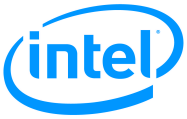
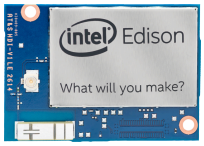




Intel® Edison Tutorial: Introduction to Linux

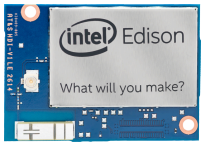
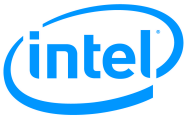
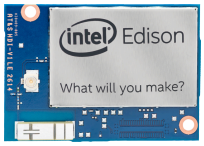


Table of Contents

Revision history		
Version	**Date**	**Comment**
1.0		Initial release





Introduction

For a good explanation on what Linux and what an operating system is, please refer to the following link:

https://www.linux.com/what-is-linux

To summarize, an operating system (OS) is software that supports a computer's basic functions such as task scheduling, execution of applications and control of peripherals. Some examples of common OS's are Windows, Mac OS X, and Linux.

The OS is comprised of a number of pieces:

1. **The Bootloader**: The software that manages the boot process of your computer.

   https://en.wikipedia.org/wiki/Booting
2. **The Kernel**: The central core of a computer's OS. It has complete control over everything that happens in the system.

   https://en.wikipedia.org/wiki/Kernel_(operating_system)
3. **Daemons**: A computer program that runs as a background process.

   https://en.wikipedia.org/wiki/Daemon_(computing)
4. **The Shell**: a text based user interface for access to an OS's services.

   https://en.wikipedia.org/wiki/Shell_(computing)
5. **Graphical Server**: Sub-system that displays graphics on your monitor.

   https://en.wikipedia.org/wiki/Display_server
6. **Desktop Environment**: a graphical user interface for access to an OS's services. They do not usually provide access to all features found in the underlying operating system.

   https://en.wikipedia.org/wiki/Desktop_environment
7. **Applications**: a computer program designed to perform a group of coordinated functions tasks or activities.

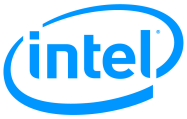
   https://en.wikipedia.org/wiki/Application_software

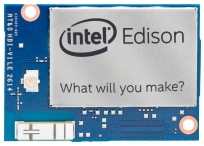
In this tutorial, you will:

1. Learn about Linux commands.

List of Required Equipment and Materials

- Intel Edison Kit
- 2x USB 2.0 A-Male to Micro B Cable (micro USB cable)
- 1x powered USB hub **OR** an external power supply
- A personal computer





Bash – The Bourne Again Shell

When you log into the Intel Edison, the first thing you see is the Bash shell, which is a shell by GNU project. The Bash shell is a command line interface (CLI) as opposed to a graphic user interface (GUI). For more information on what the Bash shell is, what the CLI is and how it differs from a GUI, please examine the below references:

- https://www.gnu.org/software/bash/
- https://en.wikipedia.org/wiki/Graphical_user_interface
- https://en.wikipedia.org/wiki/Command-line_interface
- http://www.computerhope.com/issues/ch000619.htm

While a GUI is convenient for most regular users, a CLI is much more powerful for advanced users and developers. One major advantage of a CLI is shown when copying files from one directory to another. Consider the following scenario: there exists a directory on a computer that contains both **.jpeg** and .**png** files. A user wishes to copy all files with the file extension **.jpeg** to a directory labelled **JPEG_FILES** and all files with the file extension **.png** to a directory labelled **PNG_FILES**. To do this on a GUI, the user would need to:

1. Create a new directory with the appropriate name
2. Find all files with the relevant file extension
3. Move them one by one or by selecting multiple at a time

To do this same task on a CLI, the user would need to:

1. Create the new directories
2. Issue the command “cp ./*.jpeg ../JPEG_FILES”
3. Issue the command “cp ./*.png ../PNG_FILES”

NOTE:

- **The dollar sign ‘$’ means it is a shell prompt and it is not a part of the command. For instance, if you are issuing the following command:**

  $ ls

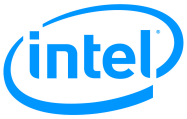
  You would only input the “ls” (without quotation marks) into the shell prompt.

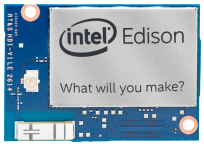
- **To issue a command, type the command into the shell prompt, and press [Enter]**

1. Access the shell on your Intel Edison by following the steps in the document labelled ***Intel Edison Tutorial – Introduction, Shell Access and SFTP***
2. To ***make a directory***, use the **mkdir** command.

