IT WORKSHOP
GXESL 208

Lab Worksheet

Semester 2
Bachelor of Technology
Applied Electronics & Instrumentation Engineering
Electronics & Communication Engineering
College of Engineering
Trivandrum

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM
KERALA

2025

List of Experiments

Syllabus

Preface

APJ Abdul Kalam Technological University, Kerala has introduced the course `IT Workshop (GXESL 208)` in first year of the B.Tech Degree programme. This worksheet has been prepared for use in the workshop sessions of the course at Department of Electronics & Communication Engineering, College of Engineering Trivandrum.

These worksheets are in the process of active development and updated versions may be found on https://jim79.github.io/it-workshop/. Feedback/suggestions may be shared via email at *arun.varghese@cet.ac.in* or *joaquim@cet.ac.in*

Dr. Arun Varghese
Joaquim Ignatious Monteiro

Experiment 1

Lab Safety

This section discusses safety in the lab. Safety guidelines help protect individuals from accidents and injury. They also help to protect equipment from damage.

General Safety

Safe working conditions help prevent injury to people and damage to computer equipment. A safe workspace is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.

Follow the basic safety guidelines to prevent cuts, burns, electrical shock, and damage to eyesight. As a best practice, make sure that a fire extinguisher and first-aid kit are available in case of fire or injury. Poorly placed or unsecured cables can cause tripping hazards in a network installation. Cables should be installed in conduit or cable trays to prevent hazards.

This is a partial list of basic safety precautions to use when working on a computer:

- Remove your watch and jewelry and secure loose clothing.
- Turn off the power and unplug equipment before performing service.
- Cover sharp edges inside the computer case with tape.
- Never open a power supply or a CRT monitor.
- Do not touch areas in printers that are hot or that use high voltage.
- Know where the fire extinguisher is located and how to use it.
- Keep food and drinks out of your workspace.
- Keep your workspace clean and free of clutter.
- Bend your knees when lifting heavy objects to avoid injuring your back.

Electrical Safety

Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

Fire Safety

Follow fire safety guidelines to protect lives, structures, and equipment. To avoid an electrical shock and to prevent damage to the computer, turn off and unplug the computer before beginning a repair.

Fire can spread rapidly and be very costly. Proper use of a fire extinguisher can prevent a small fire from getting out of control. When working with computer components, be aware of the possibility of an accidental fire and know how to react. Be alert for odors emitting from computers and electronic devices. When electronic components overheat or short out, they emit a burning odor.
If there is a fire, follow these safety procedures:

- Never fight a fire that is out of control or not contained.
- Always have a planned fire escape route before beginning any work.
- Get out of the building quickly.
- Contact emergency services for help.
- Locate and read the instructions on the fire extinguishers in your workplace before you have to use them.

Experiment 2

Computer Hardware Familiarization

JOAQUIM IGNATIOUS MONTEIRO

Instructions

1. All students should refer to the e-resources shared on this web link https://jim79.github.io/it-workshop/ and come prepared for the workshop session.
2. All Students should write all the questions from worksheet in the rough record before coming to the workshop session.
3. Answers to the above questions are to be found out during the workshop session and written in the rough record.
4. Rough record has to be completed during the workshop session.
5. Student should get the rough record verified and signed by the faculty before leaving the lab.

Hardware Familiarization exercises

Students are to complete the tasks in following sections to familiarize with computer hardware.

2.1 CPU Box (Chassis)

CPU Box is the enclosure that houses all the computer components, providing protection and structural support.

Tasks:

1. List the typical materials used for constructing a CPU chassis.
2. Identify the form factor of the CPU case provided to you.
3. Identify the types of form factors supported by different chassis.
4. List the cooling mechanisms commonly integrated into a CPU chassis.

2.2 Motherboard

A motherboard is the main circuit board in a computer system. It connects all of the internal components, like the memory, processor, graphics card and other hardware. It also provides power to each component and allows them to communicate with each other.

Tasks:

1. Identify your motherboard model.
   Tip: Motherboard model is usually printed on the motherboard. It can be located in several possible locations; for example, it may be printed near the RAM slots, near the CPU socket, or between the PCI slots.

2. Google your motherboard model number and list the technical specifications of motherboard, including form factor, chipset, and socket type.

3. Identify the types and number of RAM slots available.

4. Describe the input/output ports provided on the motherboard.

5. Identify components on the motherboard provided and draw a layout (similiar to the figure given below) in your rough record.

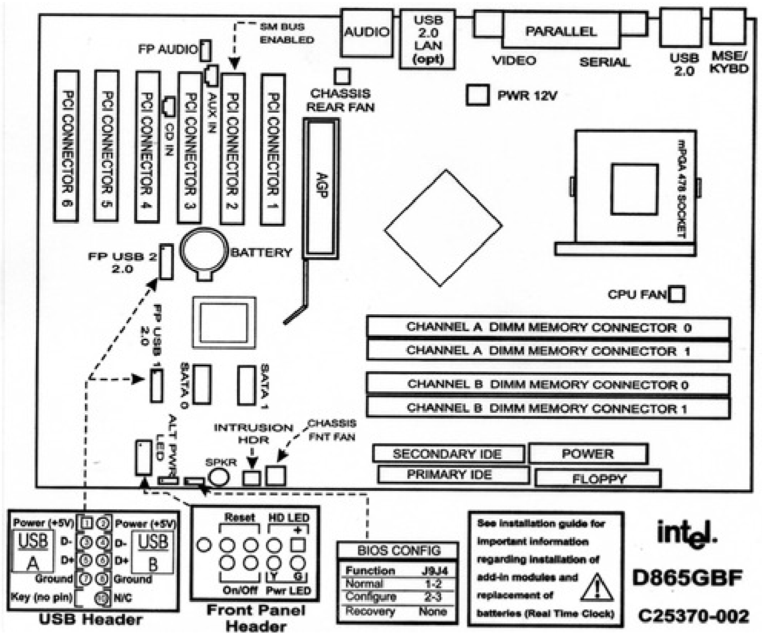


Figure 2.1:

6. Refer to the website https://motherboarddb.com/motherboards/ choose a motherboard manufactured after the year 2010 and list the following features :

- Manufacturer
- Year of manufacture
- Form factor
- Chipset
- Memory
- Number and type of USB ports
- Video outputs
- Network ports
- Audio ports
- Audio chipset
- Expansion slots
- Power connectors

2.3 CPU and Chipset

The Central Processing Unit (CPU) performs most of the processing inside a computer, while the chipset manages data flow between the processor, memory, and peripherals.

Tasks:

1. List the key specifications of a CPU, including clock speed, core count, and cache size.
2. Explain the terms power consumption and thermal design power (TDP) of a CPU.
3. Refer to the website https://www.techpowerup.com/cpu-specs/ choose a CPU manufactured after the year 2010 and list the following features :
    - Manufacturer
    - Year of manufacture
    - Socket
    - Process Size
    - Frequency
    - Number of Cores
    - Cache
    - Memory Support
    - TDP
    - Minimum Power
    - Market
    - Production Status

2.4 Storage Devices

2.4.1 Hard Disk Drive

Hard Disk Drive (HDD) is an electro-mechanical data storage device that stores and retrieves digital data using magnetic storage with one or more rigid rapidly rotating platters coated with magnetic material. HDDs are a type of non-volatile storage, retaining stored data when powered off. Modern HDDs are typically in the form of a small rectangular box.

Tasks:

1. List the technical specifications of the hard disk provided to you including storage capacity, RPM, and interface type.
2. Explain the differences between SATA, SAS, and IDE hard disks.
3. Refer to the website https://smarthdd.com/database/ choose a Hard Disk Drive (HDD) and list the following features :
   - Manufacturer
   - Model
   - Capacity
   - Interface
   - Maximum interface speed
   - Maximum read speed

2.4.2 Solid State Drive

A solid-state drive (SSD) is a type of solid-state storage device that uses integrated circuits to store data persistently.

1. Refer to the website https://smarthdd.com/database/ choose a Solid State Drive (SSD) and list the following features :
   - Manufacturer
   - Model
   - Capacity
   - Interface
   - Maximum interface speed
   - Maximum read speed
2. Compare the features of the HDD's and SSD's.

2.5 Interface Cards

Expansion cards like graphics cards, sound cards, and network cards provide additional functionalities to the computer.

Tasks:

1. List the technical specifications of a graphics card, including VRAM, clock speed.
2. Identify the ports available on interface cards, such as HDMI, DisplayPort, or audio jacks.
3. Describe the cooling mechanisms for interface cards.
4. Refer to the website https://www.techpowerup.com/gpu-specs/ choose a GPU manufactured in the year 2003 and list the following features :
   - Manufacturer
   - Year of manufacture
   - GPU Name
   - GPU Clock
   - Memory Size
   - Graphics Features
   - Bus Interface
   - Production Status
5. Refer to the website https://www.techpowerup.com/gpu-specs/ choose a GPU manufactured after the year 2020 and list the following features :
   - Manufacturer
   - Year of manufacture
   - GPU Name
   - GPU Clock
   - Memory Size
   - Graphics Features
   - Bus Interface
   - Production Status

2.6 Card Slots

Slots on the motherboard such as PCI and PCIe where interface cards are inserted.

Tasks:

1. List the types of card slots available on a motherboard and their respective uses.

2.7 Cables

Various cables like SATA, IDE, and power cables that connect internal components and peripherals.

Tasks:

1. Refer to the website https://www.cablestogo.com/learning/connector-guides/internal and list the types of cables used in a computer and their specific purposes.
2. Describe the data transfer rates of different generations of SATA cables.

2.8 SMPS (Switch Mode Power Supply/PSU)

Converts AC power from the mains to DC power used by the computer's internal components.

Tasks:

1. List the power ratings and efficiency certifications of SMPS units.
2. Identify the types of connectors provided by an SMPS for various components.
3. Describe the cooling mechanisms and protections (e.g., overvoltage) in an SMPS.
4. Refer to the website https://www.cybenetics.com/index.php?option=power-supplies choose a Power Supply Unit (PSU) and list the following features :
   - Manufacturer
   - Form factor
   - Wattage
   - Efficiency rating

2.9 NIC (Network Interface Card)

Allows the computer to connect to a network and communicate with other devices.

Tasks:

1. List the technical specifications of a NIC, including speed and connection type.
2. Describe the difference between wired and wireless NICs.
3. Refer to the website https://www.scan.co.uk/shop/computer-hardware/network-cards-accessories/rj45-network-cards choose a network card and list the following features :

- Manufacturer
- Interface
- Supported data rates

2.10 Various Ports

Description: Includes USB, HDMI, Ethernet, and audio ports used to connect external devices.

Tasks:

1. Refer to the website https://newnex.com/usb-connector-type-guide.php and list the technical specifications of USB ports, including version and data rates.
2. Refer to the website https://www.xenarc.com/different-types-of-monitor-ports.html and list the common display ports in a computer and their typical applications.

2.11 I/O Devices

Description: Input devices like keyboards and mice, and output devices like monitors and printers.

