Fundamentals of Embedded Systems

Final Project:

Multimedia Center

Cecilia Fernanda San Miguel Iturria LN: 22

May 21, 2024

An embedded system or integrated system is a controller that is part of a larger system designed to perform a dedicated function. They are used in a large number of modern devices, including household appliances such as microwaves, toasters, and washing machines.

For example, the embedded systems in a washing machine handle the opening and closing of valves to allow water to enter the system at set intervals (pre-wash, wash) and then drain when necessary. These processes are controlled by microcontrollers.

The complexity of an embedded system can vary significantly depending on the task it is designed for, ranging from a single microcontroller to a set of chips with connected peripherals and networks. [1]

A Raspberry Pi is a small computer that's about the size of a computer mouse (85 mm x 56 mm is the standard size). It looks like a PC motherboard, but all the components are already on it (CPU, memory, wireless module, USB, network, etc.). [2]

A media center, or multimedia center, is software that allows you to have and enjoy all multimedia content. It can play music, movies, display images in galleries, etc., all from multimedia content stored on a local hard drive or accessed through the network. (...) These multimedia centers can be running on a mobile device, on a PC, on an SBC as in the case of OSMC on Raspberry Pi, on a smart TV, etc. [3]

The objective of this project was to implement an embedded entertainment center that allows users to play movies, videos, music, and photos, both from removable media and online services.

Materials

- Raspberry Pi 3 or higher
- Power supply for Raspberry Pi
- HDMI cable
- MicroSD card
- USB drive with multimedia content
- Monitor
- Keyboard
- Mouse
- Speakers or headphones for the monitor

Description of the Functionality of Electronic Components

USB Drive

Data Storage: The USB drive uses non-volatile memory technology, meaning it does not require a constant power source to maintain the stored information. *USB Interface*: It connects to a USB port on a computer or other compatible device. The USB interface provides a fast connection for transferring

data between the drive and the device.

Monitor
Visual Output Device: A monitor is an output device that displays images and videos generated by the graphics card of the computer or device to which it is connected. *Connectivity*: Monitors can connect to computers and other devices through various types of cables, such as HDMI, DisplayPort, DVI, and VGA. Some modern monitors also offer USB-C connectivity.

Speakers or Headphones
Types of Connection: They can connect to devices via cables such as 3.5mm audio jack, USB, or wirelessly through Bluetooth.

Description of the Controller Card Operation
The Raspberry Pi operates by connecting to a monitor or television and using a standard keyboard and mouse. When powered on, the Raspberry Pi boots its operating system from an SD card. The operating system can be any compatible Linux-based operating system, such as Raspbian, Ubuntu, or Debian.

Once the operating system has been loaded, the Raspberry Pi is ready to be used. It can run basic and complex programs, connect to the Internet via Ethernet or Wi-Fi, and control external devices through the GPIO pins.[4]

Health Care Information and Risk Warnings
When working with the Raspberry Pi 3 board, it's important to take certain precautions, although it doesn't pose a major health risk to individuals.
At times, the board may become hot during use, so it's recommended not to touch it while in operation as it could cause a minor burn. Whenever the Raspberry Pi is connected and in use, it's advisable not to touch it, as it's connected to the electrical power and could cause minor electric shocks.

Care of Electronic Components
In general, it's recommended to handle the devices with care and avoid bumping or hitting them, as both the USB, monitor, and Raspberry Pi could cease to function properly due to such minor damages.

Controller Card Configuration
For the proper functioning of the Raspberry Pi, it will be necessary to connect all peripherals to be used, such as the keyboard, mouse, and monitor, and then power the Raspberry Pi with a micro B USB cable.

To ensure the proper functioning of the Raspberry Pi, download the Raspberry Pi OS (Legacy) Lite operating system for 64-bit with kernel version 6.1 and Debian version 11 (bullseye).

In order to use the Raspberry Pi as a multimedia center, it's necessary to install some libraries: - VLC media player: It's a multimedia library used to play audio and video files. - tkinter (tk): Provides tools for creating graphical user interfaces (GUIs) in Python applications. - os: Provides functions to interact with the operating system, such as file and directory manipulation.

Development of software components (modules) for each of the functions that the card will perform.
The operation of the multimedia center is divided into several modules:

GUI:

This directory stores the classes that display the project's customized graphical interface across all its views, generating panels and divisions that allow us to organize buttons related to images to make it more visually appealing. Likewise, this path also contains methods to clear the panels and correctly display the one in which the user is located. Additionally, methods related to the four different types of files on the USB that our Multimedia Center should be able to read can be found in this directory.

Video:
Handling video playback with the VLC library, an array is initialized, which lists the files found with extensions '.mp4', '.avi', '.mkv', '.mov', or '.wmv'. It extracts and plays the video as soon as the index of the array matches the file, incrementing each time it finishes playing the previous one and returning to index '0' when advancing from the last element of the array. This class also includes the addition of a new button, also with the tkinter library, to stop video playback at any time, also ending the audio and returning to the main menu.

