ENGINEERING DESIGN PROJECT

FINAL REPORT

FIRE SENSING SYSTEM WITH ALARM

W. M. A. A. Sovis (852)
G. D. A. Deshappriya (880)
I. D. N. Hirunika (877)
S. M. Perera (878)

Abstract

'Fire, water, and government know nothing of mercy.'
- *Proverb*

Fire can destroy your house and all of your possessions in less than an hour, and it can reduce an entire forest to a pile of ash and charred wood. It's also a terrifying weapon, with nearly unlimited destructive power. Fire kills more people every year than any other force of nature.

But at the same time, fire is extraordinarily helpful. It gave humans the first form of portable light and heat. It also gave us the ability to cook food, forge metal tools, form pottery, harden bricks and drive power plants. There are few things that have done as much harm to humanity as fire, and few things that have done as much good. It is certainly one of the most important forces in human history.

(Harris, n.d.)

Yet when it comes to safety, fire is something we all don't want near us. Thus, it is important that we come up with ways to seek protection from fire. With this in mind we designed the concept for the Fire Extinguishing Robotic System. This article will be an in-depth analysis of this abstract system that can identify and eradicate fires.

Table of Contents

Introduction

Fire is the result of applying enough heat to a fuel source, when we have got a whole lot of oxygen. Fire needs fuel, oxygen and heat, in the right combination, to occur naturally. If just one of those elements is taken away, a fire can't happen. Nature has been making fires since the planet first sprouted trees, introducing the abundance of fuel and oxygen needed for fires to take place. Today, the causes of fire are quite a bit more complicated.

The theory of fire extinguishment is based on removing any one or more of the above elements in the fire tetrahedron to suppress the fire.

Our main goal was to create something beyond what is expected of us. The goal was to create a complete fire extinguishing system that can handle a fire outbreak. The system will be able to completely decide on itself on what to do based on the inputs which can be overridden by humans and to take action on itself. And we have achieved this as follows.

According to our plan this system can do 3 main tasks. They are detecting fire, indicating fire and extinguishing fire by using necessary precautions. So, we used 3 main components that connected to the control unit of the system. And the other main component is software system. Detection and extinguishing probes are used to identify or detect the high temperature and find out the location of the fire from this unit the relevant data or information are given to the control system and extinguishing the fire can be done by this unit. After getting all those data, the control unit pass the information to the alarm and notification unit. So, the alarm will be turn on in the case of fire. The monitoring software that is connected to the control unit is a software that is used to control all the equipment and sensors of the system.

The fire detection cycle can be described as follows; smoke signatures are constantly detected by the smoke sensor. If a smoke signature is detected, the microcontroller is notified, and the temperature measurements from the sensor are taken. It will also turn on the valve that will shower water on the fire while doing so. A buzzer will sound to signal the start of the alarm. The control software will also display temperature readings from the sensors, as well as other necessary information. The valve will be closed, the alarm will be hushed, and smoke sensing will begin once the temperature sensor finds that the temperature is normal. A notification will be sent to the control software.

The control software plays a main part in our system. It can monitor status of each sensing probe modules, override alarms, turn off valves or even turn off the system completely. In this sense, our system can be considered a complete fire alarm system.

Project Overview

This fire alarm system is designed to detect, indicate and extinguish fire by using necessary precautions. The system is expected to consist of three main stages.

1. Fire sensing and extinguishing system
2. Control stage
3. Alarming and notification stage

There are four main sections and components for the proper functioning of the system. It will help to complete that all the stages that mentioned above.

1. Fire sensors
2. Extinguishing valves
3. Control unit
4. Software system
5. structural design

We can have as many of the Fire Alarm Modules as we like according to the implemented environment. For example, if we are implementing in a school, each classroom can have individual modules. But accordingly, we will have to upgrade the control system used.

