PRÁCTICA 10: ENTRADA/SALIDA I

ENTRADA/SALIDA POR POLLING EN LINUX

ARQUITECTURA DE COMPUTADORES. 2º CURSO

1. OBJETIVOS.

Los objetivos globales de esta práctica se recogen en los siguientes apartados:

- Consolidar los conocimientos adquiridos sobre los mecanismos de Entrada/Salida.
- Aprender a programar dispositivos de E/S a bajo nivel en C.
- Conocer parte de la gestión de E/S en Linux.
- Ayudar a obtener soltura a la hora de manejar *datasheets*.

Esta práctica se complementará con la sesión siguiente (E/S por Interrupciones). Entre ambas prácticas se cubrirá la totalidad de los objetivos del tema.

2. PREPARACIÓN

Antes de realizar esta sesión de laboratorio, se recomienda que el alumno:

- **Repase los conceptos generales de programación en C.**

3. INTRODUCCIÓN TEÓRICA.

Con la ayuda de la práctica voluntaria de programación en C, se ha repasado dicho lenguaje. Una vez repasado el lenguaje de programación C, se llevará a cabo la finalidad de esta práctica, que no es más que el acceso por lectura y escritura a un dispositivo de E/S mediante la comunicación con su módulo/controlador correspondiente.

En primer lugar, se realizará una breve introducción al manejo de la máquina virtual proporcionada para la presente práctica, así como al entorno de Linux y su escritorio.

Posteriormente, se expondrá el esquema de conexión y obtención de datos, para permitir al alumno comunicarse con el Reloj de Tiempo Real (RTC) con el fin de obtener y modificar la fecha y hora del sistema.

A. ENTORNOS LINUX Y ZINJAI

En esta práctica se hará uso de una máquina virtual (disponible para los alumnos) con una distribución de Linux Mint 18 (basada Ubuntu 16.04 LTS) de 32 bits y el entorno de programación en C/C++ *ZinJaI* instalado. Esta máquina virtual fue diseñada con el software de virtualización gratuito Oracle VirtualBox y se distribuye como un archivo .ova. Sin embargo, este formato es compatible con el software de virtualización VMware, que es el disponible en los PCs de prácticas.

Para descargar la máquina virtual:

- Máquina virtual: http://lara.eii.us.es:8080/sharing/kSqG05Gpu

Para descargar el software de virtualización (uno de los dos):

- VMware player: http://www.vmware.com/products/player/playerpro-evaluation.html
- Oracle VirtualBox: https://www.virtualbox.org/wiki/Downloads

El alumno deberá descargar primero el archivo “AC-Linux_Mint_18.1_32_bits.ova” a su PC (este archivo ocupa en torno a los 3 GB y por lo tanto tardará un rato en ser descargado).

Hay dos opciones de trabajo con el software de virtualización y el archivo o máquina virtual a importar “AC-Linux_Mint_18.1_32_bits.ova”: usar el programa de virtualización Oracle VirtualBox

o VMware. A veces en el aula de prácticas podrán estar instalados uno o los dos programas de virtualización. No obstante, y como se ha dicho anteriormente, puesto que la máquina virtual fue creada con Oracle VirtualBox, en caso de disponer de este software en el aula de prácticas, se recomienda su uso en lugar de VMware. Esto es debido a que la importación y ejecución de la máquina virtual es mucho más rápida que con VMware. A continuación, se explica el proceso para el software o programa VMware. El proceso de importación y ejecución con Oracle VirtualBox se detalla en el Anexo I.

Antes de importar la máquina virtual es conveniente que abra el programa WMware para comprobar que la máquina no proviniera de una sesión anterior (ver Figura 1). Si aparece la máquina virtual "AC-Linux_Mint_18.1_32_bits" deberá eliminarla seleccionándola con el menú contextual (botón derecho) y opción "Delete".

