

Introduccion

Antes de comenzar revise los siguientes apendices y complete las instrucciones que se indican

Para una explicacion detallada sobre como instalar un ambiente EDK2 en una Raspberry Pi 3, consulte el Apendice A.

Si desea instalar la maquina virtual Qemu consulte el Apendice B

Para habilitar el ambiente EDK2 y compilar para x86_64 consulte el apendice C

Una vez que tenga instalado y funcional el ambiente para compilar en arquitectura x86_64, procedemos a mostrar como compilar el paquete de ejemplo *MdeModulePkg.dsc*, el cual contine una descripcion de los archivos requeridos para construir la aplicacion UEFI:

```
rodrigo@raspberrypi:~/src/edk2 $ cat MdeModulePkg/MdeModulePkg.dsc
## @file
# EFI/PI Reference Module Package for All Architectures
#
# (C) Copyright 2014 Hewlett-Packard Development Company, L.P.<BR>
# Copyright (c) 2007 - 2021, Intel Corporation. All rights reserved.<BR>
# Copyright (c) Microsoft Corporation.
# Copyright (C) 2024 Advanced Micro Devices, Inc. All rights reserved.<BR>
#
#   SPDX-License-Identifier: BSD-2-Clause-Patent
#
##

[Defines]
  PLATFORM_NAME                = MdeModule
  PLATFORM_GUID                = 587CE499-6CBE-43cd-94E2-186218569478
  PLATFORM_VERSION              = 0.98
  DSC_SPECIFICATION             = 0x00010005
  OUTPUT_DIRECTORY             = Build/MdeModule
  SUPPORTED_ARCHITECTURES      = IA32|X64|EBC|ARM|AARCH64|RISCV64|LOONGARCH64
  BUILD_TARGETS                 = DEBUG|RELEASE|NOOPT
  SKUID_IDENTIFIER              = DEFAULT

!include MdePkg/MdeLibs.dsc.inc

[LibraryClasses]
#
# Entry point
#
PeiCoreEntryPoint|MdePkg/Library/PeiCoreEntryPoint/PeiCoreEntryPoint.inf
PeimEntryPoint|MdePkg/Library/PeimEntryPoint/PeimEntryPoint.inf
DxeCoreEntryPoint|MdePkg/Library/DxeCoreEntryPoint/DxeCoreEntryPoint.inf
```

Para esto usaremos el script *build* que se genera por defecto con la instalación de la paqueteria EDK2, antes de comenzar a usar el script debemos asegurar que las variables de ambiente que usa el proyecto EDK2 esten configuradas correctamente, ejecutamos el script de configuración:

```
rodrigo@raspberrypi:~/src/edk2 $ . ./edksetup.sh
Loading previous configuration from /home/rodrigo/src/edk2/Conf/BuildEnv.sh
Using EDK2 in-source Basetools
WORKSPACE: /home/rodrigo/src/edk2
EDK_TOOLS_PATH: /home/rodrigo/src/edk2/BaseTools
CONF_PATH: /home/rodrigo/src/edk2/Conf
rodrigo@raspberrypi:~/src/edk2 $
```

El cual nos configura las variables a los directorios correctos, despues de eso utilizaremos el script *build* con los siguientes parametros:

```
rodrigo@raspberrypi:~/src/edk2 $ build -a X64 -t GCC5 -b DEBUG -p MdeModulePkg/MdeModulePkg.dsc
```

