Secure Science Lab

Aayush Mailarpwar

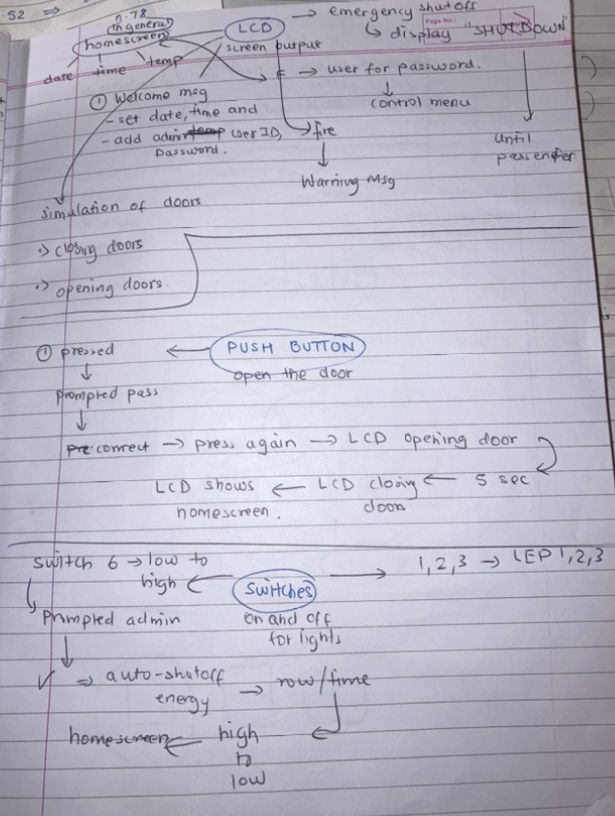
Fall 2021

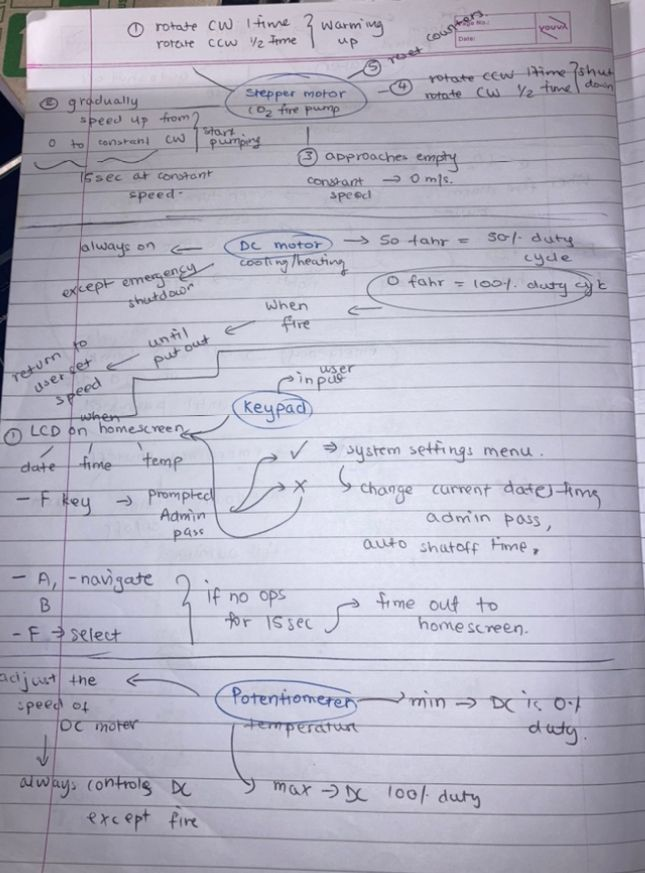


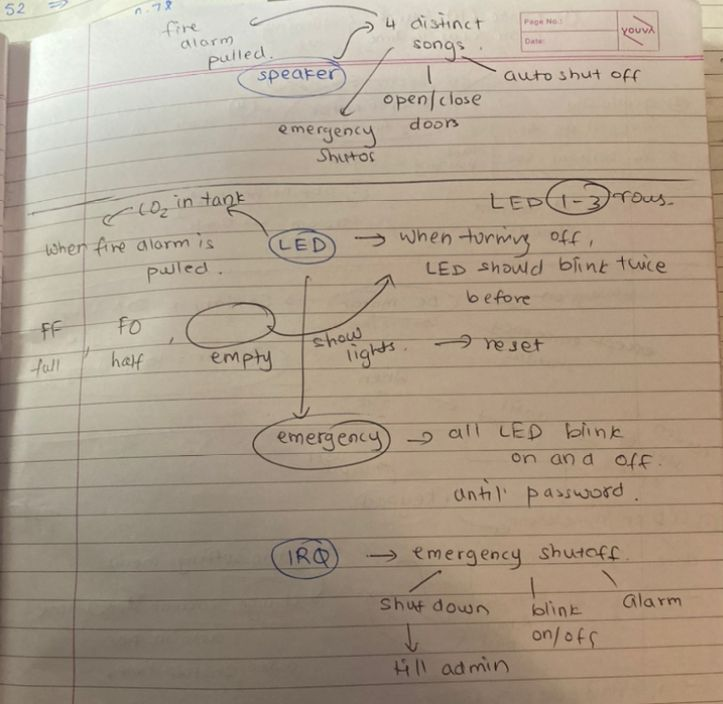
Table of Contents

1. Figures and Tables
2. Introduction
3. Design
4. Working and Not Working
5. Conclusion

Figures and Tables







Introduction

The goal of this lab to create a simulation of the Secure Science Lab using the tools we have learn throughout the semester in Assembly programming using the microcontroller connected to the various peripherals on our lab board. This simulation will have different states that the user will be in and allow the user to navigate through these different states and be able to jump from routine to routine very smoothly while making sure the states of all the variables involved are maintained and to how they must behave.

The purpose of this project is to gauge and evaluate the student’s capability of working with the microcontroller operations and system it is connected to. It is a method to see if student have learnt how to connect and utilized each distinct peripheral in orderly fashion. Our lab has worked through the concepts and application for each singular peripheral but now our lab must combine the usage and order of them to create a smooth-functioning stable project.

Assumptions Made

1. 1ms interval in the RTI\_ISR
2. Some of the times like 10 sec, 5 sec and 15 sec are more by microsecond due to minor lags in the RTI despite it not being too long.
3. DC and Speaker switch on the switches is on the whole time of the project
4. J2 and J5 pins are placed accordingly

Design

Peripherals that were used in project

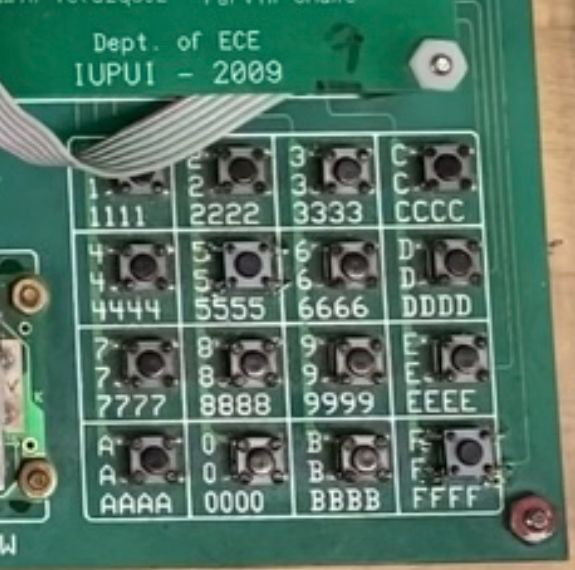