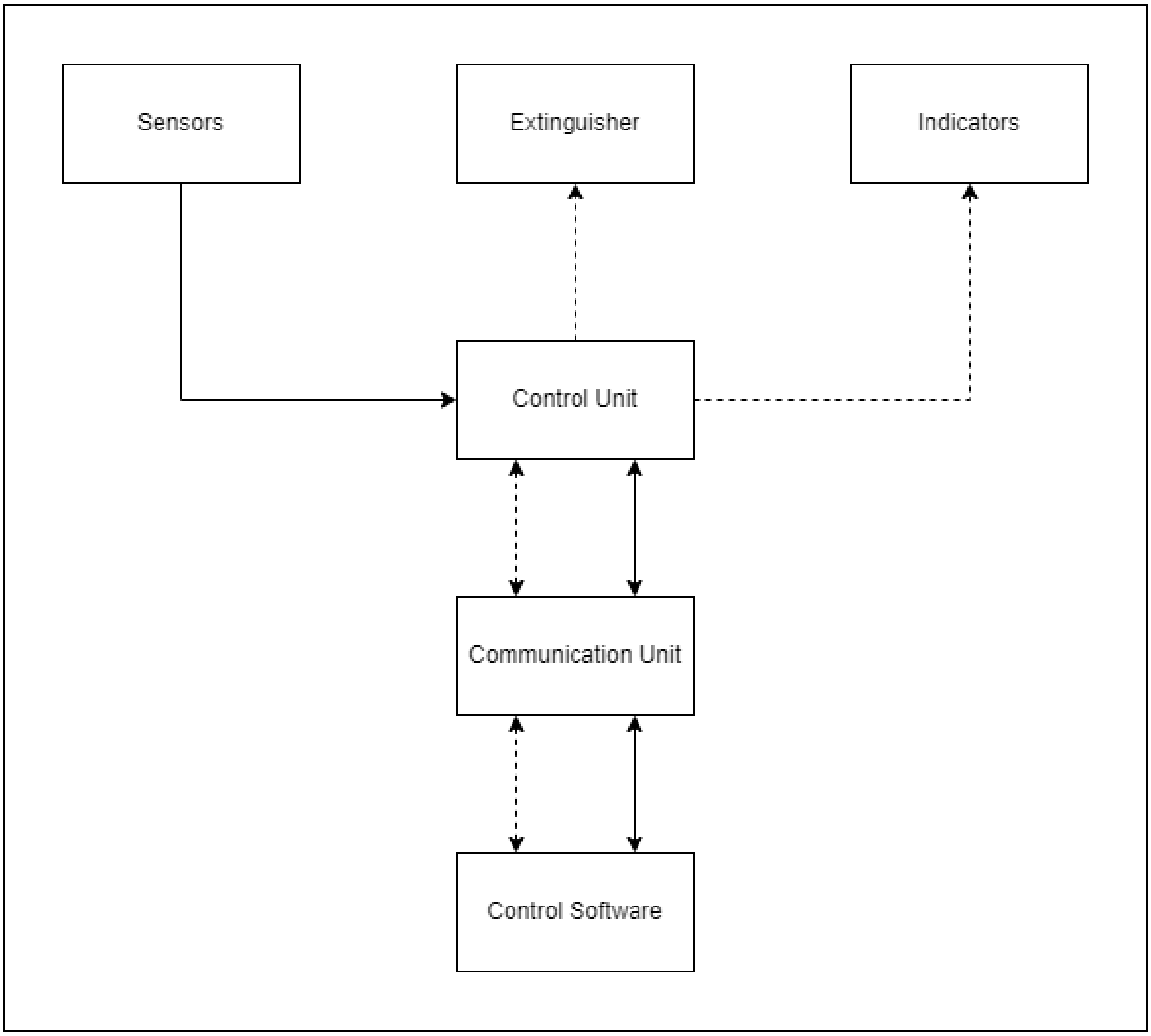


Figure 1Structure

01. Control Unit

This is the brain of the system. It consists of a microcontroller which will be controlling the whole system. This also interfaces between all other components. The microcontroller used is the Atmega 328p This has a good count of pins and is suitable for the task. It will be powerful enough to control the system.

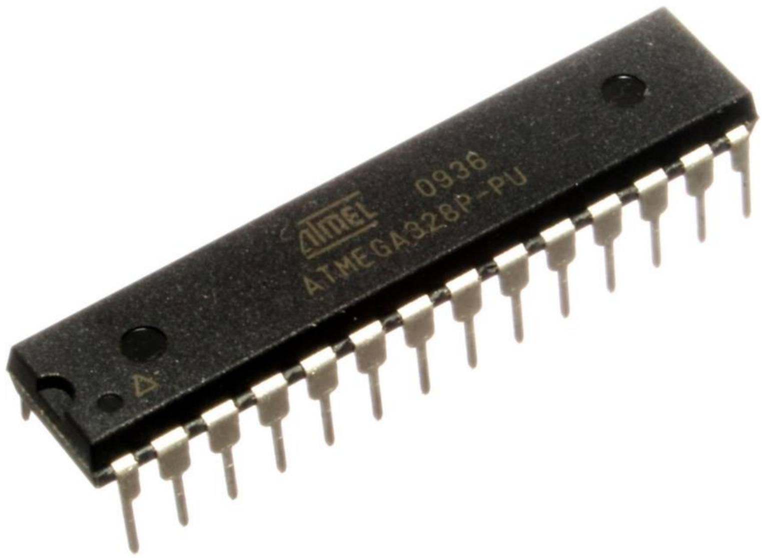


Figure 2ATmega 328p Microcontroller

Specifications

- CPU type - 8-bit AVR
- Maximum CPU speed - 20 MHz
- Performance - 20 MIPS at 20 MHz
- Flash memory - 32 KB
- SRAM - 2 KB
- EEPROM - 1 KB
- Package pin count - 28 or 32
- Capacitive touch sensing channels - 16
- Maximum I/O pins - 23
- External interrupts – 2

02. Fire Sensors

This is the module that directly interact with the fire. This consists of two components as listed below.

01. Smoke Sensor – Detect the existence of a fire.
02. Temperature and Humidity Sensor – Detect the thermal changes in the surrounding

