CMPT433 Project Write-up

Group123

System Explanation:

The system is a music player with basic functionality such as: music selection, replay, shuffle, pause, and volume control. An external 16x32 RGB LED matrix display is also employed to show realtime information about the music and the current state of the system.

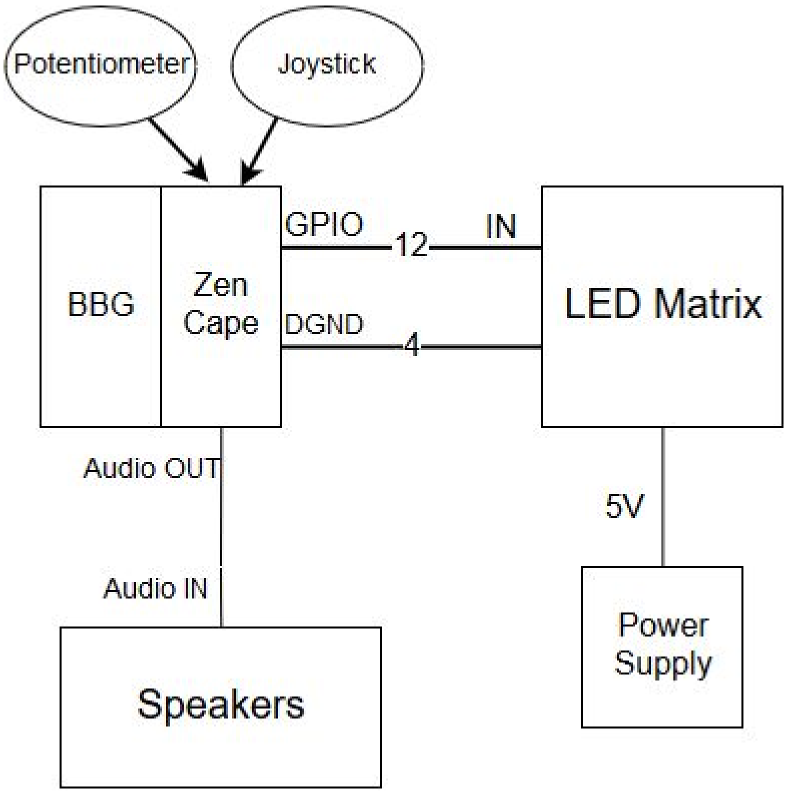

Figure 1. System block diagram.

On boot up, the user is greeted with an SFU splash screen while the system is reading the metadata of the MP3 and WAV files in the song directory. The SFU splash screen will end and the song library menu will be displayed once all metadata is gathered. Metadata for MP3 files such as song name, artist, album and year are gathered using the mpg123 library, whereas for WAV files, song name and artist are extracted from the filename since WAV files do not inherently have metadata.