Tasks:

1. List the specifications of common input devices, such as DPI for mice or key travel for keyboards.
2. Refer to the website https://www.displaydb.com/brands and list the following features of computer monitors :
   - Brand
   - Model
   - Size
   - Panel
   - Refresh rate
   - Screen aspect ratio
   - Screen resolution
3. List the connectivity options available for computer printers.

2.12 Buses

Description: Electrical pathways like data bus, address bus, and control bus that transfer data between components.

Tasks:

1. Explain the function of the address bus, data bus and control bus in a computer system.

2.13 Firmware

Description: Software programmed into read-only memory, providing low-level control for the device's hardware.

Tasks:

1. List the features of BIOS and UEFI firmware.
2. Explain how firmware updates improve hardware functionality.
3. Describe the role of firmware in initializing hardware during startup.

Experiment 3

Familiarizing basic Unix/Linux commands

DR.ARUN VARGHESE

Instructions:

- Each command in this handout are to by typed at the terminal (not copy-pasted)
- Write down each command and its output(shorten, if needed) in your rough record
- You will be asked to submit your rough record and obtain signature from faculty

The Shell

- The shell is a program that takes keyboard commands and passes them to the operating system to carry out.
- Almost all Linux distributions supply a shell program from the GNU Project called `bash`.
- The name is an acronym for **bourne-again shell**, a reference to the fact that `bash` is an enhanced replacement for `sh`, the original Unix shell program.
- We interact with the shell through the terminal.

3.1 pwd

- Print Working Directory: displays the current working directory
- When we first log in to our system (or start a terminal emulator session), our current working directory is set to our home directory.
- Each user account is given its own home directory, and it is the only place a regular user is allowed to write files.

File System Hierarchy

- Like Windows, Linux organizes its files in a hierarchical directory structure.
- The first directory in the file system is called the root directory (denoted by a /).
- Linux always has a single file system tree. Storage devices are attached (or more correctly, mounted) at various points on the tree

3.2 ls

- List the contents of of a directory
- List the contents of the root directory: `ls /`
- Listing the contents of the current directory: `ls`
- We can specify the directory to list: `ls /usr`
- We can specify multiple directories: `ls ~ /usr`
- The ∼ represents the home directory.
- Use the `-l` option to reveal more detail: `ls -l /usr`
- `ls` can also be used to know details of files: `ls -l filename`
- Example output:

```
-rw-r-r- 1 user1 workshop 573 Sep 29 22:00 foo
```

- `-t` option sorts by time modified: `ls -t`
- We can combine options: `ls -lt`
- `-a` option lists all files, even those with names that begin with a `.` (period), which are normally hidden: `ls -a`
- The `ls` command has many other options

3.3 cd

- Change our working directory:
- `cd <pathname>`
- We can specify pathnames in one of two different ways: as absolute pathnames or as relative pathnames.

- Absolute pathnames begin with the root directory (represented by `/`):For example, `cd /usr/bin; pwd`
- A relative pathname starts from the working directory.
- Special notations to represent relative positions:
  - `.` refers to the current directory
  - `..` refers to the parent directory
- From `/usr/bin` change to `/usr`: `cd ..`
- Change back to `/usr/bin`: `cd ./bin`
- We can omit the `./`: `cd bin`

Exercise:

1. `cd` to `/sys/bus/cpu/devices`
2. List the contents of the folder using `ls`
3. From `devices`, `cd` to `/sys/bus/cpu` using relative path
4. From `cpu`, list the contents of `/sys/devices` using relative path
5. Change back to your home directory

3.4 file

- Print a brief description of a file's contents: `file filename`
- `file /usr/bin/apt`

There are many kinds of files. In fact, one of the common ideas in Unix-like operating systems such as Linux is that "everything is a file".

3.5 less

- View text file's contents: `less filename`
- `less /etc/passwd`
- To exit less, press q

The less program was designed as an improved replacement of an earlier Unix program called more. The name less is a play on the phrase "less is more".

3.6 mkdir and rmdir

- Create and remove directories
- `mkdir directory...`
- `mkdir dir1`: Create the directory `dir1`.
- `mkdir dir1 dir2 dir3`: Create directories `dir1`, `dir2`, and `dir3`.
- **rmdir** : deletes an empty directory.

Exercise:

1. Create a folder named `ec` in your home folder.
2. Under `ec`, create four folders named `y1`, `y2`, `y3`, `y4` without changing from your home directory.
3. Under `y1`, create folders `ec1` and `ec2`.
4. Verify your folder structure using the command: `tree ec`.
5. Delete all folders created.

3.7 cp

Copy files and folders

Command	**Results**
cp file1 file2	Copy file1 to file2. If file2 exists, it is overwritten with the contents of file1. If file2 does not exist, it is created.
cp -i file1 file2	Same as previous command, except that if file2 exists, the user is prompted before it is overwritten.
cp file1 file2 dir1	Copy file1 and file2 into directory dir1. The directory dir1 must already exist.
cp dir1/* dir2	Using a wildcard, copy all the files in dir1 into dir2. The directory dir2 must already exist.
cp -r dir1 dir2	Copy the contents of directory dir1 to directory dir2. If directory dir2 does not exist, it is created and, after the copy, will contain the same contents as directory dir1. If directory dir2 does exist, then directory dir1 (and its contents) will be copied into dir2.

3.8 mv

Move and rename files

Command	Results
mv file1 file2	Move `file1` to `file2`. If `file2` exists, it is overwritten with the contents of `file1`. If `file2` does not exist, it is created. In either case, `file1` ceases to exist.
mv -i file1 file2	Same as the previous command, except that if `file2` exists, the user is prompted before it is overwritten.
mv file1 file2 dir1	Move `file1` and `file2` into directory `dir1`. The directory `dir1` must already exist.
mv dir1 dir2	If directory `dir2` does not exist, create directory `dir2` and move the contents of directory `dir1` into `dir2` and delete directory `dir1`. If directory `dir2` does exist, move directory `dir1` (and its contents) into directory `dir2`.

3.9 rm

- Remove(delete) files and directories
- `rm item ...`
- To delete directories, use the `-r` (or `-recursive`) option.

Exercise:

1. In your home directory, create a folder named `temp`.
2. Change directory to `temp`.
3. Create two directories `dir1` and `dir2` in `temp`.
4. Copy the `passwd` file from the `/etc` folder to the `temp` folder using the `-v` option.
5. Once again, copy the `passwd` file from `/etc` to `temp`, this time using the `-i` option.
6. Rename the `passwd` file in `temp` to `fun`.
7. Move `fun` to `dir1`.
8. Move `dir1` to `dir2`.
9. Delete `dir2`.

10. Delete `temp`.

Wildcards Using wildcards (which is also known as globbing) allows you to select filenames based on patterns of characters.

Pattern	Matches
*	All files
g*	Any file beginning with g
b*.txt	Any file beginning with b followed by any characters and ending with .txt
Data???	Any file beginning with Data followed by exactly three characters

3.10 alias

- Aliases are commands that we can define ourselves, built from other commands.
- `alias name='string'`
- `alias foo='cd /usr; ls; cd -'`
- `alias rm='rm -i'`
- To remove an alias, the `unalias` command is used, like so:
    - `unalias foo`

3.11 type

- Display a command's type
- `type command`
- Examples:
- `type type`
- `type ls`
- `type cp`

3.12 which

- Display an executable's location
- **which** command
- Examples:
- `which ls`
- `which` works only for executable programs, not builtins or aliases
- `which cd`

3.13 man

- Display manual page
- `man mkdir`
- Many executable programs support a `-help` option that displays a description of the command's supported syntax and options
- `ls -help`

3.14 IO Redirection

stdout, stderr, and stdin:

- Keeping with the Unix theme of "everything is a file," programs such as `ls` actually send their results to a special file called standard output (often expressed as `stdout`) and their status messages to another file called standard error (`stderr`).
- By default, both standard output and standard error are linked to the screen and not saved into a disk file.
- In addition, many programs take input from a facility called standard input (`stdin`), which is, by default, attached to the keyboard.
- **Redirecting stdout:**
- `ls /usr/bin > output.txt`
- `less output.txt`
- **Overwrite:**

- `> output.txt`
- `less output.txt`
- **To append:** »
- **Redirecting stderr:**
- `ls /bin/usr`
- `ls /bin/usr 2> output.txt`
- **For redirecting stdin**: <

3.15 cat

- Shows the contents of a file, all at once
- `cat filename`
- Example: `cat /etc/os-release`
- Because `cat` can accept more than one file as an argument, it can also be used to join files together.

3.16 grep

- Used to find a specific string within files
- `grep pattern filename`
- Let us create a text file to search:
- `ls /usr/bin > out.txt`
- Search for pattern `zip` in the file:
- `grep zip out.txt`
- This gives all programs in `/usr/bin` that had the name `zip` in it.

3.17 wc

- Used to display the number of lines, words, and bytes contained in files.
- The `-l` option limits its output to report only lines.
- Example: Count the number of executable programs in `/usr/bin`:
- `ls /usr/bin > out.txt`
- `wc -l out.txt`

3.18 |(pipe)

- Using the pipe operator `|`, the output of one command can be piped into the input of another.
- The pipe can be used to link commands together to perform more complex tasks that would otherwise take multiple steps (and possibly writing information to disk).
- Example: Count the number of executable programs in `/usr/bin`:
- `ls /usr/bin | wc -l`
- Example: Find all programs in `/usr/bin` that has the name `zip` in it:
- `ls /usr/bin | grep zip`

The Difference Between > and |

- The redirection operator > connects a command with a file, while the pipeline operator `|` connects the output of one command with the input of a second command.
- `command1 > file1`
- `command1 | command2`

3.19 echo

- prints its text arguments on standard output
- `echo this is a test`
- Using `echo` to create a simple text file:
- `echo 'sample text' > myfile.txt`

3.20 touch

- modifies the time-stamp of a file
- `touch filename`
- If the file does not exist, `touch` creates the file.
- So touch can be used to create files, though it is not its primary purpose

3.21 Text Editors

- Text editors fall into two basic categories: graphical and text-based.
- `gedit` is a graphical editor.
- Popular text-based editors are `nano`, `vi`, and `emacs`.
- The `nano` editor is a simple, easy-to-use editor while `vi` and `emacs` are more powerful editors.
- `nano myfile`
- The file will open or one will be created if it doesn't exist
- **Exercise:** Use `nano` to open a file, make edits, save it and exit.

3.22 printenv

- The shell maintains a body of information during our shell session called the environment.
- To see all the environment variables and their values:
- `printenv`
- `printenv` can also print the value of a specific variable:
- `printenv HOME`
- `printenv PATH`
- When we type a command in the shell, it searches the list of directories in the `PATH` variable.

3.23 history

- bash maintains a history of commands that have been entered. This list of commands is kept in your home directory in a file called $.bash\_history$
- To view command history:
- `history`
- In this listing, each command is numbered.
- If you want to reissue a command at line number 20:
- `!20`
- The `!` can also be followed by a string to repeat the last history list item starting with that string. Example: `!c`

3.24 chmod

- change permissions for a file or directory
- Octal or symbolic representation:
- `> foo.txt`
- `ls -l foo.txt`
- `chmod 600 foo.txt`
- `ls -l foo.txt`
- `chmod +x`

Exercise:

1. Create a file with some text in it.
2. Check its permissions.
3. Change its permissions to `r__r__r`.
4. Try to make changes to the file.
5. Try to delete the file using `rm`

3.25 Interrupting programs and signalling EOF

- Use `Ctrl-C` to stop running a command:
- `sleep 60`
- `Ctrl-C`
- `sleep 60; echo hi`
- `Ctrl-C`
- Use `Ctrl-D` to indicate EOF.

