Fall 2024

California State University, Northridge

Department of Electrical & Computer Engineering

ECE 425

Microprocessor Systems

Final Project Report

Buzzer-Enhanced Audio Tuning for Buzzer Output eXperience

(BEATBOX)

Performed By: Antonio Anzora Jr

Instructor: Aaron Nanas

Introduction:

In this project, I will design an interactive musical device using the TM4C123GXL LaunchPad micrcontroller, including a rotary encoder, LCD, and magnetic buzzer to create a customizable sound experience. The user will navigate through a menu using the rotary encoder, selecting different songs displayed on the LCD. Once a song is selected, the magnetic buzzer will play the corresponding musical tune.

Background and Methodology:

The Beatbox Project integrates several embedded systems concepts, utilizing the TM4C123GH6PM micrcontroller to control various peripherals, including the rotary encoder, 16x2 LCD, and DMT - 1206 magnetic buzzer. The system creates an interactive interface where users select songs through the rotary encoder, with song selections displayed on the LCD, and sound played through the DMT - 1206 magnetic buzzer.

Outcome:

The BeatBox project enables users to select and play different songs using the rotary encoder. The LCD provides visual feedback on the song selection, and the DMT- 1206 magnetic buzzer plays the chosen music, creating an engaging audio-visual experience.

Hardware Components:

- TM4C123GH6PM Microcontroller: The central unit controller for managing all peripherals.
- Rotary Encoder: Allows users to select songs by rotating and pressing the encoder.
- 16x2 LCD: Displays the current song selection and system status.
- DMT- 1206 Magnetic Buzzer: Plays the selected song

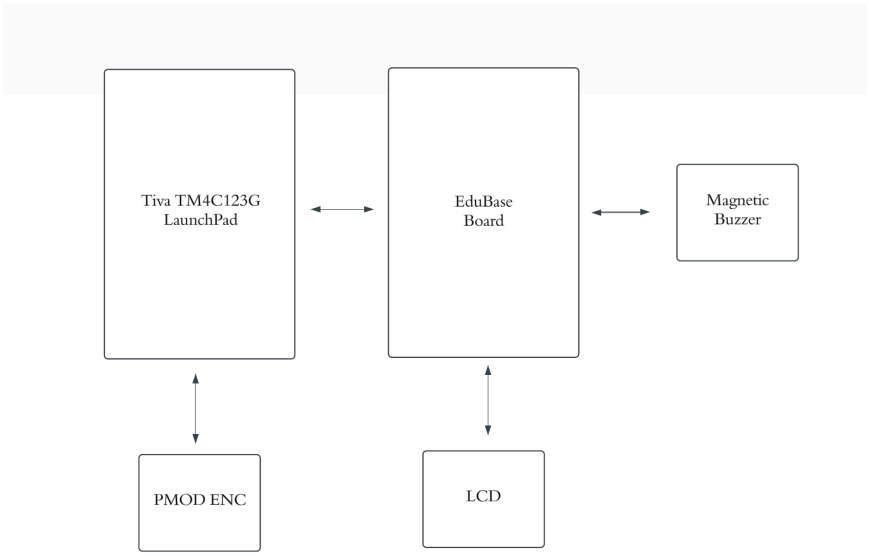
Software Components:

- Timer Interrupts: Used for time management when generating song notes and playing them through the buzzer.
- GPIO (General-Purpose Input/Output): For reading input from the rotary encoder and controlling the LCD and magnetic buzzer
- Keil uVision5: Integrated Development (IDE) for writing and compiling code for the TM4C123GH6PM microcontroller.

TM4C123GH6PM Peripherals Used:

- SysTick Timer: Used for managing delays in the system and creating precise timing for song playback.
- GPIO (General-Purpose Input/Output): Use for reading inputs from the rotary encoder (buttons and rotations) and controlling the output to the LCD and magnetic buzzer.

