Final Project: Music Player

Austin Sypolt
Casey Sobecks

ECE362
Fall 2018

Table of Contents:

Introduction | 3

List of Figures | 4

Additional Elements | 6

Software Implementation | 6

Design | 7

Division of Work | 10

Conclusion | 10

Code - Attached as a word document with project

Introduction

The purpose of our lab project is to apply knowledge gained over the course of a semester in ECE362 for the intent of designing a functional music player on our 68HC12 microcontrollers. To create this project we will be using CodeWarrior IDE to write assembly instructions to our microcontrollers. Our project,the music player, is able to play 3 songs using the buzzer by sending sequences of notes to it. There is area for the user to login or create a new user with flashing LEDs to indicate the validation process is in progress. Upon successful login the user will be able to shuffle through the listed songs and play their selected song based on the position of switch 1. A battery is also incorporated into the program (enable/disable with switch 7), the value left in the battery can be determined by the speed of the DC motor. The stepper motor will also turn every second assuming a song is currently being played, and the push and IRQ buttons will be used for skipping a song and toggling the LCD display respectively.

List of Features:

Required Project Elements:

Login

The login displays two options, allowing the user to press 1 and create a new user or press 2 and login as an existing user.
Upon desire to create a new user a 4-bit username and password will be required to be entered. Once the new user is created, they will be redirected to the main menu once again and will now be able to enter the existing user window by pressing 2.
Upon entering the existing user window, the user will be able to enter a 4-bit username and password. If incorrect an error message will be displayed and then the user will be given another chance to properly login. If correct the user will be entered into the music player program.

LEDs

While in the login screen the LEDs oscillate between the leftmost and rightmost bits for visual effect. This is implemented with a RTI.
Upon successful login by the user the LEDs will turn off.

Songs/Speaker

Our music player has three programmed songs. These songs are:

1. Darude - Sandstorm
2. Smash Mouth - Allstar
3. Pokemon Theme Song

DC Motor

The DC motor is used as the music player's rechargeable battery. When the battery is fully charged the motor spins at max speed and as the charge decreases the motor's speed will decrease accordingly (linearly). The battery lasts about 5 minutes. When the battery dies and the motor stops moves the music player should stop any music currently playing and log the current user out back to the main login page.

Switches

Switches are used for controlling the music player after a successful login. If switch 1 is set to high (1), the current song will be paused. Upon pause the LCD will display 'Paused' on the screen and the program will remain paused until further operation by the user.

Upon switch 1 being set to low (0) the music player will resume the song from its paused spot.
Switch 7 acts as the charger for the battery/DC motor. If switch 7 is high (1), the battery should return to full charge and the motor should spin at maximum speed again. Upon switch 7 being set to low (0), the battery should start decreasing charge as normal again.

Stepper Motor
The stepper motor should tick for every second in the song. One full rotation of the stepper motor should be 1 minute of a song being played. If the song is paused, then the stepper motor will be paused as well.

Keypad
The hex keypad is used to select menu items as well as being able to properly type your username and password for login and account creation.

Potentiometer
The potentiometer will be used to shuffle through the songs on the music player.
-If potentiometer value 0-40, Song 1: Darude- Sandstorm will play.
-If potentiometer value is 41-80, Song 2: Allstar- Smash Mouth will play.
-If potentiometer value is 81-127, Song 3: Pokemon Theme Song will play.

IRQ Button
Upon the IRQ button being pressed the LCD will become a blank display, the music will continue to play. Upon a second press of the IRQ button the LCD will reappear and the program will continue working as before.

RTI
The timer for the battery is controlled by the RTI, for every second that passes the timer will increase by 1, and the speed of the DC motor is dependent on the value of the timer.

Push Button
Upon press of the push button the currently playing song will stopped and the next song in order will play.
-If song 1 is being played and PB is pressed song 1 will stop and song 2 will begin.

Additional Elements of the Project

Volume Button

Our first additional feature was the inclusion of a volume button. This allowed our speaker to change between two volume levels upon flipping the bit 5 switch.