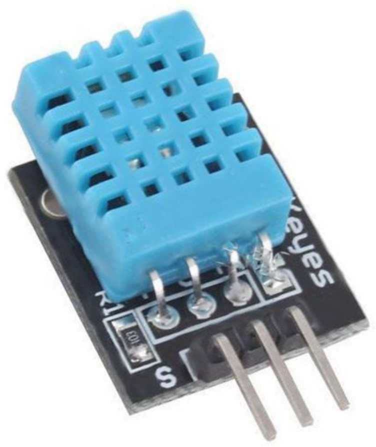


Figure 3DHT 11 Temp and Humid Sensor

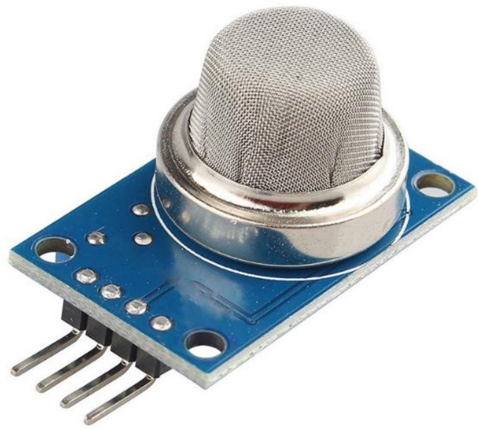


Figure 4MQ 135 CO2 Sensor

Specification

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

Specification

- Operating Voltage: 2.5V to 5.0V
- Power consumption: 150mA
- Detect/Measure: NH3, Nox, CO2, Alcohol, Benzene, Smoke
- Typical operating Voltage: 5V
- Digital Output: 0V to 5V (TTL Logic) @ 5V Vcc
- Analog Output: 0-5V @ 5V Vcc

They will sense the following three from the environment.

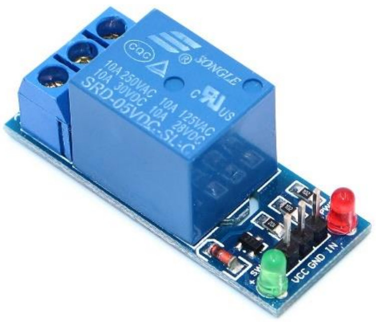
01. Temperature – DHT11 sensor
02. Humidity – DHT11 sensor
03. Carbondioxide – MQ 135 sensor

DHT 11 is communicating with the microcontroller digitally while the MQ 135 senses analog.

03. Extinguisher

The extinguisher consists of three parts. It handles the extinguishing of the fire.

01. 5V relay – Convers the 5V control signal to 12V so that the solenoid valve can operate
02. Solenoid Valve – Controls the water flow
03. Nozzle – Sprays the water onto the fire so that the extinguishing is effective



Specification

- Supply voltage: 3.75V to 6V
- Quiescent current: 2mA
- Current when the relay is active: ~70mA
- Relay maximum contact voltage: 250VAC or 30VDC
- Relay maximum current: 10A

Figure 55V Relay

Specification

- Material: Metal + plastic
- Voltage: DC 12V
- Power: 15W
- Current: 1.25A
- Inlet and outlet thread diameter: G1/2
- Pressure: 0.02 - 0.8Mpa
- Max fluid temperature: 80°C
- Operation mode: Normally Closed

Figure 6Solenoid Valve

Figure 7Nozzle

While the relay and valve were bought, the nozzle was custom fabricated to suit the needs.

04. Indicators

The indicators give out an indication to the people about the fire. We included two indicators, both visual and auditory for the best outcome.

01. Visual – A red LED
02. Auditory – A 5V Buzzer

Figure 85V Buzzer

Figure 9Red LED

- The red LED will blink twice every second to indicate that the system is operating normally. If it stays continuously on, then there is a warning or extinguishing has started. Any other combination or no light at all means that the device is turned off or is not working properly.
- The buzzer stays off when the system is operating normal. It will beep once every second in case of a warning is issued. It will then continuously buzz in case of an extinguishing.

05. Communication Unit

This unit is responsible for communicating with the command panel. The communication unit comprises of the USB to TTL device. This device provides an interface between the PC and the microcontroller via a USB cable. Thus, data and command signals can be exchanged.

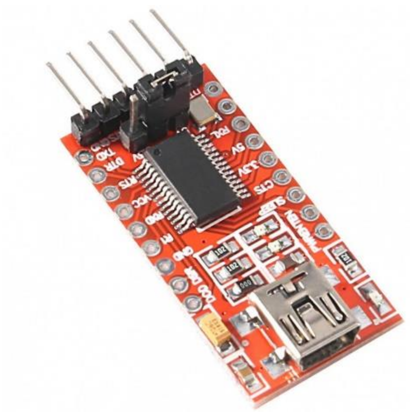


Figure 10FT232RL USB to TTL Converter

Specification

- Voltage: 3.3V, 5V
- Chipset: FT232RL
- USB power has over current protection using 500MA self-restore fuse
- RXD/TXD transceiver communication indicator
- Pin definition: DTR, RXD, TXD, VCC, CTS, GND
- Pitch: 2.54mm
- Module size: 36mm x 17.5mm

06. Control Software

The control software is a piece of software that is written so that the user can interface with the device. It connects to the device, displays data and statuses, allows the user to change settings and also view historical data for devices. The control software supports multiple devices to be operated at the same time. It is written in C# and is currently only available for Microsoft® Windows™. The source code is available on [Github](#).