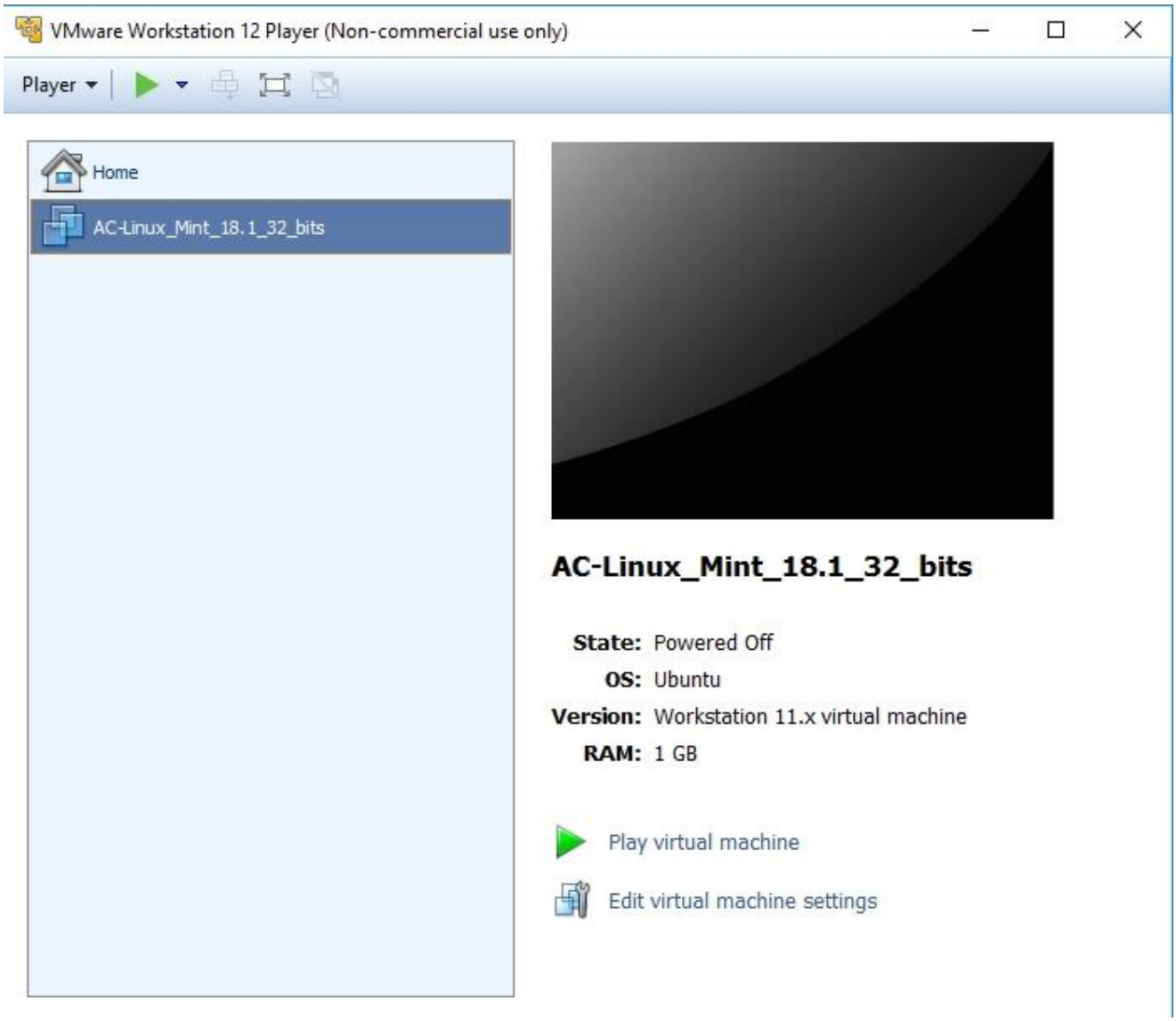


Figura 1. VMware con máquina Linux importada y lista para ser usada.

La forma de importar "AC-Linux_Mint_18.1_32_bits.ova" consiste en directamente abrirlo con el VMware. Se puede hacer haciendo doble click con el ratón o mediante el menú contextual seleccionando "Open with VMware Player". A continuación, aparecerá la pantalla de la Figura 2 en la que se pedirá confirmación para importar la máquina virtual a VMware. Hay confirmar y pulsar "Import".

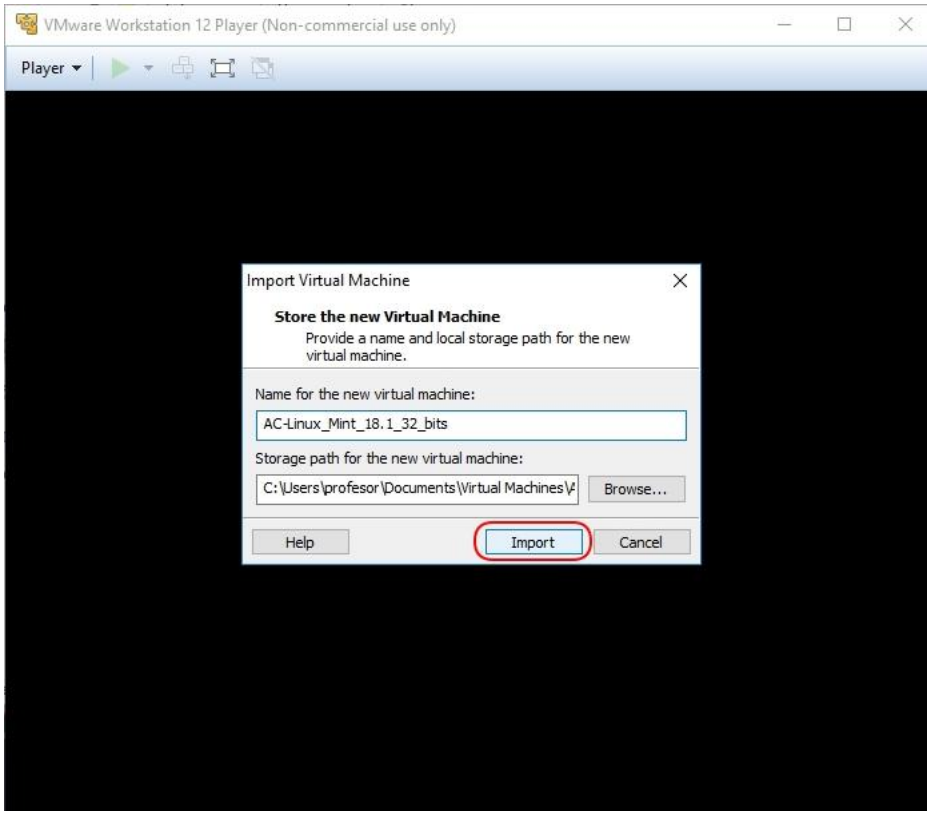


Figura 2. Importación de la máquina virtual con extensión .ova a VMware

A veces puede suceder que como la máquina virtual se ha creado con otro software, VMware tenga algún problema y muestre un aviso (ver figura 3). En este caso simplemente se pulsa “Retry” y se sigue con el proceso de importación.