1. Stepper motor – Replicates the fire routine if the switch 8 was flipped up and then down where the air vents push oxygen to put out the fire.
   1. Activated during fires
   2. Rotate clockwise for one rotation
   3. Half rotation counterclockwise
   4. Gradual increase in speed
   5. Constant rotation for 15sec
   6. Gradual decrease in speed
   7. Half rotation clockwise
   8. Rotate counterclockwise for one rotation
   9. Return to normal condition



1. DC Motor – replicates the cooling and heating fan.
   1. DC motor is connected to the temperature which is controlled by the potentiometer.
   2. Temperature can go from 0 to 100 F and 0 F is 100% duty cycle and 100 F is 0% duty cycle.
   3. This should shutoff during emergency routine
   4. This should be at 100% duty cycle during fire



1. Keypad – Main form of user input in the project
   1. Password
   2. F for select
   3. A and B for navigation
   4. Setting the Date, Time, Password, Auto shut off lights



1. Potentiometer – Controls the temperature in the lab at any point of the project traverse which is displayed on the LCD in Home screen and controls the duty cycle of the DC Motor



1. LCD – Main form of display through the project
   1. Door Animation
   2. Input when first started including Date, Time, Password
   3. Displays temperature
   4. Emergency shutdown
   5. Fire alert
   6. Navigation through control menu/ system settings



1. Push Button – Opening the lab when the lab is in Home screen
   1. On press, user is asked for password
   2. On second press, door animation occurs



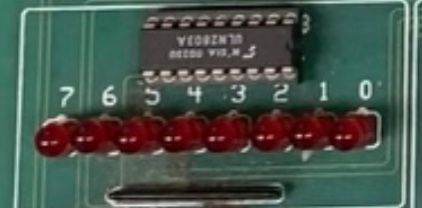
1. Switches – Triggering some vital routines throughout the program
   1. 1 -3 LED lights
   2. Fire Alarm ( Switch 8)
   3. Auto Shutoff timer (Switch 6)



