Quick Start Guide for BeagleBone

by Brian Fraser
Last update: Jan 10, 2024

Guide has been tested on

BeagleBone (Target):	**Debian 11.8**
PC OS (host):	**Debian 11.8** (or higher)

This document guides the user through

1. Installing Debian on the computer in a virtual machine.
2. Connecting to the target using serial port and SSH.
3. Cross compile and run on the target.

Table of Contents

Note: Guide not yet fully tested in SFU Surrey Linux Lab (SRYE4013). Some changes may be needed.

Formatting

1. Commands for the host Linux's console are show as:
   ```
   (host)$ echo "Hello PC world!"
   ```
2. Commands for the target (BeagleBone) Linux's console are shown as:
   ```
   (bbg)$ echo "Hello embedded world!"
   ```
3. Almost all commands are case sensitive.

Revision History

- Sept 18, 2022: Updated to Debian 11.4 (target) and Debian 11 (host).
- Sept 26: Updated link to getting VMware fusion for mac for free.

1. Host OS Setup

1. **Linux Debian 11.8+ ("bullseye")** 64bit is the supported OS on the host PC for this class. Other versions of Linux or Unix derivatives may work, but are not supported by the instructor and TA.
   - You can download **Debian 11.8** from its official site:
     https://www.debian.org/releases/bullseye/debian-installer/
     Pick the version for your processor; likely **amd64** or **arm64**
   - Use of Ubuntu is *not* recommended because of version conflicts of the C libraries between the target vs host. You might have some luck with an *older* version of Ubuntu (20.04).
2. If you currently run Windows or MacOS, you may install Linux in a virtual machine (VM). Running inside a VM means one extra level of configuration is required, but it is an effective setup for this course.
   - Under MacOS (especially for Apple processors), you may want to run VMware Fusion. There is a free version, or you can get a Fusion Pro license through SFU. Students have also had success with UTM (free on website; paid in Mac App Store).
   - Under Windows use either VMware Worstation Player or VirtualBox (both free).
     This YouTube playlist features some videos showing how to install a VM.

     I recommend VMware because under Windows because it runs well with Hype-X if you have previously installed WSL 2 or Docker.
     - If you use VMware Player, some students have found problems if you plug any USB 2.0 hardware into a USB 3.0 slot on your computer. If you are having USB problems, try rebooting your VM and your host OS, and switching the port you are plugging the device into ensuring it is a USB 2.0 device.
     - If given the choice, select "Graphical Install" (over just "install") so you get a GUI.
   - Directions for in SFU CSIL Labs (SRYE 4013, and possibly all other CSIL labs):
     - VirtualBox is installed in the labs. When you create a virtual machine it will default to "`~/VirtualBox VMs`" which is remapped to `/usr/shared/CMPT/scratch/<yourID>/VirtualBox`
       - This directory has more space than your home folder and is shared by all CSIL machines so you'll have access to your VM on any PC in CSIL.
       - The folder is access protected so only you have access to it.
     - Note that you cannot complete the assignments in the lab without a virtual machine because you do not have root permissions on the lab computers. We'll need root permissions to setup file sharing and install/upgrade programs.
     - When creating a virtual machine in the lab, I recommend 4GB ram, 20GB Hard drive.
3. If you install Linux in a VM:
   - Make the virtual hard drive large (>70GB, maybe 100GB). This gives plenty of room for extra build tools, libraries, and such.
     - In the lab, use a smaller hard drive because it is using up shared space.
   - Give it at least 2GB (up to 8GB) of RAM, video RAM, and as many processor as possible; this will speed up Linux. Also enable 3D graphics acceleration for improved performance.

4. You will want your user to be able to use the **sudo** command. Try the following command:

   ```
   (host)$ sudo ls
   ```

   If it failed with message about "... is not in the sudoer file", then try the following ([1]):

   ```
   (host)$ su
   (host)# EDITOR=nano visudo          (Unnecessary on Debian)
   (host)# /sbin/visudo
   ```

   Add the following to the end of the file, changing "`brian`" to your current user name:

   ```
   brian   ALL=(ALL) NOPASSWD:ALL
   ```

   To save and exit press CTRL+x, then type "Y" and enter to save the file ("buffer").

   Exit the `su` prompt (`` `exit` ``) and retry the `` `sudo ls` `` command!

