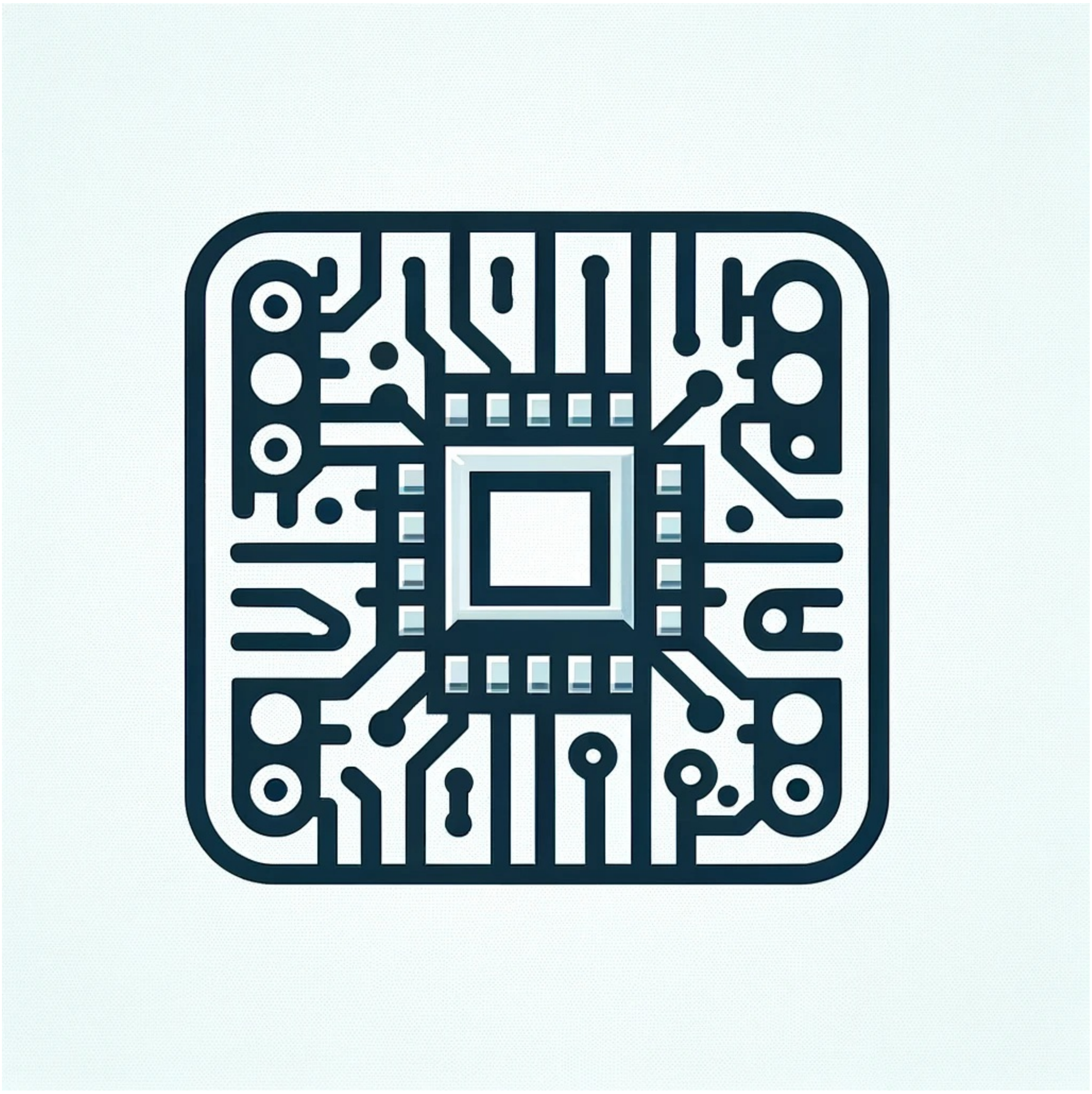
Project Documentation

Please upload your final report in **PDF** format to this folder.

The report can be in Persian or English and should include:

- Challeneges you faced during project
- Explanations about the code
- Interpretation of results



UEFI System Information Viewer

In this project, we developed a UEFI application designed to present an interactive display of system information. Leveraging UEFI services and protocols, our program retrieves and showcases comprehensive hardware and firmware details, empowering users to seamlessly inspect and interact with key system information.

UEFI stands for "Unified Extensible Firmware Interface." It is a standard interface specification that defines a software interface between the operating system and the system firmware (BIOS or UEFI firmware) during the boot process and the runtime of an operating system. UEFI is designed to replace the traditional BIOS (Basic Input/Output System) and offers several advantages, including support for larger hard drives, faster boot times, and improved security features.

