Key Lock-Box Control

Project Report

Yuchen Luo

X00192070

Contents

[Abstract 2](#_Toc125803116)

[Preface 2](#_Toc125803117)

[Introduction 3](#_Toc125803118)

[Components of the lock box 3](#_Toc125803119)

[Client Software 3](#_Toc125803120)

[Features 4](#_Toc125803121)

[Environment and Ethics 4](#_Toc125803122)

[Environment requirements: 4](#_Toc125803123)

[Ethical requirements: 4](#_Toc125803124)

[Conclusion&Discussion 4](#_Toc125803125)

[References 5](#_Toc125803126)

[Appendix 5](#_Toc125803127)

# Abstract

This project implements a web server on an Arduino. This system will interface with sensors and other embedded hardware attached to a microcontroller based development board. This Linux board and the microcontroller will communicate wirelessly.

The system will have client software that will run in a browser; this will use HTML and JavaScript to implement its functionality. Server software that will run in the Linux board will serve up the webpages to the client machines. This server software can be implemented in JavaScript using a Node.js framework.

The communications between the Linux machine and the client’s browser can be implemented in HTTP and use the "socket.io" JavaScript library.

The communications between the microcontroller boards and the Linux machine use Bluetooth.

**Keywords**: Software, Launchpad, Arduino, Programming

# Preface

I would like to express my very great appreciation to Michael Gill for his patient guidance, enthusiastic encouragement and useful critiques of this project. His willingness to give his time so generously has been very much appreciated.

I'd also want to thank my classmates for their technical advice and assistance.

# Introduction

The key owner could use this system to control access to keys for houses and premises. A key could be placed in a lock box outside a premises. An authorised person would logon to the website hosted by the Linux system. A code would be issued that would allow the lock-box to be opened via a keypad. The server would control the duration of the code remotely.

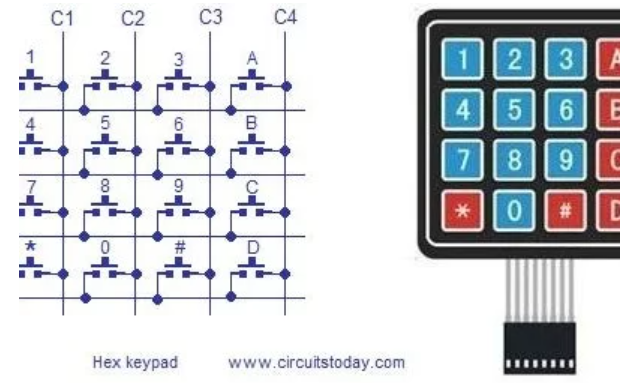
# Components of the lock box

* Arduino
* Matrix Keypad 4x4
* Micro Arduino Servo Motor SG90
* Box



## The code (of keypad 4x4)

* Make sure to declare the servo pin as 4 (any number except 2 or 3: both of them will not work if LCD occupies SDA and SCL).
* Different rows and columns belong to different pins, so make sure to declare the correct one.
* Set up own passcode for the lock.
* “resetLocker” means when the system turns back to origin: LCD prints “Get Some Food” and “Pin”, and the servo turns to 40 degrees, which locks the box (the degree depends on different servo or the position of the servo).
* “unlockdoor” works if the user enters the correct password, making the servo to turn to 110 degree (open) and the LCD print “pass”. In the other hand, the LCD will print “Wrong! Try Again” if the passcode is incorrect.
* By pressing “\*”, users can clear the password they entered; by pressing “#”, the machine can check for the passcode.



# Client Software

It is run in a browser. Server software that runs in the Linux board will serve up the webpages to the client machines.

* HTML and JavaScript

It use HTML and JavaScript to implement its functionality

* Node.js

This server software can be implemented in JavaScript using a Node.js framework.

* HTTP and “Socket.io” JavaScript library

The communications between the Linux machine and the client’s browser can be implemented in HTTP and use the "socket.io" JavaScript library.

