Robotic Band with a “Drummer” and a “Trumpeter”

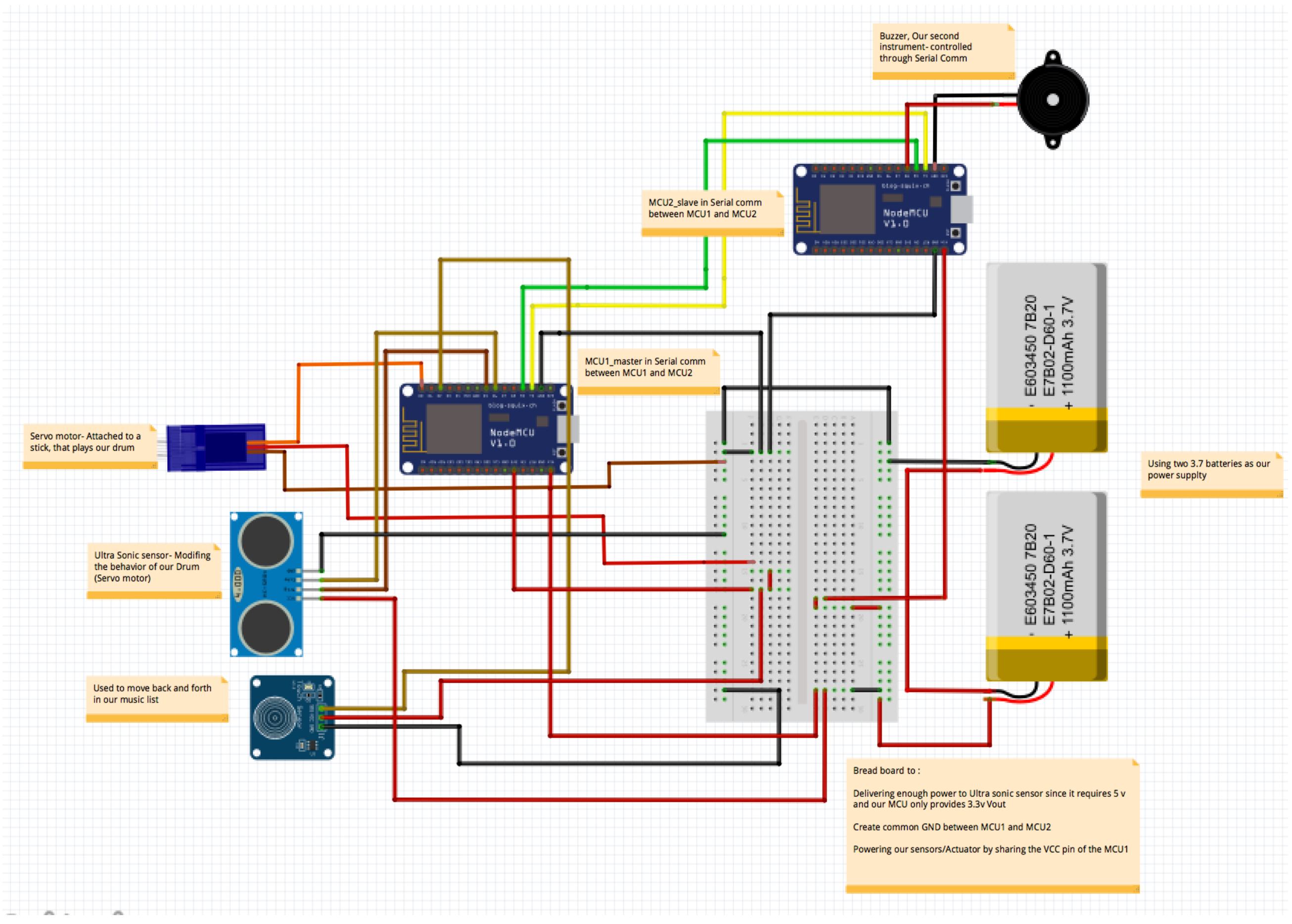
Lin Li
Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095

Project Description and Personal Contribution

In this project, we tried to build a robotic band that can play a couple of tunes with several different tempos. To reach our goal for this project, we decided to introduce a servo and a buzzer with programmed microcontrollers and sensors into our robotic band as the “drummer” and the “trumpeter”.

