Como instalar un ambiente de diseño usando el proyecto EDK2

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

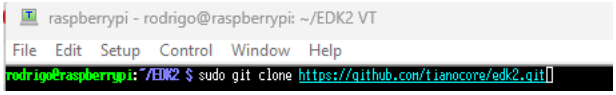
Dentro clonamos el repositorio de Git Hub, con las siguientes instrucciones:

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

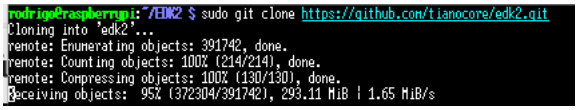
Si presenta problemas de conección use los comandos separados:

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

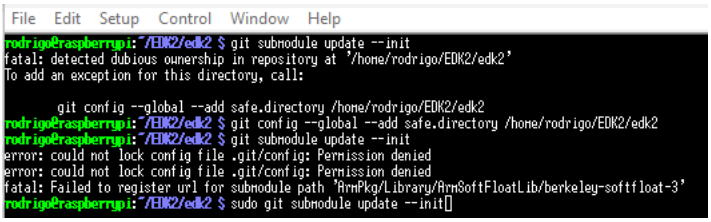
git submodule update --init #si ocurre un problema solo siga las instrucciones en pantalla



Dependiendo de su conección a Internet esta operacion tardara algún tiempo en completarse



