



**CYBER-PHYSICAL SYSTEM FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA**

Air Quality Monitor

GROUP B-11

Andikha Wisanggeni	2106731503
Stefan Agusto Hutapea	2106700744
Lauren Christy Tanudjaja	2106707870
Jeffri	2106705070

PREFACE

Puji dan syukur kita panjatkan kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmatnya sehingga kami sebagai kelompok B-11 dapat menyelesaikan tugas akhir Praktikum Sistem Siber Fisik yang berjudul “Air Quality Monitor”.

Seiring perkembangan zaman, teknologi pun juga akan semakin berkembang dan tidak bisa terlepas dari kehidupan sehari-hari. Hal ini dapat dilihat mulai dari kita bangun di pagi hari hingga nanti kita tidur di malam hari, kita tidak akan bisa lepas dari teknologi. Dengan membuat proyek ini, kami berharap kami dapat mengimplementasikan modul-modul yang telah kami pelajari pada modul-modul sebelumnya, dan juga proyek ini dapat bermanfaat bagi masyarakat sekitar.

Kami sebagai kelompok B-11 ingin mengucapkan terima kasih kepada seluruh rekan kelompok kami yang telah bekerja sama dengan baik dalam mencari ide, membuat rangkaian, menulis kode, hingga menyusun laporan. Berkat kerja sama ini, proyek kami dapat diselesaikan tepat waktu dan memberikan hasil yang memuaskan. Kami juga ingin menyampaikan rasa terima kasih kami kepada asisten laboratorium yang telah membimbing kami selama praktikum. Kami juga ingin berterima kasih kepada bang Michael Harditya yang telah memberikan saran untuk proyek kami.

Depok, May 23, 2023

Group B-11

TABLE OF CONTENTS

CHAPTER 1.....	4
INTRODUCTION.....	4
1.1 PROBLEM STATEMENT.....	4
1.2 PROPOSED SOLUTION.....	4
1.3 ACCEPTANCE CRITERIA.....	5
1.4 ROLES AND RESPONSIBILITIES.....	5
1.5 TIMELINE AND MILESTONES.....	6
CHAPTER 2.....	8
IMPLEMENTATION.....	8
2.1 HARDWARE DESIGN AND SCHEMATIC.....	8
2.2 SOFTWARE DEVELOPMENT.....	9
2.3 HARDWARE AND SOFTWARE INTEGRATION.....	10
CHAPTER 3.....	11
TESTING AND EVALUATION.....	11
3.1 TESTING.....	11
3.2 RESULT.....	12
3.3 EVALUATION.....	12
CHAPTER 4.....	13
CONCLUSION.....	13

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Polusi udara merupakan masalah yang serius yang sedang dihadapi oleh masyarakat global saat ini. Polusi udara akan menurunkan kualitas dari udara. Kualitas udara yang buruk dapat berdampak negatif pada kesehatan manusia, seperti gangguan pernapasan, alergi, dan juga penyakit-penyakit pernapasan yang lainnya. Tidak hanya itu, polusi udara juga dapat memiliki dampak negatif bagi lingkungan sekitar kita, seperti degradasi ekosistem dan perubahan iklim. Polusi udara dapat disebabkan oleh berbagai macam faktor, seperti kendaraan bermotor, asap industry, dan pembakaran biomassa. Sudah banyak upaya yang dilakukan baik dari sisi pemerintah maupun dari masyarakat, namun masih banyak orang yang kesulitan untuk memantau kualitas udara di sekitar mereka secara akurat dan efisien.

Polusi udara ini biasa terjadi di kota-kota besar yang ada di seluruh dunia, seperti Jakarta, Beijing, Busan, Hong Kong, Shanghai, dan lain-lain. Kota-kota tersebut memiliki tingkat polusi udara yang sangat tinggi. Seperti yang telah dijelaskan sebelumnya, bahwa polusi udara akan menyebabkan banyak sekali dampak negatif. Namun, para warga yang ada di kota-kota tersebut sering kali tidak memiliki akses yang memadai terhadap informasi mengenai kualitas udara yang ada di sekitar mereka, sehingga mereka sulit untuk mengambil tindakan yang diperlukan untuk melindungi diri mereka sendiri dan keluarga mereka. Keadaan ini mengarah pada perlunya solusi yang dapat membantu masyarakat untuk memantau kualitas udara di lingkungan mereka dengan mudah dan efektif. Di dunia yang semakin berkembang ini, teknologi akan terus berkembang dan inovasi akan semakin menjanjikan. Namun, kebanyakan dari alat pemantauan kualitas udara yang sudah ada masih cukup mahal dan sulit digunakan, terutama para orang awam.

1.2 PROPOSED SOLUTION

Untuk mengatasi masalah ini, solusi yang kelompok kami usulkan adalah dengan membuat “Air Quality Monitor” yang akan membantu masyarakat untuk memantau kualitas udara di lingkungan mereka dengan mudah dan akurat. Alat ini akan menggunakan Arduino

dan akan menggunakan bahasa pemrograman assembly. Alat ini juga akan menggunakan beberapa sensor, seperti akan ada sensor kelembapan yang akan diukur dengan menggunakan DHT11 dan juga akan ada MQ-series yang akan digunakan untuk mengukur parameter dari kualitas udara, seperti tingkat polutan udara. Data yang didapatkan, seperti informasi dari kualitas udara, nantinya akan ditampilkan secara real-time pada layer LCD yang telah terintegrasi. Kemudian alat akan memberikan peringatan melalui buzzer atau LED ketika kualitas udara melebihi ambang batas yang telah ditentukan kepada pengguna, dengan begitu masyarakat dapat mengambil tindakan untuk menjaga kualitas udara yang ada di lingkungan. Data yang didapatkan dari monitor tentang kualitas udara, juga akan dapat membantu masyarakat untuk melakukan perubahan dalam kebiasaan sehari-hari mereka untuk beradaptasi dengan udara yang ada, seperti menghindari aktivitas di luar ruangan saat tingkat polusi tinggi, meningkatkan ventilasi yang ada di rumah, atau juga dapat mengambil langkah-langkah untuk mengurangi polusi yang ada di sekitar mereka.

Alat ini akan sangat diperlukan di berbagai tempat, terutama di kota-kota besar di seluruh dunia yang memiliki kualitas udara yang buruk. Dengan begitu masyarakat yang ada di kota akan dapat mengetahui kualitas udara yang ada di lingkungan mereka.