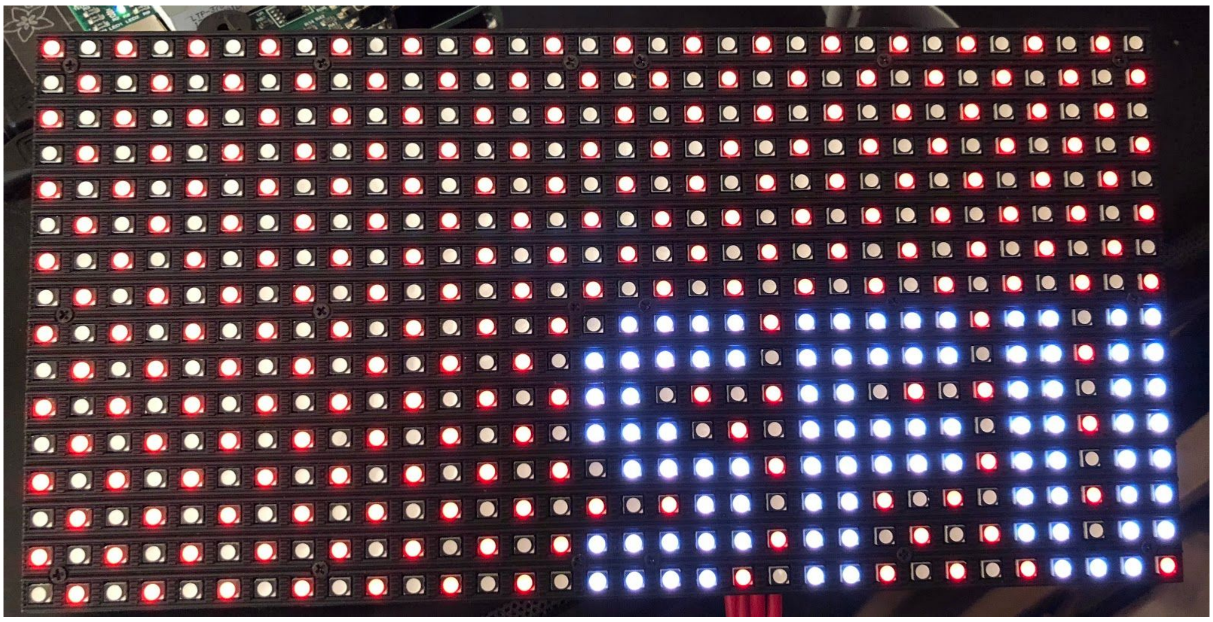

Figure 2. Boot-up screen that appears when initially turning on system.

Within the song library menu, the user can navigate with the ZenCape's joystick UP/DOWN. The user can press joystick RIGHT to view more information about a song.

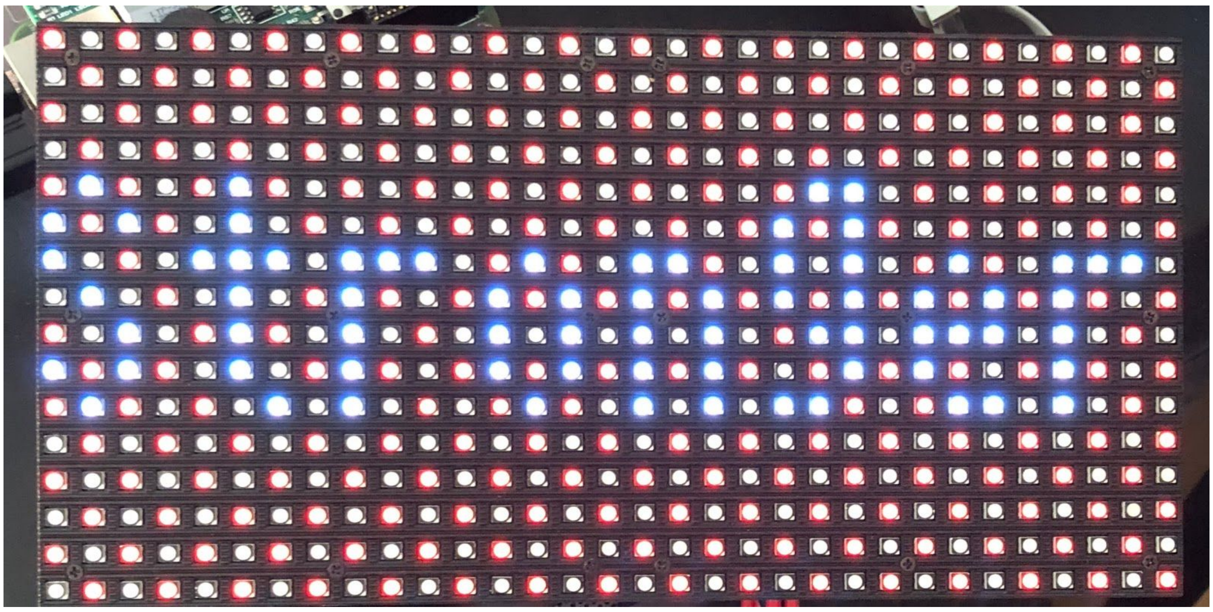

Figure 3. Song library menu, only shows one song at a time due to the limited size of display. Red background and white text.

To indicate the user is viewing information about a song, the text will turn cyan. The user can navigate through the song information with joystick UP/DOWN to view the song's artist, album and year.

