UCR

Electrical and Computer Engineering

EE 128 Mini Project

Sound Localization Platform

Team 5: Junda Jiang & Shuo Li

June 10, 2020

Abstract

For this project, our purpose is to build the Sound Localization Platform, which can rotate the platform to where sound occurs. This detection can distinguish between hand claps and surrounding background noise, also coordinate with multiple microphones.

Block Diagram

For this project, the majority of work we will implement in K64F. The 8 sound sensors will be used as input signals to K64F, then the K64F will let motor rotate to where sound occurs, also the different position of sensor represents the different rotated angle. The Block Diagram is shown as Figure 1.

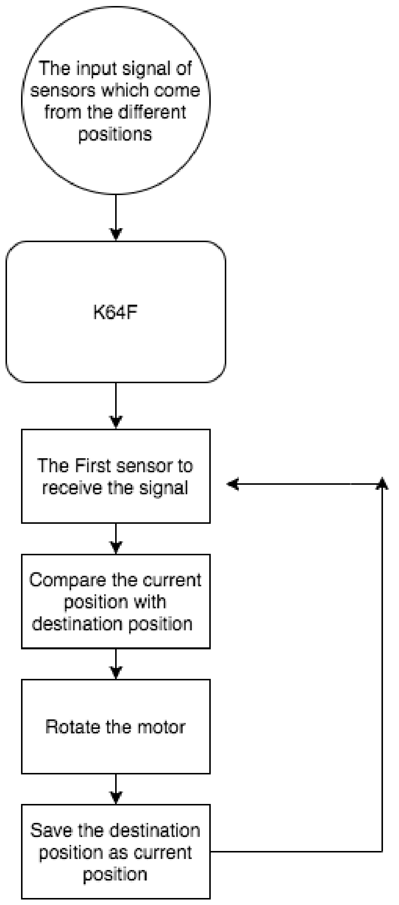
The input signal of sensors which come from the different positions
K64F
The First sensor to receive the signal
Compare the current position with destination position
Rotate the motor
Save the destination position as current position

Figure 1. Block Diagram

Circuit Schematic

The Figure 2 shows the Circuit Schematic for our project. As you can see in this schematic, we open PortC 7-10 gates and PortD 0-3 gates for 8 sound sensors, and the rest of it is similar as what we have done for Lab5. Figure3 shows the circuit we built on board with 8 sound sensors, and we used yellow stickers as directions, and you can understand the current direction as 0 degrees or 180 degrees.