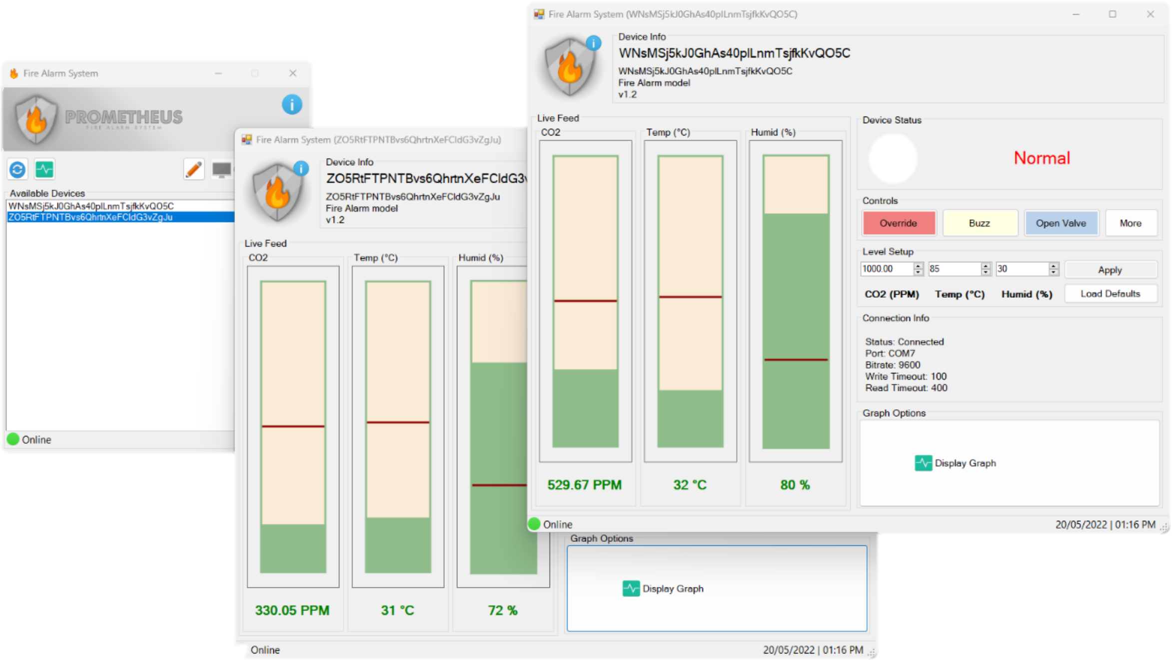


Figure 11Prometheus Control Panel Screenshot

Specification

- OS: Windows 7 or higher
- Processor: 1GHz or higher
- Dot Net Framework: 4.7 or higher
- RAM: 32MB or higher
- Storage: 16MB or higher

Development

Hardware

Prometheus is a fire alarm system and an extinguishing system that allows real time monitoring via software. It can perform smart extinguishing on site while providing connectivity to multiple devices at the same time via the control panel.

The hardware was developed in three stages:

- Prototype 01 – Creating the basic operation on breadboard
- Prototype 02 – Implementing the device on a PCB and testing in real world
- Prototype 03 (Final Prototype) – Assembly and final testing and presentation