3.26 passwd

- To change a user password:
- `passwd`
- To change another user's password:
- `passwd user2`

3.27 useradd

- Creates a new user account
- To add a new user named `user2`:
- `sudo useradd -m user2`
- The `-m` option creates a home directory for the new user automatically.
- After creating a new user, you typically need to set a password for the user account using the `passwd` command:
- `sudo passwd user2`
- To switch to `user2`:
- `sudo -u user2 bash`
- `whoami`
- `cd`
- `exit`

- To delete `user2`:
- `sudo userdel user2`
- This does not delete the home directory `/home/user2`

3.28 uname

- Displays information about a Linux machine's operating system and hardware platform.
- `uname -a`
- Prints:
  - Kernel name
  - Hostname
  - Kernel release
  - Kernel version and the build date
  - Machine architecture name
  - CPU type
  - Hardware platform
  - Operating system

3.29 time

- Measures the execution time of a specified command or program and reports various metrics, including real, user, and system time.
  - Real Time: The actual elapsed time, from start to finish, including time spent waiting for I/O and other processes.
  - User Time: The amount of CPU time spent executing user-mode instructions within the process.
  - System Time: The amount of CPU time spent executing system-level instructions on behalf of the process.
- `time ls -l`
- `time mv file1 dir1`

3.30 du

- `du [options] [file/directory]`
- The `du` (disk usage) command is used to estimate and summarize file and directory space usage indicated by the given path name.
- If you do not specify a path name, `du` assumes the current directory.
- By default, `du` gives disk usage in KB.
- `-h` option gives the output in a more human-readable form (e.g., KB, MB):
- `du -h dir1`

3.31 Installing and maintaining software

High-level tools like apt permit a package to be downloaded from a repository and installed with full dependency resolution

Command	Description
`apt update`	Update all repository info
`apt install package_name`	Install a package
`apt upgrade`	Update all installed packages
`apt autoremove`	Remove packages that are no longer needed
`apt remove package_name`	Remove an installed package
`apt purge package_name`	Remove an installed package and delete configuration files
`apt search string`	Search repositories for a package
`apt show package_name`	Show details for a package

- If a package file has been downloaded from a source other than a repository, it can be installed directly
- `sudo dpkg -i package_name`

3.32 Comperssing Files

- The `gzip` program is used to compress one or more files. When executed, it replaces the original file with a compressed version of the original.
  - `ls -l /etc > foo.txt`
  - `ls -l foo.txt`
  - `gzip foo.txt`
  - `ls -l foo.*`

- The corresponding `gunzip` program is used to restore compressed files to their original, uncompressed form.
  - `gunzip foo.txt.gz`
- The `bzip2` program is similar to `gzip` but uses a different compression algorithm that achieves higher levels of compression at the cost of compression speed.
- In most regards, it works in the same fashion as `gzip`.
- A file compressed with `bzip2` is denoted with the extension `.bz2`.

3.33 Archiving

- Archiving is the process of gathering up many files and bundling them together into a single large file. The `tar` program is the classic tool for archiving files.
- `tar <mode and options> <archive> <files or locations>`
  - c: create an archive
  - v: verbose
  - f: specify name of archive file
  - x: extract an archive
- `tar -cvf temp.tar temp`
- By default, `tar` extracts all the components to the current directory. To indicate where to extract the components, add the `-C` option and specify the path:
- `tar -xvfC <archive_name> <path>`
- **Creating a Tar Archive with Compression**:
  - `z`: use gzip compression
  - `j`: use bzip2 compression
  - `tar -cvzf temp.tgz temp`
- **Extracting a compressed archive**:
  - `tar -xvf temp.tgz`
- The `tar` command will auto-detect compression type and will extract the archive.

3.34 fdisk

- The `fdisk` command in Linux is used to manage disk partitions. It allows you to create, delete, resize, and modify partitions on your hard drive.
- Listing Partitions
  - `fdisk -l`
- Selecting a Disk
  - `sudo fdisk /dev/sdX`
  - Replace `sdX` with the appropriate disk identifier (e.g., `sda`, `sdb`).
- Creating a New Partition
  - Enter `n` to create a new partition.
  - Choose the partition type (primary or extended).
  - Specify the partition number and size.
- Deleting a Partition
  - Enter `d` to delete a partition.
  - Specify the partition number to delete.
- Saving Changes
  - Enter `w` to write the changes to the disk.
- Quitting Without Saving
  - Enter `q` to quit without saving any changes.
- Always be cautious when using `fdisk`, as making changes to disk partitions can result in data loss if not done carefully.

3.35 dmesg

- The `dmesg` command is a Linux utility that displays kernel-related messages retrieved from the kernel ring buffer.
- The ring buffer stores information about hardware, device driver initialization, and kernel module messages during system startup.
- The `dmesg` command is handy for diagnosing hardware and kernel-related issues.

Experiment 4

Familiarization of Boot process

JOAQUIM IGNATIOUS MONTEIRO

In this experiment, we try to understand the boot process in a computer. After studying the material geiven below, perform experiment and record answers to the questions given in the lab record.

4.1 Steps in computer booting

- Power-On Self Test (POST)
- Locate and Load Bootloader
- Load Operating System Kernel
- Initialize System Processes
- User Login & Desktop Environment

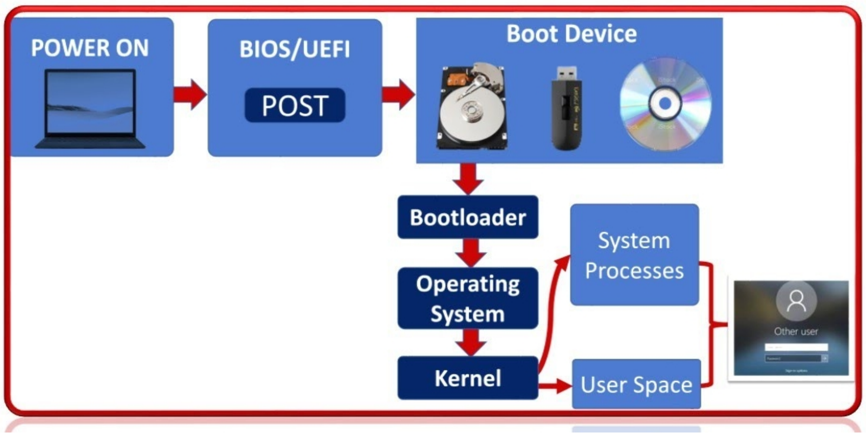


Figure 4.1: Steps in the booting process of a computer (Image Source :[1])

4.2 POST

A Power-On Self-Test (POST) is a series of diagnostic tests that a computer performs when it is turned on to ensure that all of its components are working properly.
A POST checks for the proper working of the CPU and RAM, Input and Output devices, motherboard and any other hardware components that are essential to the computer's operation

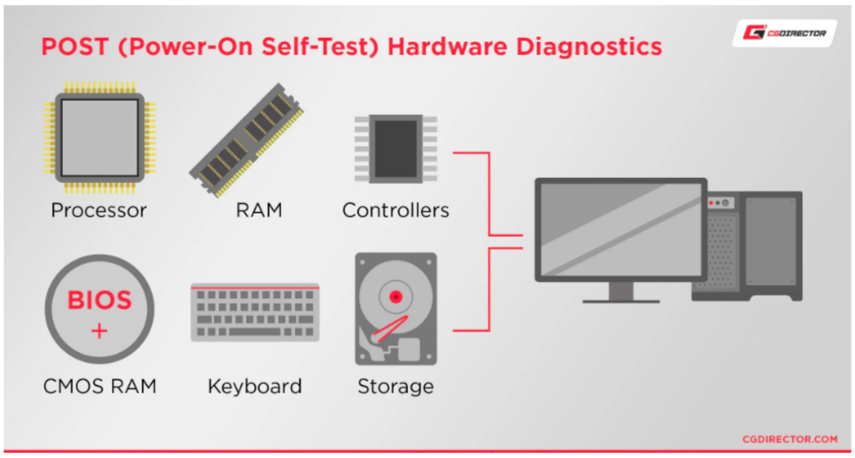
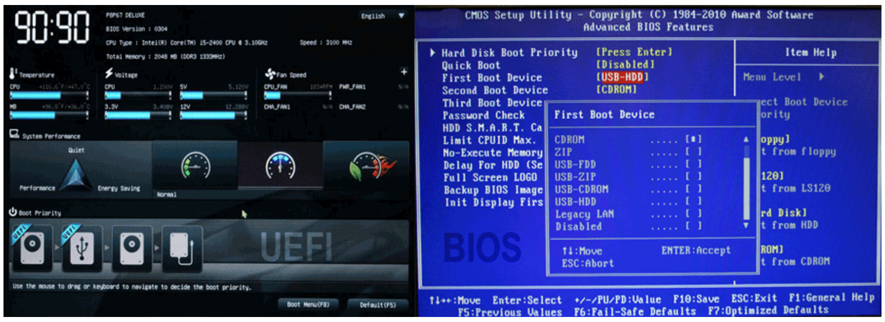


Figure 4.2: Power On Self Test (POST) in a computer (Image Source :[2])

Firmware on a Computer

- There are two types of firmware in a computer:
  - **BIOS (Basic Input/Output System)** – Found in older computer hardware, now referred to as **legacy BIOS**.
  - **UEFI (Unified Extensible Firmware Interface)** – Used in newer computer hardware.
- Many UEFI systems support **CSM (Compatibility Support Module)**, allowing users to boot older operating systems that do not support UEFI.
- Firmware functions:
  - Initializes essential hardware components.
  - Searches for a bootable device (HDD, SSD, USB drive, network, etc.).



```
90:90   P8P67 DELUXE                                              English
        BIOS Version : 0304
        CPU Type : Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz     Speed : 3100 MHz
        Total Memory : 2048 MB (DDR3 1333MHz)

Temperature                Voltage                          Fan Speed
CPU  +116.6°F/+47.0°C      CPU   1.200V    5V   5.120V      CPU_FAN  1034RPM   PWR_FAN1  N/A
MB   +96.8°F/+36.0°C       3.3V  3.408V    12V  12.288V     CHA_FAN1  N/A      CHA_FAN2  N/A

System Performance
  Quiet   Performance   Energy Saving   Normal

Boot Priority
Use the mouse to drag or keyboard to navigate to decide the boot priority.
                                                   Boot Menu(F8)   Default(F5)
```

```
        CMOS Setup Utility - Copyright (C) 1984-2010 Award Software
                          Advanced BIOS Features

 ▶ Hard Disk Boot Priority    [Press Enter]              Item Help
   Quick Boot                 [Disabled]
   First Boot Device          [USB-HDD]              Menu Level   ▶
   Second Boot Device         [CDROM]
   Third Boot Device                                 ect Boot Device
   Password Check     First Boot Device              ority
   HDD S.M.A.R.T. Ca
   Limit CPUID Max.   CDROM        ..... [■]         oppy]
   No-Execute Memory  ZIP          ..... [ ]         t from floppy
   Delay For HDD (Se  USB-FDD      ..... [ ]
   Full Screen LOGO   USB-ZIP      ..... [ ]         120]
   Backup BIOS Image  USB-CDROM    ..... [ ]         t from LS120
   Init Display Firs  USB-HDD      ..... [ ]
                      Legacy LAN   ..... [ ]         rd Disk]
                      Disabled     ..... [ ]         t from HDD
   BIOS
                      ↑↓:Move        ENTER:Accept    ROM]
                      ESC:Abort                      t from CDROM

 ↑↓→←:Move  Enter:Select  +/-/PU/PD:Value  F10:Save  ESC:Exit  F1:General Help
     F5:Previous Values  F6:Fail-Safe Defaults  F7:Optimized Defaults
```

Figure 4.3: Two types of Firmware in a PC : UEFI and BIOS(Image Source :[3])

Accessing the BIOS (Basic Input/Output System)

- Turn off your computer completely.
- Turn it back on and immediately start pressing the BIOS key repeatedly.
- The BIOS/UEFI setup screen should appear.