Melalui solusi ini, kami harap alat yang kami buat dapat memberikan manfaat yang signifikan bagi penggunanya, terutama dalam memantau kualitas udara di lingkungan. Dengan begitu, masyarakat akan dapat melakukan tindakan preventif dari polusi udara atau tindakan yang dapat mengurangi polusi udara. Dengan harga yang terjangkau dan penggunaan teknologi Arduino yang popular, kami harap solusi ini nantinya juga dapat digunakan oleh banyak orang dan dapat memberikan manfaat yang signifikan bagi masyarakat.

1.3 ACCEPTANCE CRITERIA

Air Quality Monitor dapat berjalan apabila memenuhi syarat-syarat berikut yaitu:

1. Sensor DHT11 dan MQ2 dapat membaca suhu, kelembapan, dan gas dari lingkungan dan ditampilkan oleh Arduino di MAX7219 dan serial monitor. Pengiriman data dan sinyal merupakan bagian dari communication protocol sehingga data dapat dibaca dan dikirimkan satu sama lain. Suhu dan kelembapan akan tampil di MAX7219 dan gas di serial monitor

- Buzzer akan menyala apabila kadar aman gas yang ada di lingkungan melewati kadar aman dari yang ada di dalam potongan kode.

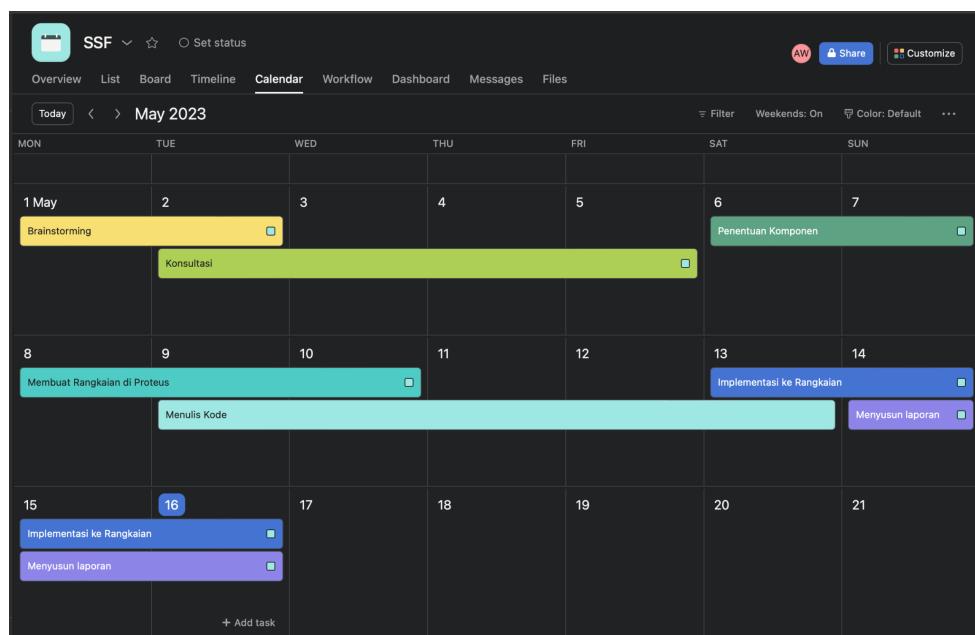
1.4 ROLES AND RESPONSIBILITIES

The roles and responsibilities assigned to the group members are as follows:

Roles	Responsibilities	Person
Menyusun Laporan	Membuat dan mengedit laporan	Andikha Wisanggeni
DHT11 Programmer	Membuat kode untuk DHT11	Stefan Agusto Hutapea
MQ2 Programmer	Membuat kode untuk MQ2	Lauren Christy Tanudjaja
Proteus Designer	Membuat desain dari rangkaian di software Proteus	Jeffri

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES



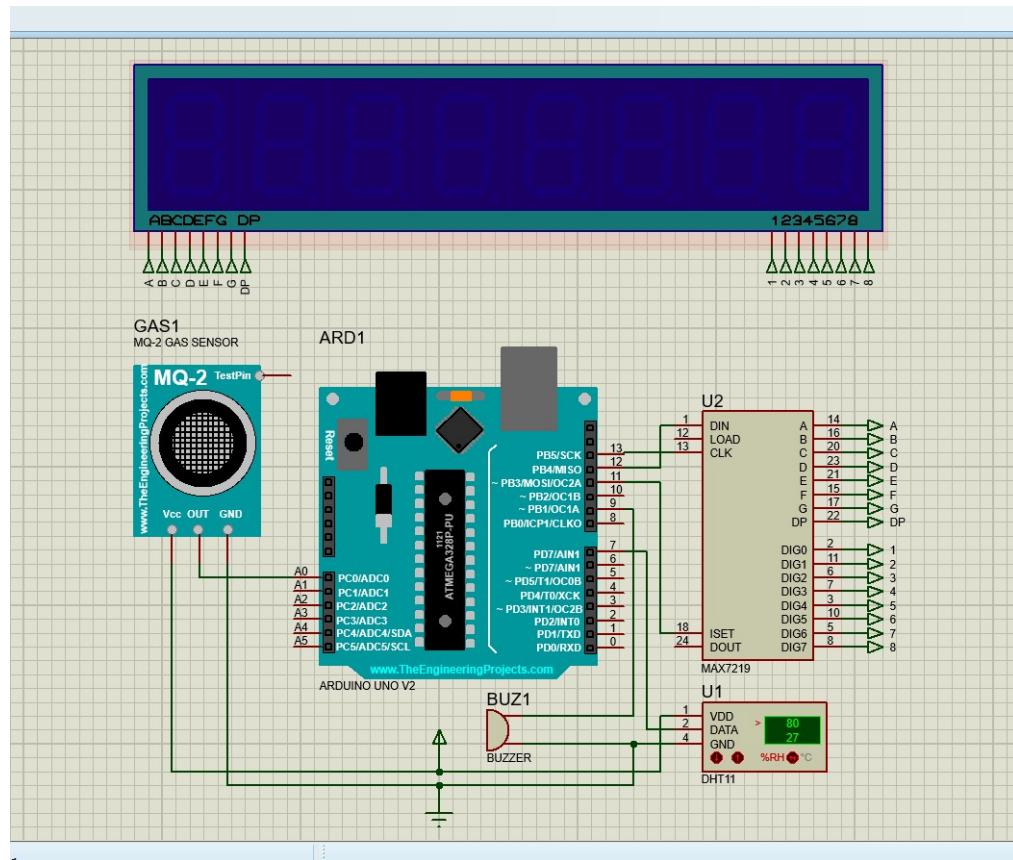
Gambar 1. Gantt Chart

- a) Brainstorming: Mencari ide untuk judul, deskripsi dari proyek akhir, dan menentukan modul-modul yang akan digunakan.
- b) Konsultasi: Bertanya kepada asisten laboratorium tentang ide yang dimiliki.
- c) Penentuan komponen: Menentukan komponen-komponen yang diperlukan untuk membuat rangkaian.
- d) Membuat rangkaian di proteus: Membuat rangkaian di software sebelum membuatnya dengan menggunakan komponen asli.
- e) Menulis kode: Menulis kode untuk dimasukkan ke arduino nantinya, dan kode akan dalam bahasa assembly.
- f) Implementasi ke rangkaian: Merangkai komponen yang telah dibeli, dan menyambungkannya dengan kode assembly yang telah dibuat.
- g) Menyusun laporan: Mengambil data yang didapatkan dan menulisnya di laporan.