My personal contribution to this project is to compose the tunes in programming language and to test the sensor functionalities.

The Schematic of the Entire System


Buzzer, Our second instrument- controlled through Serial Comm
MCU2_slave in Serial comm between MCU1 and MCU2
NodeMCU V1.0
MCU1_master in Serial comm between MCU1 and MCU2
NodeMCU V1.0
Servo motor- Attached to a stick, that plays our drum
Ultra Sonic sensor- Modifing the behavior of our Drum (Servo motor)
Used to move back and forth in our music list
E603450 7B20 E7B02-D60-1 + 1100mAh 3.7V
E603450 7B20 E7B02-D60-1 + 1100mAh 3.7V
Using two 3.7 batteries as our power supplty
Bread board to :
Delivering enough power to Ultra sonic sensor since it requires 5 v and our MCU only provides 3.3v Vout
Create common GND between MCU1 and MCU2
Powering our sensors/Actuator by sharing the VCC pin of the MCU1

As we can see in this schematic, we used two programmed microcontrollers to control the servo and the buzzer while each of the microcontroller can communicate with each other. We also connect a ultrasonic sensor and a touch sensor with the servo side microcontroller to let the robotic band be able to have interaction with the human user and the environment.

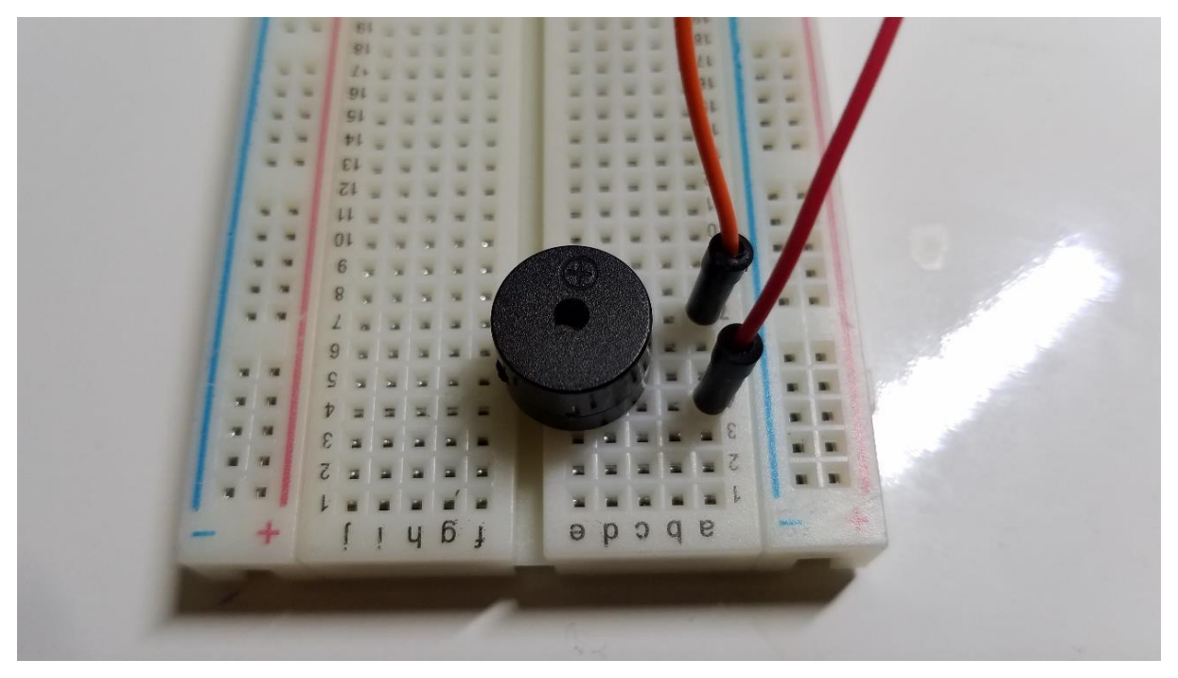


Figure 1. The “trumpeter”

