

**Oracle® VM VirtualBox
Installation Instructions for Windows 10
and
Linux Virtual Machine Creation
Targeting Avnet Development Boards**

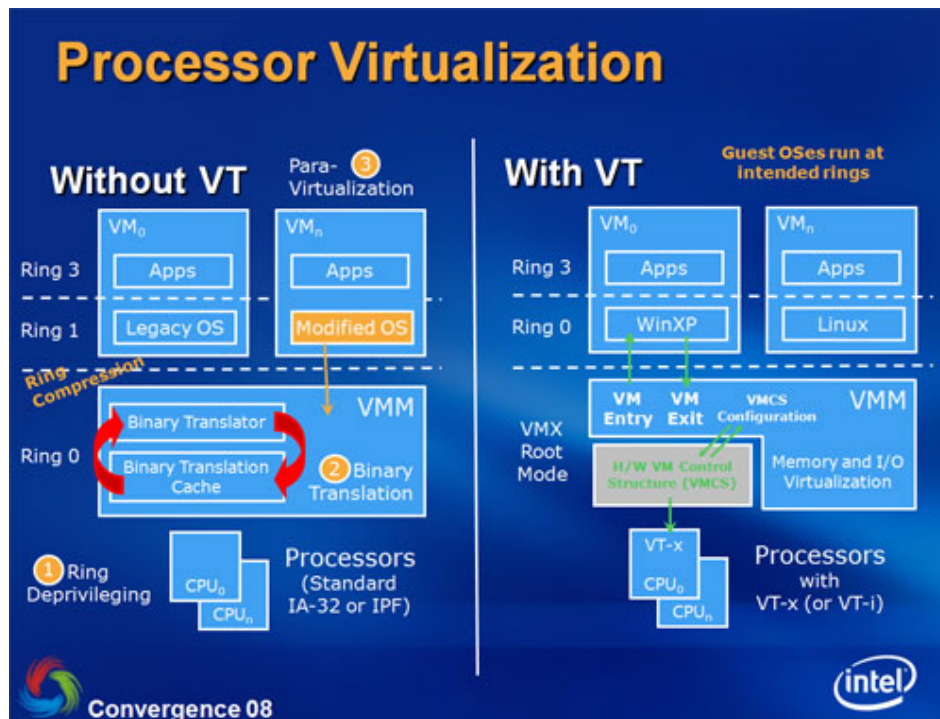
**Version 2.0
October 2019**

Preparing Your Machine

For best results with using VirtualBox Virtual Machines (VMs) as depicted in this document, some preparation of your machine is required to enable virtualization extensions for your CPU within the system BIOS (or UEFI) and to make sure that no conflicting virtualization packages are loaded.

Enabling CPU Virtualization Features:

The system BIOS (or UEFI) of a PC is responsible for enabling virtualization extensions and features of CPUs that support Virtualization Technology (VT). This is important because workstation and development workloads can co-locate while maintaining full isolation from each other. They can also freely migrate across infrastructures and scale as needed.



To enable your CPU virtualization extensions, you will need to reboot your PC and enter the (BIOS or UEFI) configuration menu. This often involved holding down a special key during the early stages of booting the PC so you may need to refer to the manufacturer documentation for your specific model of PC or motherboard to know which key is needed to enter the configuration screen. Once in the configuration screen, look for the **Intel VT-x**, **Intel Virtualization Technology**, or **Virtualization Extensions** options and make sure that they are enabled. Again, model specific PC manufacturer documentation may help here with locating the correct settings.

For further information on enabling virtualization features on the CPU of your development machine, there are other resources that provide guidance:

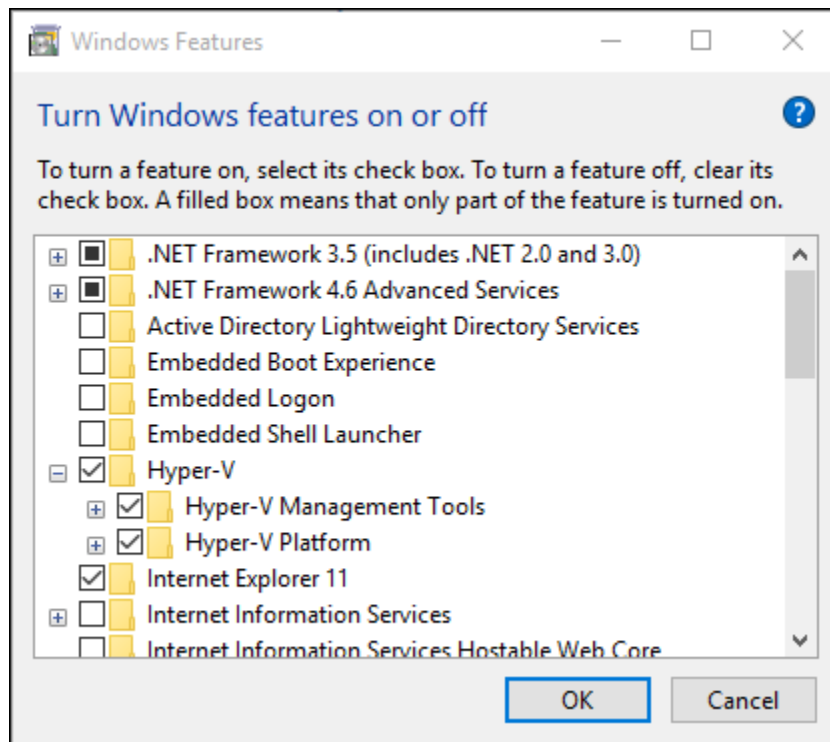
<https://www.howtogeek.com/213795/how-to-enable-intel-vt-x-in-your-computers-bios-or-uefi-firmware/>

Disabling Windows 10 Hyper-V:

Windows 10 Pro includes Microsoft provided VM enablement called Hyper-V. However, Hyper-V and VirtualBox cannot be used concurrently. The Hyper-V feature of Windows needs to be disabled in order to use VirtualBox. If Hyper-V is enabled VirtualBox may still run to a point and then VM will likely hang without any explanation from VirtualBox that Hyper-V is still enabled.

If more advanced users want to run Docker for Windows, it will require Hyper-V instead of VirtualBox.

The following dialog is where many Windows 10 features can be enabled/disabled, you can find it from the Win 10 search bar with the “Windows Features” search string and Hyper-V can be disabled by removing the checkmarks next to these features:



Microsoft has also provided an answer on the topic of disabling Windows 8 and Windows 10 Hyper-V.

https://answers.microsoft.com/en-us/windows/forum/windows_8-windows_install/how-do-i-uninstall-hyper-v/7d268911-47cd-4c52-bfe5-ea41e58067ab

Download Links

While this document shows how to setup and install a VirtualBox environment for use of a Linux virtual machine for the cross build platform, downloads are on the large side. It is recommended to begin all downloads ahead of time in order to save time later when continuing this document. The download links are also interleaved in the appropriate locations IN this document, however, it is recommended to download all files early as they can take quite some time.

VirtualBox and Extension Pack Download:

<https://www.virtualbox.org/wiki/Downloads>

Vivado and SDK:

<https://www.xilinx.com/support/download.html>

-OR-

SDSoC (includes Vivado and SDK):

<https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/sdx-development-environments.html>

Note that you will need an SDSoC license. If you only plan on using Vivado and SDK, there is **NO** need to download the SDSoC installer. If you are planning to use SDSoC, then it is strongly recommended to install the SDSoC installer **ONLY**, as this will include the Vivado HLS install, SDK, as well as the SDx extensions for the Eclipse environment.

The PetaLinux 2019.1 Installer:

<https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools/2019-1.html>

Installing Oracle VM VirtualBox on Windows 10

This document shows how to install VirtualBox and the Extension Pack to enable the use of a Linux virtual machine for the cross build platform.

General Instruction:

Install Oracle VM VirtualBox using the official VirtualBox installer. For legal distribution reasons, the VirtualBox installation executable cannot be included with any public Avnet materials. To obtain a free legal copy of the Oracle VM VirtualBox and the Extension Pack, please download from the VirtualBox website:

<https://www.virtualbox.org/wiki/Downloads>

The version downloaded may differ from the version shown in this documentation (6.0.12). Be sure to read the VirtualBox EULA to ensure you do not violate the *Personal Use and Evaluation License* (PUEL). You may also wish to consult the *VirtualBox Licensing Frequently Asked Questions* for a quick overview of the intent of the license agreements:

https://www.virtualbox.org/wiki/Licensing_FAQ

Step-by-Step Instructions:

1. To obtain a free legal copy of Oracle VM VirtualBox, download the installer from this website:

<https://www.virtualbox.org/wiki/Downloads>

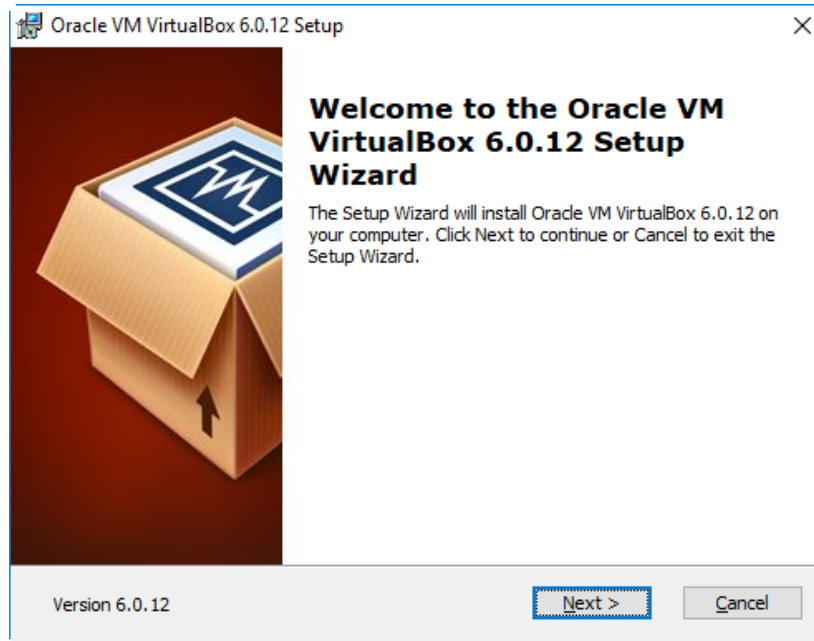
The version downloaded may differ from the version shown in this documentation (6.0.12). You also need to download the Extension Pack which is a separate download. Make sure the Extension Pack you download is the same version as your VirtualBox installer.

2. Launch the VirtualBox installer from Windows Explorer by double-clicking the self-extracting executable. Allow the installer to make changes to your computer, if so prompted.

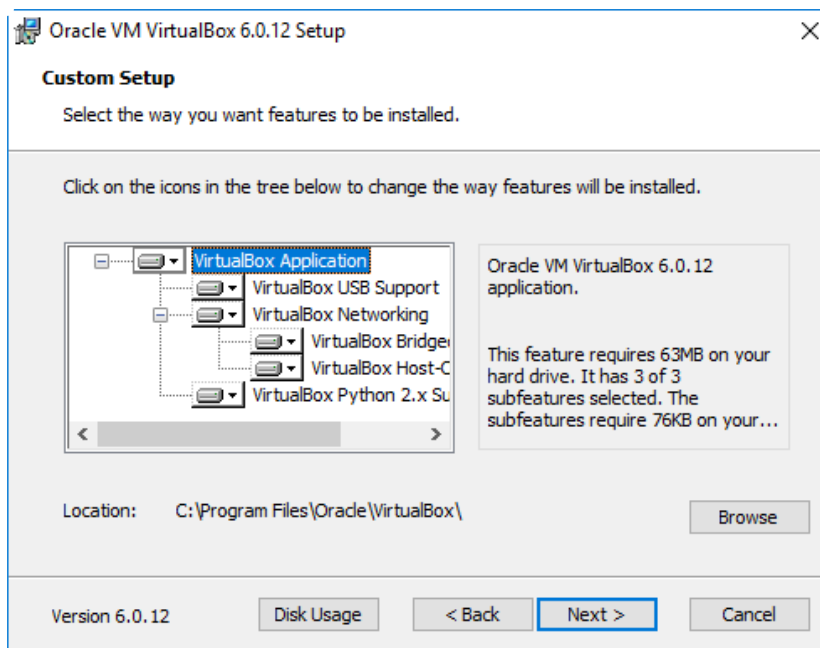
Name	Date modified	Type	Size
 VirtualBox-6.0.12-133076-Win.exe	10/14/2019 2:38 PM	Application	166,464 KB

VirtualBox Installer for Windows

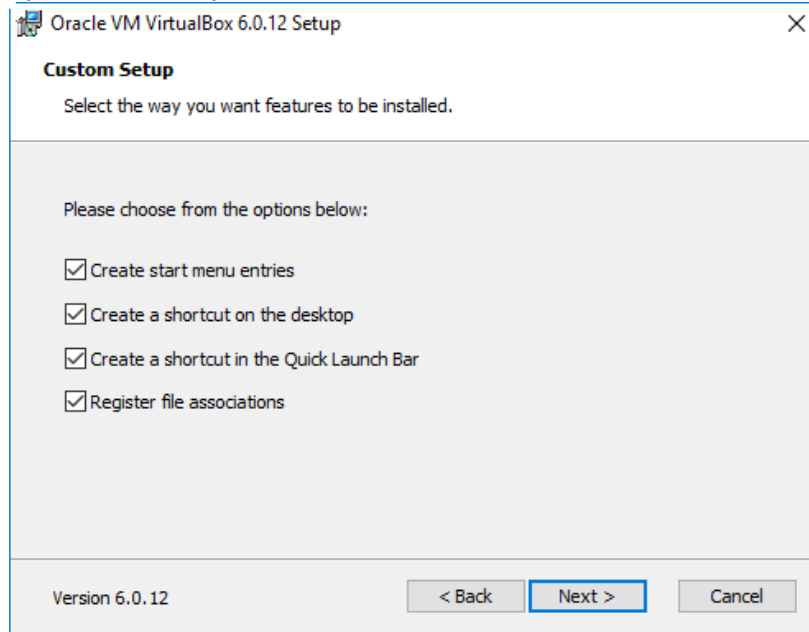
- Once the VirtualBox installation wizard appears, click the **Next** button.



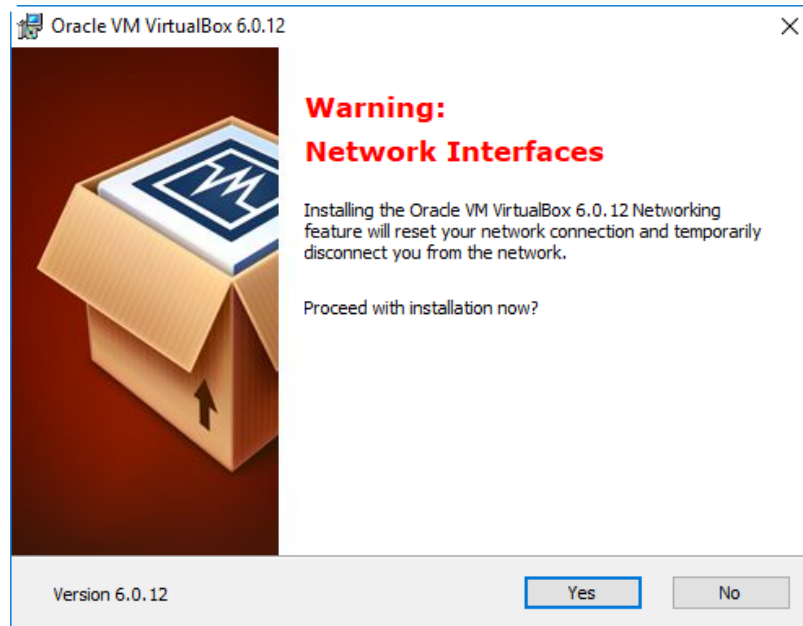
- You may accept all the installation defaults, although you may wish to change the installation location on your development platform using the **Browse** button. If the options are acceptable, click the **Next** button.



