

**INSTITUTO POLITÉCNICO
NACIONAL**



**UNIDAD INTERDISCIPLINARIA EN INGENIERÍA Y
TECNOLOGÍAS AVANZADAS**

MECATRÓNICA

Sistemas Operativos en Tiempo Real

GRUPO: 3MV11

PROFESOR: Maza Casas Lamberto

ALUMNO: Navidad Huerta Alexandro

Agosto-Diciembre 2019

**Guía De Instalación Del Sistema Operativo En
Tiempo Real *MarteOS***

REQUISITOS PARA INSTALAR MARTEOS

Para poder hacer una instalación exitosa de este sistema operativo en tiempo real hay que tener en cuenta el año de la versión que vamos a instalar, esto por cualquier problema de compilación que pueda surgir al usar una versión distinta tanto del sistema operativo “base”, el cual usualmente es alguna versión de Linux en cualquiera de sus plataformas, ya sea Debian, Ubuntu, etc., como con la versión del SOTR que queremos instalar.

Para este caso tenemos que tanto MarteOS y GNAT (Que es un conocido compilador de lenguaje Ada, basado en la infraestructura de compilación de GCC) son versiones desarrolladas en el año 2017 y 2016 respectivamente. Por lo que es importante usar un sistema operativo base cuyo año de desarrollo sea 2016 o 2017, en otras palabras, un sistema operativo en el que estemos seguros que el compilador GNAT no tendrá ningún problema a la hora de instalarse. Esto con el fin de hacer una instalación más rápida y sencilla, no quiere decir que no pueda hacerse en un sistema operativo que haya sido desarrollado recientemente, sin embargo, para esto se debe tomar en cuenta lo que el compilador necesita para poder llevar a cabo una instalación exitosa.

Lo anterior se menciona ya que para versiones como Ubuntu 18.04, Ubuntu 19.04, Debian 10.0 Buster y Debian 9.9 se tuvo un error de compatibilidad al momento de instalar el compilador GNAT. En tanto que para la versión de Ubuntu 16.04, este problema es prácticamente inexistente. Esto se debe a que este sistema operativo se desarrolló en el año 2016 y no presenta ningún problema respecto a la compatibilidad ni con el compilador GNAT, ni con el SOTR.

Una vez que se tenga instalado el sistema operativo base (en este caso Ubuntu 16.04) ya sea como un sistema operativo nativo o como una máquina virtual, tenemos que contar con los siguientes archivos:

- gnat-gpl-2016-x86_64-linux-bin.tar.gz
- marte_2.0_22Feb2017_src.tar.gz

El primer archivo podemos encontrarlo en el siguiente enlace:

<http://mirrors.cdn.adacore.com/art/5739cefdc7a447658e0b016b>

Mientras que el segundo archivo podemos encontrarlo en el siguiente URL:

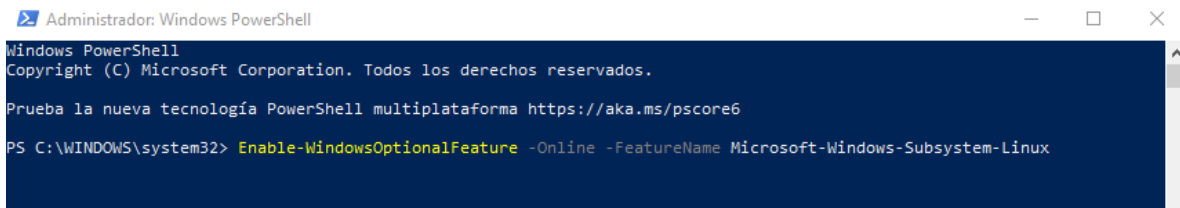
https://marte.unican.es/marte/marte_2.0_22Feb2017_src.tar.gz

El primer archivo como es de suponerse es el compilador GNAT, mientras que el segundo es el Sistema operativo en tiempo real.

Para el sistema operativo, primero es necesario seguir la guía de instalación de subsistemas para Windows, la cual se puede leer en el siguiente link:

<https://docs.microsoft.com/es-es/windows/wsl/install-win10?redirectedfrom=MSDN>

Como ahí lo menciona, necesitamos abrir PowerShell como administrador, copiar el texto de la página, pegarlo con click derecho y ejecutar.

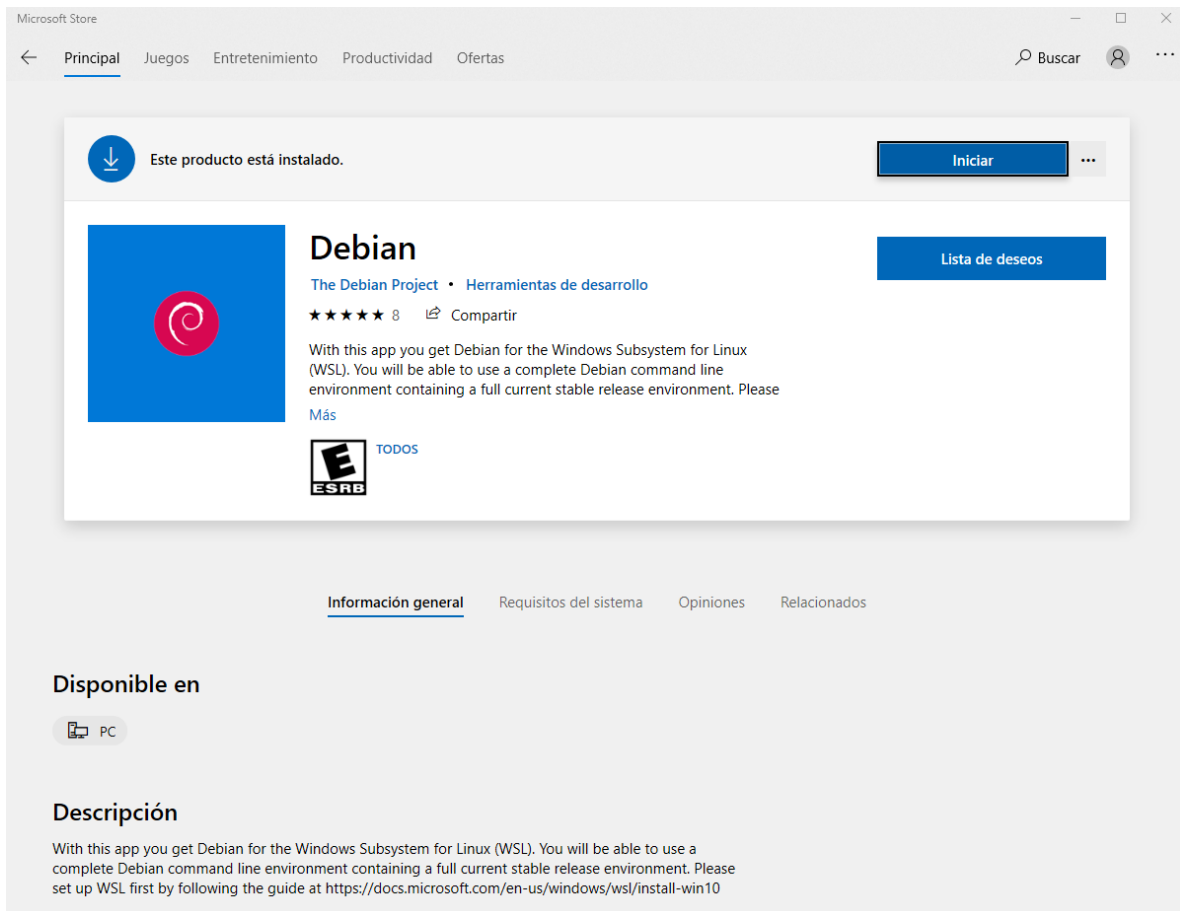


```
Administrador: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

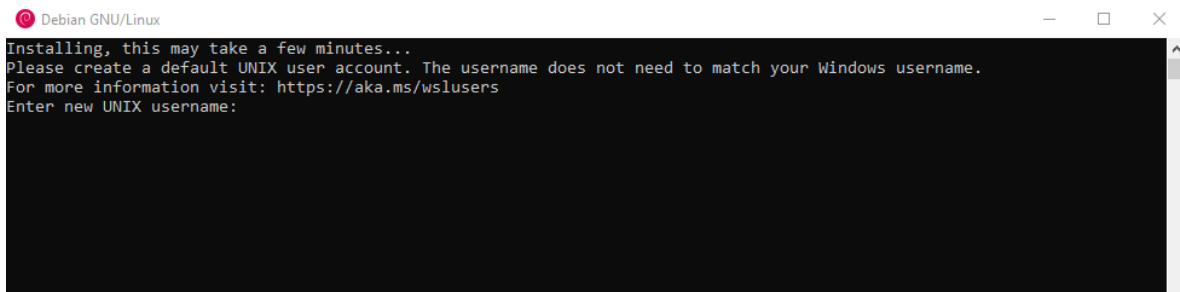
Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

Una vez que termina el proceso ya podemos descargar el subsistema de nuestra preferencia, en mi caso este será Debian, descargado directamente de la tienda oficial de Microsoft.

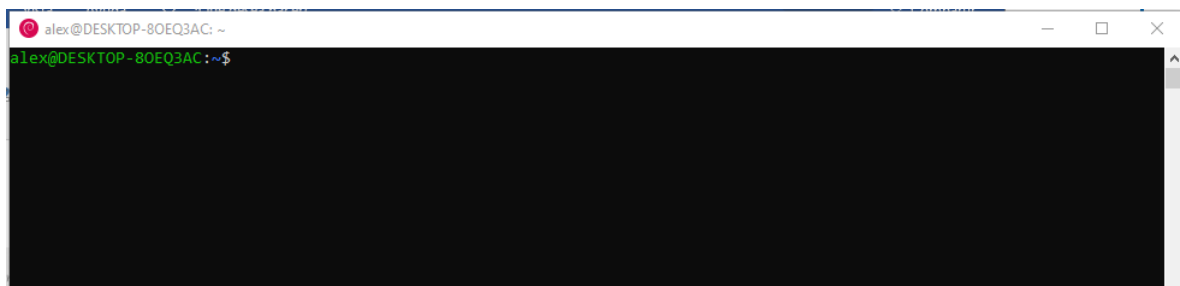


Ya que instalamos Debian, lo abrimos para registrar un usuario y la contraseña.