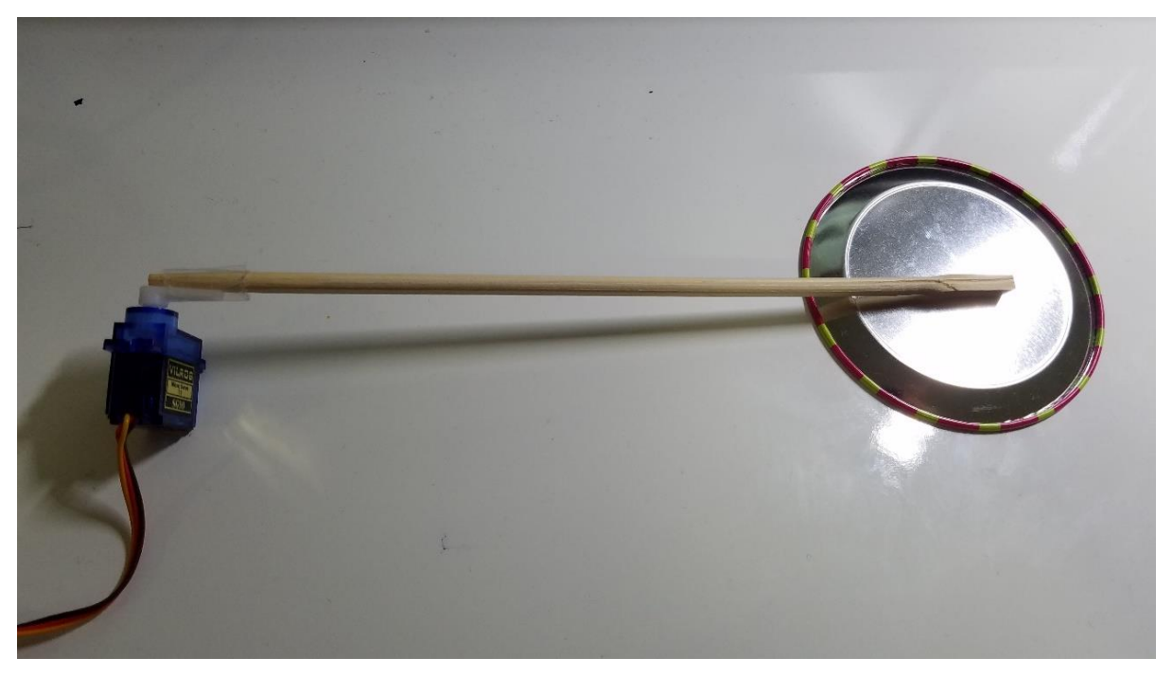
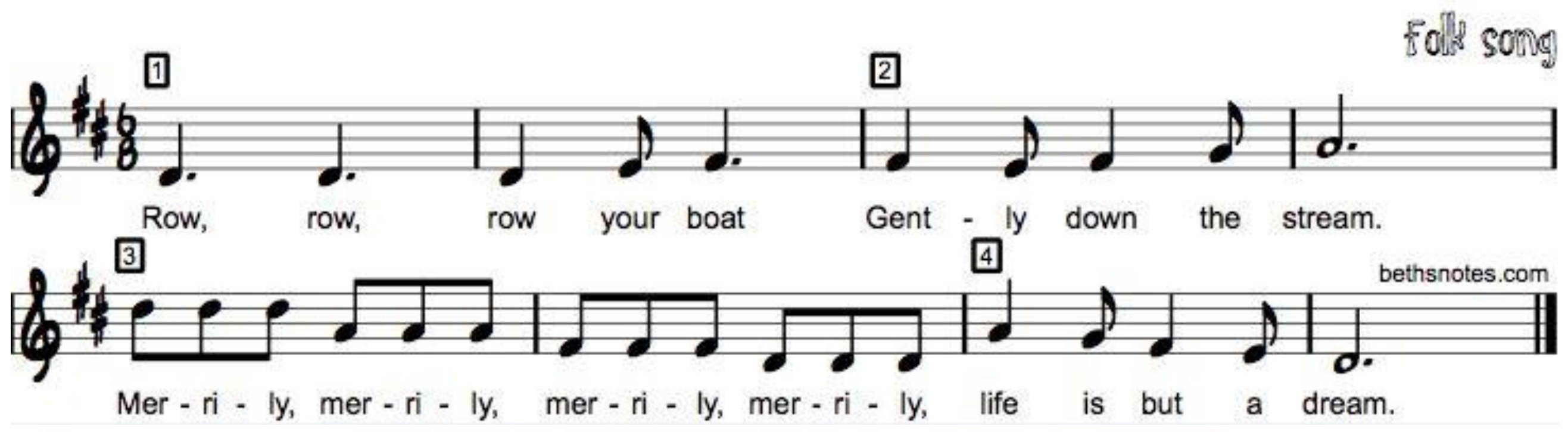