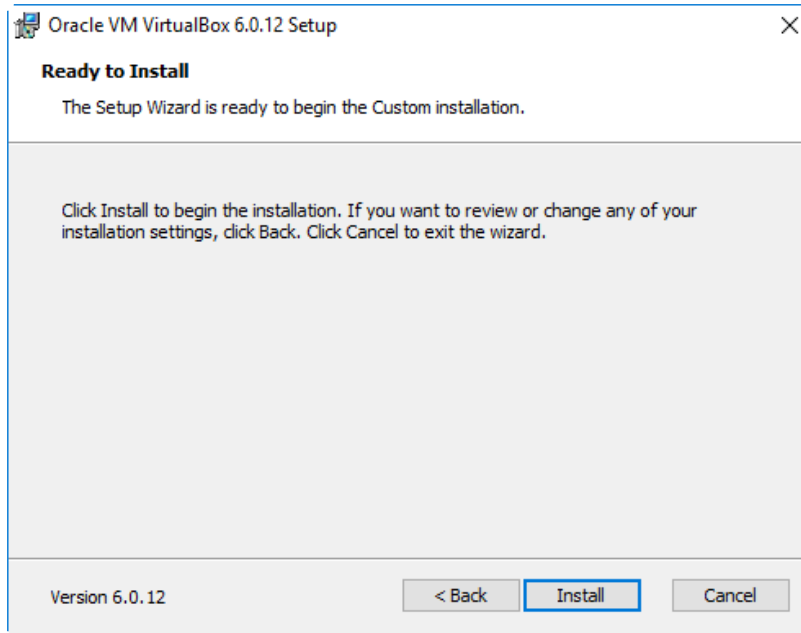
You may again accept the default options and click the **Next** button.



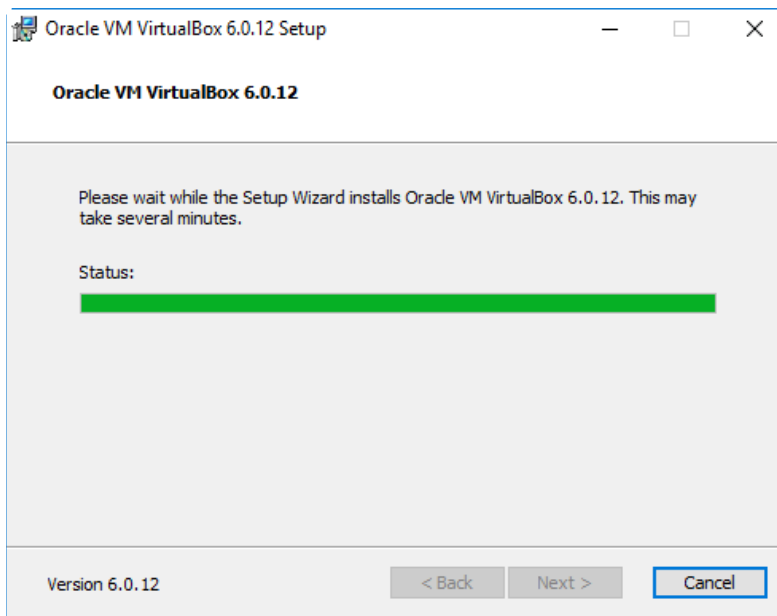
5. Click the **Yes** button to continue with the installation wizard.



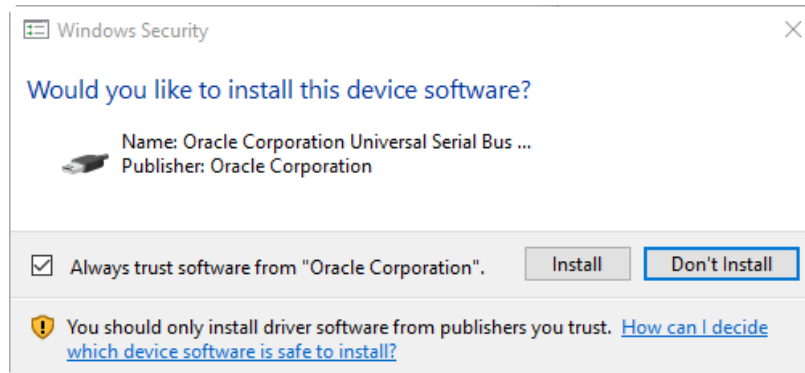
- Click the **Install** button to load VirtualBox to your development system.



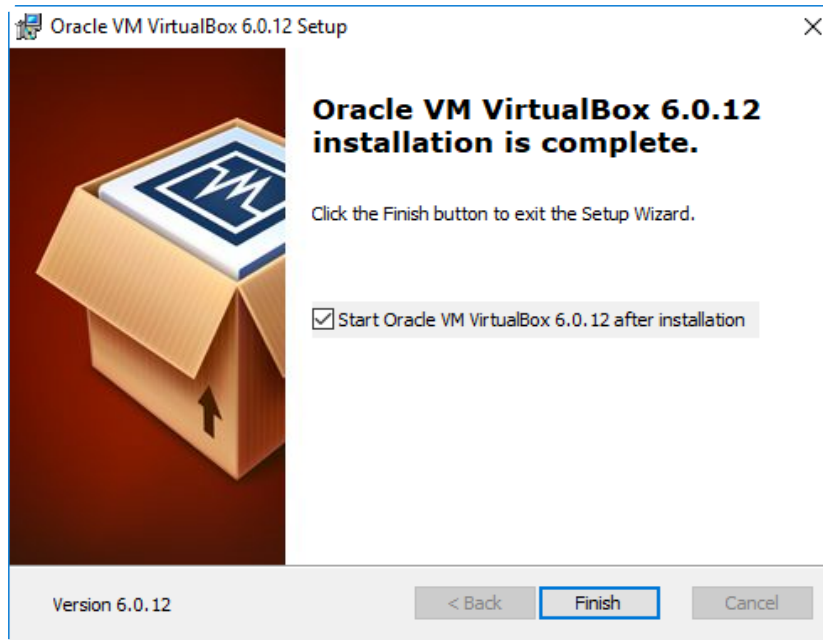
- During the installation you may receive prompts to authorize installation of various components. If prompted, allow the installer to make changes to your system, including installation of the USB interface and Network adapters.



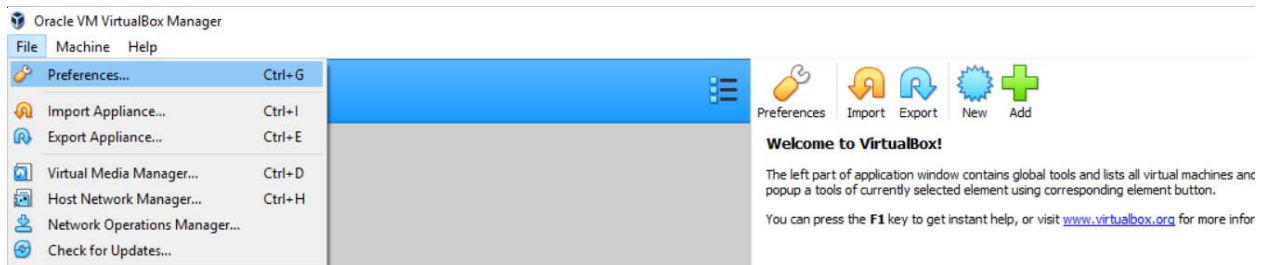
8. If you are asked to install the Oracle Corporation Universal Serial Bus device driver, or Oracle Corporation Network Adapters/Network Service, choose to install them



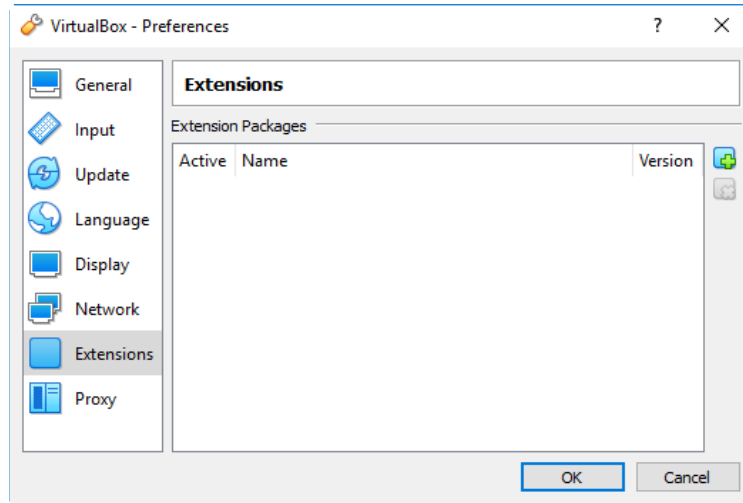
- Click the **Finish** button to complete the installation. Leave the checkbox enabled so VirtualBox will start after the installer finishes.




- Once VirtualBox starts (you can also start it from the Desktop shortcut, or the Windows Start button), the Extension Pack must be added. From the main menu, select **File > Preferences**.



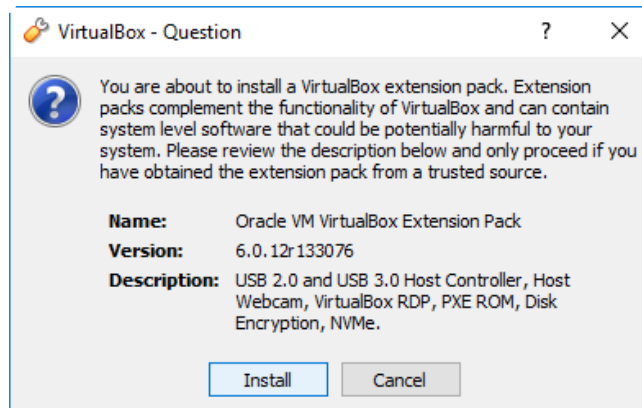
11. Select **Extensions**. Right-click in the *Extension Packages* whitespace box, and select **Add Package**.



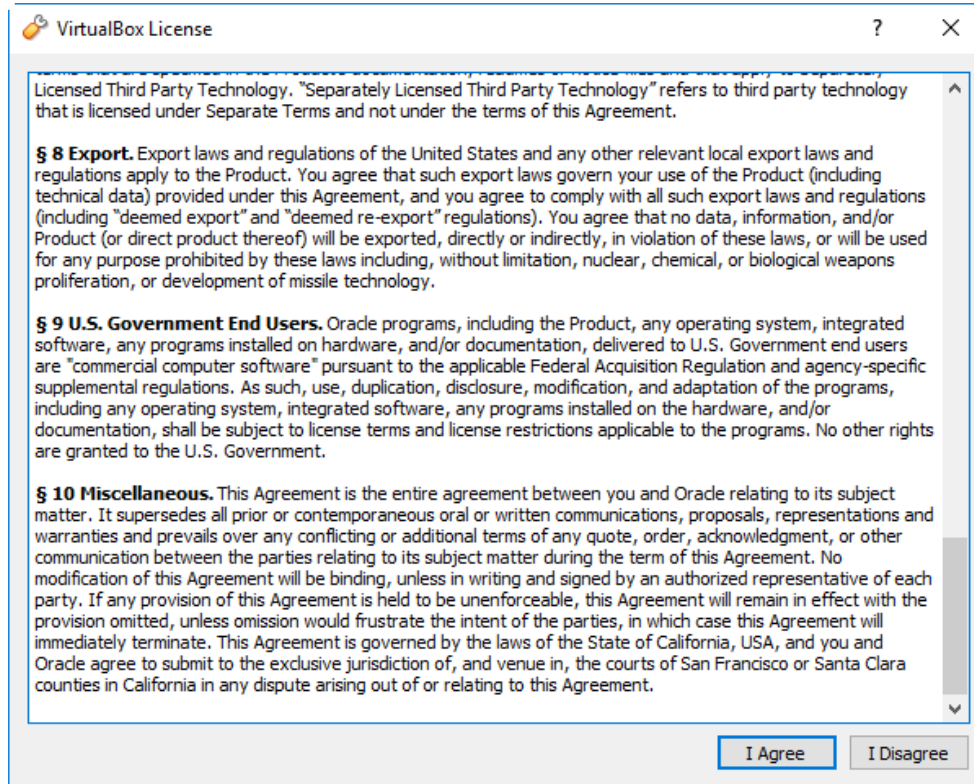
12. Browse to the location where you downloaded the VirtualBox Extension Pack compatible with your VirtualBox version. Select the Extension Pack and click the **Open** button.

Name	Date modified	Type	Size
 Oracle_VM_VirtualBox_Extension_Pack-6.0.12r133076.vbox-extpack	10/15/2019 6:34 AM	VirtualBox Extensi...	22,618 KB

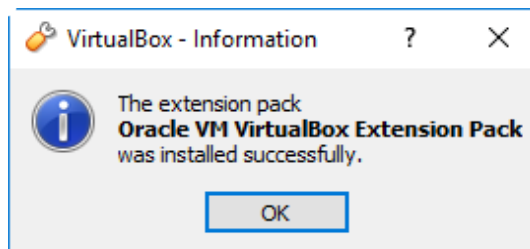
13. Click the **Install** button to add the VirtualBox Extension Pack.



14. Read the VirtualBox Extension Pack PUEL License to ensure you will not be in violation of the Oracle definition of Personal Use. See the *VirtualBox Licensing Frequently Asked Questions* for additional details. If you can accept the license conditions, scroll to the bottom of the agreement text box and click the **I Agree** button¹. If prompted, allow the installer to make changes to your development system.

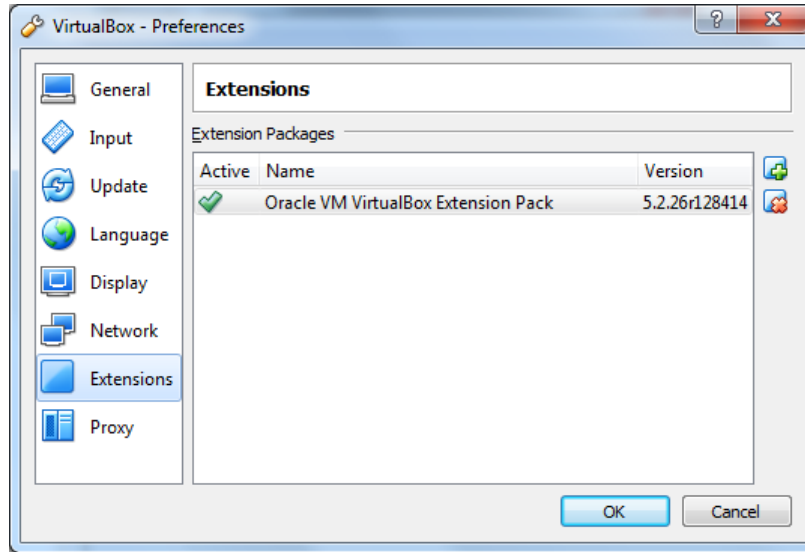


15. Click the **OK** button to complete the installation.



¹ If you must disagree, the installation will be terminated. You should either purchase a commercial license or uninstall VirtualBox from your host computer.

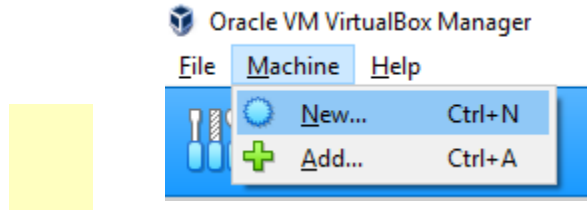
16. Click the **OK** button to return to VirtualBox.



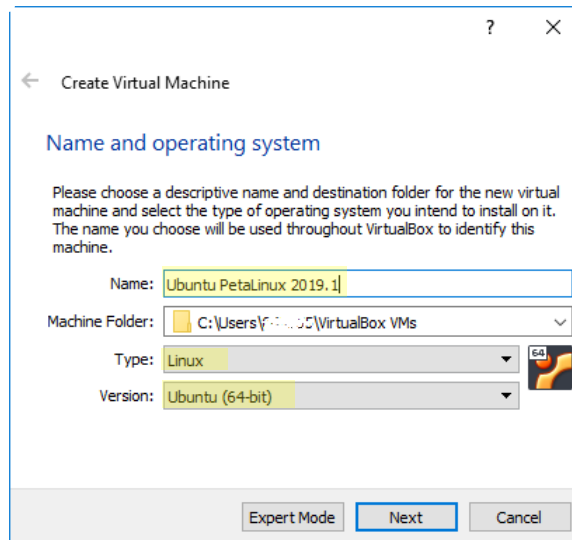
This completes the installation of VirtualBox on your host development system. VirtualBox is now ready to accept a new Virtual Machine.

Create a New Virtual Machine

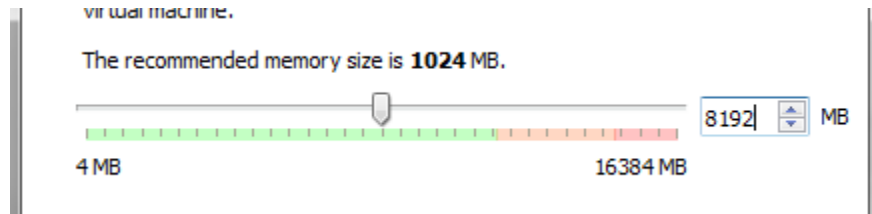
1. Launch Oracle VM VirtualBox Manager and click **Machine** → **New** icon at the upper left.