```
Debian GNU/Linux
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username:
```

Cuando se ve de la siguiente manera ya podemos continuar con la instalación del sistema operativo en tiempo real.



```
alex@DESKTOP-80EQ3AC: ~$
```

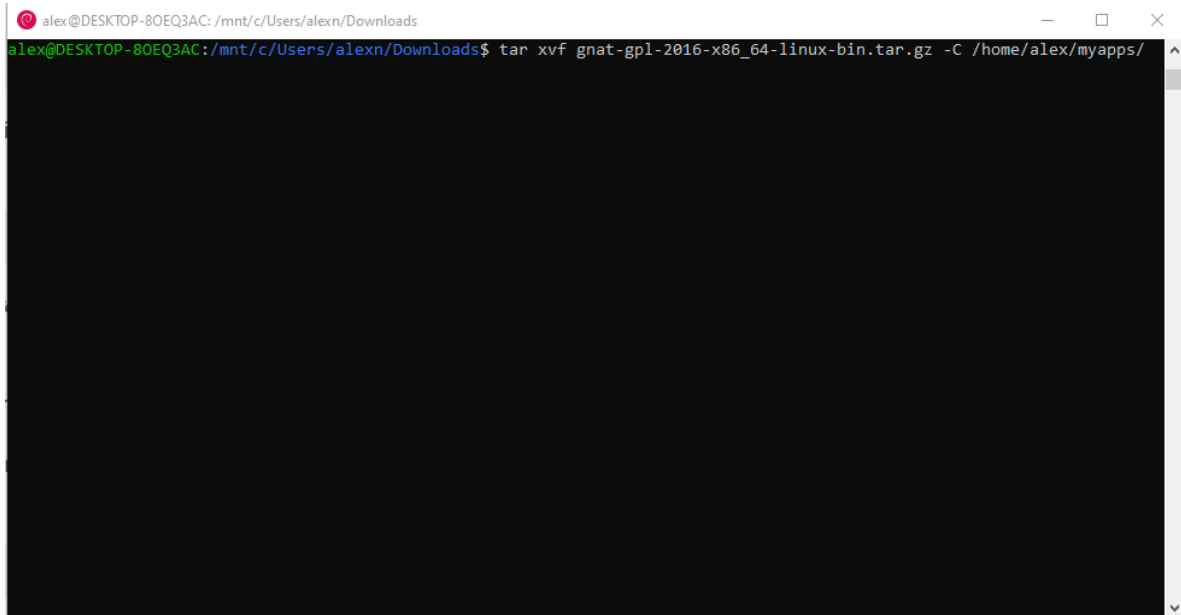
PREVIO A LA INSTALACIÓN

Una vez que se tiene instalado el sistema operativo base, lo primero que tenemos que hacer es crear una carpeta con el nombre “myapps”, y dentro de esta (mediante el comando `cd`) creamos una carpeta llamada “gnat”.



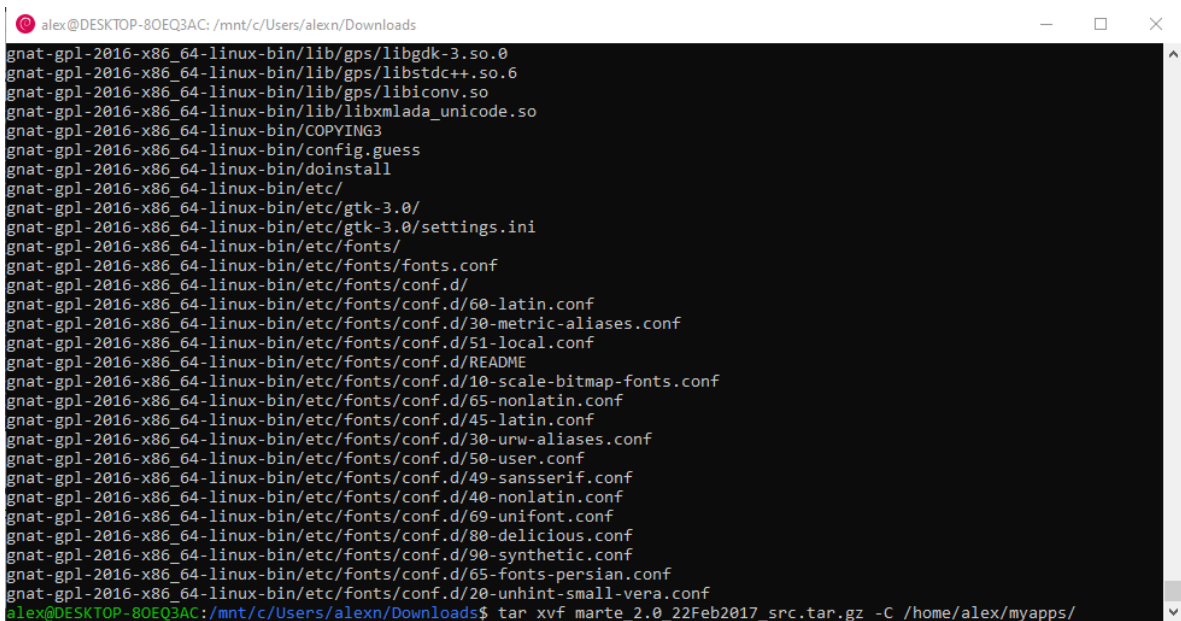
```
alex@DESKTOP-80EQ3AC: /home
alex@DESKTOP-80EQ3AC:~$ pwd
/home/alex
alex@DESKTOP-80EQ3AC:~$ mkdir myapps
alex@DESKTOP-80EQ3AC:~$ cd myapps/
alex@DESKTOP-80EQ3AC:~/myapps$ mkdir gnat
alex@DESKTOP-80EQ3AC:~/myapps$ cd
alex@DESKTOP-80EQ3AC:~$ cd ..
alex@DESKTOP-80EQ3AC:/home$
```

El siguiente paso es descomprimir los archivos que contienen el compilador GNAT así como el SOTR. Comenzando con el compilador GNAT, nos situamos en la ruta donde se descargaron los archivos comprimido, acto seguido ingresamos la siguiente instrucción “tar xvf gnat-gpl-2016-x86_64-linux-bin.tar.gz -C ruta_a_myapps”.



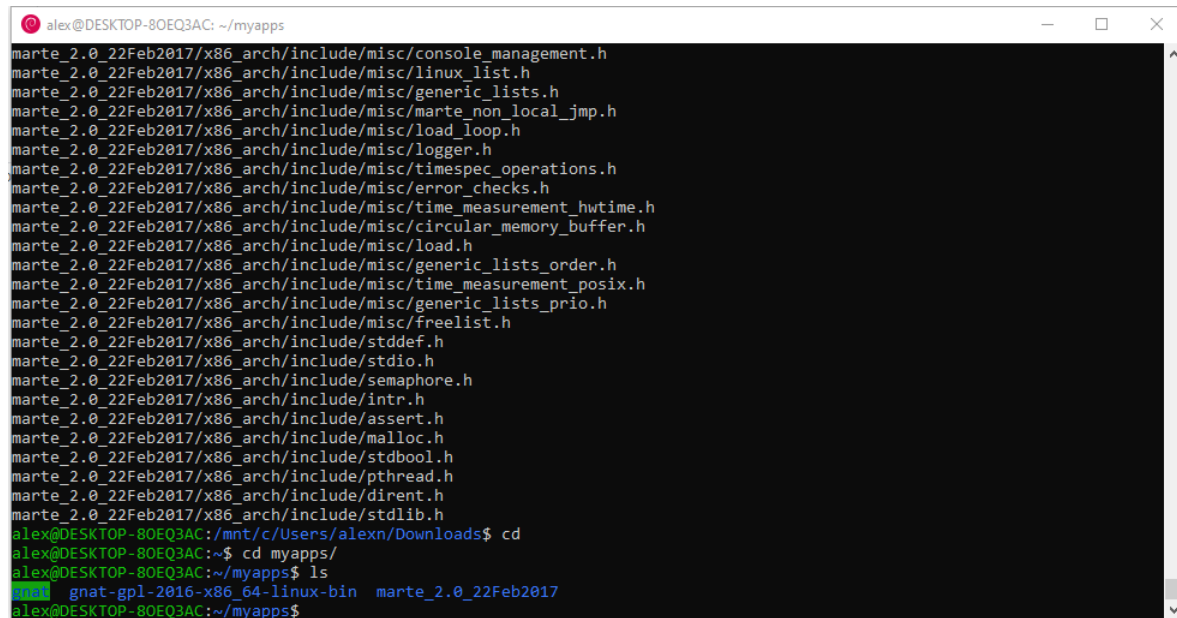
```
alex@DESKTOP-80EQ3AC: /mnt/c/Users/alexn/Downloads
alex@DESKTOP-80EQ3AC:/mnt/c/Users/alexn/Downloads$ tar xvf gnat-gpl-2016-x86_64-linux-bin.tar.gz -C /home/alex/myapps/
```

Una vez terminado este proceso, procederemos a descomprimir el archivo contenedor del SOTR y eso lo haremos con la siguiente instrucción “tar xvf marte_2.0_22Feb2017_src.tar.gz ruta_a_myapps”. Como se muestra a continuación.

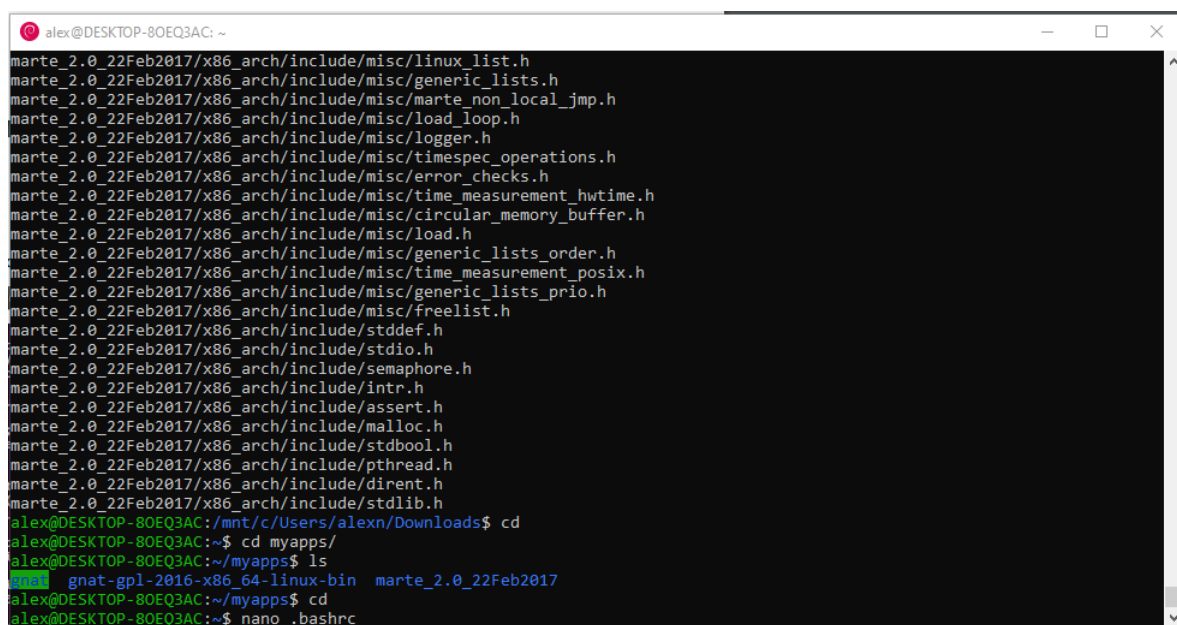