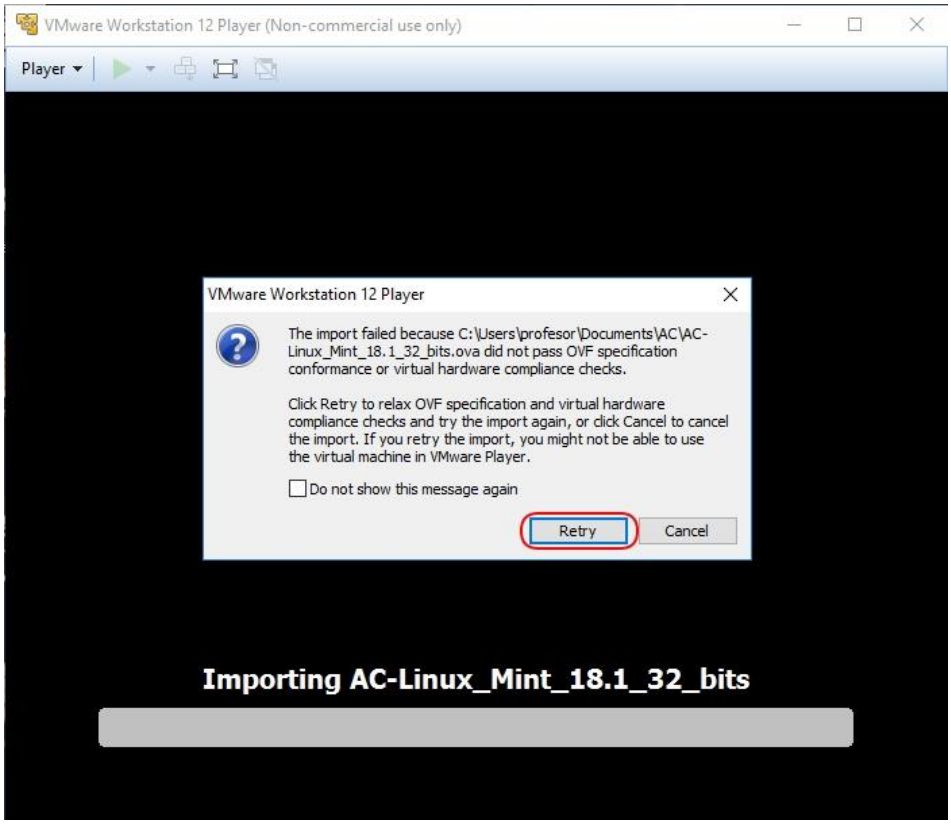


Figura 3. Aviso VMware al importar una máquina virtual con extensión .ova que ha sido creada con otro software.

Una vez arrancada la máquina virtual aparecerá el escritorio que se muestra en la Figura 5. No obstante previamente deberá ajustar si acaso la resolución del escritorio (por defecto puede estar a 800x600 píxeles) mediante el icono "Resolución de la Pantalla" (ver Figura 4) que aparece en el escritorio de la máquina virtual Linux. Para ello compruebe previamente en Windows o el sistema anfitrión, cuál es la resolución de su monitor para usar la misma resolución en la máquina virtual.

Figura 4. Icono de cambio de la resolución de la pantalla en Linux.

Una vez hecho el ajuste de la resolución podrá maximizar todo el escritorio de la máquina virtual Linux para que ocupe toda la pantalla. A partir de este punto se trabajará exclusivamente con el escritorio de la máquina virtual Linux (ver Figura 5). Si tuviera que cerrar la máquina virtual Linux por cualquier motivo (botón "Menú" de la esquina inferior izquierda + "Salir"), podría volver a arrancarla mediante VMware e inicio (ver Figura 1).

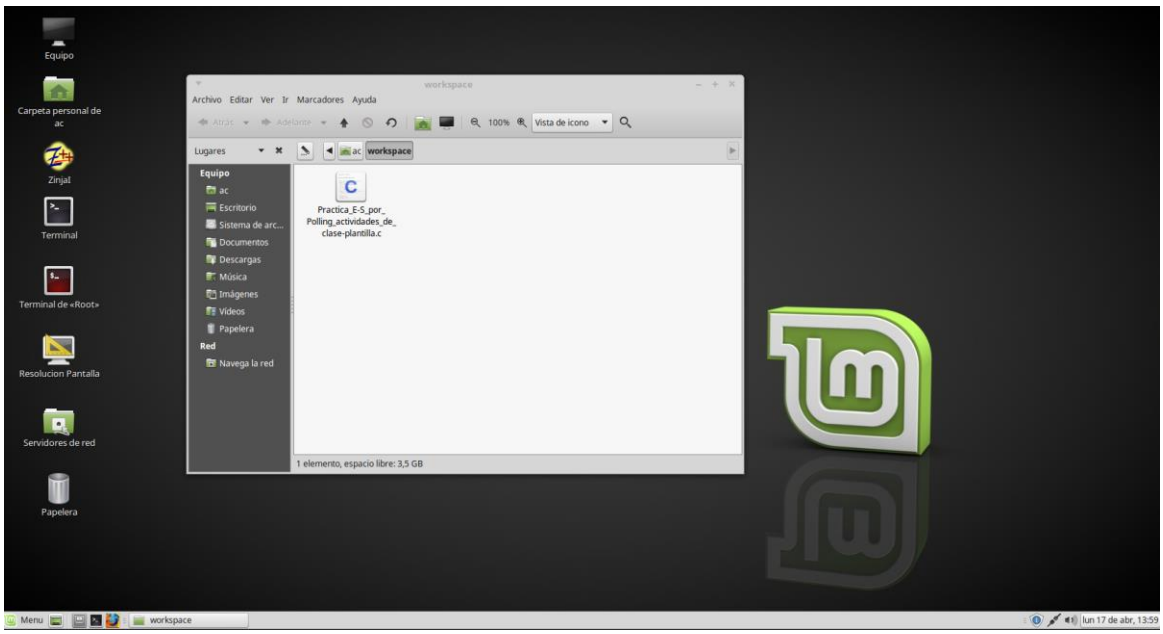


Figura 5. Escritorio inicial al arrancar la máquina virtual.

En el escritorio inicial se muestra a la izquierda una serie de iconos. Los que vamos a utilizar principalmente son el icono del gestor/explorador de archivos y carpetas, el propio de ZinJaI, los de terminal o consola de comandos con y sin permisos de *root* o superusuario (ver Figura 6).

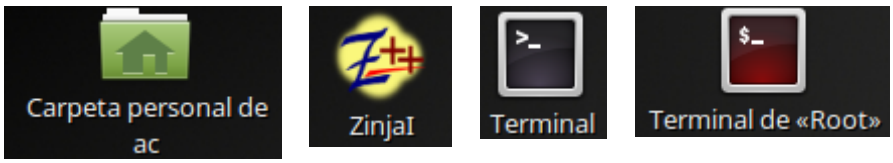


Figura 6. De izquierda a derecha: iconos de gestor archivos y carpetas, ZinJaI, terminal o consola de comandos, y terminal o consola de comandos con privilegios de administrador o superusuario.

El usuario y contraseña de acceso a dicha máquina virtual serán:

Usuario: ac
Contraseña: ac

Este usuario tiene permisos para ejecutar como superusuario. Puesto que en esta práctica se va a ejecutar programas que tienen acceso directo a registros de módulos con controladores de E/S, es necesario hacerlo con los máximos permisos. Para ello basta con teclear en una consola de comandos o terminal el comando “sudo” + “nombre_archivo_o_comando”. Alternativamente puede ejecutar la consola de comandos de “Root” o superusuario, en cuyo caso no hará falta teclear el comando “sudo”. Otra forma alternativa de ejecutar comandos o programas con privilegios de superusuario es entrar en una consola o terminal sin privilegios de “Root” o superusuario y después teclear el comando “su”. Pedirá a continuación una contraseña que será “ac”.