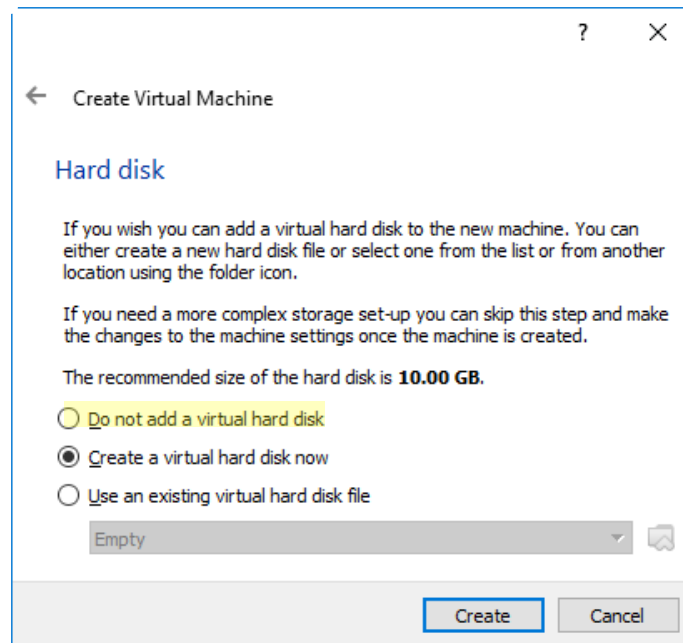
2. Select a descriptive name for the VM. Set the *Type* to **Linux** and the *Version* to one that corresponds to the OS you wish to install.
 - a. For Ubuntu, choose **Ubuntu (64-bit)**.



3. Select the amount of memory to be allocated to the Virtual Machine. Allocating more memory² will improve the VM performance, but you must leave sufficient memory available for your host system for all other concurrent processes. For a host system with 16 GB of RAM, a value of **8192** MB is recommended for the Virtual Machine, especially if you intend on using Xilinx SDSoC. You may wish to experiment with this value to optimize your performance as larger density target devices have **higher**² memory requirements. The memory can also be changed at any time even after installing the VM hosted OS. You will need to locate a balance of host and guest performance through properly balancing this. Of course the MORE memory you can provide the guest, WITHOUT causing issues for the host, the better your guest OS will run!



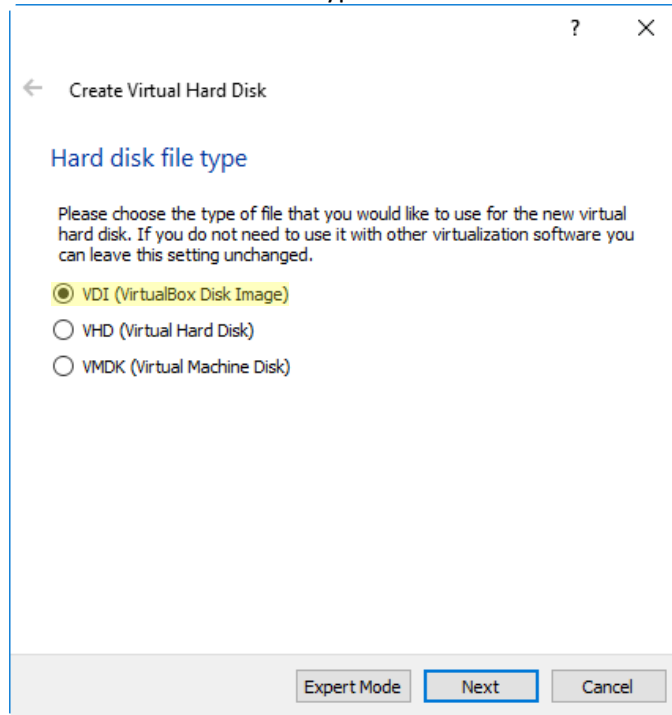
4. Click the **Create**³ button to accept the default file type for a VirtualBox Disk Image and allocate a virtual hard drive now.



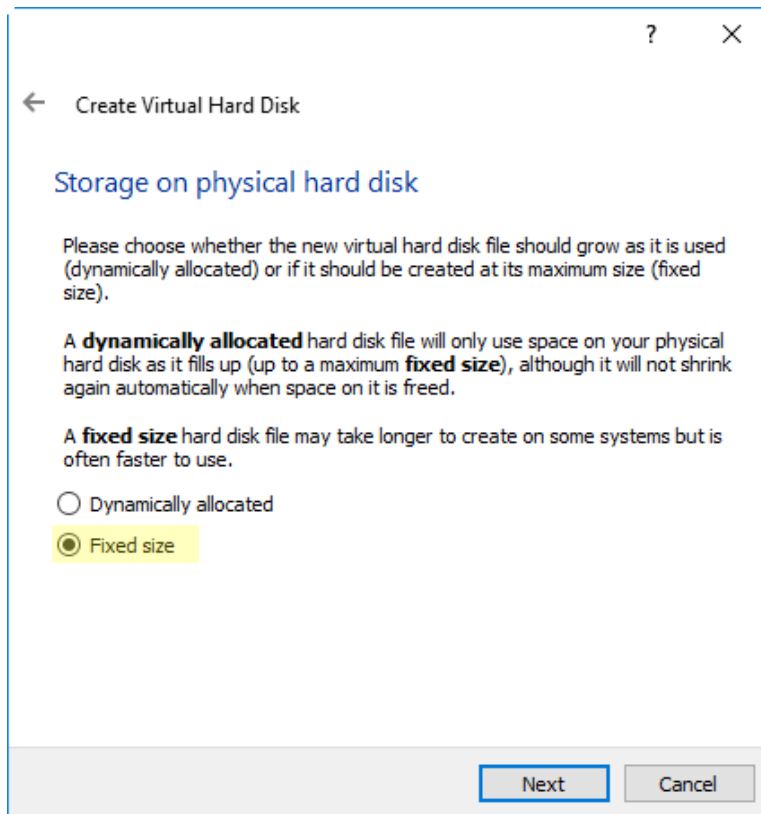
² You may need more memory if you intend to run Vivado with large Xilinx devices. Refer to <https://www.xilinx.com/products/design-tools/vivado/memory.html> for details selecting RAM for your application

³ If you are importing an existing Virtual Machine, click the "Use an existing..." button.

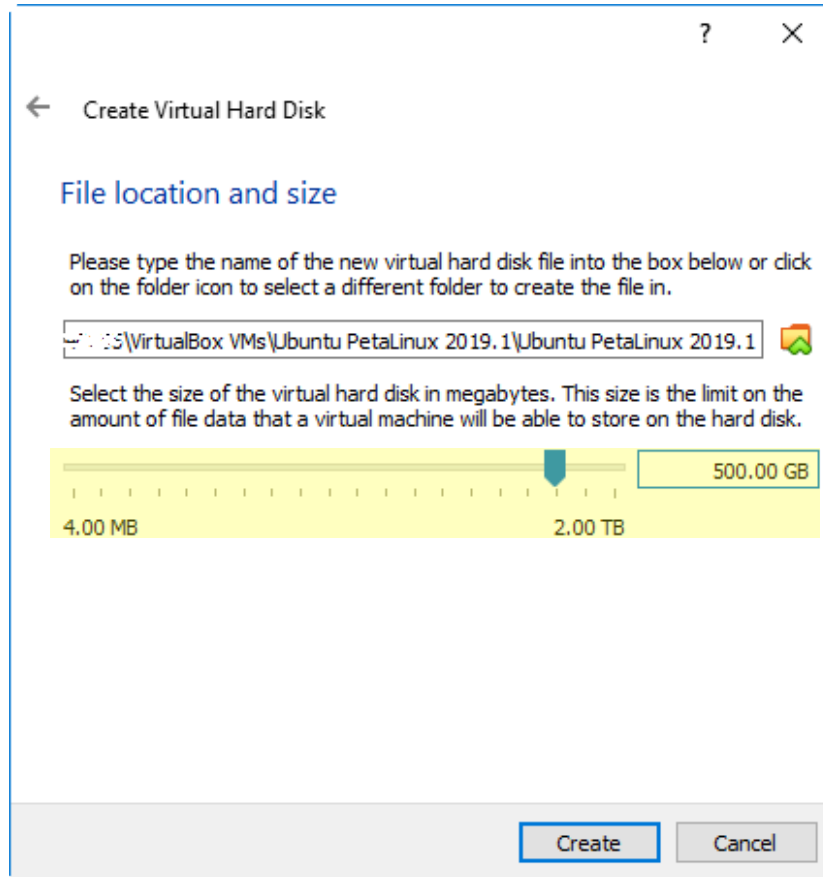
Select Next, leaving the defaults for Hard disk file type



5. Select **Fixed Size** for the physical storage on your host hard drive. This will improve overall performance of the Virtual Machine.

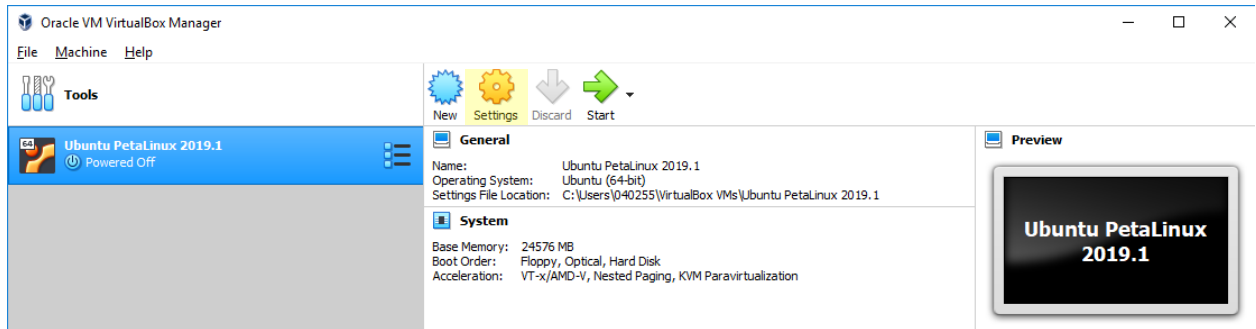


6. Select the name and location for the Virtual Machine within your host file system. The recommended size is **500.00** GB to accommodate the Xilinx tools. If this space is unavailable, 150.0 GB is sufficient if you plan to install the SDK (but and only a few of the Vivado tools). 375.0GB is sufficient for creating a PetaLinux based SDSoc platform. Note that it has been seen, if your host PC has a SSD, a dynamically allocated VDI does not appear to affect performance significantly. Click the **Create** button.

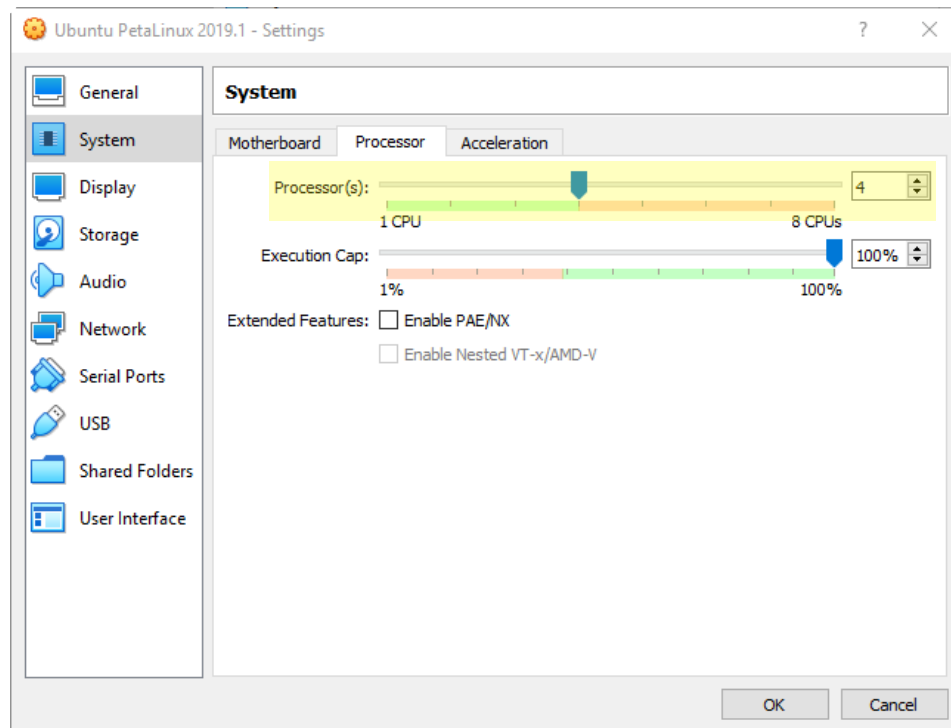


The Virtual Hard Disk may take a few minutes to create and initialize on your host file system.

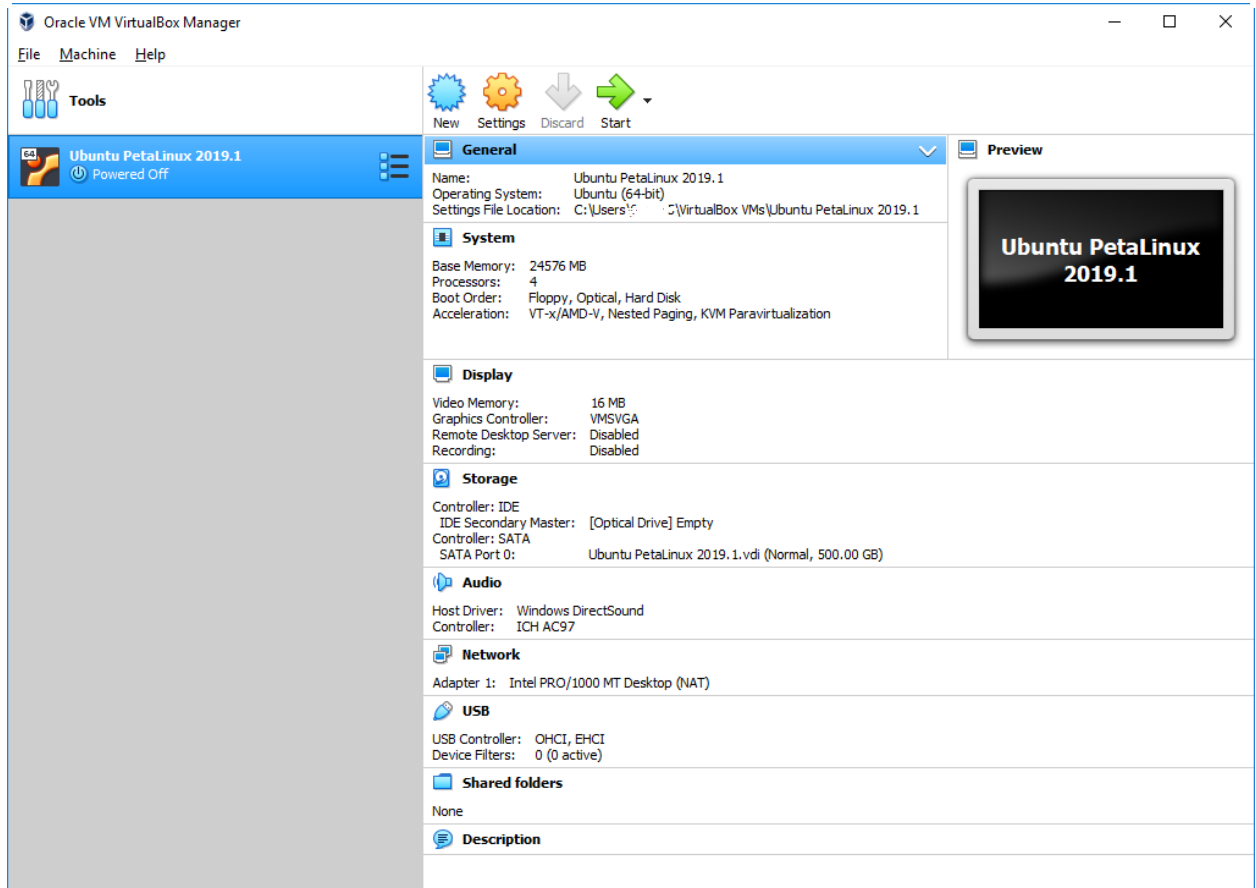
- Once the Virtual Disk completes, select the newly created VM and click on the **Settings** button to open the Settings dialog specific to the VM instance.



- Under the **System** blade and the **Processor** tab, select the number of logical CPU cores to be allocated to the Virtual Machine. Allocating more CPU cores will improve the VM performance, but you must leave sufficient CPU cores available for your host system for all other concurrent processes. For a host system with 8 CPU cores, a value of 4 CPU cores is recommended for the Virtual Machine, especially if you intend on using Xilinx SDSoC. You may wish to experiment with this value to optimize your performance as larger density target devices have **higher²** computation requirements for design placement. The memory can also be changed at any time even after installing the VM Guest OS as long as the Guest is shut down and the VM powered off. You will need to locate a balance of host and guest performance through properly balancing this.



