

University of British Columbia


Electrical and Computer Engineering

ELEC 291 Winter 2022

Instructor: Dr. Jesus Calvino-Fraga

Section 201

Project 1 - Capacitive Sensors Reaction Game

Student #	Student name	%Points	Signature
96925458	Der Chien Chang	100	
34532507	Will Chen	100	
81928293	Connor McJunkin	100	

Date of Submission: March 4th, 2022

Table of contents

1. Introduction

1.1 Project requirements

2. Investigation

2.1. Idea Generation

2.2. Design Investigation

2.3. Data Collection

2.4. Data Synthesis

2.5. Analysis of Results

3. Design

3.1. Use of Process

3.2. Need and Constraint Identification

3.3. Problem Specification

3.4. Solution Generation

3.5. Solution Evaluation

3.6. Detailed design

3.7. Solution assessment

4. Live-Long Learning

5. Conclusions

1. Introduction

The goal of this project is to design, build, program and test a microcontroller based capacitive sensors reaction game. The two-player game will produce an audible noise using a speaker, and then award or remove points from players based on their response with the sensors. Points will be tracked for each player using an LCD.

Project Requirements

- i. 8051 based Microcontroller System: For this project we will use the AT89LP51RC2 microcontroller.
- ii. Sensor Capacitors: Build at least two sensor capacitors.
- iii. A-stable Oscillators or similar: These oscillators will change their frequency with the capacitance variations of the sensor capacitors.
- iv. Speaker and LCD: The game must use both the CEM-1028 mini speaker (or similar) and the LCD.
- v. Assembly programming: All programming for this project must be completed in assembly language.
- vi. Sensitivity: The capacitive sensors should be able to reliably and quickly detect a hand on top of them.

vii. Suggested Game Rules

- a. The game will produce either a 4196Hz tone or a 3996Hz tone randomly using the CEM-1028 speaker.
- b. If the tone is 3996Hz, the first player to press its capacitor sensor wins a point.
- c. If the tone is 4196Hz, the first player that presses its capacitor sensor loses a point.
- d. The first player to reach 5 points wins the game

2. Investigation

Idea Generation

To generate ideas, our team first identified the project requirements. In our case, we had to make a game using capacitive sensors, decide on game rules, and decide what additional features we wanted to add. After deciding to keep the game rules fairly similar to the suggested, our team began to generate ideas for how to build the capacitive sensors, as well as possible extra features. After each presenting multiple ideas to the group, we selected our favourite options and began to create some prototypes of our sensor to see what material was the most responsive for our capacitive sensor, as well as testing extra features.

Design Investigation

Our main design investigation revolved around doing tests for the capacitive sensor. After building our circuit board with two breadboards, following the general design given by Dr.

Calvino Fraga, we began making our prototypes. Our sensors were made with aluminum foil, which we then covered with different materials and ran tests to find which material gave us an optimal response.

Data Collection

Our team collected data while we were testing our design. Our main focus was finding out the period, and therefore the frequency of the waveform created by our various capacitive sensors. To do this, we run the sample code on our AT89LP51RC2 microcontroller, which measure the period of the connected capacitor. For each sensor design, we measured the period with and without touching the sensor. After collecting the results, we then compared them to find which would be optimal for our design.

Data Synthesis

After acquiring the data using the methods above, our team compared the results to those that we wanted and expected. Our main focus here was ensuring that our sensors produced consistent results so that we could perform a proper analysis.

Analysis of Result

Our team appraised the validity of our conclusions through testing and research. After collecting and synthesizing our data, our team was able to identify the best sensor design moving forward. We needed a design that would produce identifiably different frequency signals when it

was being touched and when it was not being touched. After testing our sensor with various materials, we were able to identify the optimal material to continue with for the final design.

3. Design

Use of Process

When beginning our design, our group referred to in-class lectures, PowerPoint slides, the lab manual as well as additional online research. After identifying the requirements of the project, which included using the AT89LP51RC2 microcontroller, and then building a game with capacitive sensors that awards points to players based on their response to a noise produced by a CEM-1028 mini speaker. The points then had to be displayed on an LCD, and we were also able to add additional features if we wished to do so. With this in mind, our team began to generate ideas and discuss them as a group. After careful deliberation, we chose a couple of the designs that we thought would work best based on practicality and efficiency.

Need and Constraint Identification

Before identifying stakeholder needs and constraints, we first needed to identify who the stakeholders in our project were. We decided that the targeted stakeholders for this project would be anyone who will use or evaluate our capacitive sensor game, which includes students, professors and friends. With this in mind, we decided to optimize the game for the people that will play it. In order to create an optimal user experience, we needed to focus on game functionality as well as having a nice presentation.