Como se puede ver, ya tenemos el acceso en el escritorio al entorno de desarrollo ZinJaI que utilizaremos para programar en C. Como alternativa se puede hacer doble click sobre un archivo .c y se abrirá junto con el entorno ZinJaI. En la práctica el archivo sobre el que se va a programar está en el directorio /home/ac/workspace y también hay una copia en el directorio /home/ac/Documentos. Puede acceder a ellos a través del gestor de archivos y carpetas.

Una vez cargado el entorno de programación ZinJaI y completada la plantilla del programa, habrá que compilar el programa para generar el ejecutable. Para ello deberá seleccionar el menú “Ejecución” y a continuación “Compilar” (ver Figura 7).

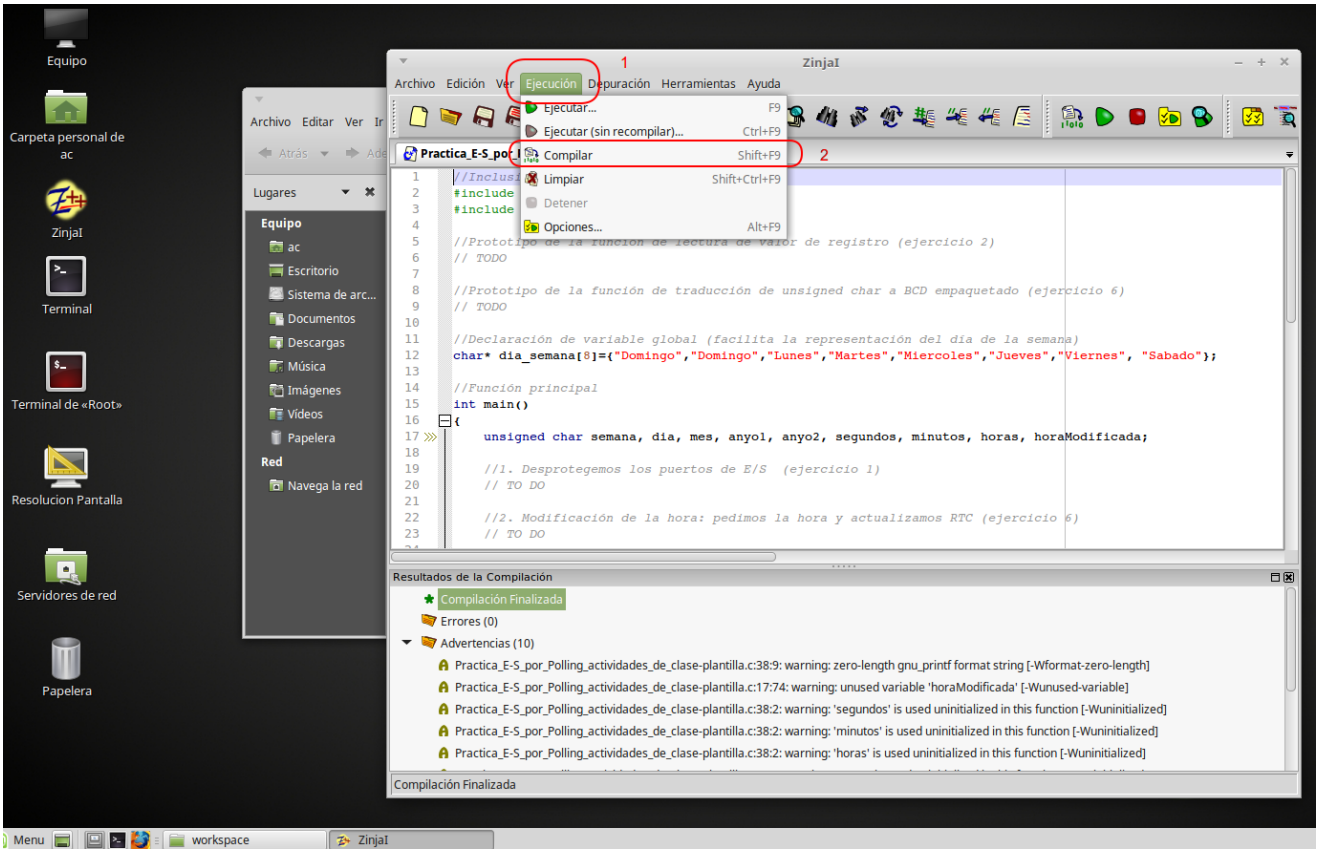


Figura 7. Generación del archivo ejecutable. Paso 1: selección del menú “Ejecución”. Paso 2: selección “Compilar”.

Una vez generado el programa ejecutable se podrá comprobar cómo en la carpeta de trabajo se ha creado un nuevo archivo binario con la extensión .bin. Para poder probar el programa habrá que ejecutarlo con privilegios de usuario. Tal y como se ha explicado anteriormente existen varias formas de hacer esto. La primera es abrir en la propia carpeta donde está el archivo ejecutable una consola o terminal de comandos con el menú contextual (botón derecho del ratón). Ver Figura 8. A continuación para ejecutar el programa con privilegios de superusuario (“Root”) deberá teclear en el terminal “sudo ./nombre_archivo.bin” (donde nombre del archivo dependerá de cómo se llamará el archivo .c compilado). Se pedirá la contraseña, que en este caso es “ac” (ver Figura 9). Como

alternativa a estos dos pasos, tal y como se ha indicado anteriormente, también puede teclear el comando “su” en el terminal, introducir la contraseña “ac” y convertir la consola o terminal en un terminal de superusuario o “Root”.