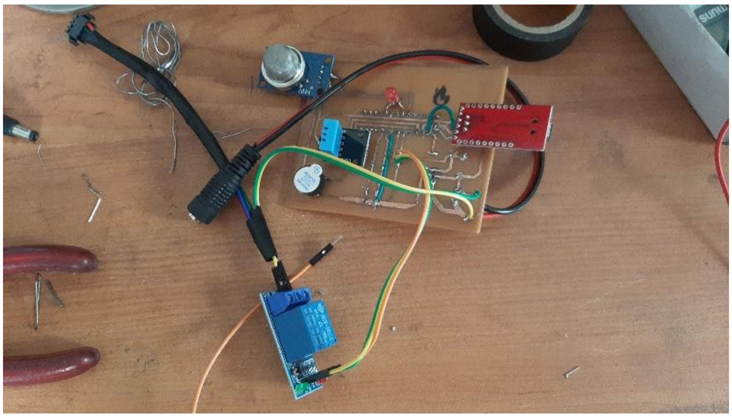


Figure 14Prototype 02

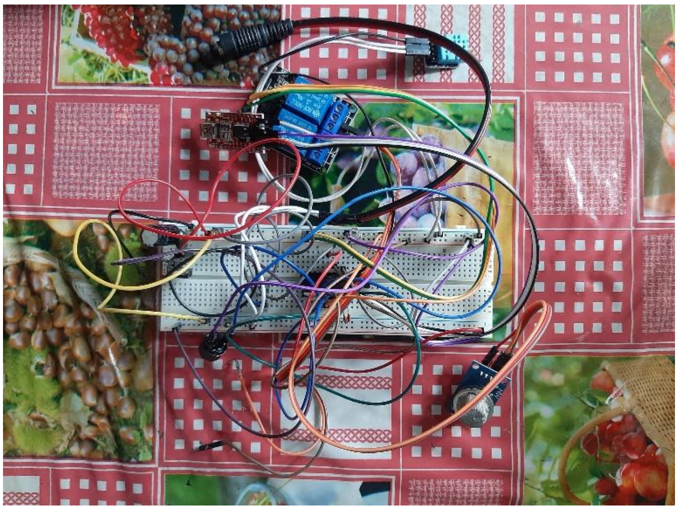


Figure 12Prototype 01

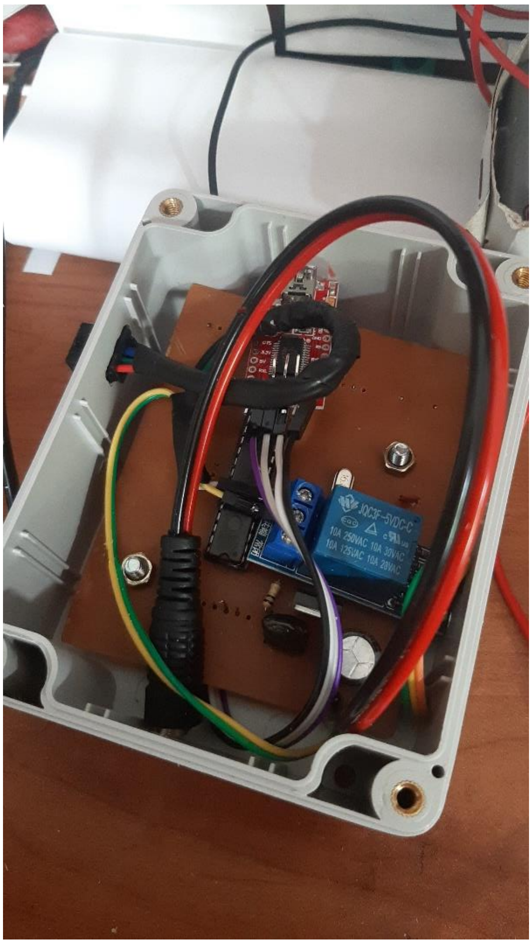


Figure 13Final Prototype

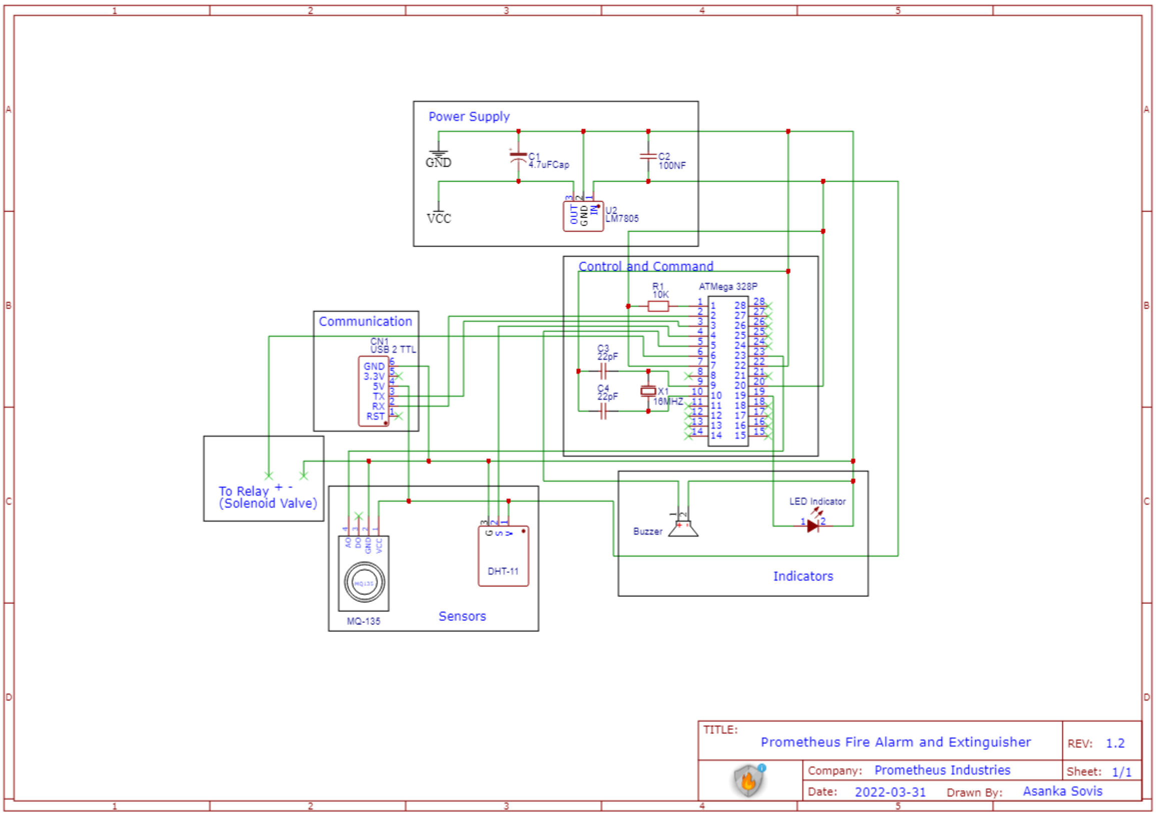


Figure 15Schematic Diagram

