



THERMOSMART

Our Team

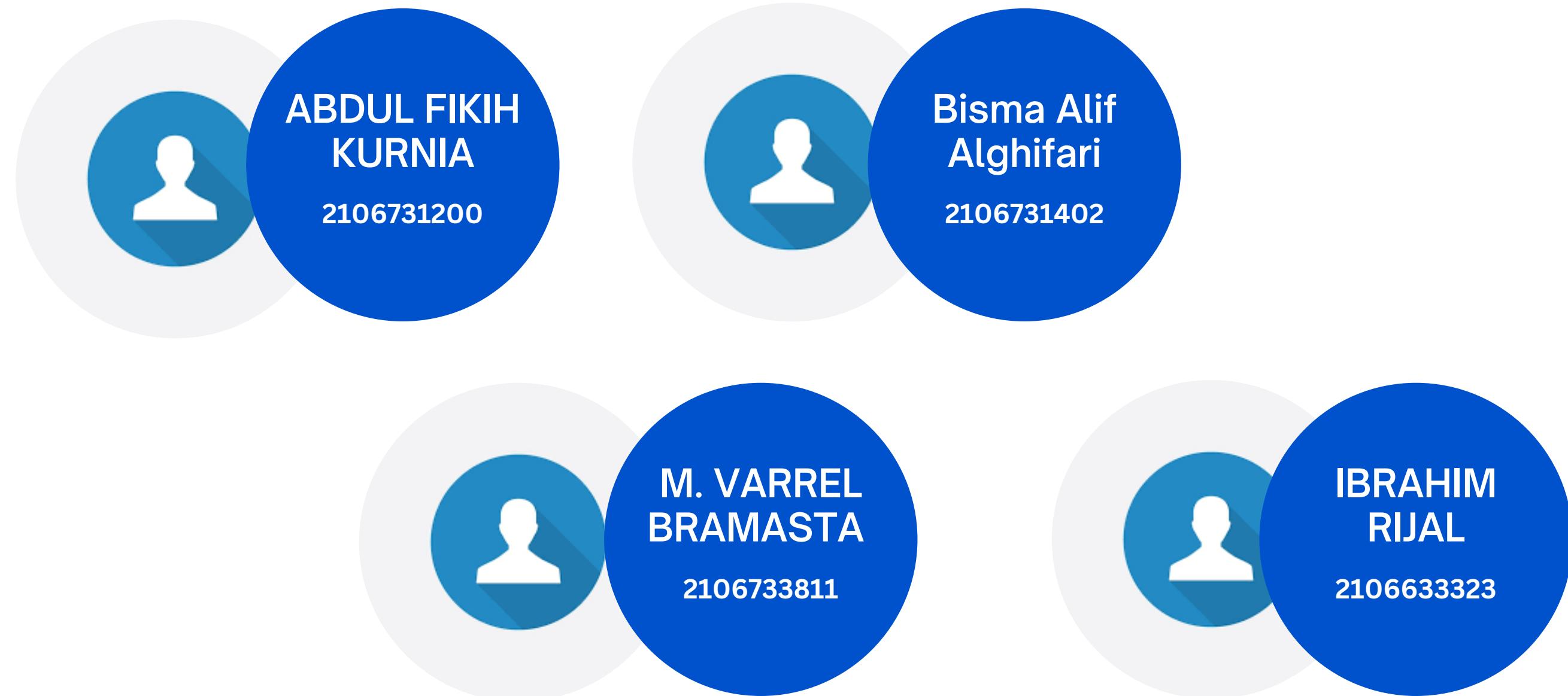


Table of Content

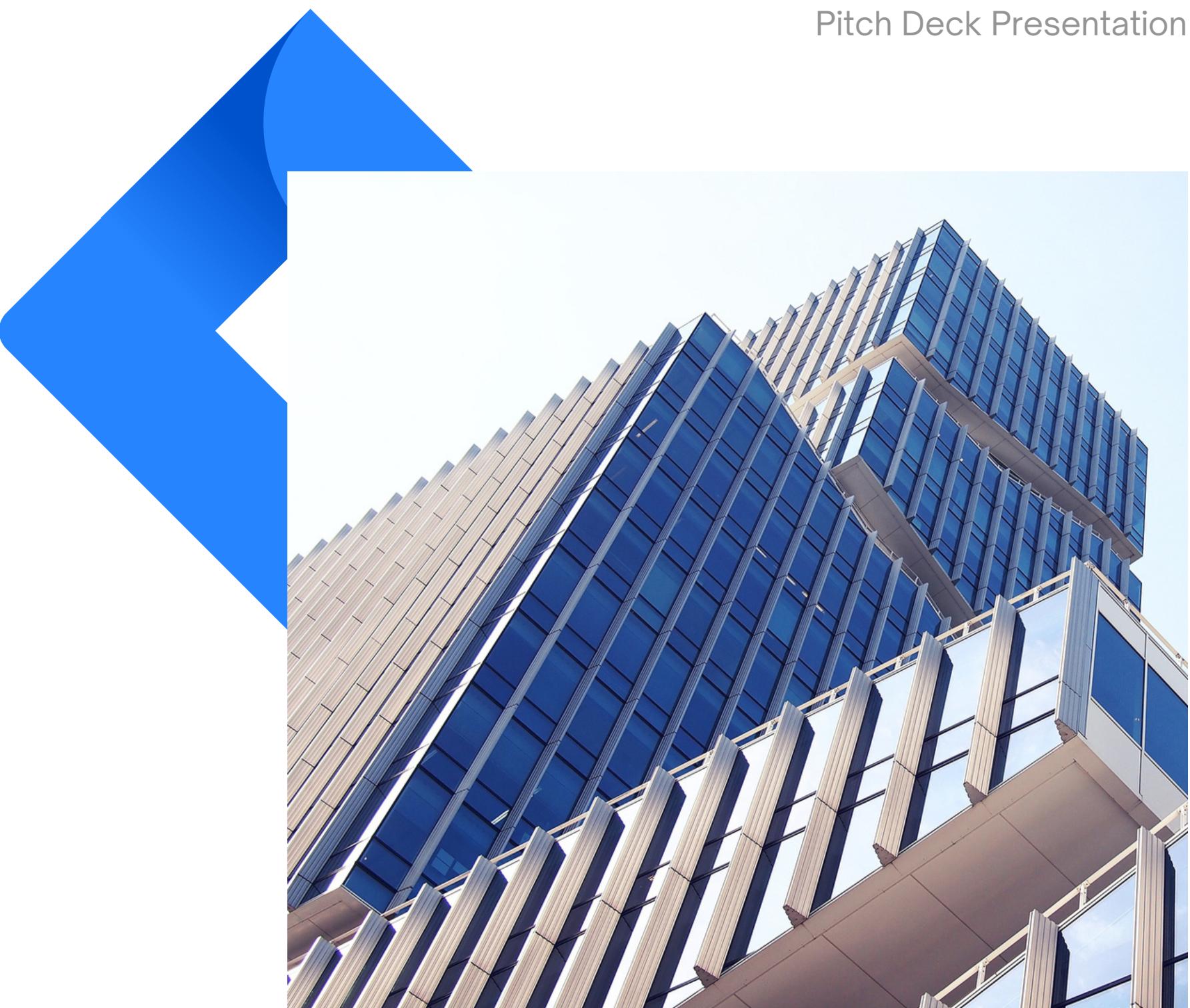
- Introduction
 - Problem Statement
 - Proposed Solution
- Implementation
 - Hardware Design Schematic
- Testing and Evaluation
 - Testing
 - Result
 - Evaluation
- Conclusion