CHAPTER 2

IMPLEMENTATION

2.1 HARDWARE DESIGN AND SCHEMATIC



Gambar 2. Rangkaian Proteus

Terdapat beberapa komponen, seperti MQ-2 Gas Sensor, arduino uno, MAX7219, DHT11, TMB12A05, dan kabel jumper. Berikut penjelasan untuk setiap komponen:

a. MQ-2 Gas Sensor

Merupakan sensor gas yang akan mendeteksi gas, seperti propana, metana, karbon monoksida, dan lain-lain. MQ-2 akan mempunyai dua pin utama, yaitu VCC sebagai sumber daya dan analog output voltage yang akan memberikan output berkaitan dengan konsentrasi gas yang terdeteksi.

b. Arduino Uno

Merupakan board microcontroller yang akan menjadi otak dari rangkaian. Microcontroller yang ada di board ini adalah ATmega328P. Arduino nanti akan memproses input dari sensor dan akan mengontrol output di MAX7219.

c. MAX7219

Merupakan sebuah chip yang dapat mengendalikan hingga 64 LED secara individual atau matrix LED 8 x 8. MAX7219 akan menampilkan nilai suhu dan kelembapan.

d. DHT11

Merupakan sensor untuk mengukur suhu dan kelembapan di lingkungan. Kemudian akan diubah menjadi output digital, yang nanti akan dikirim ke arduino uno, dan akan dibaca oleh MAX7219 yang nanti akan menampilkan nilai suhu dan kelembapan.

e. TMB12A05

Merupakan buzzer yang nanti akan berbunyi pada saat suhu dan kelembapan telah melewati nilai maksimum atau nilai tertentu. Jadi buzzer akan berfungsi sebagai alat yang memberikan peringatan berupa suara.

f. Kabel jumper

Merupakan kabel yang akan menyambungkan antar komponen baik di breadboard atau langsung. Jadi ujung kabel akan menghubungkan antar pin atau port yang ada di komponen.

2.2 SOFTWARE DEVELOPMENT

Source Code:

```
#define __SFR_OFFSET 0x00
#include "avr/io.h"
;-----
.global init_ADC
.global init_serial
.global print_ADC
.global main
;=====
main:
    RCALL init_ADC
    RCALL init_serial
    RCALL print_ADC
    RCALL SPI_MAX7219_init
;
loop:
    RCALL MAX7219_disp_text
    RCALL DHT11_sensor
;
l1: SBRS DDRB, 1
    SBI DDRB, 1           ;set pin PB1 as o/p for buzzer
    RJMP main
;
DHT11_sensor:
;-----
agn:RCALL delay_2s      ;wait 2s for DHT11 to get ready
;-----
;start_signal
;-----
    SBI DDRD, 7          ;pin PD7 as o/p
    CBI PORTD, 7         ;first, send low pulse
    RCALL delay_20ms     ;for 20ms
    SBI PORTD, 7         ;then send high pulse
;
;response signal
;-----
    CBI DDRD, 7          ;pin PD7 as i/p
w1: SBIC PIND, 7
    RJMP w1              ;wait for DHT11 low pulse
w2: SBIS PIND, 7
    RJMP w2              ;wait for DHT11 high pulse
w3: SBIC PIND, 7
    RJMP w3              ;wait for DHT11 low pulse
;
    RCALL DHT11_reading ;read humidity (1st byte of 40-bit data)
    MOV R25, R24
    RCALL DHT11_reading
    RCALL DHT11_reading ;read temp (3rd byte of 40-bit data)
;
;convert temp & humidity bytes to decimal & display on MAX7219
;
    MOV R28, R24
    LDI R29, 0x07
    LDI R30, 0x06
;
    RCALL binary2decimal ;temp in decimal
;
    MOV R28, R25
    LDI R29, 0x02
    LDI R30, 0x01
;
    RCALL binary2decimal ;humidity in decimal
```

```

;-----
;----- RJMP 11           ;go back & get another sensor reading
;=====

DHT11_reading:
    LDI   R16, 8          ;set counter for receiving 8 bits
    CLR   R24              ;clear data register
    ;
w4: SBIS PIND, 7
    RJMP w4                ;detect data bit (high pulse)
    RCALL delay_timer0    ;wait 50us & then check bit value
    ;
    SBIS PIND, 7          ;if received bit=1, skip next inst
    RJMP skp               ;else, received bit=0, jump to skp
    SEC
    ROL   R24              ;shift in 1 into LSB data register
    RJMP w5                ;jump & wait for low pulse
    skp:LSL   R24          ;shift in 0 into LSB data register
    ;
w5: SBIC PIND, 7
    RJMP w5                ;wait for DHT11 low pulse
    ;
    DEC   R16              ;decrement counter
    BRNE w4                ;go back & detect next bit
    RET
    ;return to calling subroutine
;=====

;delay subroutines
;=====

delay_20ms:           ;delay 20ms
    LDI   R21, 255
13: LDI   R22, 210
14: LDI   R23, 2
15: DEC   R23
    BRNE 15
    DEC   R22
    BRNE 14
    DEC   R21
    BRNE 13
    RET
    ;
delay_2s:             ;delay 2s
    LDI   R21, 255
16: LDI   R22, 255
17: LDI   R23, 164
18: DEC   R23
    BRNE 18
    DEC   R22
    BRNE 17
    DEC   R21
    BRNE 16
    RET
    ;
delay_timer0:         ;50 usec delay via Timer 0
    ;
    CLR   R20
    OUT   TCNT0, R20        ;initialize timer0 with count=0
    LDI   R20, 100
    OUT   OCR0A, R20        ;OCR0 = 100
    LDI   R20, 0b00001010
    OUT   TCCR0B, R20        ;timer0: CTC mode, prescaler 64
    ;
12: IN    R20, TIFR0        ;get TIFR0 byte & check
    SBRS R20, OCF0A        ;if OCF0=1, skip next instruction
    RJMP 12                ;else, loop back & check OCF0 flag
    ;
    CLR   R20
    OUT   TCCR0B, R20        ;stop timer0
    ;
    LDI   R20, (1<<OCF0A)