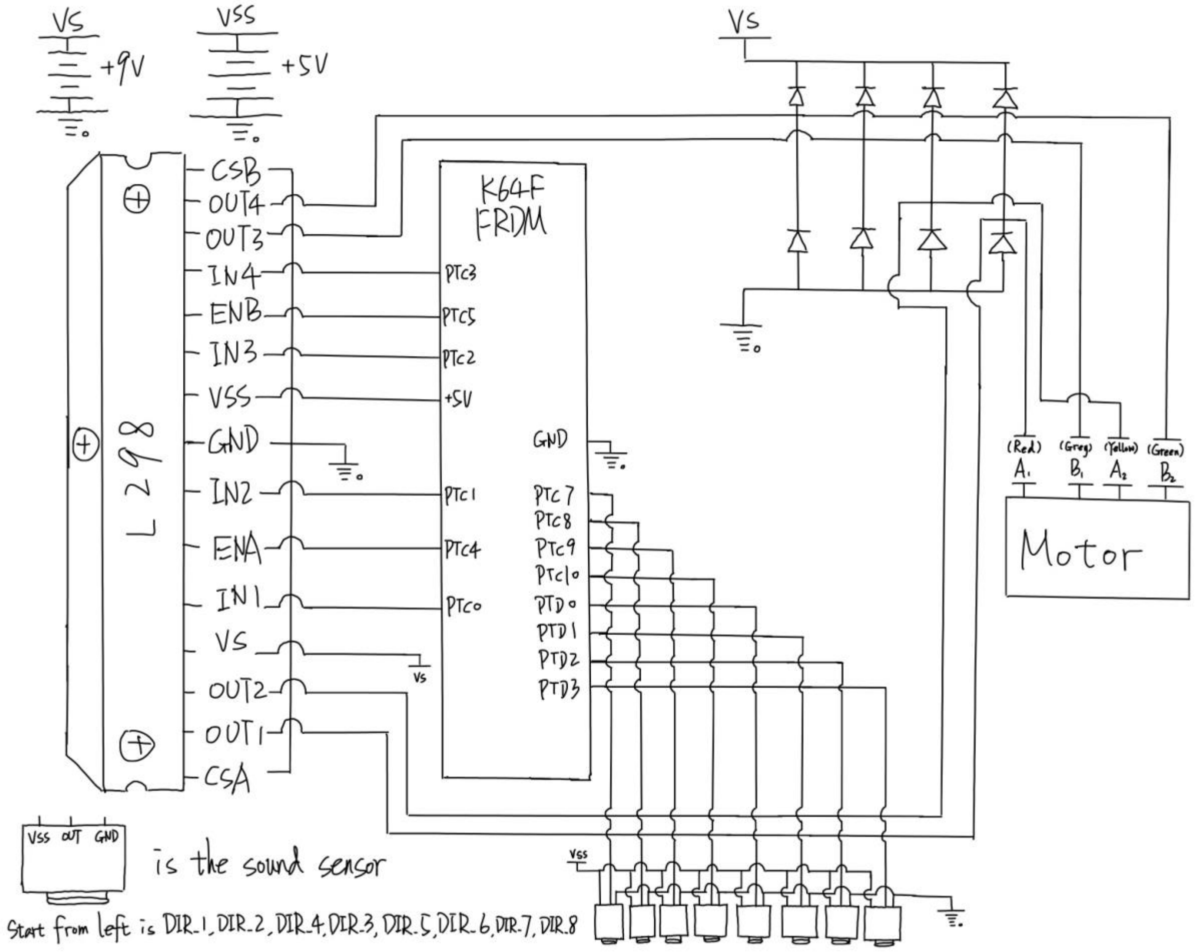

Figure 2. Circuit Schematic

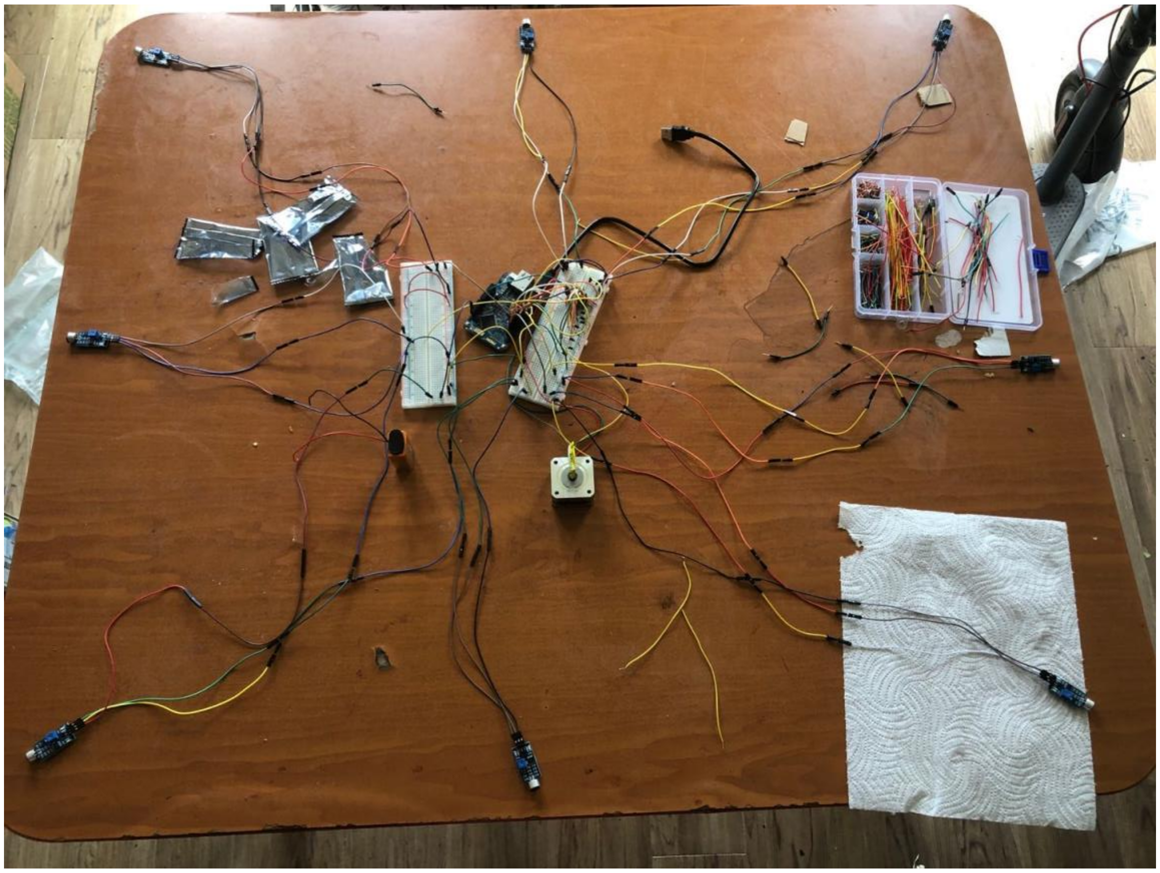
Figure 3. Circuit

Testing/Evaluation

It doesn’t require a lot for test environment, but we have to check how sensitive the sensor is and how good for the battery, because it will not detect any sound if you reduce the sensitivity or your battery is low. Also, a quiet place will be better for our project. We will hand clap to test each sound sensor to make sure it will rotate to where the sound occurs.

