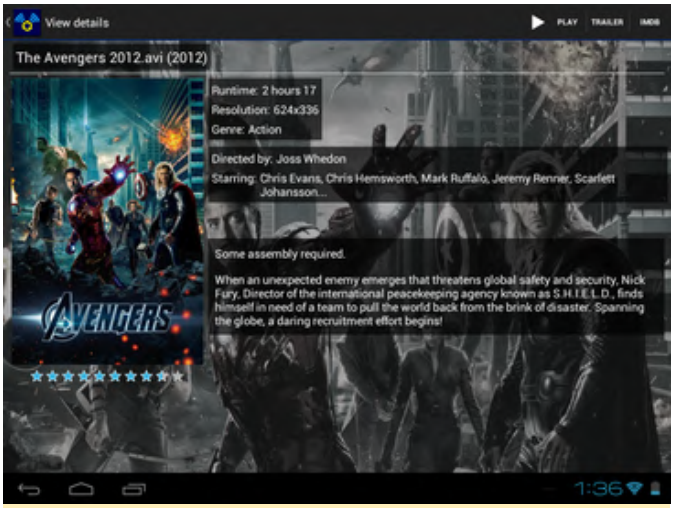


Figura 3 - Serviio

Ideal para: Usuarios principiantes, intermedios y avanzados de servidores multimedia

Para hacer streaming a varios clientes, como teléfonos, tabletas y decodificadores, Serviio es una excelente elección. Puede acceder a una gran cantidad de archivos de video y audio. Además, Serviio te permite transmitir contenido de fuentes RSS y páginas web. Puede manejar subtítulos, toneladas de formatos de listas de reproducción e incluso cuenta con soporte de imagen de cámara RAW. Tiene integración con las cosas de Trakt.tv, e incluso soporte para Alexa. Serviio es súper fácil de instalar en el ODROID-XU4, requiere poco más de una instalación JDK de Java y luego descarga el programa en sí.

OwnCloud

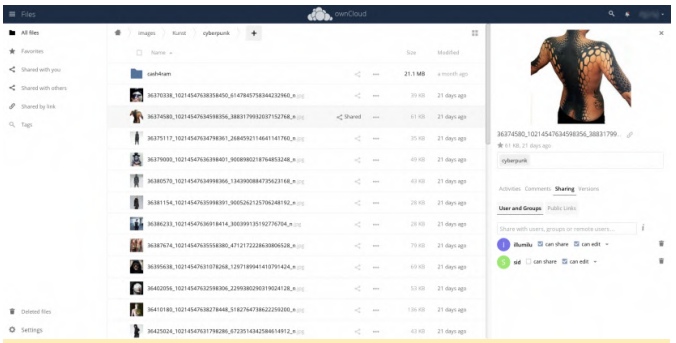


Figura 4 - OwnCloud

Ideal para: Un doble uso de servidor de archivos y multimedia

Si deseas activar un servidor NAS ODROID-XU4, OwnCloud es una excelente opción. Mientras que Emby y Plex son soluciones de servidor multimedia dedicadas, OwnCloud es una suite de servidores de archivos generales. Como tal, es más parecido a una especie de Dropbox montado por ti. Sin embargo, tus aplicaciones disponibles de reproductor de video y audio son las que reproducen el contenido multimedia. Una de las mejores soluciones es usar el protocolo WebDAV. Kodi incluso cuenta con un complemento WebDAV. OwnCloud carece de la interfaz multimedia que encuentras en las suites de servidores multimedia dedicados, pero es una excelente opción para un servidor de archivos y multimedia combinados.

