Exposure of People to Smoke Detector

Final Project for Microcontroller and Microprocessor Lab

Kevin (*1706064694)*
Group 7 – Microcontroller and Microprocessor Lab.
Faculty of Engineering, Universitas Indonesia
Depok, Indonesia

Kaizen Lolo (*1706064681*)
Group 7 – Microcontroller and Microprocessor Lab.
Faculty of Engineering, Universitas Indonesia
Depok, Indonesia

Athira Jasmine Syahira (*1706064630*)
Group 6 – Microcontroller and Microprocessor Lab.
Faculty of Engineering, Universitas Indonesia
Depok, Indonesia

John Winston (*1706064675*)
Group 6 – Microcontroller and Microprocessor Lab.
Faculty of Engineering, Universitas Indonesia
Depok, Indonesia

***Abstract*— The device created is Exposure of People to Smoke Detector. The detector functions to record the amount of smoke present and whether there is a person present. By observing these two factors, the exposure of the smoke towards the people can be determined. Alongside the smoke, the temperature is observed to create relation between the production of smoke to temperature as well. The device utilizes two Arduino UNO R3 and three sensors (PIR motion sensor, MQ135 gas sensor, and DHT11 temperature sensor). From the sensor inputs, a data log is created with time of each observation using Real Time Clock to a SD card for evaluation.**

Keywords*— *Smoke detector, Arduino UNO R3, PIR motion sensor, MQ135 gas sensor, DHT11 temperature sensor, Microcontroller, SD card, Real Time Clock, Arduino Shield XD-204.

I. INTRODUCTION

Smoke is made up of a complex mixture of gases and fine particles produced when wood and other organic materials burn. These fine particles present a threat toward human health. These microscopic particles can penetrate deep into human lungs. Constant exposure to smoke can cause a range of health problems, from burning eyes and a runny nose to aggravated chronic heart and lung diseases. As these risks are present in society, the project created is to observe and create data regarding the exposure of people to smoke by utilizing microcontroller. Specifically, for this project, the source of smoke detected is that from the sate stall in Faculty of Engineering Universitas Indonesia canteen. Observations made would show that the smoke from grilling sate can be directed better for less exposure to people passing by. The source of fire will release both smoke and heat. Detection of temperature is done to help observe the presence of smoke.

II. GENERAL EXPLANATION

This device is made to detect the exposure of smoke to people. This device will also record the date and time of the reading for data log accuracy. The data recorded will be the amount of smoke present (ppm), present of people passing by, temperature of area (°C), and time of data observation (date, time). The data recorded will be saved directly to a micro SD card. The detection of smoke is by MQ135, the detection of people passing by PIR motion sensor, and the detection of temperature is by DHT11. For determining time of observation is utilizing Real Time Clock module, the RTC is combined with the SD card module in Arduino shield XD204.

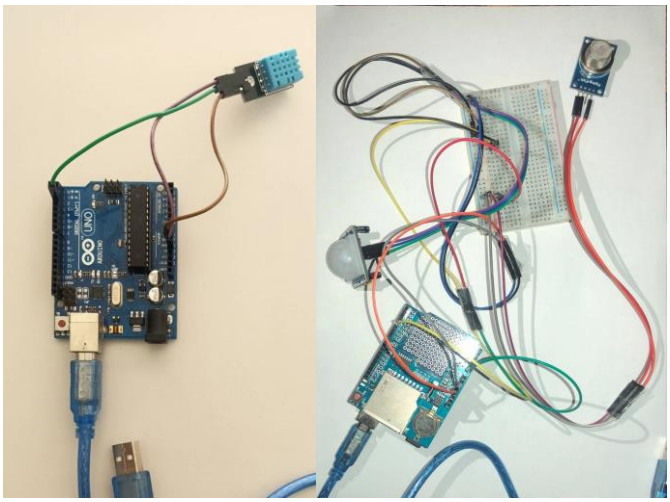
III. TECHNICAL EXPLANATION

A. Design

The device design is to incorporate the different sensors to work together. On the first Arduino UNO R3, it is connected to the Arduino Shield XD-204 as its structure has been made to fit exactly on top of the Arduino. From there the sensors are connected. As the device utilizes 2 sensors: PIR motion and MQ 135, the voltage source 5V should be shared. Therefore, the breadboard functions to ease the distribution of the voltage source and the ground. From the breadboard, the sensors are powered. As for the placement of the input from the sensors, the design created has the analog input of MQ 135 to analog pin 0 of the Arduino + shield. As for the input of the PIR motion, it is set into digital pin 4. And also, due to the Real Time Clock module and the SD card module are part of the Arduino shield, they utilize digital pins 10-13. Aside from the components, the i2c communication utilize analog pin 4 and 5.