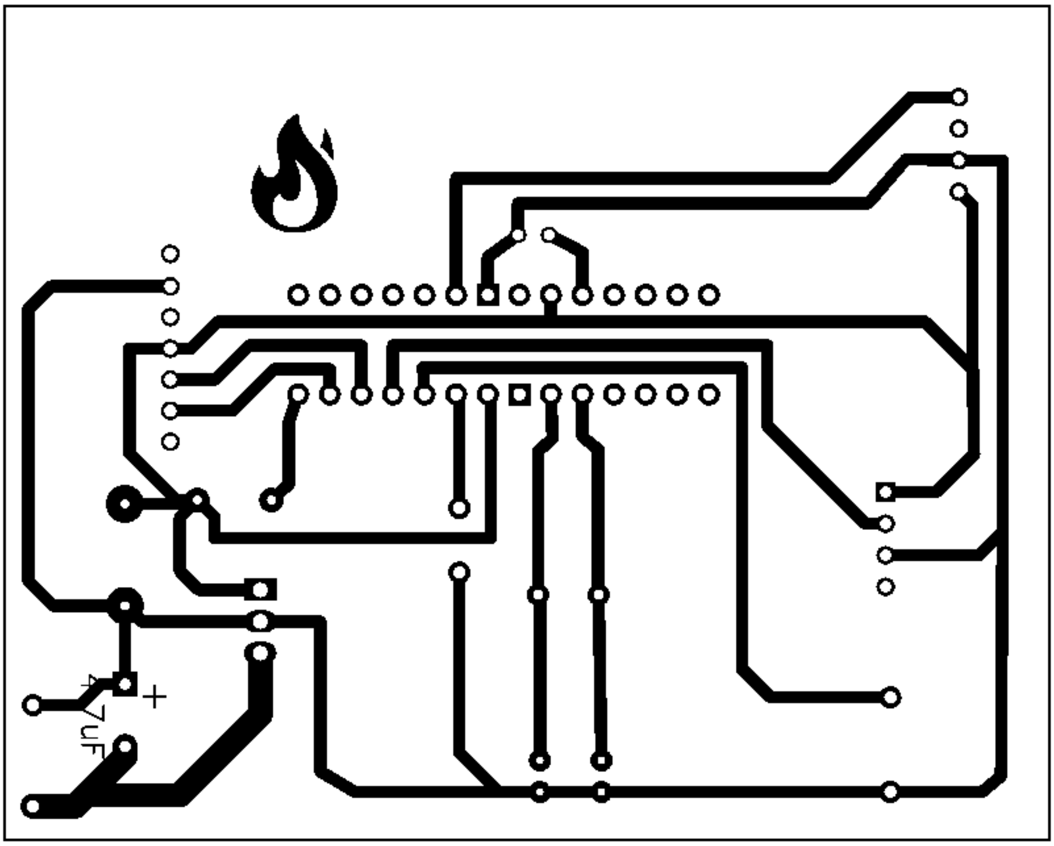


Figure 16PCB Layout

Software

Software for the microcontroller was developed in C++. Specific libraries had to be used that come with the sensors. The overall operation of the device can be summed up by the following flow chart.