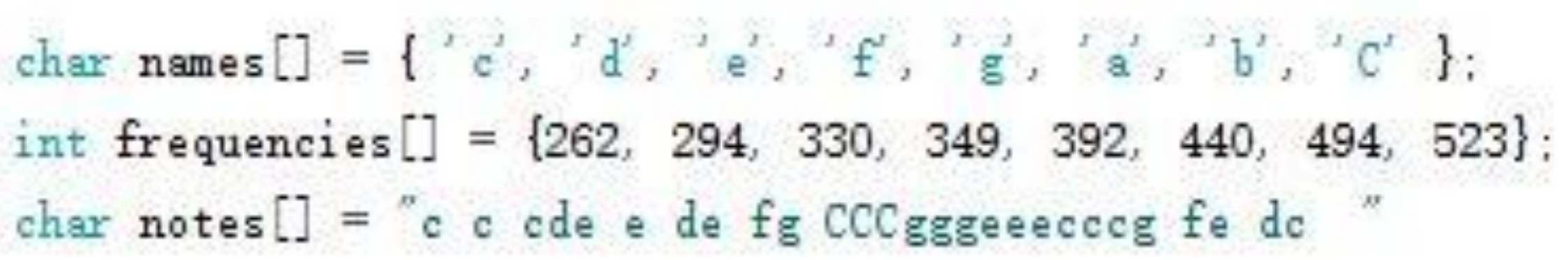


