CYBER-PHYSICAL SYSTEM FINAL PROJECT REPORT

DEPARTMENT OF ELECTRICAL ENGINEERING

UNIVERSITAS INDONESIA

SMART CHILLER

GROUP 19

IRFAN YUSUF KHAERULLAH	**2206813290**
RADITYA AKHILA GANAPATI	**2206016151**
AZRIEL DIMAS ASH-SHIDIQI	**2206059414**
DARREN ADAM DEWANTORO	**2206816600**

PREFACE

Puji dan syukur kami panjatkan atas kehadiran Tuhan yang Maha Esa atas segala rahmat dan karunianya sehingga kami dapat menyelesaikan laporan proyek akhir praktikum sistem siber fisik. Tidak lupa kami ucapkan terimakasih kepada Bapak Fransiskus Astha Ekadiyanto selaku Dosen mata kuliah sistem siber fisik dan bang Jordhie selaku asisten pendamping group 19 dalam mengerjakan proyek sistem siber fisik yaitu “Smart Chiller”.

Laporan ini dibuat untuk memenuhi salah satu komponen dari proyek akhir. Laporan ini berisikan hasil Pembuatan proyek dengan mengimplementasikan bahasa assembly pada alat yang dibuat. Laporan ini juga mencakup cara kerja, pengujian dan analisis dari proyek yang kami buat.

Kami menyadari laporan yang kami buat masih jauh dari kata sempurna, Oleh karena itu kami mengharapkan saran dan kritik dari pihak yang membaca laporan ini sehingga harapan yang kami dapat membuat laporan ini menjadi lebih baik. Kami berharap laporan ini akan menjadi bermanfaat bagi yang membacanya

Depok, May 28, 2024

Group 19

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 PROBLEM STATEMENT

Industri makanan memegang peran penting dalam menyediakan produk-produk makanan yang aman dan berkualitas kepada konsumen. Dalam hal ini, pengelolaan bahan makanan menjadi suatu hal yang sangat penting dalam menjaga kualitas dari produk. Salah satu bentuk pengelolaan terhadap bahan makanan adalah dengan mengatur tempat penyimpanan makanan (Chiller) yang digunakan. Salah satu faktor krusial dalam Chiller yang dapat mempengaruhi kualitas dari suatu bahan makanan adalah suhu penyimpanan. Penyimpanan makanan pada suhu yang tidak tepat dapat menyebabkan pertumbuhan bakteri, kerusakan nutrisi, dan menurunkan umur simpan produk. Sehingga, pemantauan suhu menjadi kunci utama dalam memastikan kualitas dari suatu bahan makanan. Namun, terdapat tantangan yang selalu menjadi perhatian besar untuk dihadapi yaitu menjaga suhu dalam Chiller untuk selalu berada pada kondisi yang stabil dan aman, sehingga dapat selalu menjaga kesegaran dari bahan makanan.

Ketidakstabilan suhu dalam chiller menjadi salah satu tantangan utama yang dihadapi. Saat ini, terdapat banyak fasilitas chiller yang masih menggunakan sistem konvensional yang kurang efisien dalam menghadapi ketidakstabilan suhu ini. Pada Akhirnya, fluktuasi suhu yang tidak terkontrol ini lah yang akan merusak kualitas dari bahan makanan. Padahal, hal seperti ini dapat dicegah dengan memberikan peringatan apabila terjadi fluktuasi suhu yang melebihi ambang toleransi, sehingga operator atau pengelola mungkin dapat menyadari adanya masalah sebelum terlambat. Dengan hal ini, kekurangan dari sistem konvensional pun dapat ditutupi.

Sistem Konvensional juga memiliki permasalahan serius pada pengendalian energi, dimana pada beberapa kasus, chiller akan tetap berfungsi meskipun dalam kondisi kosong. Pengontrolan kondisi ini menjadi perlu dilakukan mengingat tidak efisiennya dalam pengendalian energi. Padahal, jika penerapan kasus ini dapat diatasi, maka, sistem pengendalian energi ini akan dapat mengoptimalkan penggunaan energi serta meningkatkan efisiensi energi.

1.2 PROPOSED SOLUTION

Untuk mengatasi tantangan yang telah diidentifikasi dalam problem statement, proyek "Smart Chiller" akan mengimplementasikan serangkaian solusi yang terintegrasi dan adaptif untuk menjawab berbagai keresahan yang menjadi perhatian utama pada sistem chiller konvensional. Beberapa komponen kunci dalam mengatasi permasalahan yang telah disebutkan sebelumnya adalah dengan menggunakan sensor untuk dapat memantau kondisi suhu di dalam chiller secara real-time, serta memberikan peringatan jika terjadi kondisi suhu yang tidak stabil, ditambah dengan pemberlakuan tindakan korektif untuk menjaga suhu pada rentang yang aman bagi penyimpanan makanan. Serta diberlakukan sistem penghematan energi dengan memantau isi dari chiller untuk mengaktifkan dan menonaktifkan sensor pemantau suhu agar tidak terjadi pemborosan energi. Dengan seperangkat hal ini, diharapkan dapat membantu dalam menjawab keresahan-keresahan yang dialami saat menggunakan Chiller berbasis sistem Konvensional.

1.3 ACCEPTANCE CRITERIA

Kriteria keberhasilan dari Proyek Smart Chiller, diantaranya:

1. Sensor DHT11 mampu mengambil data yang ada dan menampilkan di LCD MAX7219
2. Menonaktifkan Sensor DHT11 apabila tidak ada barang disekitar Sensor HCSR04 dan mengaktifkan Sensor DHT11 apabila terdapat barang disekitar Sensor HCSR04
3. Menyalakan Buzzer dan servo ketika temperatur yang terbaca terlalu panas
4. Menyalakan LED ketika suhu terlalu dingin

1.4 ROLES AND RESPONSIBILITIES

Peran dan tanggung jawab yang dimiliki masing masing anggota kelompok, diantaranya:

Roles	Responsibilities	Person
Role 1	Mengerjakan Code, membantu membuat laporan, membuat README, Membantu membuat schematic rangkaian di proteus,membantu membuat rangkaian asli, Mempersiapkan alat yang dibutuhkan	Irfan Yusuf Khaerullah
Role 2	membantu membuat laporan, membuat Flowchart, membnatu membuat rangkaian Asli	Raditya Akhila Ganapati
Role 3	Mengerjakan Code, Membantu membuat schematic rangkaian di proteus, membantu membuat Laporan, membantu mengtroubleshooting code dan melakukan testing proyek.	Azriel Dimas Ash-Shidiqi
Role 4	Membantu membuat Laporan, Melakukan pengecekan code	Darren Adam Dewantoro

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES

No	Jenis Kegiatan	Bulan Mei Tanggal Ke-																			
		8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1.	Idea Development																					
2.	Software Development																					
3.	Hardware Design completion																					
4.	Integration and Testing of Hardware and Software																					
5.	Final Product Assembly and Testing																					

CHAPTER 2
IMPLEMENTATION

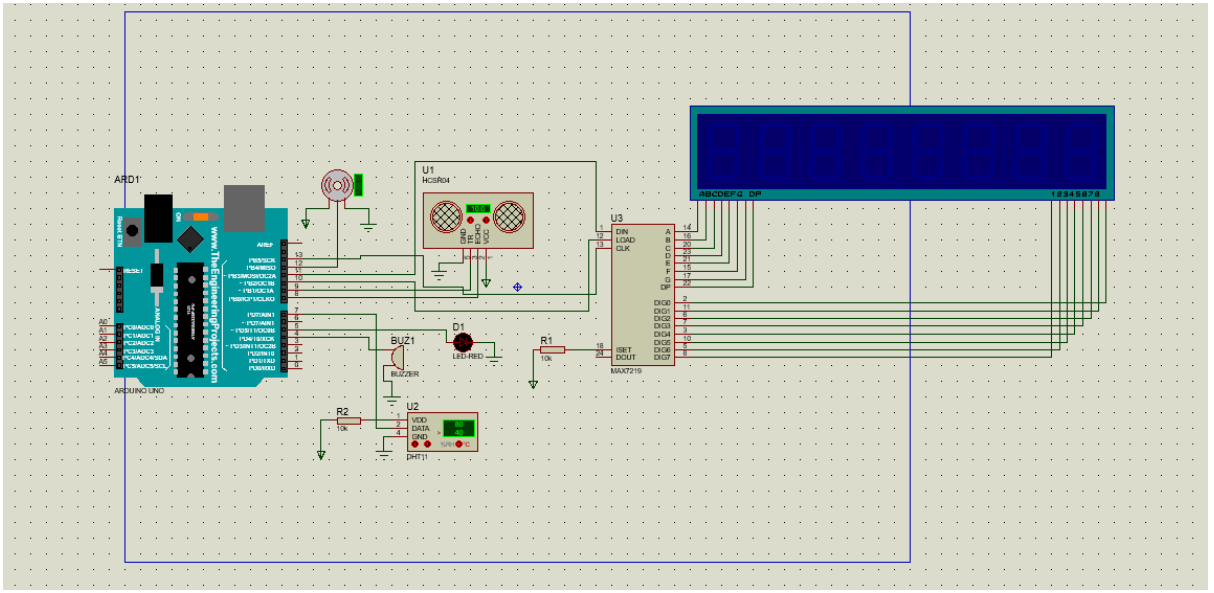
2.1 HARDWARE DESIGN AND SCHEMATIC

Proyek “Smart Chiller” ini memerlukan beberapa jenis hardware dalam implementasinya, seperti:

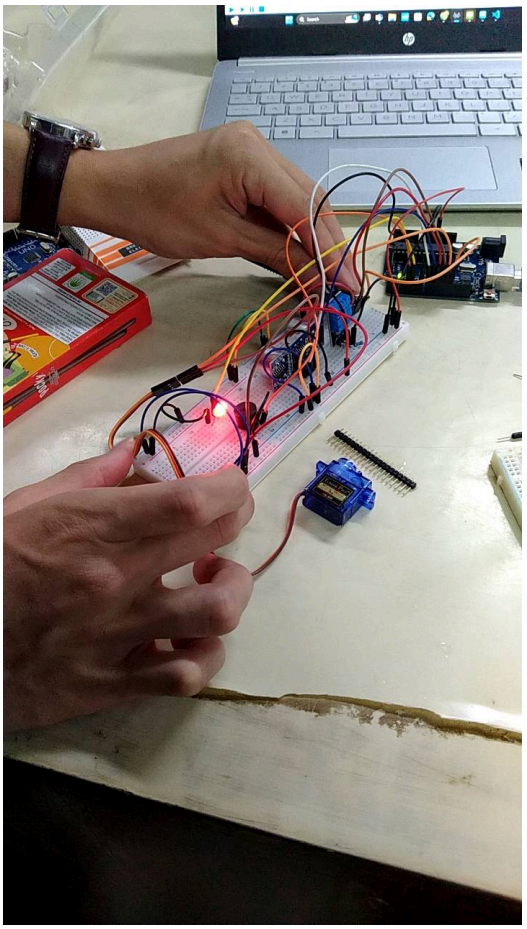
- Arduino Uno
- Sensor DHT11
- Sensor HCSR04
- Resistor
- LED
- Buzzer Active
- Servo SG90
- LCD MAX7219
- Kabel Jumper
- Breadboard

Penggunaan alat diatas dapat menciptakan perangkat “Smart Chiller”. Dengan bukti rangkaian Proteus dan rangkaian Asli:

Rangkaian Proteus



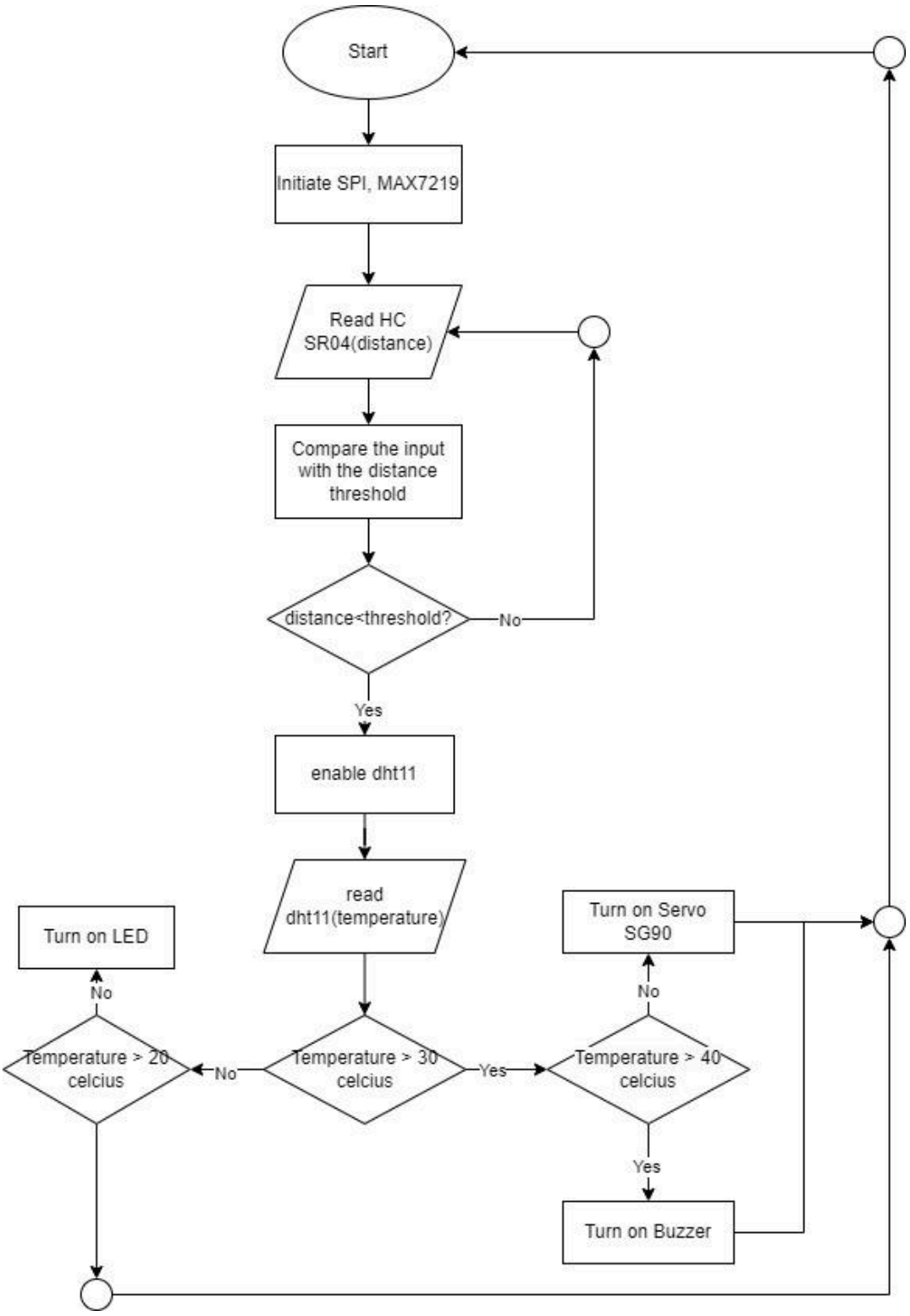
Rangkaian Asli



2.2 SOFTWARE DEVELOPMENT

Perangkat “Smart Chiller” dapat bekerja dengan memanfaatkan teori-teori yang telah dipelajari sebelumnya pada praktikum Sistem Siber Fisik. Beberapa penjelasan dari sistem “Smart Chiller” ini adalah menggunakan Sensor HCSR04 (Sensor Jarak) untuk mendeteksi ada atau tidaknya barang di dalam Chiller dengan memperkirakan suatu batasan jarak tertentu. Jika barang di dalam tidak terdeteksi, maka tidak akan melakukan pembacaan suhu (menonaktifkan sensor DHT11), dan akan terus membaca jarak untuk mendeteksi barang dalam chiller, sedangkan jika barang di dalam terdeteksi, maka akan melanjutkan ke pembacaan suhu oleh sensor DHT11 (mengaktifkan sensor DHT11). Sensor DHT11 akan melakukan pembacaan suhu, jika suhu lebih kecil dari rentang yang diinginkan, maka LED akan menyala, sedangkan jika suhu lebih besar dari rentang yang diinginkan, maka Servo SG90 akan bekerja dengan menerapkan langkah korektif untuk menurunkan suhu ke rentang yang diinginkan. Algoritma ini akan berjalan secara terus menerus selama perangkat aktif.