```

```

        OUT    TIFR0, R20      ;clear OCF0 flag
        RET
;=====
;MAX7219 subroutines
;=====
SPI_MAX7219_init:
;-----
.equ   SCK, 5
.equ   MOSI, 3
.equ   SS, 2
;-----
LDI    R17, (1<<MOSI) | (1<<SCK) | (1<<SS)
OUT   DDRB, R17          ;set MOSI, SCK, SS as o/p
;-----
LDI    R17, (1<<SPE) | (1<<MSTR) | (1<<SPR0)
OUT   SPCR, R17          ;enable SPI as master, fsck=fosc/16
;-----
LDI    R17, 0x0A          ;set segment intensity (0 to 15)
LDI    R18, 8              ;intensity level = 8
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x09          ;set decoding mode command
LDI    R18, 0b01100011    ;decoding byte
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x0B          ;set scan limit command
LDI    R18, 0x07          ;8 digits connected to MAX7219
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x0C          ;set turn ON/OFF command
LDI    R18, 0x01          ;turn ON MAX7219
RCALL send_bytes          ;send command & data to MAX7219
;-----
RET
;=====
MAX7219_disp_text:
;-----
LDI    R17, 0x08          ;select digit 7
LDI    R18, 0x0F          ;data = t
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x05          ;select digit 4
LDI    R18, 0x4E          ;data = C
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x04          ;select digit 3
LDI    R18, 0x00          ;data = space
RCALL send_bytes          ;send command & data to MAX7219
;-----
LDI    R17, 0x03          ;select digit 2
LDI    R18, 0x17          ;data = h
RCALL send_bytes          ;send command & data to MAX7219
;-----
RET
;=====
send_bytes:
CBI    PORTB, SS          ;enable slave device MAX7219
OUT   SPDR, R17          ;transmit command
;-----
112: IN    R19, SPSR
SBRS  R19, SPIF           ;wait for byte transmission
RJMP  112                ;to complete
;-----
OUT   SPDR, R18          ;transmit data
;-----
113: IN    R19, SPSR
SBRS  R19, SPIF           ;wait for byte transmission
RJMP  113                ;to complete

```

```

;-----  

SBI PORTB, SS      ;disable slave device MAX7219  

RET  

;=====  

binary2decimal:  

;-----  

CLR R26            ;set counter1, initial value 0  

CLR R27            ;set counter2, initial value 0  

;  

170: CPI R28, 100   ;compare R28 with 100  

Ret: BRMI 180       ;jump when R28 < 100  

INC R26            ;increment counter1 by 1  

SUBI R28, 100       ;R28 = R28 - 100  

RJMP 170  

;  

180: CPI R28, 10    ;compare R28 with 10  

BRMI dsp           ;jump when R28 < 10  

INC R27            ;increment counter2 by 1  

SUBI R28, 10         ;R28 = R28 - 10  

RJMP 180  

;  

dsp: MOV R18, R27  

MOV R17, R29        ;select digit  

RCALL send_bytes    ;send command & data to MAX7219  

;  

MOV R18, R28  

MOV R17, R30        ;select digit  

RCALL send_bytes    ;send command & data to MAX7219  

;  

RET  

;  

init_ADC:  

CBI DDRC, 0          ;set pin PC0 as i/p for ADC0  

LDI R20, 0xC0          ;internal 2.56V, right-justified data, ADC0  

LDI R21, 0xFF          ;Memasukkan immediate value 0xFF ke register R20.  

OUT DDRB, R21          ; Set port D menjadi output untuk low-byte  

STS ADMUX, R20  

LDI R20, 0x87          ;enable ADC, ADC prescaler CLK/128  

STS ADCSRA, R20  

RET  

;  

init_serial:  

CLR R24  

STS UCSR0A, R24        ;clear UCSR0A register  

STS UBRR0H, R24        ;clear UBRR0H register  

LDI R24, 103            ;& store in UBRR0L 103  

STS UBRR0L, R24        ;to set baud rate 9600  

LDI R24, 1<<RXEN0 | 1<<TXEN0  

STS UCSR0B, R24        ;enable RXB & TXB  

LDI R24, 1<<UCSZ00 | 1<<UCSZ01  

STS UCSR0C, R24        ;asynch, no parity, 1 stop, 8 bits  

RET  

;  

;  

print_ADC:  

LDI R23, 48            ;constants used to get ASCII values  

LDI R24, 7              ;for chars 0-->9 & A-->F  

;  

LDI R20, 0xC7          ;set ADSC in ADCSRA to start conversion  

STS ADCSRA, R20  

;  

wait_ADC:  

LDS R21, ADCSRA        ;check ADIF flag in ADCSRA  

SBRS R21, 4              ;skip jump when conversion is done (flag set)  

RJMP wait_ADC          ;loop until ADIF flag is set  

;  

LDI R17, 0xD7          ;set ADIF flag again  

STS ADCSRA, R17          ;so that controller clears ADIF

```

```

;-----  

LDS  R16, ADCL      ;get low-byte result from ADCL  

LDS  R25, ADCH      ;get high-byte result from ADCH  

;  

;  

ADD  R25, R23      ;add 48 to byte to get ASCII char 0 to 9  

;  

19: LDS  R17, UCSR0A  

    SBRS R17, UDRE0      ;test data buffer if data can be sent  

    RJMP 19  

;  

STS  UDR0, R25      ;print ADC MSD on serial monitor  

;  

PUSH R16            ;store copy of ADCH in STACK register  

ANDI R16, 0xF0      ;mask & extract high-nibble  

SWAP R16            ;swap high-nibble with low-nibble  

ADD  R16, R23      ;add 48 to byte to get ASCII char 0 to 9  

MOV  R28, R16      ;store a copy of byte in R28  

SUBI R28, 58       ;subtract 58 from R28  

BRPL A_F_MSD       ;jump if result is +ve  

;  

110: LDS  R17, UCSR0A  

    SBRS R17, UDRE0      ;test data buffer if data can be sent  

    RJMP 110  

;  

STS  UDR0, R16      ;print ADC mid digit on serial monitor  

MOV  R19, R16  

;  

POP  R16            ;restore ADCH value from STACK register  

ANDI R16, 0x0F      ;mask & extract low-nibble  

ADD  R16, R23  

MOV  R28, R16  

SUBI R28, 58  

BRPL A_F_LSD  

;  

111: LDS  R17, UCSR0A  

    SBRS R17, UDRE0      ;test data buffer if data can be sent  

    RJMP 111  

;  

STS  UDR0, R16      ;print ADC LSD on serial monitor  

;  

112: LDS  R17, UCSR0A  

    SBRS R17, UDRE0      ;test data buffer if data can be sent  

    RJMP 112  

;  

LDI  R18, 0x0A  

STS  UDR0, R18      ;print newline on serial monitor  

;  

113: LDS  R17, UCSR0A  

    SBRS R17, UDRE0      ;test data buffer if data can be sent  

    RJMP 113  

;  

LDI  R18, 0x0D  

STS  UDR0, R18      ;print carriage return on serial monitor  

;  

CPI  R19, 70        ; Compare immediate hasil high-byte dengan 70  

BRLO kaga          ; Branch ke 'mati' jika R16 lebih rendah dari 70  

CPI  R19, 70        ; Compare immediate hasil high-byte dengan 70  

    BRSH bunyi         ; Branch ke 'bunyi' jika R16 lebih tinggi dari atau  

sama dengan 70  

117:RCALL delay_sec ;1 second delay  

    RET  

;=====  

A_F_MSD:  

    ADD  R16, R24      ;add 7 to byte to get ASCII chars A to F  

    RJMP 110  

;  

A_F_LSD:  