INTRODUCTION

PROBLEM STATEMENT

In a room equipped with an air conditioner, individuals often experience discomfort due to inconsistent temperatures that do not meet their preferences. This issue is exacerbated by factors such as inadequate ventilation and leaks in the walls, which disrupt the airflow and create uneven temperature distribution throughout the room. Furthermore, the condensate water produced by the air conditioner needs to be temporarily stored. However, if the storage containers are not promptly emptied when they become full, there is a risk of flooding inside the room. Negligence in managing the condensate water can lead to physical messiness, damage to furniture, and potential seepage into the floor.





INTRODUCTION

PROPOSED SOLUTION



Smart Control System Integration

Integrating temperature and water level sensors into the air conditioning system.



Temperature Control

Utilizing a temperature sensor to monitor room temperature.



Condensate Water Management

Water level sensor to monitor the height of discharge water in the drainage tank.



GROUP A6



Figure 2 Arduino uno



Figure 3 LCD display



Figure.4 Water Level

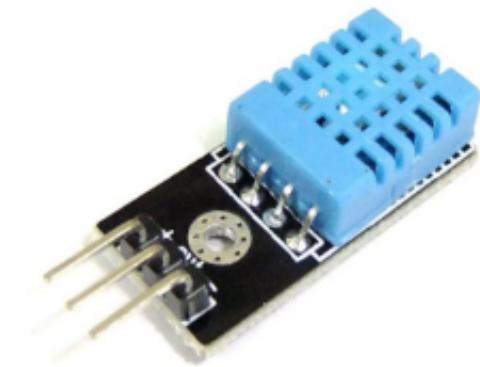


Figure.5 DHT11

IMPLEMENTATION

Hardware Design Schematic



GROUP A6



Figure.6 Buzzer



Figure.7 Relay



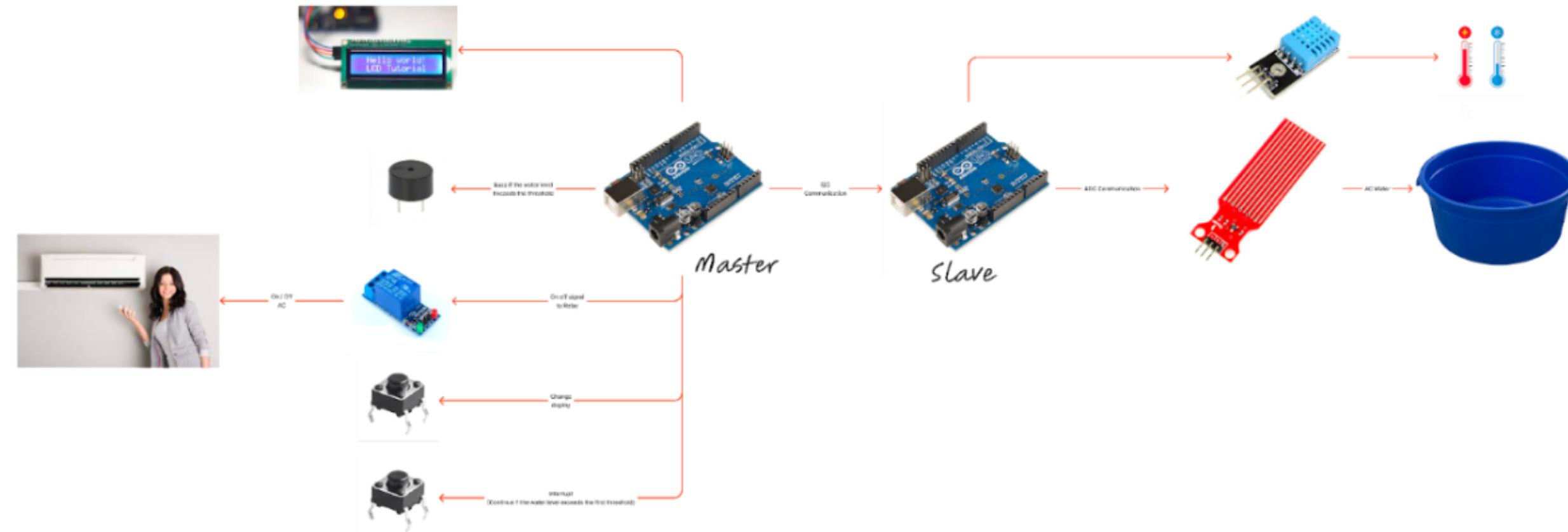
Figure.8 Button

IMPLEMENTATION

Hardware Design Schematic



GROUP A6



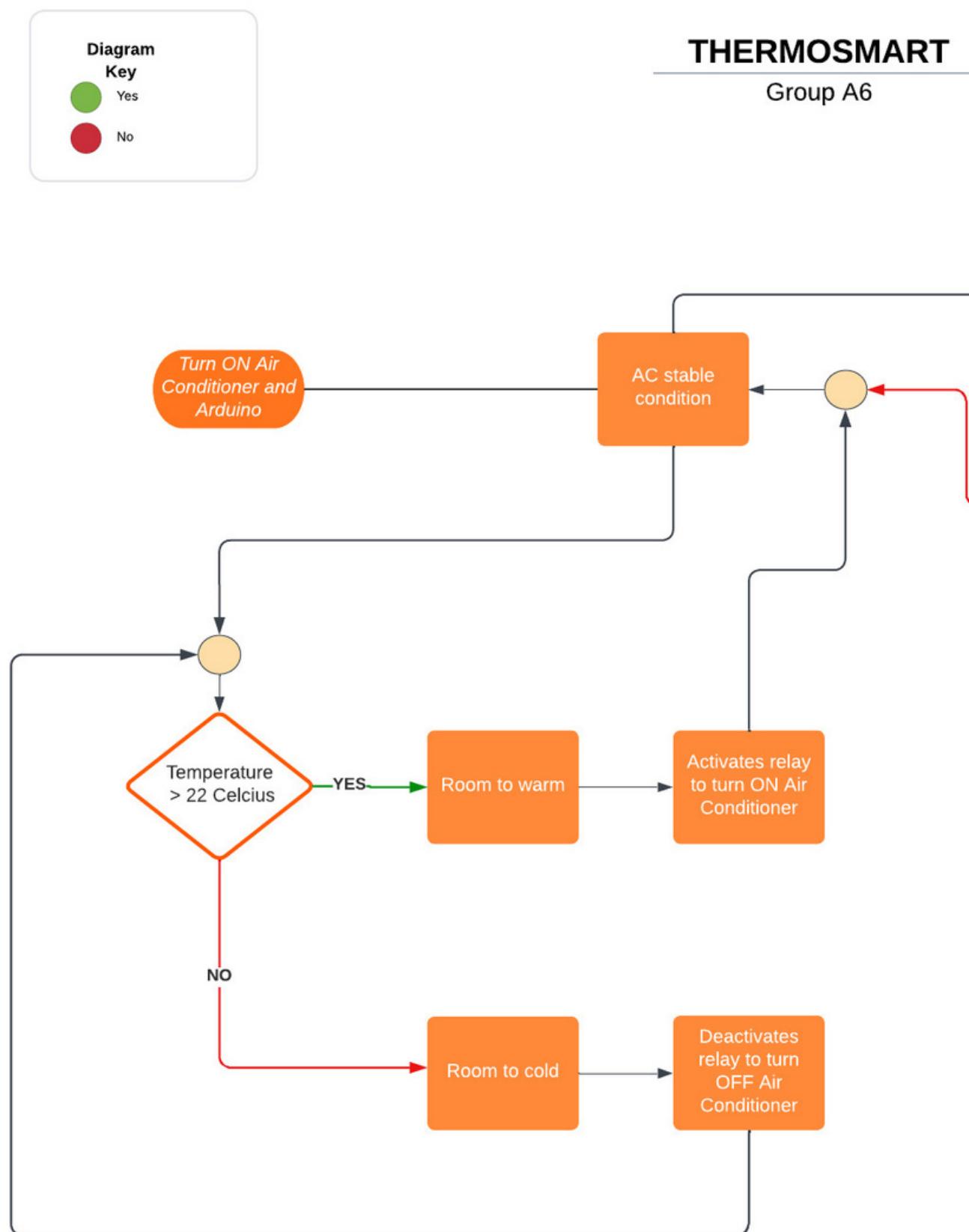
IMPLEMENTATION

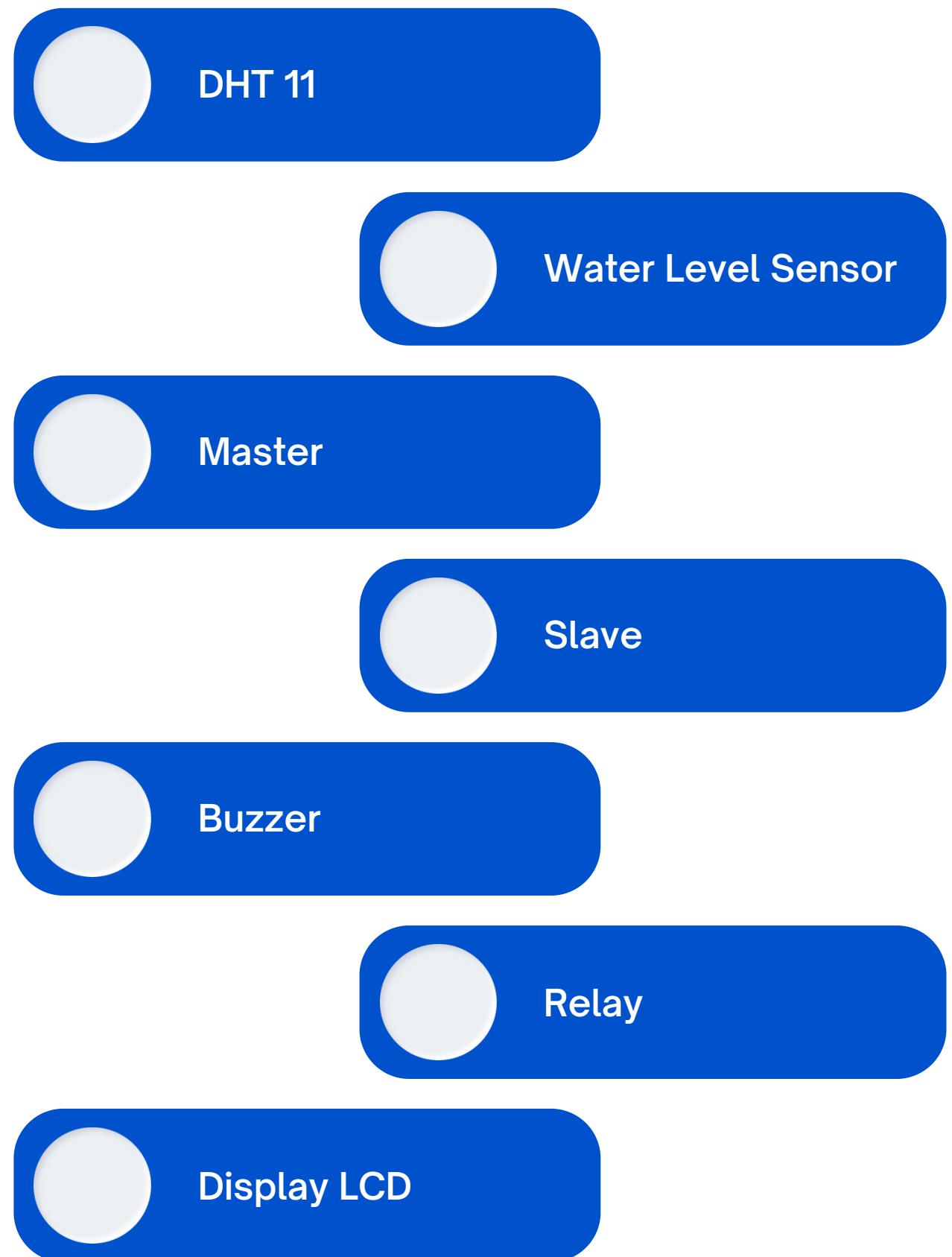
Hardware Design Schematic



SOFTWARE DEVELOPMENT

FLOWCHART





SOFTWARE DEVELOPMENT



```
1 ;-----
2 ; Assembly code to read adc measurement results from water level sensor
3 #define __SFR_OFFSET 0x00
4 #include "avr/io.h"
5 ;---- Global Declaration ---
6 .global init_ADC
7 .global init_serial
8 .global print_ADC
9 .global main
10 ;---- Main Label ----
11 main:
12 SBI DDRC, 0 ;sets bit 0 of DDR for port C to 1. Configures pin PC0 as O/P
13 LDI R20, 0x40 ;load the value 0xc0 into R20
14 STS ADMUX, R20 ;store the value of R20 to ADMUX. ADC = 5V, read ADC0 i/p pin
15 LDI R20, 0x87 ;loads the value 0x87 (10000111 in binary) into register R20