UEFI programming refers to the development of software applications, known as UEFI applications, that run on systems with UEFI firmware. UEFI applications can perform various tasks, such as configuring hardware settings, interacting with the system firmware, and providing pre-boot utilities. UEFI programming often involves using languages such as C or C++ and making use of UEFI services and protocols to interact with the underlying firmware. UEFI programming is essential for creating applications that leverage the capabilities of modern UEFI firmware, providing a more flexible and extensible environment for system initialization and management compared to the traditional BIOS. This is particularly important in the context of modern computing systems, including PCs and servers.

Some of the Important components of system information are:

- System Table: The system table, often referred to as the EFI System Table, is a data structure provided by the UEFI firmware. It contains information about various services and protocols available in the UEFI environment. UEFI applications can access this table to interact with firmware services and obtain critical information about the system.
- SMBIOS (System Management BIOS) Table: SMBIOS is a standard developed by the Distributed Management Task Force (DMTF) that defines a structure (table) for conveying system information to the operating system. This information includes details about the system's hardware components, such as the processor, memory, BIOS version, and more. UEFI applications can access the SMBIOS table to retrieve and use this hardware information.
- ACPI (Advanced Configuration and Power Interface) Tables: ACPI is an industry-standard that defines power management and configuration interfaces between the operating system and the hardware. ACPI tables contain information related to system configuration, power management, and other platform-specific details. UEFI applications can use ACPI tables to obtain information about the system's power management capabilities and configurations.
- UEFI Variables: UEFI variables are a mechanism provided by the UEFI firmware to store and retrieve configuration data. These variables are used by both the firmware and UEFI applications to store persistent information, such as boot configuration settings, system language preferences, and other runtime data. UEFI applications can access these variables to retrieve and modify system-specific settings.

The implementation of this project consists of two main components:

Display and Navigation Features:

- Implementing methods to display information.
- Implementing navigation, search, and filtering functionalities.

Getting the System Information:

- Utilizing UEFI functions and protocols to acquire essential system data.
- Retrieving key system information, including the system table, SMBIOS table, ACPI tables, and UEFI variables.

Week-by-Week Milestones

Week 1:

- Install and run Edk2
  - Install the required packages.
  - Build the BaseTools.
  - Build the Edk2.
  - It was specifically challenging because of the lack of documentation and the need to search for the required information.
- Build a "Hello World" package.
  - We used UEFI-Lessons repository (link at the end) to guide us through the process.
  - We had to learn how to use the UEFI functions and protocols to display information.

Week 2:

- Familiarization with available hardware and system firmware information
  - Understanding the functions of receiving information and proficiently working with them
  - We got familiar with the protocols (SMBIOS, ACPI, UEFI) and the information they provide.
  - It also consisted of learning about the interface they provide to access the information.
- Understanding information display protocols in UEFI.
  - We learned how to display information in UEFI.

Week 3:

- Understanding the functions of receiving information and proficiently working with them
  - Learning about functions such as "GetNumberOfProcessors," "GetProcessorInfo," "GetNumberOfEnabledProcessors," "GetSmbiosTable," "GetAcpiTable," "GetVariable," and "GetVariableName."

Week 4:

- Presenting specific information
  - We created a page for "Processor Count"
- Transitioning the base between different pages.
  - The user can navigate between pages using the arrow keys.

Week 5 and 6:

- Verifying the remaining steps for project completion.
  - We created pages for "SMBIOS Table Data," "ACPI Table," and "Network Table."
  - We implemented search and filter functionalities.
  - We incorporated word highlighting when searching.
  - We added a "Main" page to serve as the primary and initial page that users encounter.
  - We added a "Subtle Items" page to display the ACPI tables.
  - We added a "Network Interface" page to display the network interface details.
  - We added a "Network" page to display the network interfaces.
- Conducting a more detailed planning and time allocation for each task.
- Understanding functions related to text and display to present information in an organized and aesthetically pleasing manner.
  - We used display instead of terminal output which was more aesthetically pleasing.
  - This called for changes such as using highlighting from the UEFI library.

Challenges

One of the main challenges we faced was the lack of documentation. We had to search for the required information and learn how to use it. Another challenge was the lack of familiarity with the UEFI functions and protocols. We had to learn how to use them and how to display information in UEFI. We also had to learn how to use the UEFI library to display information in an organized and aesthetically pleasing manner. Another challenge was the lack of familiarity with the hardware and system firmware information. We had to learn about the protocols (SMBIOS, ACPI, UEFI) and the information they provide. It also consisted of learning about the interface they provide to access the information. Finally, we had to learn about functions such as "GetNumberOfProcessors," "GetProcessorInfo," "GetNumberOfEnabledProcessors," "GetSmbiosTable," "GetAcpiTable," "GetVariable," and "GetVariableName."