1. Speaker : Play songs four different occurrences
   1. Song1 for fire alarm
   2. Song2 for emergency routine
   3. Song3 for auto shut off
   4. Door opening and closing



1. LED
   1. Carbon dioxide levels after the fire alarm has been triggered
   2. 1 – 3 can be controlled by the switches
   3. Blinking on turning one of them off
   4. Emergency routine – all LED flashes



1. IRQ – Activate the emergency routine



Software Implementation of the Project

1. Description and Discussion

When the start the project for the first time, user is asked for the time, date and password and they must enter it using the keypad. As each key is entered, it is displayed on the LCD screen character by character. This is when the cli is not in place and the startup screen subroutine is branched to where the entry and input is done sequentially, and it will not go to the next page unless the previous page is not complete. Therefore, following a very sequential and orderly model. This included calling the keypad routine and continuously calling until there is user input. There included a bunch of displays and validation check until all the data was saved into memory for future use.

Now, that the user has been entered through the input / startup phase, they are taken to the home screen which is the start of my FSM where the date, time and temperature is displayed. These values are got from the variables that were initialized in the startup process and the temperature which received from readpot() function and after mathematical computation, we display it on the screen as decimal value between 0 and 100.

On pressing F, the user is asked to enter the password before entering the control menu where they have an opportunity to change the date, time, password, and auto shut off timers. These are all navigated by the A and B keys using the keypad and then F is to select what option. If no input is selected in 15 seconds, then user is brought to the Home screen again. The change date, time and pass overwrite the values that were initially inputted using the same subroutines that were called in the beginning.

Throughout the program running, if the IRQ button is pressed an emergency flag is set which is check in the Home routine and then we branch to an Emergency Routine where the DC motor stops by sending the max value to t\_on, lights flash using toggling in the RTI, display changes to “Emergency Routine.” The user must enter the password to exit this routine back to the Home Screen, thereby calling the Ask Password Routine using the predefined passkeys.

Throughout the program running, if the switch is flipped up and then down, the fire flag is initiated and then the Fire Routine is called where a lot of different peripherals do their job. The display shows there is a Fire and scientist have 10 sec to leave the lab and the RTI handles that and as soon as the 10 sec is reached, the stepper starts rotating and doing the routine mentioned in the peripheral analysis. Simultaneously, the LED count down from FF to 00 by dividing by zero every second and when 00 is reached, it means the CO2 tank is empty and Fire routine is existed. At this time 0 is set to t\_on so that it has 100% duty cycle. This takes about 15 seconds.

On the Home Screen, if the push button is pressed, firstly the Ask Password routine is called and on correct input the user is prompted to press the button again and the Door Animation routine occurs where I used the RTI with half a second delay between each phase of the door both opening and closing with a 5 second delay in between.

1. A whiteboard with writing on it

   Description automatically generated with medium confidenceFlow Chart
2. Error Handling

I had a bunch of flags that were checked continuously throughout the program and was checked in each routine. One thing I had to consider was the turning on and off the flag for each situation and making sure all the counters and internal flags were reinitialized to the original.

One thing I realized was when I was returning from each of the subroutines, I would branch to something like the Home Screen which would inevitably push my registers to stack according to our stack implementation practices. Therefore, I would be getting an Illegal Breakpoint error since my subroutine would not be able to return from it.

Changes I made in the design –

I did not make too many changes since it was straightforward, but I feel that using the FSM not in the main but in the Home Screen was easy to implement and less time consuming and I could cover all the required entities and peripheral simulations without exiting the subroutines.

Additions to the project –

* I added an input check for the date where it was much more precise with what can be taken into the system.
* My 10 and 15 second timer was shown on the screen using the RTI and display functions
* I made the date and time appear character by character as it was input from the keypad.

Work between Team Members

This project was solely done by me.

Working and Not Working

1. Auto shutoff lights was not able to work due to it not being able to synchronize with my RTI made clock and therefore, it could not be implemented.
2. Stepper had only basic functionality of spinning clockwise, counterclockwise, and constant speed. I was not able to implement the steady increase of the stepper motor.

Conclusion

This project was a successful one that required immense amount of planning, time, and dedication. I put in a lot of hard work, working day and night towards the end to be able to demonstrate the project. I feel that I have mastered almost all the peripherals and them simulating what must be done in each situation.

Future improvements

There were many areas the project could have had some further developments but here are a few I thought about.

1. There could have been user input validation for the password to be of a certain length and must having both numbers and A – F.
2. As the emergency routine stops the DC motor, I could have gradually increased the temperature since the fan in the science lab is not on.
3. In a situation of a fire, I could have increased the temperature to the maximum due to the heat induced.

Appendix

1. Added in software track
2. Added along with the project in zip file

Reference

No external references except for Lab Manual in ECE 362