LAB: Smart mini-fan with STM32-duino

Date: 2022/09/20

Author/Partner: 21800447 오재현/ 21800501 이기윤

Github: https://github.com/Ohjeahyun1/Embedded-controller/blob/7e0a01768d88300a05a4ca949732f234725da59e/lab1_final.ino

Demo Video: https://youtu.be/RfFo4KIwgmw

Introduction

In this lab, you are required to create a simple program that uses arduino IDE for implementing a simple embedded digital application. Refer to online arduino references for the full list of APIs.

Requirement

Hardware

- MCU
  - NUCLEO-F401RE
- Sensor:
  - Ultrasonic distance sensor(HC-SR04) x1
- Actuator/Display
  - LED
  - DC motor Pen (RK-280RA)

Software

- Arduino IDE

Problem

Procedure

The program needs to run the Fan only when the distance of an object is within a certain value.

Example: An automatic mini-fan that runs only when the face is near the fan. Otherwise turns off.

- As the button **B1** is pressed, change the fan velocity. The MODE(states) are
  - MODE(state): **OFF(0%), MID(50%), HIGH(100%)**
- When the object(face) is detected about 50 mm away, then it automatically pauses the fan temporarily.
  - Even the fan is temporarily paused, the MODE should be changed whenever the button **B1** is pressed
- When the object(face) is detected within 50mm, then it automatically runs the fan
  - It must run at the speed of the current MODE

- LED(**LED1**): Turned OFF when MODE=OFF. Otherwise, blink the LED with 1 sec period (1s ON, 1s OFF)
- Print the distance and PWM duty ratio in Tera-Term console (every 1 sec).
- Must use Mealy FSM to control the mini-fan
    - Draw a FSM(finite-state-machine) table and state diagram
    - Example Table. See below for example codes

Present State	Next State (X, Y)				Output Z			
	(0, F)	(0, T)	(1, F)	(1, T)	(0, F)	(0, T)	(1, F)	(1, T)
S0	S0	S0	S1	S1	V=0 L=0	V=0 L=0	V=0 L=1	V=50 L=1
S1								
S2								

Configuration

Ultrasonic distance sensor

Trigger:

- Generate a trigger pulse as PWM to the sensor
- Pin: **D10** (TIM4 CH1)
- PWM out: 50ms period, 10us pulse-width

Echo:

- Receive echo pulses from the ultrasonic sensor
- Pin: **D7** (Timer1 CH1)
- Input Capture: Input mode
- Measure the distance by calculating pulse-width of the echo pulse.

USART

- Display measured distance in [cm] on serial monitor of Tera-Term.
- Baudrate 9600

DC Motor

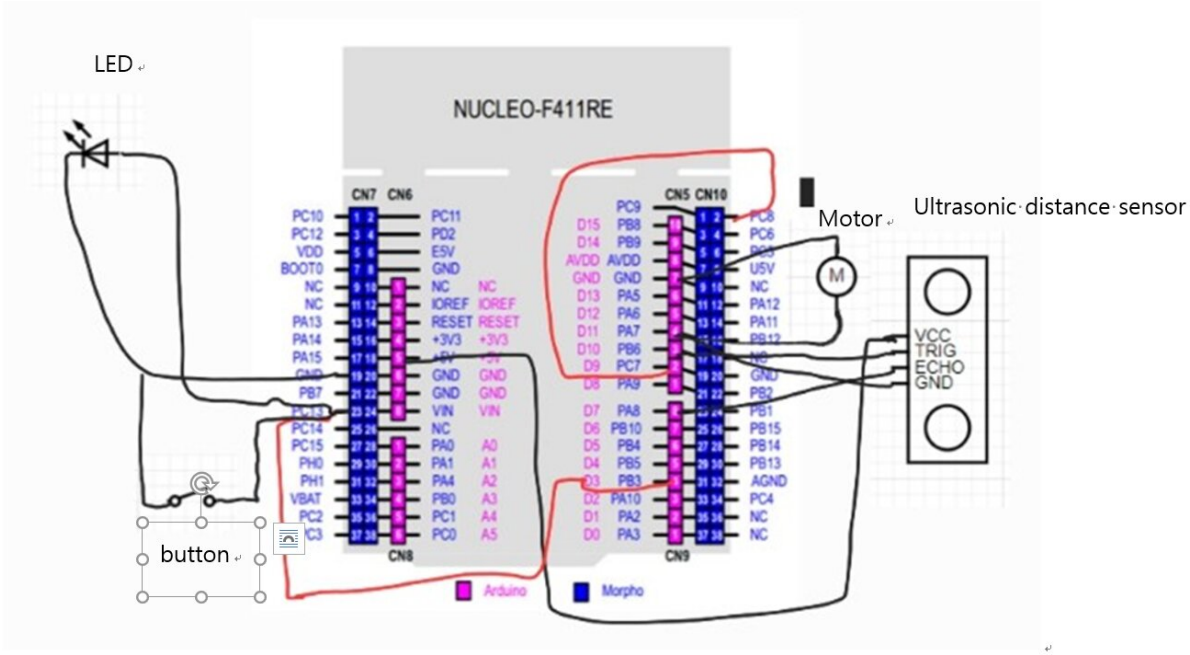
- PWM: PWM1, set 10ms of period by default
- Pin: **D11** (Timer1 CH1N)

LED

- Pin: Pin13

Circuit/Wiring Diagram

External circuit diagram that connects MCU pins to peripherals(sensor/actuator)