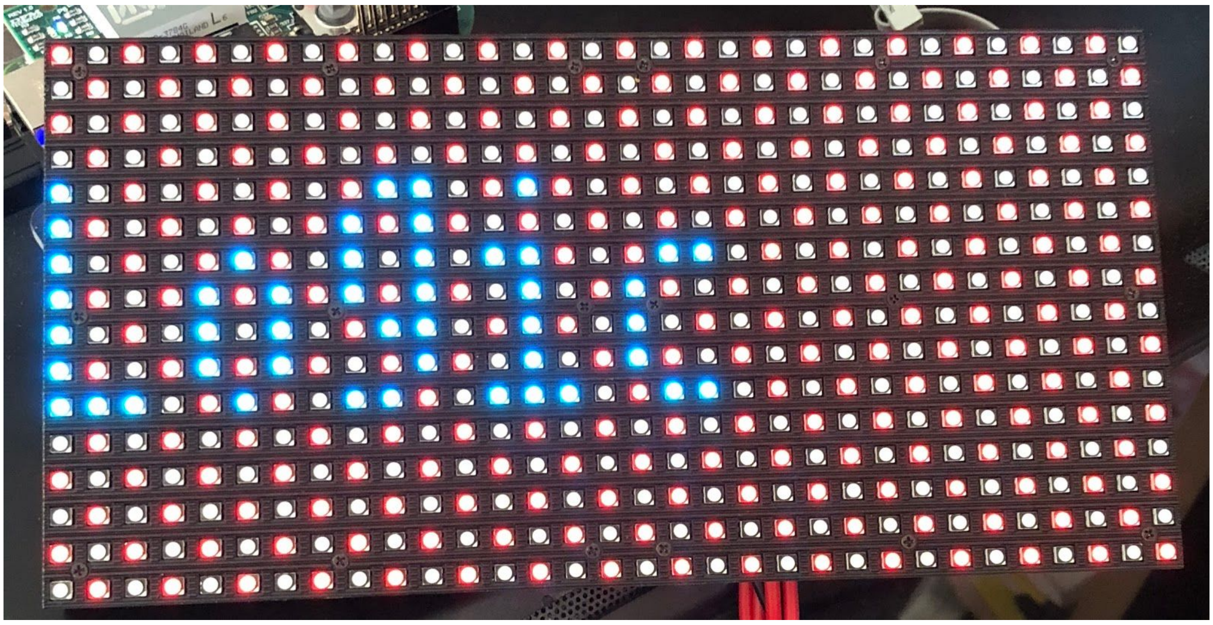

Figure 4. Information list of a particular song, cycle through the artist, album, and track name using joystick UP/DOWN. Red background and cyan text.

To play a song, the user can navigate back to the song library menu by joystick LEFT (similar to a back button) and press joystick IN to play a song. The ALSA asound library is used for audio playback. When a song is playing, the LED Matrix display will showcase the current song's name as well the current playing time and total song duration. This screen can be referred to as the "current track display". One special feature of the current track display is that if the track name at the top is too long, the track name will have a "sliding" animation. This feature was difficult to implement because it required techniques such as getting the text to wrap around the display without causing segmentation faults when sliding the text off screen.