5. If you are running a 4K display, you may want to change the UI scaling inside your VM for the Gnome graphical environment:

   ```
   (host)$ gsettings set org.gnome.settings-daemon.plugins.xsettings overrides \
     "[{'Gdk/WindowScalingFactor', <2>}]"
   (host)$ gsettings set org.gnome.desktop.interface scaling-factor 2
   ```

6. If running in a virtual machine, I recommend turning off power saving features in Linux as I have found this can lock-up the VM when it tries to enter power-saving mode:
   - In your Linux VM, go to Settings > Power
   - Black Screen: Never
   - Automatic Suspend: Off
7. Install basic software
   - **Update**

     ```
     (host)$ sudo apt update
     (host)$ sudo apt upgrade
     ```

   - **Install VS Code**

     ```
     (host)$ sudo apt install snapd
     (host)$ sudo snap install --classic code
     ```

     You may need to reboot after installing VS Code before you can run it from the terminal:

     ```
     (host)$ code
     ```

   - **Git**

     ```
     (host)$ sudo apt install git
     ```

   - **SFU VPN**

     On my computer, if I have my host OS (Windows) running the SFU VPN then I am able to connect to our GitLab server. SFU allows CS students to use the VPN.

     If you want native Linux support you may try:

     ```
     (host)$ sudo apt install openfortivpn
     ```

     Each time you need to connect to an SFU system requiring the VPN, you can run with:

     ```
     (host)$ sudo openfortivpn vpn.its.sfu.ca:10433 -u <YourSfuUserId>
     ```

     When prompted, enter your SFU password, and then a MFA code.
     Suggestion: Make a script in your home folder for this.

8. Troubleshooting:
   - If having problems with `apt` and dependencies, try following this guide: http://askubuntu.com/questions/140246/how-do-i-resolve-unmet-dependencies-after-adding-a-ppa

1 Linuxize, "How to add user to sudoers in Ubuntu": https://linuxize.com/post/how-to-add-user-to-sudoers-in-ubuntu/

Note that if using the `software-properties-gtk` tool, you may need to run it with `sudo` from the command line:
`(host)$ sudo software-properties-gtk`

- If your computer crashes when launching your VM, try disabling any automatic USB device redirects to your VM.
- If you are having troubles getting Linux to run in a VM under windows with Hyper-X/Docker/…, try:
  - Go to Start, then search for “Turn Windows features on or off”
  - Enable all of the following (will likely make VirtualBox not work; for VirtualBox to work you may need to disable Windows Hypervisor Platform, and/or Hyper-X):
    - Hyper-V
    - Virtual Machine Platform
    - Windows Hypervisor Platform
- If you install your host OS and it has no graphical user interface, then you may have needed to select “Graphical Install” during the installation instead of just “Install.”

1.1 Tips for VM on USB Stick

If working in the lab, you should use the CSIL shared space (described above) because it will be faster and store the data more reliably.

It’s *possible* to install a Virtual Machine onto a USB memory stick so that you can take it with you between computers. This can be slow, and much easier to corrupt than a fully installed VM, so use only if necessary.

Here are some tips on using a USB stick if you choose to:

- Get a USB ***3.0*** memory stick! Need 32+ GB likely. Insert it into the “SuperSpeed” USB port on the computers (has an “SS”). Best to have your USB stick formatted to exFat so that it can be read under either Windows or Linux.
- Open Oracle VM VirtualBox Manager, and change the location it stores VM’s:
  File > Preferences > General
  Change “Default Machine Folder” to be `/media/<user>/<USB Drive name>`
- In Oracle VM VirtualBox Manager create a VM.
  - Edit VirtualBox's preferences and download the Extension Pack for Virtual Box from virtualbox.org. Note installed version is 4.3.40.
  - Edit the settings of the VM (when the VM is powered off) under USB select "Enable USB 2.0 (EHCI) Controller"
- On a different PC you want to develop on, in the VM Manager select Machine → Add, and browse to your VM's folder on your USB stick.
- **Note that you must be extra careful never to pull the USB stick when the VM is running because this could corrupt the state of your VM.**
- You may also find it faster to shut-down and fully restart the VM than having the VM suspend to disk. Plus, this is one less thing to go wrong with writing to the USB drive.

- When switching a VM between computers, some settings may need to be updated such as the network adapter used for bridging, shared folders, and so forth.