One other challenge was the lack of familiarity with the UEFI library. We had to learn how to use it and how to display information in an organized and aesthetically pleasing manner. Another challenge was the lack of familiarity with the hardware and system firmware information. We had to learn about the protocols (SMBIOS, ACPI, UEFI) and the information they provide.

We also had to go through several iterations of the code design to make it more efficient and aesthetically pleasing. We had to adapt to the UEFI library and its functions to display information in an organized and aesthetically pleasing manner. This called for changes such as using highlighting from the UEFI library. We also had to learn how to use the UEFI library to display information in an organized and aesthetically pleasing manner.

Code Structure

Consult the main repo in case this one didn't work.

AzSakhtPkg

`AzSakhtPkg.dsc` is the main file that contains the list of all the modules that are part of the package. It also contains the build options for the package.

MainTable

`MainTable.c` is the main file that contains the main function of the program. It is responsible for handling and forwarding the user's input.

`Pages.h` contains the definition of the static pages.

`PageState.c` contains the main code responsible for handling the user's input and displaying the pages. It formats the current page and displays it on the screen.

Pages

Each of the files in this folder represent a page. They may create dynamic pages if they see fit (e.g. `ACPIPage.c` creates a dynamic page for each ACPI table).

Main Page

This serves as the primary and initial page that users encounter. They can select from options including "Processor Count," "SMBIOS Table Data," "ACPI Table," and "Network Table." The user's choice determines the subsequent content displayed on the page.

Processor Count Page

On this page, users can observe both the total number of processors and the count of enabled processors.

SMBIOS Page

On this page, you will discover comprehensive details about the system's BIOS, including information such as "BIOS Version," "BIOS Release Date," "Manufacturer," "Processor Version," "Processor Manufacturer," "Processor Max Speed," "Number of Cores," "Memory Size in MB," and "Cache Size."

ACPI Page

The ACPI (Advanced Configuration and Power Interface) tables embedded in the UEFI (Unified Extensible Firmware Interface) firmware furnish details regarding the system's configuration and power management. On this page, you can view the signature, OEM ID, and details of the ACPI table. Clicking on "items" will navigate you to another page titled "Subtle Items."

Conf

`target.txt` contains the target IP address.

Tools

We used the following tools to implement this project:

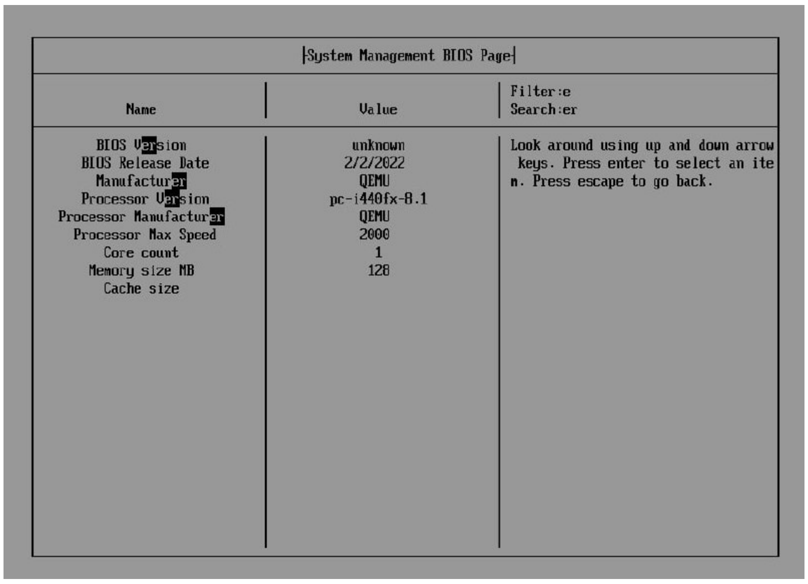
- Qemu
- EDK II

Implementation Details

Features

- Navigating between page items vertically and horizontally.
- Implementing search and filter functionality.
- Incorporating word highlighting when searching.