16 STS ADCSRA, R20 ;enables the ADC and sets its prescaler to 128
17 ;---
18 init_serial:
19 CLR R24 ;clears register R24
20 STS UCSR0A, R24 ;stores the value in R24 into UCSR0A for UART
21 STS UBRR0H, R24 ;stores the value in R24 into the upper byte of (UBRR0H) for the UART.
22 LDI R24, 51 ; store in UBRR0L 51
23 STS UBRR0L, R24 ;to set baud rate 19200
24 LDI R24, 1<<RXEN0 | 1<<TXEN0 ;sets bits in R24 corresponding to RXEN0, and TXEN0 for UART.
25 STS UCSR0B, R24 ;enable RXB & TXB ;stores the value in R24 to UCSR0B
26 LDI R24, 1<<UCSZ00 | 1<<UCSZ01 | 1<<UPM01
27 STS UCSR0C, R24 ;asynch, even parity, 1 stop, 8 bits
28
```

SOFTWARE DEVELOPMENT

WATER_LEVEL_SENSOR



```
1      ;-----  
2      ; Assembly Code  
3      ;-----  
4      #define _SFR_OFFSET 0x00  
5      #include "avr/io.h"  
6      ;-----  
7      .global DHT11_sensor  
8  
9      ;=====-----  
10     DHT11_sensor:  
11     ;-----  
12     agn:RCALL delay_2s      ;wait 2s for DHT11 to get ready  
13     ;-----  
14     ;start_signal  
15     ;-----  
16     SBI  DDRD, 3          ;pin PD7 as o/p  
17     CBI  PORTD, 3         ;first, send low pulse  
18     RCALL delay_20ms      ;for 20ms  
19     SBI  PORTD, 3         ;then send high pulse  
20     ;-----  
21     ;responce signal  
22     ;-----  
23     CBI  DDRD, 3          ;pin PD7 as i/p  
24     w1: SBIC PIND, 3  
25     RJMP w1                ;wait for DHT11 low pulse  
26     w2: SBIS PIND, 3  
27     RJMP w2                ;wait for DHT11 high pulse  
28     w3: SBIC PIND, 3  
29     RJMP w3                ;wait for DHT11 low pulse  
30     ;-----  
31     RCALL DHT11_reading ;read humidity (1st byte of 40-bit data)  
32     MOV  R25, R24  
33     RCALL DHT11_reading
```

IMPLEMENTATION

SOFTWARE DEVELOPMENT

DHT_11



```
#define __SFR_OFFSET 0x00
#include "avr/io.h"
;-----

.global main
;=====

main:
    SBI  DDRC, 0          ;set pin PC0 as i/p for ADC0
;
    LDI  R20, 0xE0         ;internal 2.56V, left-justified data, ADC0
    STS  ADMUX, R20
    LDI  R20, 0x87         ;enable ADC, ADC prescaler CLK/128
    STS  ADCSRA, R20
    JMP  LCD_write
    RET