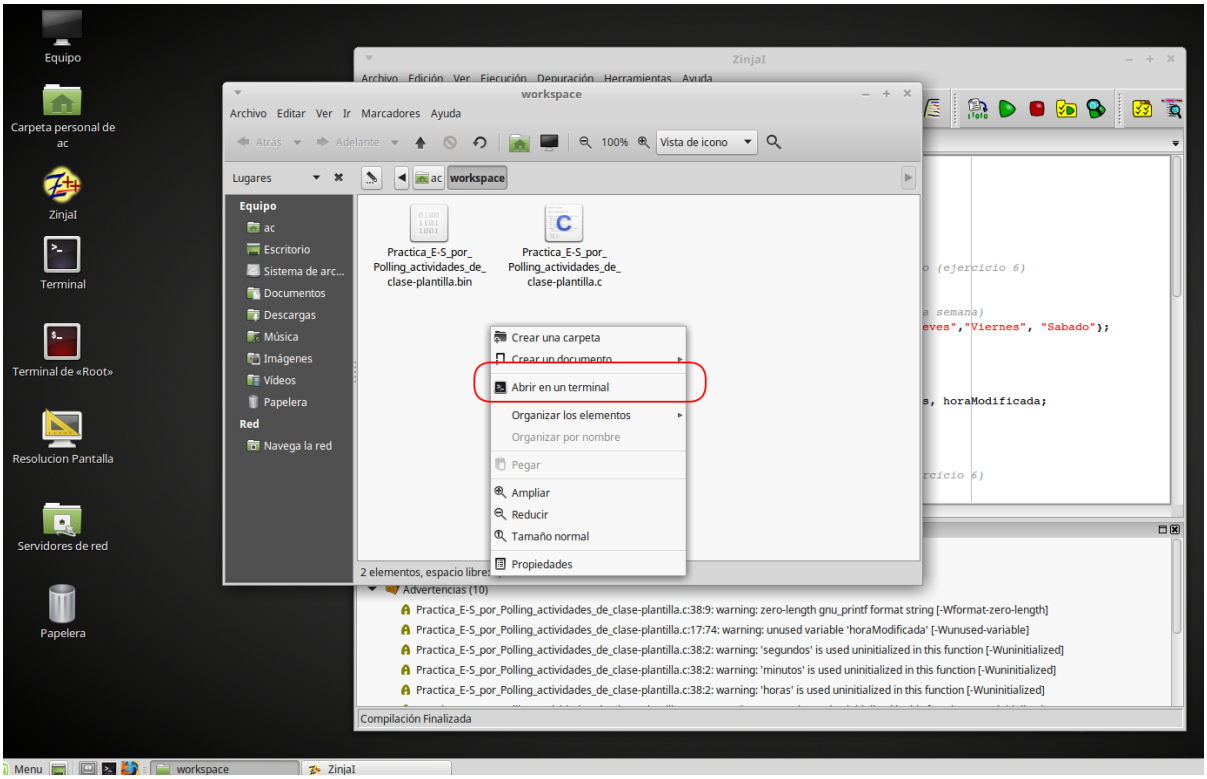


Figura 8. El archivo .bin aparece una vez compilado el programa con el entrono ZinJaI. Para ejecutarlo con privilegios de administrador habrá que abrir previamente una consola o terminal con el menú contextual (botón derecho).

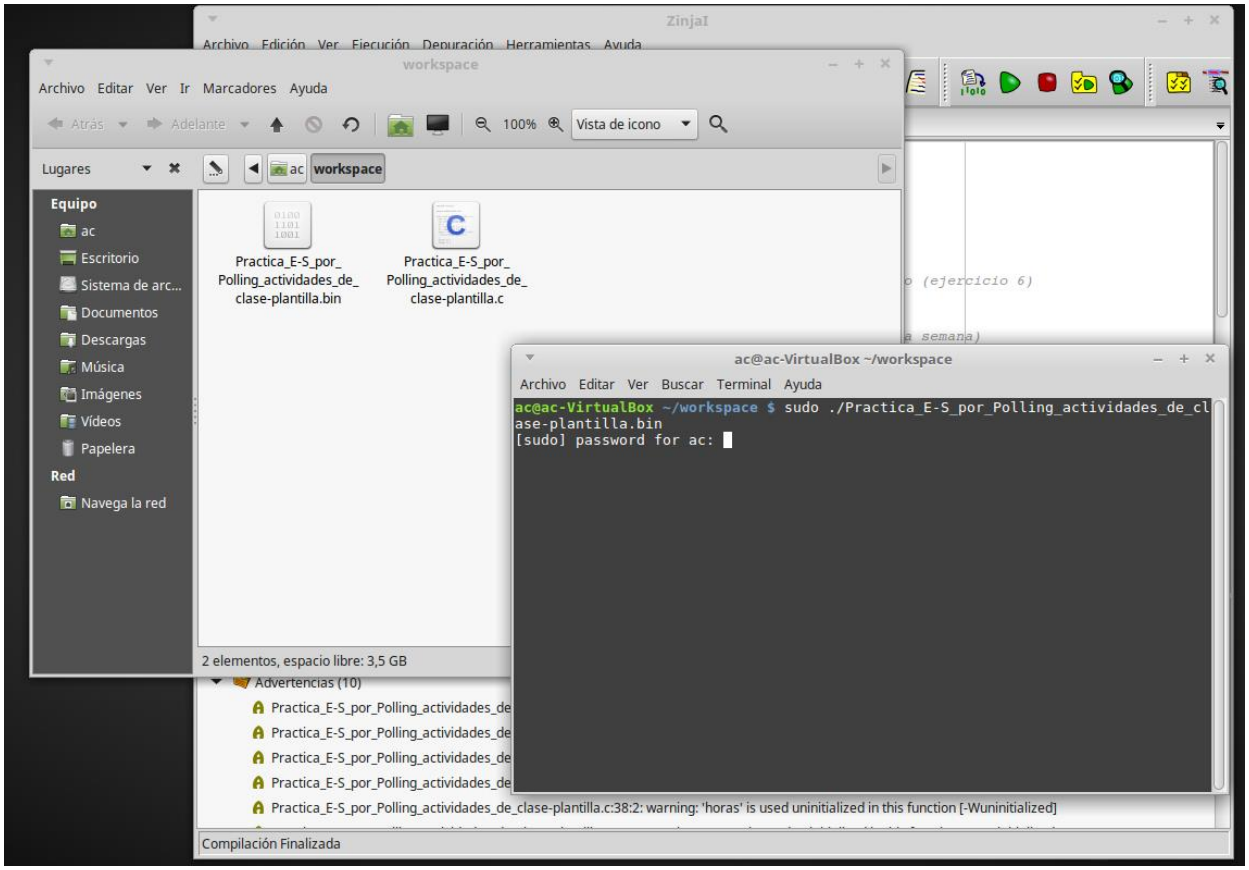


Figura 9. Ejecución del programa .bin con privilegios de superusuario (“Root”)

Otra forma alternativa para evitar tener que introducir la contraseña de superusuario siempre, es lanzar (ejecutar) un terminal de supersusuario o “Root”. Para ello a continuación habrá que posicionarse en el directorio (carpeta) donde está el archivo .bin ejecutable mediante el comando “cd /home/ac/workspace”. Finalmente, y ya sin tener que introducir la contraseña de superusuario o “Root”, teclear “./nombre_archivo.bin” (ver Figura 10).