Common BIOS Keys

- **Dell, Lenovo (Laptops & Desktops)** → F2
- **HP** → Esc or F10
- **Acer** → F2
- **ASUS** → F2 or Del
- **MSI** → Del
- **Gigabyte, ASRock (Custom Builds)** → Del or F2
- **Sony VAIO** → F2 or Assist button
- **Toshiba** → F2 or F12

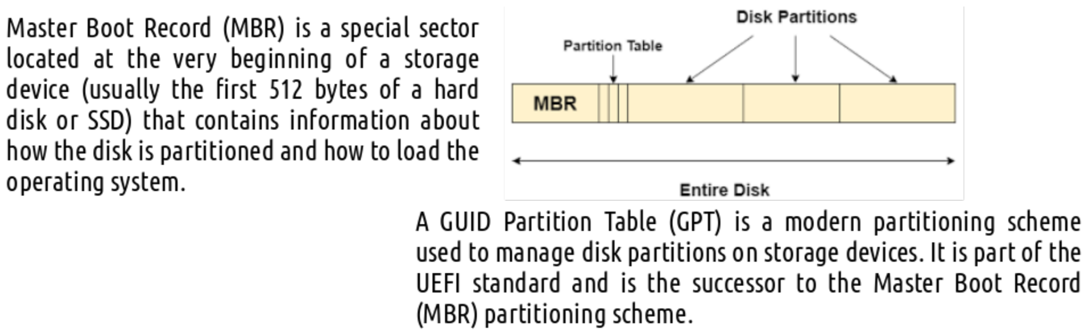
4.3 Locate and Load Bootloader

The firmware (BIOS or UEFI) identifies the partitioning Scheme ie, it determines whether the disk uses Master Boot Record (MBR) or the GUID Partition Table (GPT) format :

- **Master Boot Record (MBR)** – Used by legacy BIOS.
  - The first 512 bytes of the storage device contain the MBR.

  - It includes the partition table and the bootloader.
  - Can support only up to 2 TB of storage and a maximum of 4 primary partitions.

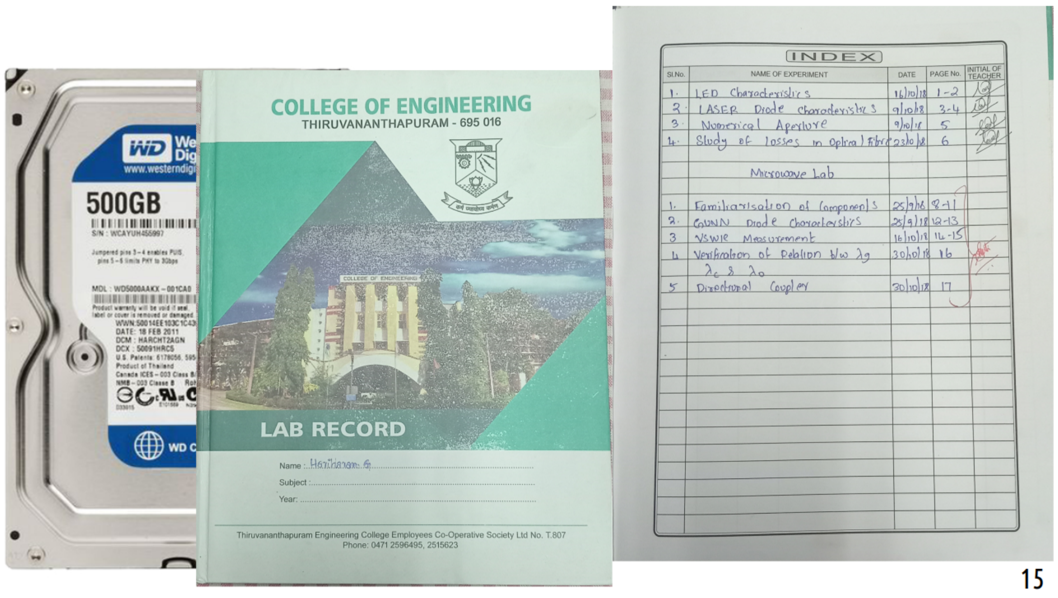
- **GUID Partition Table (GPT)** – Used by UEFI.
  - The GPT header is stored at the beginning and end of the disk.
  - It supports larger storage sizes (more than 2 TB) and up to 128 partitions.
  - Includes redundancy for recovery and better security.



Master Boot Record (MBR) is a special sector located at the very beginning of a storage device (usually the first 512 bytes of a hard disk or SSD) that contains information about how the disk is partitioned and how to load the operating system.

A GUID Partition Table (GPT) is a modern partitioning scheme used to manage disk partitions on storage devices. It is part of the UEFI standard and is the successor to the Master Boot Record (MBR) partitioning scheme.

Figure 4.4: Master Boot Record v/s GUID Partition Table (GPT)



INDEX

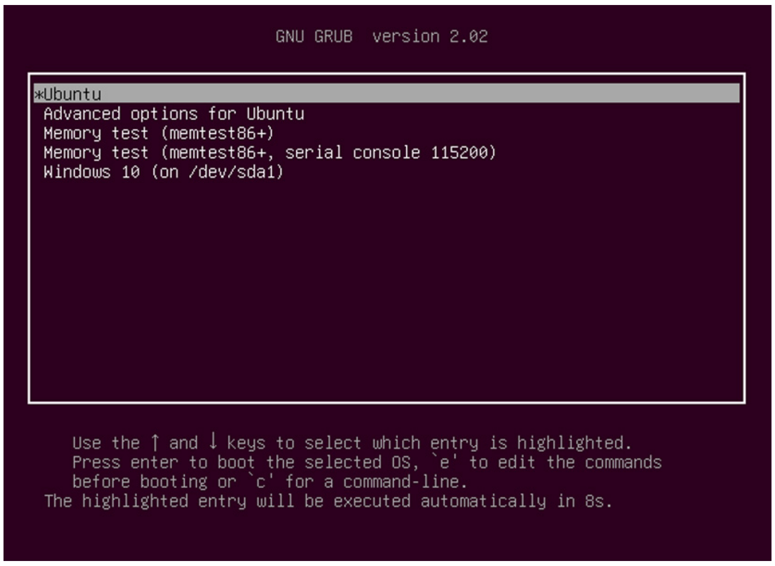
Sl.No.	NAME OF EXPERIMENT	DATE	PAGE No.	INITIAL OF TEACHER
1.	LED Characteristics	16/10/18	1-2	[illegible]
2.	LASER Diode characteristics	9/10/18	3-4	[illegible]
3.	Numerical Aperture	9/10/18	5	[illegible]
4.	Study of losses in Optical Fibre	23/10/18	6	[illegible]
	Microwave Lab			
1.	Familiarisation of Components	25/9/18	8-11	
2.	GUNN Diode Characteristics	25/9/18	12-13	
3	VSWR Measurement	16/10/18	14-15	
4	Verification of Relation b/w λ_g λ_c & λ_o	30/10/18	16	[illegible]
5	Directional Coupler	30/10/18	17	

Figure 4.5: If we assume the lab record to be a hard disk, the partition table of the hard disk is similar to the Index page of a Lab record. The partition table stores information about the structure and layout of partitions on the disk.

Loading the Bootloader

The firmware loads a bootloader into memory. A Bootloader/Bootstrap Loader is a compact software which is responsible to load the operating system into the memory of the computer The bootloader will always run whenever the computer system is started or restarted. Common bootloaders include:

- **GRUB (GRand Unified Bootloader)** – Used by Linux-based operating systems.
- **Windows Boot Manager** – Used by Windows OS.
- **Syslinux** – Lightweight bootloader mainly for Linux.
- **rEFInd** – A graphical boot manager, mainly for UEFI systems.

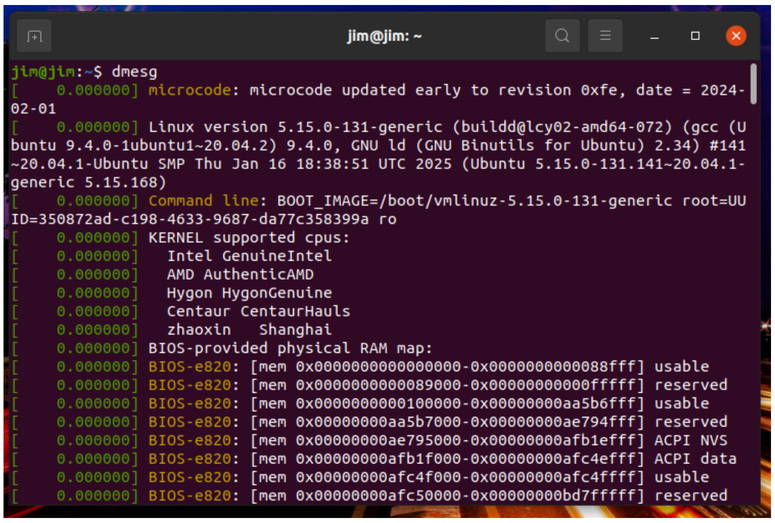


```
                    GNU GRUB  version 2.02

*Ubuntu
 Advanced options for Ubuntu
 Memory test (memtest86+)
 Memory test (memtest86+, serial console 115200)
 Windows 10 (on /dev/sda1)

     Use the ↑ and ↓ keys to select which entry is highlighted.
     Press enter to boot the selected OS, `e' to edit the commands
     before booting or `c' for a command-line.
  The highlighted entry will be executed automatically in 8s.
```

Figure 4.6: GNU GRUB (short for GNU GRand Unified Bootloader, commonly referred to as GRUB