```
alex@DESKTOP-80EQ3AC: /mnt/c/Users/alexn/Downloads
alex@DESKTOP-80EQ3AC:/mnt/c/Users/alexn/Downloads$ tar xvf marte_2.0_22Feb2017_src.tar.gz -C /home/alex/myapps/
gnat-gpl-2016-x86_64-linux-bin/lib/gps/libgdk-3.so.0
gnat-gpl-2016-x86_64-linux-bin/lib/gps/libstdc++.so.6
gnat-gpl-2016-x86_64-linux-bin/lib/gps/libiconv.so
gnat-gpl-2016-x86_64-linux-bin/lib/libxmlada_unicode.so
gnat-gpl-2016-x86_64-linux-bin/COPYING3
gnat-gpl-2016-x86_64-linux-bin/config.guess
gnat-gpl-2016-x86_64-linux-bin/doinstall
gnat-gpl-2016-x86_64-linux-bin/etc/
gnat-gpl-2016-x86_64-linux-bin/etc/gtk-3.0/
gnat-gpl-2016-x86_64-linux-bin/etc/gtk-3.0/settings.ini
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/fonts.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/60-latin.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/30-metric-aliases.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/51-local.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/README
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/10-scale-bitmap-fonts.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/65-nonlatin.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/45-latin.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/30-urw-aliases.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/50-user.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/49-sansserif.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/40-nonlatin.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/69-unifont.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/80-delicious.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/90-synthetic.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/65-fonts-persian.conf
gnat-gpl-2016-x86_64-linux-bin/etc/fonts/conf.d/20-unhint-small-vera.conf
alex@DESKTOP-80EQ3AC:/mnt/c/Users/alexn/Downloads$
```

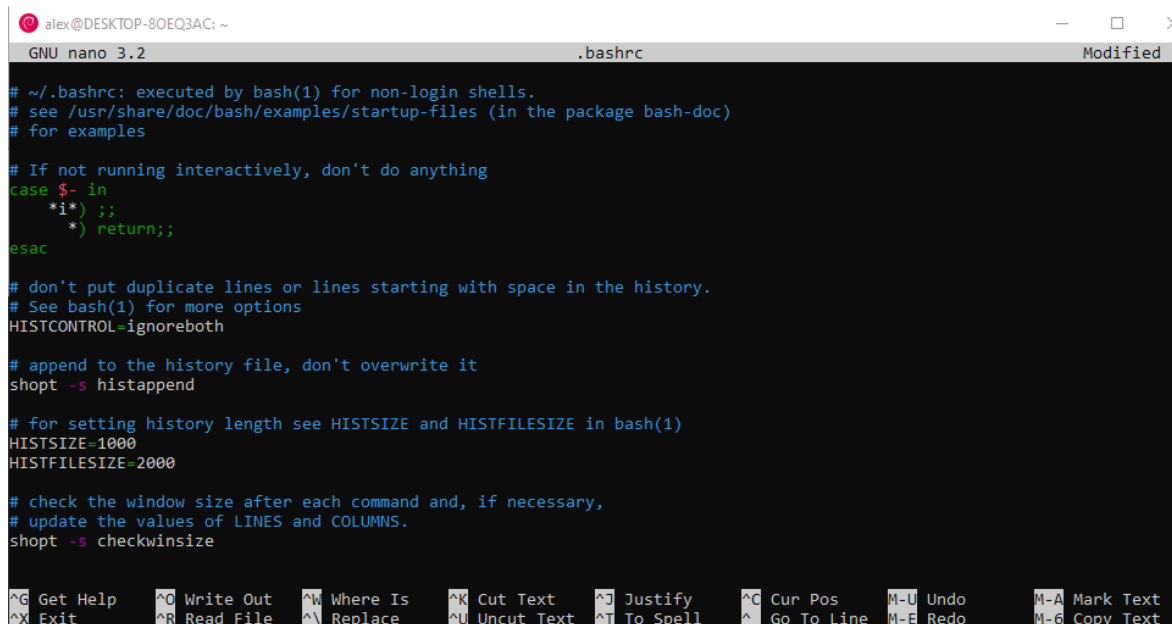
Como podemos ver, mediante el comando “ls”, ahora tenemos tres elementos dentro de la carpeta *myapps*.

A terminal window titled 'alex@DESKTOP-80EQ3AC: ~/myapps'. It displays a long list of files and directories starting with 'marte_2.0_22Feb2017/x86_arch/include/misc/'. The list includes files like 'console_management.h', 'linux_list.h', 'generic_lists.h', 'marte_non_local_jump.h', 'load_loop.h', 'logger.h', 'timespec_operations.h', 'error_checks.h', 'time_measurement_hwttime.h', 'circular_memory_buffer.h', 'load.h', 'generic_lists_order.h', 'time_measurement_posix.h', 'generic_lists_prio.h', 'freelist.h', 'stddef.h', 'stdio.h', 'semaphore.h', 'intr.h', 'assert.h', 'malloc.h', 'stdbool.h', 'pthread.h', 'dirent.h', and 'stdlib.h'. The prompt then changes to 'alex@DESKTOP-80EQ3AC:~/myapps\$ ls', which shows three items: 'gnat-gpl-2016-x86_64-linux-bin', 'marte_2.0_22Feb2017', and another 'marte_2.0_22Feb2017'.

Una vez que hemos descomprimido los archivos contenedores del compilador y el SOTR, lo siguiente es modificar el archivo *bashrc* para poder instalar el SOTR sin ningún problema, para ello haremos lo siguiente, en la misma terminal, será necesario salir a la raíz del sistema, esto lo lograremos con la instrucción “cd”, una vez ahí, ejecutaremos el comando “nano .bashrc”, como se muestra a continuación.

A terminal window titled 'alex@DESKTOP-80EQ3AC: ~'. It shows the same list of files as the previous screenshot. The prompt then changes to 'alex@DESKTOP-80EQ3AC:~/myapps\$ ls', showing the same three items. Then, the prompt changes to 'alex@DESKTOP-80EQ3AC:~/myapps\$ cd', which returns to the root directory. Finally, the prompt changes to 'alex@DESKTOP-80EQ3AC:~\$ nano .bashrc', indicating that the nano text editor is being used to edit the .bashrc file.

Esta instrucción nos dará acceso al siguiente archivo.



```
alex@DESKTOP-80EQ3AC: ~
GNU nano 3.2 .bashrc Modified

# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
  *i*) ;;
  *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

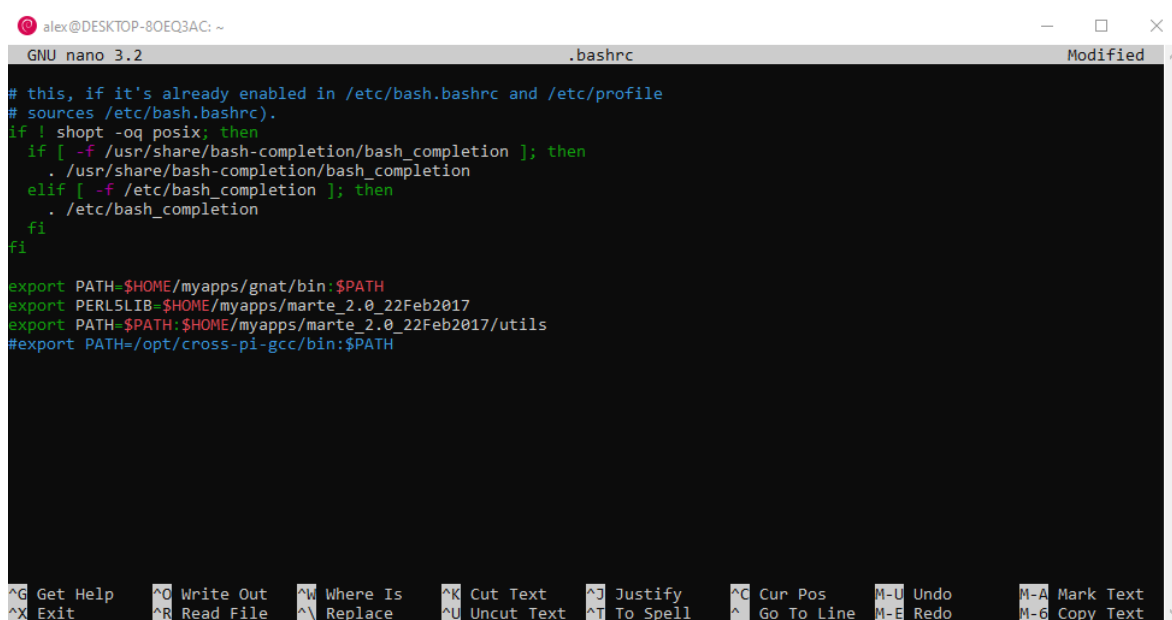
# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo      M-A Mark Text
^X Exit      ^R Read File  ^_ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line  M-E Redo      M-6 Copy Text
```

Dentro de este archivo, iremos al final y agregaremos las siguientes líneas:

- export PATH=\$HOME/myapps/gnat/bin:\$PATH
- export PERLSLIB=\$HOME/myapps/marte_2.0_22Feb2017
- export PATH=\$PATH:\$HOME/myapps/marte_2.0_22Feb2017/utils
- #export PATH=/opt/cross-pi-gcc/bin:\$PATH