- Once the processor allocation has been modified in the above step, your VM is ready to accept an operating system.



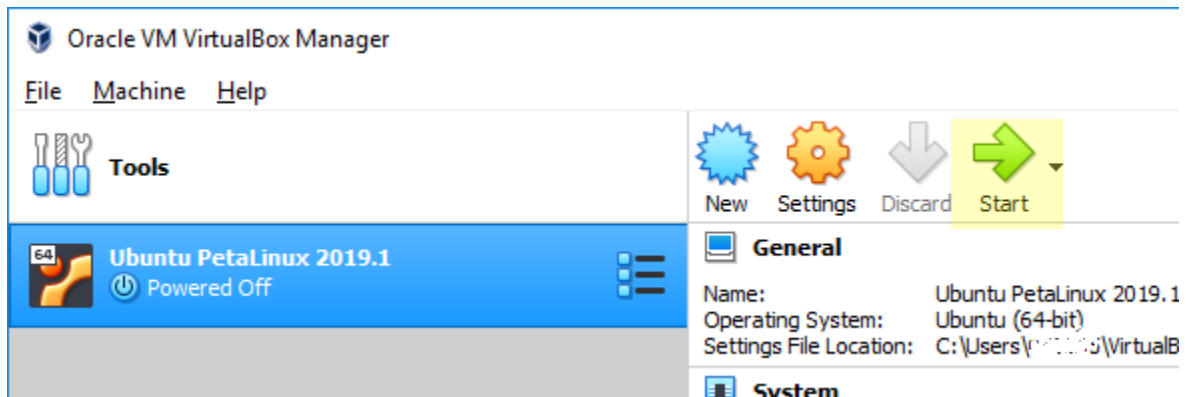
Install the Ubuntu 18.04 Operating System


To perform the steps in this section, you will need to download a bootable OS image in .iso format to your host system. While Ubuntu 19.04 is the latest, Ubuntu 18.04.1 is the recommended **version**⁴ (ubuntu-18.04.1-desktop-amd64.iso). We have also seen other sub versions of Ubuntu 18.04 work with this. The Ubuntu images can be downloaded from:

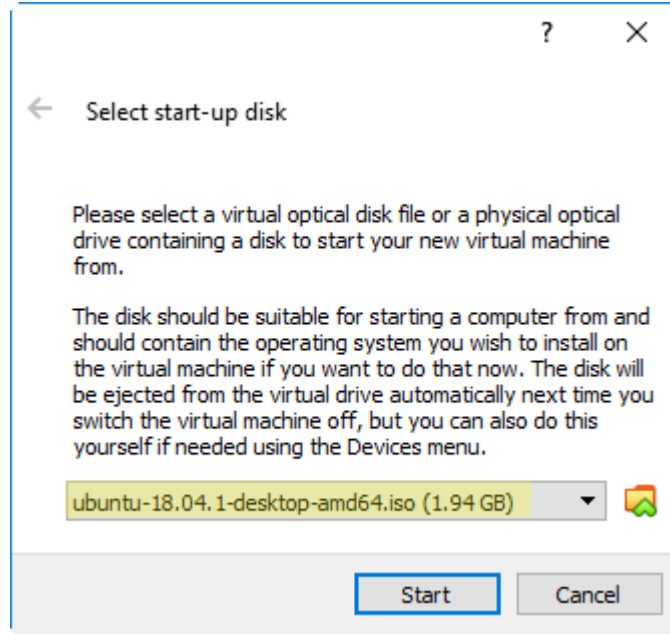
<http://www.ubuntu.com/download/desktop>

Repeat the steps outlined in **Create a New Virtual Machine**, entering Ubuntu as the **Name** of the VM. Once the Virtual Disk completes, your VM is ready to accept an operating system.

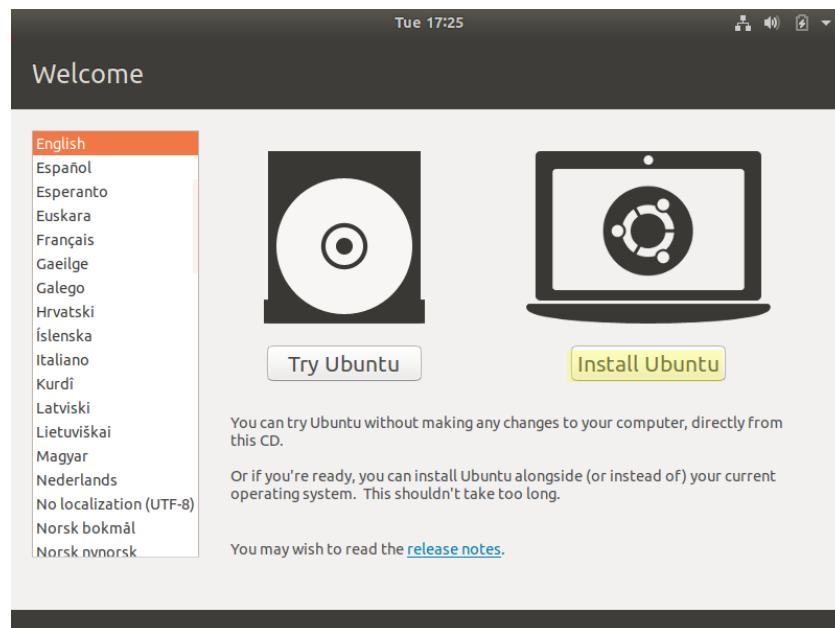
1. Launch VirtualBox (if necessary) and select the VM you wish to start in the left-hand panel. Click the **Start** button to execute the VM.



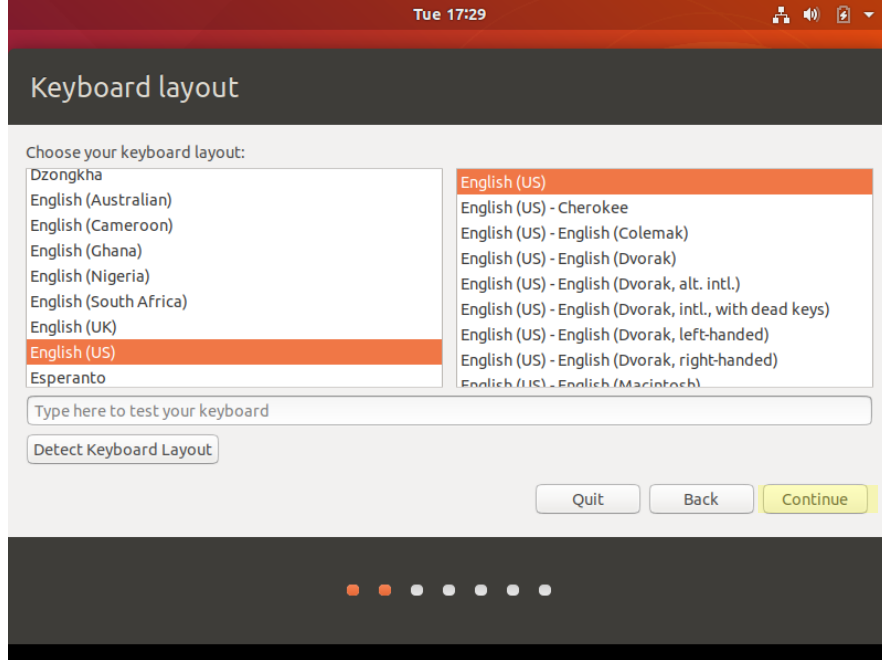
2. Select the **Browse**  icon to locate the .iso image for the OS you wish to install on your Virtual Machine. Click the **Start** button to begin.



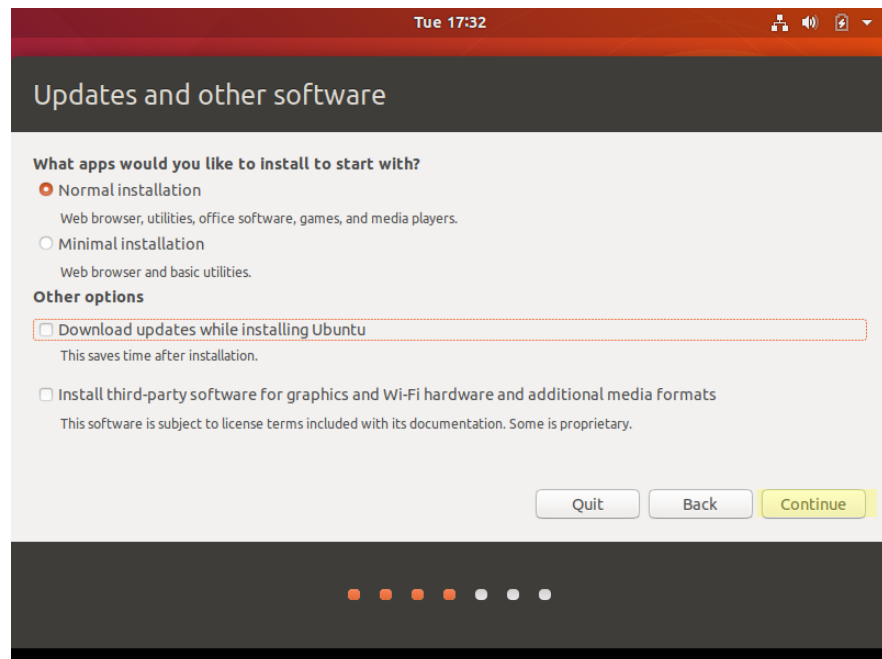
- When the Install Welcome screen appears, select English and click the **Install Ubuntu** button.



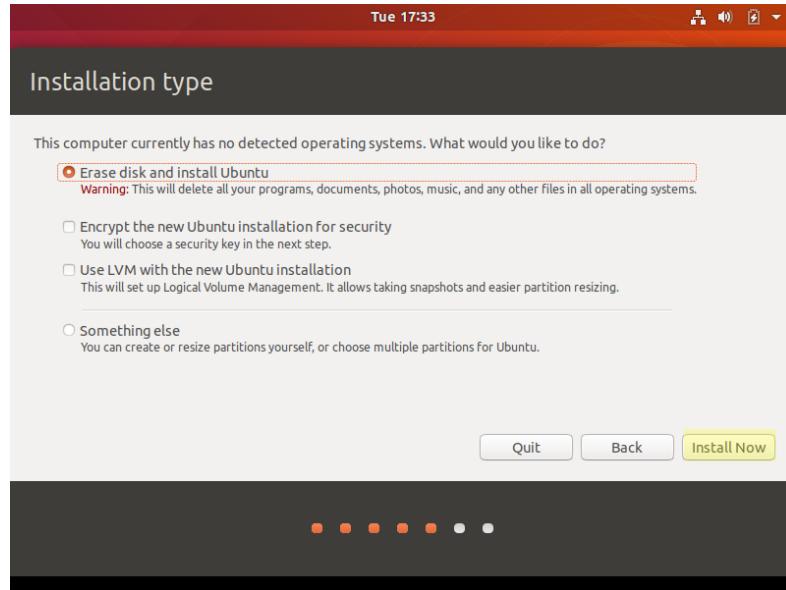
4. Select your preferred keyboard layout option. The default displays as English (US). Click the **Continue** button.



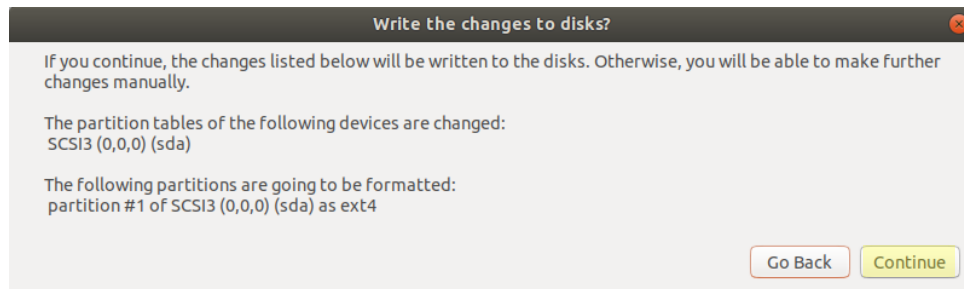
5. The installer shows requirements for installation. Uncheck Download updates while installing Ubuntu in order to speed the install. Click the **Continue** button.



- The installer displays various installation types. The default displays as **Erase disk and install Ubuntu**. Click the **Install Now** button.



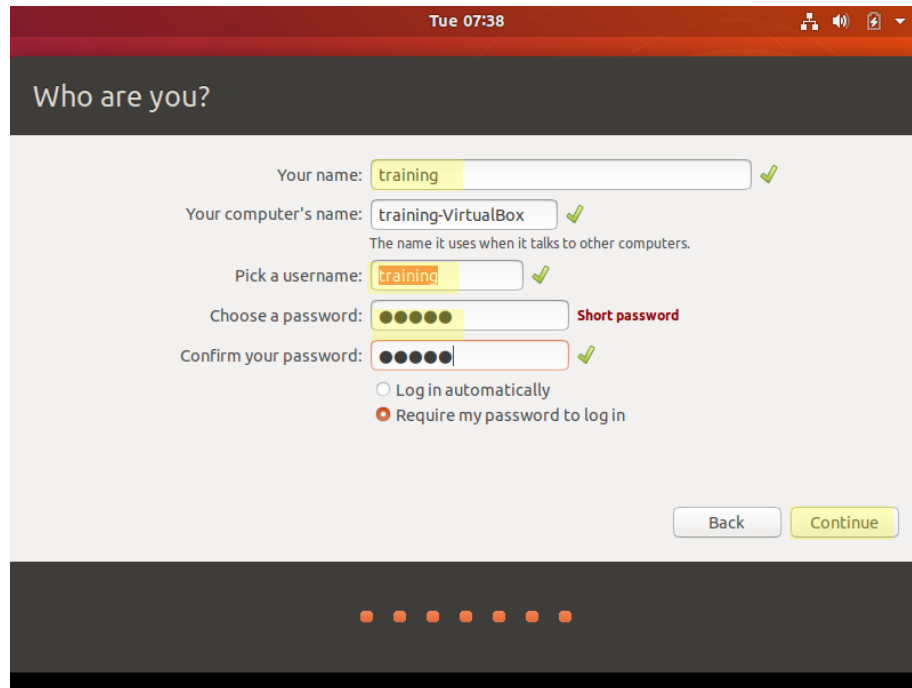
As this is a new installation, we want all changes written to the disks. Click the **Continue** button.



- Select your time zone and click the **Continue** button.



8. Enter the primary user name for the Virtual Machine. In this case, create a default user name *training*. The system will auto-populate the computer name and username. Enter and confirm a password. Click the **Continue** button.

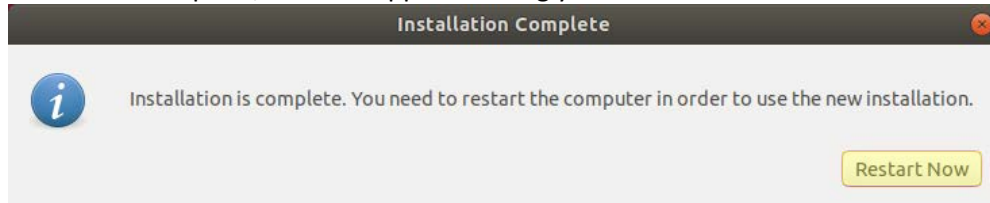


The screenshot shows a window titled "Who are you?" with a dark red header bar. The window contains the following fields and options:

- Your name:** A text input field containing "training" with a green checkmark to its right.
- Your computer's name:** A text input field containing "training-VirtualBox" with a green checkmark to its right. Below this field is the text "The name it uses when it talks to other computers."
- Pick a username:** A text input field containing "training" with a green checkmark to its right.
- Choose a password:** A password input field with five green dots and a red "Short password" warning to its right.
- Confirm your password:** A password input field with five black dots and a green checkmark to its right.
- Log in options:** Two radio buttons: "Log in automatically" (unselected) and "Require my password to log in" (selected).
- Buttons:** "Back" and "Continue" buttons at the bottom right.

At the bottom of the window, there is a dark grey bar with six orange dots in the center.

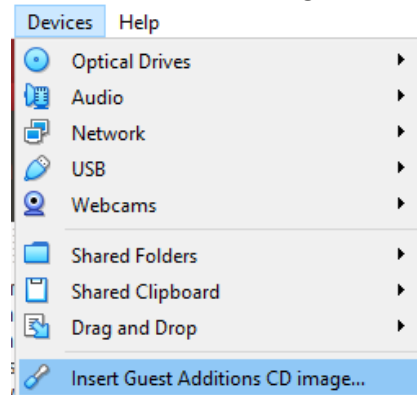
9. The installation displays a Welcome graphic and proceeds with installation. When the installation is complete, a screen appears asking you to restart. Click the **Restart Now** button.