On the second Arduino UNO R3, the DHT 11 is connected. The voltage source 5V and ground is connected to the DHT11, and the input of the reading from the sensor is connected to digital pin 2.

From the design made, the result is the following:



B. Code

1. Arduino 1 (Arduino Shield including RTC and SD Card Module, PIR Motion Sensor, and MQ135 Gas Sensor)

```
PROJECT
#include <SPI.h>
#include <SD.h>
#include <Wire.h>
#include "RTClib.h"

// Microcontroller Project Smoke Detector - Group 6 & 7
//By: Kevin, John Winston, Athira Jasmine, Kaizen Lolo O

// how many milliseconds between grabbing data and logging it. 1000 ms is once a second
#define LOG_INTERVAL  1000 // mills between entries (reduce to take more/faster data)

// how many milliseconds before writing the logged data permanently to disk
// set it to the LOG_INTERVAL to write each time (safest)
// set it to 10*LOG_INTERVAL to write all data every 10 datareads, you could lose up to
// the last 10 reads if power is lost but it uses less power and is much faster!
#define SYNC_INTERVAL 1000 // mills between calls to flush() - to write data to the card
uint32_t syncTime = 0; // time of last sync()

#define ECHO_TO_SERIAL   1 // echo data to serial port
#define WAIT_TO_START    0 // Wait for serial input in setup()

// connection of the digital pins, digital pins 10-13 has been used for the shield
#define redLEDpin 2
#define greenLEDpin 3
```

This part is used to include the library that we need on the whole program dan device, such as RTC, SD Card module, etc. It also defines the function for the serial communication and data logging. The Arduino Shield XD 204 (includes RTC and SD card module) has used the digital pins 10-13.

```
PROJECT
int pirSensor = 4;          // the pin that the sensor is atteched to (PIR Sensor)
int ledPIR = 13;            // the pin that the LED is atteched to (PIR Sensor)
int state = LOW;            // by default, no motion detected (PIR Sensor)
int val = 0;                // variable to store the sensor status (value) (PIR Sensor)
int gas135Value;            // declaration of gas sensor MQ135

// connection of the analog pins
//analog pin 4 and 5 has been used for i2c communication
#define BANDGAPREF 14           // special indicator that we want to measure the bandgap
#define aref_voltage 3.3        // we tie 3.3V to ARef and measure it with a multimeter!
#define bandgap_voltage 1.1     // this is not super guaranteed but its not -too- off

RTC_DS1307 RTC; // define the Real Time Clock object

// for the data logging shield, we use digital pin 10 for the SD cs line
const int chipSelect = 10;

// the logging file
File logfile;
```

This part declares the pin used by the external sensor of Arduino Uno R3 such as PIR motion sensor and MQ135 gas sensor. Analog pins 4 and 5 have been used by the Arduino Shield XD204 for i2c communication. The, we used to log the file to be stored in the SD card.

```
PROJECT
void error(char *str)
{
  Serial.print("error: ");
  Serial.println(str);

  // red LED indicates error
  digitalWrite(redLEDpin, HIGH);

  while(1);
}
```