An example demonstrating the utilization of these features is presented below:



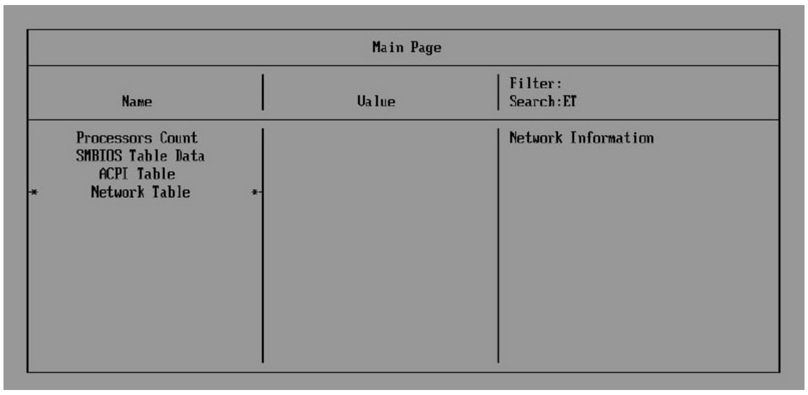
```
                          |System Management BIOS Page|

         Name            |          Value          | Filter:e
                         |                         | Search:er

      BIOS Version       |         unknown         | Look around using up and down arrow
    BIOS Release Date    |        2/2/2022         |  keys. Press enter to select an ite
      Manufacturer       |          QEMU           | m. Press escape to go back.
    Processor Version    |       pc-i440fx-8.1     |
 Processor Manufacturer  |          QEMU           |
   Processor Max Speed   |          2000           |
       Core count        |           1             |
     Memory size MB      |          128            |
       Cache size        |                         |
```

Pages

In this section, a concise overview of each page is provided. Every page is equipped with both search and filter functionalities.

"Main" Page:

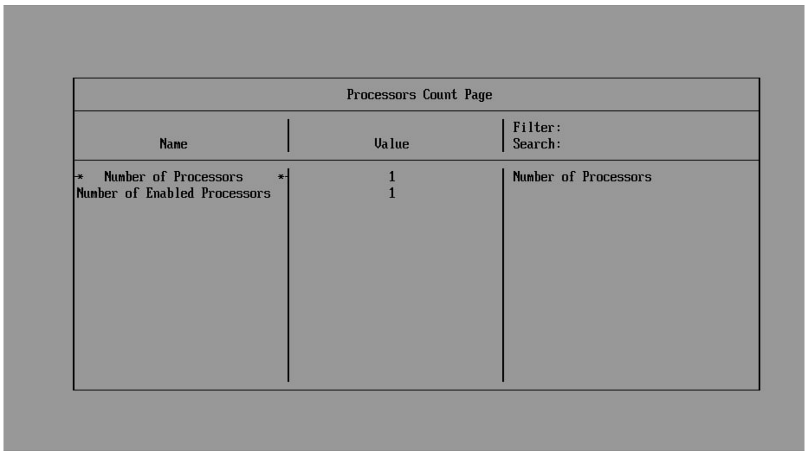
This serves as the primary and initial page that users encounter. They can select from options including "Processor Count," "SMBIOS Table Data," "ACPI Table," and "Network Table." The user's choice determines the subsequent content displayed on the page.



```
                               Main Page

         Name           |          Value          | Filter:
                        |                         | Search:ET

     Processors Count   |                         | Network Information
    SMBIOS Table Data   |                         |
       ACPI Table       |                         |
*     Network Table    *|                         |
```

"Processor count" page:

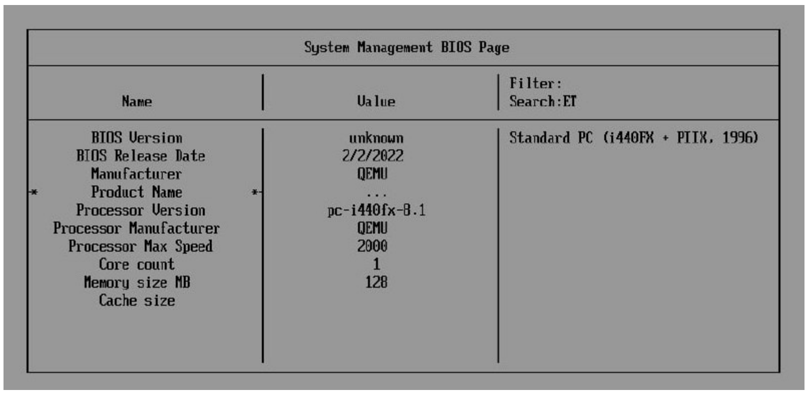
On this page, users can observe both the total number of processors and the count of enabled processors.



```
                       Processors Count Page

          Name            |      Value      | Filter:
                          |                 | Search:

*  Number of Processors  *|        1        | Number of Processors
Number of Enabled Processors       1        |
```

"System Management Bios" page:

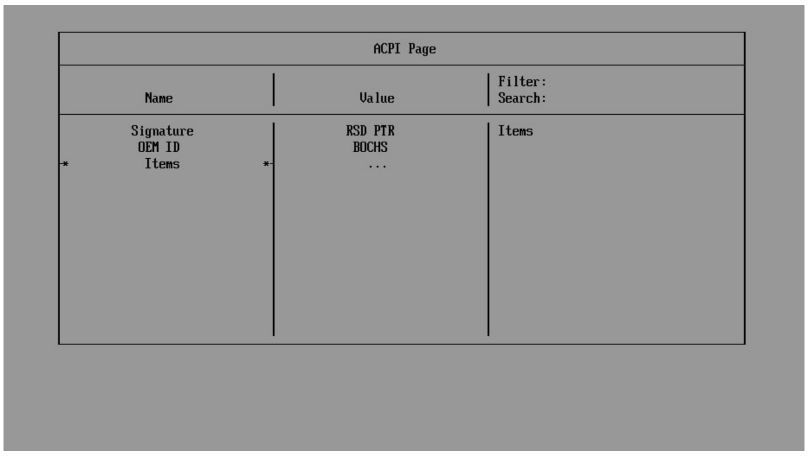
On this page, you will discover comprehensive details about the system's BIOS, including information such as "BIOS Version," "BIOS Release Date," "Manufacturer," "Processor Version," "Processor Manufacturer," "Processor Max Speed," "Number of Cores," "Memory Size in MB," and "Cache Size."



```
                     System Management BIOS Page
        Name          |          Value          | Filter:
                      |                         | Search:ET
      BIOS Version    |         unknown         | Standard PC (i440FX + PIIX, 1996)
   BIOS Release Date  |        2/2/2022         |
      Manufacturer    |          QEMU           |
*     Product Name   *|           ...           |
   Processor Version  |      pc-i440fx-8.1      |
 Processor Manufacturer|         QEMU            |
  Processor Max Speed |          2000           |
       Core count     |            1            |
     Memory size MB   |           128           |
       Cache size     |                         |
```

"ACPI" page:

The ACPI (Advanced Configuration and Power Interface) tables embedded in the UEFI (Unified Extensible Firmware Interface) firmware furnish details regarding the system's configuration and power management. On this page, you can view the signature, OEM ID, and details of the ACPI table. Clicking on "items" will navigate you to another page titled "Subtle Items."

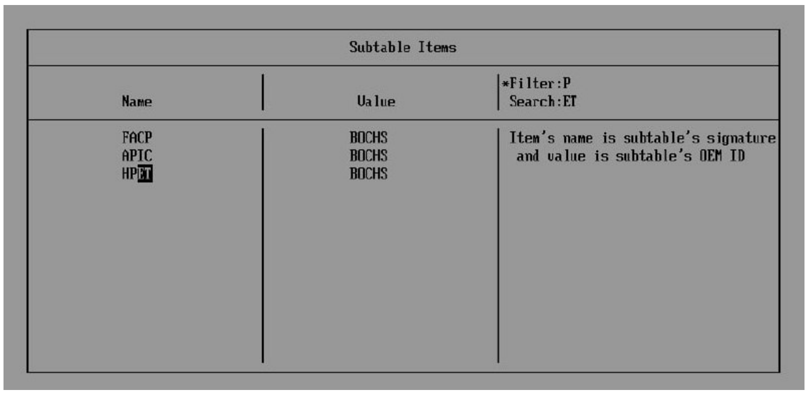


```
                          ACPI Page
        Name          |          Value          | Filter:
                      |                         | Search:
       Signature      |         RSD PTR         | Items
        OEM ID        |          BOCHS          |
*       Items        *|           ...           |
```

"Subtle Items" page:

The exact ACPI tables present can vary depending on the system and firmware implementation, but commonly encountered tables include:

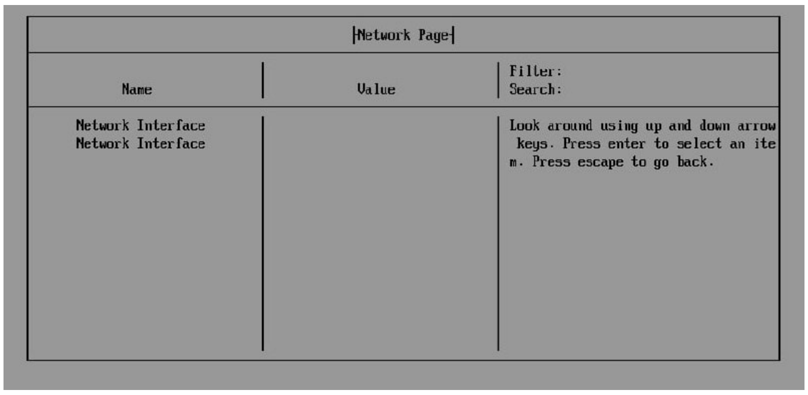
- RSDP (Root System Description Pointer): Contains information about the location and structure of other ACPI tables.
- RSDT (Root System Description Table): Lists the 32-bit physical addresses of other ACPI tables.
- XSDT (Extended System Description Table): Similar to RSDT but supports 64-bit physical addresses.
- FACP (Fixed ACPI Description Table): Provides information about system power management and configuration.
- DSDT (Differentiated System Description Table): Contains the main ACPI description for the system, providing details about devices, power management, and more.
- SSDT (Secondary System Description Table): Supplemental tables that provide additional information and can be loaded dynamically.
- APIC (Multiple APIC Description Table): Describes the system's interrupt controllers.
- HPET (High Precision Event Timer): Provides information about the system's timer capabilities.
- SBST (Smart Battery Specification Table): Presents information about the system's smart battery if applicable.
- BGRT (Boot Graphics Resource Table): Contains information about the system's logo or boot graphics.
- SLIT (System Locality Information Table): Describes the relative distances between processors in a multi-processor system.



```
                        Subtable Items
        Name          |          Value          |*Filter:P
                      |                         | Search:ET
         FACP         |          BOCHS          | Item's name is subtable's signature
         APIC         |          BOCHS          |  and value is subtable's OEM ID
         HPET         |          BOCHS          |
```

"Network" page:

On this page, you can select any desired network interface, and upon selection, you will be redirected to another page displaying specific details about the chosen interface.

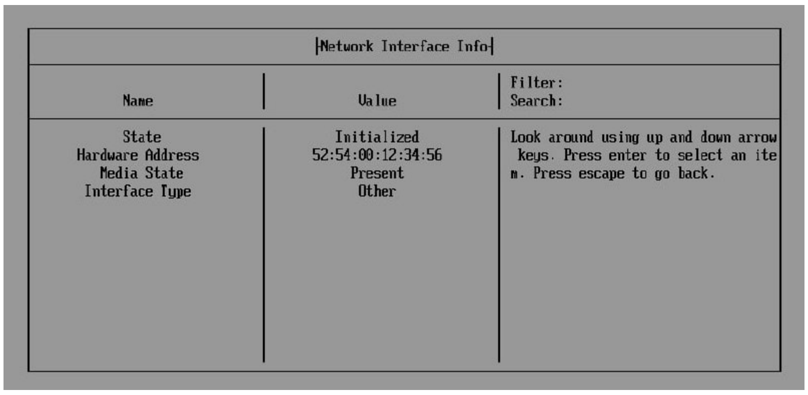


```
                              |Network Page|
        Name          |          Value          | Filter:
                      |                         | Search:
  Network Interface   |                         | Look around using up and down arrow
  Network Interface   |                         |  keys. Press enter to select an ite
                      |                         | m. Press escape to go back.
```

"Network Interface" page:

On this page, you will find in-depth details regarding your chosen network interface. The information presented includes:

- "State": current operational status of the network interface.
- "Hardware Address": a unique identifier assigned to the network interface. It is a combination of letters and numbers assigned to network interfaces for communication on the physical network.
- "Media state": link status or connection status of the network interface.
- "Interface type": the technology or protocol used by the network interface for communication.



```
                          |Network Interface Info|
        Name          |          Value          | Filter:
                      |                         | Search:
        State         |       Initialized       | Look around using up and down arrow
   Hardware Address   |    52:54:00:12:34:56    |  keys. Press enter to select an ite
     Media State      |         Present         | m. Press escape to go back.
    Interface Type    |          Other          |
```

How to Run

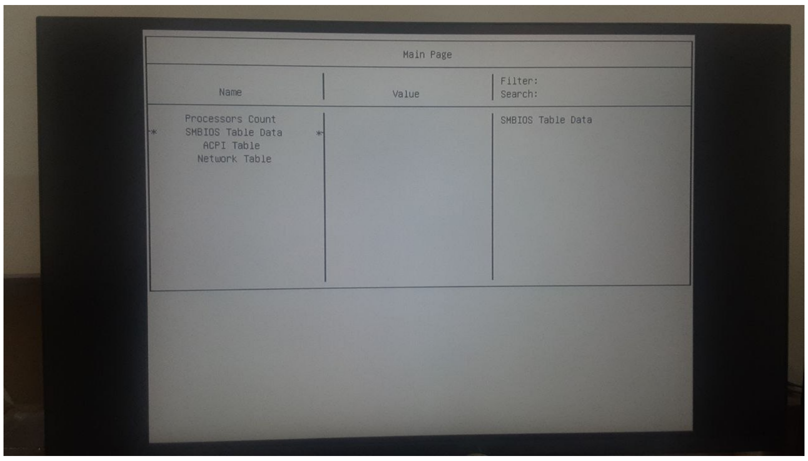
- Edit `setup_env.sh` to match your system and run `. ./setup_env.sh`.
- Use `make run` to build the package and start qemu. Then enter the following commands to run the program:

```
Shell> FS0:MainTable.efi
```