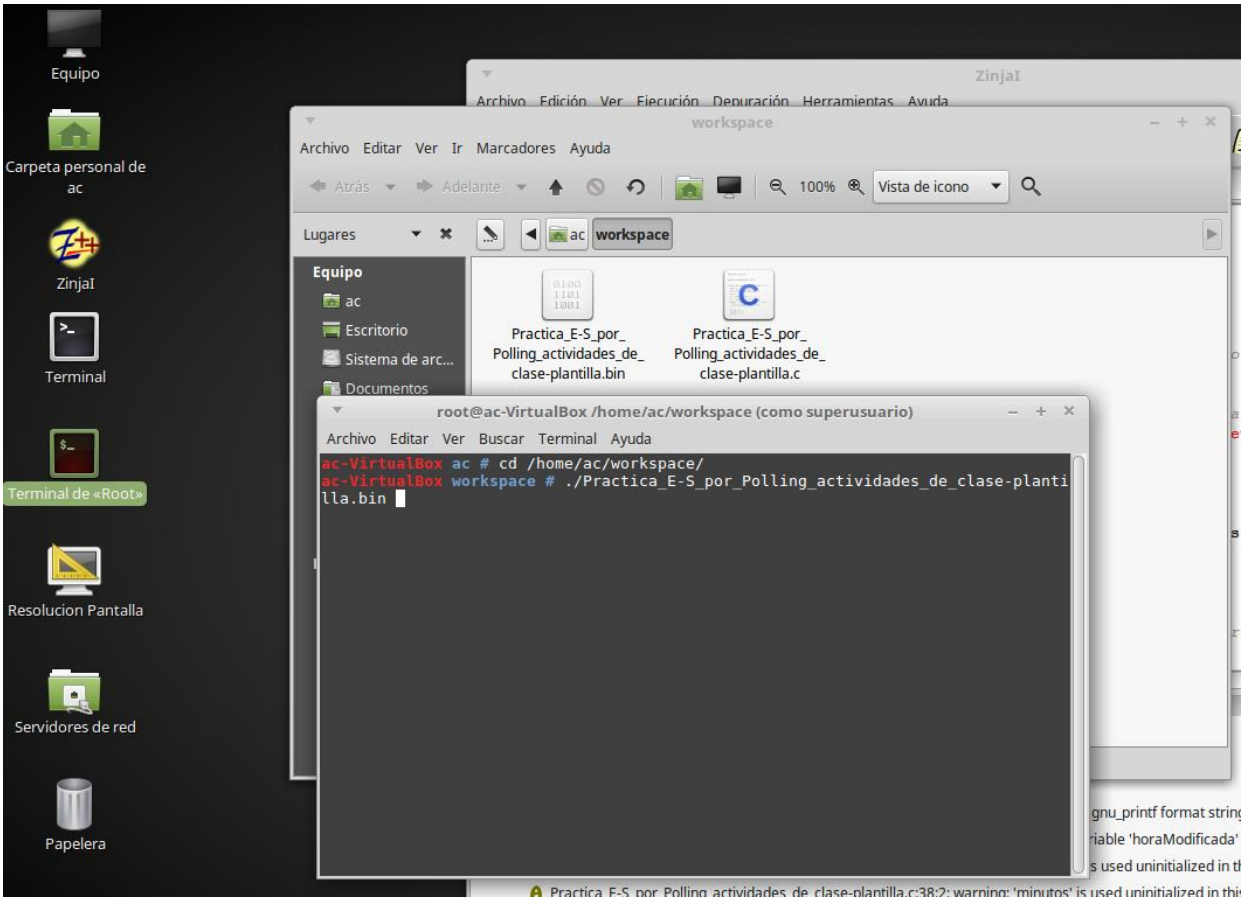


Figura 10. Ejecución de un programa con un terminal de “Root” o superusuario.

B. DRIVERS EN LINUX

Un aspecto destacable a tener en cuenta es que no se va a poder depurar el programa con el entorno ZinJaI mediante el establecimiento de puntos de ruptura (menú "Depuración"). Esto es debido a que el programa necesita ser ejecutado con privilegios de superusuario ya que debe ejecutarse en el espacio de *kernel* (núcleo del sistema operativo) y no en el espacio de usuario de Linux. Ejecutar un programa en el espacio de *kernel* es potencialmente peligroso ya que una instrucción incorrecta podría bloquear todo el sistema operativo. De ahí que en esta práctica se trabaje con una máquina virtual (además de que no se dispone de la contraseña de "Root" de los PCs de prácticas). En caso de bloqueo bastará con cerrar la máquina virtual y volver a restaurarla tal y como se explica al principio de este apartado.

Lo que se va a hacer en esta práctica de hecho es programar algo que podría equivaler a lo que realiza un *driver* o módulo en Linux. Los *drivers* o módulos en Linux son los que realmente acceden al módulo/controlador de E/S mediante el uso de técnicas de *polling* (encuesta), interrupciones o DMA. Estos módulos o *drivers* crean en el sistema de archivos de Linux unos ficheros en el directorio /dev para comunicarse con los programas de usuario. Los programas de usuario hacen uso de estos ficheros para poder leer y/o escribir datos (ver Figura 11).