If the Restart appears to “freeze”, you can force a reboot manually:

- From the main VirtualBox menu, select **File | Close**.
- In the *Close Virtual Machine* dialog, select **Power off the machine** and click the **OK** button.
- In the Oracle VM VirtualBox Manager, select your Virtual Machine and click the **Start** button.

If Ubuntu requests that you remove the installation media and press enter to continue, it is easiest to select Devices-> Insert Guest Editions CD image...



10. Click through the “What’s new in Ubuntu

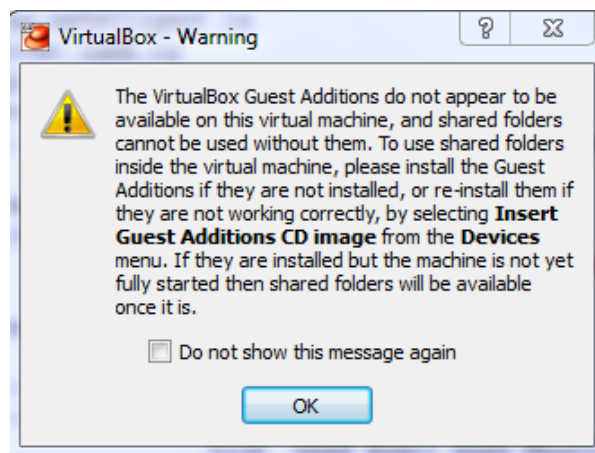


Configuring VirtualBox for Better Ease of Use

Configuring VirtualBox on a supported Linux system should be straightforward, but depending on the precise hardware configuration of your development system, there could be some challenges to create an optimal environment. This section outlines a few of the common issues that may be encountered along with some suggested workarounds to help increase the ease of use for the environment.

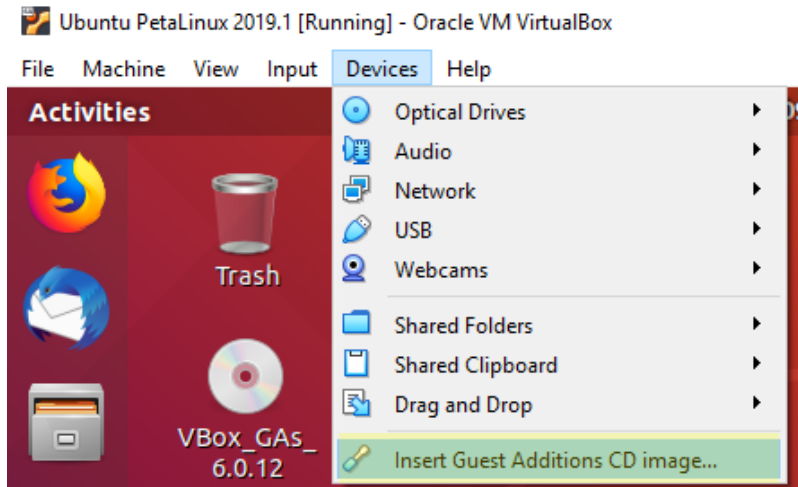
VirtualBox Guest Additions Installation (File Sharing)

1. It was found with the latest versions of VirtualBox, we did not have to install the Guest Editions. If you do not get the message from the following step, continue to section “VirtualBox Shared Folders”
2. The use of shared folders allows for easy transfer of files between the host and guest systems. To use the shared file facility of VirtualBox, you must install the Guest Additions (VBoxGuestAdditions.iso). If you attempt to use the shared folder facility without the Additions, you will receive the following error message.

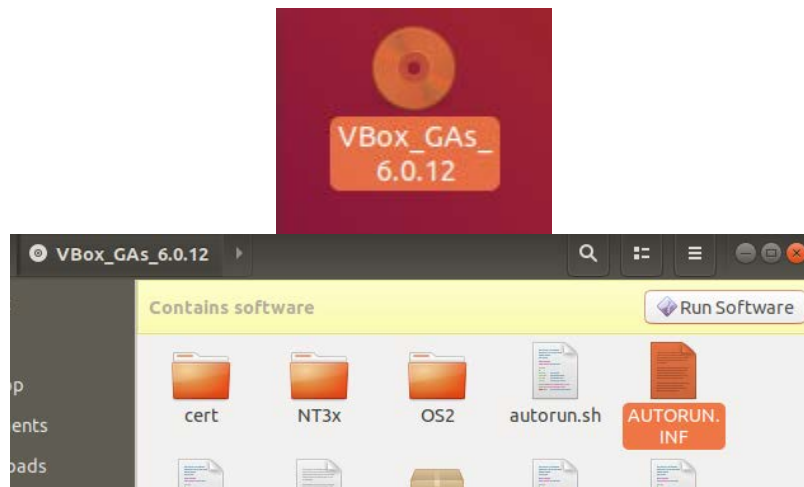


After Guest Additions are installed, you can move the cursor between the Virtual Machine and the host OS without having to use the *Right Ctrl* key to recapture the cursor in the host.

- a. If the VBox_Gas_6.0.12 icon is NOT on the desktop, from the **Devices** menu, select **Insert Guest Additions CD image...**



- b. If you need to, double click the icon, then when prompted, click the **Run** button to execute the installation.



- c. Note, if this does not auto-run
 - i. open a terminal
 - ii. `cd /media/training/Vbox_Gas_6.0.12`
 - iii. `sudo ./autorun.sh`
Enter your root password
 - iv. Skip to the “VirtualBox Shared Folders” section
- d. The Guest Additions should install and verify with no failures.
 - i. If this does fail, reboot and attempt the same procedure to install the Guest Additions

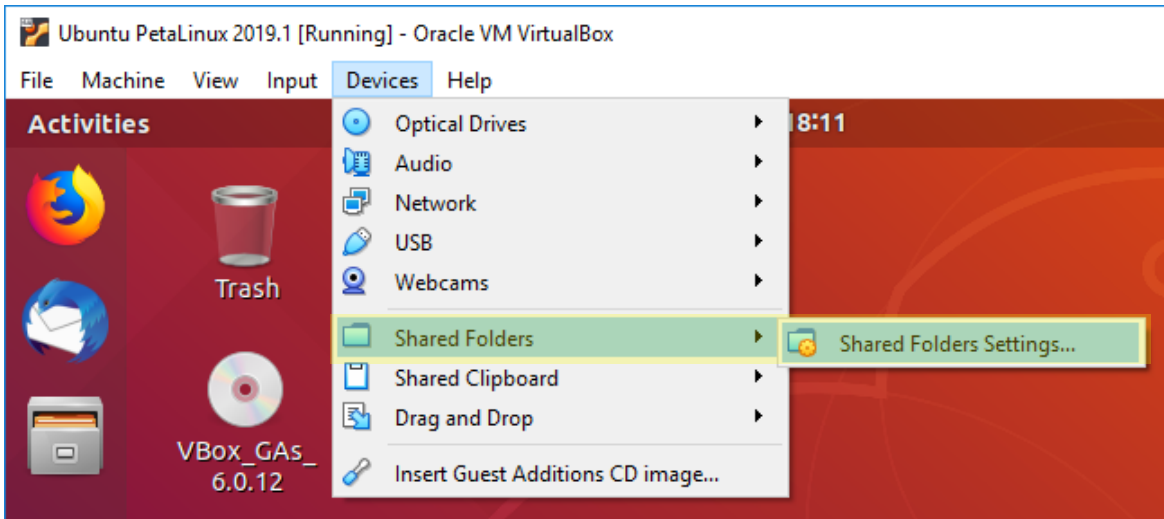
- e. If the installation states that it needs to be reset before kernel modifications can occur, do so and then come back and continue at the “VirtualBox Shared Folders” section
- f. Press the **Enter** key to close the installation window. Be sure to restart Linux to ensure that the Guest Additions is started properly before moving on to a later section of this guide.

```
This system is currently not set up to build kernel modules.  
Please install the gcc make perl packages from your distribution.  
VirtualBox Guest Additions: Running kernel modules will not be replaced until  
the system is restarted  
Press Return to close this window...  
█
```

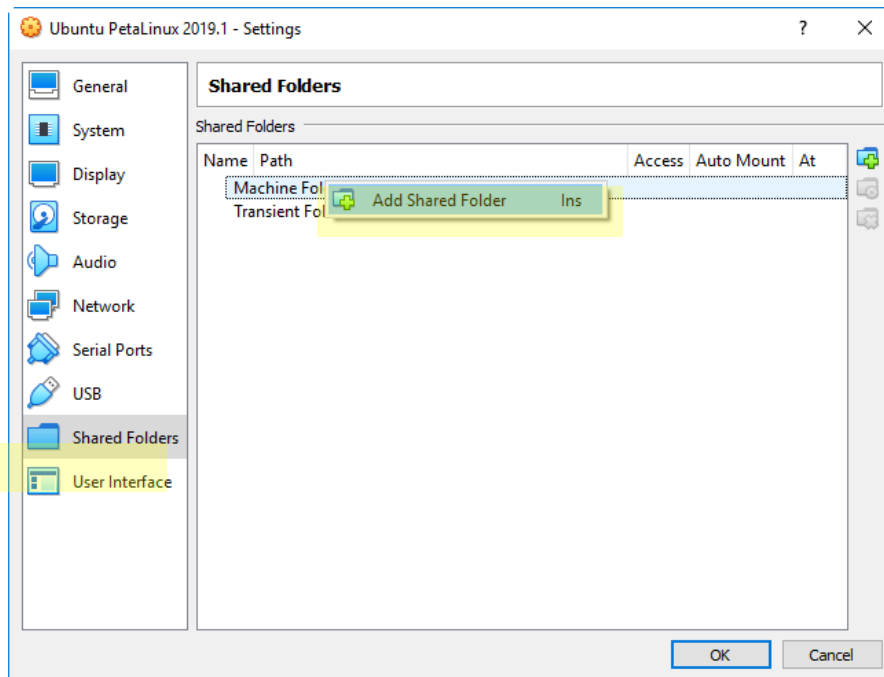
VirtualBox Shared Folders

Once Guest Additions have been installed, you must select a folder to share between the host and guest systems. This folder is used to transfer files to/from the Virtual Machine and the Host system.

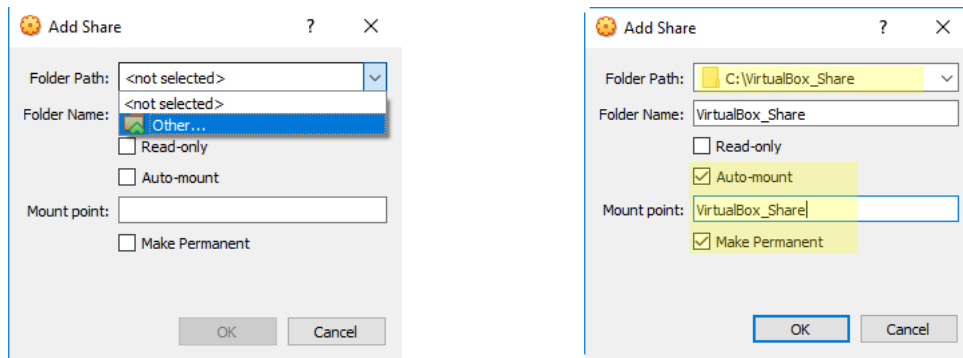
1. From the VirtualBox main menu, select **Devices > Shared Folder > Shared Folders Settings...**



2. Right-click *Machine Folders* and select **Add Shared Folder**.



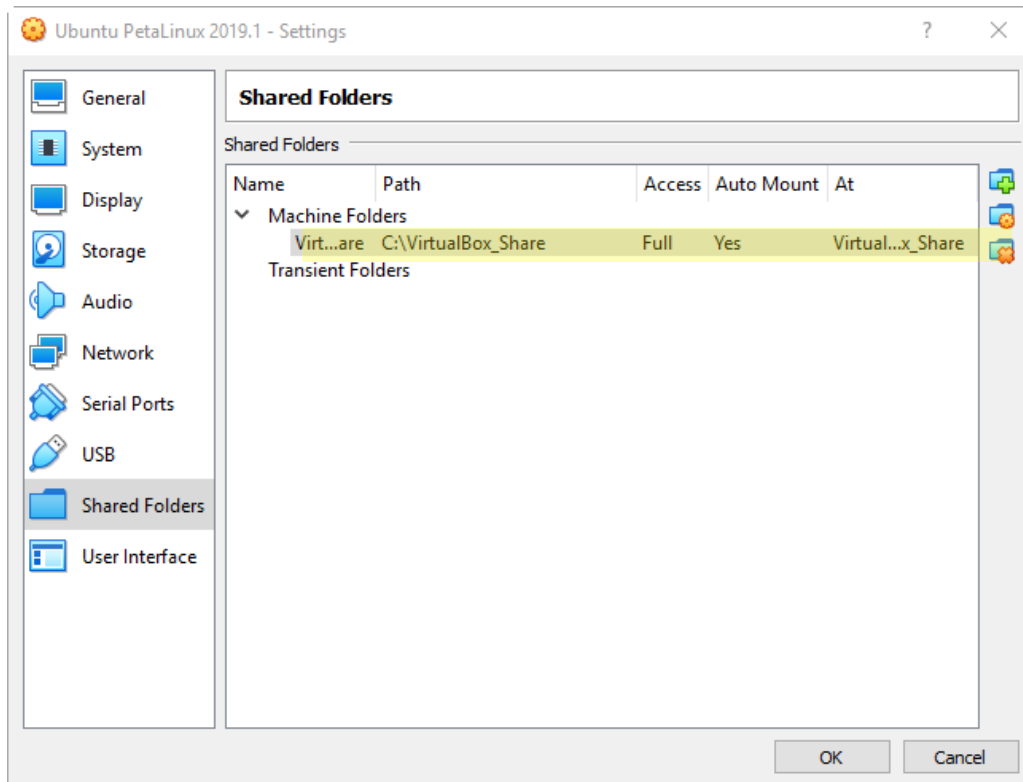
- In the *Folder Path* box, click the dropdown arrow on the right. Select the **Other** entry to open a Windows Explorer pane. Browse to the location in Windows where you want to set up a shared folder and click **Select Folder** in the Explorer pane. Click the checkboxes for **Auto-mount** and **Make Permanent**. Click the **OK** button. For consistency with Lab and training materials, it is suggested to use C:\VirtualBox_Share and VirtualBox_Share as shown in the image below



- The location of the shared folder in Windows is shown in the Path column. The folder will Auto-mount and Full Access is allowed. The corresponding folder in the Linux VM is /media/sf_<Windows Folder Name>. In the example shown, this corresponds to:

/media/sf_VirtualBox_Share

Click the **OK** button to close the panel.



5. Shared folders are only available to user accounts that are also members of the group *vboxsf*. This means the user account created earlier must be added to this group. This can only be done in Ubuntu 16.04 from the command line. To view available groups and members, open a Terminal window but selecting the Dash and then searching for Terminal. In a Terminal window enter one or more of the following commands.