4.4 Load Operating System Kernel

The bootloader loads the operating system kernel from the storage device into RAM. Essential system drivers and components are initialized.



```
jim@jim:~$ dmesg
[    0.000000] microcode: microcode updated early to revision 0xfe, date = 2024-
02-01
[    0.000000] Linux version 5.15.0-131-generic (buildd@lcy02-amd64-072) (gcc (U
buntu 9.4.0-1ubuntu1~20.04.2) 9.4.0, GNU ld (GNU Binutils for Ubuntu) 2.34) #141
~20.04.1-Ubuntu SMP Thu Jan 16 18:38:51 UTC 2025 (Ubuntu 5.15.0-131.141~20.04.1-
generic 5.15.168)
[    0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-5.15.0-131-generic root=UU
ID=350872ad-c198-4633-9687-da77c358399a ro
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Hygon HygonGenuine
[    0.000000]   Centaur CentaurHauls
[    0.000000]   zhaoxin   Shanghai
[    0.000000] BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x0000000000088fff] usable
[    0.000000] BIOS-e820: [mem 0x0000000000089000-0x00000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000aa5b6fff] usable
[    0.000000] BIOS-e820: [mem 0x00000000aa5b7000-0x00000000ae794fff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000ae795000-0x00000000afb1efff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x00000000afb1f000-0x00000000afc4efff] ACPI data
[    0.000000] BIOS-e820: [mem 0x00000000afc4f000-0x00000000afc4ffff] usable
[    0.000000] BIOS-e820: [mem 0x00000000afc50000-0x00000000bd7fffff] reserved
```

Figure 4.7: On a Linux PC, the `dmesg` command is used to display messages generated by the kernel during the boot process and while the system is running. The OS Kernel version 5.15.0-131-generic is displayed on the second line

4.5 Initialize System Processes

The kernel starts the init system (e.g., init, systemd in Linux, or wininit.exe in Windows).System services, background processes, and hardware drivers are loaded.

4.6 User Login & Desktop Environment

The system reaches the login screen or directly loads the desktop environment. User authentication is required (username/password).The user's session starts, and the system is fully operational.

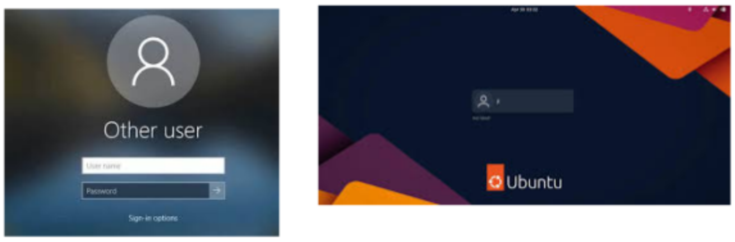


Figure 4.8: Login screens on Windows and Ubuntu OS's

Experiment 5

Installation of Linux and Windows Operating Systems

JOAQUIM IGNATIOUS MONTEIRO

In this experiment, we perform a fresh installation of Ubuntu Linux and Windows 10 in a dual-boot configuration on a PC. This setup allows the user to choose between Ubuntu Linux and Windows when starting the system.

After studying the material geiven below, perform experiment and record answers to the questions given in the lab record.

5.1 Prerequisites

- Download the Ubuntu 24.04 LTS OS image from this link [4]
- Download the Windows 10 OS image from this link [5]
- Create a Multi-OS Bootable USB (created using Ventoy [6] or separate USBs for Windows and Ubuntu).
- Backup important data before proceeding.
- Ensure BIOS is set to **UEFI mode** and Secure Boot is disabled.
- It will also be better to disable CSM support on the BIOS. (Compatibility Support Module (CSM) is a BIOS option that allows a UEFI system to boot in legacy BIOS mode)
- Ensure the system has a stable power source.

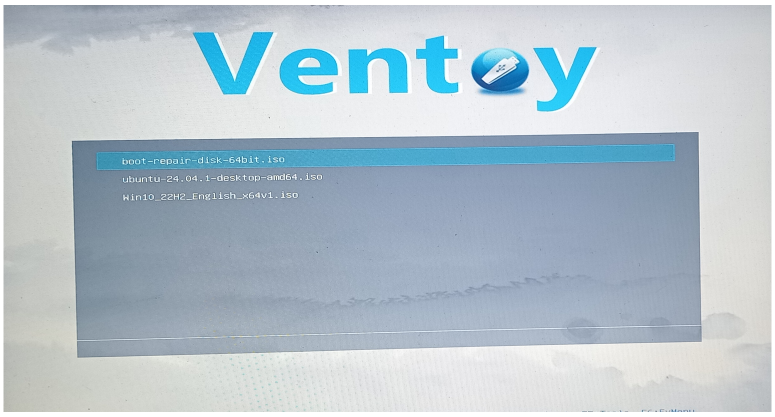

Ventoy
boot-repair-disk-64bit.iso
ubuntu-24.04.1-desktop-amd64.iso
Win10_22H2_English_x64v1.iso

Figure 5.1: Screenshot of Multi-OS Bootable USB created using Ventoy.ISO files for Boot-Repair, Ubuntu 24.04 and Windows 10

5.2 Step 1: Format the Hard Drive using GParted

1. Boot from an Ubuntu Live USB by restarting the PC and selecting the USB device from the boot menu (usually by pressing **F2, F12, Del, or Esc**).

2. Select **Try Ubuntu**.

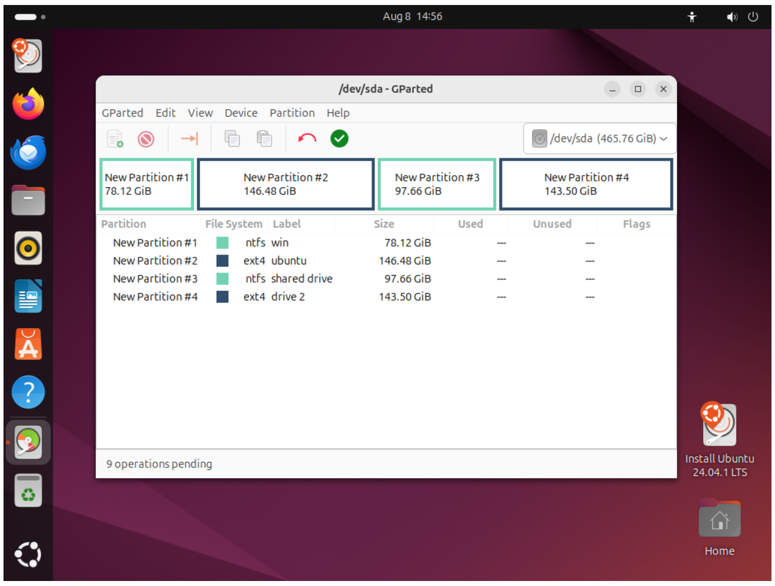
3. Open **GParted** (Press ‘Windows key‘ and type ‘gparted‘).

4. Create a **New GPT Partition Table**:
   - Select drive (e.g., ‘/dev/sda‘).
   - Go to **Device** → **Create Partition Table**.
   - Choose **GPT** and confirm.

5. Assuming a 500 GB Hard disk, create partitions as follows: (See Figure 5.2)
   - **Windows Partition:** 80GB, NTFS, Label: ‘Windows‘
   - **Ubuntu Partition:** 160GB, EXT4, Label: ‘Ubuntu‘
   - **Shared Drive:** 100GB, NTFS, Label: ‘Shared‘
   - **Extra Linux Partition:** 144GB, EXT4, Label: ‘Linux_Data‘

6. Click **Apply** and exit GParted.

7. Restart the PC and boot from the Windows 10 USB.


Aug 8 14:56
/dev/sda - GParted
GParted Edit View Device Partition Help
/dev/sda (465.76 GiB)
New Partition #1 78.12 GiB
New Partition #2 146.48 GiB
New Partition #3 97.66 GiB
New Partition #4 143.50 GiB
Install Ubuntu 24.04.1 LTS
Home

Partition		File System	Label	Size	Used	Unused	Flags
New Partition #1	🟩	ntfs	win	78.12 GiB	---	---	
New Partition #2	■	ext4	ubuntu	146.48 GiB	---	---	
New Partition #3	🟩	ntfs	shared drive	97.66 GiB	---	---	
New Partition #4	■	ext4	drive 2	143.50 GiB	---	---	

9 operations pending

Figure 5.2: Screenshot of GParted after the required partitions have been created on the hard disk

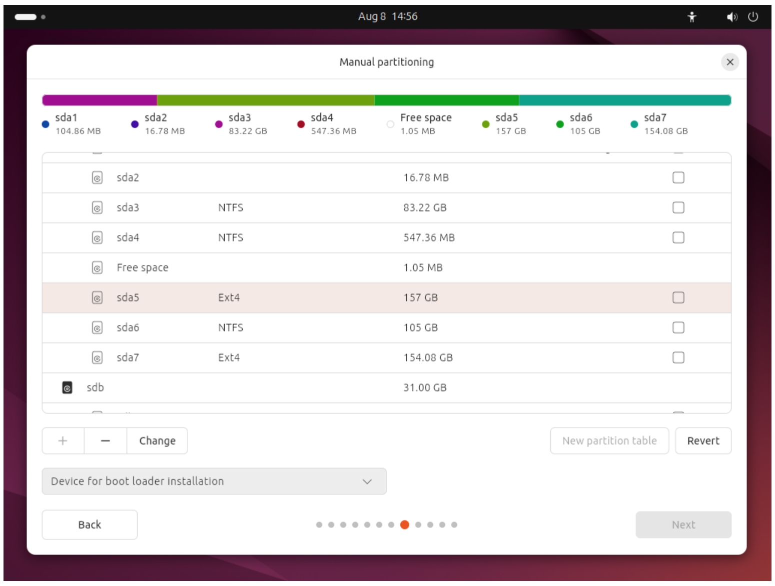
5.3 Step 2: Install Windows 10

1. Boot from the Windows 10 USB installer by selecting it in the boot menu.
2. Click **Install Now** and select **Custom Installation**.
3. Choose the `Windows` partition (100GB NTFS) created earlier.
4. Click **Format**, then click **Next** to proceed.
5. Windows will begin copying files and restart multiple times.
6. Follow the on-screen setup:
   - Choose region, keyboard layout, and language.
   - Set up a user account and password.
   - Select **I don't have a product key** if installing without activation.
   - Choose **Custom Settings** to disable unwanted tracking options.
   - Let Windows complete the installation.
7. Once inside Windows: **(Optional step in the lab experiment)**
   - Install necessary drivers and updates.
   - Open **Disk Management** (`Win + X` →Disk Management) to verify partitions.
8. Restart the PC and boot from the Ubuntu USB installer.

5.4 Step 3: Install Ubuntu 24.04

1. Boot from the Ubuntu USB installer by restarting the PC and selecting the USB device from the boot menu.
2. Once the installer initializes, choose your preferred **language**.
3. Configure any **accessibility settings** if required and select your **keyboard layout**.
4. The option to connect to a Wi-Fi network can be skipped.
5. Choose **Install Ubuntu** when prompted.
6. Select **Interactive installation** when given the choice between Interactive and Automated Installation.
7. Select **Extended selection** when prompted.
8. Skip the option to install third-party software if desired.
9. Choose **Manual Installation** to manually assign partitions.
10. **Partition Setup**

    (a) Locate the **157GB EXT4 partition (sda5)** and assign it as **/ (root)**. Click **Change** to confirm.