Account Logon Security

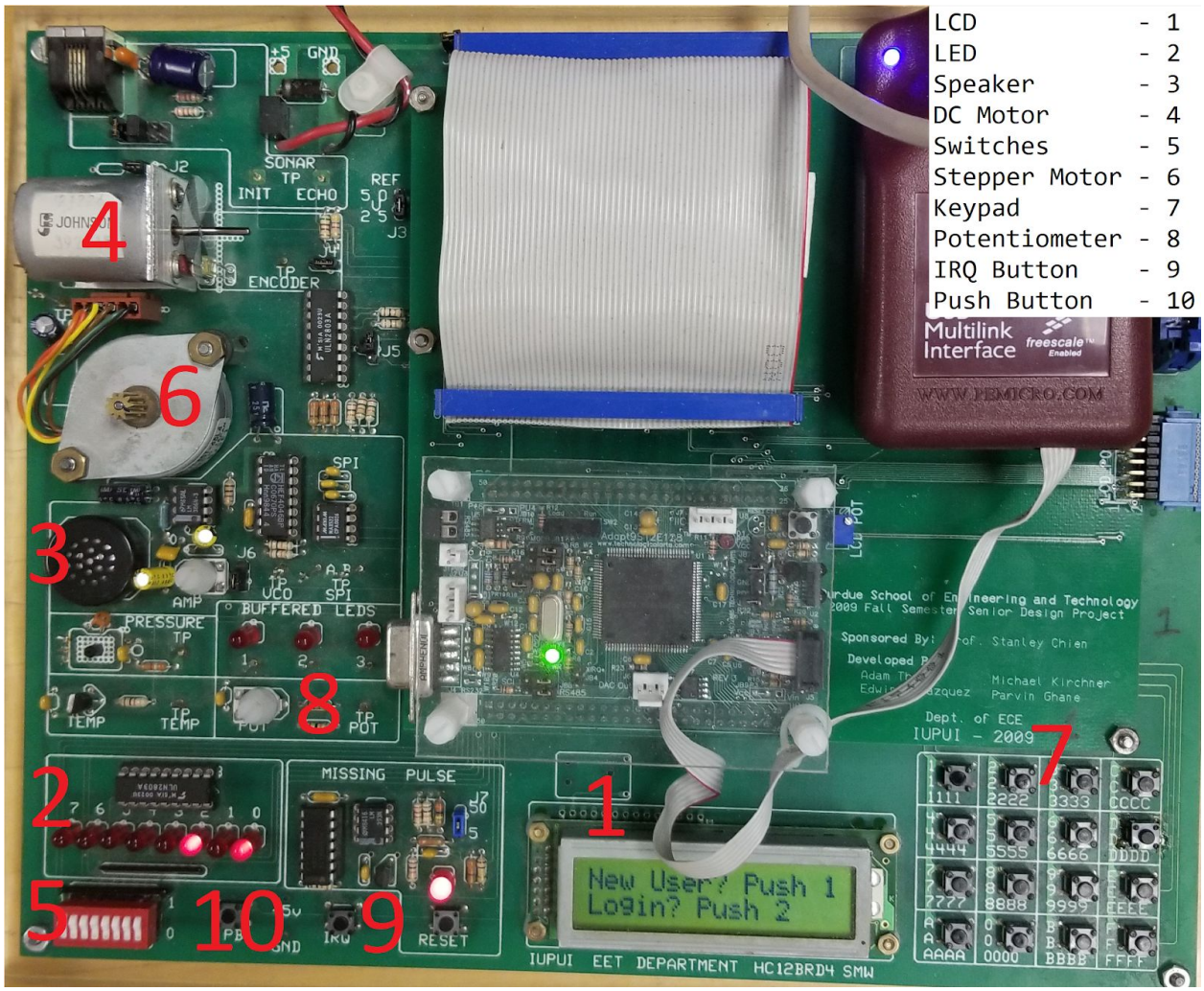
Our second feature was the implementation of x's into our project LCD display to provide further ambiguity to people trying to steal our precious music player information, as well as giving the user identification where they are in the input process.

Software Implementation

In the development of our music player there were various software aspect used in our 68HC12 microcontroller. Control variables, song tables, subroutines and interrupts were all used in the implementation of our program to ensure it worked properly and efficiently.

- **Control Variables**
  These were used throughout the project to hold the value of states so that the program runs through the proper layout design and the program works properly.
- **Song Tables**
  3 song tables were implemented, these were sequences initialized at the start of the program containing a list of notes to be indexed through and outputted to our speaker, playing the chosen song as directed by the sequence of notes.
- **Subroutines**
  Subroutines were used when trying to manage larger areas of code easier, or deeper code implementation just wasn't doing the trick
- **Interrupts**
  Changes were made to our linker file (.prm) to use 2 interrupts, the IRQ and the RTI. The IRQ was used for the clearing and reappearing of the LCD display upon press of the IRQ button. The RTI was used to properly run through every element of the program in a sequential order whilst minimizing hardware overhead and speeding it up through limiting the use of manually implemented debounces and delays.

Motorola 68HC12 Microcontroller



The Motorola 68HC12 microcontroller. The labelled portions are all the present elements on the board that were used for features in our project.