Nota: en caso de algun problema siga las intrucciones que le indican



Esta operacion tomara 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/ucb-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/ARMmbed/mbedtls) registered for path 'CryptoPkg/Library/MbedTlsLib/mbedtls'
Submodule 'CryptoPkg/Library/Openssl1Lib/openssl' (https://github.com/openssl/openssl) registered for path 'CryptoPkg/Library/Openssl1Lib/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/pulibfdt.git) registered for path 'MdePkg/Library/BaseFdtLib/libfdt'
Submodule 'MdePkg/Library/MipiSysTLib/mipisyst' (https://github.com/MIPI-Alliance/public-mipi-sys-t.git) registered for path 'MdePkg/Library/MipiSysTLib/mipisyst'
Submodule 'RedfishPkg/Library/JsonLib/jansson' (https://github.com/akheron/jansson) registered for path 'RedfishPkg/Library/JsonLib/jansson'
Submodule 'SecurityPkg/DeviceSecurity/SpdmLib/libspdm' (https://github.com/DMTF/libspdm.git) registered for path 'SecurityPkg/DeviceSecurity/SpdmLib/libspdm'
Submodule 'UnitTestFrameworkPkg/Library/CmockaLib/cmocka' (https://github.com/tianocore/edk2-cmocka.git) registered for path 'UnitTestFrameworkPkg/Library/CmockaLib/cmocka'
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, sera necesario otorgar permisos completos a todos los grupos a la carpeta recien 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
drwxrwxrwx 36 root root 4096 Jul 12 14:13 edk2
rodrigo@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
rodrigo@raspberrypi:~/EDK2 $ cd edk2/
rodrigo@raspberrypi:~/EDK2/edk2 $ ls
ArmPkg              BaseTools           CryptoPkg           edksetup.sh  FatPkg          IntelFsp2WrapperPkg  Maintainers.txt  NetworkPkg      pip-requirements.txt  RedfishPkg  SignedCapsulePkg  UefiCpuPkg
ArmPlatformPkg      Conf                DynamicTablesPkg    EmbeddedPkg  FmpDevicePkg    License-History.txt  MdeModulePkg     OvmfPkg         PrmPkg                SecurityPkg  SourceLevelDebugPkg  UefiPayloadPkg
ArmVirtPkg          CONTRIBUTING.md     edksetup.bat        EmulatorPkg  IntelFsp2Pkg    License.txt          MdePkg           PcAtChipsetPkg  ReadMe.rst            ShellPkg     StandaloneMmPkg      UnitTestFrameworkPkg
rodrigo@raspberrypi:~/EDK2/edk2 $
```

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

sudo ./edksetup.sh

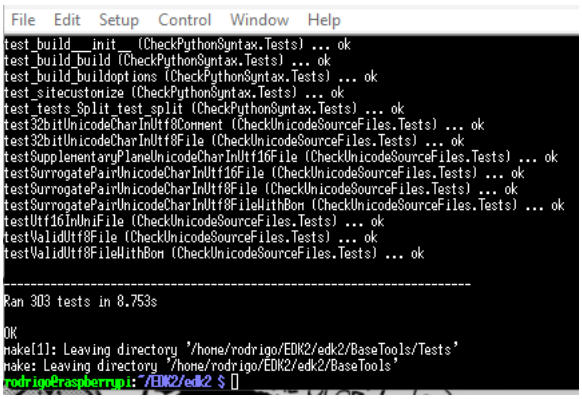
```
rodrigo@raspberrypi:~/EDK2/edk2 $ sudo ./edksetup.sh
Using EDK2 in-source Basetools
WORKSPACE: /home/rodrigo/EDK2/edk2
EDK_TOOLS_PATH: /home/rodrigo/EDK2/edk2/BaseTools
CONF_PATH: /home/rodrigo/EDK2/edk2/Conf
Copying $EDK_TOOLS_PATH/Conf/build_rule.template
     to /home/rodrigo/EDK2/edk2/Conf/build_rule.txt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

```
rodrigo@raspberrypi:~/EDK2/edk2 $ sudo make -C BaseTools/
make: Entering directory '/home/rodrigo/EDK2/edk2/BaseTools'
make -C Source/C
make[1]: Entering directory '/home/rodrigo/EDK2/edk2/BaseTools/Source/C'
Attempting to detect HOST_ARCH from 'uname -m': aarch64
Detected HOST_ARCH of AARCH64 using uname.
mkdir -p .
mkdir ./libs
make -C Common
make[2]: Entering directory '/home/rodrigo/EDK2/edk2/BaseTools/Source/C/Common'
gcc  -c  -I .. -I ../Include/Common -I ../Include/ -I ../Include/IndustryStandard -I ../Common/ -I .. -I . -I ../../../../MdePkg/Include/AArch64/ -I ../../../../MdePkg/Include -MD -fshort-uchar -fno-strict-aliasing -funapv -fno-delete-
null-pointer-checks -Wall -Werror -Wno-deprecated-declarations -Wno-stringop-truncation -Wno-restrict -Wno-unused-result -nostdlib -g -O2  BasePeCoff.c -o BasePeCoff.o
gcc  -c  -I .. -I ../Include/Common -I ../Include/ -I ../Include/IndustryStandard -I ../Common/ -I .. -I . -I ../../../../MdePkg/Include/AArch64/ -I ../../../../MdePkg/Include -MD -fshort-uchar -fno-strict-aliasing -funapv -fno-delete-
null-pointer-checks -Wall -Werror -Wno-deprecated-declarations -Wno-stringop-truncation -Wno-restrict -Wno-unused-result -nostdlib -g -O2  BinderFuncs.c -o BinderFuncs.o
gcc  -c  -I .. -I ../Include/Common -I ../Include/ -I ../Include/IndustryStandard -I ../Common/ -I .. -I . -I ../../../../MdePkg/Include/AArch64/ -I ../../../../MdePkg/Include -MD -fshort-uchar -fno-strict-aliasing -funapv -fno-delete-
null-pointer-checks -Wall -Werror -Wno-deprecated-declarations -Wno-stringop-truncation -Wno-restrict -Wno-unused-result -nostdlib -g -O2  CommonLib.c -o CommonLib.o
gcc  -c  -I .. -I ../Include/Common -I ../Include/ -I ../Include/IndustryStandard -I ../Common/ -I .. -I . -I ../../../../MdePkg/Include/AArch64/ -I ../../../../MdePkg/Include -MD -fshort-uchar -fno-strict-aliasing -funapv -fno-delete-
null-pointer-checks -Wall -Werror -Wno-deprecated-declarations -Wno-stringop-truncation -Wno-restrict -Wno-unused-result -nostdlib -g -O2  Crc32.c -o Crc32.o
gcc  -c  -I .. -I ../Include/Common -I ../Include/ -I ../Include/IndustryStandard -I ../Common/ -I .. -I . -I ../../../../MdePkg/Include/AArch64/ -I ../../../../MdePkg/Include -MD -fshort-uchar -fno-strict-aliasing -funapv -fno-delete-
null-pointer-checks -Wall -Werror -Wno-deprecated-declarations -Wno-stringop-truncation -Wno-restrict -Wno-unused-result -nostdlib -g -O2  Decompress.c -o Decompress.o
```

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.