```

```

ADD  R16, R24          ;add 7 to byte to get ASCII chars A to F
RJMP 111
;-----
delay_sec:           ;1s delay
    LDI  R20, 255
114: LDI  R21, 255
115: LDI  R22, 80
116: DEC  R22
    BRNE 116
    DEC  R21
    BRNE 115
    DEC  R20
    BRNE 114
    RET
;-----
bunyi:
    SBIS PORTB, 1
    SBI PORTB, 1
    RJMP 117
;-----
kaga:
    CBI PORTB, 1
    RJMP 117

```

Kode assembly yang diberikan adalah program untuk mikrokontroler berbasis arsitektur AVR, yang ditulis dalam bahasa assembly. Program ini memiliki beberapa fungsi yang terdapat di dalamnya, yaitu:

- **init_ADC:** Fungsi ini digunakan untuk menginisialisasi ADC (Analog-to-Digital Converter) pada mikrokontroler. ADC digunakan untuk mengubah sinyal analog menjadi sinyal digital. Fungsi ini mengatur pengaturan ADC seperti tegangan referensi dan preskaler ADC.
- **init_serial:** Fungsi ini digunakan untuk menginisialisasi komunikasi serial pada mikrokontroler. Fungsi ini mengatur pengaturan untuk mengirim dan menerima data melalui modul komunikasi serial.
- **print_ADC:** Fungsi ini digunakan untuk membaca nilai hasil konversi dari ADC dan mencetak nilainya ke monitor serial. Fungsi ini melakukan konversi dari nilai ADC ke format ASCII dan mengirimkannya melalui komunikasi serial.
- **DHT11_sensor:** Fungsi ini digunakan untuk membaca data dari sensor DHT11 yang mengukur suhu dan kelembaban. Fungsi ini mengatur pin mikrokontroler untuk mengirim sinyal ke sensor dan membaca responsnya.
- **MAX7219_disp_text dan send_bytes:** Fungsi-fungsi ini digunakan untuk mengirim data ke modul display MAX7219. Fungsi MAX7219_disp_text mengatur teks yang

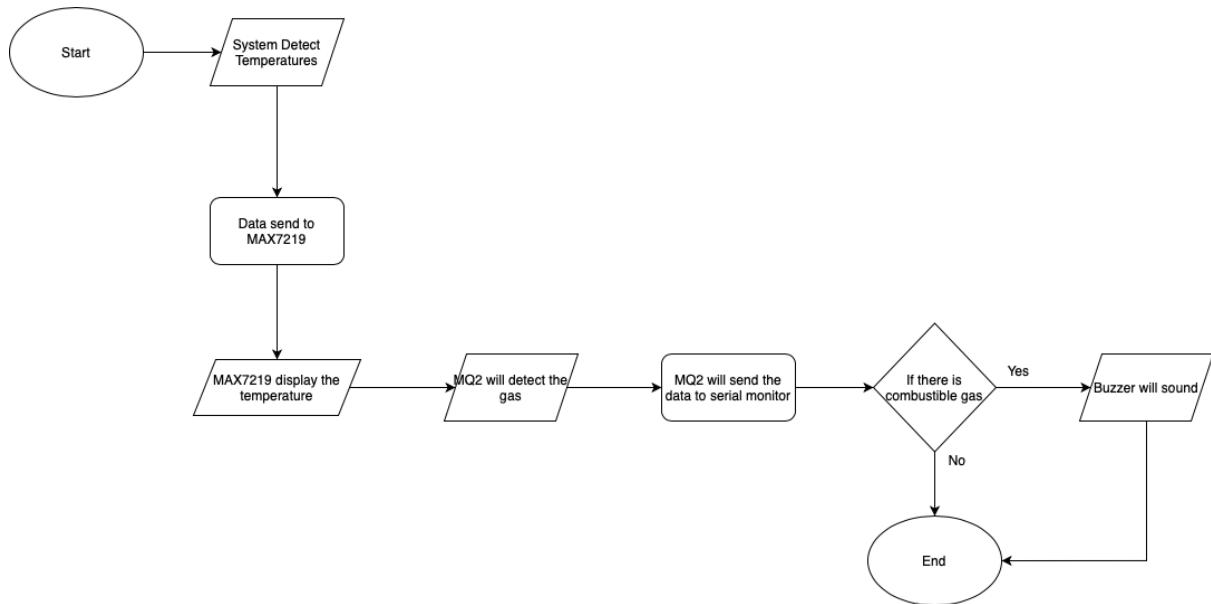
akan ditampilkan pada display, sedangkan fungsi send_bytes mengirim data ke modul display melalui SPI (Serial Peripheral Interface).

- **delay_20ms dan delay_2s:** Fungsi-fungsi ini digunakan untuk memberikan jeda waktu dalam program. delay_20ms memberikan jeda waktu sebesar 20 milidetik, sedangkan delay_2s memberikan jeda waktu sebesar 2 detik.
- **binary2decimal:** Fungsi ini digunakan untuk mengonversi bilangan biner menjadi bilangan desimal. Fungsi ini mengambil dua byte dari register (R28 dan R29) yang berisi bilangan biner dan mengonversinya menjadi bilangan desimal.

Selain fungsi-fungsi tersebut, terdapat juga label dan instruksi-instruksi lain yang mengatur alur program, seperti RCALL untuk pemanggilan fungsi, RJMP untuk melakukan lompatan ke label tertentu, BRNE untuk melakukan lompatan jika flag register tidak nol, dan lainnya.

