

## Introduccion

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

## 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
0  coreboot  Documents  edk2  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls
Bookshelf  Desktop  Downloads  edk2-docker  exor_nuv  ha_th  Music  Public  Templates  uefi-dev  Videos  ZIMMERSPACE
rodrigo@raspberrypi:~$ sudo mkdir EDK2
rodrigo@raspberrypi:~$ ls
0  coreboot  Documents  edk2  edk2-docker  exor_nuv  ha_th  Music  Public  Templates  uefi-dev  Videos  ZIMMERSPACE
Bookshelf  Desktop  Downloads  EDK2  edk2-platforms  file  LCD-show  Pictures  src  thincient_drives  UEFI-GPT-image-creator  Wls
rodrigo@raspberrypi:~$ cd EDK2/
rodrigo@raspberrypi:~/EDK2$ ls
rodrigo@raspberrypi:~/EDK2$
```

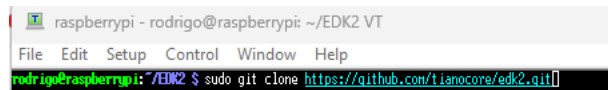
Dentro clonamos el repositorio de Git Hub, con las siguientes instrucciones:

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

Si presenta problemas de conexión use los comandos separados:

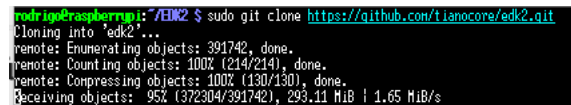
```
git clone https://github.com/tianocore/edk2.git
```

*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
rodri@raspberrypi:~/EDK2/edk2 $ git submodule update --init
fatal: detected dubious ownership in repository at '/home/rodri/EDK2/edk2'
To add an exception for this directory, call:

    git config --global --add safe.directory /home/rodri/EDK2/edk2
rodri@raspberrypi:~/EDK2/edk2 $ git config --global --add safe.directory /home/rodri/EDK2/edk2
rodri@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'
rodri@raspberrypi:~/EDK2/edk2 $ sudo git submodule update --init[]
```

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

```
rodri@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/HbedtlsLib/hbedtls' (https://github.com/hackebrot/hbedtls) registered for path 'CryptoPkg/Library/HbedtlsLib/hbedtls'
Submodule 'CryptoPkg/Library/OpensslLib/openssl' (https://github.com/openssl/openssl) registered for path 'CryptoPkg/Library/OpensslLib/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/devicetree-org/libfdt.git) registered for path 'MdePkg/Library/BaseFdtLib/libfdt'
Submodule 'MdePkg/Library/NipisysLib/nipisys' (https://github.com/NIP-Alliance/public-nipisys-1.git) registered for path 'MdePkg/Library/NipisysLib/nipisys'
Submodule 'RedfishPkg/Library/IsonLib/jansson' (https://github.com/bitiron/jansson) registered for path 'RedfishPkg/Library/IsonLib/jansson'
Submodule 'SecurityPkg/DeviceSecurity/SpdLib/libspdn' (https://github.com/Intel/libspdn.git) registered for path 'SecurityPkg/DeviceSecurity/SpdLib/libspdn'
Submodule 'UnitTestFrameworkPkg/Library/ChockaLib/chocka' (https://github.com/1ianocore/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/rodri/EDK2/edk2/ArmPkg/Library/ArmSoftFloatLib/berkeley-softfloat-3'...
Cloning into '/home/rodri/EDK2/edk2/BaseTools/Source/C/BrotliCompress/brotli'...
```

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

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

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

Esto permitira que los diferentes scripts puedan terminar la configuración sin reestriccion alguna. Ingrese al directorio que se genero tras clonar el repositorio:

`cd ./edk2`

```
File Edit Setup Control Window Help
rodri@raspberrypi:~/EDK2 $ cd edk2/
rodri@raspberrypi:~/EDK2/edk2 $ ls
ArmPkg      BaseTools  CryptoPkg  FatPkg      IntelFsp2Pkg  Maintainers.txt  NetworkPkg  PipRequirements.txt  RedfishPkg  SignedCapsulePkg  UefiCpuPkg
ArmPlatformPkg  Conf      DynamicLibrariesPkg  EmbeddedPkg  FirmwarePkg  LicenseHistory.txt  OemPkg      Pkg          SecurityPkg  SourceLevelDebugPkg  UefiPayloadPkg
ArmVirtPkg  CONTRIBUTING.md  edksetup.bat  EmulatorPkg  IntelFsp2Pkg  License.txt       MdePkg      PchChipsetPkg  ShellPkg    StandalonePkg      UnitTestFrameworkPkg
rodri@raspberrypi:~/EDK2/edk2 $
```

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

`sudo ./edksetup.sh`

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

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

`sudo make -C BaseTools/`

La operacion tomara tiempo en completarse

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.

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

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

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, 48288

```

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).

```

rodrigo@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:

```

rodrigo@raspberrypi:~/EDK2/edk2 $ ./edksetup.sh
Loading previous configuration from /home/rodrigo/EDK2/edk2/Conf/BuildEnv.sh
Using EDK2 in-source BaseTools
WORKSPACE: /home/rodrigo/EDK2/edk2
EDK_TOOLS_PATH: /home/rodrigo/EDK2/edk2/BaseTools
CONF_PATH: /home/rodrigo/EDK2/edk2/Conf
rodrigo@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
rodrigo@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/rodrigo/EDK2/edk2
EDK_TOOLS_PATH = /home/rodrigo/EDK2/edk2/BaseTools
CONF_PATH      = /home/rodrigo/EDK2/edk2/Conf
PYTHON_COMMAND = python3

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

Active Platform = /home/rodrigo/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 basicos son "*qemu-system*", "*libvirt-daemon-system*" y "*virt-manager*", asi 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 de alta su usuario del sistema para que pueda acceder a los servicios de virtualizacion:

```

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

```

Sustituya su nombre de ususario en la linea 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 linea de comandos:

```

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

```

Esto inicializara una sesion 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 sesion use la orden "*reset -s*" en el prompt del UEFI-Shell y regresera 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-
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 $
```