Flowchart



Source code

```
;------------------------

; Assembly Code
```

```
;-------------------------

#define __SFR_OFFSET 0x00

#include "avr/io.h"

;-------------------------

.global main


main:

    RCALL SPI_MAX7219_init

    RCALL MAX7219_display_digit

agn:

    RCALL HCSR04_read_distance

    CPI R28, 15

    BRLO activate_dht11

    RJMP deactivate_dht11


activate_dht11:

    SBI PORTD, 2

    RCALL delay_ms

    RCALL DHT11_sensor

    RCALL check

    RJMP agn


deactivate_dht11:

    CBI PORTD, 2

    RCALL delay_ms
```

```
    RJMP agn



HCSR04_read_distance:
    SBI   DDRB, 1         ;pin PB1 as o/p (Trigger)
    CBI   DDRB, 0         ;pin PB0 as i/p (Echo)
    SBI   PORTB, 1
    CBI   PORTD, 1
    CBI   PORTD, 0
    RCALL delay_timer0
    CBI   PORTB, 1        ;send 10us high pulse to sensor
    RCALL echo_PW         ;compute Echo pulse width count
    RET


echo_PW:


    LDI   R20, 0b00000000
    STS   TCCR1A, R20     ;Timer 1 normal mode
    LDI   R20, 0b11000101 ;set for rising edge detection &
    STS   TCCR1B, R20     ;prescaler=1024, noise cancellation ON


rising_edge:
    IN    R21, TIFR1
    SBRS  R21, ICF1
    RJMP  rising_edge     ;loop until rising edge is detected
```

```
        LDS   R16, ICR1L      ;store count value at rising edge
        OUT   TIFR1, R21      ;clear flag for falling edge detection
        LDI   R20, 0b10000101
        STS   TCCR1B, R20     ;set for falling edge detection

    falling_edge:
        IN    R21, TIFR1
        SBRS  R21, ICF1
        RJMP  falling_edge    ;loop until falling edge is detected
        LDS   R28, ICR1L      ;store count value at falling edge
        SUB   R28, R16        ;count diff R22 = R22 - R16
        OUT   TIFR1, R21      ;clear flag for next sensor reading
        RET



    DHT11_sensor:
        RCALL delay_2s      ; wait 2s for DHT11 to get ready
        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
        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)
      MOV    R28, R24             ; store temperature in R28 for
display_digit
     MOV    R31, R24           ; store temperature in R31 for servo
and LED condition
     LDI    R29, 0x07         ; MSD temp will be display_digited on
digit 6
     LDI    R30, 0x06         ; LSD temp will be display_digited on
digit 5
    RCALL binary_to_decimal ; temp in decimal
    MOV   R28, R25
        LDI     R29,  0x02              ; MSD humidity will be
display_digited on digit 1
        LDI     R30,  0x01              ; LSD humidity will be
display_digited on digit 0
    RCALL binary_to_decimal   ; humidity in decimal
    RET                     ; RETurn to loop
```

```
check:
    CPI   R31, 30          ; compare temperature in R31 with 30
    BRLO  servo_off        ; if temperature < 30, servo off
    CBI   PORTD, 5         ; turn off LED
    RCALL servo_on         ; turn on servo

servo_off:
    CBI   PORTD, 4         ; turn off LED
    CPI   R31, 20          ; compare temperature in R31 with 20
    BRGE  led_off          ; if temperature >= 20, led off
    SBI   PORTD, 5         ; turn on LED
    RET                    ; RETurn to looping

led_off:
    CBI   PORTD, 5         ; turn off LED
    RET                    ; RETurn to looping


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  skip           ; else, received bit=0, jump to skip
        SEC                  ; set carry flag (C=1)
        ROL   R24            ; shift in 1 into LSB data register
        RJMP  w5             ; jump & wait for low pulse

skip: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_10us:
        LDI R21, 10

delay_10us_loop:
        NOP
        DEC R21
        BRNE delay_10us_loop
        RET

delay_20ms:                  ; delay 20ms
        LDI   R21, 255
l3: LDI   R22, 210
```

```
l4: LDI   R23, 2
l5: DEC   R23
    BRNE  l5
    DEC   R22
    BRNE  l4
    DEC   R21
    BRNE  l3
    RET


delay_2s:               ; delay 2s
    LDI   R21, 255
l6: LDI   R22, 255
l7: LDI   R23, 164
l8: DEC   R23
    BRNE  l8
    DEC   R22
    BRNE  l7
    DEC   R21
    BRNE  l6
    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

    l2: IN    R20, TIFR0      ; get TIFR0 byte & check
        SBRS  R20, OCF0A      ; if OCF0=1, skip next instruction
        RJMP  l2              ; else, loop back & check OCF0 flag
        CLR   R20
        OUT   TCCR0B, R20     ; stop timer0
        LDI   R20, (1<<OCF0A)
        OUT   TIFR0, R20      ; clear OCF0 flag
        RET

    SPI_MAX7219_init:
        SBI   DDRD, 4        ; pin PD4 as o/p buzzer
        SBI   DDRD, 5        ; pin PD5 as o/p LED
        .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<<SPIE)|(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_display_digit:
        LDI   R17, 0x08       ; select digit 7
        LDI   R18, 0x0F       ; data = t (defg) (00001111)
        RCALL send_bytes      ; send command & data to MAX7219
        LDI   R17, 0x05       ; select digit 4
        LDI   R18, 0x4E       ; data = C (adef) (01001110)
        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 (cefg) (00010111)
        RCALL send_bytes      ; send command & data to MAX7219
```

```
    RET


send_bytes:
    CBI   PORTB, SS       ; enable slave device MAX7219
    OUT   SPDR, R17       ; transmit command


wait_transmission1:
    IN    R19, SPSR
    SBRS  R19, SPIF           ; wait for byte transmission
    RJMP  wait_transmission1  ; to complete
    OUT   SPDR, R18           ; transmit data


wait_transmission2:
    IN    R19, SPSR
    SBRS  R19, SPIF           ; wait for byte transmission
    RJMP  wait_transmission2  ; to complete
    SBI   PORTB, SS           ; disable slave device MAX7219
    RET


binary_to_decimal:
    CLR   R26             ; set counter1, initial value 0
    CLR   R27             ; set counter2, initial value 0


convert_100:  CPI   R28, 100      ; compare R28 with 100
RET:  BRMI  convert_10            ; jump when R28 < 100
```

```
        INC   R26                    ; increment counter1 by 1
        SUBI  R28, 100               ; R28 = R28 - 100
        RJMP  convert_100

convert_10:CPI   R28, 10       ; compare R28 with 10
        BRMI  display_digit    ; jump when R28 < 10
        INC   R27              ; increment counter2 by 1
        SUBI  R28, 10          ; R28 = R28 - 10
        RJMP  convert_10

display_digit:
        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

servo_on:
        CPI   R31, 40      ; compare temperature in R31 with 40
        BRLO  buzzer_Off   ; if temperature < 40, buzzer off
        SBI   PORTD, 4     ; turn on buzzer

buzzer_Off:
```

```
        LDI   R17, 0
        OUT   SPCR, R17       ; enable SPI as master, fsck=fosc/16
        SBI   DDRB, 4         ; pin PB4 o/p for servo control

    again_servo:
        LDI   R26, 8          ; counter for # of rotation pos
        LDI   ZL, lo8(rotate)
        LDI   ZH, hi8(rotate)

    ls1: LPM   R24, Z+        ; load rotation pos
        RCALL rotate_servo    ; & rotate servo
        DEC   R26
        BRNE  ls1             ; go back & get another rotate pos
        RJMP SPI_MAX7219_init ; go back & repeat

    rotate:
        .byte 40,70,90,110,180,110,90,70

    rotate_servo:
        LDI   R20, 10
        SBI DDRB, 1

    ls2: SBI   PORTB, 4
        RCALL delay_timer0_servo
        CBI   PORTB, 4            ; send msec pulse to rotate servo
```

```
          RCALL delay_20ms_servo        ; wait 20ms before re-sending
pulse

         DEC   R20

         BRNE  ls2                     ; go back & repeat PWM signal


     bak:RCALL delay_ms          ; 0.5s delay

         RET                     ; & RETurn to main subroutine


     delay_timer0_servo:         ; delay via Timer0

         CLR   R21

         OUT   TCNT0, R21        ; initialize timer0 with count=0

         MOV   R21, R24

         OUT   OCR0A, R21

         LDI   R21, 0b00001100

         OUT   TCCR0B, R21       ; timer0: CTC mode, prescaler 256


     ls3: IN    R21, TIFR0       ; get TIFR0 byte & check

         SBRS  R21, OCF0A        ; if OCF0=1, skip next instruction

         RJMP  ls3               ; else, loop back & check OCF0 flag

         CLR   R21

         OUT   TCCR0B, R21       ; stop timer0

         LDI   R21, (1<<OCF0A)

         OUT   TIFR0, R21        ; clear OCF0 flag

         RET
```

```
delay_20ms_servo:                ; delay 20ms
    LDI   R21, 255
ls4: LDI   R22, 210
ls5: LDI   R23, 2
ls6: DEC   R23
    BRNE  ls6
    DEC   R22
    BRNE  ls5
    DEC   R21
    BRNE  ls4
    RET


delay_ms:                ; delay 0.5s
    LDI   R21, 255
ls7 :LDI   R22, 255
ls8 :LDI   R23, 41
ls9 :DEC   R23
    BRNE  ls9
    DEC   R22
    BRNE  ls8
    DEC   R21
    BRNE  ls7
    RET
```