Secara keseluruhan, program ini mengatur beberapa fitur pada mikrokontroler, termasuk ADC, komunikasi serial, pengendalian modul display MAX7219, dan pengambilan data dari sensor DHT11.

Flowchart:



Gambar 3. Flowchart

Rangkaian akan dimulai dengan sensor DHT11 akan mendetect suhu dan kelembapan yang ada di lingkungan. Kemudian data tentang suhu dan kelembapan akan dikirim ke MAX7219. MAX7219 akan menampilkan hasilnya dengan bentuk seven segment. Lalu sensor MQ2 akan mendeteksi gas dan akan menampilkannya ke serial monitor. Pada saat dicek apabila terdeteksi gas yang mudah terbakar maka buzzer akan berbunyi dan kemudian program selesai. Namun apabila tidak ada gas yang mudah terbakar terdeteksi, maka buzzer tidak akan berbunyi dan program akan selesai.

2.3 HARDWARE AND SOFTWARE INTEGRATION

Jika kita melihat dari proteus, MAX7219 akan dihubungkan dengan Arduino Uno, yaitu pada DIN akan disambungkan PB4, kemudian CLK akan tersambung dengan PB5, dan ISET akan tersambung dengan PD7. Output pada MAX7219 akan berdasarkan logic dari seven segment. DHT11 akan terhubung dengan arduino uno, yaitu pada VDD akan terhubung dengan VCC, DATA akan terhubung dengan PD7, dan GND akan terhubung dengan Ground. Buzzer akan terhubung dengan Ground dan juga PD7 pada arduino uno. MQ2 akan terhubung sesuai dengan portnya, yaitu VCC dengan VCC, GND dengan Ground, dan OUT akan terhubung dengan PC0 pada arduino uno.

Untuk software digunakan Arduino IDE yang mengimplementasikan bahasa Assembly untuk mengeksekusi Air Quality Monitor ini. Pengetikan dan pengiriman program ke board dapat dilakukan menggunakan Arduino Integrated Development Environment yang sudah memuat library yang diperlukan oleh Arduino secara tertanam, dan memiliki visualisasi yang mudah dimengerti.

Pada kode ini diimplementasikan modul-modul yang telah dipelajari selama praktikum Sistem Siber Fisik selama setengah semester genap ini. Berikut merupakan penjabaran rinci untuk setiap pengimplementasian materi yang digunakan:

Implementasi I/O

Pada Arduino Uno, terdapat tiga port digital input/output yang dapat digunakan, yaitu Port B, C, dan D. Setiap port ini memiliki 8 bit, memungkinkan kami untuk mengendalikan atau membaca 8 pin digital pada satu port. Port B, C, dan D untuk mengontrol berbagai peralatan elektronik seperti LED, relay, dan lainnya.

Port B pada Arduino Uno terdiri dari 8 pin digital yang diberi nomor 8 hingga 15. Pin 8 hingga 13 digunakan untuk mengendalikan peralatan elektronik, sementara pin 14 dan 15 digunakan untuk komunikasi I2C. Dengan menggunakan Port B, dapat dikontrol berbagai perangkat elektronik dengan mudah. Sementara itu, Port D pada Arduino Uno juga memiliki 8 pin digital, yaitu pin 0 hingga 7. Pin-pin ini dapat digunakan untuk mengendalikan peralatan elektronik sesuai dengan kebutuhan proyek yang sedang dibuat.

Dalam proyek yang telah kami buat, Port B digunakan untuk mengontrol buzzer, sedangkan Port D digunakan untuk menghubungkan sensor MQ2 dan DHT11. Dengan menggunakan Port B dan D, kami dapat mengintegrasikan buzzer serta sensor-sensor tersebut ke dalam sistem yang sedang dikembangkan.

ADC (Analog to Digital Converter)

ATmega328 memiliki built-in ADC, dengan spesifikasi memiliki 10-bit level menggunakan metode successive approximation, dan 8 multiplexed input channel. Hasil konversi 10-bit data dimasukan ke dalam register ADCL (untuk low-byte) dan ADCH (untuk high-byte). ATmega328 memiliki tiga referensi tegangan yaitu AVcc, internal 2.56V, atau pin AREF. Dalam proyek ini, kami mengimplementasikan ADC dari input sensor MQ2. Hal ini dikarenakan sensor MQ2 menggunakan input analog, sehingga harus diconvert ke sinyal digital agar dapat dibaca.

Serial Monitor

Pada proyek ini, kami menggunakan serial monitor untuk menampilkan live-track dari sensor MQ2 yang telah diconvert dari analog ke digital dengan menggunakan ADC. Dengan demikian kami dapat melihat kondisi dari udara secara langsung dengan nilai yang ditampilkan oleh Serial Monitor dan ditandai juga dengan menggunakan Buzzer jika angka yang ditampilkan terlalu tinggi (menampilkan tingkat tingginya gas beracun pada udara)

Aritmatika

Karena besar register pada mikrokontroler yang kami gunakan adalah 8-bit, maka operasi aritmatika yang dijalankan pada AVR kebanyakan dikhkususkan untuk operasi 8-bit. Lalu kami menggunakan modul aritmatika untuk membandingkan nilai hasil pembacaan dari sensor MQ-2 dengan nilai parameter yang telah kami tetapkan sebagai threshold untuk batas dimana udara tersebut masih tergolong bersih dan tidak banyak mengandung gas beracun. Kami menetapkan parameter di angka 70, sehingga buzzer akan berbunyi dan memberikan

warning jika nilai hasil pembacaan dari sensor MQ-2 naik keatas 70. Pada modul ini, digunakan OPCODE BRSH dan BRLO untuk pengaplikasian pada buzzer dan BRNE untuk delay. Berikut adalah penjelasan rinci mengenai opcode yang digunakan dalam proyek ini:

1. BRSH (Branch if Same or Higher):

Opcode BRSH digunakan untuk melakukan lompatan (branch) ke alamat memori tertentu jika flag carry (C) dalam register status menghasilkan nilai 1 (atau set). Biasanya, opcode ini digunakan dalam konteks perbandingan angka unsigned. Jika kondisi memenuhi (nilai lebih besar atau sama), maka program akan melanjutkan eksekusi dari alamat memori yang ditentukan, sedangkan jika kondisi tidak terpenuhi, program akan melanjutkan eksekusi dari alamat berikutnya.

2. BRLO (Branch if Lower):

Opcode BRLO digunakan untuk melakukan lompatan (branch) ke alamat memori tertentu jika flag carry (C) dalam register status menghasilkan nilai 0 (atau clear). Seperti BRSH, opcode ini juga digunakan dalam perbandingan angka unsigned. Jika kondisi memenuhi (nilai lebih kecil), maka program akan melanjutkan eksekusi dari alamat memori yang ditentukan. Jika kondisi tidak terpenuhi, program akan melanjutkan eksekusi dari alamat berikutnya.