   Issue the following command:

   $ mkdir directory1





mkdir is a command that creates a directory if it does not already exist.
The format to issue the command is as follows:

mkdir [option] directory_name

You can ignore the **[option]** for now.
For more information, please refer to the below reference:
http://www.linfo.org/mkdir.html

3. In a GUI based file system viewer such as Finder on Mac or Explorer on Windows, you can change which directory you are in by clicking the directory you wish to navigate to.

   To do the same in Bash, you must issue a ***change directory***, or **cd** command. Issue the below command to navigate to the directory you created in the previous step.

   $ cd directory1

```
root@edison:~# mkdir directory1
root@edison:~# cd directory1
root@edison:~/directory1#
```

Figure 1: Bash session after making a new directory and navigating to it

   For more information, please refer to the below reference:
   http://www.linfo.org/cd.html

4. In Linux, there are three standard streams to allow user interaction with the shell: standard input, standard output, and standard error. Standard input is usually read from the keyboard, and text being passed to standard output is displayed on your monitor.

   For more information about what these streams are, please refer to the below references:
   https://en.wikipedia.org/wiki/Standard_streams
   http://www.linfo.org/standard_output.html
   https://linux.startcom.org/docs/fr/Introduction%20to%20Linux/sect_05_01.html

   Sometimes you may want to print a string to standard output through the shell. To do this, issue the below command and examine your terminal.

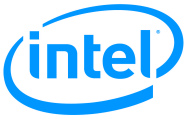
   $ echo hello

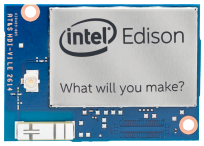
```
root@edison:~/directory1# echo hello
hello
```

Figure 2: Bash session after successfully issuing the echo command

   For more information, please refer to the below reference:
   http://www.linfo.org/echo.html

5. The standard streams can be redirected by using the '>' operator.





Issue the below command to redirect standard output to a file.

$ echo blah > file1.txt

6. To display read a file to and print it to standard output, use the command **cat**.

   Issue the below command:

   $ cat file1.txt

```
root@edison:~/directory1# cat file1.txt
blah
```

Figure 3: Bash session after successfully issuing the cat command

   For more information, please refer to the below reference:
   http://www.linfo.org/cat.html

7. **$ echo hello > file1.txt**
   Redirection with > overwrites the previous content.

8. **$ cat file1.txt**
   As you can see, the content has changed from "blah" to "hello".

```
root@edison:~/directory1# cat file1.txt
hello
```

Figure 4: Demonstrating that the '>' operator overwrites the output file

9. **$ echo world >> file1.txt**
   >> redirects the standard output and append to a file.

10. **$ cat file1.txt**
    Now, the content of the file includes both "hello" and "world".

```
root@edison:~/directory1# cat file1.txt
hello
world
```

Figure 5: Bash after successfully issuing the command cat

11. To ***copy*** a file, use the command **cp**.

    The format for this command is as follows:

    cp <source_file> <destination_file>

    $ cp file1.txt file2.txt

    For more information, please refer to the below reference:
    http://www.linfo.org/cp.html

12. **$ cat file2.txt**
    You can see that the content of file2.txt is the same as that of file1.txt.

```
root@edison:~/directory1# cat file2.txt
hello
world
```

Figure 6: Bash after successfully issuing the command cat

13. In a GUI based file system viewer such as Finder on Mac or Explorer on Windows, you can visually see all files in a directory.

    To do the same in Bash, you must issue a ***list***, or **ls** (LS) command. Issue the below command to view the contents of the current working directory.

    $ ls

    For more information, please refer to the below reference:
    http://linuxcommand.org/man_pages/ls1.html

14. **$ ls –l**
    (LS -L) "-l" is an option for **ls** command. With this option, **ls** command uses a long listing format. The displayed output includes the permission, the owner, the size, the modified time, and the name of each file or directory.

```
root@edison:~/directory1# ls
file1.txt  file2.txt
root@edison:~/directory1# ls -l
-rw-r--r--    1 root     root            12 Sep 18 00:44 file1.txt
-rw-r--r--    1 root     root            12 Sep 18 01:07 file2.txt
```

Figure 7: Bash output after issuing the commands "ls" and "ls -l"

15. As you can see, you could be issuing the same command over and over again. To help users save time, the shell has a convenient **command-line history** tool. To access this, press the up arrow key to look at previous commands, and press the down arrow key to look for more recent commands. Press [Enter] to execute them or use the left and right arrow keys to edit the command first.

    Follow the below steps to better understand this tool.

    Press the up-arrow key once. **DO NOT PRESS [Enter]**.
    The command "**ls -l**", is displayed in the shell prompt.