bedasarkan kode yang dibuat, fungsi dari SPI_MAX7219_init berfungsi untuk melakukan inisialisasi yang mengatur moder SPI dan mengirim perintah dan data ke MAX7219 yang berfungsi untuk menontrol seven segment display. fungsi MAX7219_display_digit digunakan untuk menampilkan data awal pada display dan akan ada loop yang memanggil HCSR04_read_distance yang digunakan untuk membaca jarak. Pada subroutine HCSR04_read_distance diatur PINB 1 sebagai trigger dan PINB0 sebagai echo dan akan mengirim trigger pulse ke sensor HCSR04 dan memanggil subroutine echo_PW

Jika jarak lebih kecil dari 15 maka sensor DHT11 akan menagktfikan activate_dht11 jika tidak maka akan menonaktifkan deactivate_Dht11. Selanjutnya dht11_sensor akan membaca 40 bit data dari sensor d dan meyimpan suhu di register 28 dan 31 yang digunakan dalam display dan control selanjtunya akan memanggil binary_to_decimal untuk mengkonversi data suhu kelembapan menjadi decimal. Selanjutnya DHT11_reading akan membaca bit data dari sensor DHT11 dan menyimpan register 24.

Selanjutnya subroutine check akan mengcompare suhu yang disimpan di R31 dan menonaktfikan dan aktfikan LED dan Servo berdasarkan hasil comparenya jika suhu lebih besar dari 40 maka servo akan menyala dan buzzer akan berbunyi, jika suhu diantara 30-40 maka Led dan servo akan menyala tetapi Led menyala tidak secara konstan, jika suhu dibawah 20 maka Led akan menyala secara konstan

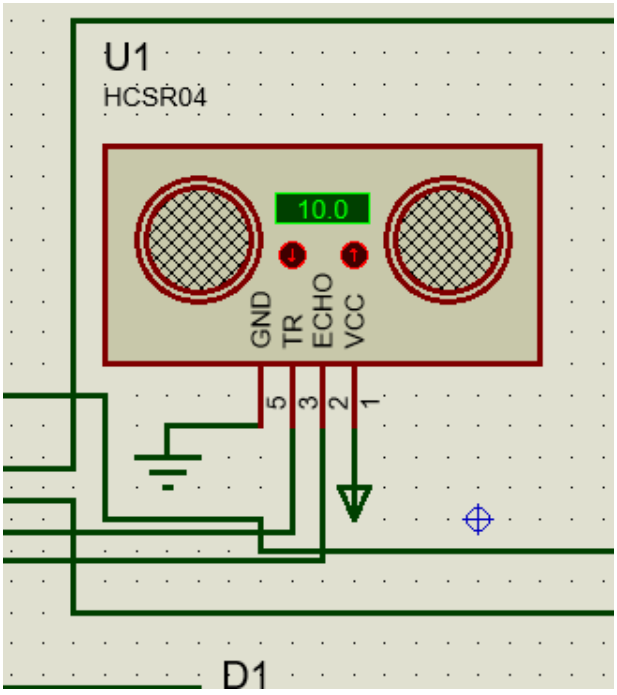
Slanjutnya ada subroutine display dan delay seperti MAX7219_display_digit yang digunakan digit pada 7 segment display, send_bytes yang digunakan untuk mengirim data ke MAX7219 melalui SPI, binary_to_decimal yang mengkonversi nilai biner suhu atau kelembapan menjadi nilai decimal untuk dsiplau dan fungsi fungsi delay yahg ada untuk memberikan delay menggunakan loop atau timer.

2.3 HARDWARE AND SOFTWARE INTEGRATION

A.Sensor HC-SR04 adalah sensor ultrasonik yang digunakan untuk mengukur jarak. Sensor ini bekerja dengan mengirimkan pulsa suara ultrasonik dan mengukur waktu yang dibutuhkan bagi pulsa tersebut untuk kembali setelah memantul dari suatu objek. Komponen dari HC SR04:

-Transmitter (Trig Pin): Memancarkan gelombang ultrasonik.

-Receiver (Echo Pin): Menerima gelombang ultrasonik yang dipantulkan kembali.



