EC 450 MICROPROCESSORS
FINAL PROJECT

Bluetooth Singing Sign

By Jackson Hsu and Krystal Kallarackal

GOAL

The purpose of our project was to build a challenging and interesting device that did a certain number of functions with the MSP430G2553 launch pad. Our goal was to develop a device with all the components we had readily available and utilizing all the components' features. The components we had readily available were a 16x2 LCD display, an 8 ohms Speaker, and a HC06 Bluetooth Module. With these components we planned to build a device that would display a message, play a sound (song), and be controlled remotely via Bluetooth and voice commands. A usage for this device would be to install this outside an office door or an entrance to a room or building. The user would connect to the device using their phone and, using voice commands, instruct the device to perform certain actions. Actions could range from welcoming guests with a cheery song or, for more sinister purposes, chase them away with sirens and warning messages.

DESIGN

Component Set-Up:

The parts we used to design our project are a MSP430G2553 (and its pin connector wires), a breadboard (and wires), a HC06 Bluetooth module, an 8ohms Speaker, an Android Phone, and a 16x2 LCD display. Please refer to the last page for Figures 1-3 for the schematic and pictures of the setup.

LCD SETUP

To set up the LCD we soldered 16 pins into the LCD's 16 holes for the pins. We made sure none of the soldered pins were touching. Then the pins were placed into the breadboard along with the LCD as shown in Figure 3. Pins 1 and 16 were connected to ground. Pins 2 and 15 were connected to power source (VCC). LCD pins 4, 5, 6, 11, 12, 13, 14 were connected to the MSP430 pins P2.4, P2.5, P1.0, P2.0, P2.1, P2.2, and P2.3 respectively as shown in the schematic (Figure 1). The next step in setting up the LCD is to connect the potentiometer to adjust the contrast. The left and right pins of the potentiometer were connected to ground and power (which to which doesn't matter) and the middle pin was connected to the 3rd pin of the LCD. The LCD is now set up and ready to interact with the MSP430.

HC06 BLUETOOTH SETUP

To set up the HC06 Bluetooth module we connect the 4 pins on the HC06 to the MSP430. The 4 pins on the HC06 are labeled as VCC, GND, TXD, and RXD. VCC and GND are connected to power and ground respectively. TXD and RXD are connected to P1.2 and P1.1 respectively. The HC06 Bluetooth module is now set up and ready to interact with the MSP430.

SPEAKER SETUP

To set up the 8ohms speaker, we simply connect one wire to ground (typically the black wire) and the other wire (typically the red wire) to an output pin on the MSP430. For this project and design we connected the speaker to pin P1.6 for output.

Android Phone SETUP

To set up the Android Phone to interact with the HC06 Bluetooth, and therefore the MSP430, we simply need to install the provided Android Application (Figure 2). Once the MSP430 has been loaded with the code and is turned on the HC06 will show a blinking red light. Hit the Choose BT Device and select the corresponding HC06 device. The Android Phone should now be ready to interact with the MSP430.

Code and Function:

LCD UTILIZATION

The code for the LCD Display was based on the “LCD Example Demo” provided by Professor Giles. The files lcd.c and lcd.h, taken from Professor Giles’ example, were left largely unchanged except for some small changes for compatibility. We setup the system clocks and outputs similarly to the example provided in the “LCD Example Demo.” P1.1 and P1.2 were selected as outputs for RXD and TXD respectively. Then we called functions from lcd.c, such as LCD_setup(), LCD_init(), to setup and initialize the LCD so it could send instructions to the LCD. The messages displayed on the LCD were changed by using the function LCD_send_string(). The functions sends different messages, which are stored in const char arrays, and calls the different arrays depending on the specific instruction received. The LCD can display only a maximum of 32 characters (16 per line for 2 lines).

SPEAKER UTILIZATION

To play different songs, which would be output to pin P1.6, we wrote 5 functions. Function tone() would take two integers for tone and duration and determine the frequency played and for how long. Function delay_ms() would take an integer and suspend execution for however long based on the integer taken. Function notone() takes an integer and calls delay_ms() to create pauses between notes. Function toneInit() is used to initialize the timers and clocks necessary for the speaker. Finally different songs are made in the functions seq1-5() and stored in FLASH. In the seq() functions the songs are built by calling tone() and notone() in varying orders with varying values to play the desired songs. Notes and songs were based on piano notes and editions. The base we used for our code can be found at this link <https://www.youtube.com/watch?v=gdANH8ikfT4>.

BLUETOOTH AND APP UTILIZATION

To utilize and integrate Bluetooth into the design, the Android Application would send different signals depending on the voice command given. The APP recognizes 10 voice commands, which are “On”, “Off”, “Increase”, “Decrease”, “Flash”, and the numbers 1 through 5. Depending on the voice command, the APP will send an 8bit hexadecimal signal to the MSP430 via HC06 Bluetooth module. For example, the voice command “On” sends 0x41 and the MSP430 gives instructions based on that signal. Another example, the voice command “Decrease” would send the signal 0x3C and the MSP430 would give instructions to display a message “No Blizzard Elsa” followed by calling the seq() to play the song “Do You Want to Build a Snowman” from the movie *Frozen*.