This segment is just used to indicate whether some errors are occurring in the program or not.

```
PROJECT
void setup(void)
{
  Serial.begin(9600);
  Serial.println();

  // use debugging LEDs
  pinMode(redLEDpin, OUTPUT);
  pinMode(greenLEDpin, OUTPUT);
  pinMode(ledPIR, OUTPUT);      // initalize LED as an output (PIR Sensor)
  pinMode(pirSensor, INPUT);    // initialize sensor as an input (PIR Sensor)

  #if WAIT_TO_START
    Serial.println("Type any character to start");
    while (!Serial.available());
  #endif //WAIT_TO_START

  // initialize the SD card
  Serial.print("Initializing SD card...");
  // make sure that the default chip select pin is set to output, even if you don't use it:
  pinMode(10, OUTPUT);

  // see if the card is present and can be initialized:
  if (!SD.begin(chipSelect)) {
    error("Card failed, or not present");
  }
```

This segment defines the input and output of the sensor. It also prints some output to the serial monitor to indicate the initializing process of the SD card.

```
PROJECT
  Serial.println("card initialized.");

  // create a new file
  char filename[] = "LOGGER00.CSV";
  for (uint8_t i = 0; i < 100; i++) {
    filename[6] = i/10 + '0';
    filename[7] = i%10 + '0';

    if (! SD.exists(filename)) {
      // only open a new file if it doesn't exist
      logfile = SD.open(filename, FILE_WRITE);
      break;  // leave the loop!
    }
  }

  if (! logfile) {
    error("couldnt create file");
  }

  Serial.print("Logging to: ");
  Serial.println(filename);
```

This segment creates a file used to store the data of the sensor record. The name of the file is LOGGER00 in the format of .CSV. The '00' character in the name of the file will be incremented if any new file is created.

PROJECT§

```
  // connect to RTC
  Wire.begin();
  RTC.begin();

  if (!RTC.isrunning()) {
    logfile.println("RTC failed");
    RTC.adjust(DateTime(F(__DATE__), F(__TIME__)));

  #if ECHO_TO_SERIAL
    Serial.println("RTC failed");
  #endif  //ECHO_TO_SERIAL
  }

  logfile.println("millis,stamp,datetime,ppm,object detected,vcc");

  #if ECHO_TO_SERIAL
    Serial.println("millis,stamp,datetime,ppm,object detected,vcc");
  #endif //ECHO_TO_SERIAL
}
```

This part is used to run the RTC to obtain the real time and date when the sensor is recording the data. Firstly, we need to adjust the time and data of RTC using the function of RTC.adjust. Then, the data that will be stored in the SD card is millis, stamp, real date and time, the value of PPM recorded, the object detection, and the vcc.

PROJECT§

```
void loop(void)
{
  DateTime now = RTC.now();

  // delay for the amount of time we want between readings
  delay((LOG_INTERVAL -1) - (millis() % LOG_INTERVAL));

  digitalWrite(greenLEDpin, HIGH);

  // log milliseconds since starting
  uint32_t m = millis();
  logfile.print(m);           // milliseconds since start
  logfile.print(", ");

  #if ECHO_TO_SERIAL
    Serial.print(m);          // milliseconds since start
    Serial.print(", ");
  #endif // ECHO_TO_SERIAL
```

This segment is used to start the data logging based on the real time and date. It firstly obtains the data of milis and stamp. The, the data will be logged into SD card and show on the serial monitor.

PROJECT§

```
  // log time
  logfile.print(now.unixtime()); // seconds since 1/1/1970
  logfile.print(", ");
  logfile.print('"');
  logfile.print(now.year(), DEC);
  logfile.print("/");
  logfile.print(now.month(), DEC);
  logfile.print("/");
  logfile.print(now.day(), DEC);
  logfile.print(" ");
  logfile.print(now.hour(), DEC);
  logfile.print(":");
  logfile.print(now.minute(), DEC);
  logfile.print(":");
  logfile.print(now.second(), DEC);
  logfile.print('"');
```

This part used to log the real time and date of the system, including year, month, day, hour, minute, and second into the SD card.

PROJECT§

```
  #if ECHO_TO_SERIAL
    Serial.print(now.unixtime()); // seconds since 1/1/1970
    Serial.print(", ");
    Serial.print('"');
    Serial.print(now.year(), DEC);
    Serial.print("/");
    Serial.print(now.month(), DEC);
    Serial.print("/");
    Serial.print(now.day(), DEC);
    Serial.print(" ");
    Serial.print(now.hour(), DEC);
    Serial.print(":");
    Serial.print(now.minute(), DEC);
    Serial.print(":");
    Serial.print(now.second(), DEC);
    Serial.print('"');
  #endif //ECHO_TO_SERIAL
```

While, this part is used to print the real time & date data onto the serial monitor.