Algorithm

Overview

Flowchart or FSM table/graph goes here

Mealy FSM Table

Present State	Next State (X, Y)				Output Z			
	(0, F)	(0, T)	(1, F)	(1, T)	(0, F)	(0, T)	(1, F)	(1, T)
S0	S0	S0	S1	S1	V=0 L=0	V=0 L=0	V=0 L=1	V=50 L=1
S1	S1	S1	S2	S2	V=0 L=1	V=50 L=1	V=0 L=1	V=100 L=1
S2	S2	S2	S0	S0	V=0 L=1	V=100 L=1	V=0 L=0	V=0 L=0

Description with Code

- Lab source code: https://github.com/Ohjeahyun1/Embedded-controller/blob/7e0a01768d88300a05a4ca949732f234725da59e/lab1_final.ino
- Description 1

```
// State table definition
typedef struct {
  uint32_t out[4][2];     // output = FSM[state].out[input][PWM or LED]
  uint32_t next[4];       // nextstate = FSM[state].next[input]
} State_t;

//Define the state according to the state table

State_t FSM[3] = {
  { {{0   , LOW }, {0   , LOW}    ,{0, HIGH},{80, HIGH}} , {S0,S0,S1, S1} },
  { {{0   , HIGH}, {80  , HIGH}  ,{0, HIGH},{160, HIGH}}, {S1,S1,S2, S2} },
  { {{0   , HIGH}, {160  , HIGH},{0, LOW} ,{0, LOW}}    , {S2,S2,S0, S0} }
};
```

Use the FSM State Table to define the output and the next state for the input and present state

- Description 2

```
/*ultra sonic distance detection code*/

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(10);

  duration = pulseIn(echoPin, HIGH);
  distance = (float)duration / 58.0*10;
  if (distance < 50) input2 = 1;          //On if closer than 50mm ->input2
on
  else input2 = 0;                        //Off if further than 50mm -
>input2 off
```

- Description 3

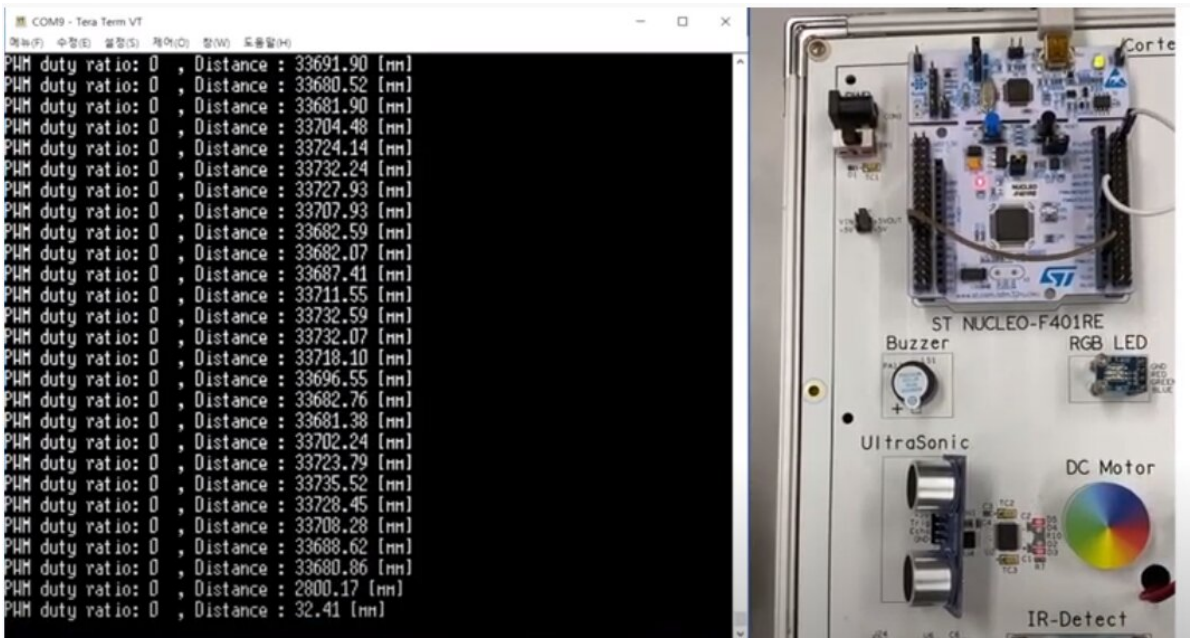
```
void pressed(){
  input1=1;
}

void nextState(){
  //Determine the output according to state table input1,2

  if(input1==0&&input2==0) {input=0;}
  else if(input1==0&&input2==1) {input=1;}
  else if(input1==1&&input2==0) {input=2;}
  else if(input1==1&&input2==1) {input=3;}

  pwmOut = FSM[state].out[input][PWM];
  ledOut = FSM[state].out[input][LED];

  nextstate = FSM[state].next[input];

  state = nextstate;  //state update
```

```
  //Initialization
  input1=0;
  input2=0;
  input =0;
}
```

input1 and input2 are the values for the input.
Input1 is to change when the button is pressed.
Input2 is whether the fan is operating or not depending on the distance.
The following states and outputs were determined based on the values of input1 and input2.
I also initialized each value at the end.

Results and Analysis

Results



Pressing the button changes the mode of controlling the fan speed. If the mode is not OFF, the LED is blinking. Also, if the mode is not Off and the distance detected by UltraSonic is less than 50mm, the fan will operate.

Demo Video

Analysis

By pressing the button, the speed of the fan could be adjusted and the distance could be sensed to perform the fan motion.

Reference