;

LCD_write:
    LDI  R23, 48           ;constants used to get ASCII values
    LDI  R24, 7             ;for chars 0-->9 & A-->F
;
    LDI  R16, 0xFF
    OUT  DDRD, R16         ;set port D o/p for data
    OUT  DDRB, R16         ;set port B o/p for command
    CBI  PORTB, 0           ;EN = 0
    RCALL delay_ms         ;wait for LCD power on
;
    RCALL LCD_init         ;subroutine to initialize LCD
;
    RCALL disp_msg          ;display message on 1st line
;
;   LDI  R16, 0xC0          ;cursor beginning of 2nd line
;   RCALL command_wrt
    RCALL delay_ms
;
    LDI  R16, 0x20
    RCALL data_wrt          ;display LSD on LCD
```

IMPLEMENTATION

SOFTWARE DEVELOPMENT

DISPLAY_LCD





```
#define __SFR_OFFSET 0x00
#include "avr/io.h"
;-----
.global main
;=====
main:

LDI R16, 0b00010000 ;to toggle PB5 (D13)
LDI R17, 0b00000000
;-----
SBI DDRD, 5          ;set PD5 for o/p
OUT PORTD, R17        ;PD5 = 0
CBI DDRC, 0           ;set PC0 as i/p
SBI PORTC, 0          ;input pullup
rjmp loop

loop:
SBIC PINC, 0
RJMP loop

relay_toggle:
EOR R17, R16          ;R17 = R17 XOR R16
OUT PORTD, R17
;LDI R18, 0
RJMP loop

relay_toggle_on:
SBI PORTB, 3
RET

relay_toggle_on:
CBI PORTB, 3
RET
```

SOFTWARE DEVELOPMENT

RELAY





```
4  #define __SFR_OFFSET 0x00
5  #include "avr/io.h"
6  ;-----
7  .global main
8  ;-----
9  main:
10 LDI  R16, 0b00100000 ;to toggle PB5 (D13)
11 LDI  R17, 0b00000000
12 ;-----
13 SBI  DDRB, 5          ;set PB5 for o/p
14 OUT  PORTB, R17       ;PB5 = 0
15 CBI  DDRC, 0
16 SBI  PORTC, 0
17 LDI  R18, 100
18 agn:
19     LDI  R17, 0b00000000
20     OUT  PORTB, R17
21     SBIS PINC, 0
22     RJMP l1
23     RJMP agn
24
25 ;Subroutine to generate 1000hz square-wave
26 l1: RCALL delayhalfms ;apply delay via timer0
27 ;-----
28 EOR  R17, R16          ;R17 = R17 XOR R16
29 OUT  PORTB, R17
30 SBIS PINC, 0           ;toggle PB5
31 RJMP l1                ;go back & repeat toggle
32 RJMP agn
33 ;-----
34 delayhalfms:           ;0.5 ms delay via Timer0 for generating 1kHz tone
35 ;-----
36 CLR  R20
37 OUT  TCNT0, R20        ;initialize timer0 with count=0
38 LDI  R20, 125
39 OUT  OCR0A, R20        ;OCR0 = 125
```