3. BRNE (Branch if Not Equal):

Opcode BRNE digunakan untuk melakukan lompatan (branch) ke alamat memori tertentu jika flag zero (Z) dalam register status menghasilkan nilai 0 (atau clear). Biasanya, opcode ini digunakan dalam perbandingan angka signed atau unsigned. Jika kondisi memenuhi (nilai tidak sama), program akan melanjutkan eksekusi dari alamat memori yang ditentukan. Jika kondisi tidak terpenuhi, program akan melanjutkan eksekusi dari alamat berikutnya.

Communication

Pada proyek ini, kami juga menggunakan communication protocol yakni SPI. Penggunaan SPI pada proyek ini adalah komunikasi antara sensor DHT11 dengan Arduino dan ditampilkan di MAX7219. Hal ini dapat digunakan untuk mengirim dan menerima data dengan cepat antara perangkat mikrokontroler dan perangkat keras eksternal, sehingga memungkinkan kontrol dan akses data yang efisien. MAX7219 akan menampilkan suhu dan kelembaban yang dibaca dari sensor DHT11 dalam format teks.

Sensor

Untuk mengimplementasikan modul ini, kami menggunakan 2 buah sensor yang berbeda, yakni DHT11 dan MQ2, dimana DHT11 mengambil input digital dan MQ2 mengambil input analog yang akan di convert ke digital dengan menggunakan ADC. Sensor DHT11 digunakan untuk mengambil data humidity dan temperature dari ruangan tempat perangkat tersebut berada. Sementara MQ2 berfungsi untuk mendeteksi gas beracun yang ada disekitar sensor dan memberikan hasil pembacaan dari MQ2 dalam bentuk heksadesimal. Semakin tinggi angkanya, semakin tinggi tingkat racun dalam udara yang ada di sekeliling sensor tersebut.

CHAPTER 3

TESTING AND EVALUATION

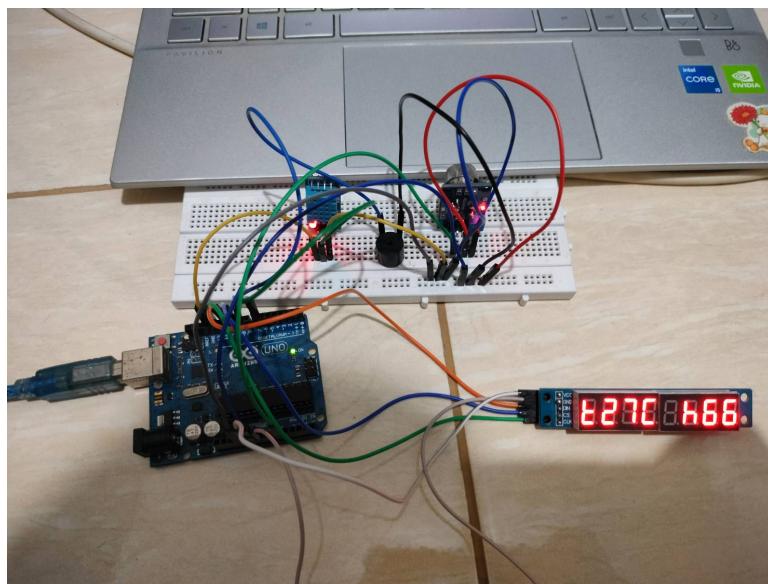
3.1 TESTING

Setelah proses perangkaian dan penulisan kode dilakukan, maka proses percobaan dapat dilakukan. Untuk melakukan percobaan, pertama akan dilakukan dalam kondisi normal dan mengukur hasil yang diperoleh dan membuktikan apakah sensor DHT11 dan MQ-2 Gas Sensor berjalan dan ditampilkan di MAX7219.

Kemudian untuk mengetahui apakah buzzer berjalan dengan baik, maka percobaan akan dilakukan dengan memberikan kondisi tertentu dimana buzzer akan menyala. Dalam percobaan ini, akan dilakukan dengan mengganti batas aman yang seharusnya tinggi akan diturunkan sehingga buzzer akan berbunyi ketika telah melewati batas aman tersebut.

3.2 RESULT

Berdasarkan percobaan yang telah dilakukan sebelumnya, maka diperoleh hasil sebagai berikut. Untuk percobaan untuk mengetahui apakah DHT11 dan MQ-2 Sensor telah berhasil mengambil data dari lingkungan dan mengirimkan data ke Arduino, maka dapat dilihat pada hasil berikut:



Gambar 4. Hasil Rangkaian

Kemudian, akan dilakukan percobaan untuk mengetahui apakah buzzer bekerja atau tidak. Potongan kode yang akan diganti adalah batas aman buzzer untuk menyala yaitu pada kode berikut:

```
322    CPI R19, 70      ; Compare immediate hasil high-byte dengan 70
323    BRLO kaga       ; Branch ke 'mati' jika R16 lebih rendah dari 70
324    CPI R19, 70      ; Compare immediate hasil high-byte dengan 70
325    BRSH bunyi      ; Branch ke 'bunyi' jika R16 lebih tinggi dari atau sama dengan 70
326    RCALL delay_sec ; 1 second delay
327    RET
```

Gambar 5. Cuplikan Kode

Dengan mengganti kode tersebut, buzzer telah menyala dan menandakan batas aman yang diturunkan tersebut telah berhasil.

3.3 EVALUATION

Evaluasi dari proyek Air Quality Monitor merupakan proyek yang akan ngecek suhu dan kelembapan, dan juga akan mengecek apakah ada gas yang mudah terbakar atau tidak, seperti metana, propana, dan lain-lain. Pada saat alat atau sistem dijalankan maka Air Quality Monitor akan memproses data menjadi analog dan akan ditampilkan di seven segment, yaitu MAX7219, dan juga saat ada gas yang terdeteksi maka buzzer akan menyala. Testing yang telah dilakukan telah memberikan hasil yang akurat, hal ini dapat ditunjukkan dengan MAX7219 menampilkan temperatur dalam celcius dan juga kelembapan, dan juga dapat dibuktikan pada saat ada gas yang kami berikan, maka buzzer akan berbunyi. Dengan menggunakan sistem ini masyarakat dapat mengetahui suhu dan kelembapan dan juga mengetahui apakah ada gas atau tidak.