Aug 8 14:56

Manual partitioning

● sda1 104.86 MB · ● sda2 16.78 MB · ● sda3 83.22 GB · ● sda4 547.36 MB · ○ Free space 1.05 MB · ● sda5 157 GB · ● sda6 105 GB · ● sda7 154.08 GB

sda2		16.78 MB	☐
sda3	NTFS	83.22 GB	☐
sda4	NTFS	547.36 MB	☐
Free space		1.05 MB	
sda5	Ext4	157 GB	☐
sda6	NTFS	105 GB	☐
sda7	Ext4	154.08 GB	☐
sdb		31.00 GB	

Figure 5.3: Selecting the 157GB EXT4 partition (sda5) for Ubuntu installation.

    (b) Install the bootloader on `/dev/sda`.

(c) Review the partition setup and click **Install Now**.

11. **Final Installation Steps**

    (a) Follow the on-screen prompts:

    - Choose your **timezone**.
    - Create a **user account** with a password.
    - Wait for the installation to complete.

    (b) Restart the PC after installation completes and remove the USB when prompted.

Your dual-boot system is now ready! Enjoy using both Windows and Ubuntu.

5.5 Common Dual-Boot Issues

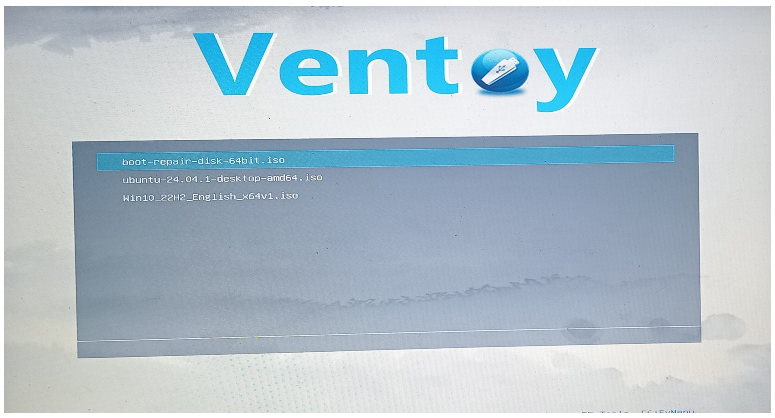
1. **GRUB Bootloader Missing or Overwritten**
   Windows updates or reinstallations can overwrite GRUB, preventing access to Ubuntu.
2. **Ubuntu Not Showing in Boot Menu**
   Windows Boot Manager might be set as the default, skipping GRUB.
3. **GRUB Error Messages**
   Errors like `grub rescue>` or `no such partition` can appear due to incorrect bootloader installation.
4. **Windows Boot Failure After Installing Ubuntu**
   Misconfigured boot order or partitioning can cause Windows to fail to boot.
5. **UEFI/Legacy Mode Mismatch**
   If Windows is installed in UEFI mode and Ubuntu in Legacy (or vice versa), boot issues arise.
6. **Secure Boot Interference**
   Some UEFI firmware settings prevent GRUB from loading.
7. **Wrong Boot Disk or Missing EFI Partition**
   The system might not find the boot partition due to misconfiguration.

5.6 Solutions Using Boot-Repair-Disk

Step 1: Boot into Boot-Repair-Disk

1. Insert the USB and reboot.

2. Access BIOS/UEFI settings (press **F2, F12, DEL, or ESC** during boot).
3. Set USB as the first boot device and disable Secure Boot if necessary.
4. Save and exit to boot into Boot-Repair-Disk ISO .



```
boot-repair-disk-64bit.iso
ubuntu-24.04.1-desktop-amd64.iso
Win10_22H2_English_x64v1.iso
```

Figure 5.4: Screenshot of Multi-OS Bootable USB created using Ventoy.ISO files for Boot-Repair, Ubuntu 24.04 and Windows 10

Step 3: Run Boot Repair

1. Choose **Recommended repair** (fixes most issues automatically).
2. If the issue persists, open **Advanced options**:
   - **Reinstall GRUB** on the correct disk (usually `/dev/sda`).
   - **Repair EFI partitions** if using UEFI.
   - **Restore MBR** if using Legacy mode.
3. Follow on-screen instructions and reboot after the repair completes.

Step 4: Set Boot Order

Ensure GRUB is set as the default bootloader in BIOS/UEFI settings.

Step 5: Verify Dual Boot

If GRUB loads but Windows is missing, run:

```
sudo update-grub
```

If Windows still doesn't appear, use:

```
sudo os-prober
sudo update-grub
```

Step 6: Fix Secure Boot Issues (If Needed)

Disable Secure Boot in BIOS or sign the GRUB bootloader with `mokutil` if Secure Boot is necessary.

5.7 Alternative Manual Fixes

Reinstall GRUB Using Ubuntu Live USB

Boot into a Ubuntu Live USB and run:

```
sudo mount /dev/sdX /mnt
sudo grub-install --root-directory=/mnt /dev/sdX
sudo update-grub
```

(Replace `/dev/sdX` with your actual boot disk.)

Repair Windows Bootloader

If Windows fails to boot:

1. Boot from a Windows installation USB.
2. Open **Command Prompt** by pressing `Shift + F10` and run:

   ```
   bootrec /fixmbr
   bootrec /fixboot
   bootrec /scanos
   bootrec /rebuildbcd
   ```

3. Restart and re-run Boot-Repair to restore GRUB.

Summary

- Always install **Windows first, then Ubuntu** for easier dual-boot setup.
- Keep backups before attempting repairs.
- If dual-boot issues persist, consider **rEFInd** as an alternative boot manager.

Experiment 6

Introducing Repositories - Git

JOAQUIM IGNATIOUS MONTEIRO

6.1 Introduction to Version Control Systems and Git

Version Control Systems (VCS) are essential tools in software development, enabling developers to track changes, collaborate efficiently, and manage code versions. A VCS allows multiple developers to work on the same project without overwriting each other's work, providing a history of modifications for easy rollback when needed.

Git is a distributed version control system that has become the industry standard for source code management. Unlike centralized version control systems, Git provides a local repository on each developer's machine, ensuring faster performance and offline capabilities. It is widely used in open-source and commercial projects for efficient code collaboration and management.

6.2 Git and Hosting Platforms: GitHub, GitBucket, and GitLab

While Git is a version control system, hosting platforms such as GitHub, GitBucket, and GitLab provide cloud-based repositories for collaboration, issue tracking, and deployment automation.

6.2.1 GitHub

GitHub is the most popular Git hosting platform, offering extensive features such as pull requests, issue tracking, and integration with CI/CD tools. It is widely used by open-source communities and enterprises.

6.2.2 GitBucket

GitBucket is an open-source Git platform similar to GitHub, designed for self-hosting Git repositories. It provides a web-based interface with repository management, issues, and pull request support.

6.2.3 GitLab

GitLab is a DevOps lifecycle tool that provides Git repository management, CI/CD automation, and security features. It offers both cloud-based and self-hosted solutions.

6.3 Creating a GitHub Account

To use GitHub, a user must first create an account.

1. Visit GitHub's website.
2. Click on **Sign up**.
3. Enter a username, email, and password.
4. Verify the account through email confirmation.
5. Complete the setup process.

Once the account is created, users can start hosting repositories and collaborating on projects.

6.4 Creating a Repository on GitHub

Repositories on GitHub store project files and track changes.

1. Log in to GitHub.
2. Click on the **+** icon in the top-right corner and select **New repository**.
3. Enter a repository name and optional description.
4. Choose the repository visibility: **Public** or **Private**.
5. Optionally, initialize with a README file.
6. Click **Create repository**.

Once created, the repository URL can be used to push local projects to GitHub.

6.5 Installing and Configuring Git

Before using Git, it needs to be installed and configured. Follow the instructions on this web page `https://github.com/git-guides/install-git` and install Git on your PC. *Git should be already installed on the lab pc*

6.5.1 Check Installed Version

```
git --version
```

6.5.2 Set User Credentials

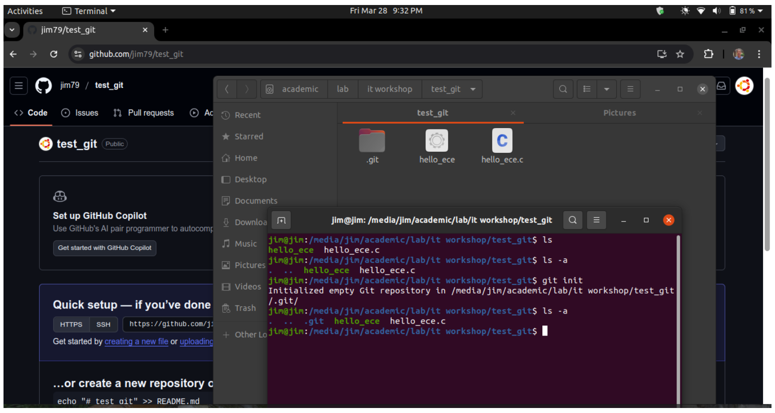
```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
```

6.6 Creating and Managing Repositories

To start using Git, a repository (repo) must be initialized.

6.6.1 Initialize a New Local Repository

```
git init
```



```
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls
hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls -a
.  ..  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git init
Initialized empty Git repository in /media/jim/academic/lab/it workshop/test_git
/.git/
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls -a
.  ..  .git  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ 
```

Figure 6.1: Initialize a New Local Repository

6.6.2 Check Repository Status

```
git status
```

6.7 Staging and Committing Changes

To track changes, files need to be added to the staging area and then committed.