Login Page

New User Page

Existing User Page

Music Player Screen (Whether song 1 automatically plays depends lays first song depending on the switch 1 value, high bit set plays the song, and low bit pauses it)

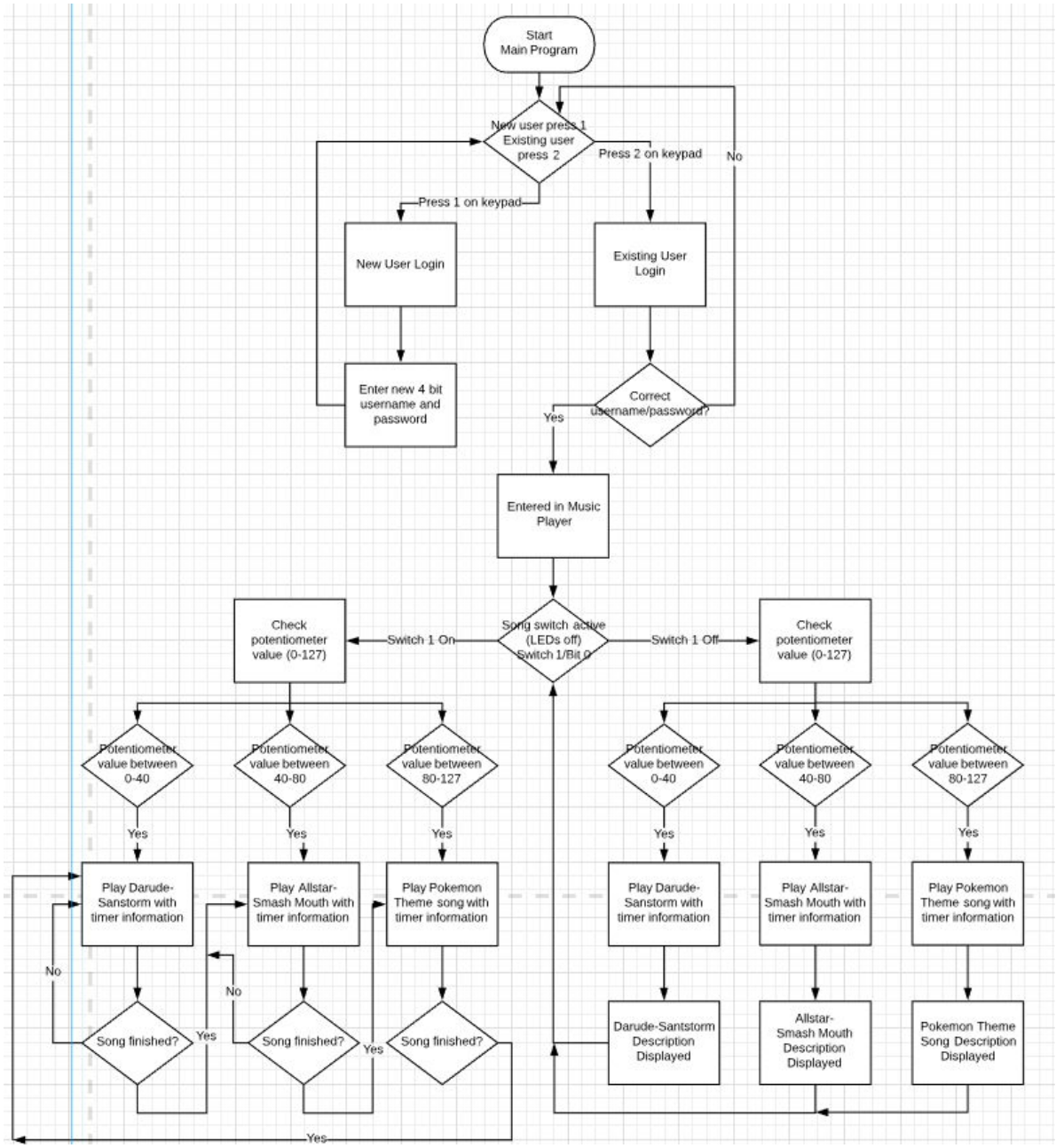
Paused Music Screen

Music Player Shuffle (The song displayed should change when the potentiometer is turned).

