New York City College of Technology

300 JAY STREET, BROOKLYN, NEW YORK, 11201

Computer Engineering Technology

CET4711
Computer Controlled System Design I

Lab Report # 6

Finite State Machine

Student:
Yeraldina Estrella
Prof: Zia
Date: 3/25/15

Objective

To learn basic concepts of Finite State Machine (FSM).

To learn basic application of Finite State Machine model in Computer Controlled System design.

To learn the implementation of Finite State Machine Library for Arduino.

To learn the design of a simple computer controlled system by using a Finite State Machine implementation in software.

Procedure

Part 1 (Installation of Third Party Libraries)

Obtain and import the following Library Files to Arduino.
Button.zip
FSM.zip
LED.zip

Part 2 (Hardware and Software Test)

- Wire the circuit on the schematic diagram (Figure 1).
- Open the FSMExample arduino program.
- Compile and upload the program and verify that the circuit works as expected

Part 3 (Program Modification)

Modify the FSMExample to obtain the following:
Add more states to turn a buzzer on and to turn off the buzzer.

Diagrams

Hardware/ Software: Schematic diagram (drawn on *Fritzing*)

Figure 1 (Final State Machine Schematic Diagram)

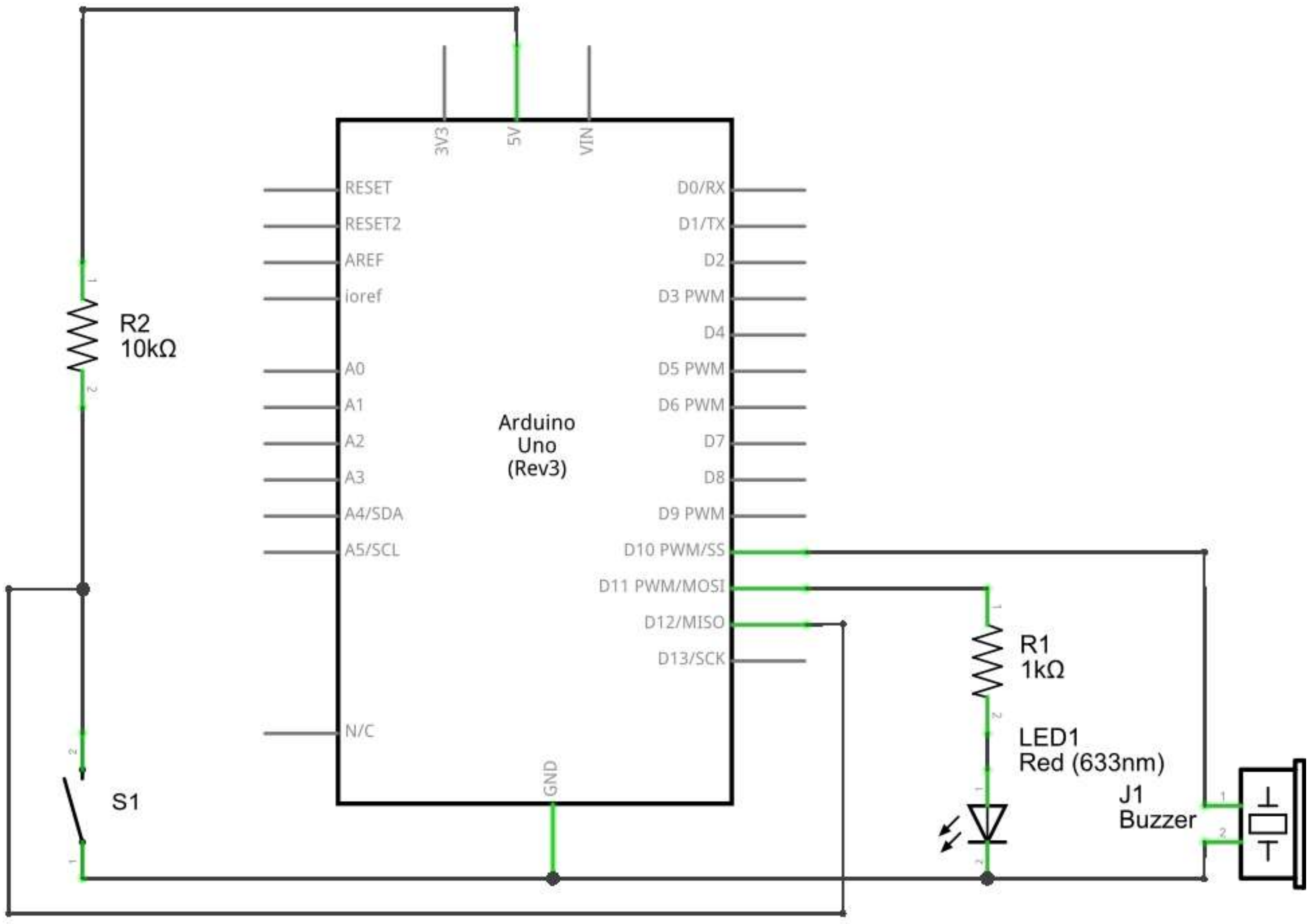
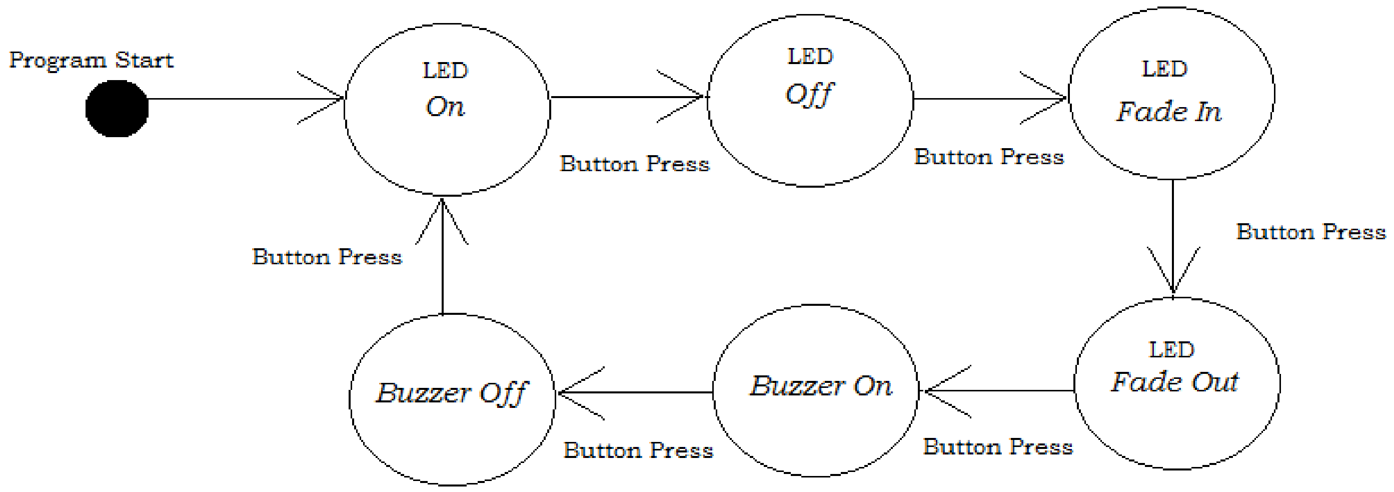


Figure 2 (Final State Machine Diagram)