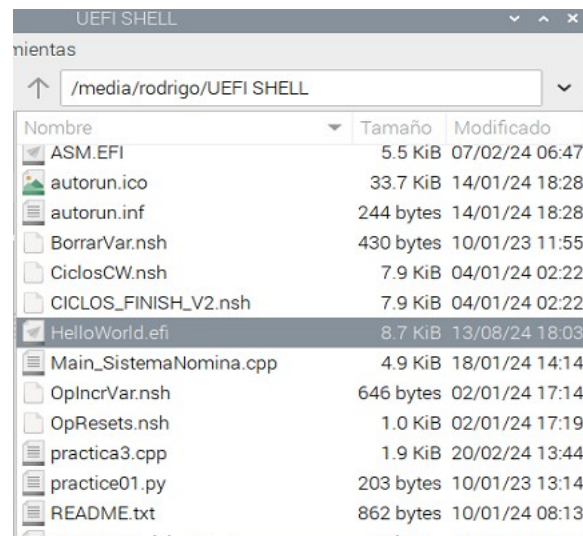
El parametro *-a X64* indica que tipo de arquitectura estamos "apuntando" compilar nuestra aplicacion (es decir al microprocesador que ejecutara la aplicacion), la opcion *-t GCC5* se refiere a usar el conjunto de herramientas de compilacion GCC5 (el anexo C describe cual compilador es el usado), y por ultimo la orden *-p MdeModulePkg/MdeModulePkg.dsc* indica la dirección del archivo de descripcion que contiene el listado de los archivos usados para la compilación de nuestra aplicación, cuando la compilacion termine enviara un mensaje de exito.

```
make: No se hace nada para 'tbuild'.
make: No se hace nada para 'tbuild'.
Building ... /home/rodrigo/src/edk2/MdeModulePkg/Universal/CapsulePei/CapsulePei.inf [X64]
Building ... /home/rodrigo/src/edk2/MdeModulePkg/Universal/Variable/MmVariablePei/MmVariablePei.inf [X64]
make: No se hace nada para 'tbuild'.
Building ... /home/rodrigo/src/edk2/MdeModulePkg/Bus/Pci/UfsPciHcPei/UfsPciHcPei.inf [X64]
make: No se hace nada para 'tbuild'.
make: No se hace nada para 'tbuild'.
make: No se hace nada para 'tbuild'.
- Done -
Build end time: 22:19:09, Aug.13 2024
Build total time: 00:01:12
rodrigo@raspberrypi:~/src/edk2 $
```

Ahora para usar nuestro paquete debemos de copiarlo o moverlo a una USB con sistema de archivo FAT32, el cual se encuentra en la direcccion siguiente:

```
rodrigo@raspberrypi:~/src/edk2/Build/MdeModule/DEBUG_GCC5/X64 $ pwd
/home/rodrigo/src/edk2/Build/MdeModule/DEBUG_GCC5/X64
rodrigo@raspberrypi:~/src/edk2/Build/MdeModule/DEBUG_GCC5/X64 $ ls HelloWorld.efi
HelloWorld.efi
rodrigo@raspberrypi:~/src/edk2/Build/MdeModule/DEBUG_GCC5/X64 $
```

El nombre del paquete construido fue *HelloWorld.efi*. Lo copiamos a nuestra USB:



Y ejecutamos nuestra maquina virtual con acceso a la unidad USB, para esto necesitamos averiguar el puerto en el que se encuentra nuestra usb:

```

rodrigo@raspberrypi:~/src/edk2 $ lsusb
Bus 001 Device 004: ID ffff:5678 Blackpcs
Bus 001 Device 003: ID 0424:ec00 Microchip Technology, Inc. (formerly SMSC) SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Microchip Technology, Inc. (formerly SMSC) SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
rodrigo@raspberrypi:~/src/edk2 $

```

En este caso es el puerto 4, tomamos esta informacion y la agregamos a los parametros de nuestra llamada a Qemu:

```

rodrigo@raspberrypi:~/src/edk2 $ lsusb
Bus 001 Device 004: ID ffff:5678 Blackpcs
Bus 001 Device 003: ID 0424:ec00 Microchip Technology, Inc. (formerly SMSC) SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Microchip Technology, Inc. (formerly SMSC) SMC9514 Hub
rodrigo@raspberrypi:~/src/edk2 $ sudo qemu-system-x86_64 -bios OVMF.fd -m 256M -net none -nographic -usb -device usb-ehci,id=ehci -device usb-host,hostbus=1,hostaddr=4

```

Observe como el numero de Bus y el numero de direccion de puerto coinciden, una vez iniciada la sesion virtual del UEFI-Shell:

```

UEFI Interactive Shell v2.2
edk2-stable202311 (https://github.com/pbatard/UEFI-Shell)
UEFI v2.70 (EDK II, 0x00010000)
Mapping table
  FS0: Alias(s):HD1a0b:;BLK2:
        PciRoot(0x0)/Pci(0x3,0x0)/USB(0x0,0x0)/HD(1,MBR,0x0674A67F,0x800,0x3A97800)
  BLK0: Alias(s):
        PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
  BLK1: Alias(s):
        PciRoot(0x0)/Pci(0x3,0x0)/USB(0x0,0x0)
Press ESC in 4 seconds to skip startup.nsh or any other key to continue.
Shell> fs0: (A)
FS0:\> HelloWorld.efi (B)
UEFI Hello World! (C)
FS0:\>

```

Accedemos a la memoria USB con `fs0:` indicado en (A), escribimos el nombre de nuestra aplicacion `HelloWorld.efi` (B) y observamos el resultado en (C) donde se muestra el mensaje UEFI Hello World!.