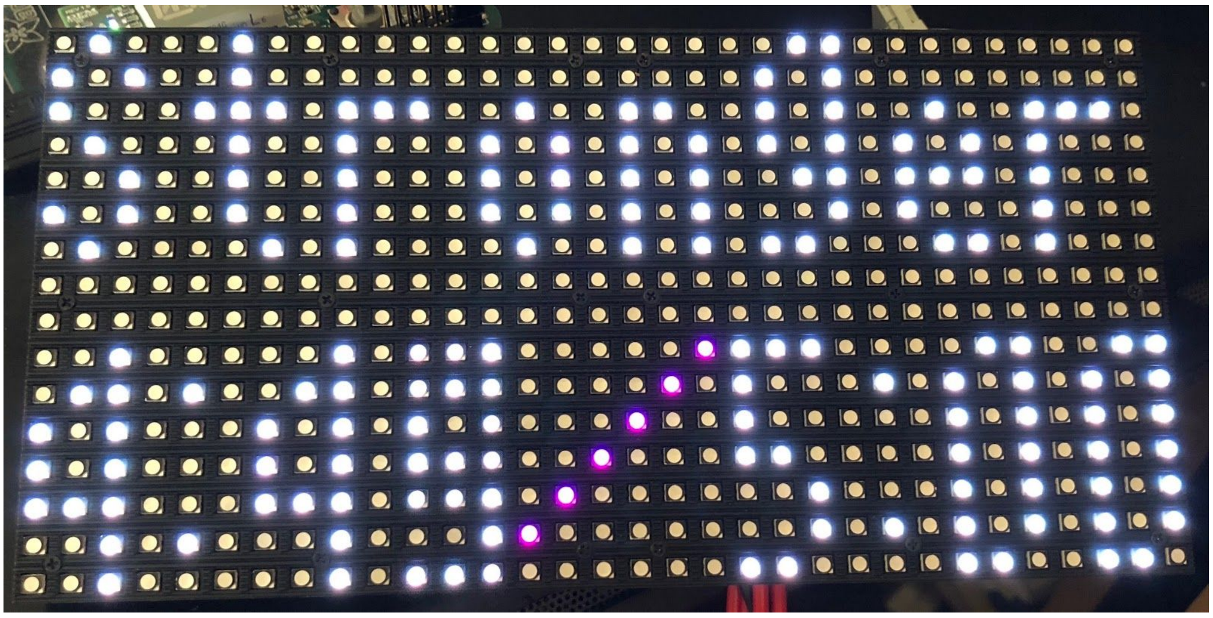

Figure 5. Current track display giving real-time feedback of current status of song playing.

The joystick functionality differs depending on whether the user is on the current track display or on the song library menu. As the user is on the current track display, joystick UP is used to toggle the repeat functionality of a song. Joystick DOWN toggles the shuffle functionality of a song. Joystick LEFT/RIGHT will play the previous and next song. Joystick IN will play/pause the current song.

To switch between the current track display and song library menu. The song library menu can be reopened by holding on joystick UP. Exit the song library menu with joystick LEFT. To adjust the volume of the song, the user can control the potentiometer on the BeagleBone.

When the program begins, the main thread initializes the modules, starting a thread for the splash screen. When the initialization is complete, it stops that thread, then constantly refreshes the display and updates the timer. One thread listens for joystick inputs, while another checks for potentiometer changes. A main audio thread manages audio playback threads (and sleeps when not needed). When changing to another song, the thread associated with the currently playing song is closed, and a new one created.

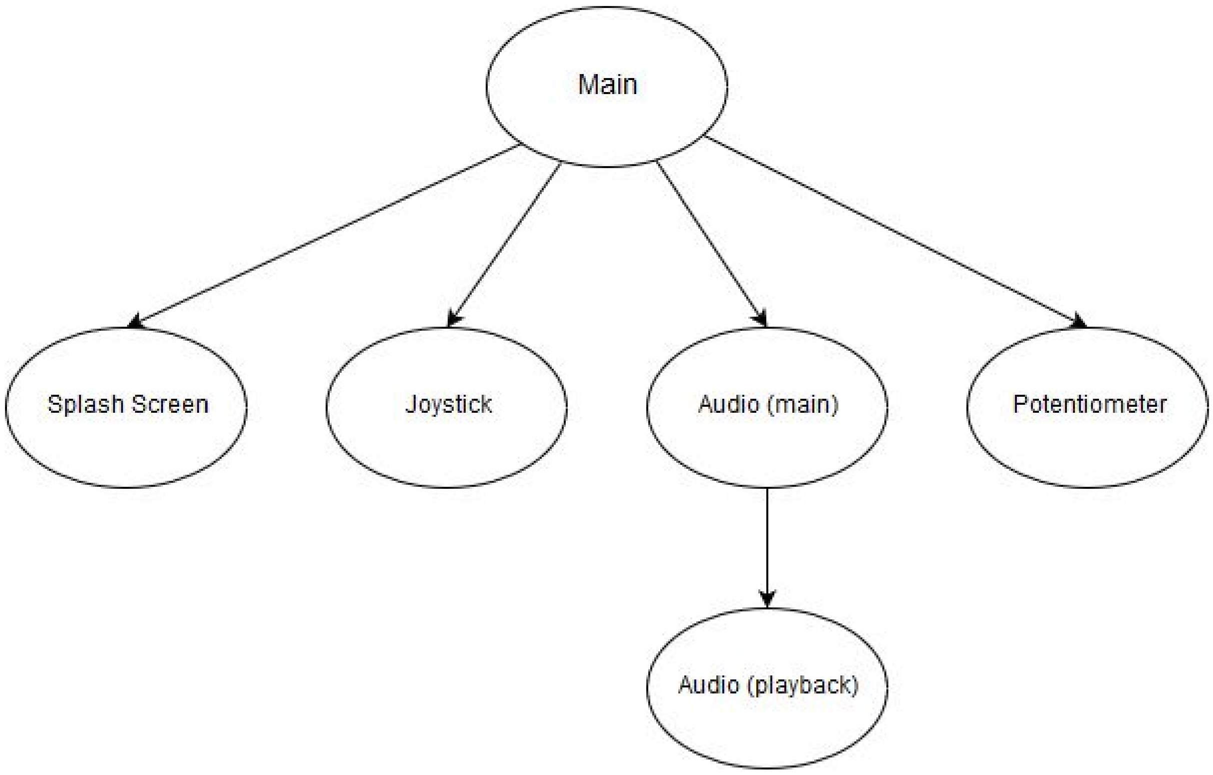

Figure 6. Thread organization diagram.