This command lists all groups:

```
$ getent group
```

This command lists a specific group named *vboxsf*:

```
$ getent group vboxsf
```

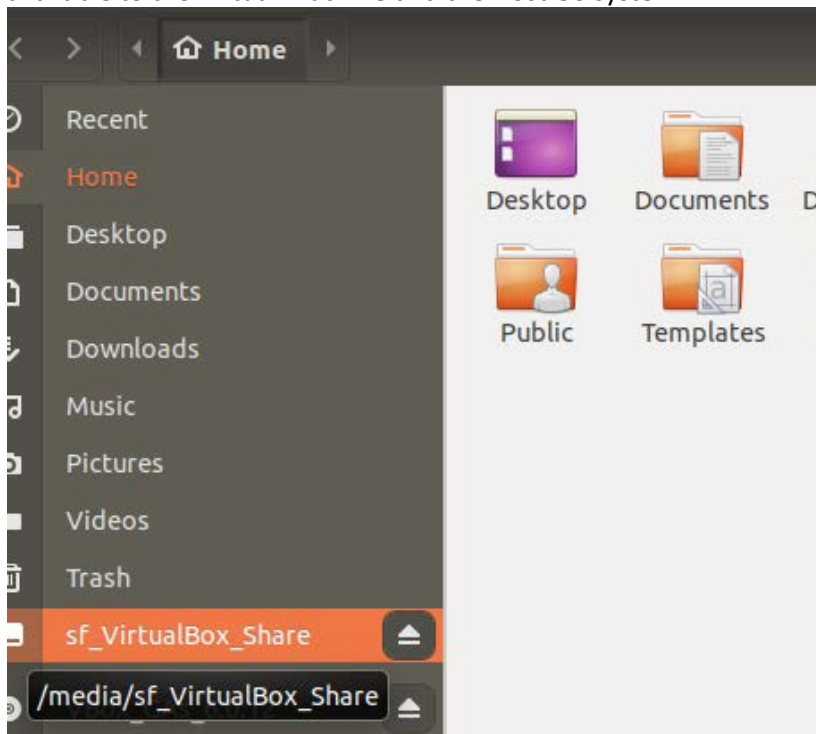
To add an existing user to an existing group, in a Terminal enter the following command:

```
$ sudo usermod -a -G vboxsf <current username>
```

6. Reboot the Virtual Machine.

7. The selected user name will belong to the *vboxsf* group on the next login. To access the shared folder from the Virtual machine, browse to: */media/sf_<sharename>*

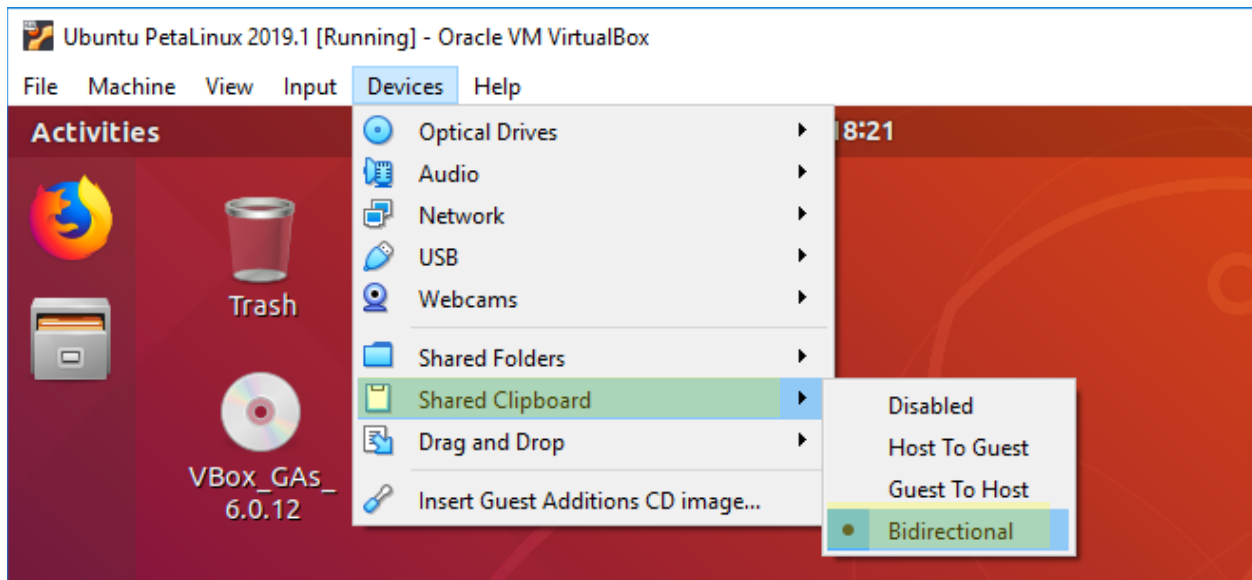
In this example, the folder in Windows is named *VirtualBox_Share*, so the *<sharename>* in Linux is *sf_VirtualBox_Share*, automatically mounted in the */media* folder. Any files in this folder are available to the Virtual Machine and the Host OS system.



VirtualBox Shared Clipboard

Once the Guest Additions have been installed, you can enable the Shared Clipboard which will make it easier to copy and paste text strings from tutorial documents from the host system to the guest systems. This is very useful for later tutorials where it is desirable to copy command prompt instructions verbatim from the tutorial guide document directly into the command prompt of the guest system.

1. From the VirtualBox main menu, select **Devices > Shared Clipboard > Bidirectional**



Network Bridging

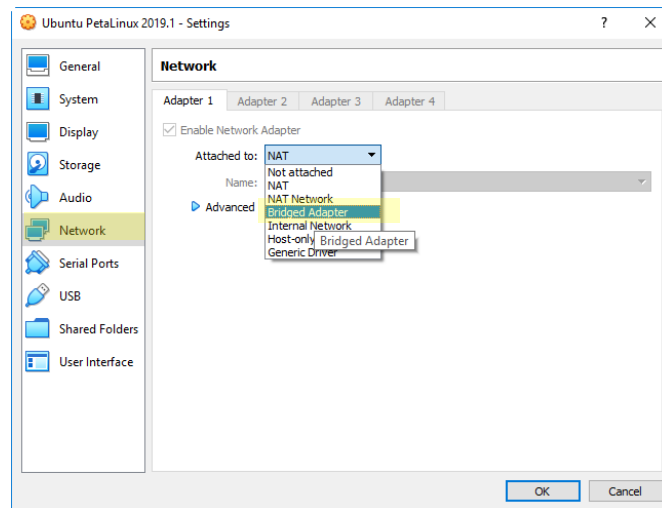
When VirtualBox is installed with its default options, the networking is set up to use Network Address Translation (NAT). This allows your Virtual Machine complete outgoing access to your LAN and/or the Internet, but it assigns an internal IP address that may not be compatible with your LAN and the IP address of your host system.

If you would like your Virtual Machine to accept an address from a local DHCP server, you can change the default network type to Bridged. This will make the Virtual Machine available to any other connected device on the same subnet on your LAN.

1. From the main VirtualBox menu in a running Virtual Machine, click on the **Settings** button. If the button is not visible, select **Machine > Settings**.



2. Select the Network entry in the left panel. Select the tab for your NIC (typically **Adapter 1**) and expand the dropdown menu for the *Attached to* field.



3. Select **Bridged Adapter** from the dropdown menu and click the **OK** button to save the changes. Wait a few seconds for your Virtual Machine to request an address from the local DHCP server. Once complete, the VM will now have an address on your local subnet, accessible to all devices on your LAN.

If you do not receive a new IP address after a minute, stop and restart the Ethernet service.

Configuring Ubuntu 18.04

Configuring Ubuntu on a supported Virtual Machine should be straightforward, but there could be some additional steps needed to create an optimal environment. This section outlines a few of the common issues that may be encountered.

Set a root user password

By default, Ubuntu does not set a password for the root user. You can do this by simply invoking the **sudo passwd** command. You supply your own user password, then set the root user password.

```
sudo passwd  
<Enter user password>  
<Enter new root password>  
<Confirm new root password>
```

```
training@training-VirtualBox:~$ sudo passwd  
[sudo] password for training:  
Enter new UNIX password:  
Retype new UNIX password:  
passwd: password updated successfully
```

From this point forward, you will be able to precede a command with **sudo** to obtain root authority.

Change the default shell to bash for PetaLinux

If you intend to use the PetaLinux tool-chain under Ubuntu, you will find that it requires the bash shell as the default to execute correctly. By default, Ubuntu uses the dash shell, which is an extension of the bash shell with a few additional features and optimized for faster execution. Unfortunately, the dash shell is not compatible with the current PetaLinux tool-chain.

A description for the dash shell and its potential issues can be found here:

<https://wiki.ubuntu.com/DashAsBinSh>

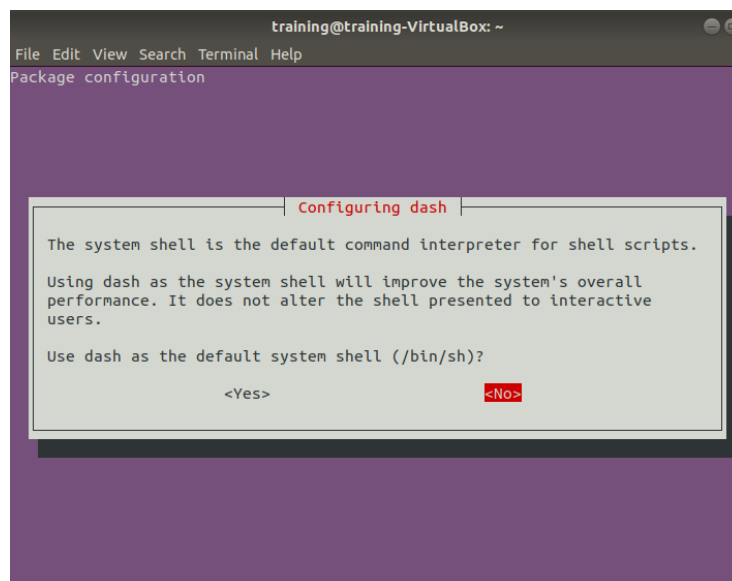
1. Most distributions use `/bin/sh` as a symbolic link to points to the actual default shell. Under your Ubuntu environment, to determine the current shell, enter:

```
$ ls -l /bin/sh
```

2. To change the default shell for all terminal windows, enter:

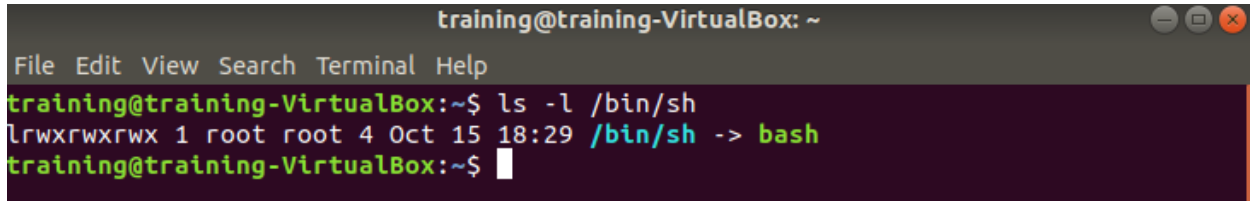
```
$ sudo dpkg-reconfigure dash
```

3. Select the option to remove dash as the default shell when prompted.



4. When the change is complete, close all open Terminal windows and open a new Terminal.

5. Verify the default shell is bash using the commands shown previously.

A terminal window titled "training@training-VirtualBox: ~" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command "ls -l /bin/sh" being executed, resulting in the output "lrwxrwxrwx 1 root root 4 Oct 15 18:29 /bin/sh -> bash". The prompt "training@training-VirtualBox:~\$" is shown before and after the command.

```
training@training-VirtualBox: ~
File Edit View Search Terminal Help
training@training-VirtualBox:~$ ls -l /bin/sh
lrwxrwxrwx 1 root root 4 Oct 15 18:29 /bin/sh -> bash
training@training-VirtualBox:~$
```

Xilinx Vivado/SDK Installation

Installing Vivado/SDK tools on a supported Linux system should be straightforward, but depending on the precise configuration of your development system, there could be some challenges to create an optimal environment. This section outlines a few of the common issues that may be encountered.

To perform the steps in this section, you need to download a tar.gz image to your host system. The most recent images can be downloaded from: <http://www.xilinx.com/support/download.html>.

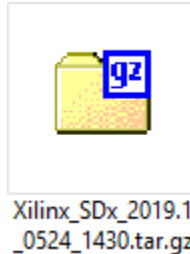
NOTE: Once the download is complete, you may want to verify there is an md5sum utility installed by default. Open a terminal window and enter the following:

```
$ md5sum -b <path to your compressed-Vivado-tar.gz file>
```

The valid checksums for the different download archives are available from the Xilinx download site.

Install Vivado in the VirtualBox Linux VM

1. Copy the All-OS version of the compressed Vivado or SDSoc installer from your host system to your VM desktop from the file share. (2019.1 SDx installer shown below)



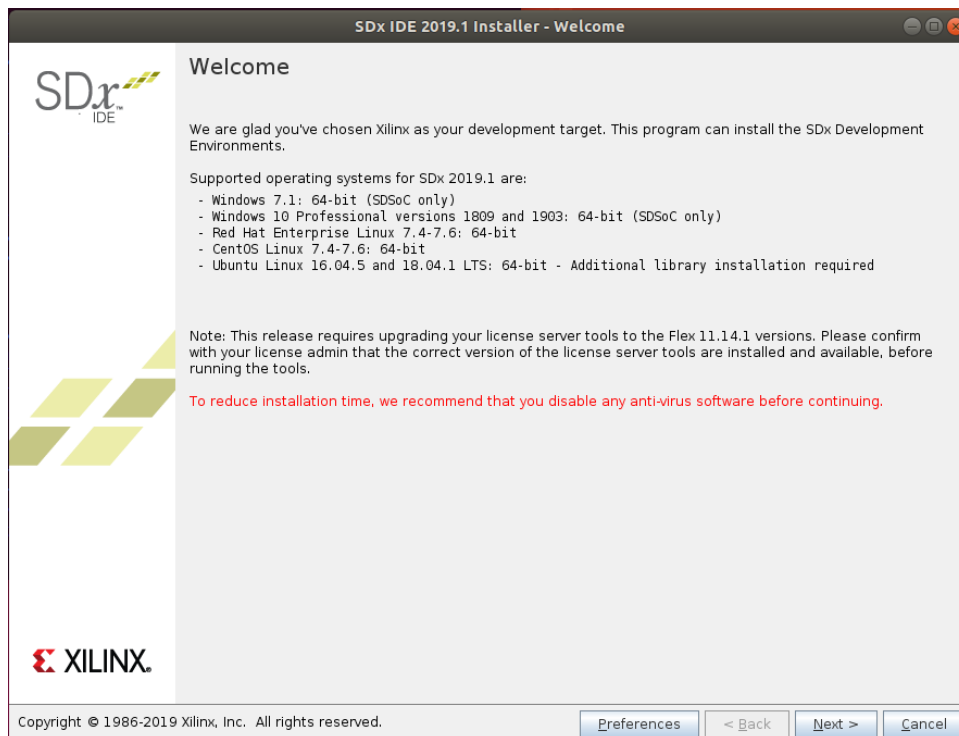
2. Open a terminal window and decompress the installer on your desktop. It will create a new folder automatically in the current directory of your Terminal window.

```
$ tar -xvzf ~/Desktop/<Compressed-Vivado-Installer-Name>
```

3. Change into the new folder and execute the installer setup script. You will need root privilege to install into the default directory of **/tools/Xilinx** (preferred).

```
$ cd ./Xilinx_Vivado_SDK_2019.1_0524_1430/  
-OR-  
$ cd ./Xilinx_Vivado_SDx_2019.1_0524_1430/  
$ sudo ./xsetup
```

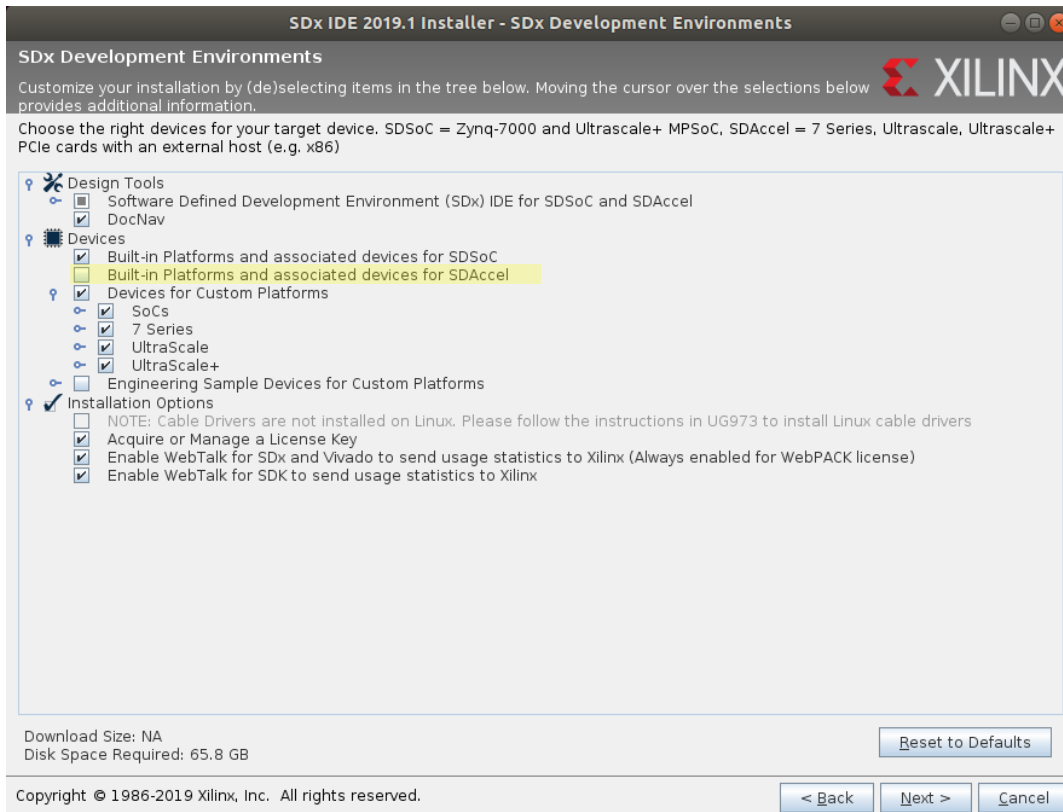
4. The Vivado 2019.1 Installer lists the operating systems officially supported by Xilinx. The tools may run on other Linux distributions and versions, but there will likely be some manual configuration required. Installation on unsupported systems is beyond the scope of these notes.