Image:
The idea of the array that directly retrieves the number of images from the directory where they are stored is resumed. It increments as 3000 ms pass, meaning it changes the image after three seconds and continues playing the list in a loop.

Audio:
We implement an array that serves as a list and reference to audio files in formats '.mp3', '.wav', '.ogg', and '.flac'. NOTE: To listen to the audios, an audio output is required either via USB-A or through the 3.5mm JACK port provided by the Raspberry Pi.

Mixed:
For this case, an interface is initially displayed, also generated with tkinter, which allows us to select between accessing the video, image, or audio option. Depending on the selection, it will display the interface of the option and use its programming logic.

IMG: This directory stores all the multimedia content for the graphical interface, such as logos for applications that are referenced to a button that directs to open hyperlinks with the open-source Chromium codebase.

UTIL:
In this PATH, we find programming logic to adapt images to the interface using the PIL library, importing ImageTk. Similarly, this is where arrays are initialized for reference to various types of files, mounting and dismounting USB for external storage reading, as well as internet connection using NetworkManager. It also references buttons and images to hyperlinks we access with Chromium, modifying the interface to remove the typical navigation bar seen in a periphery to the web interface, as well as the search bar.

It is also worth mentioning the importance of the script that automates the configuration, initialization, and updating of packages, tools, and libraries necessary to run the multimedia center.

Integration of software components or modules into a software solution:
GUI
To have better control over the display windows, different methods and classes were configured to divide the functionalities as follows: - Classes stored in the GUI directory to display the customized graphical interface. - Methods to organize buttons related to images and clear panels. - Methods to handle the reading of four different types of files on USB. *homeScreen.py* - Allows viewing the main screen with the logos of the applications. *smartTvAppGui.py* - Displays the screen with the name of our SmartTV, author names, start button, and a menu with options.

Video:
- Class to handle video playback using the VLC library. - Array to list and play video files. - Button to stop video playback and return to the main menu. *videoFrame.py* - Allows playing a video behind another. *videoOptionsScreen.py* - Allows interacting with a menu to select the multimedia to play.

Image:
- Class to handle image playback with changes every three seconds. - Array to store and play images. *imagesScreen.py* - Allows displaying images from the USB in presentation mode.

Audio:

- Class to handle audio playback. - Array to list and play audio files in different formats. *musicScreen.py* - Allows playing music.

Mixed:
- Interface generated with tkinter to select between video, image, or audio options. - Specific programming logic for each selected option. *mixtoScreen.py* - Allows selecting the multimedia medium to be played, menu with options in icon.

IMG:
- Directory to store multimedia content for the graphical interface, such as logos for applications. - Reference to hyperlinks using Chromium.

UTIL:
- Programming logic to adapt images to the interface using the PIL library. - Initialization of arrays for reference to different types of files. - Mounting and dismounting of USB for external storage reading. - Internet connection using NetworkManager. - Referencing of buttons and images to hyperlinks. - Modification of the interface to remove the navigation bar and search bar. *networkScreen.py* - Allows viewing available networks to connect to.

Conclusion

The complexity of carrying out this project was understanding how the libraries for automatic content playback functioned, manipulating files from the terminal, and generally handling files. Some parts of the code had already been used in previous practices, putting into practice the knowledge acquired during the semester.

Finally, as future work, it is recommended to add functionality to control it via a remote control, as in this case, it is only operated through the keyboard. Although this report is not a tutorial, the purpose is to broadly explain how the project works.

References

[1] Sistemas embebidos (Integrados): principales aplicaciones. (n.d.). https://www.tecnologias-informacion.com/sistemasembebidos.html

[2] Fromaget, P. (2024, April 9). What is a Raspberry Pi?(Hardware, software, goal usage). Raspberry-Tips. https://raspberrytips.com/what-is-a-raspberry-pi/

[3] Isaac. (2020, December 3). OSMC: el centro multimedia para tu Raspberry Pi. Hardware Libre. https://www.hwlibre.com/osmc/

[4] Moraguez, E. R. (2024, April 29). ¿Qué es un Raspberry Pi? Cómo funciona y para qué sirve. — LovTechnology. LovTechnology. https://lovtechnology.com/que-es-un-raspberry-pi-como-funciona-y-para-que-sirve/

[5] Ltd, R. P. (n.d.). Operating system images – Raspberry Pi. Raspberry Pi. https://www.raspberrypi.com/software/operating-systems/raspberry-pi-os-64-bit

Additional

Video of functionality:
https://youtu.be/YzcmBHdCG1M
Repository:
https://github.com/Tonypina/SmartTV.git