APENDICE A

Como instalar un ambiente de diseño usando el proyecto EDK2

Comenzamos creando un directorio de trabajo, para este ejemplo lo llamaremos EDK2

```

rodrigo@raspberrypi:~$ ls
coreboot  Desktop  Downloads  edk2  edk2-docker  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls  ZWORKSPACE
Bookshelf  Desktop  Downloads  edk2  edk2-docker  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls  ZWORKSPACE
rodrigo@raspberrypi:~$ sudo mkdir EDK2
rodrigo@raspberrypi:~$ ls
coreboot  Desktop  Downloads  EDK2  edk2-docker  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls  ZWORKSPACE
Bookshelf  Desktop  Downloads  EDK2  edk2-docker  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls  ZWORKSPACE
rodrigo@raspberrypi:~$ cd EDK2/
rodrigo@raspberrypi:~/EDK2$ ls
rodrigo@raspberrypi:~/EDK2$

```

Dentro clonamos el repositorio de Git Hub, con las siguientes instrucciones:

```
sudo git clone --recurse -submodules https://github.com/tianocore/edk2.git
```

Si presenta problemas de conexión use los comandos separados:

```
git clone https://github.com/tianocore/edk2.git
```

```
git submodule update --init #si ocurre un problema solo siga las instrucciones en pantalla
```

```
raspberrypi - rodrigo@raspberrypi: ~/EDK2 VT
File Edit Setup Control Window Help
rodrigo@raspberrypi: ~/EDK2 $ sudo git clone https://github.com/tianocore/edk2.git
```

Dependiendo de su conexión a Internet esta operación tardará algún tiempo en completarse

```
rodrigo@raspberrypi: ~/EDK2 $ sudo git clone https://github.com/tianocore/edk2.git
Cloning into 'edk2'...
remote: Enumerating objects: 391742, done.
remote: Counting objects: 100% (214/214), done.
remote: Compressing objects: 100% (130/130), done.
Receiving objects: 95% (372304/391742), 293.11 MiB | 1.65 MiB/s
```

Nota: en caso de algún problema siga las instrucciones que le indican

```
File Edit Setup Control Window Help
rodrigo@raspberrypi: ~/EDK2/edk2 $ git submodule update --init
fatal: detected dubious ownership in repository at '/home/rodrigo/EDK2/edk2'
To add an exception for this directory, call:

    git config --global --add safe.directory /home/rodrigo/EDK2/edk2
rodrigo@raspberrypi: ~/EDK2/edk2 $ git config --global --add safe.directory /home/rodrigo/EDK2/edk2
rodrigo@raspberrypi: ~/EDK2/edk2 $ git submodule update --init
error: could not lock config file .git/config: Permission denied
error: could not lock config file .git/config: Permission denied
fatal: Failed to register url for submodule path 'ArmPkg/Library/ArmSoftFloatLib/berkeley-softfloat-3'
rodrigo@raspberrypi: ~/EDK2/edk2 $ sudo git submodule update --init
```