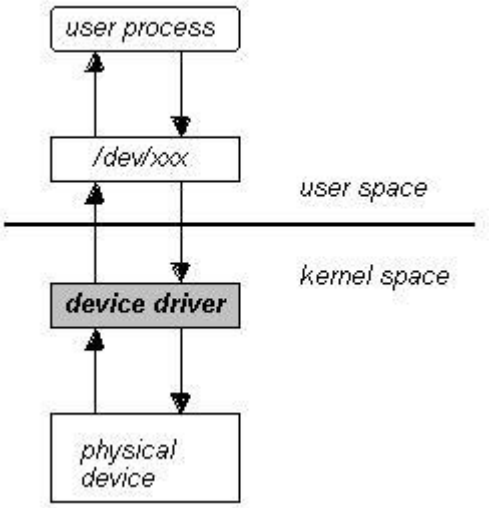


Figura 11. Esquema de acceso de un programa de usuario a un dispositivo físico/externo de E/S en Linux

Por ejemplo, en Linux hay un dispositivo virtual para obtener (leer) números aleatorios (/dev/random). En este caso se trata de dispositivo virtual que no tiene detrás un dispositivo hardware físico. Sin embargo, la forma de operar de un programa en C es idéntica a la que se usaría por ejemplo para leer un dato de un dispositivo externo de E/S. A continuación, se muestra un ejemplo del uso:

```
#include <stdio.h>

int main (int argc, char *argv[]) {

        unsigned int randval;
        FILE *f;

        f = fopen ("/dev/random", "r");              // Se abre el dispositivo "random"
        fread (&randval, sizeof (randval), 1, f);    // Se lee del dispositivo "random"
        fclose (f);                                  // Se cierra el dispositivo "random"

        printf ("%u\n", randval);                    // Se imprime el número leído


        return 0;
}
```

En un sistema operativo Linux el RTC se puede acceder a través del fichero "/dev/rtc".

C. SISTEMA DE CONTROL DE E/S DEL RTC MEDIANTE POLLING.

La tarea que se va a realizar durante la práctica consiste fundamentalmente en acceder directamente al módulo de E/S o controlador de Reloj de Tiempo Real (RTC) de nuestra máquina para leer y/o modificar la hora interna del sistema. La forma habitual de acceder al RTC en Linux por parte de un programa de usuario es utilizar el driver o módulo a tal efecto y leer del fichero "/dev/rtc". Este dispositivo externo también se conoce como reloj BIOS o reloj CMOS

Como se ha visto en clase de teoría, existen tres mecanismos de comunicación con la E/S en un computador: E/S por *polling* (encuesta o sondeo), E/S por interrupciones y DMA.

El mecanismo de sincronización por *polling* en Linux se puede implementar leyendo o escribiendo directamente en puertos de E/S. Es por tanto más fácil de comprender y permite dar un primer paso en el manejo de dispositivos de E/S. Para poder utilizar el mecanismo de interrupciones o el DMA en Linux habría que crear o cargar un driver, algo que está fuera del ámbito de la asignatura. Así pues, en esta práctica se utilizará E/S por *polling*. Tal y como se dice en la bibliografía de la asignatura, este mecanismo se basa en que el procesador controla directamente el dispositivo interrogando constantemente el controlador de E/S para determinar cuando está disponible para poder actuar sobre él.

Reloj de Tiempo Real

En los antiguos PCs el RTC se implementaba mediante el chip de Motorola MC146818. Aunque en las placas base de los modernos PCs ya no se usa dicho chip, por motivos de compatibilidad, el *kernel* de Linux emula el comportamiento de este chip haciendo creer a los programas y drivers que dicho dispositivo existe físicamente. El RTC también es emulado dentro de la máquina virtual y dicho dispositivo toma inicialmente la hora y fecha de la máquina anfitriona.

El MC146818 incorpora un completo reloj con alarma, calendario, interrupción periódica programable, generador de onda cuadrada y 64 bytes libres de RAM estática de bajo consumo. Los primeros 10 bytes de esta RAM son empleados para gestionar la fecha y la hora y los 4 siguientes son registros (A, B, C y D); los 50 restantes quedan a disposición del usuario.

El siguiente cuadro refleja la estructura de la memoria del MC146818. Los primeros 10 bytes son empleados para la fecha y hora. De los 128 registros (posiciones de memoria) sólo se reseñan aquellas que van a ser útiles a lo largo de esta práctica. Cada posición alberga un byte.

Dirección (hex)	Contenido	Dirección (hex)	Contenido
00	Segundos del RTC	01	Segundo de la alarma
02	Minutos del RTC	03	Minuto de la alarma
04	Hora del RTC	05	Hora de la alarma
06	Día de la semana	07	Día del mes
08	Mes	09	Año (dos últimas cifras)
0A	Status A	32	Año (dos primeras cifras)

El MC146818 está diseñado para ser usando en un bus multiplexado. Esto quiere decir que se necesitan dos accesos al dispositivo para leer o escribir un valor en una celda de memoria de la RAM CMOS. El primer acceso es de escritura y se usa para decirle al MC146818 qué posición de la memoria CMOS queremos leer o escribir. El segundo acceso será de lectura o escritura, dependiendo del tipo de operación que queramos realizar con la memoria.

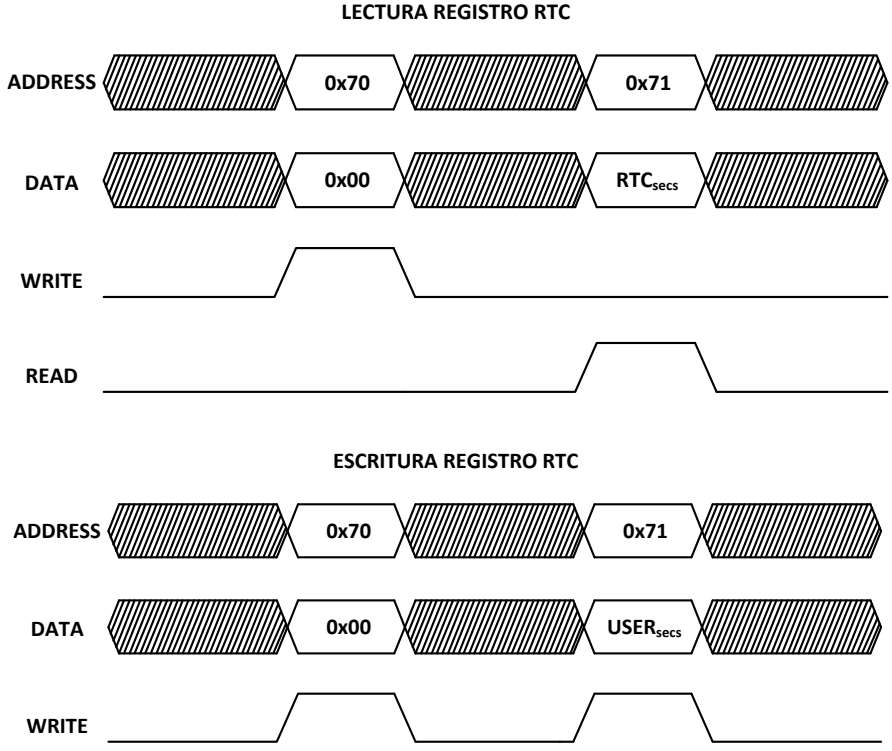
Para el primer acceso se usa el puerto de E/S 70h. El valor de 8 bits escrito en este puerto indica al dispositivo qué posición de la CMOS queremos usar.