Block Diagram:



Pinout Used:

Peripheral	Tiva LaunchPad Pin	Pin Function	Notes
DMT - 1206 Magnetic Buzzer	PC4	GPIO Output	Outputs Sound
PMOD ENC	PD0	Pin 1 (A)	Output of button A in the encoder shaft
PMOD ENC	PD1	Pin 2 (B)	Output of button B in the encoder shaft
PMOD ENC	PD2	Pin 3 (BTN)	Output of the integral push button in the encoder shaft
PMOD ENC	PD3	Pin 4 (SWT)	Output of the onboard switch
LCD	PA2	Data Pin 4	Data for LCD Communication
LCD	PA3	Data Pin 5	Data for LCD Communication
LCD	PA4	Data Pin 6	Data for LCD Communication
LCD	PA5	Data Pin 7	Data for LCD Communication
LCD	PC6	LCD Enable Pin	Controls LCD data read/write
LCD	PE0	Register Select	LCD Control Pin for Data/Command

Components Used:

Description	Quantity	Manufacturer
Tiva C Series TM4C123G LaunchPad	1	Texas Instruments
USB-A to Micro-USB Cable	1	N/A
EduBase Board	1	Trainer4Edu
Small Flathead Screwdriver	1	N/A

Analysis and Results:

Figure 1: Displayed menu design on the LCD

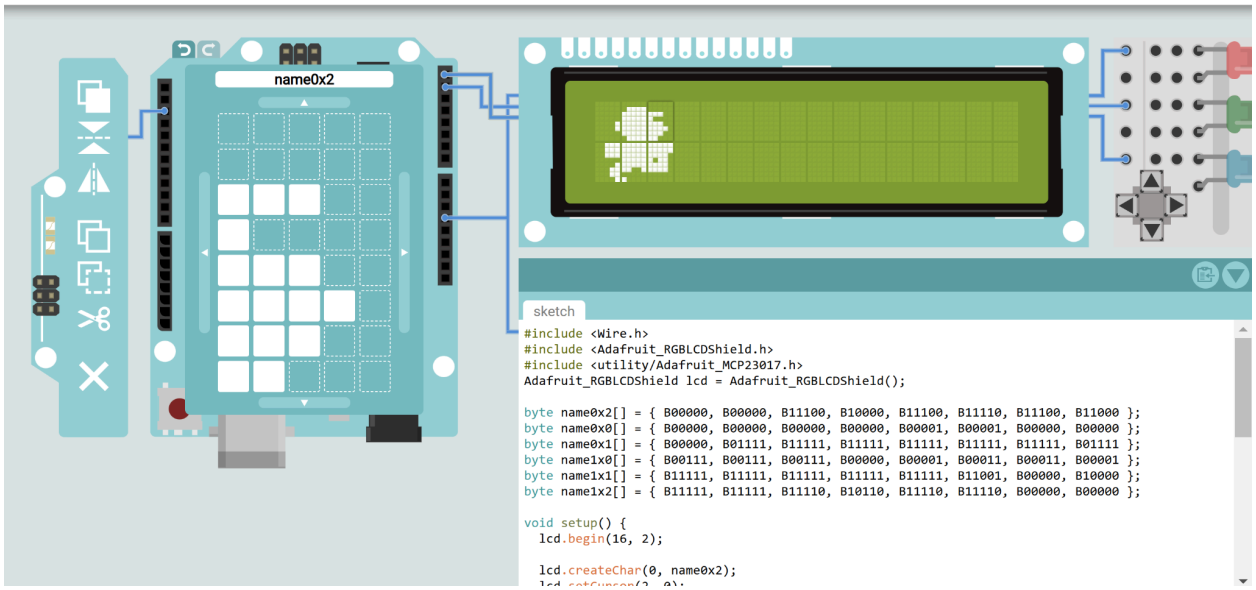


Figure 2: Simulated Mario Design Using LCD Design App

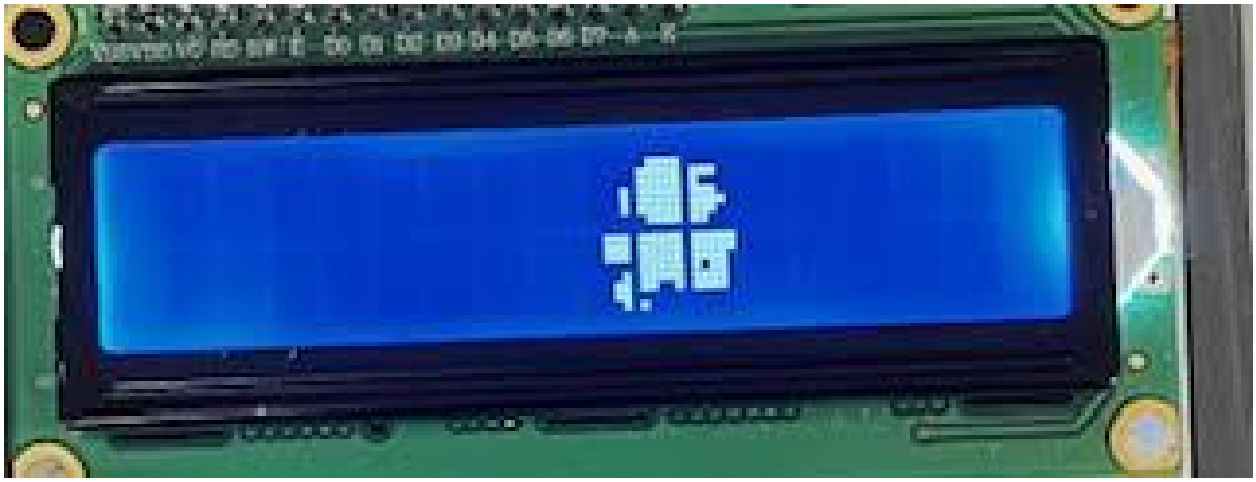


Figure 3: Mario Design Displayed on Physical 16x2 LCD

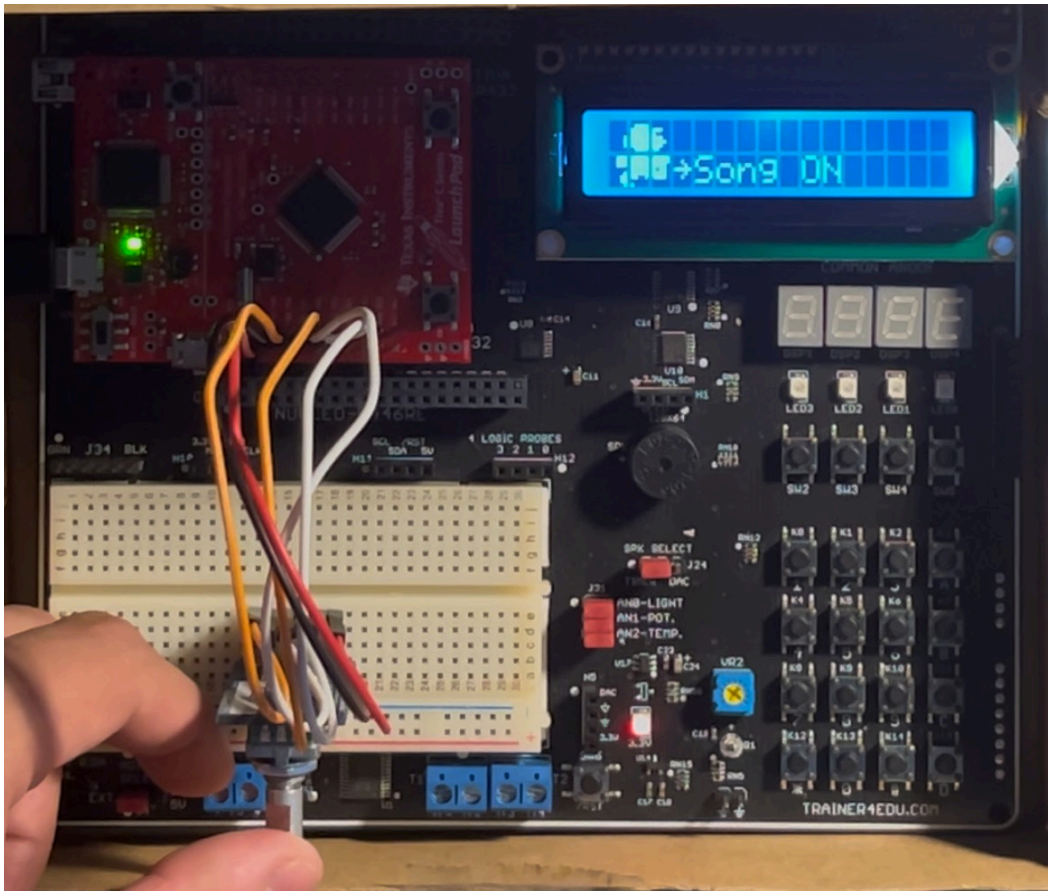


Figure 4: Mario Design Displayed on 16x2 LCD With Song Playing in the Background

```
442
443  //--------------MARIO BODY Header Functions--------------------
444 /**
454  void EduBase_LCD_Display_Mario_Right_Hat(void);
455
456 /**
466  void EduBase_LCD_Display_Mario_Torso(void);
467
468 /**
478  void EduBase_LCD_Display_Mario_Left_Leg(void);
479
480 /**
490  void EduBase_LCD_Display_Mario_Middle_Hat(void);
491
492 /**
502  void EduBase_LCD_Display_Mario_Left_Hat(void);
503
504 /**
514  void EduBase_LCD_Display_Mario_Right_Leg(void);
515
516 /**
517   * @brief Displays the complete Mario design on the LCD.
518   *
519   * This function updates the LCD to show the full image of Mario,
520   * combining all previously displayed parts into a cohesive design.
521   *
522   * @param None
523   *
524   * @return None
525   */
526  void EduBase_LCD_Display_Mario_Full_Image(void);
527
```

Figure 5: Mario Displayed on LCD Using EduBase_LCD_Display_Mario_Full_Image

```
enum Custom_Character_CGRAM_Locations
{
  //MARIO BODY LOCATION
  MARIO_RIGHT_HAT_LOCATION        = 0x00,
  MARIO_MIDDLE_HAT_LOCATION       = 0x01,
  MARIO_LEFT_HAT_LOCATION         = 0x02,
  RIGHT_ARROW_LOCATION            = 0x03,
  MARIO_RIGHT_LEG_LOCATION        = 0x04,
  MARIO_TORSO_LOCATION            = 0x05,
  MARIO_LEFT_LEG_LOCATION         = 0x06,
```