2. Basic Wiring and First Boot

1. Connect the BeagleBone Green to the host PC is using a **single micro-USB cable**.

   a) Plug the USB-A end into the PC
   b) Plug the USB-micro end into the port on the BeagleBone beside the Ethernet jack, underneath the board.

   Note that the Zen cape (top board) also has a USB-micro connector on the top; this is explained in the next section.

2. You'll see the lights on the BeagleBone start flashing!
3. Other connections, such as the Zen cape's USB-micro, or an Ethernet cable are discussed in later sections and guides.

Note that the USB-micro connection to the BeagleBone provides the following:

- Power to the BeagleBone
  - If you have extra hardware plugged into the BeagleBone, you may find that it begins pulling too much current from the PC's USB port. In which case you may need a powered hub, or to run the system off a USB power adapter (plug it into the wall and it gives power to a USB-A port).
- Ethernet access between the host and target over USB via "Ethernet over USB".
- Mapping (part of) the BeagleBone as a mass storage device to the host PC.

We only are using the first two of these during this course.

3. Screen

Screen is a tool which allows you to interact with a serial port through Linux. To connect to the BBG using a serial connection (via Screen), you'll need either a Zen Cape, or an "FTDI Serial TTL-232 USB Cable". If you have neither, you can skip this section.

1. Zen Cape's TTL to USB setup:
   - Connect using two microUSB cables:
     microUSB Cable 1: BeagleBone (bottom board) to host (PC)
     micreUSB Cable 2: Zen cape (top board) to host (PC)

     Cable 2 gives us access to the serial port via the Zen Cape's TTY to USB adapter chip (by FTDI). This connection is going **from the Host (PC) to the Zen cape;** nothing should be plugged into the BBG's USB A port (rectangular).
     a) Plug the USB A end of the cable into the PC
     b) Plug the microUSB end into the Zen cape (top board)
   - If using a virtual machine (VM), you may need to map the adapter to the VM.
     In VMware Player:
     - Go to the menu Player > Removable Devices > "`future devices ft231x usb uart`", or "`future devices ft230x basic uart`" or similar.
     - Click it to map it to the VM
     - You may also use the icon in upper right (on tool bar); right click it, and select connect.

     In VirtualBox:
     - Go to the menu Devices > USB and check "`FTDI FT230X Basic UART`" (or the like)
     - You can also use Devices > USB > USB Settings to configure the VM to automatically gain control of this device.
   - In Linux, just after connecting the device you can see if it has been configured correctly:
     `(host)$ sudo dmesg`
     - This shows messages produced by the kernel. You should see the last few lines being:

```
[..] usb 2-2: new full-speed USB device number 4 using ohci-pci
[..] usb 2-2: New USB device found, idVendor=0403, idProduct=6015, bcdDevice=10.00
[..] usb 2-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[..] usb 2-2: Product: FT230X Basic UART
[..] usb 2-2: Manufacturer: FTDI
[..] usb 2-2: SerialNumber: DA010RLT
[..] ftdi_sio 2-2:1.0: FTDI USB Serial Device converter detected
[..] usb 2-2: Detected FT-X
[..] usb 2-2: FTDI USB Serial Device converter now attached to ttyUSB0
```

     - This shows that the device is connected to `/dev/ttyUSB0`. If you have multiple devices connected, yours may connect to `ttyUSB1` or higher.

2. Check `/dev/ttyUSB0` exists with the following command (expected output shown on next line):
   `(host)$` **`ls -al /dev/ttyUSB0`**
   `crw-rw---- 1 root dialout 188, 0 Sep  1 09:57 /dev/ttyUSB0`

3. Install the `Screen` program to be able to read/write to the serial port:
   `(host)$` **`sudo apt install screen`**
4. Start `Screen` with the following command:
   `(host)$` **`sudo screen /dev/ttyUSB0 115200`**
   - Where `/dev/ttyUSB0` is the port. May be `ttyUSB1`, etc.
   - `115200` is the "baud rate", or the speed of the connection. 115200 is the default serial port speed for the BeagleBone.
   - NOTE: If in SFU labs, you do not have root permission on host OS. Do not try any `sudo` commands in the *host OS* in the lab. Our IT staff will be automatically notified and they may contact you about it. In the lab use `sudo` only inside your VM.