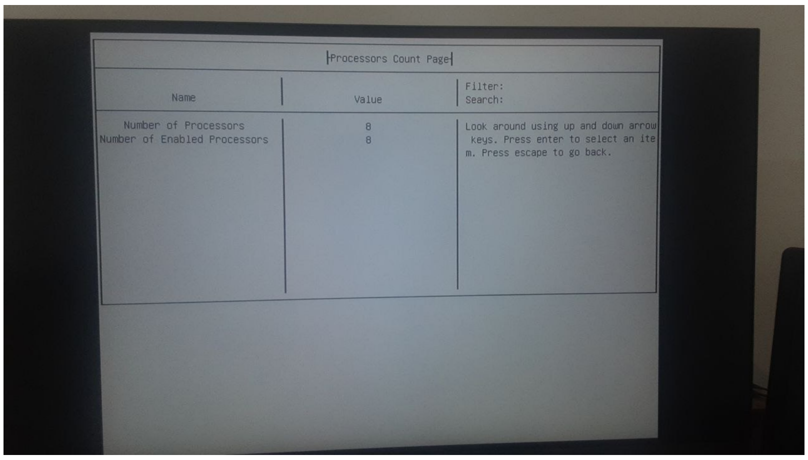
- Press `Ctrl-A X` to terminate.

Results

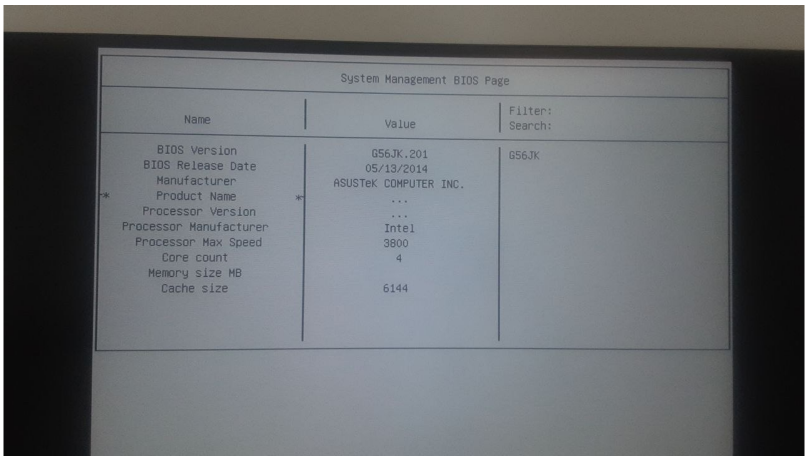
You can observe our program performance on an actual system below:

1- 

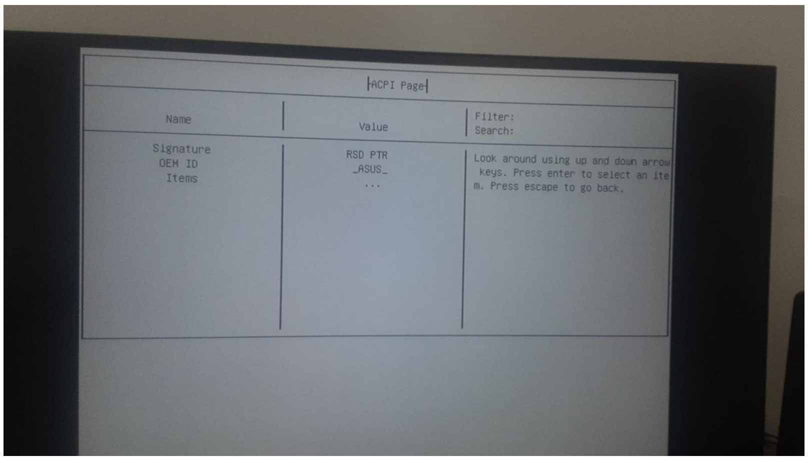
```
                              Main Page
        Name          |          Value          | Filter:
                      |                         | Search:
   Processors Count   |                         | SMBIOS Table Data
*  SMBIOS Table Data  *                         |
      ACPI Table      |                         |
    Network Table     |                         |
```