```
alex@DESKTOP-80EQ3AC: ~
GNU nano 3.2 .bashrc Modified ^

# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

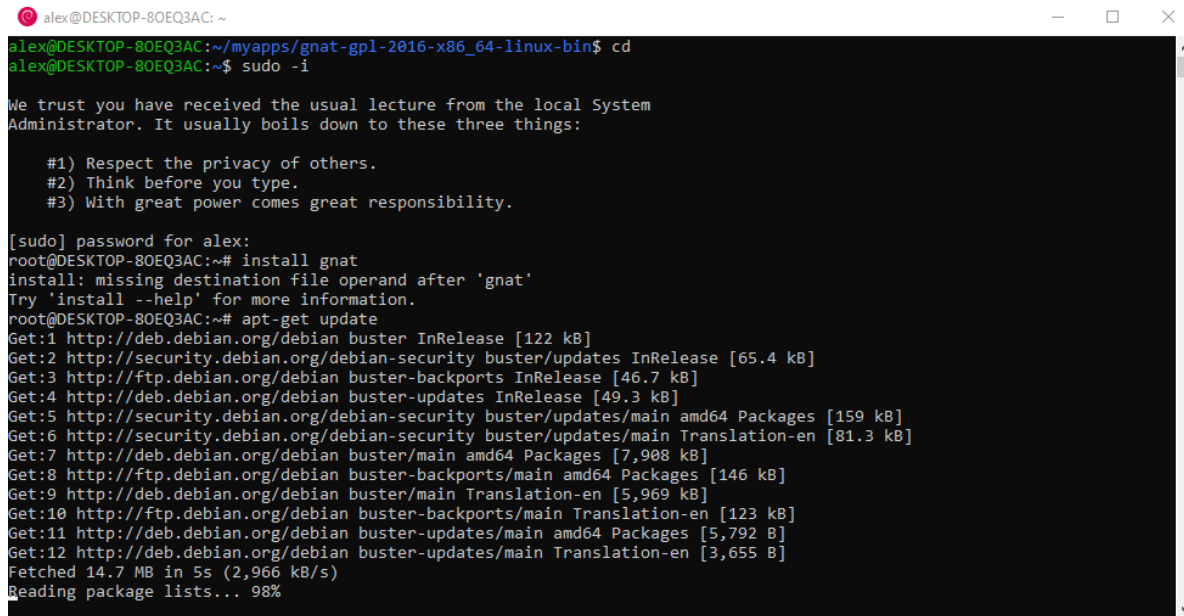
export PATH=$HOME/myapps/gnat/bin:$PATH
export PERLSLIB=$HOME/myapps/marte_2.0_22Feb2017
export PATH=$PATH:$HOME/myapps/marte_2.0_22Feb2017/utils
#export PATH=/opt/cross-pi-gcc/bin:$PATH

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo      M-A Mark Text
^X Exit      ^R Read File  ^_ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line  M-E Redo      M-6 Copy Text
```

Finalmente, como lo indica el archivo, hay que guardar los cambios efectuados (CTRL+O) para después salir del archivo (CTRL+X).

Es importante que luego de hacer esto, cerremos la terminal en la que nos encontrábamos, esto para que los cambios que acabamos de realizar puedan efectuarse exitosamente.

Luego de abrir la terminal de Debian nuevamente, ingresamos a la raíz mediante el comando “sudo -i”, y luego de escribir la contraseña del usuario ingresamos el comando “apt-get update”.

A screenshot of a terminal window titled 'alex@DESKTOP-80EQ3AC: ~'. The user is in a directory '~/myapps/gnat-gpl-2016-x86_64-linux-bin'. They run 'cd' and then 'sudo -i'. The terminal shows the root prompt and a message about system administrator responsibilities. Then, the user runs 'install gnat', which fails with a missing destination file operand. Next, they run 'apt-get update', which shows a list of packages to be updated from various Debian mirrors, including security updates and main packages. The process fetches 14.7 MB in 5 seconds at 2,966 kB/s and reads package lists at 98% completion.

```
alex@DESKTOP-80EQ3AC: ~  
alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$ cd  
alex@DESKTOP-80EQ3AC:~$ sudo -i  
  
We trust you have received the usual lecture from the local System  
Administrator. It usually boils down to these three things:  
  
#1) Respect the privacy of others.  
#2) Think before you type.  
#3) With great power comes great responsibility.  
  
[sudo] password for alex:  
root@DESKTOP-80EQ3AC:~# install gnat  
install: missing destination file operand after 'gnat'  
Try 'install --help' for more information.  
root@DESKTOP-80EQ3AC:~# apt-get update  
Get:1 http://deb.debian.org/debian buster InRelease [122 kB]  
Get:2 http://security.debian.org/debian-security buster/updates InRelease [65.4 kB]  
Get:3 http://ftp.debian.org/debian buster-backports InRelease [46.7 kB]  
Get:4 http://deb.debian.org/debian buster-updates InRelease [49.3 kB]  
Get:5 http://security.debian.org/debian-security buster/updates/main amd64 Packages [159 kB]  
Get:6 http://security.debian.org/debian-security buster/updates/main Translation-en [81.3 kB]  
Get:7 http://deb.debian.org/debian buster/main amd64 Packages [7,908 kB]  
Get:8 http://ftp.debian.org/debian buster-backports/main amd64 Packages [146 kB]  
Get:9 http://deb.debian.org/debian buster/main Translation-en [5,969 kB]  
Get:10 http://ftp.debian.org/debian buster-backports/main Translation-en [123 kB]  
Get:11 http://deb.debian.org/debian buster-updates/main amd64 Packages [5,792 B]  
Get:12 http://deb.debian.org/debian buster-updates/main Translation-en [3,655 B]  
Fetched 14.7 MB in 5s (2,966 kB/s)  
Reading package lists... 98%
```

Al terminar el proceso correspondiente ahora instalamos *vim* mediante el comando “apt-get install vim”.


```
alex@DESKTOP-80EQ3AC: ~  
alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$ cd  
alex@DESKTOP-80EQ3AC:~$ sudo -i  
  
We trust you have received the usual lecture from the local System  
Administrator. It usually boils down to these three things:  
  
#1) Respect the privacy of others.  
#2) Think before you type.  
#3) With great power comes great responsibility.  
  
[sudo] password for alex:  
root@DESKTOP-80EQ3AC:~# install gnat  
install: missing destination file operand after 'gnat'  
Try 'install --help' for more information.  
root@DESKTOP-80EQ3AC:~# apt-get update  
Get:1 http://deb.debian.org/debian buster InRelease [122 kB]  
Get:2 http://security.debian.org/debian-security buster/updates InRelease [65.4 kB]  
Get:3 http://ftp.debian.org/debian buster-backports InRelease [46.7 kB]  
Get:4 http://deb.debian.org/debian buster-updates InRelease [49.3 kB]  
Get:5 http://security.debian.org/debian-security buster/updates/main amd64 Packages [159 kB]  
Get:6 http://security.debian.org/debian-security buster/updates/main Translation-en [81.3 kB]  
Get:7 http://deb.debian.org/debian buster/main amd64 Packages [7,908 kB]  
Get:8 http://ftp.debian.org/debian buster-backports/main amd64 Packages [146 kB]  
Get:9 http://deb.debian.org/debian buster/main Translation-en [5,969 kB]  
Get:10 http://ftp.debian.org/debian buster-backports/main Translation-en [123 kB]  
Get:11 http://deb.debian.org/debian buster-updates/main amd64 Packages [5,792 B]  
Get:12 http://deb.debian.org/debian buster-updates/main Translation-en [3,655 B]  
Fetched 14.7 MB in 5s (2,966 kB/s)  
Reading package lists... Done  
root@DESKTOP-80EQ3AC:~# apt-get install vim
```

De la misma manera instalamos *bless* con “apt-get install bless” y *make* con “apt-get install make”