Figure 6: Custom Character CGRAM Locations for Mario's Body Parts

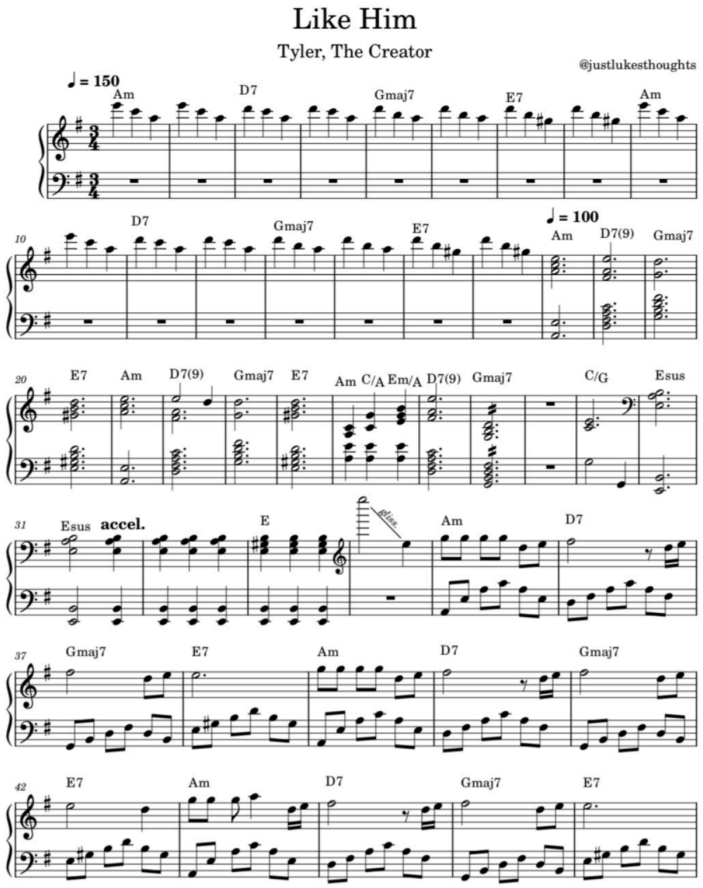


Figure 7: Tyler, The Creator - Like Him Music Sheet

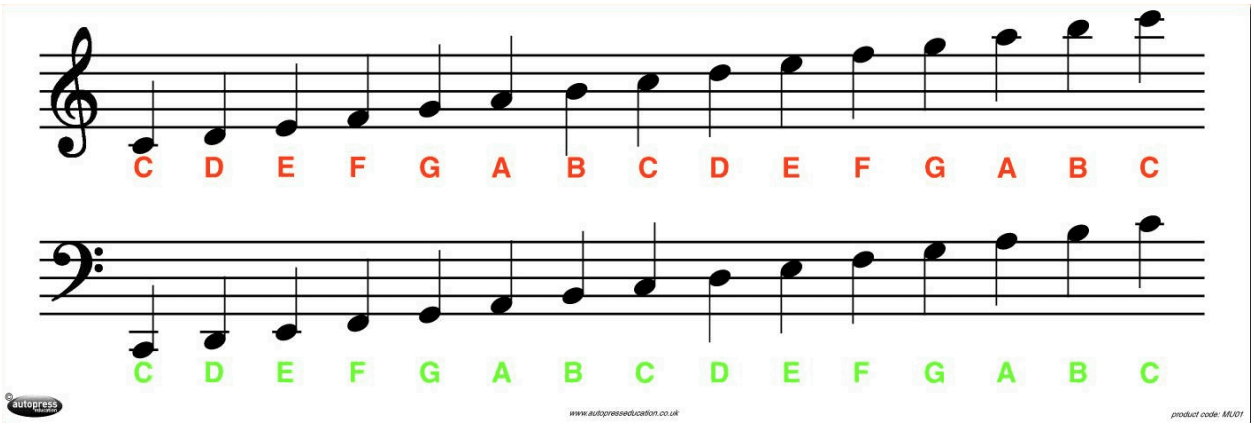


Figure 8: Music Note Chart Guide

Figure 7 displays Tyler, The Creator's piano sheet, while Figure 8 helps explain the musical notes. Together, the figures highlight both the creative style and the musical structure.

Since this project focuses on the design of the BeatBox, its functionality is currently limited to displaying Mario on the LCD. It does not support other designs at this stage. The following tasks can be performed with the BeatBox:

- Display Mario's full image on the LCD screen
- Play a selected song when the user selects an option via the rotary encoder
- Provide real-time visual feedback on the LCD for song selection and current status

The main source file contains the following functions:

- *PMOD_ENC_Task()*
    - Handles the PMOD encoder's state changes, detects button presses, and tracks the encoder's rotation.
- *Display_Main_Menu(int main_menu_state)*
    - Displays the main menu on the LCD using a switch statement to choose between options like 'Super Mario Bros', 'Doom Theme', and 'Like Him'.
- *Process_Main_Menu_Selection()*
    - Performs playing the theme songs based on the selected menu state or actions like displaying Mario's full image.
- *EduBase_LCD_Int()*
    - Initializes the LCD. This function configures the LCD to prepare it for displaying text or custom characters.
- *EduBase_LCD_Create_Custom_Character()*
    - Creates custom characters for the LCD, such as Mario's body parts.