Division of Labor

Krystal Kallarackal: -Developed App -Provided MSP430, LCD, and wires
-Helped solder pins to LCD Display

Jackson Hsu: -Researched project idea, setup, and required components
-Developed and Researched code -Setup board and components
-Provided HC06, Breadboard, wires, speaker, and potentiometer
-Wrote the project report -Drew Schematics

Success, Difficulties, and Future Possibilities

Overall the project was a success. With the goal of developing a sign that displayed messages and produced songs based on voice commands sent by an Android Application via Bluetooth. Our device did just that and could be easily installed onto almost any entrance as a sign. It played five different songs and could easily play more if given the proper coding. It also displayed 7 different messages on the LCD Display with potential for more with the proper coding. The app was able to connect to the HC06 Bluetooth module and send signals based on voice commands with up to 6 functional commands. We achieved what our goal for the project was and therefore deemed it overall successful.

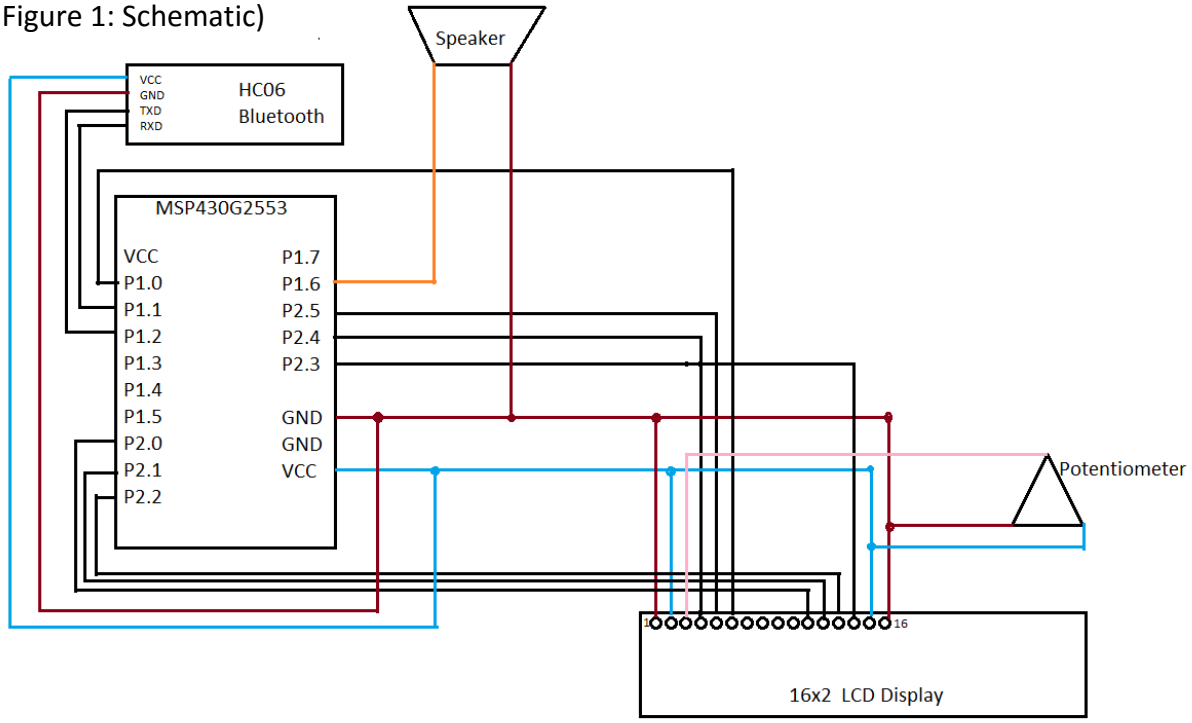
Every project, coding especially, comes with major difficulties. My major difficulty turned out to be something ridiculous. The app was “developed” in a short amount of time and therefore used Googles’ voice recognition to receive voice commands. For whatever reason the app could not recognize the numbers “2”,”4”, or “5”. Instead you would get “to”, “for”, and “v”(???). No matter how much I twisted my pronunciation, and I am a native English speaker with English as my first language, I could not get the app to recognize the numbers. Therefore I

was limited to only 6 voice commands. Another difficulty was translating the musical sheets into notes and durations. I had to relearn how to read sheet music which was difficult for a man who has not played an instrument since middle school. The hardest part was translating the theme song of the series *House M.D* by ear (it didn't have piano sheet music). Normally I would give up and pick a different song, however I was sure that Professor Giles' would appreciate the reference, so I pushed on through straining my ear to pick up the right notes and beats. However, to my devastation, he didn't seem to recognize my poorly translated version. It was a very sad moment for me.

To make this device more successful I would develop a better app utilizing better voice recognition as well as giving it more features. I would develop an option for the user to be able to change the display to whatever they input from the app without having to reload new code to the MSP430. I would also adjust the MSP430 to include an SD card port to store songs as well as install a second speaker. The user would be able to select songs based on the songs loaded into the SD card and play them. No recoding and no pain-staking note translation would be necessary. All you would have to do is change the SD card's contents and the MSP430 would read the contents. Song selection would be a list on the phone that is transmitted from the MSP430 and selected by touch (buttons). Voice commands would only be used for changing display and activating preset instructions. The app would be aesthetically pleasing and user-friendly with options to change display, play song selection, turn the device on/off, and recognize preset voice commands. The device would be aesthetically pleasing, run on rechargeable batteries, have LED light-up options, and be capable of volume adjustment. I would make this device something that might actually be marketable.

Schematic and Picture

(Figure 1: Schematic)



(Figure 2 : Screenshot of Android App)

(Figure 3 : Picture of Components Set up)