5. Follow the instructions as shown in the Installer GUI.

Note: If installing Vivado, if you also want to install the SDK, select *SDK Development Kit* from the menu.

 - a. **Read** and **Accept** all license agreements.
 - b. Select the Vivado edition (version) you wish to install (or SDK Standalone).
 - c. Select the Devices you need.
 - i. If choosing SDSoC, you can deselect SDACCEL
 - d. Cable Drivers are no longer installed by this installer under Linux. See instructions in the **Install Missing Cable Drivers** section of this document.



- e. Select the default directory for installation. Depending on the size of your virtual disk, you may be space limited here. You can delete the compressed installer archive to free 20-25 GB of disk space, if necessary.
 - f. If needed for your device or tool selection, obtain and install a license for your tools. If you are planning to use a free WebPACK license for development on your target platform, there is nothing further that is needed and you can close the Vivado License Manger dialog.
6. You may optionally delete the entire folder where you decompressed the installer to free up additional disk space.

Adjust GTK Version Used for Vivado in the VirtualBox Linux VM

It is a known issue that SDK will fail to launch on Ubuntu 16.04 without a workaround. This is also true for Ubuntu 18.04 and this is addressed in Xilinx Answer Record AR67580:

<https://www.xilinx.com/support/answers/67580.html>

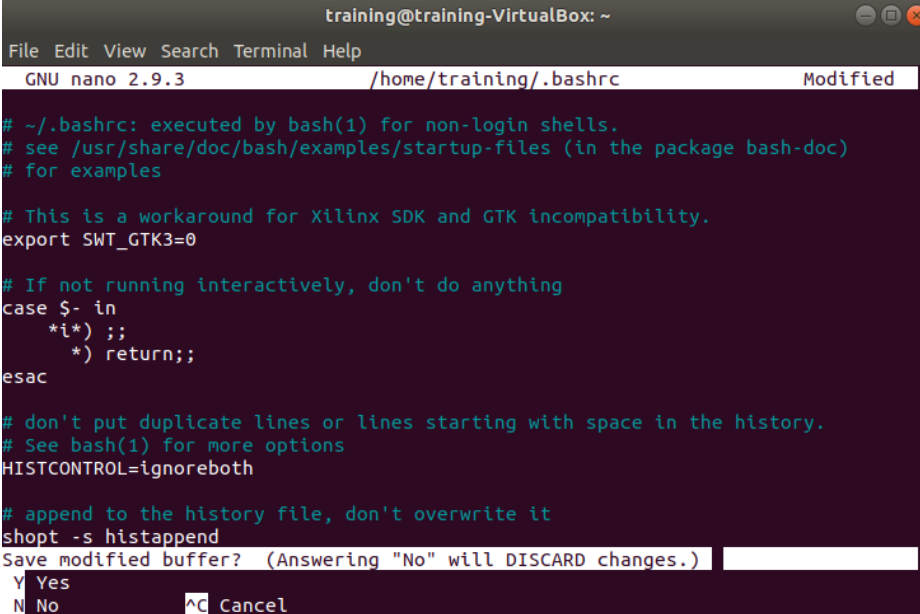
The issue is with the GTK version shipped with Ubuntu 18.04, which has issues with the eclipse. In order to work around this issue, set the environment variable **SWT_GTK3** to 0.

1. To temporarily set this environment variable for the current terminal session, use the following command:

```
$ export SWT_GTK3=0
```

2. To permanently set this environment variable for the all future terminal sessions, insert this comment and export command near the top of the `~/ .bashrc` file using your favorite editor:

```
# This is a workaround for Xilinx SDK and GTK incompatibility.  
export SWT_GTK3=0
```



```
training@training-VirtualBox: ~  
File Edit View Search Terminal Help  
GNU nano 2.9.3 /home/training/.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  
  
# This is a workaround for Xilinx SDK and GTK incompatibility.  
export SWT_GTK3=0  
  
# If not running interactively, don't do anything  
case $- in  
  *(*) ;;  
  *) 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  
Save modified buffer? (Answering "No" will DISCARD changes.)  
Y Yes  
N No  Cancel
```

3. Save the edits to the `~/ .bashrc` file.

Install make

Ubuntu 18.04 does not include make as part of the default install.

1. Run the below command to install make

```
$ sudo apt install make
```

```
training@training-VirtualBox:~/Xilinx_SDx_2019.1_0524_1430$ sudo apt 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 0 not upgraded.
Need to get 154 kB of archives.
After this operation, 381 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu bionic/main amd64 make amd64 4.1-9.1ubuntu1 [154 kB]
Fetched 154 kB in 1s (142 kB/s)
Selecting previously unselected package make.
(Reading database ... 161867 files and directories currently installed.)
Preparing to unpack .../make_4.1-9.1ubuntu1_amd64.deb ...
Unpacking make (4.1-9.1ubuntu1) ...
Setting up make (4.1-9.1ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
training@training-VirtualBox:~/Xilinx_SDx_2019.1_0524_1430$
```

Create Symbolic Link for gmake

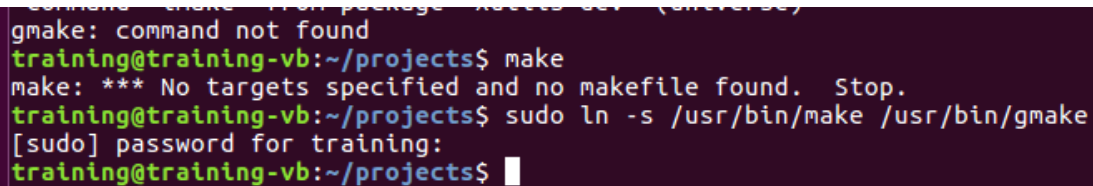
It is a known issue that SDK can fail to build, and by extension SDSoc's compiler flow, when using Ubuntu 16.04. This is also true for Ubuntu 18.04. Using the below workaround we can address this issue. It is also addressed more generically in Xilinx Answer Record AR68344:

<https://www.xilinx.com/support/answers/68344.html>

The issue is that Ubuntu does not include gmake by default. While make IS included.

2. To permanently set this we will create a symbolic link. As this will become permanent, we will not have to perform any further modifications.

```
$ sudo ln -s /usr/bin/make /usr/bin/gmake
```



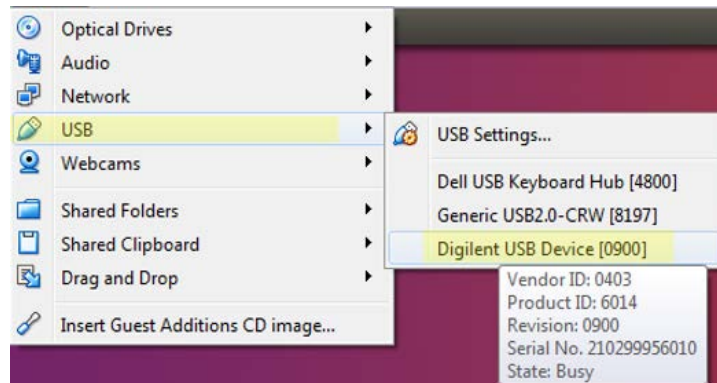
```
gmake: command not found
training@training-vb:~/projects$ make
make: *** No targets specified and no makefile found. Stop.
training@training-vb:~/projects$ sudo ln -s /usr/bin/make /usr/bin/gmake
[sudo] password for training:
training@training-vb:~/projects$
```

As you can see from the above image, gmake does not exist on this install, while make does.

Install GTK Terminal

The Ubuntu **Serial port terminal** (or GTK Terminal) application is used in some of the Avnet Reference Designs and Tutorial materials since it allows for simple connection to USB-UART of many development board platforms.

1. Plug in your development board and connect the USB-UART port to the PC so that the USB-UART device is recognized under Windows first. Then locate your USB-UART device under the **Devices**→**USB** selection menu and click on it to remove the device from the host operating system and add it to the guest operating system. In this example the Digilent USB Device [0900] device is used as it represents the USB-UART of the Avnet MiniZed board.



2. Once the device is detected and enumerated under Ubuntu, the USB-UART port should be listed under one of the `/dev/ttyUSBx` device entries. Locate the device entry for the USB-UART and make a note of this device for use in a later step.

```
$ ls /dev/ttyUSB*
```

```
training@training-VirtualBox: ~
training@training-VirtualBox:~$ ls /dev/ttyUSB*
/dev/ttyUSB0 /dev/ttyUSB1
training@training-VirtualBox:~$
```

3. To make it easier to launch the terminal app (GtkTerm) without needing to provide the root password each time, open a command window and add the current user to the group for the `/dev/ttyUSBx` devices used for USB-UART:

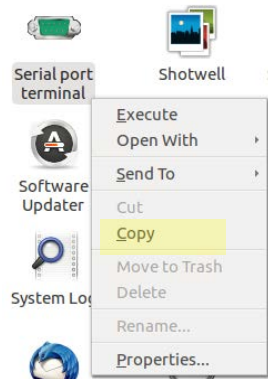
```
$ sudo usermod -a -G dialout <current username>
```

4. Install the gkterm package:

```
$ sudo apt-get install gkterm
```

5. **Reboot the Virtual Machine to force the changes to take effect.**

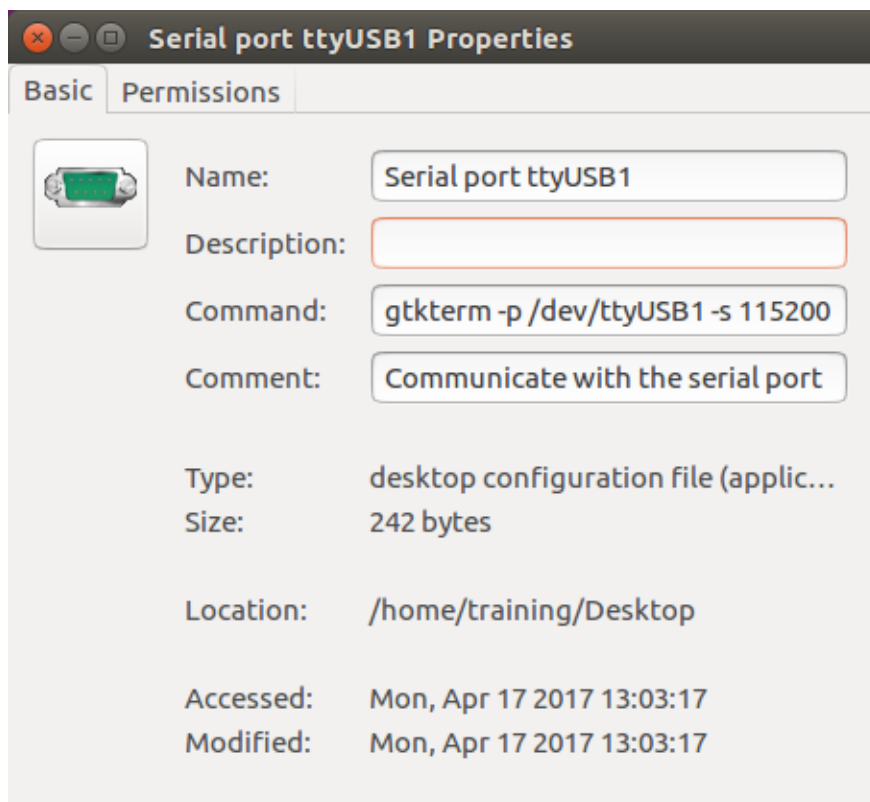
6. Create a Desktop icon by copying and pasting **Serial port terminal (gtkterm)** application from the `/usr/share/applications` folder directly to the `~/Desktop` folder:



7. Right-click on the new **Serial port terminal (gtkterm)** application Desktop icon and select the **Properties** option.
8. Within the Properties window, set the app attributes to match the USB-UART device attached to the system, in this example the USB-UART is attached to the `/dev/ttyUSB1` device entry:

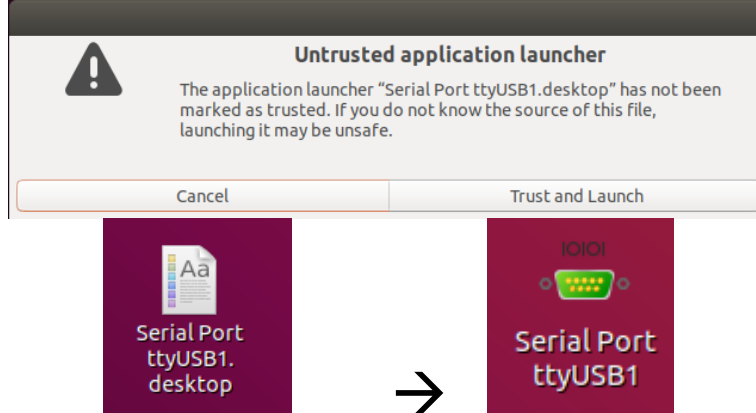
Name: **Serial port ttyUSB1**

Command: **gtkterm -p /dev/ttyUSB1 -s 115200**



9. Close the Properties window

10. The first time you launch this shortcut, you will need to trust it to launch the application. This also serves to correct the ICON image

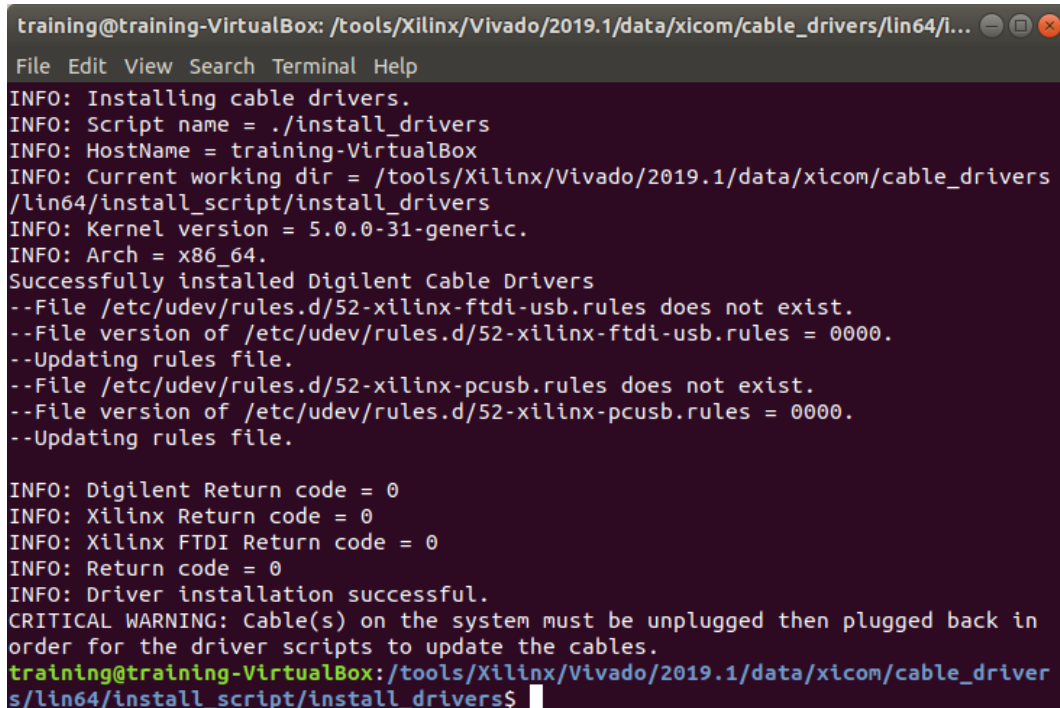


Install Missing Cable Drivers

The drivers, which were not included with the install, can be installed manually.