- *EduBase_LCD_Clear_Display()*
    - Clears the LCD Screen to reset the display area, following new content to be shown.
- *EduBase_LCD_Set_Cursor()*
    - Moves the cursor to a specified position on the LCD screen. This is used to place characters or strings at specific locations.
- *EduBase_LCD_Set_Data()*
    - Sends data to the LCD. This is used for displaying characters.
- *EduBase_LCD_Display_String()*
    - Displays a string of characters on the LCD, starting from the cursor position.
- *EduBase_LCD_Scroll_Display_Right()*
    - Scrolls the content of the LCD to the right. This function is used to create the scrolling effect for Mario's image.
- *EduBase_LCD_Scroll_Display_Left()*
    - Scrolls the content of the LCD to the left. This function is the opposite of the previous function.
- *Play_Super_Mario_Bros_Theme()*
    - Plays the Super Mario Bros theme song. This function triggers the buzzer to output the sound associated with the theme.
- *Play_Doom_Theme_Song()*
    - Plays the Doom theme song using the magnetic buzzer.
- *Tyler_The_Creator_Like_Him()*
    - Plays "Like Him" by Tyler, The Creator using the magnetic buzzer.

- *Buzzer_Init()*
    - Initializes the buzzer connected to the microcontroller.
- *SysTick_Delay_Init()*
    - Initializes the SysTick timer, which is used to create delays in the program.
- *SysTick_Delay1ms()*
    - Creates a delay of 1 millisecond. The function introduces pauses for timing-related tasks, such as scrolling the display or waiting between actions.
- *PMOD_ENC_Init()*
    - Initializes the PMOD encoder interface, configuring the necessary hardware for reading the encoder's input.
- *Timer_0A_Interrupt_Init()*
    - Initializes Timer 0A to trigger interrupts, periodically calling the PMOD_ENC_Task() function to check the encoder state.
- *PMOD_ENC_Get_State()*
    - Retrieves the current state of the PMOD encoder, used to check its status for changes in rotation or button presses.
- *PMOD_ENC_Button_Read()*
    - Reads the button state of the PMOD encoder, used to detect button presses and determine when the user interacts with the encoder.
- *PMOD_ENC_Get_Rotation()*
    - Retrieves the encoder's rotation value, indicating how far the encoder has been turned.

Video Demonstration Links:

- Super Mario Bros Theme and Doom Theme Demonstration: https://drive.google.com/drive/folders/1fADChxdK__N7KiT_LdD1QoKrhce_fi9T?usp=drive_link
- Tyler, The Creator - Like Him Demonstration:https://drive.google.com/drive/folders/1rKp0JsDf6UEdUzbWrCa92Q6y5mm0yF5g?usp=drive_link

Conclusion:

While working on this project, I learned how to implement custom characters onto the LCD. However, I realized that I was limited in terms of the number of custom characters I could display, as most LCDs only support 8 CGRAM characters. If I were to use a different type of LCD, one that supports more than 8 CGRAM characters, I could display more complex and unique characters. My original idea for the LCD was to display Mario and a star, where Mario would chase the star across the screen. Once Mario reached the star in the next frame, the star would disappear, triggering the start of the Super Mario Bros theme song. However, I encountered challenges with playing music on the magnetic buzzer. The buzzer struggled to produce songs with a lot of melody and bass, as the sound quality and repetitiveness weren't the best. In the future, I could explore using a modified speaker with better sound control and the ability to adjust the volume. This project helped me understand the limitations of certain hardware components, such as the buzzer's inability to repeat multiple notes and the LCD's restrictions of 8 CGRAM characters. Despite these limitations, I enjoyed working with the rotary encoder and designing the LCD menu. The smooth functionality of the rotary encoder for navigating options was one of my favorite parts of the project.

References:

PMOD ENC (Rotary Encoder) Data Sheet: Digilent, Inc. (n.d.). *PMOD ENC Data Sheet*. Retrieved from https://digilent.com/reference/_media/reference/pmod/pmodenc/pmodenc_rm.pdf?srsltid=AfmBOoqGuRduKRyqDuDW_Uk6_oYhJHZM36cV5-Gg02aYbkueOgXW91SC

TM4C123GH6PM Microcontroller Data Sheet: Texas Instruments. (n.d.). *TM4C123GH6PM Microcontroller Data Sheet*. Retrieved from https://www.ti.com/lit/ds/symlink/tm4c123gh6pm.pdf

Musical Keynotes: Robson Couto. (n.d.). *Arduino Songs*. Retrieved from https://github.com/robsoncouto/arduino-songs