IMPLEMENTATION

SOFTWARE DEVELOPMENT

BUZZER





```
4  #define __SFR_OFFSET 0x00
5  #include "avr/io.h"
6  ;-----
7  .global main
8  ;=====
9  main:
10    .equ SCK, 5
11    .equ MOSI, 3
12    .equ SS, 2
13
14    ;-----
15    ; SPI for DHT
16    ;-----
17    LDI R17, (1<<MOSI)|(1<<SCK)|(1<<SS)
18    OUT DDRD, R17      ; set MOSI, SCK, SS as o/p
19    LDI R17, (1<<SPE)|(1<<MSTR)|(1<<SPR0)
20    OUT SPCR, R17      ; enable SPI as master, fsck=fosc/16, mode 0
21    RJMP again
22    RET
23
24 again:
25    ;-----
26    ; SPI for DHT and Water Sensor
27    ;-----
28    RCALL dht_start    ;data read in R18
29    CBI PORTD, SS      ;enable slave device
30    OUT SPDR, R18      ;transmit byte to slave
31    loop:
32      IN R18, SPSR
33      SBRS R18, SPIF   ;wait for byte transmission
34      RJMP loop        ;to complete
35
36    RCALL sensor_Water
37    OUT SPDR, R16      ; transfer low byte sensor water
38    loop1:
39      IN R16, SPSR
```

IMPLEMENTATION

SOFTWARE DEVELOPMENT

MASTER





```
4 #define __SFR_OFFSET 0x00
5 #include "avr/io.h"
6 ;-----
7
8 .global main
9 ;-----
10 ;
11 ;-----
12 SPI_slave:
13 ;
14
15 main:
16     SBI DDRC, 0      ;set pin PC0 as i/p for ADC0
17 ;
18     LDI R20, 0xE0    ;internal 2.56V, left-justified data, ADC0
19     STS ADMUX, R20
20     LDI R20, 0x87    ;enable ADC, ADC prescaler CLK/128
21     STS ADCSRA, R20
22     LDI R21, 0xF0
23     OUT DDRD, R21 ;port D[7:4] is o/p
24     LDI R17, (1<<SPE)
25     OUT SPCR, R17 ;enable SPI as slave
26
27 ;
28 ; SPI reading
29 ;
30 122: ;read byte temp
31     IN R18, SPSR
32     SBRS R18, SPIF ;wait for byte reception
33     RJMP 122
34     IN R28, SPDR    ;i/p byte from data register
35
36     LDI R17, (0<<SPE)
37     OUT SPCR, R17 ;disable SPI as slave
38     RCALL LCD_write
39
```