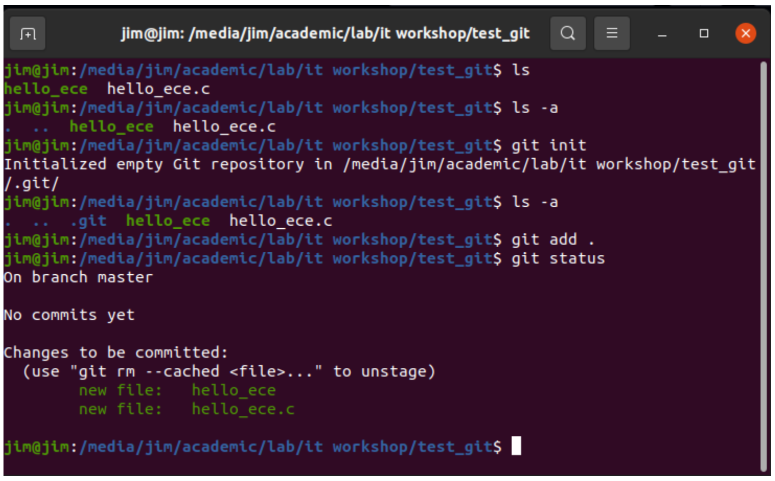
6.7.1 Add Files to Staging Area

```
git add <filename>     # command syntax
```

If all files in the local repository are to be added to the staging area use the command.

```
git add .
```

Notice the space and dot (.) after add

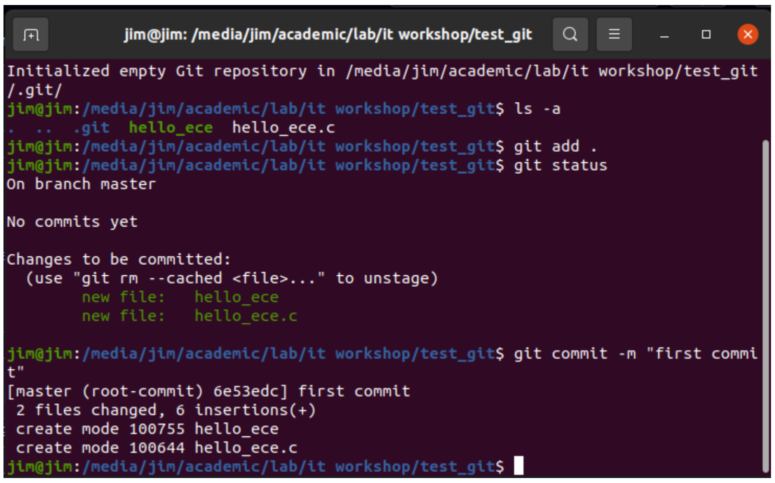


```
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls
hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls -a
.  ..  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git init
Initialized empty Git repository in /media/jim/academic/lab/it workshop/test_git
/.git/
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls -a
.  ..  .git  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git add .
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello_ece
        new file:   hello_ece.c

jim@jim:/media/jim/academic/lab/it workshop/test_git$
```

Figure 6.2: Check Repository Status, staging commits

6.7.2 Commit Changes

```
git commit -m "<commit message>"   # command syntax

git commit -m "initial commit"
```



```
Initialized empty Git repository in /media/jim/academic/lab/it workshop/test_git
/.git/
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls -a
.  ..  .git  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git add .
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello_ece
        new file:   hello_ece.c

jim@jim:/media/jim/academic/lab/it workshop/test_git$ git commit -m "first commi
t"
[master (root-commit) 6e53edc] first commit
 2 files changed, 6 insertions(+)
 create mode 100755 hello_ece
 create mode 100644 hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$
```

Figure 6.3: Commit Changes

6.7.3 View Commit History

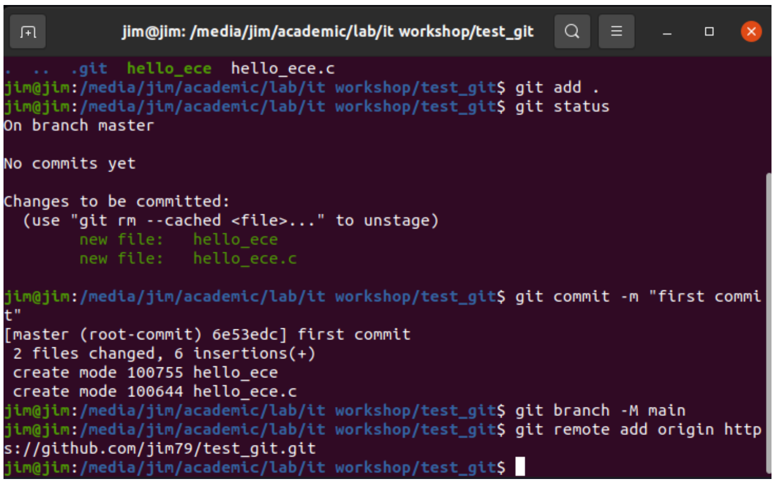
```
git log
```

6.8 Pushing and Pulling Changes

To share local changes with a remote repository, use push and pull commands.

6.8.1 Connect to a Remote Repository

```
git remote add origin https://github.com/user/repo.git
```

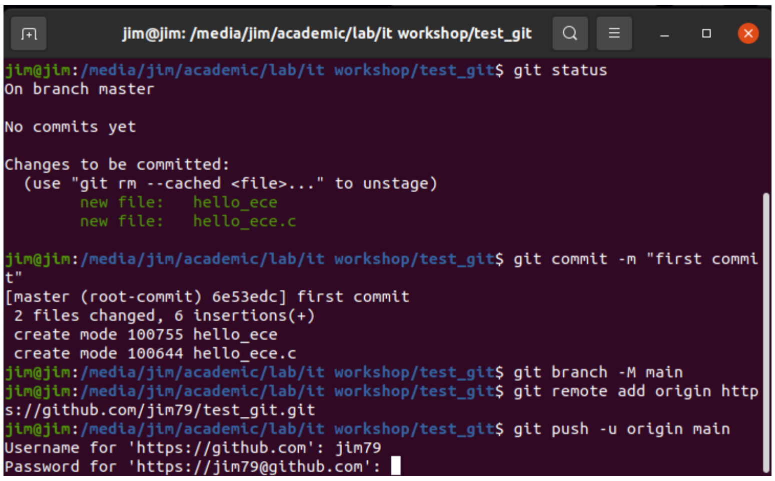


```
.  ..  .git  hello_ece  hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git add .
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello_ece
        new file:   hello_ece.c

jim@jim:/media/jim/academic/lab/it workshop/test_git$ git commit -m "first commi
t"
[master (root-commit) 6e53edc] first commit
 2 files changed, 6 insertions(+)
 create mode 100755 hello_ece
 create mode 100644 hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -M main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git remote add origin http
s://github.com/jim79/test_git.git
jim@jim:/media/jim/academic/lab/it workshop/test_git$ 
```

Figure 6.4: Commit Changes

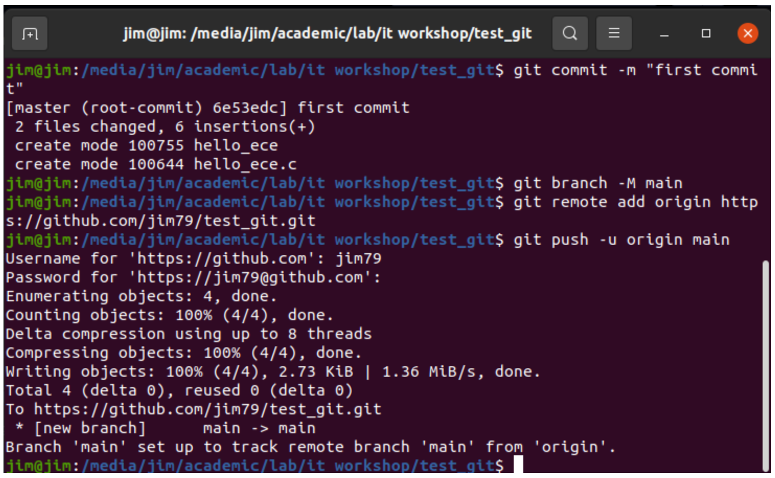
6.8.2 Push Changes to Remote Repository

```
git push -u origin main
```



```
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello_ece
        new file:   hello_ece.c

jim@jim:/media/jim/academic/lab/it workshop/test_git$ git commit -m "first commi
t"
[master (root-commit) 6e53edc] first commit
 2 files changed, 6 insertions(+)
 create mode 100755 hello_ece
 create mode 100644 hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -M main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git remote add origin http
s://github.com/jim79/test_git.git
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git push -u origin main
Username for 'https://github.com': jim79
Password for 'https://jim79@github.com': 
```

Figure 6.5: Push Changes to Remote Repository



```
jim@jim: /media/jim/academic/lab/it workshop/test_git
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git commit -m "first commi
t"
[master (root-commit) 6e53edc] first commit
 2 files changed, 6 insertions(+)
 create mode 100755 hello_ece
 create mode 100644 hello_ece.c
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -M main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git remote add origin http
s://github.com/jim79/test_git.git
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git push -u origin main
Username for 'https://github.com': jim79
Password for 'https://jim79@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 2.73 KiB | 1.36 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To https://github.com/jim79/test_git.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
jim@jim:/media/jim/academic/lab/it workshop/test_git$
```

Figure 6.6: Pushing changes to remote repository completed

6.8.3 Pull Latest Changes from Remote Repository

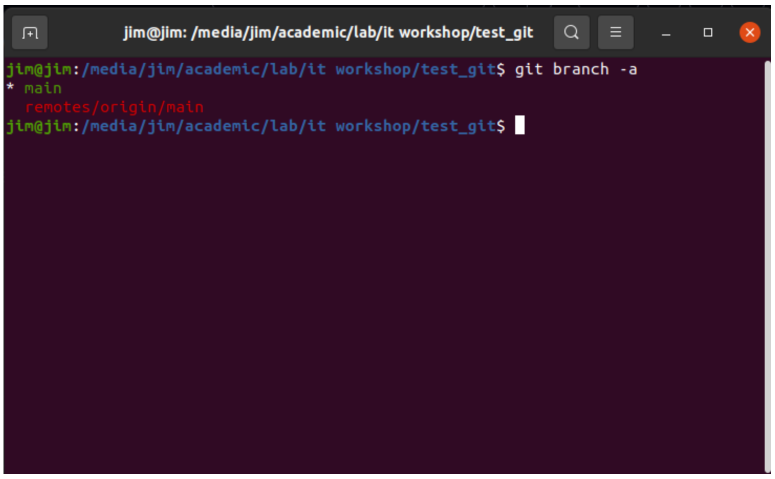
```
git pull origin main
```

6.9 Branching and Merging

Branches allow parallel development without affecting the main branch.

6.9.1 List the branches in the repository

```
git branch
```



```
jim@jim: /media/jim/academic/lab/it workshop/test_git
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -a
* main
  remotes/origin/main
jim@jim:/media/jim/academic/lab/it workshop/test_git$
```

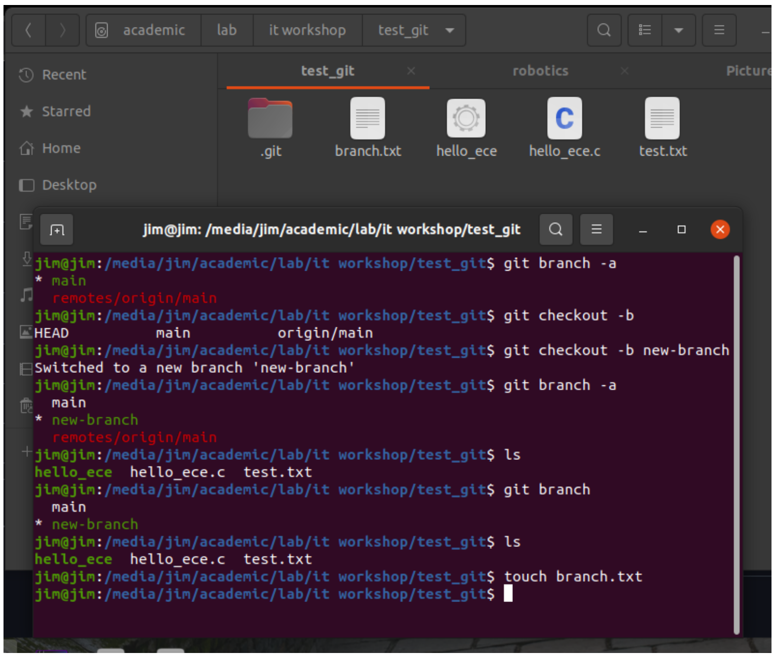
Figure 6.7: List the branches in the repository