PROJECT§

```
  //GAS MQ135
  gas135Value = analogRead(0);       // read analog input pin 0

  //PIR
  val = digitalRead(pirSensor);   // read sensor value

  if (val == HIGH) {            // check if the sensor is HIGH
    digitalWrite(ledPIR, HIGH);   // turn LED ON
    delay(100);                 // delay 100 milliseconds

    if (state == LOW) {
      state = HIGH;       // update variable state to HIGH
      }
  }

  else {
    digitalWrite(ledPIR, LOW); // turn LED OFF
    delay(200);              // delay 200 milliseconds

    if (state == HIGH){
      state = LOW;       // update variable state to LOW
      }
  }
```

This segment is used to run the external sensor (MQ135 and PIR motion sensor) to start recording the data. The MQ135 gas sensor uses the analog reading while PIR motion sensor uses digital reading.

```
PROJECT §

  #if ECHO_TO_SERIAL
    Serial.print(", ");
    Serial.print(gas135Value, DEC);
    Serial.print(" PPM");
    //delay(100);
    logfile.print(",");
    logfile.print(gas135Value, DEC);

    if (state == HIGH) {
        Serial.print(", Motion detected!");
        }

    else{
        Serial.print(", Motion stopped!");
        }

    logfile.print(", ");
    logfile.print(state);

  #endif //ECHO_TO_SERIAL

  // Log the estimated 'VCC' voltage by measuring the internal 1.1v ref
  analogRead(BANDGAPREF);
  delay(10);
```

This segment used to print the data recorded by MQ135 and PIR motion sensor onto the serial monitor and log those data in the SD card.

```
PROJECT §

int refReading = analogRead(BANDGAPREF);
float supplyvoltage = (bandgap_voltage * 1024) / refReading;

logfile.print(", ");
logfile.print(supplyvoltage);

#if ECHO_TO_SERIAL
  Serial.print(", ");
  Serial.print(supplyvoltage);
#endif // ECHO_TO_SERIAL

logfile.println();

#if ECHO_TO_SERIAL
  Serial.println();
#endif // ECHO_TO_SERIAL

digitalWrite(greenLEDpin, LOW);

// Now we write data to disk! Don't sync too often - requires 2048 bytes of I/O to SD card
// which uses a bunch of power and takes time
if ((millis() - syncTime) < SYNC_INTERVAL) return;
syncTime = millis();
```

This segment is used to print the supply voltage onto the serial monitor and store those data into the SD card.

```
PROJECT §

  // blink LED to show we are syncing data to the card & updating FAT!
  digitalWrite(redLEDpin, HIGH);
  logfile.flush();
  digitalWrite(redLEDpin, LOW);

}
```

Finally, the data has been synced to the SD card and the FAT will be updated.

2. Arduino 2 (DHT11 Temperature Sensor)

```
DHTTester §

#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));

  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);
```