Esta operación tomará cierto tiempo en completarse si falla en algún punto simplemente re-ejecute el comando.

```
rodrigo@raspberrypi: ~/EDK2/edk2 $ sudo git submodule update --init
Submodule 'SoftFloat' (https://github.com/uec-bar/berkeley-softfloat-3.git) registered for path 'ArmPkg/Library/ArmSoftFloatLib/berkeley-softfloat-3'
Submodule 'BaseTools/Source/C/BrotliCompress/brotli' (https://github.com/google/brotli) registered for path 'BaseTools/Source/C/BrotliCompress/brotli'
Submodule 'CryptoPkg/Library/MbedtlsLib/mbedtls' (https://github.com/MbedTLS/mbedtls) registered for path 'CryptoPkg/Library/MbedtlsLib/mbedtls'
Submodule 'CryptoPkg/Library/OpenSSL/lib/openssl' (https://github.com/openssl/openssl) registered for path 'CryptoPkg/Library/OpenSSL/lib/openssl'
Submodule 'MdeModulePkg/Library/BrotliCustomDecompressLib/brotli' (https://github.com/google/brotli) registered for path 'MdeModulePkg/Library/BrotliCustomDecompressLib/brotli'
Submodule 'MdeModulePkg/Universal/RegularExpressionDxe/oniguruma' (https://github.com/kkos/oniguruma) registered for path 'MdeModulePkg/Universal/RegularExpressionDxe/oniguruma'
Submodule 'MdePkg/Library/BaseFdtLib/libfdt' (https://github.com/device-tree-org/libfdt.git) registered for path 'MdePkg/Library/BaseFdtLib/libfdt'
Submodule 'MdePkg/Library/NipisysLib/nipisys' (https://github.com/NIPISYS/public-nipisys-1.git) registered for path 'MdePkg/Library/NipisysLib/nipisys'
Submodule 'RedfishPkg/Library/IsonLib/ison' (https://github.com/siberson/ison) registered for path 'RedfishPkg/Library/IsonLib/ison'
Submodule 'SecurityPkg/DeviceSecurity/SpdLib/libspdn' (https://github.com/DMTF/libspdn.git) registered for path 'SecurityPkg/DeviceSecurity/SpdLib/libspdn'
Submodule 'UnitTestFrameworkPkg/Library/Chockalib/chocka' (https://github.com/tianocore/edk2-chocka.git) registered for path 'UnitTestFrameworkPkg/Library/Chockalib/chocka'
Submodule 'UnitTestFrameworkPkg/Library/GoogleTestLib/googletest' (https://github.com/google/googletest.git) registered for path 'UnitTestFrameworkPkg/Library/GoogleTestLib/googletest'
Submodule 'UnitTestFrameworkPkg/Library/SubhookLib/subhook' (https://github.com/zeex/subhook.git) registered for path 'UnitTestFrameworkPkg/Library/SubhookLib/subhook'
Cloning into '/home/rodrigo/EDK2/edk2/ArmPkg/Library/ArmSoftFloatLib/berkeley-softfloat-3'...
Cloning into '/home/rodrigo/EDK2/edk2/BaseTools/Source/C/BrotliCompress/brotli'...
```

Tras completar ejecute el segundo comando para actualizar el repositorio, será necesario otorgar permisos completos a todos los grupos a la carpeta recién creada (edk2):

`sudo chmod -R 777 /home/rodrigo/EDK2/edk2/`

```
rodrigo@raspberrypi: ~/EDK2 $ ls -l
total 4
drwxr-xr-x 36 root root 4096 Jul 12 14:13 edk2
rodrigo@raspberrypi: ~/EDK2 $ sudo chmod -R 777 /home/rodrigo/EDK2/edk2/
rodrigo@raspberrypi: ~/EDK2 $ ls -l
total 4
drwxr-xr-x 36 root root 4096 Jul 12 14:13 edk2
rodrigo@raspberrypi: ~/EDK2 $
```

Esto permitirá que los diferentes scripts puedan terminar la configuración sin restricción alguna. Ingrese al directorio que se generó tras clonar el repositorio:

`cd ./edk2`

```
File Edit Setup Control Window Help
rodrigo@raspberrypi: ~/EDK2 $ cd edk2/
rodrigo@raspberrypi: ~/EDK2/edk2 $ ls
ArmPkg      BaseTools  CryptoPkg  edksetup.sh  FatPkg      IntelFsp2Pkg  Maintainers.txt  NetworkPkg  pip-requirements.txt  RedfishPkg  SignedCapsulePkg  UefiCpuPkg
ArmPlatformPkg  Conf       EmbeddedPkg  EmbeddedPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg      EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg
CONTRIBUTING.md  edksetup.bat  EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg      EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg  EmulatorPkg
rodrigo@raspberrypi: ~/EDK2/edk2 $
```

Y ejecute el comando (observe que la carpeta está llena de los archivos del proyecto):

`sudo ./edksetup.sh`

```
rod@rod-raspberrypi: /HEIM2/edk2 $ sudo ./edksetup.sh
Using EDK2 in-source BaseTools
WORKSPACE: /home/rod/edk2/EDK2/edk2
EDK_TOOLS_PATH: /home/rod/edk2/EDK2/edk2/BaseTools
CONF_PATH: /home/rod/edk2/EDK2/edk2/Conf
Copying SEDK_TOOLS_PATH/conf/build_rule.template
to /home/rod/edk2/EDK2/edk2/conf/build_rule.txt
Copying SEDK_TOOLS_PATH/conf/tools_def.template
to /home/rod/edk2/EDK2/edk2/conf/tools_def.txt
Copying SEDK_TOOLS_PATH/conf/target.template
to /home/rod/edk2/EDK2/edk2/conf/target.txt
rod@rod-raspberrypi: /HEIM2/edk2 $
```