6.9.2 Create a New Branch

```
git branch <branch-name>

git branch new-branch   # create a branch 'new-branch'
```

6.9.3 Switch to a Branch

```
git checkout <branch-name>

git checkout new-branch  # Switch to new-branch
```



```
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -a
* main
  remotes/origin/main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git checkout -b
HEAD          main          origin/main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git checkout -b new-branch
Switched to a new branch 'new-branch'
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch -a
  main
* new-branch
  remotes/origin/main
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls
hello_ece  hello_ece.c  test.txt
jim@jim:/media/jim/academic/lab/it workshop/test_git$ git branch
  main
* new-branch
jim@jim:/media/jim/academic/lab/it workshop/test_git$ ls
hello_ece  hello_ece.c  test.txt
jim@jim:/media/jim/academic/lab/it workshop/test_git$ touch branch.txt
jim@jim:/media/jim/academic/lab/it workshop/test_git$ 
```

Figure 6.8: Adding a file 'branch.txt' in the new-branch

6.9.4 Push the Branch changes to remote repository

```
git push -u origin <branch name>

git push -u origin new-branch  # set the new-branch as
   # upstream and push to remote repo
```

6.9.5 Merge a Branch

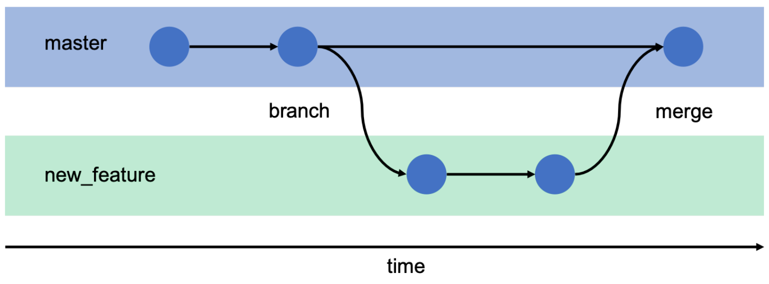


Figure 6.9: Concept of Git branches [7]

When you are ready to make the modifications made in the 'new-branch' available to everyone, you have to merge the change back to the main branch.

```
git checkout main      # switch to the main branch

git merge <branch name>

git merge new-branch     # merge changes in new-branch
# to main branch
```

6.9.6 Delete a Branch

```
git checkout main    # switch to the main branch

git branch -d new-branch  # delete new-branch
```

6.10 Cloning a repository

The concept of a "clone" just means to copy an entire repository. This clones a remote repository to the local system.

```
git clone <remote repo name>    # command syntax
git clone https://github.com/user/repo.git
```

6.11 Fork a repository

The concept of fork is to copy someone's GitHub repository and put it in your GitHub account.

```
Go to repo you wish to clone
Click the Fork icon near the top right corner
```

Experiment 7

Remaining experiments

Kindly refer the **lab record** provided on the web page https://jim79.github.io/it-workshop/ for details on the following experiments.

- Shell programming in Linux
- Familiarizing basic networking commands
- Familiarisation of Development Environments
- Familiarisation of LaTeX
- Familiarizing networking hardware

Resources & References

[1] Y. Video, “Steps in the booting process of a computer,” 2023. Accessed: 2025-02-21.

[2] A. Pronek, “Diagnosing desktop motherboard component failures,” April 2024. Accessed: Feb. 21, 2025.

[3] A. Kumar, “BIOS v/s UEFI Explained,” April 2024. Accessed: Feb. 21, 2025.

[4] Canonical Ltd., “Download ubuntu desktop,” 2025. Accessed: 2025-02-22.

[5] Microsoft Corporation, “Download windows 10 disc image (iso file),” 2025. Accessed: 2025-02-22.

[6] Ventoy Project, “Ventoy – a new bootable usb solution,” 2025. Accessed: 2025-02-22.

[7] dallasdatascience.com, “Branching (git branch),” 2025. Accessed: 2025-03-29.

SEMESTER S2

IT WORKSHOP

(Common to Group A&B)

Course Code	**GXESL208**	**CIE Marks**	50
Teaching Hours/Week (L: T:P: R)	0:0:2:0	**ESE Marks**	50
Credits	1	**Exam Hours**	2 Hrs. 30 Min.
Prerequisites (if any)	None	**Course Type**	Lab

Course Objectives:

1. To provide a basic understanding about computer hardware, software, and computer network.
2. To familiarize the learner with the web development process using HTML, CSS, and Javascript.

Details of Experiment

Expt. No	**Experiment (Minimum 10 Experiments)**
1	Practice Computer Hardware – Familiarization CPU Box, Motherboard, CPU & Chip-set, Interface cards, Card slots, Hard disk, Cables, SMPS, NIC, Various ports, etc. Computer Peripherals - I/O Devices. Storage devices, Interface cards – Buses – Firmware
2	Familiarization of Boot process
3	Familiarizing installation of Linux and Windows operating systems
4	Familiarizing basic Unix/Linux commands - ls, mkdir, cp, mv, grep, rmdir, chmod, useradd, passwd, history, dmesg, cpuinfo, uname, du, time, write, fdisk

5	Familiarizing networking hardware - RJ45, UTP, fibre, switch, NIC, router, Wireless Access Point (WAP), modem
6	Familiarizing basic networking commands - ifconfig, ping, traceroute, nslookup, ssh, scp, telnet, ftp
7	View network traffic using Wireshark/Packet tracer
8	Familiarizing the steps how to configure and establishing a network connecting
9	Shell programming in Linux(bash)
10	Create a web page and deploy on a local web server.
11	Use Javascript to validate forms.
12	Create an image slider using HTML, CSS, and JavaScript. Allow users to navigate between images using previous and next buttons.
13	Familiarisation of LaTeX - Basic only
14	Familiarisation of Development Environments - Visual studio code, Sublime Text, Atom
15	Introducing Repositories - Git / Bitbucket

Course Assessment Method

(CIE: 50 Marks, ESE 50 Marks)

Continuous Internal Evaluation Marks (CIE):

Attendance	Preparation/Pre-Lab Work, experiments, Viva and Timely completion of Lab Reports / Record. (Continuous Assessment)	Internal Exam	Total
5	25	20	50

End Semester Examination Marks (ESE):

Procedure/ Preparatory work/Design/ Algorithm	Conduct of experiment/ Execution of work/ troubleshooting/ Programming	Result with valid inference/ Quality of Output	Viva voce	Record	Total
10	15	10	10	5	50

Mandatory requirements for ESE:

- Submission of Record: Students shall be allowed for the end semester examination only upon submitting the duly certified record.

Course Outcomes (COs)

At the end of the course the student will be able to:

Course Outcome		Bloom's Knowledge Level (KL)
CO1	Experiment with the fundamental hardware components of a computer and how to interface them with software systems.	**K3**
CO2	Make use of the command line of Linux operating system and shell programming.	**K3**
CO3	Experiment with the data network communication scenarios using Wireshark.	**K3**
CO4	Develop basic websites using HTML, CSS & JavaScript and manage the versions.	**K3**

K1- Remember, K2- Understand, K3- Apply, K4- Analyse, K5- Evaluate, K6- Create

CO-PO Mapping Table

	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**
CO1	3	3	3		3							3
CO2	3	3	3	3	3							3
CO3	3	3	3	3	3							3
CO4	3	3	3	3	3							3

1: Slight (Low),2: Moderate (Medium),3: Substantial (High), : No Correlation

Text Books				
Sl. No	**Title of the Book**	**Name of the Author/s**	**Name of the Publisher**	**Edition and Year**
1	Invitation to Computer Science	G.Michael Schneider, Judith Gersting	Cengage	2/e, 2020
2	LINUX for Developers: Jumpstart Your Linux Programming Skills	William Rothwell	Person	1/e, 2018
3	HTML, CSS, and JavaScript All in One, Sams Teach Yourself	Julie C. Meloni Jennifer Kyrnin	Pearson	1/e, 2018

Reference Books				
Sl. No	**Title of the Book**	**Name of the Author/s**	**Name of the Publisher**	**Edition and Year**
1	The Architecture of Computer Hardware, Systems Software, & Networking: An Information Technology Approach	Irv Englander	Wiley	5/e, 2014
2	Mastering Git : Attain expert level proficiency with Git for enhanced productivity and efficient collaboration	Jakub Narębski	Packt	1/e, 2016
3	Web Design with HTML, CSS, JavaScript and Jquery	Jon Duckett	Wiley	1/e, 2014

Video Links (NPTEL, SWAYAM…)	
Sl. No.	**Link ID**
1	https://overthewire.org/wargames/bandit/
2	https://www.w3schools.com/

Continuous Assessment (25 Marks)

1. **Preparation and Pre-Lab Work (7 Marks)**

    - Pre-Lab Assignments: Assessment of pre-lab assignments or quizzes that test understanding of the upcoming experiment.
    - Understanding of Theory: Evaluation based on students' preparation and understanding of the theoretical background related to the experiments.

2. **Conduct of Experiments (7 Marks)**

    - Procedure and Execution: Adherence to correct procedures, accurate execution of experiments, and following safety protocols.
    - Skill Proficiency: Proficiency in handling equipment, accuracy in observations, and troubleshooting skills during the experiments.
    - Teamwork: Collaboration and participation in group experiments.

3. **Lab Reports and Record Keeping (6 Marks)**

    - Quality of Reports: Clarity, completeness and accuracy of lab reports. Proper documentation of experiments, data analysis and conclusions.
    - Timely Submission: Adhering to deadlines for submitting lab reports/rough record and maintaining a well-organized fair record.

4. **Viva Voce (5 Marks)**

    - Oral Examination: Ability to explain the experiment, results and underlying principles during a viva voce session.

Final Marks Averaging: The final marks for preparation, conduct of experiments, viva, and record are the average of all the specified experiments in the syllabus.

Evaluation Pattern for End Semester Examination (50 Marks)

1. **Procedure/Preliminary Work/Design/Algorithm (10 Marks)**

- Procedure Understanding and Description: Clarity in explaining the procedure and understanding each step involved.
- Preliminary Work and Planning: Thoroughness in planning and organizing materials/equipment.
- Algorithm Development: Correctness and efficiency of the algorithm related to the experiment.
- Creativity and logic in algorithm or experimental design.

2. **Conduct of Experiment/Execution of Work/Programming (15 Marks)**

- Setup and Execution: Proper setup and accurate execution of the experiment or programming task.

3. **Result with Valid Inference/Quality of Output (10 Marks)**

- Accuracy of Results: Precision and correctness of the obtained results.
- Analysis and Interpretation: Validity of inferences drawn from the experiment or quality of program output.

4. **Viva Voce (10 Marks)**

- Ability to explain the experiment, procedure results and answer related questions
- Proficiency in answering questions related to theoretical and practical aspects of the subject.

5. **Record (5 Marks)**

- Completeness, clarity, and accuracy of the lab record submitted