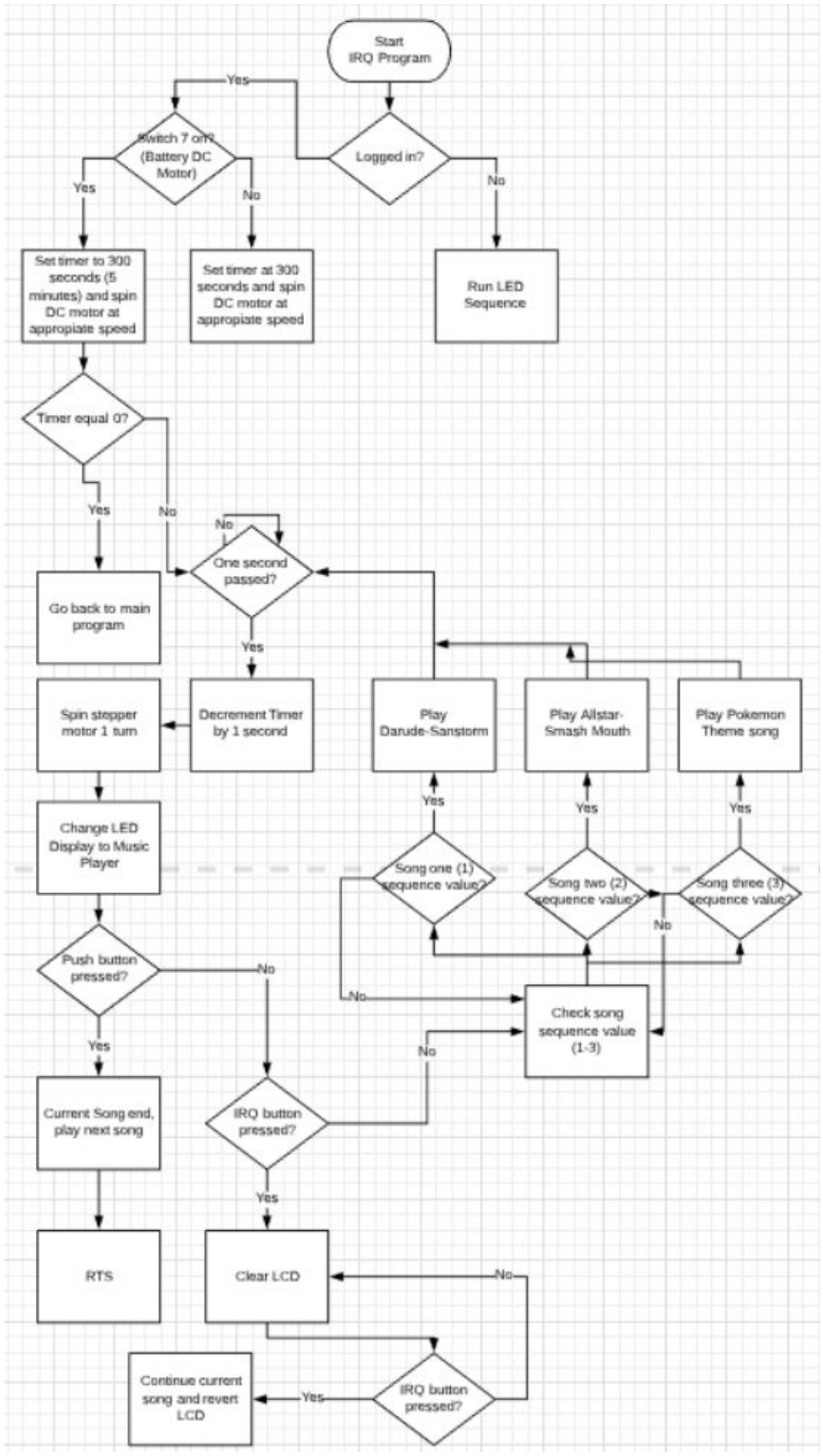
Implementation and Design of Program:

The flowchart designs visible on the following page give insight and enlightenment to the path our group followed for this project, the code is implemented accordingly as well to allow for ease of understand and use by the project members.

Main Program Flowchart:



IRQ Program Flowchart:



Incomplete areas of the project:
Upon the completion and presentation of our project there were a few aspect missing. This includes:
- Inability to get our music player to run properly
- Stepper motor doesn't run with time
- DC motor doesn't spin properly
- IRQ double press doesn't reappear LCD
- Push button implementation does not exist

These areas of our project were left incomplete due to time constraints and will to work on literally anything else.

Division of Work
Our team consisted of two members, Austin Sypolt and Casey Sobecks. The workload was divided up evenly and for the most part all of the additions to the project were done while working on it together either during lab time or present in the lab at another time throughout the week. Casey took the lead on the initial coding implementations of the project working heavily on the login portion of the project. The focus of organization of the project workflow was done by Austin in an effort to keep the code as clean as possible, however eventually the size of the program just led to more and more organizational problems causing various difficulties while attempting to add the areas mentioned above.

Conclusion
Although not all required elements of our project were completed the learning outcomes gained from this project were immense. We successfully learned how to use various aspects and applications presented to us by the 68HC12 microcontrollers. This was done by writing assembly language code learned from ECE362 and the various lab applications.