Para asegurar que la configuración de los directorios de trabajo es la correcta, confirmada la configuración ejecute el comando siguiente para crear las herramientas básicas de compilación:

```
sudo make -C BaseTools/
```

[illegible]

La operacion tomara tiempo en completarse

```
File Edit Setup Control Window Help
test_build__init__(CheckPythonSyntax.Tests) ... ok
test_build_build(CheckPythonSyntax.Tests) ... ok
test_build_buildoptions(CheckPythonSyntax.Tests) ... ok
test_sitecustomize(CheckPythonSyntax.Tests) ... ok
test_tests.Split test_split(CheckPythonSyntax.Tests) ... ok
test32bitUnicodeCharInUtf8Comment(CheckUnicodeSourceFiles.Tests) ... ok
test32bitUnicodeCharInUtf8File(CheckUnicodeSourceFiles.Tests) ... ok
testSupplementaryPairInUnicodeCharInUtf16File(CheckUnicodeSourceFiles.Tests) ... ok
testSurrogatePairInUnicodeCharInUtf16File(CheckUnicodeSourceFiles.Tests) ... ok
testSurrogatePairInUnicodeCharInUtf8File(CheckUnicodeSourceFiles.Tests) ... ok
testSurrogatePairInUnicodeCharInUtf8FileWithBom(CheckUnicodeSourceFiles.Tests) ... ok
testUtf16InUtf8File(CheckUnicodeSourceFiles.Tests) ... ok
testValidUtf8File(CheckUnicodeSourceFiles.Tests) ... ok
testValidUtf8FileWithBom(CheckUnicodeSourceFiles.Tests) ... ok

-----
Ran 303 tests in 8.753s

OK
make[1]: Leaving directory '/home/rodrigo/EDK2/edk2/BaseTools/Tests'
make: Leaving directory '/home/rodrigo/EDK2/edk2/BaseTools'
rodrigo@raspberrpi:~/EDK2/edk2 $
```

Observe que no haya habido errores durante la ejecución, encaso de existir vuelva e ejecutar el comando y preste atención a los mensajes de error.

Ahora realicemos los ajustes en el archivo de configuracion para indicar que tipo de archivo EDK2 UEFI de arranque queremos crear, utilice el comando:

```
sudo vi Conf/target.txt
```

```
rodrigo@raspberrypi:~/EDK2/edk2 $ sudo vi Conf/target.txt
```

En el archivo encuentre las variables que se muestran en la siguiente tabla:

ACTIVE_PLATFORM = ArmVirtPkg/ArmVirtQemu.dsc

TARGET = DEBUG

TARGET_ARCH = AARCH64

TOOL CHAIN TAG = GCC5

Modifique los valores en caso de ser necesario:

```
# Copyright (c) 2006 - 2019, Intel Corporation. All rights reserved.<BR>
# SPDX-License-Identifier: BSD-2-Clause-Patent
#
# ALL Paths are Relative to WORKSPACE
#
# Separate multiple LIST entries with a SINGLE SPACE character, do not use comma characters.
# Un-set an option by either commenting out the line, or not setting a value.
#
# PROPERTY      Type      Use      Description
# -----
# ACTIVE_PLATFORM  Filename  Recommended Specify the WORKSPACE relative Path and Filename
#                                     of the platform description file that will be used for the
#                                     build. This line is required if and only if the current
#                                     working directory does not contain one or more description
#                                     files.
ACTIVE_PLATFORM  = EmulatorPkg/EmulatorPkg.dsc ← (A)
"Conf/target.txt" [dos] 70L, 4828B
```