Figure 2. The “drummer”

Composing Tunes for the “Trumpeter”


Folk song
1
Row, row, row your boat
2
Gent - ly down the stream.
3
Mer - ri - ly, mer - ri - ly, mer - ri - ly, mer - ri - ly,
4
life is but a dream.
bethsnotes.com

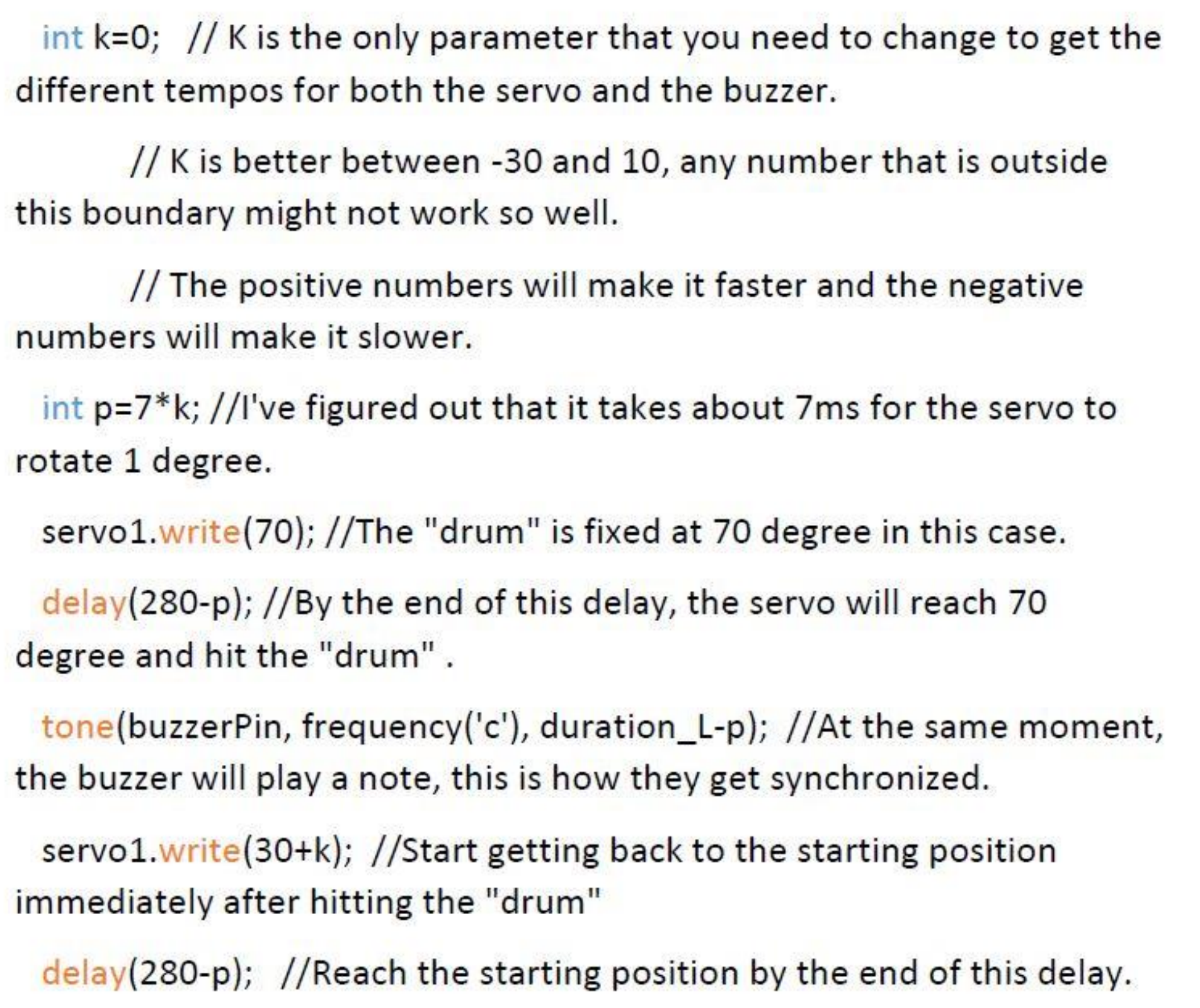


```
char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
int frequencies[] = {262, 294, 330, 349, 392, 440, 494, 523};
char notes[] = "c c cde e de fg CCCgggeeecccg fe dc  ";
```

Above is the original note chart of one of our tunes and a part of the code that shows how our “trumpeter”(Figure1) plays the tune.

I first defined the music notes with corresponding characters and frequencies. Then I composed the song with the characters according to the original note chart.

Composing Beats for the “Drummer”



```
  int k=0;   // K is the only parameter that you need to change to get the
different tempos for both the servo and the buzzer.
          // K is better between -30 and 10, any number that is outside
this boundary might not work so well.
          // The positive numbers will make it faster and the negative
numbers will make it slower.
  int p=7*k; //I've figured out that it takes about 7ms for the servo to
rotate 1 degree.
  servo1.write(70); //The "drum" is fixed at 70 degree in this case.
  delay(280-p); //By the end of this delay, the servo will reach 70
degree and hit the "drum" .
  tone(buzzerPin, frequency('c'), duration_L-p);  //At the same moment,
the buzzer will play a note, this is how they get synchronized.
  servo1.write(30+k);  //Start getting back to the starting position
immediately after hitting the "drum"
  delay(280-p);   //Reach the starting position by the end of this delay.
```

Above is another piece of the code that I wrote to implement the “drummer”(Figure2) behavior on a servo. As we can see in the figure, I attached an end of a chopstick to the servo leg and placed a metal plate by the other end of the chopstick.

In the code, I defined the ending angle as 70 degree. In other words, whenever the servo rotates to 70 degree, the chopstick will hit the metal plate to make a sound.

The delay time in the code is controlling the gap between two beats and I used a single variable “k” to control the overall tempo of the beats.