The constraint for this project is that we must program the game in 8051 assembly language on CrossIDE to load the project onto the AT89LP52 microcontroller.

Problem Specification

With a focus on game functionality and user experience, we were able to identify what needed to be done to achieve these things. For optimal game functionality, we needed to create sensors that give clear and consistent readings to our astable oscillator, and that also had enough of a difference in frequency to be able to clearly identify when the sensor is being pressed. This will ensure that the game awards points in a consistent fashion. We also wanted to create an optimal user experience, which involves presentation and ease of use. We needed to create a clean look for our breadboard and LCD, and we decided that we also wanted to add some sort of game start noise that will play whenever the game begins and the celebration noise that will play when someone wins the game.

Solution Generation

We decided to make our capacitive sensor with aluminum foil, which we then covered with additional materials to ensure safety. We initially attempted to cover the aluminum foil with plastic folder but found that the plastic folder was too thick, which made it difficult to tell if the sensor was being pressed. We then tried covering the aluminum foil with saran wrap, and this produced the results we were hoping for. For creating an optimal user experience, we made sure our breadboard wiring was clean and well put together, we created push buttons to start and restart the game, and designed the LCD interface in a way we thought was best. We also decided

to add a celebration song that will play when someone wins the game, along with a congratulations message to run across the LCD.

Solution Evaluation

We decided to use saran wrap for our final design because it provided the best frequency readings for the sake of our game. We needed a design that would make it clear when the sensor is being touched, which means that we needed the frequency to change by a consistently noticeable amount so that we could program our code to take these readings and decide if the sensor has been touched. Using saran wrap, the frequency of the signal produced by the capacitive sensor consistently changed by at least 10Hz when touched and was therefore an acceptable design for our project. We also decided to add the song “Twinkle Twinkle Little Star” to play as a celebration when someone wins the game, as we felt this was a tune most people would recognize and appreciate.

Detailed Designs

As for our design, we built on the two tone speaker file that was given on piazza. We use a few major codes to accomplish this project. There are CheckA/B function, the A/B plate, and A/B counter.

The Check function determines whether a plate is pressed or not. First of all, we set the threshold value to 4775 and 4845 for A plate and B plate respectively. 4775 is the A plate’s period when A plate is pressed and 4845 is the B plate’s period when B plate is pressed. We compare the period produced by the plate to a threshold value to check if the plate is pressed or not. In order to prevent both A player and B player add point at the same time, we make the code to check A

first before checking B. We set $mf = 1$ if A is pressed; therefore, B is only checked if $mf = 0$ and set $mf = 1$ after B is pressed. In this case, only one of the player is adding point in a single frequency. If either of A or B is pressed, mf will be set to 1 and it would jump to its corresponding counter, which would determine to add or subtract value based on which frequency that the system is currently displaying. In order to check the plate is pressed at which frequency, we set the low frequency as ($toneflag = 1$) and high frequency as ($toneflag = 0$)

The plate function on the other hand, is what we use to measure the frequency of the plate. The function uses the timer 2 to determine what frequency that the plate capacitor is on and changes the value if the plates are pressed.

<pre> A_plate: ; Measure the period applied to pin P2.0 clr TR2 ; Stop counter 2 mov TL2, #0 mov TH2, #0 jb P2.0, \$ jnb P2.0, \$ setb TR2 ; Start counter 0 jb P2.0, \$ jnb P2.0, \$ clr TR2 ; Stop counter 2, TH2-TL2 has the period ; save the period of P2.0 for later use mov Period_A+0, TL2 mov Period_A+1, TH2 ljmp check_A </pre>	<pre> B_plate: ; Measure the period applied to pin P2.1 clr mf clr TR2 ; Stop counter 2 mov TL2, #0 mov TH2, #0 jb P2.1, \$ jnb P2.1, \$ setb TR2 ; Start counter 0 jb P2.1, \$ jnb P2.1, \$ clr TR2 ; Stop counter 2, TH2-TL2 has the period ; save the period of P2.1 for later use mov Period_B+0, TL2 mov Period_B+1, TH2 ljmp check_B </pre>
--	---

And lastly the counter is determining the winner. We use a similar idea to the timer in lab 2 to determine the winner when one of the player's scores reaches 5.

The entire process is as such:

1. Produce a random value.

SetupRandom:

```

    setb TR0
    jnb RandomSeed, $
    mov Seed+0, TH0
    mov Seed+1, #0x01
    mov Seed+2, #0x87
    mov Seed+3, TL0
    clr TR0

```

2. Display frequency 1 when the random value is higher than the threshold value, vice versa. (this part is from the random generator code on Piazza).