Cara Kerja :

- Mengirim Sinyal: Ketika Trig Pin diberi pulsa tinggi selama 10 mikrodetik, transmitter mengirimkan serangkaian gelombang ultrasonik pada frekuensi 40 kHz.
- Menunggu Pantulan: Gelombang ini akan merambat melalui udara dan jika mengenai objek, gelombang akan dipantulkan kembali ke sensor.

- Menerima Sinyal: Receiver mendeteksi gelombang yang kembali dan menghasilkan pulsa pada Echo Pin yang durasinya sama dengan waktu yang dibutuhkan gelombang untuk pergi ke objek dan kembali.
- Menghitung Jarak: Jarak dihitung berdasarkan durasi pulsa Echo dengan rumus berikut:

  Jarak= (Waktu Tempuh×Kecepatan Suara) / 2

  Kecepatan suara di udara adalah sekitar 343 meter per detik. Saat Jarak terukur kurang dari 15 cm, maka akan mengaktifkan sensor DHT11

B.Sensor DHT11 adalah sensor yang digunakan untuk mengukur suhu dan kelembaban. Pada percobaan ini kami hanya akan menggunakan DHT11 untuk mengukur suhu tempat makan. Berikut cara kerja dari DHT11:

- Pengukuran Suhu: Sensor suhu menggunakan elemen termistor yang mengubah resistansinya sesuai dengan perubahan suhu. Chip kontrol mengubah perubahan resistansi ini menjadi data suhu digital.
- Pengiriman Data: Sensor DHT11 mengirimkan data melalui satu pin data menggunakan protokol digital satu kawat (one-wire protocol)

C. Data suhu yang terbaca pada DHT11 akan ditampilkan pada LCD MAX7219. Data yang didapat dari DHT11 berbentuk biner sehingga perlu diubah dalam bentuk desimal, lalu ditampilkan pada LCD.

CHAPTER 3
TESTING AND EVALUATION

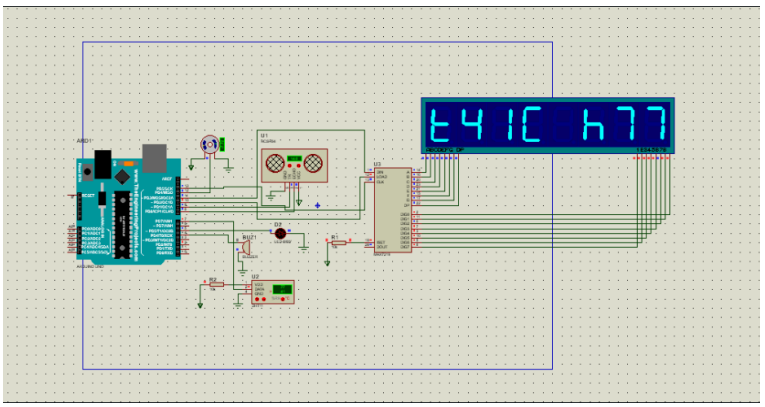
3.1 TESTING

Urutannya:

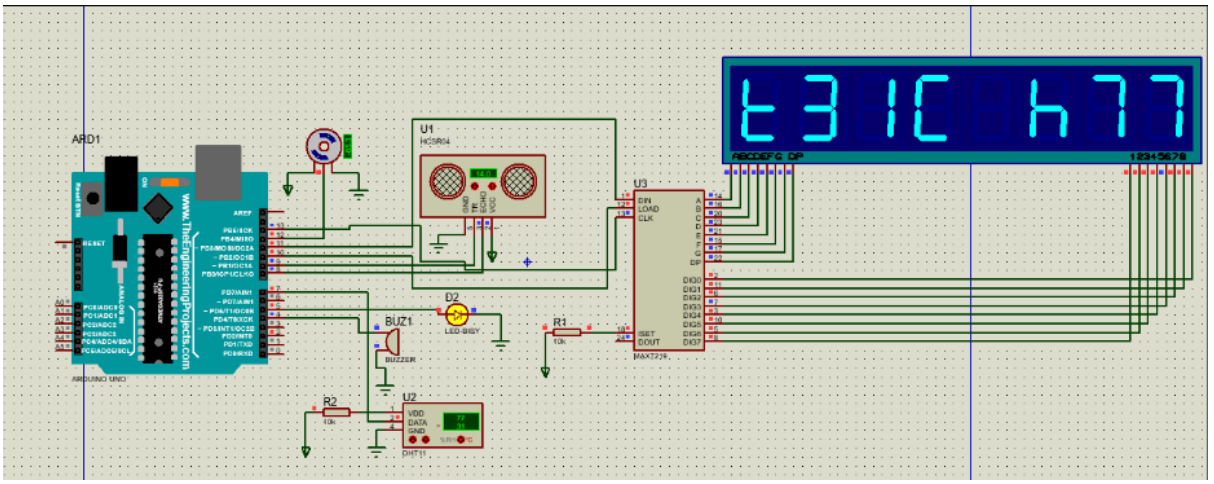
- Sensor jarak akan mengecek apakah ada makanan dalam jarak 15 cm atau tidak jika tidak ada maka sensor dan komponen lain tidak akan bekerja. Jika ada maka Sensor dan komponen lainnya akan
- Jika Sensor jarak mendeteksi makanan dalam 15cm maka sensor DHT11 akan bekerja dan akan membaca data suhu dan kelembapan yang ada dan menampilkannya di LCD MAX7219
- Jika sensor suhu mendeteksi bahwa suhu yang ada dibawah 20 maka akan menyalakan LED secara konstan , jika suhu mendeteksi diantara 20-30 maka LED akan menyala tetapi tidak konstan. Jika suhu berada diantara 30-40 maka LED dan servo akan menyala, Jika Suhu diatas 40 maka LED servo dan buzzer akan menyala