CHAPTER 4

CONCLUSION

Air Quality Monitor merupakan alat untuk memantau suhu dan kelembapan lingkungan. Pada alat ini terdapat DHT11 yang akan berfungsi untuk menjadi sensor dan mengambil data tentang suhu dan kelembapan lingkungan. Data yang didapat tentang suhu dan kelembapan akan ditampilkan di seven segment. Kemudian juga ada sensor MQ-2 yang berfungsi untuk mendeteksi jika ada gas di lingkungan. Kedua sensor ini masing masing akan mengambil data dari lingkungan dan diterima di Arduino Uno. Berdasarkan kedua data tersebut juga akan dianalisis hasil yang diperoleh, dan apabila hasil yang didapat melebihi standar atau batas aman yang telah ditentukan, maka buzzer akan menyala dan memberikan peringatan.

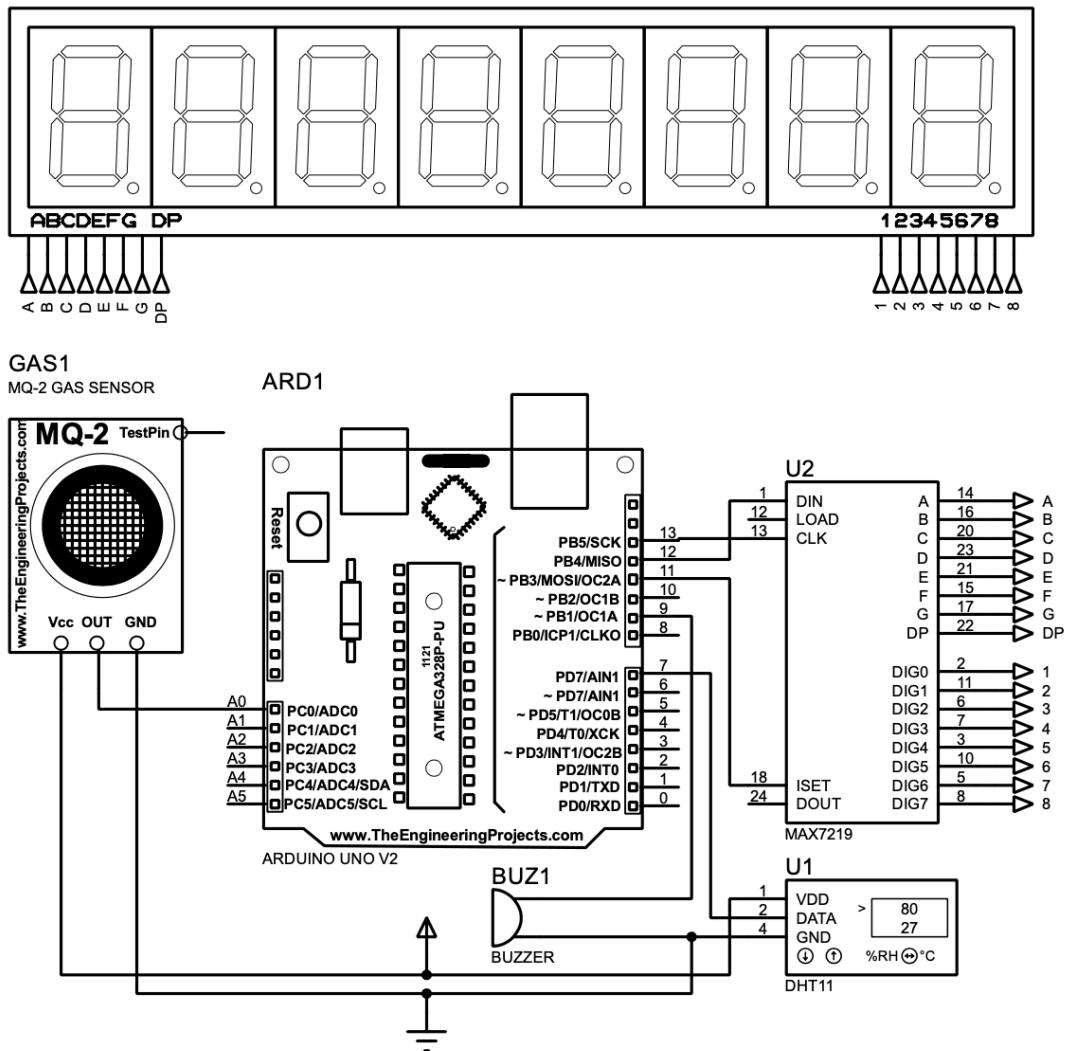
Proyek ini telah mengimplementasikan beberapa modul yang telah dipelajari selama praktikum, yaitu modul 2, 3, 4, 5, dan 9. Dengan begitu proyek Air Quality Monitor akan membantu masyarakat untuk mengetahui suhu dan kelembapan, sehingga masyarakat dapat melakukan tindakan preventif. Kami harap alat yang kami buat dapat dipergunakan oleh masyarakat dengan baik, kemudian juga kami berharap alat yang kami dapat dikembangkan lagi.

REFERENCES

- [1] IQAir. "World Air Quality Ranking." [Online]. Available: <https://www.iqair.com/id/world-air-quality-ranking>. [Accessed: May 16, 2023].
- [2] Modul 2 hingga 9 Praktikum Sistem Siber Fisik. [Online]. Available: <https://emas2.ui.ac.id/course/view.php?id=38402>. [Accessed: May 6, 2023].
- [3] Mouser Electronics. "DHT11 Technical Data Sheet Translated Version." [Online]. Available: <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>. [Accessed: May 9, 2023]
- [4] Electronicos Caldas. "DHT11 Datasheet." [Online]. Available: https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf. [Accessed: May 9, 2023]
- [5] Pololu. "MQ2 Gas Sensor Datasheet." [Online]. Available: <https://www.pololu.com/file/0J309/MQ2.pdf>. [Accessed: May 11, 2023]
- [6] Octopart. "Parallax Datasheet." [Online]. Available: <https://datasheet.octopart.com/27930-Parallax-datasheet-13536831.pdf>. [Accessed: May 11, 2023]
- [7] Anas Kuzechie. "Air Quality Monitoring System using MQ-2 Gas Sensor and Arduino." [Online]. Available: https://www.youtube.com/watch?v=wHhLtpEA5ws&list=PL09ZAP7_T_LmlX5vctZV4PfZwMNzjX1F&index=11. [Accessed: May 9, 2023]
- [8] Nza Hawrami. "Arduino MQ2 Smoke Gas Sensor Circuit Setup Tutorial." [Online]. Available: <https://www.youtube.com/watch?v=wKxRwpLii9E>. [Accessed: May 9, 2023]
- [9] Circuit Basics. "How to Set Up the DHT11 Humidity Sensor on an Arduino." [Online]. Available: <https://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/>. [Accessed: May 9, 2023]

APPENDICES

Appendix A: Project Schematic



Appendix B: Documentation