Testing the Ultrasonic Sensor

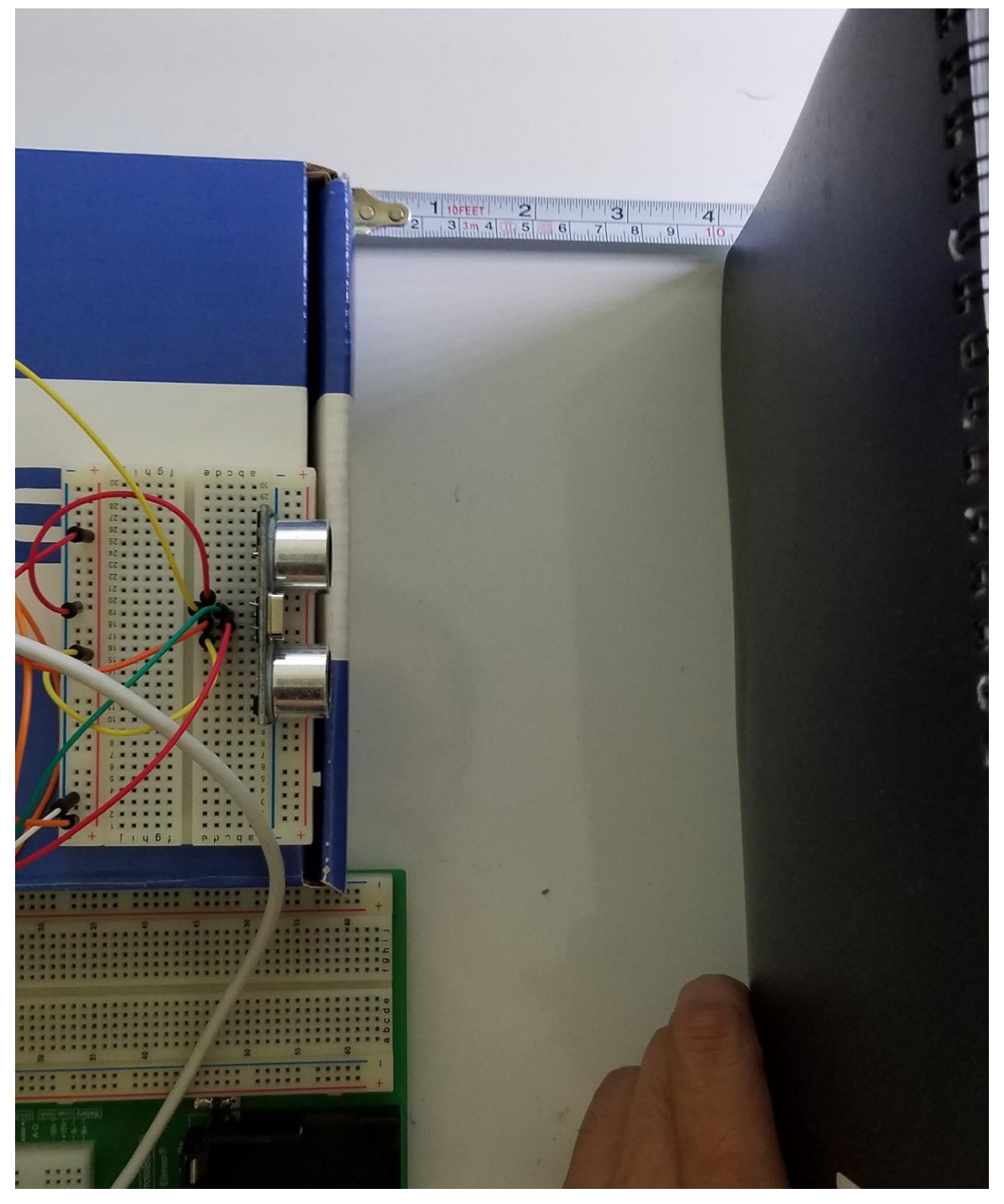


Figure 3. Test1 set up for 10cm

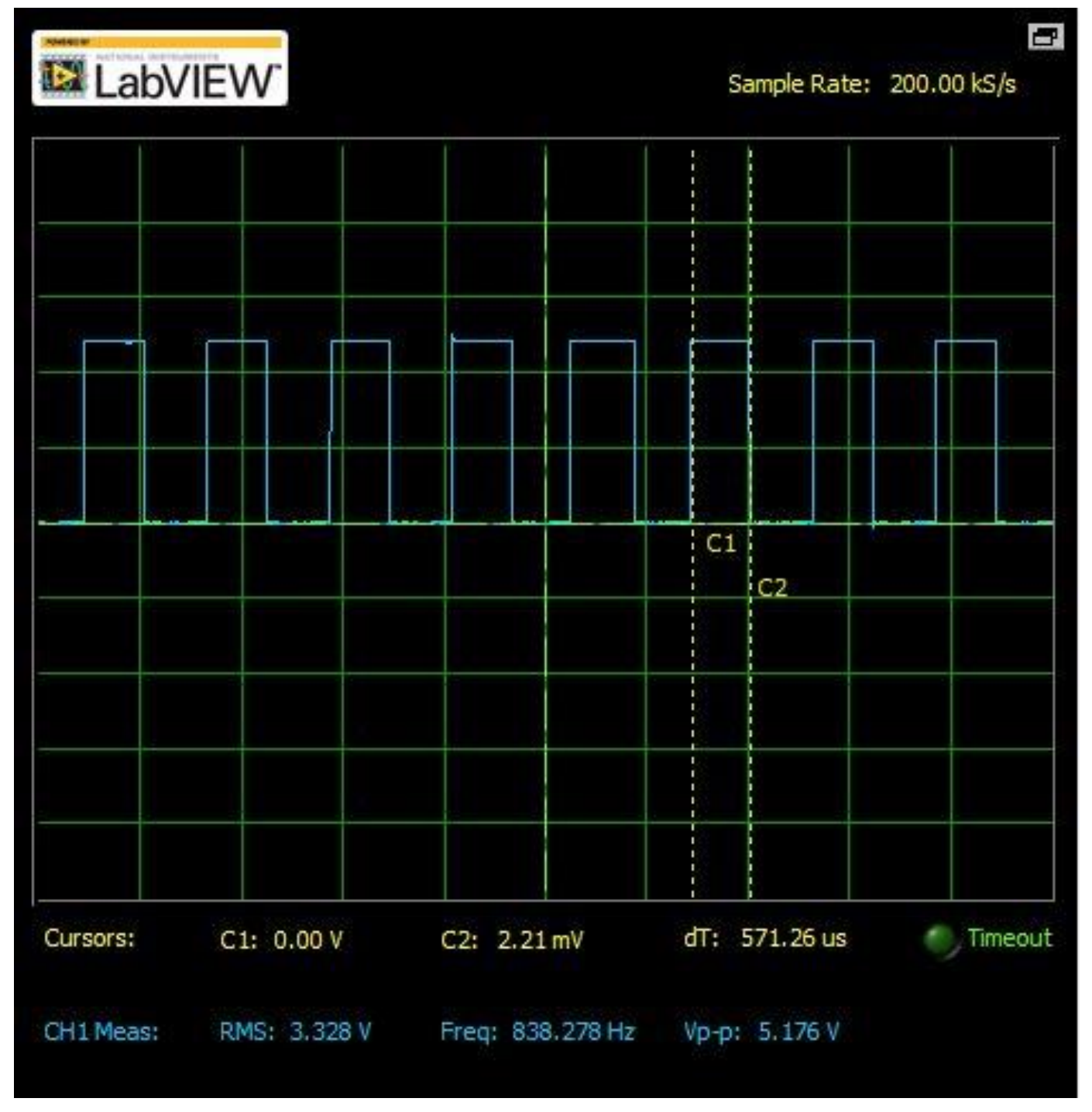


Figure 4. Test1 oscilloscope result

In our robotic band, we used the ultrasonic sensor to sense the environment and interact with the human users.

The band will change its tune tempo according to the distance that is detected by the ultrasonic sensor.

The figures above are showing the set up and the result from one of my ultrasonic sensor tests.

The table below is showing the data that is collected from the tests.

Table 1. Ultrasonic sensor testing data

	Sound Wave Travelling Time	Computed Distance	Actual Distance
Test1	571µs	9.7cm	10cm
Test2	1120µs	19.0cm	20cm
Test3	1740µs	29.6cm	30cm

Conclusions

The most exciting moment that I have experienced in this project was when I successfully made the drummer and the trumpeter play in the perfect harmony.

I have also figured out the coding strategy of composing different robotic instruments with the same tempo.

Based on the experience I have gained from this project, I’m confident to add more robotic members into our band to play more complicated tunes in the future.

Contact

Name: Lin Li
School: UCLA
Email: lilin8546@gmail.com
https://github.com/ckwojai/EE183_JamBand
Phone: 213-249-5983