La figura en (A) muestra el valor de configuración de ACTIVE_PLATFORM, en este caso debe cambiarse al valor anteriormente indicado (en este caso comentamos la línea y sustituimos con una que tiene el valor requerido).

```
rodriago@raspberrypi:~/EDK2/edk2 $ cat Conf/target.txt
# Copyright (c) 2006 - 2019, Intel Corporation. All rights reserved.<BR>
# SPDX-License-Identifier: BSD-2-Clause-Patent
#
# ALL Paths are Relative to WORKSPACE
#
# Separate multiple LIST entries with a SINGLE SPACE character, do not use comma characters.
# Un-set an option by either commenting out the line, or not setting a value.
#
# PROPERTY      Type      Use      Description
# -----
# ACTIVE_PLATFORM  Filename  Recommended Specify the WORKSPACE relative Path and Filename
#                                     of the platform description file that will be used for the
#                                     build. This line is required if and only if the current
#                                     working directory does not contain one or more description
#                                     files.
#ACTIVE_PLATFORM  = EmulatorPkg/EmulatorPkg.dsc
ACTIVE_PLATFORM  = ArmVirtPkg/ArmVirtQemu.dsc
```

Tras realizar la misma acción en todas las líneas requeridas, volvemos a ejecutar el comando de configuración:

```
rodriago@raspberrypi:~/EDK2/edk2 $ ./edksetup.sh
Loading previous configuration from /home/rodriago/EDK2/edk2/Conf/BuildEnv.sh
Using EDK2 in-source BaseTools
WORKSPACE: /home/rodriago/EDK2/edk2
EDK_TOOLS_PATH: /home/rodriago/EDK2/edk2/BaseTools
CONF_PATH: /home/rodriago/EDK2/edk2/Conf
rodriago@raspberrypi:~/EDK2/edk2 $
```

Seguido de el comando "Build" (el cual es de python) para comenzar la creación de nuestro archivo de EDK2-BIOS para nuestra Máquina Virtual.

```
File Edit Setup Control Window Help
rodriago@raspberrypi:~/EDK2/edk2 $ build
Build environment: Linux-6.1.21-v8+-aarch64-with-glibc2.31
Build start time: 15:34:26, Jul.12 2024

WORKSPACE      = /home/rodriago/EDK2/edk2
EDK_TOOLS_PATH = /home/rodriago/EDK2/edk2/BaseTools
CONF_PATH      = /home/rodriago/EDK2/edk2/Conf
PYTHON_COMMAND = python3

Processing meta-data
Architecture(s) = AARCH64
Build target    = DEBUG
Toolchain       = GCC5

Active Platform = /home/rodriago/EDK2/edk2/ArmVirtPkg/ArmVirtQemu.dsc
.....[]
```

Comenzara el proceso de compilación de nuestro archivo indicado por la "barra de progreso", este proceso puede tardar un momento el cual dependera de las capacidades de computo de nuestro hardware.


```

rodrigo@raspberrypi:~$ sudo apt-get install qemu-system libvirt-daemon-system virt-manager
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
libvirt-daemon-system ya está en su versión más reciente (7.0.0-3+deb11u2).
qemu-system ya está en su versión más reciente (1:5.2+dfsg-11+deb11u3).
virt-manager ya está en su versión más reciente (1:3.2.0-3).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.

```

Los tres paquetes básicos son "*qemu-system*", "*libvirt-daemon-system*" y "*virt-manager*", así mismo si requiere instale el paquete de Qemu:

```

rodrigo@raspberrypi:~$ sudo apt-get install qemu
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
qemu ya está en su versión más reciente (1:5.2+dfsg-11+deb11u3).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.

```

Tras haber instalado los paquetes de alta su usuario del sistema para que pueda acceder a los servicios de virtualización:

```

rodrigo@raspberrypi:~$ sudo adduser rodrigo libvirt
El usuario 'rodrigo' ya es un miembro de 'libvirt'.

```

Sustituya su nombre de usuario en la línea de comando anterior, y para que los programas de segundo plano del software virtual se activen reinicie el sistema:

```

rodrigo@raspberrypi:~$ sudo reboot

```

Cuando el sistema vuelva a iniciar, compruebe que funciona con la siguiente línea de comandos:

```

rodrigo@raspberrypi:~$ sudo qemu-system-x86_64 -bios OVMF.fd -m 256M -net none -nographic

```

Esto inicializa una sesión virtual del UEFI-Shell:

```

PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell> help reset
Resets the system.

RESET [-u [string]]
RESET [-s [string]]
RESET [-c [string]]

-s - Performs a shutdown.
-u - Performs a warm boot.
-c - Performs a cold boot.
string - Describes a reason for the reset.

NOTES:
1. This command resets the system.
2. The default is to perform a cold reset unless the -u parameter is specified.
3. If a reset string is specified, it is passed into the Reset() function, and the system records the reason for the system reset.
Shell> reset -s
Reset with <null string> (0 bytes)
rodrigo@raspberrypi:~$

```

Para terminar la sesión use la orden "*reset -s*" en el prompt del UEFI-Shell y regresará al Prompt de la raspberry pi.