3.2 RESULT

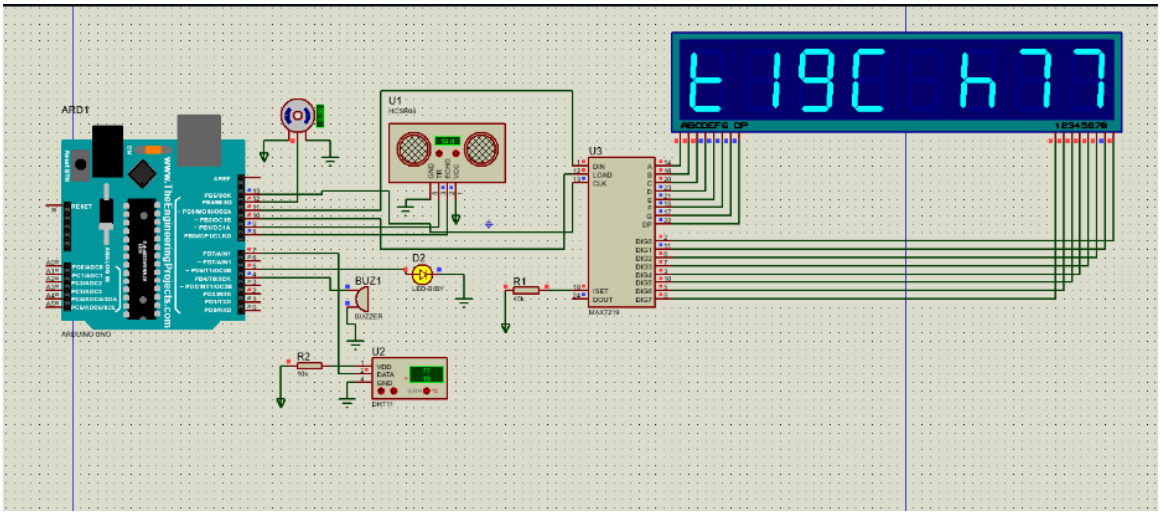
Berdasarkan alat yang telah kami buat, kami telah berhasil mengintegrasikan antara kodingan dengan perangkat kerasnya. Sensor DHT11 berhasil membaca kondisi temperatur ruangan dan berhasil ditampilkan pada layar MAX7219. Selain itu, kerja dari kipas dan buzzer juga telah sesuai dengan keinginan kami dimana saat temperatur di atas 40 derajat Celcius, maka buzzer akan menyala dan servo akan menyala untuk mendinginkan ruangan. Hal ini dapat dilihat pada rangkaian proteus berikut :



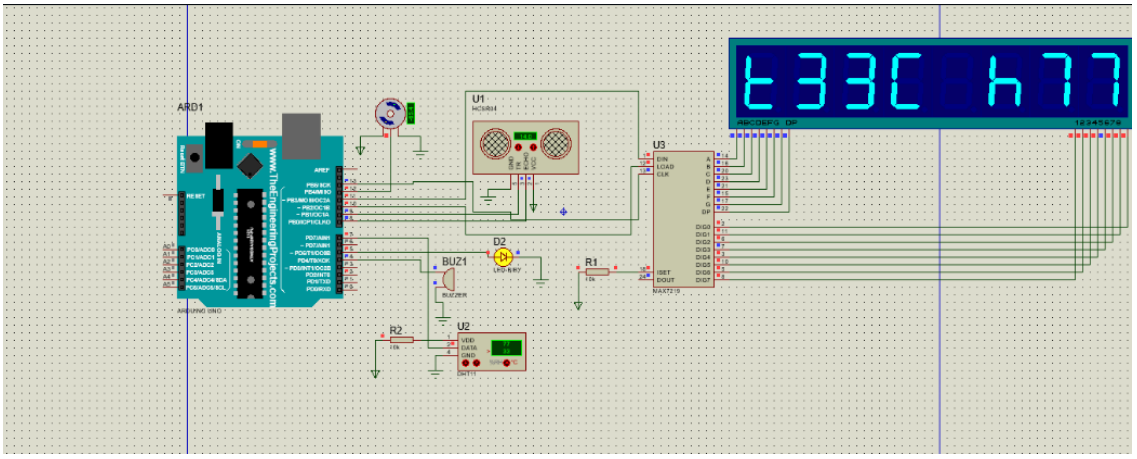
saat temperatur di antara 30 - 40 derajat Celcius, maka led akan menyala secara tidak konstan dan servo akan menyala untuk mendinginkan ruangan. Hal ini dapat dilihat pada rangkaian proteus berikut :

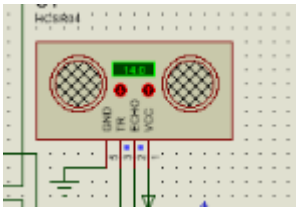


saat temperatur di bawah 20 derajat Celcius, maka led akan menyala secara konstan . Hal ini dapat dilihat pada rangkaian proteus berikut