```
alex@DESKTOP-80EQ3AC: ~  
After this operation, 33.3 MB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://deb.debian.org/debian buster/main amd64 libgpm2 amd64 1.20.7-5 [35.1 kB]  
Get:2 http://deb.debian.org/debian buster/main amd64 vim-runtime all 2:8.1.0875-5 [5,775 kB]  
Get:3 http://deb.debian.org/debian buster/main amd64 vim amd64 2:8.1.0875-5 [1,280 kB]  
Fetched 7,090 kB in 2s (4,043 kB/s)  
Selecting previously unselected package libgpm2:amd64.  
(Reading database ... 9861 files and directories currently installed.)  
Preparing to unpack .../libgpm2_1.20.7-5_amd64.deb ...  
Unpacking libgpm2:amd64 (1.20.7-5) ...  
Selecting previously unselected package vim-runtime.  
Preparing to unpack .../vim-runtime_2%3a8.1.0875-5_all.deb ...  
Adding 'diversion of /usr/share/vim/vim81/doc/help.txt to /usr/share/vim/vim81/doc/help.txt.vim-tiny by vim-runtime'  
Adding 'diversion of /usr/share/vim/vim81/doc/tags to /usr/share/vim/vim81/doc/tags.vim-tiny by vim-runtime'  
Unpacking vim-runtime (2:8.1.0875-5) ...  
Selecting previously unselected package vim.  
Preparing to unpack .../vim_2%3a8.1.0875-5_amd64.deb ...  
Unpacking vim (2:8.1.0875-5) ...  
Setting up libgpm2:amd64 (1.20.7-5) ...  
Setting up vim-runtime (2:8.1.0875-5) ...  
Setting up vim (2:8.1.0875-5) ...  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vim (vim) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vimdiff (vimdiff) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rvim (rvim) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/rview (rview) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/vi (vi) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/view (view) in auto mode  
update-alternatives: using /usr/bin/vim.basic to provide /usr/bin/ex (ex) in auto mode  
Processing triggers for libc-bin (2.28-10) ...  
root@DESKTOP-80EQ3AC:~# apt-get install bless
```

```
alex@DESKTOP-80EQ3AC: ~  
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 2006 VeriSign, Inc. - For authorized use  
only", CN=VeriSign Class 3 Public Primary Certification Authority - G5  
Certificate added: C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU="(c) 2008 VeriSign, Inc. - For authorized use  
only", CN=VeriSign Universal Root Certification Authority  
Certificate added: C=US, OU=www.xrampsecurity.com, O=XRamp Security Services Inc, CN=XRamp Global Certification Authorit  
y  
128 new root certificates were added to your trust store.  
Import process completed.  
Done  
done.  
Processing triggers for libgdk-pixbuf2.0-0:amd64 (2.38.1+dfsg-1) ...  
root@DESKTOP-80EQ3AC:~# apt-get install make  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Suggested packages:  
  make-doc  
The following NEW packages will be installed:  
  make  
0 upgraded, 1 newly installed, 0 to remove and 22 not upgraded.  
Need to get 341 kB of archives.  
After this operation, 1,327 kB of additional disk space will be used.  
Get:1 http://deb.debian.org/debian buster/main amd64 make amd64 4.2.1-1.2 [341 kB]  
Fetched 341 kB in 0s (1,057 kB/s)  
Selecting previously unselected package make.  
(Reading database ... 22784 files and directories currently installed.)  
Preparing to unpack .../make_4.2.1-1.2_amd64.deb ...  
Unpacking make (4.2.1-1.2) ...  
Setting up make (4.2.1-1.2) ...  
root@DESKTOP-80EQ3AC:~#
```

Por ultimo instalamos gcc, qemu y unzip mediante los mismos comandos. Para salir de la raíz escribimos “exit”.

```
alex@DESKTOP-80EQ3AC:~$ sudo -i  
[sudo] password for alex:  
root@DESKTOP-80EQ3AC:~# apt-get install gcc  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
  
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/examples  
/home/alex/myapps/marte_2.0_22Feb2017/kernel write_marte_c_headers.adb  
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utills$ cd  
alex@DESKTOP-80EQ3AC:~$ sudo apt-get install qemu  
[sudo] password for alex:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following NEW packages will be installed:  
  qemu  
0 upgraded, 1 newly installed, 0 to remove and 20 not upgraded.  
Need to get 69.5 kB of archives.  
After this operation, 98.3 kB of additional disk space will be used.  
Get:1 http://security.debian.org/debian-security buster/updates/main amd64 qemu amd64 1:3.1+dfsg-8+deb10u3 [69.5 kB]  
Fetched 69.5 kB in 0s (334 kB/s)  
Selecting previously unselected package qemu.  
(Reading database ... 28534 files and directories currently installed.)  
Preparing to unpack .../qemu_1%3a3.1+dfsg-8+deb10u3_amd64.deb ...  
Unpacking qemu (1:3.1+dfsg-8+deb10u3) ...  
Setting up qemu (1:3.1+dfsg-8+deb10u3) ...
```

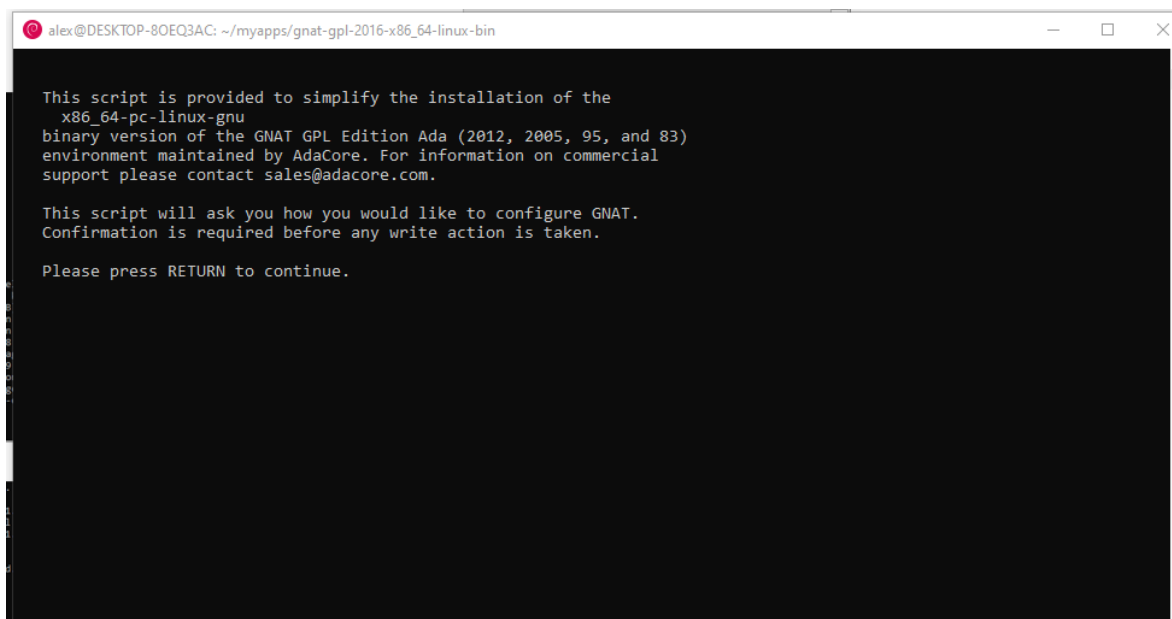
```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/examples
/home/alex/myapps/marte_2.0_22Feb2017/kernel write_marte_c_headers.adb
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utills$ cd
alex@DESKTOP-80EQ3AC:~$ sudo apt-get install qemu
[sudo] password for alex:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  qemu
0 upgraded, 1 newly installed, 0 to remove and 20 not upgraded.
Need to get 69.5 kB of archives.
After this operation, 98.3 kB of additional disk space will be used.
Get:1 http://security.debian.org/debian-security buster/updates/main amd64 qemu amd64 1:3.1+dfsg-8+deb10u3 [69.5 kB]
Fetched 69.5 kB in 0s (334 kB/s)
Selecting previously unselected package qemu.
(Reading database ... 28534 files and directories currently installed.)
Preparing to unpack .../qemu_1%3a3.1+dfsg-8+deb10u3_amd64.deb ...
Unpacking qemu (1:3.1+dfsg-8+deb10u3) ...
Setting up qemu (1:3.1+dfsg-8+deb10u3) ...
alex@DESKTOP-80EQ3AC:~$ cd myapps/marte_2.0_22Feb2017/examples/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ ls
ada          drivers      hello_world.adb  logger          posix        time_measurement
appsched     games       hello_world.c.c  Makefile        README       widgets
clock_modulation hardware_interrupts hello_world_cc.cc oscilloscope    speaker
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$
```

INSTALACIÓN DEL SOTR

Luego en la carpeta myapps nuevamente abrimos una nueva terminal. Donde entraremos a la carpeta contenedora del compilador GNAT, esto lo lograremos con la instrucción “cd gnat-gpl-2016-x86_64-linux-bin/”, una vez situados en esa carpeta ejecutaremos la siguiente instrucción “./doinstall”.

```
alex@DESKTOP-80EQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin
alex@DESKTOP-80EQ3AC:~$ cd myapps/gnat-gpl-2016-x86_64-linux-bin/
alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$ ./doinstall
```

Y se nos presentara una ventana donde se nos describe la versión del compilador que vamos a instalar, para seguir con la instalación solo presionamos “ENTER”.



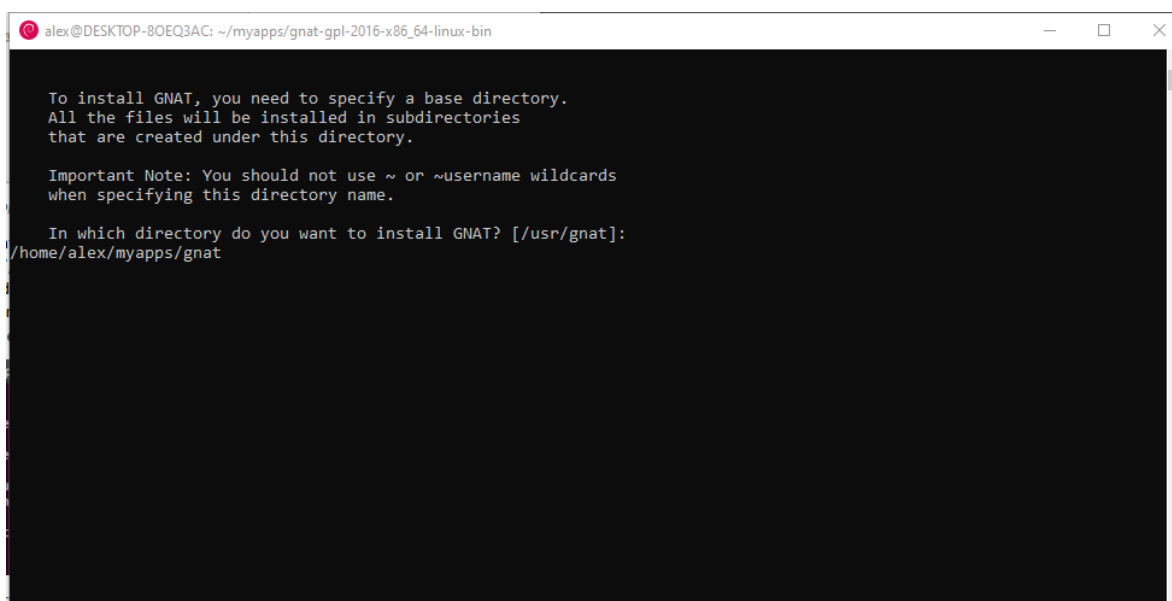
```
alex@DESKTOP-8OEQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

This script is provided to simplify the installation of the
x86_64-pc-linux-gnu
binary version of the GNAT GPL Edition Ada (2012, 2005, 95, and 83)
environment maintained by AdaCore. For information on commercial
support please contact sales@adacore.com.

