RV Educational Institutions ®
RV College of Engineering ®

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi

Approved by AICTE, New Delhi

EXPERIENTIAL LEARNING PROJECT TITLE: BUILDING OS FROM SCRATCH

Faculty Name :Jyothi shetty
R.V. College of Engineering, Bangalore-59

1)INTRODUCTION

Design and implement a minimal operating system capable of performing basic tasks such as process management, memory management, and file system operations. The operating system should be able to boot on standard PC hardware and provide a command-line interface for user interaction. Key functionalities to include bootstrapping, VGA, grub, memory management
PCI.

2) PROBLEM STATEMENT

GOALS:

- To build an basic cli interface os
- It should have basic grub control
- Is should perform basic functions such as printing,reading.
- Drivers such as keyboard.
- PCIs

3)VGA/ PCIs

VGA:
The VGA implementation in this repository comprises files for register access (reg_.c and reg_.h), VGA configuration (vga_config.c and vga_config.h), and central VGA functionality (`vga.c and `vga.h). VGA hardware, including configuring display modes, managing graphics memory, and controlling the display controller. O

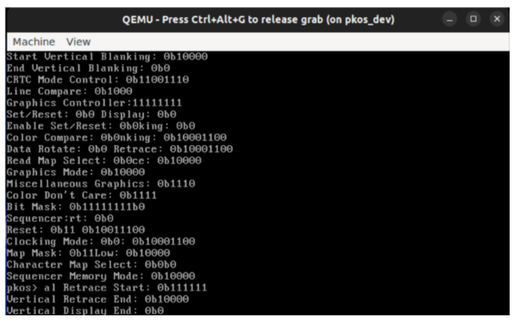
LSPCIE:
command within the Operating System (OS) kernel designed to enumerate PCI (Peripheral Component Interconnect) devices connected to the system. This functionality is crucial for hardware detection and device driver initialization, providing users with information such as vendor ID, device ID, class, and other relevant details for each PCI device

4) METHODOLOGY

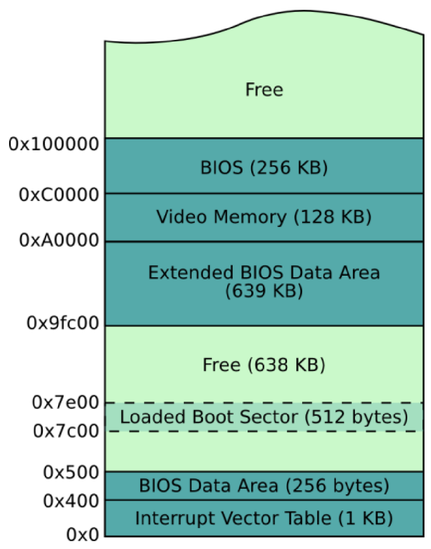
Step 1: Basic BIOS Interrupt Functionalities
- Printing on TTY: Use BIOS interrupt 0x10 for text output.
- Reading in Assembly: Use interrupt 0x16 for keyboard input.
- Waiting: Utilize interrupt 0x15, AH=0x86 for waiting.
- Changing Video Modes: Use interrupt 0x10, AH=0 for text mode and AH=0x12 for VGA.
Step 2: Creating GDT
- Define GDT segments for code, data, stack.
- Load GDT using LGDT instruction.
Step 3: Loading Kernel and Handing Over Control
- Load kernel at 0x1000.
- Jump to kernel entry point for control transfer.
Step 4: Kernel Functionalities
- Develop drivers for hardware.
- Implement standard library functions.
- Create memory allocation routines.
- Configure PCI devices.
- Handle interrupts.
- Manage processes.
- Develop networking.
- Implement kernel debugging.
- Enforce security measures.

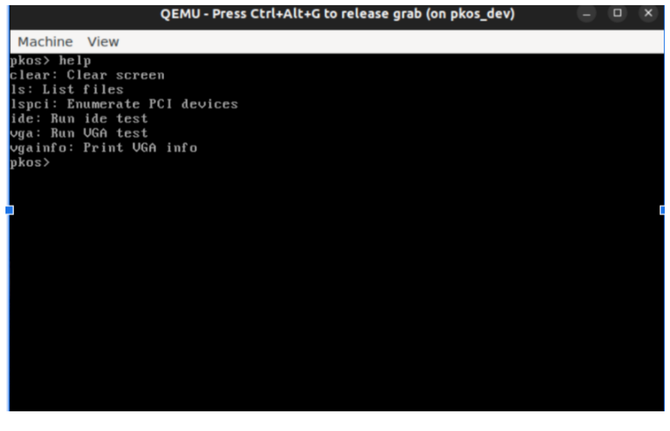
I.

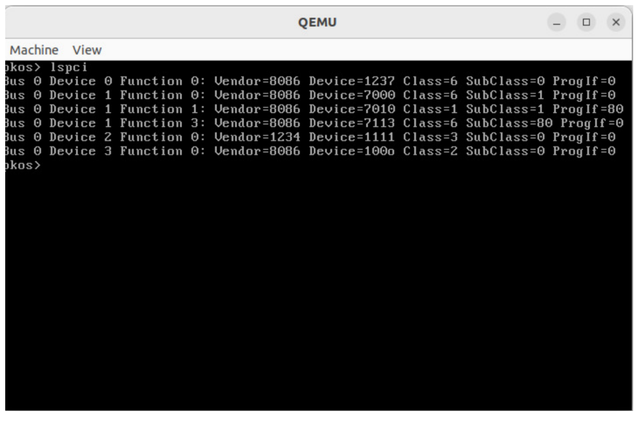
5)IMAGES



9)







6)Conclusion

This project involves building a basic operating system from scratch using a simple language assembly, Using computers interact with hardware. setting up basic tasks like showing text on the screen, handling keyboard inputs, and waiting for certain actions. create a map for the computer's memory called the Global Descriptor Table (GDT), which helps organize how the computer stores and retrieves information.

After laying this groundwork, load our operating system into the computer's memory and start running there. From this point, we gradually add more features to our operating system. We create drivers to make sure hardware like keyboards and screens work correctly, add functions for common tasks like copying data, and set up ways to manage memory efficiently. We also configure devices plugged into the computer, handle interruptions, and manage processes like running different programs.

Ultimately, this project guides us through building a fully functional operating system step by step, teaching us how computers handle basic and advanced tasks along the way.

TEAM MEMBERS

1.Sai varun konda (1RV22CS171
2. Sahil S Naik [1RVCS169