APENDICE C

Compilacion de modulos EDK2 para arquitectura X86_64 usando una raspberry pi

Tras haber completado las instrucciones del **Apéndice A**, y verificar que contamos con un ambiente de desarrollo funcional (al menos para Arquitectura AARCH64=ARM), debemos realizar unos ajustes adicionales a las herramientas del paquete EDK2 para poder compilar para arquitectura x86_64, lo primero es instalar la familia de compilacion GCC requerida por las herramientas:

```
Archivo  Editar  Pestañas  Ayuda
rodrigo@raspberrypi:~$ sudo apt-get install gcc-x86-64-linux-gnu
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
gcc-x86-64-linux-gnu ya está en su versión más reciente (4:10.2.1-1).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
rodrigo@raspberrypi:~$
```

Tras instalar el paquete del compilador, verificamos la ruta en la cual fue instalado el compilador:

```
rodrigo@raspberrypi:~$ which x86_64-linux-gnu-gcc
/usr/bin/x86_64-linux-gnu-gcc
rodrigo@raspberrypi:~$
```

Copiamos esta dirección para modificar el archivo de configuracion ubicado en la carpeta *Conf* de nuestro ambiente de desarrollo:

```
rodrigo@raspberrypi:~/src/edk2/Conf$ ls
BuildEnv.sh  build_rule.txt  ReadMe.txt  target.txt  tools_def.txt
rodrigo@raspberrypi:~/src/edk2/Conf$ sudo vim tools_def.txt
```

Usamos el editor de nuestra preferencia, (en este caso Vim con la opcion de numeros de linea):

```
85 DEFINE CYGWIN_BINX64      = c:/cygwin/opt/tiano/x86_64-pc-mingw64/x86_64-pc-mingw64/bin/
86
87 DEFINE GCC48_IA32_PREFIX  = ENV(GCC48_BIN)
88 DEFINE GCC48_X64_PREFIX  = ENV(GCC48_BIN)
89
90 DEFINE GCC49_IA32_PREFIX  = ENV(GCC49_BIN)
91 DEFINE GCC49_X64_PREFIX  = ENV(GCC49_BIN)
92
93 DEFINE GCCNOLTO_IA32_PREFIX = ENV(GCCNOLTO_BIN)
94 DEFINE GCCNOLTO_X64_PREFIX = ENV(GCCNOLTO_BIN)
95 #DEFINE GCC5_IA32_PREFIX   = ENV(GCC5_BIN)
96 DEFINE GCC5_IA32_PREFIX   = /usr/bin/x86_64-linux-gnu- (B)
97 #DEFINE GCC5_X64_PREFIX   = ENV(GCC5_BIN)
98 DEFINE GCC5_X64_PREFIX   = /usr/bin/x86_64-linux-gnu- (B)
99 DEFINE GCC_IA32_PREFIX    = ENV(GCC_BIN)
100 DEFINE GCC_X64_PREFIX     = ENV(GCC_BIN)
101 DEFINE GCC_HOST_PREFIX    = ENV(GCC_HOST_BIN)
102
103 DEFINE UNIX_IASL_BIN       = ENV(IASL_PREFIX)iasl
104 DEFINE WIN_IASL_BIN        = ENV(IASL_PREFIX)iasl.exe
105
106 DEFINE IASL_FLAGS          =
-- VISUAL --
```

Copie las líneas que se muestran en (A) y las "originales" las puede borrar o comentar, sustituya las nuevas líneas con la información mostrada (B), la cual es la dirección donde está instalado el compilador.

Cuando compile para arquitectura X64 o IA32, usando el script de python *build* ajuste el parametro -t

GCC5 (tool chain GCC5), ya que fue la variable de ambiente para el compilador que modificamos. Por ejemplo, compilar OVMF.fd (archivo de Qemu) usaremos:

```
rodrigo@raspberrypi:~/src/edk2 $ build -a X64 -t GCC5 -b DEBUG -p MdeModulePkg/MdeModulePkg.dsc
```

Tras termine el proceso de compilacion deberemos ver el mensaje de compilación terminada sin errores:

```
- Done -  
Build end time: 21:56:10, Aug.13 2024  
Build total time: 00:01:19  
rodrigo@raspberrypi:~/src/edk2 $
```