Interim Report - Media Center

COE718 - Embedded Systems Design

Fall 2024

Eric Muzzo

Student # 501019745

Department of Computer and Electrical Engineering

Toronto Metropolitan University

1. Abstract

The media player being developed in this project consists of several modules that interface with different physical components on the MCB1700 development board. The concepts and methodologies employed learned in the course thus far will be used to achieve a media center with a photo gallery, MP3 player, and a game center. Each of these core modules utilizes the LCD screen, and joystick on the development board. This report serves as a progress summary of the ongoing development of the media center application.

2. Introduction

The media center as an entire application consists of several sub-applications; the main menu, the photo gallery, the MP3 player, and the game center. Since each sub-application contains its own logic, functionality and controls, a modular approach is taken to divide the project into smaller components. This means that the source code will consist of a C file for each sub-application, each with their own main function to run their specific program.

A modular design approach means that the code will be much more readable which also makes debugging much simpler. With this in mind, I considered what inputs, outputs, and functions each sub-application needed to perform to develop them separately.

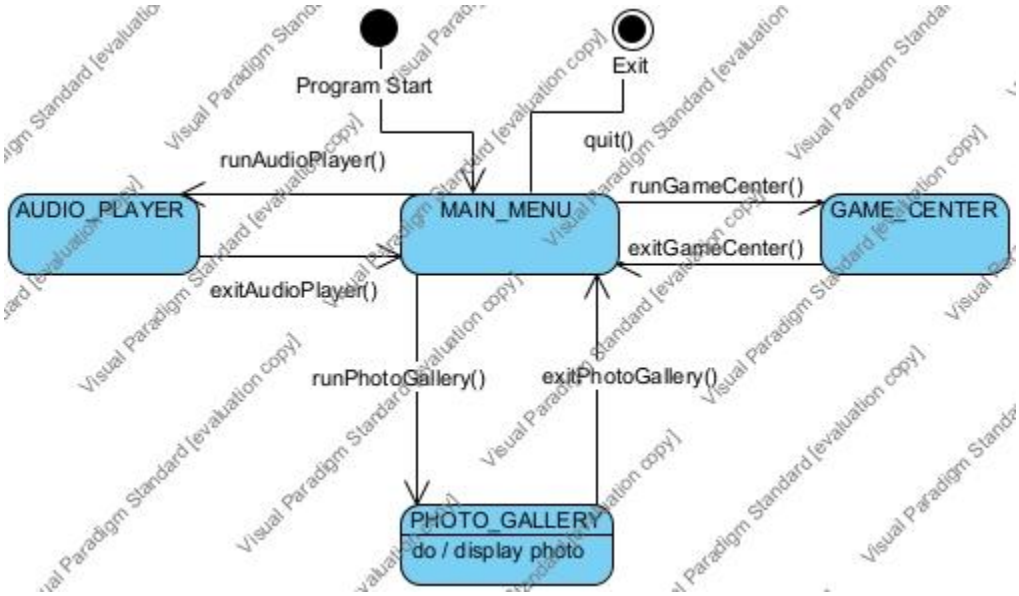

Figure 1: State Diagram

A state machine will be used to globally determine the state of the application. With this approach, the main C file will be able to easily remain in an infinite loop while constantly monitoring the application state. This allows for easy task switching to various sub-applications depending on the system state.

3. System Design

The media center is broken up into 4 core components; the main menu, the photo gallery, the MP3 player, and the game center. The source code is structured in a similar approach to achieve modularity. Each component has its own controller C file that handles all of the logic and I/O. A state machine will be used to determine the application state and transfer control to and from different components. Using a state machine allows for the main function to effectively monitor and task switch.

The main function initializes the app and all of the hardware components. An infinite

loop runs the program continuously while keeping track of the current application state. By calling the main functions of the other components (ie. photo gallery), control is passed to that sub-component and the state of the application is updated. This is implemented by having a pre-defined enumeration type for the application state. Each component's main function returns an AppState.

The main menu is relatively simple and contains a basic UI listing the 3 application options. This module is responsible for tracking the user's current selection while responding to inputs from the joystick. When the select button is triggered, this module's main function will return the selected AppState back to the main controller, which will then pass control to the specified app.

The photo gallery horizontally displays an array of C coded image files which the user can cycle through using the joystick. As the user cycles through the indices of the array, this module updates the LCD to display each image. If the user wishes to return to the main menu, they can do so using the joystick, in which case this module's main function will return the MainMenu AppState. Again, the media center's main loop will observe the state change and pass control back to the main menu.

The MP3 player will present a list of files for the user to select to play off of the development board. This module handles the logic of connecting to and disconnecting from the PC when entered into this state. The development board's onboard potentiometer will be used to control the volume output, with a similar implementation as done in the course labs.

Finally, the game center aims to house at the minimum one game; flappy bird. This has yet to be implemented and may change depending on timing constraints and difficulty. This game will involve several components; graphics controller to display image based game components, score keeping, and collision detection. Each of which will be developed using the divide and conquer approach. That is, breaking down each game function into its smallest components with simple inputs and output

4. Completed Work

Thus far, the application project directory has been laid out. This includes all of the sub-component application controller C files shown in *Figure 1*. I have defined the AppState enumeration type in its own file, where I can then import it to the rest of my modules. I have created the main controller function that runs an infinite loop using a switch statement inside of a while loop. This function calls the main functions of the rest of the modules.

I have completed the main menu module using the GLCD_DisplayString() function to list the applications for now. I plan to replace these with more UI friendly image

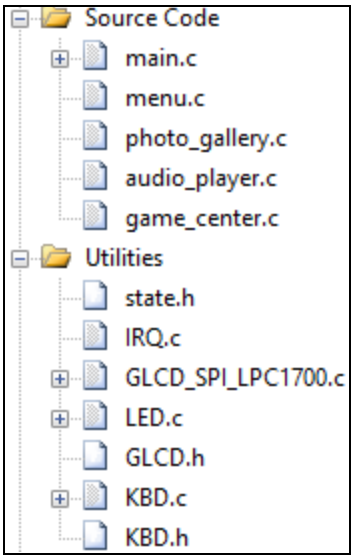

Figure 2: Project Directory

icons for each app. The main menu module keeps track of the cursor position, while also repeatedly calling an updateMainMenuDisplay() function to visually show the user what menu option is selected via the LCD.

I have also begun work on the photo gallery module. For now I have just converted some images into C source code and imported them into my project library. I have also begun the selection control mechanism for allowing the user to cycle through images, but have yet to display them to the LCD.

5. Remaining Work

In *Table 1*, the remainder of the work to be completed as well as a plan to complete it is outlined.

Task	Description	Target Date
Complete photo gallery module	Implement image display mechanism	Nov. 16
Build audio player module	Design and code the audio player module and functions	Nov. 18
First in-lab debug and testing	Test code on the dev board at the lab	Nov. 20
Build game 1	Build the c code for the flappy bird game	Nov. 22
Integrate game into game module	Implement the control logic to run the flappy bird game inside of the game center module	Nov. 23
Final in-lab debug and testing	Test full application	Nov 25

Table 1: Remaining Work Breakdown

6. Conclusion

As there still remains a significant amount of work to be completed for the project deadline, the use of a modular programming approach will allow me to effectively build out components of the app independently. This will result in a very simple integration of programs for the final result. I do feel as though the game component will be challenging since it is the most complex component. However, I believe that if I follow my plan for task completion, I will be on track for the project deadline.