This segment includes the DHT11 library and defines the digital output used by the DHT11 sensor. Firstly, it reads the data of humidity and temperature into the float variable.

```
DHTTester §

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

Then, the program will indicate some output if the sensor reading is error. Otherwise, the data will be computed based on the heat index in Fahrenheit and Celcius. Finally, the data will be previewed onto the serial monitor.

C. *Some Common Mistakes*

Although our device can receive a reliable and valid output/data, minor mistakes were encountered during the making, and testing of this device, such as:

- Output of motion detector can sometimes be delayed.
- Clocking of RTC can sometimes be unstable, where it is faster than the real time.

IV. TOOLS

- Two Arduino UNO R3
- MQ 135 Sensor (smoke CO2 sensor)
- PIR Sensor (Motion Sensor)
- DHT 11 (Temperature Sensor)
- Arduino Shield XD-204 (RTC and SD card module)
- Jumpers
- Breadboard
- Micro SD card and Adaptor

V. BUDGETING & PLANNING

Component	Number	Cost (Rp.)
MQ 135 Sensor	1	35.000
PIR Motion	1	20.000
DHT 11 Sensor	1	18.000
Arduino Shield XD-204	1	35.000
Arduino UNO R3	2	130.000
Jumper	1	15.000

Breadboard	1	20.000
Micro SD Card + Adaptor	1	58.000
Total	Rp.331.000	

Date	Activity
18-22 November	Finding Ideas
24-28 November	Research about the Topics
2-3 December	Buying the components and testing the sensors
4-6 December	Making the project & the coding
9 December	Finishing (code & device)
10 December	Data acquisition

VI. SYSTEM IMPLEMENTATION

This device will be implemented in Faculty of Engineering Universitas Indonesia Canteen near the Sate stall. The smoke from the stall although have been blown by a fan towards one direction, still disturbs people passing by as it is not directed upwards towards the sky. With that being said, the test is conducted from two points (in front and side of the sate stall) to obtain more and valid data. We did a 6 hours data taking which includes the time of high number of people passing by and also the lowest number of people passing to make the data more reliable. We put the PIR motion sensor towards the hallway where people pass by to detect the movement and read it as presents of people and put the MQ135 facing the hallway so as to not intentionally detect all the smoke produced by the stall.

VII. GROUP MEMBER RESPONSIBILITY

Athira Jasmine	Testing the MQ 135 sensor, built the device, field testing
John Winston	Testing the DHT 11 sensor, built the device, combining the code
Kaizen Lolo	Testing the PIR motion sensor, built the device, field testing
Kevin	Testing the shield (RTC & SD card), built the device, combining the code

VIII. SYSTEM/ PROJECT DOCUMENTATION

A. Field Testing

B. Video

This link contains the full video of our project idea, design, tools and equipment, coding, and also the simulation of the device:
https://youtu.be/SQ90DqUSus4

IX. CONCLUSION

This project is a Smoke Detector device that can detect the smoke exposure to people around the Sate stall inside Kantek. We use Arduino to write the coding for our project, and for the device itself, we use MQ135 as the smoke detector, PIR motion sensor as the motion detector, DHT 11 for the temperature sensor, and Arduino Shield which include the RTC and SD card. The data that we gain from the SD card consist of time and date, ppm (particle per million), motion detect, and vcc. This project/device is conducted to fulfil our Final Project for Microprocessor and Microcontroller class and laboratory

REFERENCE

- Arduino, “PIR Motion Sensor: How to Use PIRs w/ Arduino & Raspberry Pi”, create.arduino.cc [Online]. Available: https://create.arduino.cc/projecthub/electropeak/pir-motion-sensor-how-to-use-pirs-w-arduino-raspberry-pi-18d7fa [Accessed on 5 December 2019]
- Dejan, “Arduino SD Card and Data Logging to Excel Tutorial”, howtomechatronics.com [Online]. Available: https://howtomechatronics.com/tutorials/arduino/arduino-sd card-data-logging-excel-tutorial/ [Accessed on 5 December 2019]
- “Membuat Alat Monitor Gas Polutan Udara dan Debu Serta Suhu - Kelembaban Menggunakan DHT22 sensor MQ135 MQ7 dan data logger SDcard dilengkapi RTC ds3231”, Website and Blog from kursuselektronikaku.blogspot.com [Online]. Available: http://kursuselektronikaku.blogspot.com/2018/05/membuat-alat-monitor-gas-polutan-udara.html [Accessed on 4 December 2019]
- Aswinth Raj, “Arduino Data Logger (Log Temperature, Humidity, Time on SD Card and Computer)”, Circuit Digest [Online]. Available: https://circuitdigest.com/microcontroller-projects/arduino-data-logger-project [Accessed on 4 December 2019]
- RandomNerdTutorials.com [Online] Available: https://randomnerdtutorials.com/arduino-temperature-data-logger-with-sd-card-module/ [Accessed on 4 December 2019]
- etechnophiles.com [Online]. Available: https://etechnophiles.com/beginners-guide-to-ir-sensor-and-how-to-use-it-with-arduino/ [Accessed on 28 November 2019]