     If you want to use Screen inside the host OS in the lab, you may have access to `/dev/ttyUSB0` already so just type the command without `sudo`.
5. In `screen`, press Enter (perhaps a couple times) to see text on the serial port if you are connecting to a target device which is already running.
   - User name is `debian` and password is `temppwd`
     On older boards, you may log in as `root` with no password
6. Screen commands:
   - Help: Ctrl-a then ?
   - **Quit: Ctrl-a then **
7. Trouble shooting
   - After launching `screen`, if you immediately see the message: `[screen is terminating]` then you may have forgotten to use `sudo`. (In SFU labs, you do not have sudo (root) access.)
   - If you briefly see the following message at the bottom of the screen:
     `Cannot exec 'ttyUSB0': No such file or directory`
     it likely means that you have not correctly connected the Zen Cape's USB-to-TTL-serial-adapter, or if you are using a virtual machine, that you have not mapped it to the virtual machine correctly.
     - Ensure that you are connecting the **micro-USB connector on the Zen cape to the PC** (not looping it back to the BBG). Once both micro-USB cables are connected, you should have two USB cables connecting to your PC.
     - For the virtual machine, try disconnecting the USB-to-TTL adapter from the VM, and then reconnecting it.
     - To verify you are connecting the correct USB device to the VM, do the following:
       - Unplug all the USB connections from the Zen/BBG
       - Make it so later `dmesg` shows us a message:
         `(host)$ echo "----------------------" | sudo tee /dev/kmsg`
       - Plug in just the ZEN cape to the PC
       - If needed, map the FTDI connection to the VM

    - Check what your Linux host says:
      `(host)$ sudo dmesg`
    - You should now see the message:
      `[...] usb 2-2: FTDI USB Serial Device converter now attached to ttyUSB0`

- If you run `screen` and see nothing, try:
    - press enter: if the board is at the log-in prompt it will show you the prompt.
    - ensure the BBG has power (lights are flashing on the BBG)
    - reboot the BBG: press the reset button on the BBG. It's the left most button as you look at the Ethernet connector end of the board. This should trigger the board to start spitting out data on the serial port.
    - reboot your VM (if applicable), computer, and BBG.
    - If after all of these steps `screen` still shows just a blank screen, then it is possible that the BBG is not connected to the Zen cape correctly. You may want to ask Dr. Brian for help in doing the following:
      - Remove the Zen cape from the BBG
      - Ensure your Zen cape has a blue connector on the bottom, beside the pins for P9.
      - Ensure your BBG has a row of pins beside P9. Ensure they are straight.
      - Reconnect the Zen cape to the BBG, ensuring that the pins go into the connector.
      - You may need to slightly bend the pins or the connector to make this happen.
- If your Screen session locks up or begins dropping characters or corrupting text, try:
    - In a VM, disconnect the USB to TTL adapter from the VM and/or physically disconnect the USB cable. Perhaps try plugging it back into another USB slot.
    - Ensure `screen` is running with the correct baud rate (115200).
    - Ensure your VM has more than the minimum video memory (if there's a setting for this).
    - Try increasing its access to 3D acceleration and the number of CPU cores (if possible).
    - Try closing the `screen` session (Ctrl-A then \), and restarting the screen session. Sometimes under VirtualBox it seems that closing `screen` also locks up, but it then resolves within a minute and successfully quits `screen`.
- If trying to reconnect gives the error: "Failed to attach the USB device FTDI TTL232R-3V3..." and "... is busy with a previous request", with VirtualBox then you may need to kill the `VBoxSVC.exe` process in the Host OS, and/or reboot the host OS.
- Your USB adapter may also come up on `/dev/ttyACM0`, then it likely means that you have not correctly plugged in or mapped the FTDI connection to the host Linux machine. The ttyACM0 port is a software emulated serial port and will not be sufficient.
- If the VM has problems (locks up, etc) when you connect the BBG to the PC, if you are automatically mapping both the BBG and the Zen cape to the VM then it may cause problems.
    - Try disabling the configuration in the VM's settings which automatically map the BBG and Zen to the VM and see if that helps.

4. Connecting via SSH and NFS

4.1 Networking

Right *now*, follow **Sections 1 and 2** in the **Networking** guide posted on the course website, then come back here.