Functionality Not Working Well

Audio underruns cause the song to stop playing for a short time (<1s), and cause the LED Matrix screen to go blank for an instant. These are infrequent, happening once at the start of every 10 or so songs. They are easily detectable by sight or sound, and the audio library will write to console if it occurs as well.

Another issue is LED flickering, which was caused by multithreading because we noticed that as the number of threads increased so did the obviousness of the flickering. The solution that we believe can be used to resolve this issue is to incorporate the PRU (programmable realtime unit) into the source code to run code independently from the main BeagleBone Green CPU.

Features Table:

Features / Description	Completeness	Code	Notes
WAV Files audio playback	4.5	C	Occasionally underrun due to the processing power required for the LED Matrix Display. Used ALSA for writing to PCM.
MP3 Files audio playback	4.5	C	Occasionally underrun due to the processing power required for the LED Matrix Display. Used ALSA for writing to PCM. Used mpg123 for decoding and metadata extraction.
Audio volume control through potentiometer	5	C	
Audio playback features through joystick ● Play next/prev ● Play/Pause ● Toggle Repeat ● Toggle Shuffle	4.5	C	Shuffle uses a stack data structure to keep track of previous shuffled songs but not upcoming shuffled songs. So next shuffled song may repeat played songs.
Song menu through joystick	5	C	
LED Matrix Display	4	C	Issues of flickering were present, issue could have been mitigated by implementing PRU, but due to time constraints and the need to completely refactor the code we did not attempt this daunting task. The base code of the LED matrix display is obtained from Brian Fraser's website under the guides in the "Links" section. The link is provided below: https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2015-student-howtos/Adafruit16x32LEDMatrixGuideForBBB-Code/

Extra Hardware

Adafruit 16x32 LED Matrix Display (A how-to guide has been created detailing how to get it working)

Extra Software

mpg123 (for MP3 decoding and metadata extraction)

<u>Instructions for working with mpg123:</u>

1. Install mpg123 on target by running

```
# apt-get install mpg123
```

2. Copy libmpg123.so.0 on target to directory asound_lib_BBB, then rename it to libmpg123.so

First:

```
# cd /usr/lib/arm-linux-gnueabihf/
```

Then:

```
# cp libmpg123.so.0 /mnt/remote/asound_lib_BBB/libmpg123.so
```

Troubleshooting: If libmpg123.so.0 is not there, run the following to find the file path.

```
# ldconfig -p | grep libmpg123.so.0
```

3. On host, navigate to project root directory, run

```
$ make
```

Troubleshooting: If you get this error:

```
$ fatal error: mpg123.h no such file or directory
```

Then run:

```
$ sudo apt-get install libmpg123-dev
```

Setbacks / Challenges

There was setbacks at the start of the project when we were unable to properly fit jumper wires from the LED display to the pins situated on the ZenCape. After discussing this with Brian during office hours it was discovered that the pins were not tall enough resulting in the female end of the jumper wires to always bounce out when attempting to insert them. We were able to luckily resolve this by using Arduino stackable headers which surprisingly did not bounce back out when we inserted them into the ZenCape's pins which extend the BeagleBone Green's pins.

When reading audio files, we initially loaded the entire file into memory when the song was chosen, causing long delays before the song actually began playing — especially long for MP3 files because of the decoding. To resolve this issue, the program was changed to only read enough to fill the audio buffer.

Acknowledgments / References

We would like to first acknowledge Brian Fraser and Andy Sun for providing help any questions our group had during office hours and on Piazza. Also, the list of guides we would like to acknowledge that we referred to or looked at can be found below.

16x32 RGB LED Matrix Display References and Guides:

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2015-student-howtos/Adafruit16x32LEDMatrixGuideForBBB-Code/

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/GPIOGuide.pdf

https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2015-student-howtos/Adafruit16x32LEDMatrixGuideForBBB.pdf

https://learn.adafruit.com/connecting-a-16x32-rgb-led-matrix-panel-to-a-raspberry-pi?view=all

Mpg123 Library API Reference:
https://www.mpg123.de/api/modules.shtml