2- 

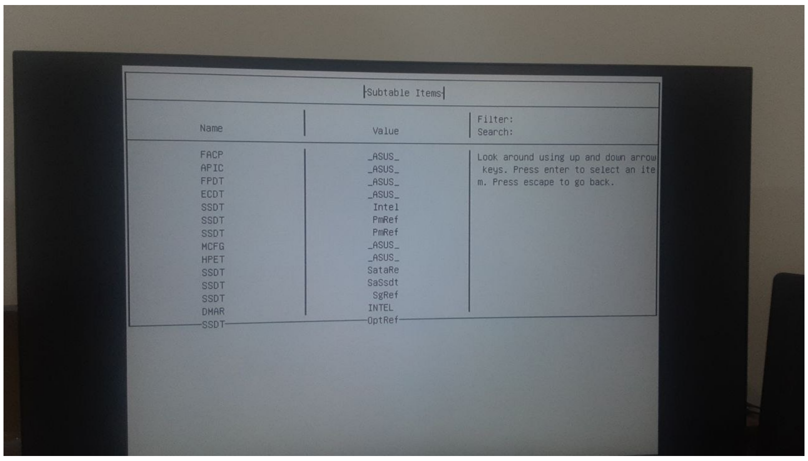
```
                         Processors Count Page

        Name                   |         Value         | Filter:
                               |                       | Search:
  Number of Processors         |           8           | Look around using up and down arrow
Number of Enabled Processors   |           8           |  keys. Press enter to select an ite
                               |                       | m. Press escape to go back.
```

3- 

```
                         System Management BIOS Page

        Name                   |         Value         | Filter:
                               |                       | Search:
      BIOS Version             |       G56JK.201       | G56JK
    BIOS Release Date          |      05/13/2014       |
      Manufacturer             |  ASUSTeK COMPUTER INC. |
*     Product Name            *|          ...          |
    Processor Version          |          ...          |
  Processor Manufacturer       |         Intel         |
   Processor Max Speed         |         3800          |
       Core count              |           4           |
     Memory size MB            |                       |
       Cache size              |         6144          |
```

4- 

```
                               ACPI Page

        Name                   |         Value         | Filter:
                               |                       | Search:
      Signature                |        RSD PTR        | Look around using up and down arrow
       OEM ID                  |        _ASUS_         |  keys. Press enter to select an ite
        Items                  |          ...          | m. Press escape to go back.
```

5- 

```
                             Subtable Items

        Name                   |         Value         | Filter:
                               |                       | Search:
        FACP                   |        _ASUS_         | Look around using up and down arrow
        APIC                   |        _ASUS_         |  keys. Press enter to select an ite
        FPDT                   |        _ASUS_         | m. Press escape to go back.
        ECDT                   |        _ASUS_         |
        SSDT                   |         Intel         |
        SSDT                   |         PmRef         |
        SSDT                   |         PmRef         |
        MCFG                   |        _ASUS_         |
        HPET                   |        _ASUS_         |
        SSDT                   |        SataRe         |
        SSDT                   |        SaSsdt         |
        SSDT                   |         SgRef         |
        DMAR                   |         INTEL         |
        SSDT                   |        OptRef         |
```

Discussion

In this project, we developed a UEFI application designed to present an interactive display of system information. Leveraging UEFI services and protocols, our program retrieves and showcases comprehensive hardware and firmware details, empowering users to seamlessly inspect and interact with key system information. We implemented methods to display information and to navigate, search, and filter functionalities. We also utilized UEFI functions and protocols to acquire essential system data. We retrieved key system information, including the system table, SMBIOS table, ACPI tables, and UEFI variables.

We presented a concise overview of each page. Every page is equipped with both search and filter functionalities. We also discussed the challenges we faced during the project. One of the main challenges we faced was the lack of documentation. We had to search for the required information and learn how to use it. Another challenge was the lack of familiarity with the UEFI functions and protocols. We had to learn how to use them and how to display information in UEFI. We also had to learn how to use the UEFI library to display information in an organized and aesthetically pleasing manner. Another challenge was the lack of familiarity with the hardware and system firmware information. We had to learn about the protocols (SMBIOS, ACPI, UEFI) and the information they provide. It also consisted of learning about the interface they provide to access the information. Finally, we had to learn about functions such as "GetNumberOfProcessors," "GetProcessorInfo," "GetNumberOfEnabledProcessors," "GetSmbiosTable," "GetAcpiTable," "GetVariable," and "GetVariableName."

Related Links

Some links related to your project come here.

- EDK II
- UEFI Lessons

Authors

Authors and their github link come here.

- @panizhalvachi
- @Arman17Babaei
- @yasamanzolfi