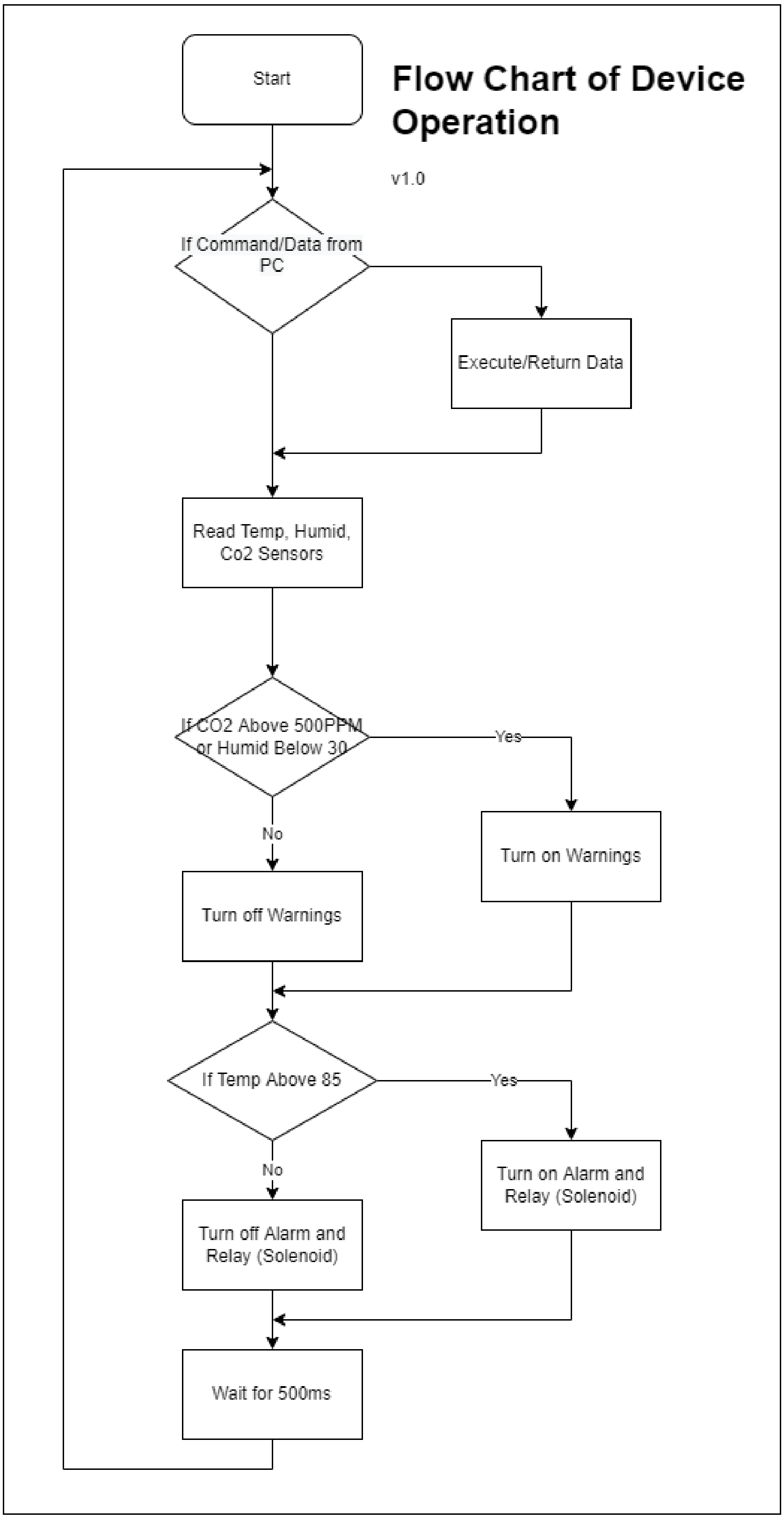


Figure 17Operation Flow Chart

The Control Panel was written in C#. This program uses Windows APIs to connect to the COM ports of the computer to identify Prometheus devices connected.

Specifications

- Connect multiple devices and add alias names to them for easy identification
- Monitor all sensor data from the devices from dedicated control panels
- Notifications for critical events
- Override warnings and control devices right from the PC
- Log sensor readings, visualise and export them
- Automatic reconnecting

Protocol

The protocol was also an important aspect of the device. It allows the device to communicate with the Control Panel. All communication happen via USB between the PC and the FS232 module. The device communicate from its end via the COM ports while the device communicate from its end via the USB to TTL module. The communication is carried out in JSON. The communication protocol can be summed up as follows.

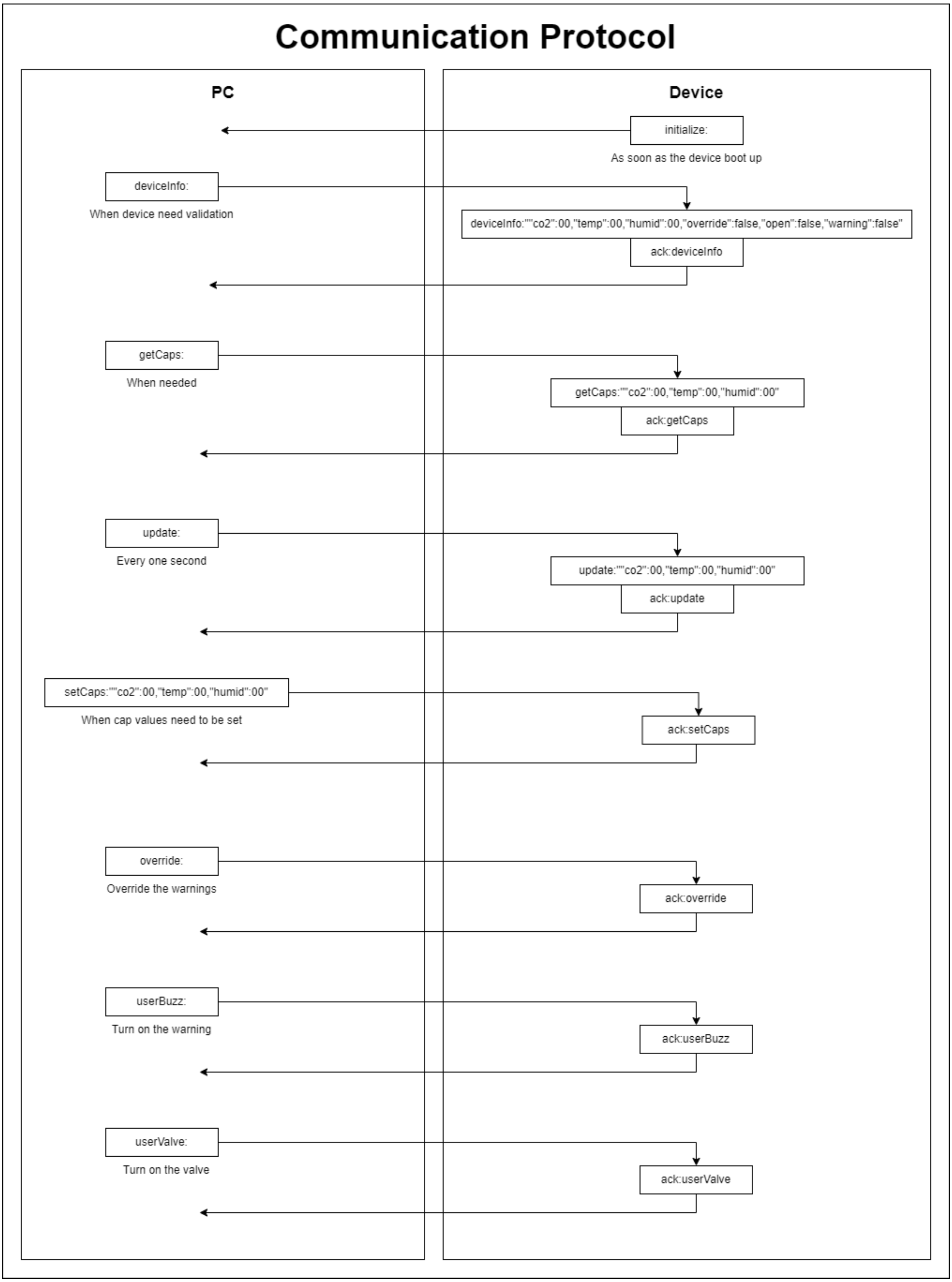


Figure 18Communication Protocol

Challenges

The project was a challenge right from the start. Since we were yet on second semester, we had little idea about how to develop a complete electronic system. On top of this the COVID 19 pandemic came and the project had to be delayed by two years. This means that we had time to get ready for the project.