Pick_frequency:

```

    clr mf
    lcall Random

    mov x+0, Seed+0
    mov x+1, Seed+1
    mov x+2, Seed+2
    mov x+3, Seed+3
    Load_y(2147483648)

    lcall x_lt_y

    jb mf, Frequency2 ;mf = 1, goes to frequency 2
    ljmp Frequency1

```

3. Plate checking if any of the plate is pressed, if so moves to counter function, if not, move on to display score and displaying the same score.

Check_A:

```

    mov x+0, Period_A+0
    mov x+1, Period_A+1
    mov x+2, #0
    mov x+3, #0

    load_y(4775)
    lcall x_gt_y

    jb mf, A_counter ;mf = 1, A is pressed
    jnb mf, display_pointA_sc

```

Check_B:

```

    mov x+0, Period_B+0
    mov x+1, Period_B+1
    mov x+2, #0
    mov x+3, #0

    load_y(4845)
    lcall x_gt_y

    jb mf, B_counter ;mf = 1, A is pressed
    jnb mf, display_pointB_sc

```

4. The counter determines whether the player gains and loses points after they pressed the plate. This is determined by which frequency was produced by the speaker, which depends on the toneflag value.
5. Display the score/ the winner if there are any.

```

display_pointA:
    Set_Cursor(1, 14)    ; the place in the LCD where we want the BCD counter value
    Display_BCD(PointsA)
    jnb mf, B_plate      ; if A is not pressed then check B
    ljmp Pick_frequency  ; if A is pressed goes to the next beep

display_pointB:
    Set_Cursor(2,14)
    Display_BCD(PointsB)
    ljmp Pick_frequency

```

Aside from the main function of the game, we added a few more features. One is to display music before and after the game. And the other is to display a congratulation text after we have a winner. For the music, we simply sent out the appropriate frequency after a given amount of time, high for the ending music we combine it with the text displayed. Every change of text would also mean a sound out of the speaker. To play the “Twinkle Twinkle Little Star”, we try out different frequency of the speaker to find out the corresponding music notes. And eventually, we found 7 different frequency that corresponds to 7 different music notes.

```

MUSIC1 EQU ((65536-(CLK/TIMER0_RATE2)))
MUSIC2 EQU ((65536-(CLK/TIMER0_RATE3)))
TIMER0_RATE4 EQU ((2048*2)-2300)
TIMER0_RATE5 EQU ((2048*2)-2200)
MUSIC3 EQU ((65536-(CLK/TIMER0_RATE4)))
MUSIC4 EQU ((65536-(CLK/TIMER0_RATE5)))
TIMER0_RATE6 EQU ((2048*2)-2050)
TIMER0_RATE7 EQU ((2048*2)-1750)
MUSIC5 EQU ((65536-(CLK/TIMER0_RATE6)))
MUSIC6 EQU ((65536-(CLK/TIMER0_RATE7)))
TIMER0_RATE8 EQU ((2048*2)-1400)
MUSIC7 EQU ((65536-(CLK/TIMER0_RATE8)))

```

Solution Assessment

The functionality of our game relies on the capacitive sensors working as intended. This means that we want the sensors to give consistent frequency readings relative to user input, as this will ensure that points are awarded properly. After testing different designs, we found that this setup gave the most consistent and accurate readings, and awarded points properly about 90% of the time. With this design, sensor A produced a period of about 4775 when being touched and 4650 when not, and sensor B produced a period of 4845 when touched and 4700 when not. These differences were large enough for us to input a threshold value into our code that signifies whether the sensors are being pressed. Other designs did not perform as well, and often would not produce much of a change in period when pressed.

The rest of our design functioned as intended. The music that plays before and after the game, as well as the LCD all function properly and meets the needs of the stakeholders.

4. Life-Long Learning

Throughout this project, our team relied on various skills learnt from prerequisite courses. We found CPEN 211 to be particularly useful in designing our circuit and writing our code.

When attempting to implement our extra functionality of playing “Twinkle Twinkle Little Star” upon the conclusion of the game, we discovered a knowledge gap. In order to play the song, we needed to know what frequencies correspond to each note of the song. After researching online, we discovered the seven different frequencies required to play the song and were able to implement them into our code.

This project allowed us to gain valuable experience designing and testing our own systems, which will prove useful as we continue to work as students of electrical engineering.

5. Conclusions

Our game is designed to award points to players based on their response to the sound produced by the game. Players are meant to press their respective capacitive sensor, when they hear a low pitched noise, and leave it alone when they hear a low pitched noise. Our game also plays music upon initializing the game, and celebratory music with a congratulations message displayed on the LCD when someone wins the game. Our design provides the most consistent and enjoyable user experience we found possible, although we did occasionally have trouble getting the sensors to always detect when they were pressed. Overall, we spent about 20 hours designing and building our game.