# Features

Using the QR code without keypads

# Environment and Ethics

The highlight of it is its ultra-low power consumption. Ultra-low-power means that this board has a low-power mode, after entering the low-power mode it can standby for a long time with very little consumption, waking up the CPU for work at regular intervals, which is necessary to save energy when there are energy sources.

## Environment requirements:

The efficient design of circuits thus minimizing components;

I initiate the design of PCB boards to be as small as possible.

In the design and production of projects keep power use to a minimum, by switching off unused equipment when appropriate;

All electronic/electrical waste be disposed of correctly.

## Ethical requirements:

Maintain a clean work environment by designing software that is efficient and avoids wasting computing resources. Understand how to utilize equipment correctly; WEEE standard

Any work should be free of plagiarism, and all help should be properly acknowledged.

The Comreg standard is used in telecommunications.

Copyright and software licensing should be honoured.

# Conclusion&Discussion

This project has given me a systematic grasp of the process of learning to do a project. I was confused about the logical relationship between the various sections at the beginning, so I should have a better grasp of the project as a whole in advance for the next study.

# References

# Appendix

## Arduino code

#include <Keypad.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <Servo.h>

// Declare the Servo pin

int servoPin = 4;

// Create a servo object

Servo Servo1;

#define KEY\_ROWS 4 // Keypad rows

#define KEY\_COLS 4 // Keypad columns

#define LCD\_ROWS 2 // LCD rows

#define LCD\_COLS 16 // LCD columns

// 設置按鍵模組

char keymap[KEY\_ROWS][KEY\_COLS] = {

{'1', '2', '3', 'A'},

{'4', '5', '6', 'B'},

{'7', '8', '9', 'C'},

{'\*', '0', '#', 'D'}

};

byte rowPins[KEY\_ROWS] = {13, 12, 11, 10};

byte colPins[KEY\_COLS] = {9, 8, 7, 6};

Keypad keypad = Keypad(makeKeymap(keymap), rowPins, colPins, KEY\_ROWS, KEY\_COLS);

String passcode = "2580"; // Setting password

String inputCode = ""; // Save the password

bool acceptKey = true; // Accepting the password entered

LiquidCrystal\_I2C lcd(0x27, 16, 2);

void clearRow(byte n) {

byte last = LCD\_COLS - n;

lcd.setCursor(n, 1); // Change to the 2nd row, then prints "Pin:"

for (byte i = 0; i < last; i++) {

lcd.print(" ");

}

lcd.setCursor(n, 1);

}

void resetLocker() {

lcd.clear();

lcd.print("Get Some Food");

lcd.setCursor(0, 1); // Change to the 2nd row

lcd.print("PIN:");

lcd.cursor();

Servo1.write(40); //The servo locks

acceptKey = true;

inputCode = "";

}

void unlockdoor(){ //controls servo that locks the door

Servo1.write(110); //Degree when the lock opens

lcd.setCursor(0,1);

clearRow(0);

lcd.print("\*\*\*\* PASS! \*\*\*\*");

delay(1500); //To make sure the LCD receives the message

}

void checkPinCode() {

acceptKey = false; // Not accepting any enter for passcode

clearRow(0); // Clear from the 0 numer

lcd.noCursor();

lcd.setCursor(0, 1); // Change to the 2nd row

// Check the passcode

if (inputCode == passcode) {

lcd.print("\*\*\*\* PASS! \*\*\*\*");

unlockdoor();

} else {

lcd.print("WRONG! Try Again!");

}

delay(7000);

resetLocker(); // Reset LCD and servo

}

void setup() {

digitalWrite( 5 , LOW );

Serial.begin(9600);

lcd.init();

lcd.backlight();

Servo1.attach(servoPin);

Servo1.write(40); //Degree when the servo locks

resetLocker();

}

void loop() {

char key = keypad.getKey();

if (acceptKey && key != NO\_KEY) {

if (key == '\*') { // Clear

clearRow(4); // Clear from the 4th row

inputCode = "";

} else if (key == '#') { // Check passcode

checkPinCode();

} else {

inputCode += key; // Save

lcd.print('\*');

}

}

}