IMPLEMENTATION

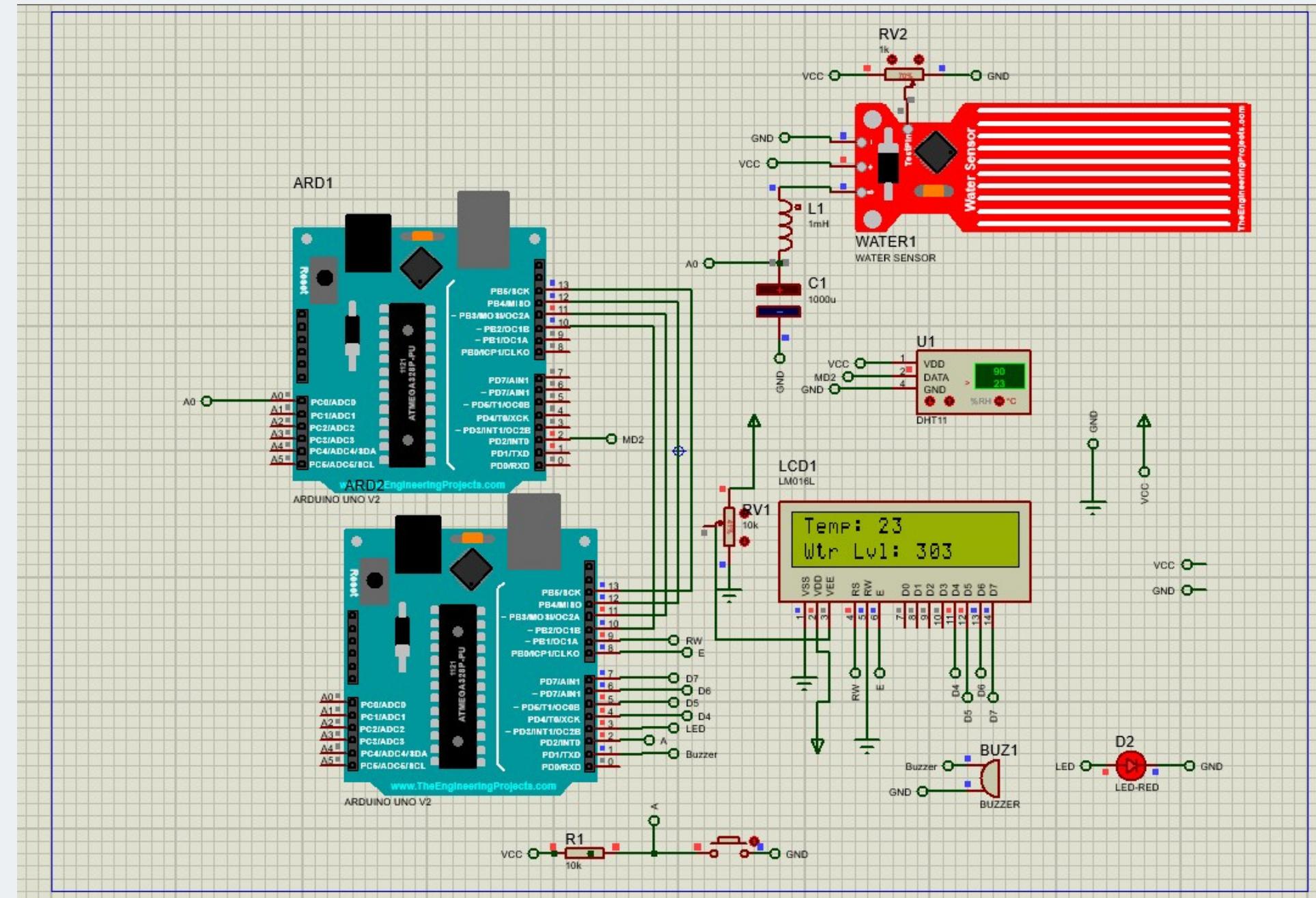
SOFTWARE DEVELOPMENT

SLAVE



GROUP A6

Hardware and Software Integration



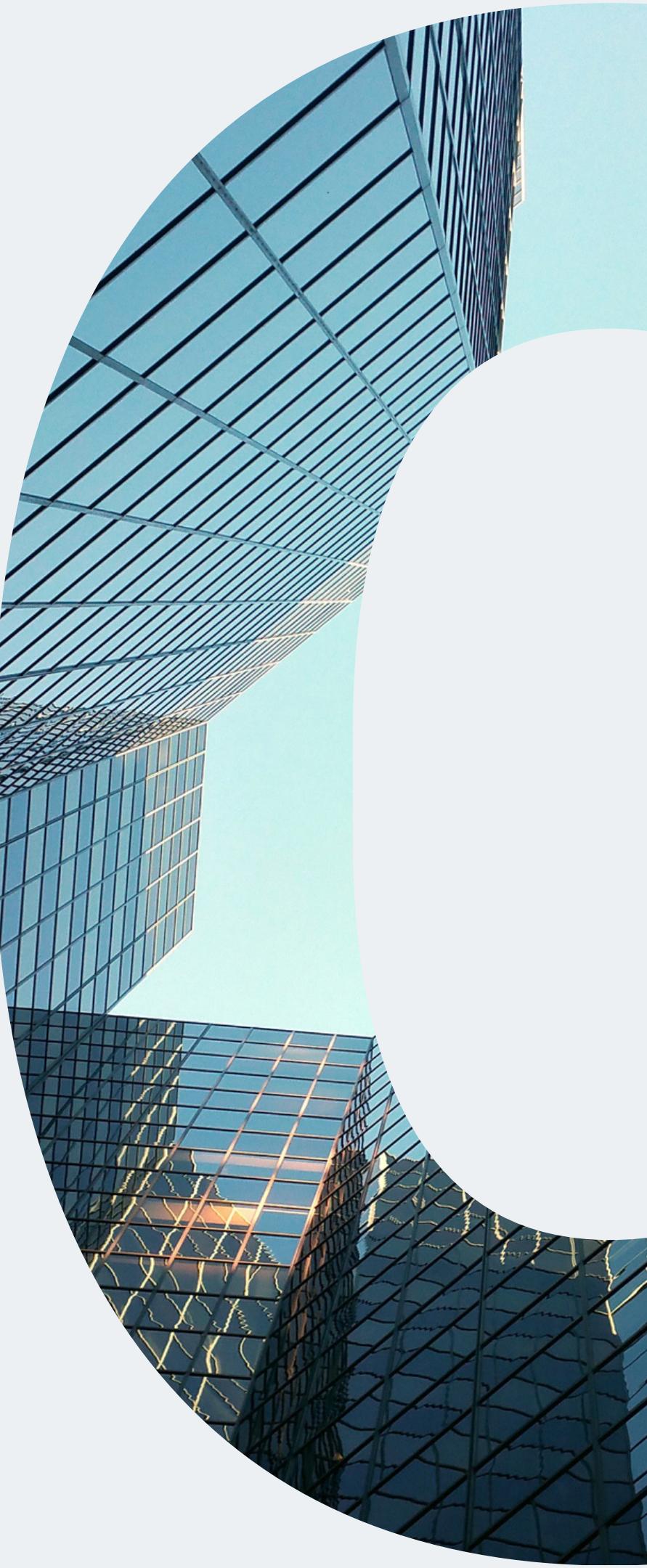


TESTING AND EVALUATION

Testing and Results

Low Water Level & High Temperature

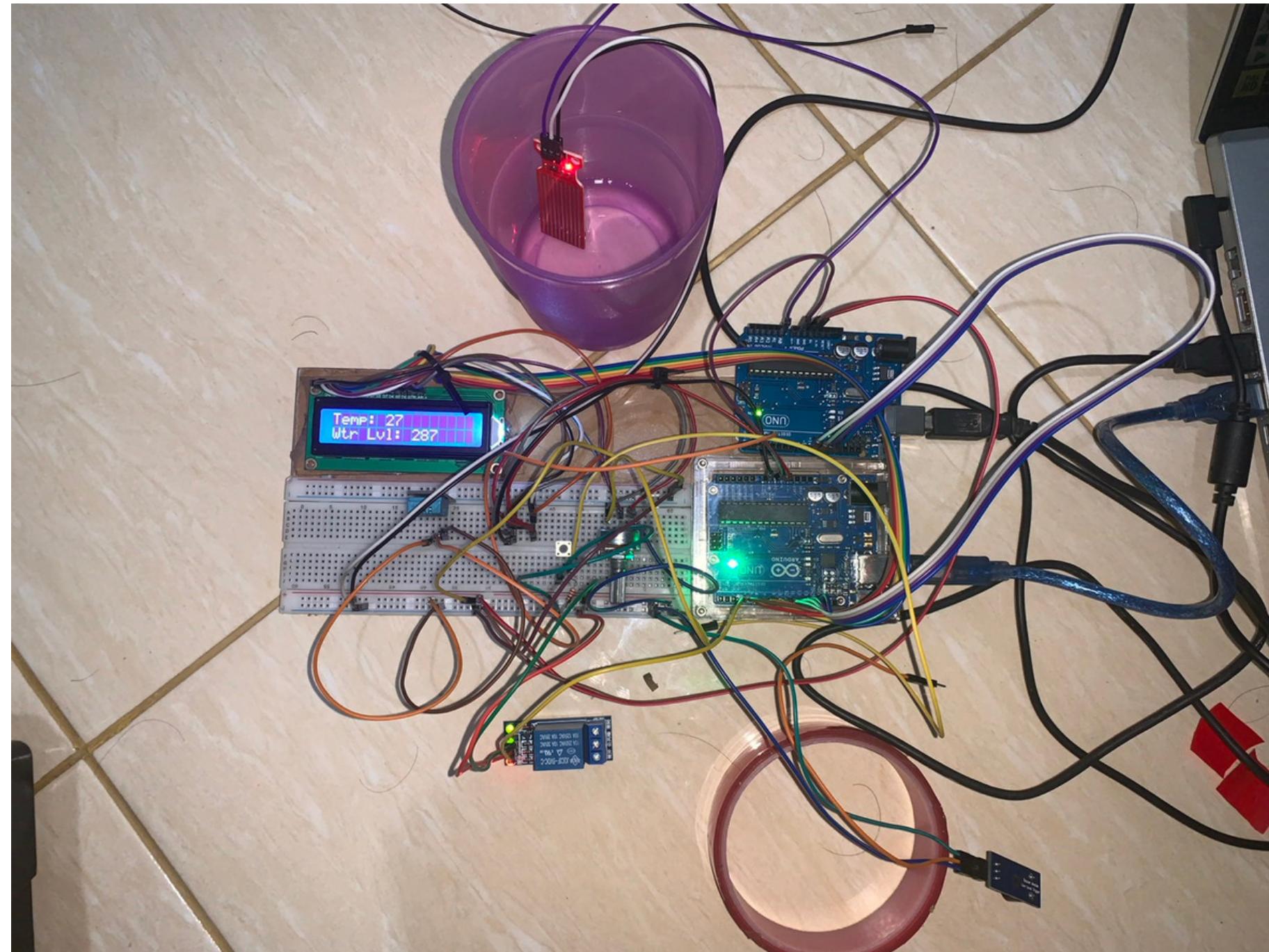
The first condition to test is a scenario with a low water level and a high temperature. In this case, the water level in the condensate container is below the predefined threshold (under 400 value), indicating that it does not require attention. The temperature in the room is above the desired range (above 22 Celcius), signifying the need for cooling. During this test, the expected output from Thermosmart is as follows. The LED indicator turns ON and the AC is turned ON to cool the room and maintain a comfortable temperature for the occupants.