This script will ask you how you would like to configure GNAT.
Confirmation is required before any write action is taken.

Please press RETURN to continue.
```

Posteriormente, se nos preguntara la dirección de la carpeta donde queremos instalar el compilador. Es aquí donde entra en juego la carpeta “gnat” que creamos en un inicio dentro de la carpeta “myapps”. Entonces la cadena que ingresaremos, quedara de la siguiente manera “/home/usuario/myapps/gnat”, con esto estaremos indicando exactamente dónde queremos que se instale el compilador.



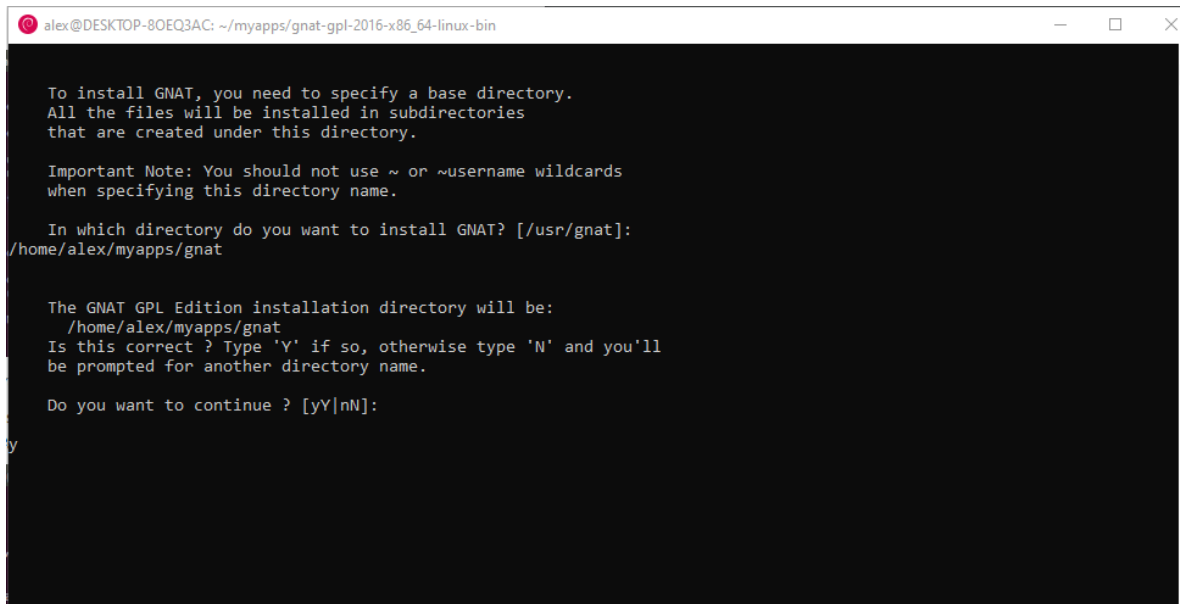
```
alex@DESKTOP-8OEQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

To install GNAT, you need to specify a base directory.
All the files will be installed in subdirectories
that are created under this directory.

Important Note: You should not use ~ or ~username wildcards
when specifying this directory name.

In which directory do you want to install GNAT? [/usr/gnat]:
/home/alex/myapps/gnat
```

Para proseguir solo oprimimos la tecla “ENTER”. Se nos preguntara si la ruta de instalación es la correcta, a lo que solo debemos oprimir “Y” y luego “ENTER”.



```
alex@DESKTOP-80EQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

To install GNAT, you need to specify a base directory.
All the files will be installed in subdirectories
that are created under this directory.

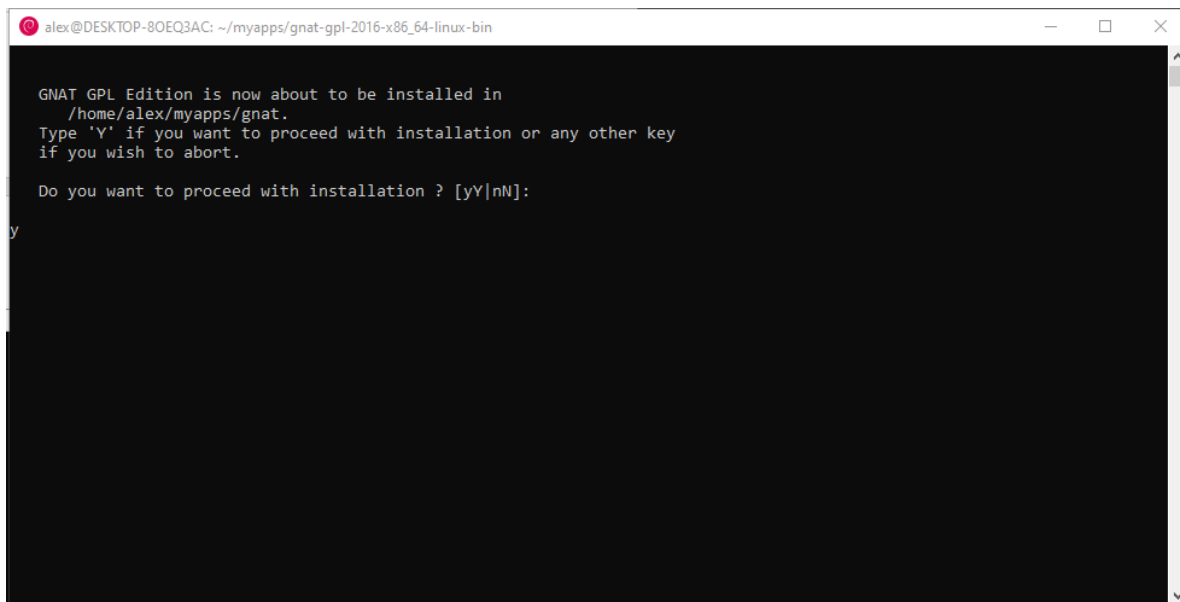
Important Note: You should not use ~ or ~username wildcards
when specifying this directory name.

In which directory do you want to install GNAT? [/usr/gnat]:
/home/alex/myapps/gnat

The GNAT GPL Edition installation directory will be:
/home/alex/myapps/gnat
Is this correct ? Type 'Y' if so, otherwise type 'N' and you'll
be prompted for another directory name.

Do you want to continue ? [yY|nN]:
y
```

Acto seguido se nos preguntara si queremos proceder con la instalación, y haremos lo mismo que anteriormente presionamos la tecla “Y” y seguido de eso la tecla “ENTER”.



```
alex@DESKTOP-80EQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

GNAT GPL Edition is now about to be installed in
/home/alex/myapps/gnat.
Type 'Y' if you want to proceed with installation or any other key
if you wish to abort.

Do you want to proceed with installation ? [yY|nN]:
y
```

Visualizaremos una ventana como la siguiente, señal de que el proceso de instalación ha comenzado.

```
alex@DESKTOP-80EQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

GNAT GPL Edition is now about to be installed in
/home/alex/myapps/gnat.
Type 'Y' if you want to proceed with installation or any other key
if you wish to abort.

Do you want to proceed with installation ? [yY|nN]:
y
rm -fr "/home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4"/rts*
rm -fr "/home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4/adalib" "/home/alex/myapps/gnat/lib/gcc/x86_64-pc-l
linux-gnu/4.9.4/adalib"
rm -fr "/home/alex/myapps/gnat/share/doc/gnat"
rm -fr "/home/alex/myapps/gnat/share/examples/gnat"
mkdir -p "/home/alex/myapps/gnat"
rm -f "/home/alex/myapps/gnat"/lib/libgcc*
rm -f "/home/alex/myapps/gnat"/bin/gnat[!p]*
rm -f "/home/alex/myapps/gnat"/bin/gnatpp*
rm -f "/home/alex/myapps/gnat"/bin/gnatprep*
rm -f "/home/alex/myapps/gnat"/bin/gnat
rm -f "/home/alex/myapps/gnat"/bin/gpr*
rm -f "/home/alex/myapps/gnat"/bin/gcc "/home/alex/myapps/gnat"/bin/x86_64-pc-linux-gnu-gcc
for d in bin lib libexec lib32 lib64 include \
    doc examples share etc DLLs x86_64-pc-linux-gnu; do \
    if [ -d "$d" ]; then \
        tar cf - "$d" | (cd "/home/alex/myapps/gnat" && tar xf -); \
    fi \
done
```

El proceso anterior puede demorar varios minutos, por lo que es importante no cerrar la ventana hasta que podamos visualizar lo siguiente, señal de que el proceso ha culminado.