A continuación, se usa el puerto de E/S 71h (lectura/escritura) para leer o escribir el byte cuya dirección se debe haber especificado en el paso anterior.

Las posiciones 00h a 09h, más la 32h contienen la información de la fecha y la hora. Estas posiciones sólo pueden consultarse si el bit 7 de la posición 0Ah (Status A) es 0. Este bit está a 1 durante un breve lapso de tiempo de forma periódica cada vez que se actualiza el valor de la hora (generalmente la actualización se produce cada segundo). La posición 0Ah (status) puede consultarse en cualquier momento.

El resto de los bits de "Status A" contienen la configuración del dispositivo. En esta práctica usaremos la configuración por defecto que está especificada en el arranque de Linux, con lo que no hará falta tocarla.
La información correspondiente a la fecha y hora está codificada en BCD empaquetado (dos dígitos BCD en cada byte).

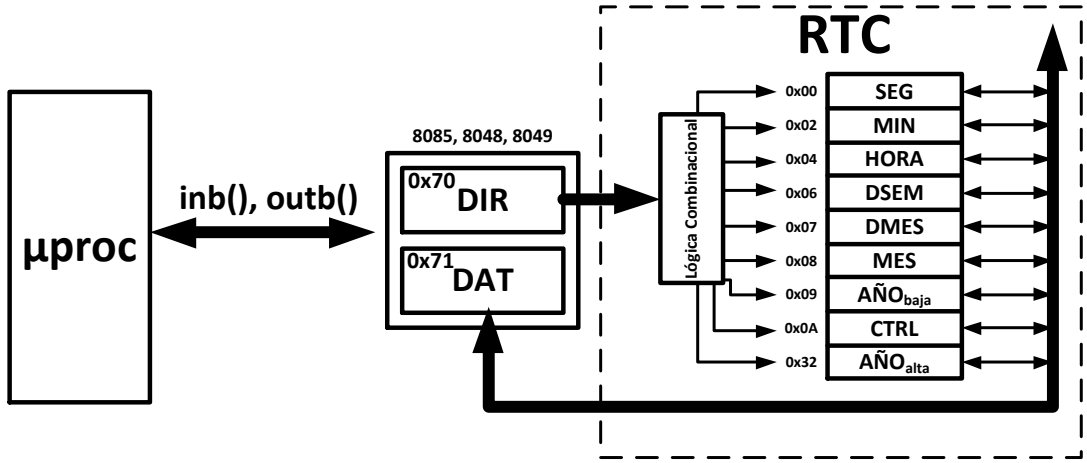
El campo "día de la semana" está codificado de tal forma que el lunes equivale al 1, el martes al 2, etc… hasta el domingo que puede estar codificado como 7 (si el sistema está configurado para que la semana comience en lunes) o como 0 (si el sistema está configurado para que la semana comience en domingo). A efectos prácticos esto significa que si leemos 0 como día de la semana, lo procesaremos como si hubiéramos leído un 7. De esta forma la secuencia de días de la semana es 1, 2, 3, 4, 5, 6, 7 independientemente de la configuración del sistema.



Para acceder a los puertos de E/S en Linux se emplean las funciones 'inb' y 'outb' disponibles en la librería io.h (situada en sys/io.h). El prototipo de tales funciones es el siguiente:

unsigned char inb(unsigned short int port);
void outb (unsigned char value, unsigned short int port);

La función 'outb' nos permite realizar un acceso en escritura al puerto de E/S, su primer parámetro indica el valor a escribir mediante un dato de 8 bits sin signo y el segundo parámetro establece el puerto de E/S en el que se escribe. Por otro lado, la función 'inb' nos permite realizar el acceso en lectura teniendo que especificar por parámetro el puerto de E/S en el que queremos leer, dicha función devolverá el dato leído como un número de 8 bits sin signo.



Resumiendo:

a) Lectura de valor del RTC: acceso de escritura al puerto 0x70, escribiendo el dato que queremos leer (tabla anterior); y, posteriormente, acceso por lectura al puerto 0x71.

b) Escritura de valor del RTC: acceso de escritura al puerto 0x70, escribiendo el dato que queremos escribir (tabla anterior); y, posteriormente, acceso por escritura al puerto 0x71 del valor que queremos aportar al RTC.

c) Hay que recordar que antes de leer y/o escribir en una posición de la memoria del RTC, hay que comprobar que no se está actualizando en dicho momento. Más aún, hay que acceder al RTC SOLO cuando acaba de ser actualizado, para poder tener el tiempo de un pulso de reloj completo para acceder y leer/escribir todos los valores.

ANEXO I: importación y ejecución de la máquina virtual con Oracle VirtualBox

Antes de importar la máquina virtual es conveniente que abra el programa Oracle VirtualBox para comprobar que la máquina no existiera de una sesión anterior. Si aparece la máquina virtual "AC-Linux_Mint_18.1_32_bits" deberá primero eliminarla seleccionándola con el menú contextual (botón derecho) y opción "Delete".
Una vez hecho esto se puede pasar a importar la máquina virtual. Esta operación se realiza en 3 pasos que se detallan en la Figura 12.

Figura 12. Importación máquina virtual en Oracle VirtualBox.

Seleccione la opción "Archivo" del menú principal y el submenú "Importar servicio virtualizado" (Paso 1). A continuación le aparecerá una ventana con el título de "Servicio a importar". Busque la ruta en su PC donde haya guardado el archivo .ova (Paso 2). Finalmente pulse el botón "Siguiente" y acepte o pulse siguiente en las siguientes pantallas (Paso 3).
Para ejecutar la máquina virtual deberá seleccionar la máquina virtual ya importada que le aparecerá en la parte izquierda de VirtualBox, y pulsar el botón "Iniciar" con el icono de una flecha verde. En la Figura 12 este botón aparece con el nombre de "Mostrar".
El modo de trabajo con la máquina virtual a continuación es análogo a lo que se describe a partir de la página 4 de este boletín de prácticas.