    Press the up-arrow key twice. **DO NOT PRESS [Enter]**.
    The command "**cat file2.txt**" is displayed in the shell prompt.

    Press the down-arrow key once. **DO NOT PRESS [Enter]**.
    The command "**ls**" is displayed in the shell prompt. **Press [Enter]**.

16. Sometimes, you may not remember the exact name of a file. To help users with this, the shell has a convenient **auto-complete** tool. To access this, type some portion of the prefix of the command you wish to issue, and press **[Tab]**.

    Type "**cat file1**" into the shell prompt. **DO NOT PRESS [Enter]**,

    Press [Tab].
    The shell prompt now displays "**cat file1.txt**".
    Press [Enter].

    Type **"cat file**" into the prompt. **DO NOT PRESS [Enter]**.
    Press [Tab] twice **without pressing [Enter]**.

    This displays file1.txt and file2.txt.

    Press [Tab] once **without pressing [Enter]**. This will list the all possible completions that start with "file" again. **Press [Enter]**.

17. **$ cd ..**
    This command lets you navigate to the parent of your current working directory. "." means the current working directory and ".." means the parent of your current working directory.

```
root@edison:~/directory1# cd ..
root@edison:~#
```

Figure 8: Navigating to the parent directory by issuing the command "cd .."

18. **$ mkdir directory2**

19. The command to ***move*** *or rename* a file or a directory is **mv**.

    Follow the below steps to develop an understanding of how **mv** works.

    $ echo foo > foo.txt
    $ cat foo.txt
    Notice how foo.txt contains "foo".
    $ mv foo.txt bar.txt
    $ cat foo.txt
    Notice how the shell prompt returns an error message.
    This is because the file named **foo.txt** is now named **bar.txt**.
    $ cat bar.txt

Notice how the shell prompt returns “foo”. This is what was contained in foo.txt before.

```
root@edison:~#  echo foo > foo.txt
root@edison:~#  cat foo.txt
foo
root@edison:~#  mv foo.txt bar.txt
root@edison:~#  cat foo.txt
cat: foo.txt: No such file or directory
root@edison:~#  cat bar.txt
foo
```

Figure 9: Bash after issuing each of the above commands

$ mv directory1/* directory2
mv command moves files to the specified destination. * is a wildcard character that represents zero or more characters. Thus,

directory1/* means everything in **directory1/** *including subfolders*

BE VERY CAREFUL USING THE WILDCARD CHARACTER

YOU CAN POTENTIALLY MODIFY OR DELETE EVERY FILE ON YOUR SYSTEM AT ONCE

$ ls directory1
This command lists the contents of the directory directory1. As you can see, nothing is listed because the contents of this directory were moved to directory2.

$ ls directory2

```
root@edison:~# ls directory1
root@edison:~# ls directory2
file1.txt  file2.txt
```

Figure 10: Contents of directory1 and directory2

20. **$ cd directory2**

21. The command to *delete or **rem**ove* files or directories is **rm**. **BE VERY CAREFUL**

    UNDOING A DELETE IN LINUX IS VERY DIFFICULT

    YOU MAY NEVER BE ABLE TO RECOVER A DELETED FILE IN LINUX

    $ ls
    $ rm file1.txt
    $ ls

Notice how file1.txt is no longer present in the list of files.

```
root@edison~/directory2# ls
file1.txt  file2.txt
root@edison~/directory2# rm file1.txt
root@edison~/directory2# ls
file2.txt
```

Figure 11: Contents of directory2 after deleting file1.txt

To delete a folder, add the option **-r**.

$ mkdir test1
$ ls
$ rm test1
Notice you get an error
$ rm -r test1
$ ls

```
root@edison~/directory2# mkdir test1
root@edison~/directory2# ls
file2.txt  test1
root@edison~/directory2# rm test1
rm: cannot remove `test1': Is a directory
root@edison~/directory2# rm -r test1
root@edison~/directory2# ls
file2.txt
root@edison~/directory2#
```

Figure 12: Contents of directory2 after deleting test1/

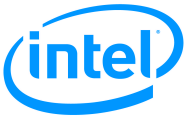
$ cd
$ rm -r directory1
$ rm -r directory2

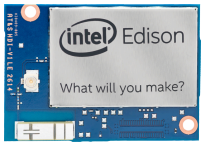
22. To reboot the Intel Edison, issue the below command.

    $ reboot

    Please keep all cables connected while rebooting. You will need to log in again once the reboot is complete.

23. To shut the Intel Edison down, issue the below command.





Wait until the system is completely off, then disconnect the power and USB cables.
$ shutdown -h now