```
alex@DESKTOP-80EQ3AC: ~/myapps/gnat-gpl-2016-x86_64-linux-bin

GNAT GPL is now installed. To launch it, you must put
/home/alex/myapps/gnat/bin
in front of your PATH environment variable. The following
commands enable you to do this:
    PATH="/home/alex/myapps/gnat/bin:$PATH"; export PATH  (Bourne shell)
    setenv PATH "/home/alex/myapps/gnat/bin:$PATH"      (C shell)
Thank you for installing GNAT GPL Edition!

alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$
```

Lo siguiente es salir a la carpeta “myapps” y acceder a la carpeta “marte_2.0_22Feb2017”, para lograr eso, ejecutaremos la instrucción “cd ..” y luego de ello, ejecutaremos la instrucción “cd marte_2.0_22Feb2017/”. Una vez ahí procederemos a instalar el SOTR, para lo cual ejecutaremos la instrucción “./minstall”, con lo que visualizaremos lo siguiente.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017

GNAT GPL is now installed. To launch it, you must put
/home/alex/myapps/gnat/bin
in front of your PATH environment variable. The following
commands enable you to do this:
  PATH="/home/alex/myapps/gnat/bin:$PATH"; export PATH  (Bourne shell)
  setenv PATH "/home/alex/myapps/gnat/bin:$PATH"      (C shell)
Thank you for installing GNAT GPL Edition!

alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$ cd ..
alex@DESKTOP-80EQ3AC:~/myapps$ cd marte_2.0_22Feb2017/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ ./mininstall
```

Para continuar presionamos la tecla “ENTER”. Y a diferencia del compilador este proceso es bastante rápido, por lo que pronto visualizaremos lo siguiente, en señal de que el proceso ha terminado exitosamente.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017

GNAT GPL is now installed. To launch it, you must put
/home/alex/myapps/gnat/bin
in front of your PATH environment variable. The following
commands enable you to do this:
  PATH="/home/alex/myapps/gnat/bin:$PATH"; export PATH  (Bourne shell)
  setenv PATH "/home/alex/myapps/gnat/bin:$PATH"      (C shell)
Thank you for installing GNAT GPL Edition!

alex@DESKTOP-80EQ3AC:~/myapps/gnat-gpl-2016-x86_64-linux-bin$ cd ..
alex@DESKTOP-80EQ3AC:~/myapps$ cd marte_2.0_22Feb2017/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ ./mininstall
main::check_if_gnat_is_present() called too early to check prototype at ./mininstall line 30.
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/templates/global.pl line 16.

Welcome to MaRTE OS installation process
=====
      http://marte.unican.es/
Universidad de Cantabria, SPAIN

To perform successfully this installation compilers:
  - GNAT-GPL-2016-x86_64 (for x86, linux and linux_lib architectures)
  - GNAT-GPL-2016-arm-elf (for rpi architecture)
should already be installed in your system, and GNAT 'bin/' directory
should be in front of your $PATH environment variable.

You also need to have write permission on the GNAT compiler directory.

Press any key to start the installation...
```

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017
Copying utilities from templates directory...

Change gnat-gpl-2016 libraries for the 32bits versions
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
Set 32bits libs in /home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4

    --- (-) MaRTE OS installation script finished (-) ---

You may want to add "/home/alex/myapps/marte_2.0_22Feb2017/utls"
to your $PATH environment variable to have direct access to MaRTE
tools (mgntmake, mgcc, mkmarte, mkrtsmarteuc, msetcurrentarch, etc.)

In this installation, MaRTE OS can generate applications for the
following architectures:

- linux: Linux operating system
- linux_lib: Linux operating system (using Linux file system)
- x86: x86 bare machine

This is a MaRTE source distribution, so you must compile MaRTE libraries
before using them. For example, for "linux" architecture execute:

    $ msetcurrentarch linux && mkrtsmarteuc && mkmarte

For more information go to chapter 1.2 of the 'INSTALL' document

alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$
```

Estamos cerca de terminar la instalación, lo siguientes es acceder a la carpeta “utls”, lo que lograremos con la siguiente instrucción “cd utls/” ya que esta carpeta esta dentro de la carpeta “marte_2.0_22Feb2017”, una vez estando dentro de la carpeta utls, procederemos a definir la arquitectura sobre la que trabajara nuestro SOTR, para ello ingresaremos la siguiente instrucción a la terminal “msetcurrentarch”.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/utls

Change gnat-gpl-2016 libraries for the 32bits versions
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
Set 32bits libs in /home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4

    --- (-) MaRTE OS installation script finished (-) ---

You may want to add "/home/alex/myapps/marte_2.0_22Feb2017/utls"
to your $PATH environment variable to have direct access to MaRTE
tools (mgntmake, mgcc, mkmarte, mkrtsmarteuc, msetcurrentarch, etc.)

In this installation, MaRTE OS can generate applications for the
following architectures:

- linux: Linux operating system
- linux_lib: Linux operating system (using Linux file system)
- x86: x86 bare machine

This is a MaRTE source distribution, so you must compile MaRTE libraries
before using them. For example, for "linux" architecture execute:

    $ msetcurrentarch linux && mkrtsmarteuc && mkmarte

For more information go to chapter 1.2 of the 'INSTALL' document

alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ cd utls/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utls$ msetcurrentarch
```

Y podremos ver que tenemos varias arquitecturas para elegir. Entre ellas están:

- X86
- Linux
- Linux_lib
- Rpi


```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/utls
In this installation, MaRTE OS can generate applications for the
following architectures:
- linux: Linux operating system
- linux_lib: Linux operating system (using Linux file system)
- x86: x86 bare machine

This is a MaRTE source distribution, so you must compile MaRTE libraries
before using them. For example, for "linux" architecture execute:

$ msetcurrentarch linux && mkrtsmarteuc && mkmarte

For more information go to chapter 1.2 of the 'INSTALL' document

alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ cd utls/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utls$ msetcurrentarch
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
Current architecture:none

Available architectures status:
x86:      RTS (gnat_rts/rts-marteuc_x86): NOT Compiled
          Lib MaRTE (objs/x86_objs):      NOT Compiled
linux:    RTS (gnat_rts/rts-marteuc_linux): NOT Compiled
          Lib MaRTE (objs/linux_objs):     NOT Compiled
linux_lib: RTS (gnat_rts/rts-marteuc_linux_lib): NOT Compiled
          Lib MaRTE (objs/linux_lib_objs): NOT Compiled
rpi:      NOT available
```

Para esta instalación, elegimos la arquitectura x86, por lo que ingresaremos la siguiente instrucción en la terminal “msetcurrentarch x86 i386”, una vez ejecutada, visualizaremos lo siguiente.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/utls
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utls$ msetcurrentarch x86 i386
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
msetcurrentarch: setting x86 as the default architecture...
Use of uninitialized value within %GNAT_VER_LIBS_GCC in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/msetcurrentarch line 224.
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/x86_arch/hwi/marte-x86_processor_type_i386.ads /home/alex/myapps/marte_2.0_22Feb2017/x86_arch/hwi/marte-x86_processor_type.ads
cd /home/alex/myapps/marte_2.0_22Feb2017/tests/ada/reports && cp reports-init.adb.x86 reports-init.adb
rm -f /home/alex/myapps/marte_2.0_22Feb2017/arch
ln -s /home/alex/myapps/marte_2.0_22Feb2017/x86_arch /home/alex/myapps/marte_2.0_22Feb2017/arch
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/sys_marte/marte_*.h /home/alex/myapps/marte_2.0_22Feb2017/arch/include/sys/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/marte-kernel-devices_table.ads /home/alex/myapps/marte_2.0_22Feb2017/kernel/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/marte-direct_io.ads /home/alex/myapps/marte_2.0_22Feb2017/kernel/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/console_management.adb /home/alex/myapps/marte_2.0_22Feb2017/misc/
rm -f /home/alex/myapps/marte_2.0_22Feb2017/misc//console_management.c.c
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/marte-configuration_parameters.ads /home/alex/myapps/marte_2.0_22Feb2017/kernel/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/marte-kernel-file_system.ads /home/alex/myapps/marte_2.0_22Feb2017/kernel/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/console_management.adb /home/alex/myapps/marte_2.0_22Feb2017/misc/
ln -s -f /home/alex/myapps/marte_2.0_22Feb2017/arch/arch_dependent_files/libm.a /home/alex/myapps/marte_2.0_22Feb2017/objs/x86_objs/
rm -f /home/alex/myapps/marte_2.0_22Feb2017/lib && ln -s /home/alex/myapps/marte_2.0_22Feb2017/objs/x86_objs /home/alex/myapps/marte_2.0_22Feb2017/lib
```

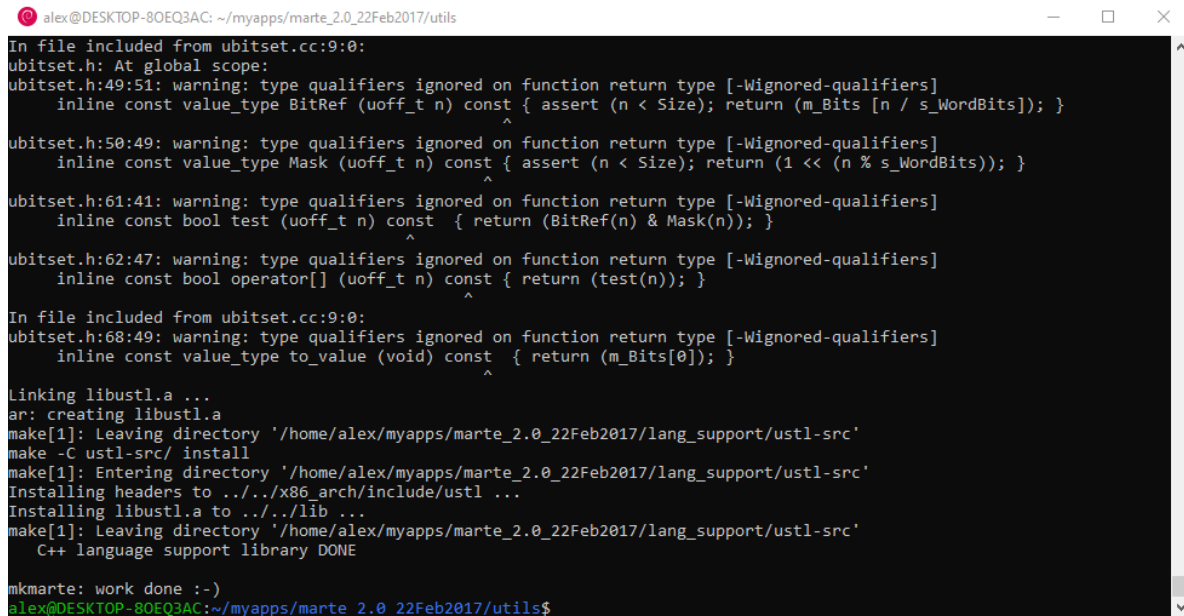
Como podemos ver, la arquitectura que elegimos ha sido puesta como default para el funcionamiento de nuestro SOTR.