Modified FSMExample Program

```
/*
 FSMExample
 This program uses external transitions, caused by a user button presses.
 Each button press will cause the FSM to advance to the next State.
*/
//http://playground.arduino.cc/uploads/Code/FSM.zip
#include <FiniteStateMachine.h>
//http://playground.arduino.cc/uploads/Code/Button.zip
#include <Button.h>
//http://playground.arduino.cc/uploads/Code/LED.zip
#include <LED.h>

const byte NUMBER_OF_STATES = 6; //how many states are we cycling through?

//initialize states
State On = State(ledOn);
State Off = State(ledOff);
State FadeIn = State(ledFadeIn);
State FadeOut = State(ledFadeOut);

FSM ledStateMachine = FSM(On);     //initialize state machine, start in state: On

#define buzzer 10    // buzzer is connected to pin 10
State BuzzOn = State(buzzerOn);
State BuzzOff = State(buzzerOff);

Button button = Button(12,PULLUP); //initialize the button (wire between pin 12 and ground)
LED led = LED(11);                 //initialize the LED
byte buttonPresses = 0;            //counter variable, holds number of button presses

void setup()
{
 pinMode(buzzer, OUTPUT);   // configure buzzer pin as output pin
}
void loop(){
  if (button.uniquePress()){
    //increment buttonPresses and constrain it to [ 0, 1, 2, 3, 4, 5 ]
    buttonPresses = ++buttonPresses % NUMBER_OF_STATES;
    switch (buttonPresses){

      case 0: ledStateMachine.transitionTo(On); break;
      case 1: ledStateMachine.transitionTo(Off); break;
      case 2: ledStateMachine.transitionTo(FadeIn); break;
      case 3: ledStateMachine.transitionTo(FadeOut); break;

      case 4: ledStateMachine.transitionTo(BuzzOn); break;
      case 5: ledStateMachine.transitionTo(BuzzOff); break;
    }
  }
  ledStateMachine.update();
}
//utility functions
void ledOn(){ led.on(); }
void ledOff(){ led.off(); }
void ledFadeIn(){ led.fadeIn(500); }
void ledFadeOut(){ led.fadeOut(500); }
void buzzerOn(){ digitalWrite(buzzer,HIGH); }
void buzzerOff(){ digitalWrite(buzzer,LOW); }
//end utility functions
```

Troubleshooting

The LED did not turn on.

- What was the problem that you encountered? The LED was not turning on.
- What did you think was the cause of the problem? The wiring of the LED.
- What was done to fix the problem? I revised the diagram and found that was properly connected. I found out that the jumper cable connecting the LED to ground was malfunctioning.

Discussion

An Arduino program or scrip contains five sections which are: comments, initialization statements, setup functions, loop functions and users' defined functions. This laboratory was performed using the first four sections of an Arduino program. I studied the functionality of the analog input and analog output and data communication functions of the Arduino microcontroller. The analog value of an analog pin can be displayed in the serial monitor of the Arduino IDE. This is when data communication between the microcontroller and a PC is established. An analog pin can be used as an analog output to send a signal or as an analog input to receive a signal. In this lab, digital input, digital output and data communication was implemented.

The finite state machine has many applications in computer controlled systems designs. Finite state machines can be implemented in many controlled systems projects such as a security system where an alarm is turn on by pressing a button and turn off by pressing a button again. In this lab, I learned to that finite state machine can be implemented in software. In the lab, the FiniteStateMachine (FSM) example program was modified to produce a Finite State Machine as the one in Figure 2. Three libraries were added to the Arduino libraries in order to perform the FSM example program. The states of the each function are initialized in the program code right before the setup function. The FSM was initialized to turn on the LED at the beginning of the program. To change the state of the FSM, switch cases were used in the loop function. Each case indicates the state of the FSM as the switch button is pressed. After the loop function, the utility functions were declared. (refer to the modified FSM program). As described in Figured 2, the FSM starts by initially having the LED on and as the switch button is pressed, the LED turns Off and by pressing the switch button again the LED fade In, and subsequently, the LED fade Out, Buzzer turn ON, Buzzer turn Off and as a final button pressed, the FSM goes back to the initial state (LED ON).