After this, the biggest hurdle was to source components. With the current situation in the country, it was hard to find the required components in the market; especially the microcontroller. Then it also affected the fabrication of the nozzle part as fabrication workshops give priority to major work due to power cuts.

Other than this, we ran into the following hardware problems:

- Limited knowledge in communication protocols
- Limited knowledge in building an efficient power supply
- Picking the best component available for each task
- It is hard to find proprietary information about fire alarm systems

Hardware problems were followed by software problems:

- Biggest hurdle was the time it takes to build a complete C# software
- Limited knowledge in interfacing with ports
- Debugging takes time for microcontrollers
- Giving the ability to connect multiple devices in parallel to the software

Even after all these challenges, we still managed to complete the task on time. We eliminated most of the hurdles by using online sources such as tutorials and papers. We also built an emulator device so that its easier to debug the software.

However, we still had to bow down to the pandemic and supply issues and reschedule our work accordingly.

Eventually we managed to have complete and working system that operated well even in the demonstration phase. We consider that a big win for our team.

Specifications

Both hardware and software were tested and we came up with the operational specifications for the device as follows.

Hardware

- 12V input
- Minimum 500mA
- Micro USB (If connecting to the Control Panel Software)

Software

- OS: Windows 7 or higher
- Processor: 1GHz or higher
- Dot Net Framework: 4.7 or higher
- RAM: 32MB or higher
- Storage: 16MB or higher

Operational Specifications

- Device issues a warning
    - when carbon dioxide goes above 1000PPM.
    - with humidity drops below 30%.
- warning and turn on the valve at 85°C.

NOTE: These values are set on par with the international guidelines. Please refer to the links in the reference section for additional information

Limitations

Since this is still the first release of our system, there are a number of both identified and unidentified limitations in our system. Some key limitations we identified are as follows.

Hardware

- Transistor of power supply overheat with extended use making the readings unstable (Added a heat sink that extends the time)
- Accuracy of the Carbon dioxide sensor increase with time, thus it is not accurate at first
- As Dr. pointed out, the relays can have back EMFs that can damage the circuit

Software

- Sometimes the application fails to detect the devices connected
- Protocol uses JSON strings which is not efficient
- Protocol doesn't have parity checks in place
- Communication is not encrypted and fool proof
- Minor bugs in the software

Improvements

After reviewing our progress and feedback from the lecturers, we identified few key improvements that we can make to our system.

- Improve the power supply as pointed out by Dr. to use a buck converter instead of the transistor circuit
- Use more accurate hardware
- Implement a better communication protocol
- using the existing one
- Fix bugs in the software

Budget

The budget was based on all the expenses that took place when building the system. This includes all the physical components as well as the fabrication costs.

Component	Price (LKR)
Solenoid Water Valve	900
Single Lane Relay	400
DHT11 Humidity and Temperature Sensor	350
MQ-135 Gas Sensor Module	390
Red LED	50
Buzzer	30
ATmega 328P	3,600
FT232RL USB to TTL Converter	550
12V Power Supply	950
Nozzle and Adapter	2,300
Other components (Capacitors, Transistors, Jumper wires etc.)	600
Total	10,120

Conclusion

We consider the end result a complete success. We managed to hear positive feedback from our lecturers and we also had a working demonstration in the end. This means that we can safely say that we achieved the end goals listed in our proposal. With this we also decided to release our source code to the public. We will also continue to develop the system even further as we believe that having better fire extinguishing systems is important for the fast-moving world.

Additional Resources

We released all the software components to the public domain and the resources can be found from the following links.

- Source code of the device software: https://github.com/asankaSovis/prometheus-fire-alarm
- Source code of the control panel software: https://github.com/asankaSovis/prometheus-command-panel
- Demonstration video: https://youtu.be/5h3k6kufkHo

References

CO2Meter.com. (2021, 5 3). *The Importance of CO2 in Fire Suppression Applications*. Retrieved from CO2Meter.com: https://www.co2meter.com/blogs/news/11417829-co2-fire-supression-systems-inspection-safety

CO2Meter.com. (2022, 4 25). *CO2 Gas Concentration Defined*. Retrieved from CO2Meter.com: https://www.co2meter.com/blogs/news/15164297-co2-gas-concentration-defined

Harris, T. (n.d.). *How Fire Works*. Retrieved 05 29, 2022, from https://science.howstuffworks.com/environmental/earth/geophysics/fire.htm

Various. (2022, 5 28). *Fire detection*. Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Fire_detection

Various. (2022, 1 9). *Fire protection*. Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Fire_protection

Various. (2022, 1 22). *Fire sprinkler system*. Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Fire_sprinkler_system

Various. (2022, 2 25). *Heat detector*. Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Heat_detector

Various. (2022, 5 10). *Smoke detector*. Retrieved from Wikipedia, the free encyclopedia: https://en.wikipedia.org/wiki/Smoke_detector

Katta, V. R., Takahashi, F. U. M. I. A. K. I., & Linteris, G. T. (2003). Numerical investigations of CO2 as fire suppressing agent. In Fire Safety Science: Proceedings of the Seventh International Symposium, International Association for Fire Safety Science (pp. 531-544).

Table of Figures