OpenMediaVault

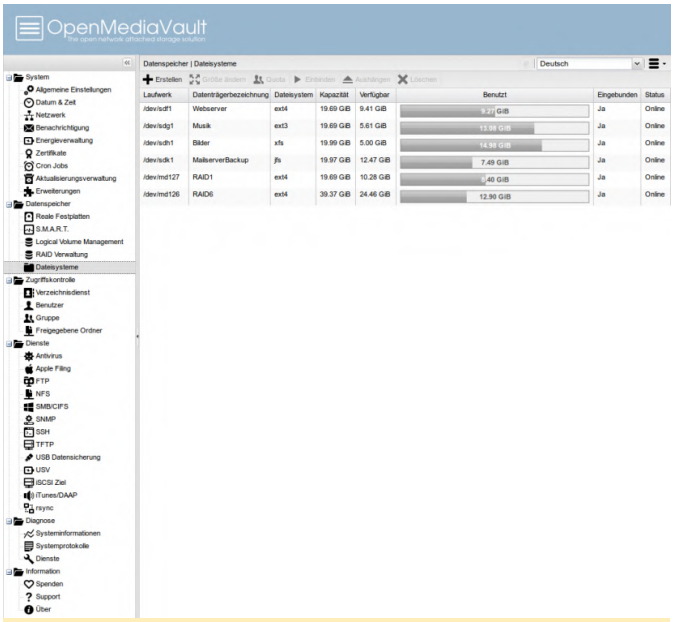


Figura 5 - OpenMediaVault

Ideal para: Un doble uso de servidores de archivos y multimedia

En lugar de simplemente instalar un sistema operativo y luego ejecutar aplicaciones de servidor multimedia, quizás prefieras una imagen de servidor ODROID-XU4 como OpenMediaVault (OMV). OMV te permite activar rápidamente un NAS completo y, por lo tanto, utilizar tu ODROID-XU4 como servidor. Esta distribución de Linux basada en Debian proporciona una instalación modular. Una vez que hayas arrancado OpenMediaVault, tendrás acceso a una gran variedad de utilidades, como clientes BitTorrent, SSH, SFTP, SMB, servidor de medios DAAP, RSync y mucho más. Además, hay un repositorio masivo de complementos que incluye complementos para Plex y Emby. Si estás buscando una imagen completa del servidor para ejecutar en el ODROID-XU4, OpenMediaVault es uno de los mejores candidatos.

NextCloud

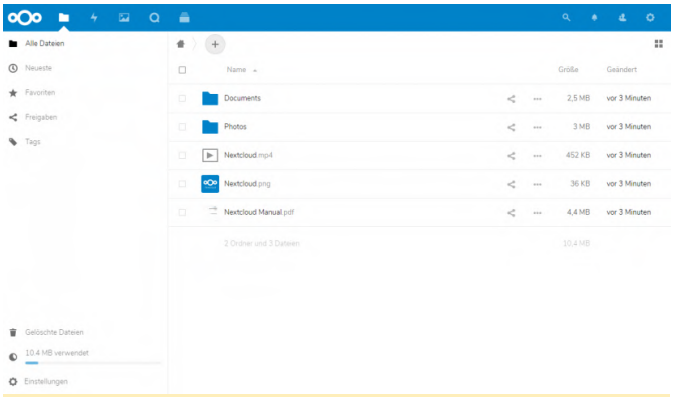


Figura 6 - NextCloud

Ideal como: Combo de servidor de archivos y multimedia

Similar a OwnCloud, NextCloud ofrece casi exactamente el mismo conjunto de características. Sin embargo, mientras OwnCloud promociona los niveles de código abierto y empresarial, NextCloud usa exactamente el mismo código para los niveles gratuitos y de pago. A pesar de su responsabilidad en el servicio de archivos, puede usar NextCloud como servidor multimedia. Dado que NextCloud incluye soporte para WebDAV, incluso puede usar el complemento WebDAV para Kodi para acceder a tus archivos multimedia desde el front-end del centro multimedia Kodi.

Reflexiones finales

El ODROID-XU4 es una de las mejores placas del mercado para creadores e inventores. Junto con el software adecuado, es un dispositivo NAS muy asequible que consume muy poca energía. Plex, Emby y Serviio son excelentes soluciones de servidor multimedia, mientras que NextCloud y OwnCloud agregan funcionalidad de servidor de archivos. Por otro lado, tienes la posibilidad de instalar una aplicación de servidor multimedia dedicada como Plex, así como un programa de servidor de archivos como NextCloud y ejecutarlos en paralelo. OpenMediaVault es una completa y excelente imagen de servidor multimedia. Un servidor multimedia con ODROID-XU4 es un proyecto simple pero funcional que maximiza las capacidades de este ingenioso SBC.

Este artículo apareció originalmente en Electromaker.io
en **https://www.electromaker.io/blog/article/best-odroid-xu4-media-server-software-options**.