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

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

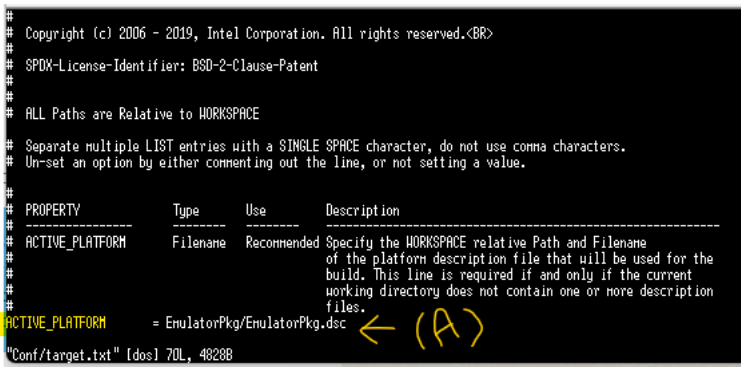
ACTIVE_PLATFORM = ArmVirtPkg/ArmVirtQemu.dsc

TARGET = DEBUG

TARGET_ARCH = AARCH64

TOOL_CHAIN_TAG = GCC5

Modifique los valores en caso de ser necesario:



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 linea 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 accion en todas las lineas requeridas, volvemos a ejecutar el comando de configuracion:

```
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 Maquina 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.

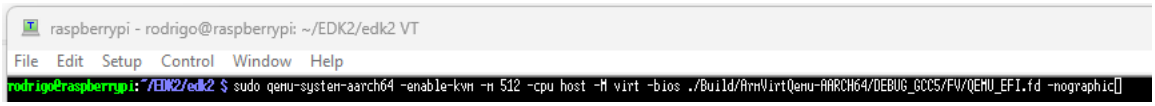
```
File  Edit  Setup  Control  Window  Help

Generate Region at Offset 0x40000
   Region Size = 0x40000
   Region Name = DATA

Generate Region at Offset 0x80000
   Region Size = 0x40000
   Region Name = None

GUID cross reference file can be found at /home/rodrigo/EDK2/edk2/Build/ArmVirtQemu-AARCH64/DEBUG_GCC5/FV/Guid.xref

FV Space Information
FVMAIN [99%Full] 6823680 (0x681f00) total, 6823656 (0x681ee8) used, 24 (0x18) free
FVMAIN_COMPACT [57%Full] 2093056 (0x1ff000) total, 1193184 (0x1234e0) used, 899872 (0xdbb20) free

- Done -
Build end time: 15:46:21, Jul.12 2024
Build total time: 00:11:54

rodrigo@raspberrypi:~/EDK2/edk2 $ ls Build/ArmVirtQemu-AARCH64/DEBUG_GCC5/FV/QEMU_EFI.fd
Build/ArmVirtQemu-AARCH64/DEBUG_GCC5/FV/QEMU_EFI.fd
rodrigo@raspberrypi:~/EDK2/edk2 $
```

Cuando la operación termine pude ubicar el archivo usando:

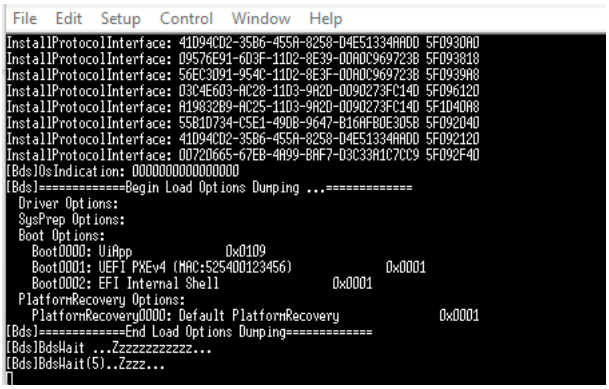
ls Build/ArmVirtQemu-AARCH64/DEBUG_GCC5/FV/QEMU_EFI.fd

Puede ejecutar el archivo en la maquina virtual de Qemu, con el siguiente comando:

sudo qemu-system-aarch64 -enable-kvm -m 512 -cpu host -M virt -bios ./Build/ArmVirtQemu-AARCH64/DEBUG_GCC5/FV/QEMU_EFI.fd -nographic



Cuando termine la ejecucion veremos la salida que nos mostraria el puerto de Debug de nuesta Maquina Virtual.



Hemos completado la creación de un IFWI-BIOS para nuestra maquina virtual