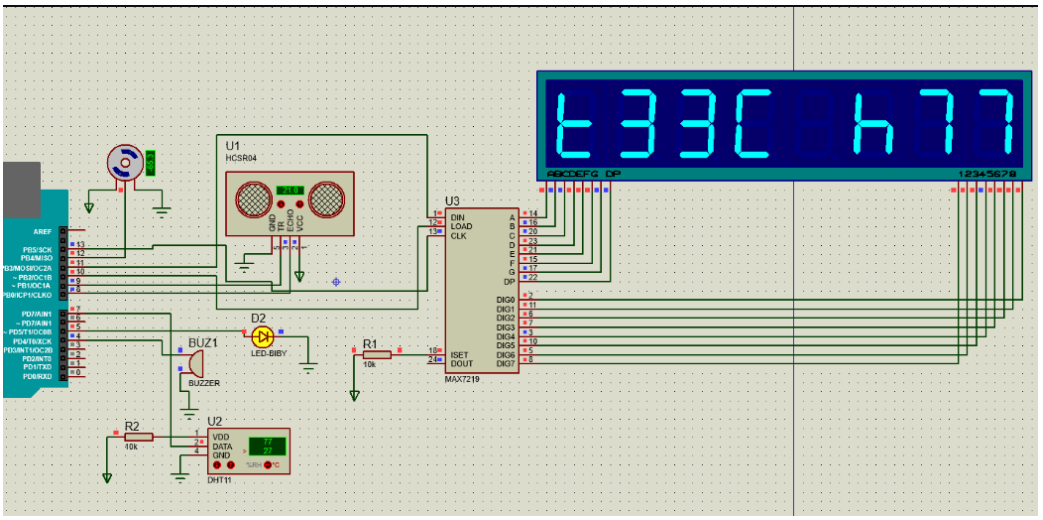


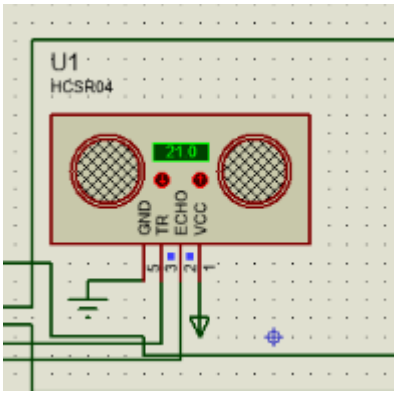
Kami juga berhasil menerapkan implementasi Sensor Jarak yang sesuai kami harapkan ketika sensor jarak mendeteksi barang dalam jaran 15 cm maka akan mengaktfikan sensor DHT 11 dan akan membaca data datanya. Hal Ini dapat dilihat pada rangkaian proteus berikut :





Saat jarak diatas 15 cm maka akan menonaktfikan sensor DHT11 dan data yang akan ditampilkan pada LCD adalah data yang terakhir dibaca oleh Sensor DHT11 sebelum di nonaktifkan. Hal Ini dapat dilihat pada rangkaian proteus berikut :





3.3 EVALUATION

Performa dari Smart chiller sudah sesuai dengan kriteria dan harapan yang diinginkan, Tetapi proyek ini juga masih ada kekurangan yaitu seperti Sensor DHT11 yang membaca data tidak selalu cepat sehingga dapat menimbulkan masalah jika membaca datanya terlalu lama. Meskipun begitu, Sensor DHT11 tetap berperan sebagai salah satu sensor utama dalam proyek ini. servo yang digunakan juga memiliki kekurangan seperti pergerakan kipas yang dimiliki tidak terlalu efisien, meskipun begitu, servo ini tetap digunakan sebagai kipas dalam proyek

CHAPTER 4
CONCLUSION

Smart Chiller yang kami buat sudah sesuai dengan harapan dan kriteria yang kami tentukan. Alat ini dapat digunakan untuk untuk memonitoring kualitas makanan bedasarkan suhu dari chiller yang menyimpan makanan. Selain itu alat ini memanfaatkan sensor HCSR04 untuk mendeteksi apakah ada makanan atau tidak didalam chiller. Jika terdapat makanan maka akan mengaktifkan Sensor DHT11 jika tidak maka tidak akan mengaktfikan sensor DHT11. Dengan menggunakan sensor DHT11 suhu dapat dibaca secara real time dan ditampilkan melalui LCD MAX7219. Smart chiller juga menggunakan aktuator yang dapat memberikan pertanda ketika chiller berada diluar suhu yang ditentukan seperti led yang akan menyala secara konstan saat suhu dibawah 20 dan led dan servo yang akan menyala saat suhu diantara 30-40 dan led servo dan buzzer yang akan menyala ketika suhu sudah diatas 40 derajat. Kesimpulan dari proyek Smart Chiller adalah proyek telah bekerja dengan baik dan memberikan output yang sesuai dengan harapan sehingga dari harapan kami alat ini dapat bermanfaat untuk banyak orang.

REFERENCES

[1] Industrial Quick Search, “Chiller: What is it? How Does It Work? Types & Uses,” *Industrial Quick Search Directory*. https://www.iqsdirectory.com/articles/chillers.html (accessed May 25, 2024).

[2] R. Santos, “Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino | Random Nerd Tutorials,” *Random Nerd Tutorials*, Oct. 07, 2021. https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/ (accessed May 25, 2024).

[3] Ø. N. Dahl, “Arduino Buzzer Tutorial: Play Melodies with Your Arduino,” *Build Electronic Circuits*, Nov. 15, 2023. https://www.build-electronic-circuits.com/arduino-buzzer/ (accessed May 25, 2024).

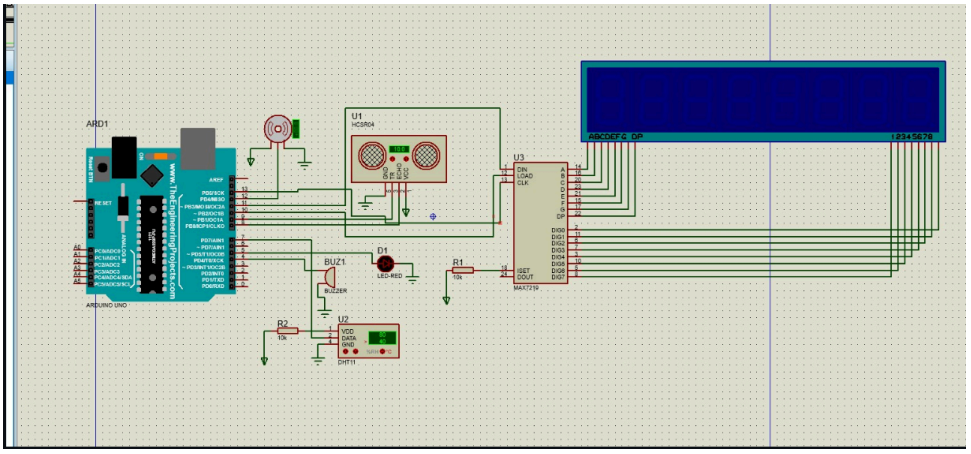
[4] “Assembly via Arduino (part 19) - dht11 sensor,” *YouTube*, 29-Nov-2021. [Online]. Available: https://www.youtube.com/watch?v=vnLpzvkCUq8. [Accessed: 28-May-2024]

[5] “Assembly via Arduino (part 20) - DHT11 data on MAX7219 display,” *YouTube*, 02-Dec-2021. [Online]. Available: https://www.youtube.com/watch?v=UjWbE2Q75ms. [Accessed: 28-May-2024]

[6] “Assembly via Arduino - Programming HC-SR04 sensor,” Anas Kuzechie Projects. [Online]. Available: https://akuzechie.blogspot.com/2021/12/assembly-via-arduino-programming-hc.html. [Accessed: 28-May-2024]

APPENDICES

Appendix A: Project Schematic



Appendix B: Documentation