1. Open a Terminal window.
2. At the command prompt, enter:

```
$ cd
/tools/Xilinx/Vivado/2019.1/data/xicom/cable_drivers/lin64/install_script/install_drivers/
$ sudo ./install_drivers
```



```
training@training-VirtualBox: /tools/Xilinx/Vivado/2019.1/data/xicom/cable_drivers/lin64/i...
File Edit View Search Terminal Help
INFO: Installing cable drivers.
INFO: Script name = ./install_drivers
INFO: HostName = training-VirtualBox
INFO: Current working dir = /tools/Xilinx/Vivado/2019.1/data/xicom/cable_drivers
/lin64/install_script/install_drivers
INFO: Kernel version = 5.0.0-31-generic.
INFO: Arch = x86_64.
Successfully installed Digilent Cable Drivers
--File /etc/udev/rules.d/52-xilinx-ftdi-usb.rules does not exist.
--File version of /etc/udev/rules.d/52-xilinx-ftdi-usb.rules = 0000.
--Updating rules file.
--File /etc/udev/rules.d/52-xilinx-pcusb.rules does not exist.
--File version of /etc/udev/rules.d/52-xilinx-pcusb.rules = 0000.
--Updating rules file.

INFO: Digilent Return code = 0
INFO: Xilinx Return code = 0
INFO: Xilinx FTDI Return code = 0
INFO: Return code = 0
INFO: Driver installation successful.
CRITICAL WARNING: Cable(s) on the system must be unplugged then plugged back in
order for the driver scripts to update the cables.
training@training-VirtualBox:/tools/Xilinx/Vivado/2019.1/data/xicom/cable_driver
s/lin64/install_script/install_drivers$
```

Vivado and SDK Permission Issues

When you invoke Vivado, you **MAY** see the following error message, indicating a permission issue in the directory hierarchy:

```

training@training-VirtualBox: /opt/Xilinx/Vivado/201 /data/xicom/cable_drivers/lin64/
CRITICAL WARNING: [Common 17-741] No write access right to the local Tcl store a
t '/home/training/.Xilinx/Vivado/2016.4/XilinxTclStore'. XilinxTclStore is rever
ted to the installation area. If you want to use local Tcl Store, please change
the access right and relaunch Vivado.
ERROR: [Common 17-1257] Failed to create directory '/opt/Xilinx/Vivado/2016.4/tc
lapp'.
start_gui
Error: Failed to save the Vivado user preferences file. Reason: '/home/training/
.Xilinx/Vivado/2016.4/vivado.ini (Permission denied)'
Failed to create the shortcut directory: '/home/training/.Xilinx/Vivado/2016.4/s
hortcuts'
Failed to create the layout directory: '/home/training/.Xilinx/Vivado/2016.4/lay
outs/application'
Failed to create the commands directory: '/home/training/.Xilinx/Vivado/2016.4/c
ommands'
Failed to create the layout directory: '/home/training/.Xilinx/Vivado/2016.4/lay
outs/'
Mar 29, 2017 10:59:04 AM java.util.prefs.FileSystemPreferences$1 run
INFO: Created user preferences directory.
Failed to create directory: /home/training/.profile
Error: Failed to save the Vivado user preferences file. Reason: '/home/training/
.Xilinx/Vivado/2016.4/vivado.ini (Permission denied)'
Error: Failed to save the Vivado user preferences file. Reason: '/home/training/

```

This indicates that the hidden folder named `.Xilinx`, where license and configuration information is stored, is read-only for the invoking username. One method to correct this is to change ownership of the directory hierarchy using root authority.

1. If you have the above error, change to the directory above the `.Xilinx` folder, which in this case is the `/home/<current user>` folder. Enter the following commands:

```

$ cd ~
$ sudo chown <current username> -R .Xilinx
$ sudo chgrp <current username> -R .Xilinx

```

2. Run the Xilinx Vivado and SDK 64-bit environment settings scripts in a terminal window to setup the environment path to the tools install folder so that Vivado and SDK can be located in the next steps.

```

$ source /tools/Xilinx/Vivado/2019.1/settings64.sh
$ source /tools/Xilinx/SDK/2019.1/settings64.sh
    -- Optional if using SDSoc --
$ source /tools/Xilinx/SDx/2019.1/settings64.sh

```

3. Launch Vivado or the SDK as usual using the fixed permissions on the .Xilinx folder.

To start Vivado from a Terminal window, enter the following command:

```
$ vivado &
```

To start the Xilinx SDK from a terminal window, enter the following command:

```
$ xsdk &
```

If you installed SDSoC and you want to start the Xilinx SDSoC from a terminal window, enter the following command:

```
$ sdx &
```


Install Missing Desktop Icons

In some cases Vivado and/or SDK desktop icons may be missing. These can be manually added through the use of some launcher shell scripts and by adding some desktop icon entries.

1. Change to the home directory for the current user.

```
$ cd ~
```

2. Using your favorite text editor edit a new shell script named **vivado_launch.sh**

```
$ gedit vivado_launch.sh
```

Paste the following text into that script file, save the contents and exit. This creates a script that is capable of invoking the settings script needed for Vivado to launch correctly.

```
#!/bin/bash
. /tools/Xilinx/Vivado/2019.1/settings64.sh
vivado &
```

3. Using your favorite text editor edit a new shell script named **sdk_launch.sh**

```
$ gedit sdk_launch.sh
```

Paste the following text into that script file, save the contents and exit. This creates a script that is capable of invoking the settings script needed for Xilinx SDK to launch correctly.

```
#!/bin/bash
# This is a workaround for Xilinx SDK and GTK incompatibility.
export SWT_GTK3=0
. /tools/Xilinx/SDK/2019.1/settings64.sh
xsdk &
```

4. If you have installed SDSoC, using your favorite text editor edit a new shell script named **sdx_launch.sh**
Otherwise skip to the next step

```
$ gedit sdx_launch.sh
```

Paste the following text into that script file, save the contents and exit. This creates a script that is capable of invoking the settings script needed for Xilinx SDK to launch correctly.

```
#!/bin/bash
# This is a workaround for Xilinx SDK and GTK incompatibility.
export SWT_GTK3=0
. /tools/Xilinx/SDx/2019.1/settings64.sh
sdx &
```

5. Change permissions of both launcher scripts so that they can be executed.

```
$ chmod u+x vivado_launch.sh
$ chmod u+x sdx_launch.sh
      -- Optional if using SDSoC --
$ chmod u+x sdx_launch.sh
```

6. Change directories to the home Desktop folder of the current user.

```
$ cd ~/Desktop
```

- Using your favorite text editor edit a new file named **Vivado.desktop**

```
$ gedit Vivado.desktop
```

Paste the following text into that script file, save the contents and exit. This creates a desktop launcher icon that is capable of invoking the Vivado launcher script created in earlier steps.

```
#!/usr/bin/env xdg-open

[Desktop Entry]
Version=1.0
Type=Application
Terminal=false
Exec=/home/training/vivado_launch.sh
Name=Vivado 2019.1
Comment=Xilinx Vivado Design Suite 2019.1
Icon=/tools/Xilinx/Vivado/2019.1/doc/images/vivado_logo.ico
StartupNotify=true
```

- Using your favorite text editor edit a new file named **SDK.desktop**

```
$ gedit SDK.desktop
```

Paste the following text into that script file, save the contents and exit. This creates a desktop launcher icon that is capable of invoking the Xilinx SDK launcher script created in earlier steps.

```
#!/usr/bin/env xdg-open

[Desktop Entry]
Version=1.0
Type=Application
Terminal=false
Exec=/home/training/sdk_launch.sh
Name=Xilinx SDK 2019.1
Comment=Xilinx Software Development Kit 2019.1
Icon=/tools/Xilinx/SDK/2019.1/data/sdk/images/sdk_logo.ico
StartupNotify=true
```

9. If using SDSoC, using your favorite text editor edit a new file named **SDX.desktop**
Otherwise, skip to the next step

```
$ gedit SDX.desktop
```

Paste the following text into that script file, save the contents and exit. This creates a desktop launcher icon that is capable of invoking the Xilinx SDK launcher script created in earlier steps.

```
#!/usr/bin/env xdg-open

[Desktop Entry]
Version=1.0
Type=Application
Terminal=false
Exec=/home/training/sdx_launch.sh
Name=Xilinx SDx 2019.1
Comment=Xilinx SDSoC Development Kit 2019.1
Icon=/tools/Xilinx/SDx/2019.1/docs/images/sdx_icon.ico
StartupNotify=true
```

10. Change permissions of both launcher scripts so that they can be executed.

```
$ chmod u+x Vivado.desktop
$ chmod u+x SDK.desktop
-- Optional if using SDSoC --
$ chmod u+x SDX.desktop
```

Once the execution permission is changed, and you have trusted the applications, the icons will populate on the Desktop with the correct graphics. Note that SDx is optional.



PetaLinux Installation

Installing PetaLinux tools on a supported Linux system should be straightforward, but depending on the precise configuration of your development system, there could be some challenges to create an optimal environment. This section outlines a few of the common issues that may be encountered. For more information on the installations of PetaLinux tools, please refer to Xilinx User Guide UG1144.

TFTP Server Install and Setup

1. Install the following packages:

```
$ sudo apt-get install xinetd tftpd tftp
```

2. Create /etc/xinetd.d/tftp and put this entry:

```
service tftp
{
protocol      = udp
port         = 69
socket_type  = dgram
wait        = yes
user        = nobody
server      = /usr/sbin/in.tftpd
server_args = /tftpboot
disable     = no
}
```

3. Create a tftpboot folder, this should match what is in the server_args, and assign read/write permissions with the following command:

```
$ sudo mkdir /tftpboot
$ sudo chmod ugo+rw /tftpboot/
```

4. Restart the xinetd service.
For new systems:

```
$ sudo service xinetd restart
```

For older systems:

```
$ sudo /etc/init.d/xinetd restart
```

Now our tftp server is up and running.

Install openssl Libraries for PetaLinux

As of 2015.2, the PetaLinux tools require the **openssl** libraries on the host system which can be installed under Ubuntu using the following command:

```
$ sudo apt-get --yes install libssl-dev
```

Install Additional System Tools and Library Dependencies for PetaLinux

According to Xilinx User Guide UG1144, PetaLinux 2019.1 tools require some additional system tools and libraries to be installed on the host system. For exact system dependencies, refer to UG1144 document.

If you are using Ubuntu, this can be accomplished in a terminal window with the following command:

```
$ sudo apt-get --yes install tofrodos iproute2 gawk gcc git-core  
make net-tools libncurses5-dev tftpd zlib1g-dev flex bison lib32z1  
lib32ncurses5 lib32ncursesw5 lib32gomp1 xvfb chrpath socat autoconf  
libtool texinfo gcc-multilib libsdl1.2-dev libglib2.0-dev  
zlib1g:i386 xterm python
```

Adjusting Permissions of Vivado and PetaLinux Install Folder

As of 2019.1, PetaLinux tools require the installation as a non-root user so the permissions of these folders under the /opt folder must be adjusted accordingly.

This can be accomplished in a terminal window with the following commands:

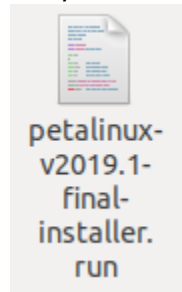
```
$ sudo chmod ugo+w /tools/  
$ sudo mkdir /tools/petalinux-v2019.1-final  
$ sudo chmod -R ugo+w /tools/Xilinx  
$ sudo chmod -R ugo+w /tools/petalinux-v2019.1-final  
$ sudo chown -R <current username>:<current user group>  
/tools/petalinux-v2019.1-final
```

Install PetaLinux into /opt/petalinux-v2019.1-final Folder

The PetaLinux 2019.1 Installer (TAR/GZIP – 7.7 GB) takes about 1.5-4 hours depending on the download speed. The file can be downloaded here:

<https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools/2019-1.html>

1. Copy the PetaLinux installer from your host system or Downloads folder to your VM desktop.



2. Open a terminal window and launch the installer while specifying the target install folder.

```
$ cd ~/Desktop
$ ./petalinux-v2019.1-final-installer.run /tools/petalinux-
v2019.1-final
```

3. The Vivado 2019.1 Installer will take several minutes to verify the integrity of the packed installer and then extract itself.
4. Read the PetaLinux license agreements and, if you can accept the license conditions, press enter to view the license, use keyboard **page up/down** keys to scroll through the agreement, press the **q** key to quit viewing the agreement when finished, and press the **y** key if you accept the license conditions. If prompted, allow the installer to make changes to your development system.

```
training@training-VirtualBox: ~/Desktop
File Edit View Search Terminal Help
INFO: Checking installation environment requirements...
INFO: Checking free disk space
INFO: Checking installed tools
INFO: Checking installed development libraries
INFO: Checking network and other services
INFO: Checking installer checksum...
INFO: Extracting PetaLinux installer...

LICENSE AGREEMENTS

PetaLinux SDK contains software from a number of sources. Please review
the following licenses and indicate your acceptance of each to continue.

You do not have to accept the licenses, however if you do not then you may
not use PetaLinux SDK.

Use PgUp/PgDn to navigate the license viewer, and press 'q' to close

Press Enter to display the license agreements
Do you accept Xilinx End User License Agreement? [y/N] > y
Do you accept Webtalk Terms and Conditions? [y/N] > y
Do you accept Third Party End User License Agreement? [y/N] > y
INFO: Installing PetaLinux...
```

- Be sure to source the `/tools/petalinux-v2019.1-final/settings.sh` script prior to attempting to use the PetaLinux tools for development.

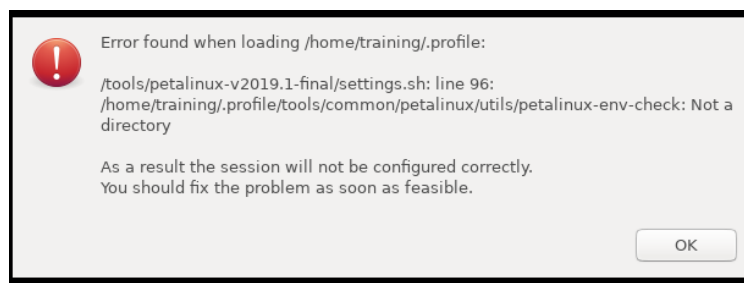
```
$ source /tools/petalinux-v2019.1-final/settings.sh
```

- To permanently set this environment variable for the all future terminal sessions, insert this comment and export command near the top of the `~/.bashrc` file using your favorite editor:

```
# This automatically sets up the PetaLinux tools environment.
source /tools/petalinux-v2019.1-final/settings.sh
```

```
training@training-VirtualBox: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 /home/training/.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
# This is a workaround for Xilinx SDK and GTK incompatibility.
export SWT_GTK3=0
# This automatically sets up the PetaLinux tools environment.
source /tools/petalinux-v2019.1-final/settings.sh
# If not running interactively, don't do anything
case $- in
  *(*) ;;
  *) return;;
esac
# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

- Save the edits to the `~/.bashrc` file
- With Ubuntu 18.04 it has been observed that some system variables are not set immediately upon boot. This can cause the operating system to HALT during boot. It does not cause serious issues aside from preventing the continuation of boot until you click “OK.” As such, until a proper solution comes to fruition, a workaround should be implemented to make workflow efficient



- Edit `settings.sh` located at `/tools/petalinux-v2019.1-final`
- Scroll to line 96


```
FL  
"${PETALINUX}"/tools/common/petalinux//utils/petalinux-env-check  
□
```

- c. Comment out the line shown in the above figure to hard-code the variable ``${PETALINUX}` to be instead `/tools/petalinux-v2019.1-final/`

```
#"${PETALINUX}"/tools/common/petalinux/utils/petalinux-env-check  
/tools/petalinux-v2019.1-final/tools/common/petalinux//utils/petalinux-env-check
```

- d. Save this file

Appendix

If you have a need to license the Vivado tools installed on your virtual machine to handle other Xilinx devices or IP Cores that are not covered by the free Xilinx WebPACK License, you will need to perform the workaround outlined in Xilinx Answer Record 60510:

<https://www.xilinx.com/support/answers/60510.html>

There is also some additional, distribution specific, guidance provided in this MiniZed.org Community Forum Post.

<http://minized.org/content/running-vivado-centos-7-virtual-machine>

This limitation of the licensing tool being able to read the MAC ID from the modern Ubuntu Ethernet interface naming conventions will likely be fixed in a future release of the Xilinx tools.

Revision History

Version	Date	Details
1.0	Feb 19, 2015	VirtualBox 4.3, CentOS 6.5, CentOS 7
1.1	Feb 24, 2015	Device tree reverse compilation, Ethernet Adapter Names
1.2	November 2015	Removed CentOS 6.5 support, validated all instructions, and restructured/standardized document.
1.3	August 2016	CentOS 7 support, Ubuntu 16.04, CentOS PetaLinux 2016.2, and Xilinx Vivado/SDK 2016.2
1.4	April 2017	Updated for PetaLinux 2016.4 and Xilinx Vivado/SDK 2016.4 tools, removed instructions and support for CentOS, removed unnecessary sections
1.5	July 2017	Updated for PetaLinux 2017.1 and Xilinx Vivado/SDK 2017.1 tools
1.6	August 2017	Updated for PetaLinux 2017.2 and Xilinx Vivado/SDK 2017.2 tools
1.7	September 2017	Corrected download link to PetaLinux 2017.2 tools and adjusted section headers
1.8	August 2018	Updated for PetaLinux 2018.2 and Xilinx Vivado/SDK/SDSoC 2018.2 tools; added cumulative download section
1.9	January 2019	Updated for PetaLinux 2018.3 and Xilinx Vivado/SDK/SDSoC 2018.3 tools, tools moved to new folder Added AR relating to gmake (SDSoC critical)
1.10	March 2019	Added notes for disabling Hyper-V under Windows 10 and allocating additional CPU cores for the VM
2.0	October 2019	Updated for VirtualBox 6 as well as Xilinx 2019.1 tools