Para continuar, seguiremos las indicaciones que vienen hasta abajo, que es ejecutar las instrucciones “mkrtsmarteuc” y posteriormente “mkmarte”, es importante que sea en ese orden, de lo contrario visualizaremos una ventana de error.

La primera instrucción tardará un poco en ejecutarse, pero es importante no cerrar la ventana de la terminal. Una vez que visualicemos lo siguiente, podemos continuar a ejecutar la instrucción “mkmarte”.

Y de hecho como podemos ver, al final podemos ver que se nos invita a ejecutar la instrucción “mkmarte” es por ello que es importante respetar el orden de ejecución de estas instrucciones.

Una vez que se ejecute la instrucción “mkmarte” podremos visualizar lo siguiente.



```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/utls
In file included from ubitset.cc:9:0:
ubitset.h: At global scope:
ubitset.h:49:51: warning: type qualifiers ignored on function return type [-Wignored-qualifiers]
    inline const value_type BitRef (uoff_t n) const { assert (n < Size); return (m_Bits [n / s_WordBits]); }
                                                    ^
ubitset.h:50:49: warning: type qualifiers ignored on function return type [-Wignored-qualifiers]
    inline const value_type Mask (uoff_t n) const { assert (n < Size); return (1 << (n % s_WordBits)); }
                                                    ^
ubitset.h:61:41: warning: type qualifiers ignored on function return type [-Wignored-qualifiers]
    inline const bool test (uoff_t n) const { return (BitRef(n) & Mask(n)); }
                                                    ^
ubitset.h:62:47: warning: type qualifiers ignored on function return type [-Wignored-qualifiers]
    inline const bool operator[] (uoff_t n) const { return (test(n)); }
                                                    ^
In file included from ubitset.cc:9:0:
ubitset.h:68:49: warning: type qualifiers ignored on function return type [-Wignored-qualifiers]
    inline const value_type to_value (void) const { return (m_Bits[0]); }
                                                    ^
Linking libustl.a ...
ar: creating libustl.a
make[1]: Leaving directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
make -C ustl-src/ install
make[1]: Entering directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
Installing headers to ../../x86_arch/include/ustl ...
Installing libustl.a to ../../lib ...
make[1]: Leaving directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
C++ language support library DONE

mkmarte: work done :-)
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utls$
```

Hasta este punto, terminamos con la instalación del SOTR MaRTE OS, lo siguiente es probar que realmente funcione, para poder hacerlo, requeriremos de la aplicación QEMU, si no la tenemos instalada, lo siguiente es ejecutar la instrucción “cd” para salir a la carpeta raíz y una vez ahí, ejecutar la instrucción “sudo apt-get install qemu”, acto seguido la terminal nos pedirá la contraseña usuario, se ingresa y se procede con la instalación de esta aplicación. Este proceso tardara unos minutos. Para saber que la instalación fue exitosa, deberemos visualizar lo siguiente.

Lo siguiente es ir a la carpeta de ejemplos de marte, para lo que ejecutaremos la siguiente instrucción “cd myapps/marte_2.0_22Feb2017/examples”. Una vez ahí ejecutamos la aplicación “ls” para visualizar los archivos que hay dentro de la carpeta de ejemplos.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/examples
ar: creating libustl.a
make[1]: Leaving directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
make -C ustl-src/ install
make[1]: Entering directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
Installing headers to ../../x86_arch/include/ustl ...
Installing libustl.a to ../../lib ...
make[1]: Leaving directory '/home/alex/myapps/marte_2.0_22Feb2017/lang_support/ustl-src'
C++ language support library DONE

mkmarte: work done :-)
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/utls$ cd ..
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ examples/
-bash: examples/: Is a directory
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017$ cd examples/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ ls
ada          drivers      hello_world.adb  logger         posix         time_measurement
appsched     games       hello_world.c.c  Makefile       README       widgets
clock_modulation hardware_interrupts hello_world_cc.cc oscilloscope  speaker
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ mgcc hello_world.c.c
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line
16.
gcc -nostdinc -I/home/alex/myapps/marte_2.0_22Feb2017/arch/include hello_world.c.c -m32 -march=i686 /home/alex/my
apps/marte_2.0_22Feb2017/arch/call_main/wrapper_main.c.o -Wl,-T,/home/alex/myapps/marte_2.0_22Feb2017/utls/linker.lds
-static -nostartfiles -L/home/alex/myapps/marte_2.0_22Feb2017/lib -L/home/alex/myapps/marte_2.0_22Feb2017/gnat_rts/rts/
adalib -L/home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4 -lmarte -lgnat -lgcc_sjlj
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ ls
ada          clock_modulation hardware_interrupts hello_world_cc.cc oscilloscope  speaker
a.out        drivers          hello_world.adb    logger         posix         time_measurement
appsched     games           hello_world.c.c    Makefile       README       widgets
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$
```

Como podemos ver, hay archivos para C, C++ y Ada, nosotros trabajaremos con el archivo “hello_world.c.c”, por consiguiente ejecutaremos la siguiente instrucción “mgcc hello_world.c.c” con esto habremos compilado este archivo y una forma de verificar que la compilación fue exitosa es revisar que se haya creado un archivo de nombre “a.out”, volvemos a ejecutar la instrucción “ls” para verificar que en efecto esto paso.

Ilustración 22 verificación de la creación del archivo a.out

Lo siguiente es ejecutar la siguiente instrucción “mgcc hello_world.c.c -o mprogram” con lo que crearemos un archivo de nombre “mprogram” asociado a la compilación de nuestro archivo “hello_world.c.c”.

Luego de esto crearemos un archivo ejecutable asociado al archivo C, esto lo lograremos con la siguiente instrucción “make hello_world_c.exe”.

```
alex@DESKTOP-80EQ3AC: ~/myapps/marte_2.0_22Feb2017/examples
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ cd examples/
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ ls
ada          drivers          hello_world.adb  logger          posix          time_measurement
appsched     games           hello_world.c.c  Makefile        README        widgets
clock_modulation hardware_interruptions hello_world.cc.cc oscilloscope    speaker
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ gcc hello_world.c.c
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
gcc -nostdinc -I/home/alex/myapps/marte_2.0_22Feb2017/arch/include hello_world.c.c -m32 -march=i686 /home/alex/myapps/marte_2.0_22Feb2017/arch/call_main/wrapper_main.c.o -Wl,-T,/home/alex/myapps/marte_2.0_22Feb2017/utls/linker.lds -static -nostartfiles -L/home/alex/myapps/marte_2.0_22Feb2017/lib -L/home/alex/myapps/marte_2.0_22Feb2017/gnat_rts/rts/adalib -L/home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4 -lmarte -lgnatl -lgnat -lmarte -lgcc_sjlj
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ ls
a.out        clock_modulation hardware_interruptions hello_world.cc.cc oscilloscope speaker
a.out        drivers          hello_world.adb      logger          posix          time_measurement
appsched     games           hello_world.c.c      Makefile        README        widgets
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ gcc hello_world.c.c -o mprogram
Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
gcc -nostdinc -I/home/alex/myapps/marte_2.0_22Feb2017/arch/include hello_world.c.c -m32 -march=i686 -o mprogram /home/alex/myapps/marte_2.0_22Feb2017/arch/call_main/wrapper_main.c.o -Wl,-T,/home/alex/myapps/marte_2.0_22Feb2017/utls/linker.lds -static -nostartfiles -L/home/alex/myapps/marte_2.0_22Feb2017/lib -L/home/alex/myapps/marte_2.0_22Feb2017/gnat_rts/rts/adalib -L/home/alex/myapps/gnat/lib/gcc/x86_64-pc-linux-gnu/4.9.4 -lmarte -lgnatl -lgnat -lmarte -lgcc_sjlj
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$ make hello_world.c.exe
>> Compiling hello_world.c.exe: Use of uninitialized value in concatenation (.) or string at /home/alex/myapps/marte_2.0_22Feb2017/utls/globals.pl line 16.
[OK]
alex@DESKTOP-80EQ3AC:~/myapps/marte_2.0_22Feb2017/examples$
```

Para verificar la creación de estos dos últimos archivos, volvemos a ejecutar la instrucción “ls”. Y como podemos ver, en efecto se han creado. El último paso es ejecutar la siguiente instrucción “qemu-system-i386 -kernel hello_world_c.exe”, esta instrucción nos ayudara a visualizar en una venta de qemu, el archivo ejecutable corriendo sobre nuestro SOTR.

```
== M a R T E  O S ==
V2.0 2017-02-22
Copyright (C) Universidad de Cantabria, SPAIN
TLFS 2.3.2 dynamic memory pool: 131514368 bytes
Devices initialization...
Major Number 1 (stdin) Keyboard ...OK
Major Number 2 (stdout) Text/Serial ...OK
Major Number 3 (stderr) Text/Serial ...OK
Hello, I'm a C program running on MaRTE OS.
_exit(0) called; rebooting...
Press a key to reboot
```

Con esto comprobamos que el SOTR MaRTE OS se instaló correctamente y ahora podemos trabajar sobre él.