4.2 SSH

1. SSH from your host PC to the target device:
   ```
   (host)$ ssh debian@192.168.7.2
   ```
   - Password is `temppwd`
   - Where the IP address is the address of the target device (BeagleBone). If your BeagleBone is using DHCP from a router, you can find its IP address by running `ip addr` on the target over the serial connection (using `screen`).
   - The "`debian@`" tells SSH we want to connect as a specific user, the user named `debian`, rather than the user name we logged into the host PC with.
   - You may be asked to accept a security certificate; type "yes".
   - If you have made significant changes to the target device (such as reinstalling its operating system or file system), it may require you to remove a saved security certificate from the host PC before allowing you to SSH to the target. It will give you the command to execute in the error message.
2. Optional: Use `sshpass` to SSH to the target without having to retype password.
   - Install:
     ```
     (host)$ sudo apt install sshpass
     ```
   - Run
     ```
     (host)$ sshpass -p temppwd ssh debian@192.168.7.2
     ```
     If this command fails, try first connecting via `ssh` command.
3. Via the SSH connection, you can do anything would do via the serial connection. For example:
   ```
   (bbg)$ echo I am connected via SSH
   (bbg)$ cd /proc
   (bbg)$ cat uptime
   (bbg)$ cat cpuinfo
   ```
4. Check the version of software image installed on your BeagleBone with the following:
   - Version of Debian:
     ```
     (bbg)$ cat /ID.txt
     BeagleBoard.org Debian Bullseye Minimal Image 2023-10-07
     ```
   - Version of Linux Kernel:
     ```
     (bbg)$ uname -a
     Linux BeagleBone 5.10.168-ti-r72 #1bullseye SMP PREEMPT Sat Sep 30
     03:37:21 UTC 2023 armv7l GNU/Linux
     ```

5. In general, prefer using the SSH connection over the serial connection because it is usually faster and less error prone.
6. Troubleshooting
    - `sshpass` will fail (doing nothing, showing no output) if you have not yet SSH'd into the target and it needs you to accept the identification information for the device.
      If `sshpass` fails, first connect with `ssh` and then retry using `sshpass`.
    - If your BBG shows a different version, then you may want to reflash your board.

4.3 Reflash the BBG

If directed to upgrade the version of software on the BBG, you must reflash it using a microSD card. See "Reflash your BBG" guide on the course website for steps. You should do this now before continuing the setup so you don't need to redo any steps later.

WARNING: When following the reflashing guide, don't run the enable-beagle-flasher command on your BBG when booted from the eMMC. This will make your board unbootable and require you to reflash your board from a microSD card.

4.4 NFS

Right *now*, follow the steps in the NFS guide posted on the course website.

5. Cross-compile Setup

5.1 Setup folders & Tools

1. On the *host* PC, create a directory for your work and for downloading (public):
   ```
   (host)$ mkdir -p ~/cmpt433/work
   (host)$ mkdir -p ~/cmpt433/public
   ```
   - Change `~/cmpt433/public` to have global read/write permissions:
     ```
     (host)$ chmod a+rw ~/cmpt433/public
     ```
   - Make sure none of your source code, makefiles, or the like is in the public directory. This directory can be read by everyone, so someone could take your work and hand it in, causing great problems for both students! It is OK, though, to have compiled programs and downloadable images (for U-Boot, the kernel, and the file system) in the public directory.
2. Install the cross-compiler and required tools:[2]
   ```
   (host)$ sudo apt install gcc make
   (host)$ sudo apt install gcc-arm-linux-gnueabihf
   (host)$ sudo apt install binutils-arm-linux-gnueabihf
   ```
   - The `hf` in the above commands means hardware floating point and is important![3]
   - If you choose to install a non-standard version of the compiler, create a link in the file system to redirect to your newly installed cross compiler:
     ```
     (host)$ cd /usr/bin
     (host)$ sudo ln -s arm-linux-gnueabihf-gcc-10 arm-linux-gnueabihf-gcc
     ```
3. Check tools installed with the following. It should display the compiler's version info:
   ```
   (host)$ arm-linux-gnueabihf-gcc --version
   arm-linux-gnueabihf-gcc (Debian 10.2.1-6) 10.2.1 20210110
   Copyright (C) 2020 Free Software Foundation, Inc.
   This is free software; see the source for copying conditions.  There is NO
   warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
   ```

2 You can choose to install a different version of the GCC cross compiler; however, going with a very new version may create binaries which depend on a newer version of GLibC than is installed on the target. To install the latest GCC, use: `(host)$ sudo apt install gcc-arm-linux-gnueabihf`