Discussion

We need to check how sensitive the sensor is and how good for the battery, also a quiet place will help us test it better. Most of question are still on equipment, such as the motor will always rotate without stopping or the sensor cannot detect the hand clap so that the motor will not rotate to where the sound occurs. If we still have time to improve, we will make it more sensitive and accurate, especially in do more search about equipment.

References

We don’t use any reference except the lecture power points from this course and the content of Lab5 for this course.

Roles and Responsibilities of Group Members

Contribution: 50% for each member (including setting environment and work on real device, and writing report).

Conclusion

From this project, we have learnt more about how to use K64F to control the motor depend on different input signals, and sometimes it would not work well as we think, because we need to consider how sensitive the sensor is and how good for the battery, then we need to test in a quiet place to make sure it won't have a big background noise. All in all, the project works successfully and we enjoyed a lot. I will list all components where we buy, and share the link to show you our demo videos in YouTube.

DAOKI 5PCS High Sensitivity Sound Microphone Sensor Detection:
https://www.amazon.com/dp/B00XT0PH10/ref=cm_sw_r_sms_c_api_i_XgiWEbW4RW298

9 V Battery Clip Connector Long Cable Connection:
https://www.amazon.com/dp/B0779ZSNS3/ref=sspa_dk_detail_0?psc=1&pd_rd_i=B0779ZSNS3&pd_rd_w=7cnm2&pf_rd_p=48d372c1-f7e1-4b8b-9d02-4bd86f5158c5&pd_rd_wg=7zHbh&pf_rd_r=JRA0HA9V5PFXJZWQSTMF&pd_rd_r=6f187ca9-68af-43f8-acba-d5aa4aaaad3c&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUFDMEFEQjZTSVVWV0kmZW5jcnlwdGVkSWQ9QTAxMTU3NjExR0RHNDNMNTZXWDNEJmVuY3J5cHRlZEFkSWQ9QTAwODQ1NTIzU0hXSkRMOVMzU0JJJndpZGdldE5hbWU9c3BfZGV0YWlsJmFjdGlvbj1jbGljay1JlZGlyZWN0JmRvTm90TG9nQ2xpY2s9dHJ1ZQ==

The uploaded video in YouTube:

The first demo:
https://www.youtube.com/watch?v=5i8fGfyQY_0

The second demo:
https://www.youtube.com/watch?v=lnquqX_a4K0

Appendix

Since we don't use any reference online except the lecture power points from this course and the content of Lab5 for this course, we will upload the whole file.

```
#include "fsl_device_registers.h"

#include "MK64F12.h"

#include <stdio.h>

#include <stdlib.h>


void software_delay(unsigned long delay) //Setting a delay function so that we can change the speed as desired

{
```

```c
        while (delay > 0) delay--;
}

//Define the clockwise function so that the pointer can rotate clockwise
void clockwise(void){
        unsigned int Delay = 20000;
        for(int i = 0; i<=5; i++){
                GPIOC_PDOR = 0x3A;
                software_delay(Delay);
                GPIOC_PDOR = 0x39;
                software_delay(Delay);
                GPIOC_PDOR = 0x35;
                software_delay(Delay);
                GPIOC_PDOR = 0x36;
                software_delay(Delay);
        }
}

//Define the counterclockwise function so that the pointer can rotate counterclockwise
void counterclockwise(void){
        unsigned int Delay = 20000;
        for(int i = 0; i<=5; i++){
                GPIOC_PDOR = 0x36;
                software_delay(Delay);
                GPIOC_PDOR = 0x35;
                software_delay(Delay);
                GPIOC_PDOR = 0x39;
                software_delay(Delay);
                GPIOC_PDOR = 0x3A;
                software_delay(Delay);
        }
}

int main(void)
{
        SIM_SCGC5 |= SIM_SCGC5_PORTD_MASK;  //Enable Port D Clock Gate Control
        SIM_SCGC5 |= SIM_SCGC5_PORTC_MASK;  //Enable Port C Clock Gate Control

        PORTC_GPCLR = 0x07BF0100; //Setting the pins 0-5, 7-10 of the Port C as GPIO
        PORTD_GPCLR = 0X000F0100; //Setting the pins 0-3 of the Port D as GPIO
```

```
	GPIOC_PDDR = 0x0000003F;  //Setting the pins 0-5 of the Port C as Output and pins 7-10 of the Port C as Input
	GPIOD_PDDR = 0x00000000;  //Setting the pins 0-3 of the Port D as Input

	unsigned long Delay; //Declare the variable Delay
	unsigned long DIR_1=0, DIR_2=0, DIR_3=0, DIR_4=0, DIR_5=0, DIR_6=0, DIR_7=0, DIR_8=0; //Declare the
variable DIR_1 - DIR_8 for sensor detections
	uint32_t input_val_c = 0, input_val_d = 0; //Declare the variable input_val_c and input_val_d
	int origin = 0, sensor1 = 45, district1= 90, sensor2 = 135, district2 = 180, sensor3 = 225, district3 = 270, sensor4 =
315;  //Declare the variables for different angles in the map
	int current_angle = 0; // Declare the variable that indicates the current angle of the pointer.
	int count = 0; //Declare the variable count so that the pointer rorates to the correct position
	int destination = -1; //Declare and assign -1 to the variable destination as 0 is one of our angles so we have to
assign initial value other than 0

	while (1) {
		input_val_c = GPIOC_PDIR & 0x780;  //Read and Store pins 7-10 of the Port C to input_val_c
		input_val_d = GPIOD_PDIR & 0x0F;  //Read and Store pins 0-3 of the Port D to input_val_d
		DIR_1 = (input_val_c & 0x80);  //Store the pin 7 of the Port C
		DIR_2 = (input_val_c & 0x100);  //Store the pin 8 of the Port C
		DIR_3 = (input_val_c & 0x200);  //Store the pin 9 of the Port C
		DIR_4 = (input_val_c & 0x400);  //Store the pin 10 of the Port C
		DIR_5 = (input_val_d & 0x1);  //Store the pin 0 of the Port D
		DIR_6 = (input_val_d & 0x2);  //Store the pin 1 of the Port D
		DIR_7 = (input_val_d & 0x4);  //Store the pin 2 of the Port D
		DIR_8 = (input_val_d & 0x8);  //Store the pin 3 of the Port D

		destination = -1;  //Assign -1 to destination everytime enter the loop so that the pointer can stop without
hearing sounds
		count = 0;  //Assign 0 to count so that the pointer can rotate to the correct position every time


    //If the sound sensors hears sounds, indicates the angle of destination, we use "else-if" instead of "if" for DIR_2 to
DIR_3 so that the destination will be the location where the first sound the sensor received (As the sensors are sensitive and
some other sensors located at nearby position may hear the sound when we make sounds to the target position
		if(DIR_1 != 0){
			destination = sensor1;
		}
		else if(DIR_2 != 0){
			destination = sensor2;
		}
```

```
        else if(DIR_3 != 0){
            destination = sensor4;
        }
        else if(DIR_4 != 0){
            destination = sensor3;
        }
        else if(DIR_5 != 0){
            destination = origin;
        }
        else if(DIR_6 != 0){
            destination = district1;
        }
        else if(DIR_7 != 0){
            destination = district2;
        }
        else if(DIR_8 != 0){
            destination = district3;
        }


    //Make sure the pointer works as desired. The pointer will turn to the target destination based on smallest angel to the
destination (e.g. Assume the pointer's current position is 45 degree, when we make sounds at 270 degree, the pointer will
turn to 270 degree counterclockwise instead of clockwise
        if((destination != current_angle) && destination != (-1)){
            temp = temp + 1;
            if(current_angle <180){
                if(abs((destination - current_angle) > 180)){
                    count = ((abs(destination - current_angle))/45);
                    for(int i=0; i<(8-count); i++){
                        counterclockwise();
                    }
                    current_angle = destination;
                }else if((abs(destination - current_angle) <= 180)){
                    if((destination - current_angle) > 0){
                        count = ((abs(destination - current_angle))/45);
                        for(int i=0; i<count; i++){
                            clockwise();
                        }
                        current_angle = destination;
                    }
                    else if((destination - current_angle) < 0){
                        count = ((abs(destination - current_angle))/45);
                        for(int i=0; i<count; i++){
                            counterclockwise();
                        }
                        current_angle = destination;
                    }
                }
            }else if(current_angle >=180){
                if((abs(destination - current_angle) <= 180)){
                    if((destination - current_angle) > 0){
                        count = ((abs(destination - current_angle))/45);
                        for(int i=0; i<count; i++){
                            clockwise();
                        }
                        current_angle = destination;
                    }else if(destination - current_angle < 0){
                        count = ((abs(destination - current_angle))/45);
                        for(int i=0; i<count; i++){
                            counterclockwise();
                        }
                        current_angle = destination;
                    }
                }else if((abs(destination - current_angle) > 180)){
                    count = ((abs(destination - current_angle))/45);
                    for(int i=0; i<(8-count); i++){
                        clockwise();
                    }
                    current_angle = destination;
                }
            }
        }
    }
    return 0;
}
```