3 If you are targeting an old release for the BeagleBone which does not support hardware floating point, you'll want to install: `gcc-arm-linux-gnueabi` and `binutils-arm-linux-gnueabi`

5.2 Build for Host vs Target

1. Create a new directory, called `myApps`, inside your work folder for all the applications you'll be writing:
   ```
   (host)$ mkdir -p ~/cmpt433/work/myApps/quickStart
   (host)$ cd ~/cmpt433/work/myApps/quickStart
   ```
2. Create a `helloworld.c` program inside the `quickStart` folder. Use VS Code (`code`) to edit the program. Note that the `&` at the end of a command makes it execute in the background, which here allows you to keep using your terminal while `gedit` is running.
   ```
   (host)$ code helloworld.c
   ```
3. Write the hello world program in C and save the file:
   ```
   #include <stdio.h>
   int main(int argc, char* args[])
   {
         printf("Hello embedded world!\n");
         return 0;
   }
   ```
4. Compile and run the program **for the host OS**:
   ```
   (host)$ gcc helloworld.c -o helloworld_host
   (host)$ ./helloworld_host
   ```
5. Compile the program **for the target OS** (Linux on the BeagleBone):
   ```
   (host)$ arm-linux-gnueabihf-gcc helloworld.c -o helloworld_target
   ```
   - Try running the `helloworld_target` application on the host OS. What happens?
6. Analyze the two executables you have just compiled using `readelf`:
   ```
   (host)$ readelf -h ./helloworld_host
   …
   Machine:  Advanced Micro Devices X86-64
   …
   Flags:    0x0

   (host)$ readelf -h helloworld_target
   …
   Machine:  ARM
   …
   Flags:    0x5000400, Version5 EABI, hard-float ABI
   ```
   - Use `readelf` when you need to identify which OS an executable is compiled for.
7. Troubleshooting
   - When running `arm-linux-gnueabihf-gcc`, if you get the error `sys/cdefs.h` not found, you may be missing the 32-bit libraries. Try:
     ```
     (host)$ sudo apt install libc6-dev-armhf-cross
     ```
     Or use the Aptitude package manager:
     ```
     (host)$ sudo apt install aptitude
     (host)$ sudo aptitude gcc-arm-linux-gnueabihf binutils-arm-linux-gnueabihf
     ```

5.3 Running via NFS

1. Complete the NFS guide posted on the website to map the `~/cmpt433/public` folder on the target board.
2. On the host PC, copy the `helloworld_target` executable to the `~/cmpt433/public/` folder.
   ```
   (host)$ cd ~/cmpt433/work/myApps/quickStart
   (host)$ cp helloworld_target ~/cmpt433/public/
   ```
3. On the target, change to the mounted NFS folder:
   ```
   (bbg)$ cd /mnt/remote
   ```
4. On the target, run the executable:
   ```
   (bbg)$ ./helloworld_target
   ```
5. Congratulations! You've done some embedded development!
6. Repeat these steps for the `helloworld_host` executable and see the result. Should the host version work on the target?
7. Troubleshooting:
   - If you get a “File not found” error, even though your program is obviously in the correct spot, it may mean that you have a mismatch between the cross-compiler on your host and your target’s OS in terms of hardware vs software floating point. Use `readelf` on your executable to see if it uses software or hardware floating point (the `Flags`). You can check what your target requires by compiling the Hello World example program natively on the target (using `gcc` directly on the BeagleBone), and run `readelf` on the program it generates.

6. Done

If you are using a VM, now that you have it configured, you may want to take a snapshot of it in case something goes wrong in the future.

- If running VMware:
  - Tell your Linux VM to shutdown. Make sure you tell Linux to shutdown, rather than clicking the X on the VM to close the window.
    (Note that if you are just suspending the VM, clicking the X and choosing suspend is a good choice for day-to-day use).
  - Open VMware player, right-click on your VM and go to settings.
  - On the Options tab, under General copy the Working Director path.
  - In your host OS, open your file manager and browse to the copied path.
  - Copy the whole folder to a backup location (may be numerous GB!)
    Or, you could ZIP the whole folder and move that.
  - To restore later, you should be able to copy this saved copy back over-top of the original (rename original to a temporary name first, just in case).
- If running VirtualBox, it supports taking snapshots within its interface.
  - Note that a snapshot is not a full copy of the VM, but a diff that can be used to restore the VM to this state.