GROUP A6

Pitch Deck Presentation



Results Low Water Level & High Temperature

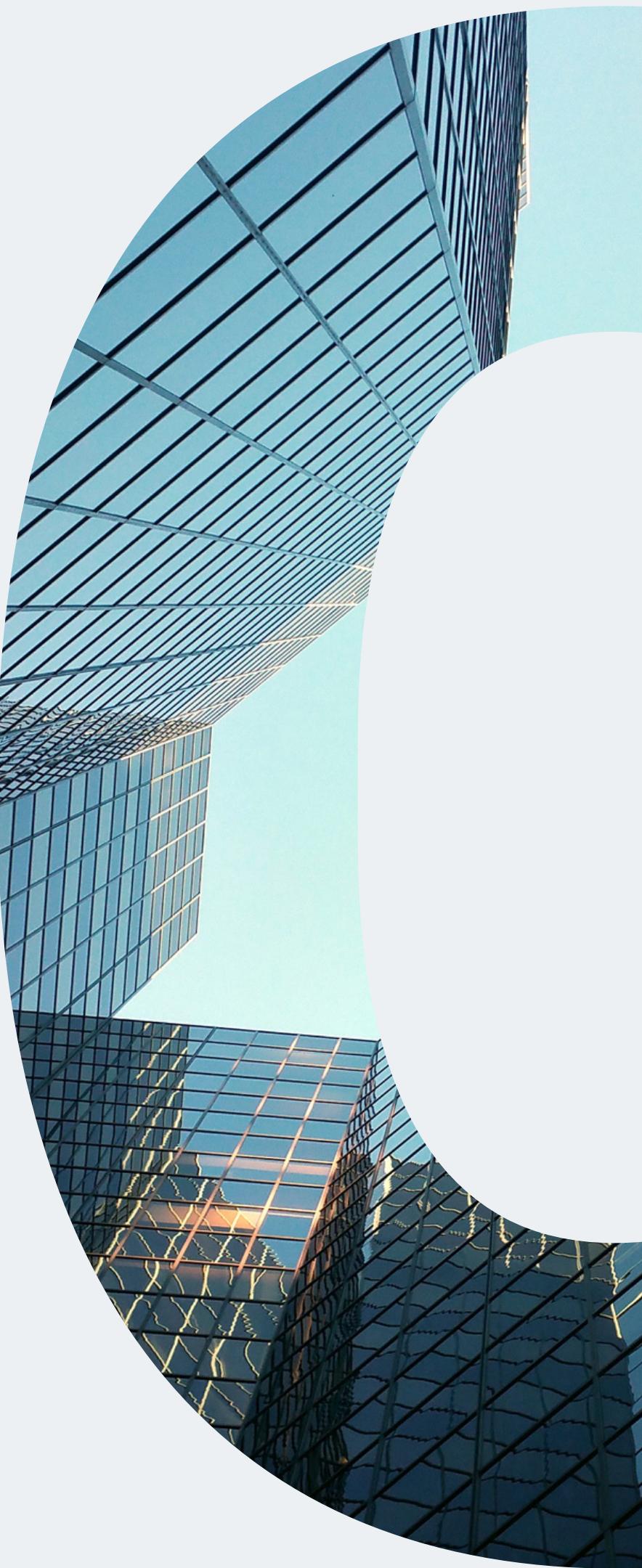
TESTING AND EVALUATION

TESTING AND EVALUATION

Testing and Results

High Water Level & High Temperature

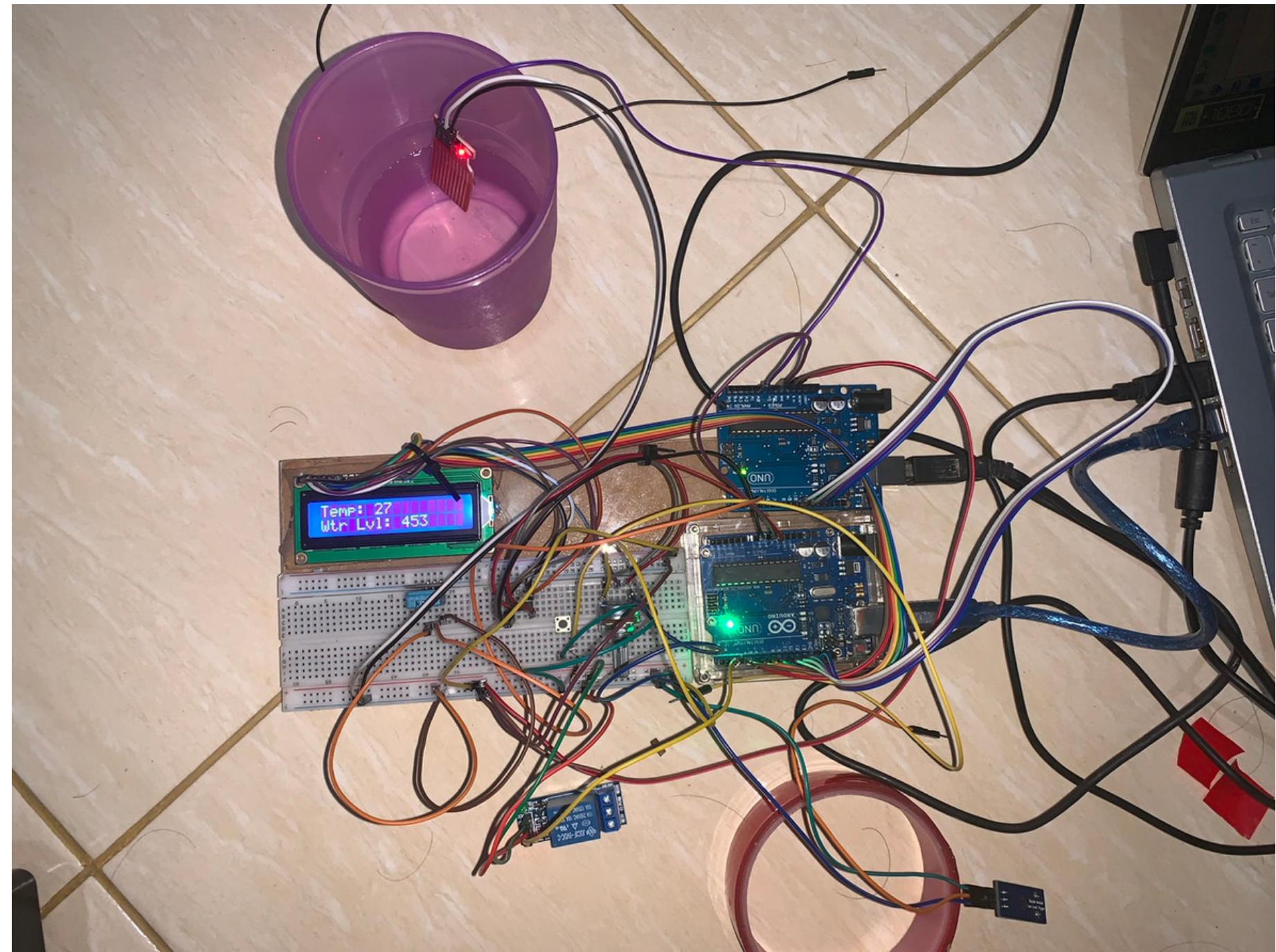
During this test, the expected output from Thermosmart is as follows. The LED indicator turns OFF, indicating that the water level is high and has reached its maximum capacity. Also the temperature is above the desired range. This serves as a visual alert to the user to promptly empty the container to prevent overflow or potential damage. Additionally, the AC is turned OFF to prevent further cooling and stopping the water leakage from the container. This condition will also trigger a buzzer to notify the user that the drainage tank is Full.





GROUP A6

Pitch Deck Presentation



Results High Water Level & High Temperature

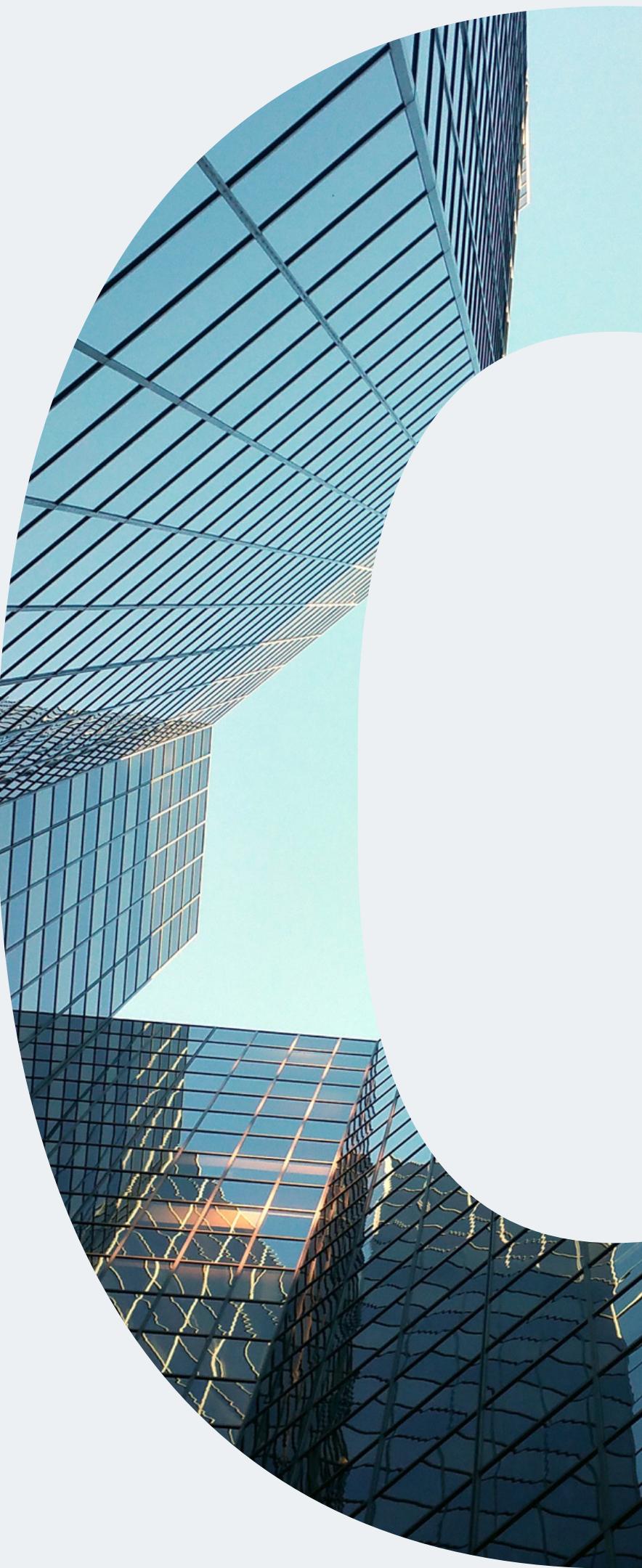
TESTING AND EVALUATION

TESTING AND EVALUATION

Testing and Results

Low Water Level & Low Temperature

During this third condition test, the expected output from Thermosmart is as follows. The LED indicator turns OFF, indicating the low water level that does not require attention. However, since the temperature is already within the desired range, the AC remains OFF. This ensures energy efficiency by not activating unnecessary cooling when the temperature is already comfortable.

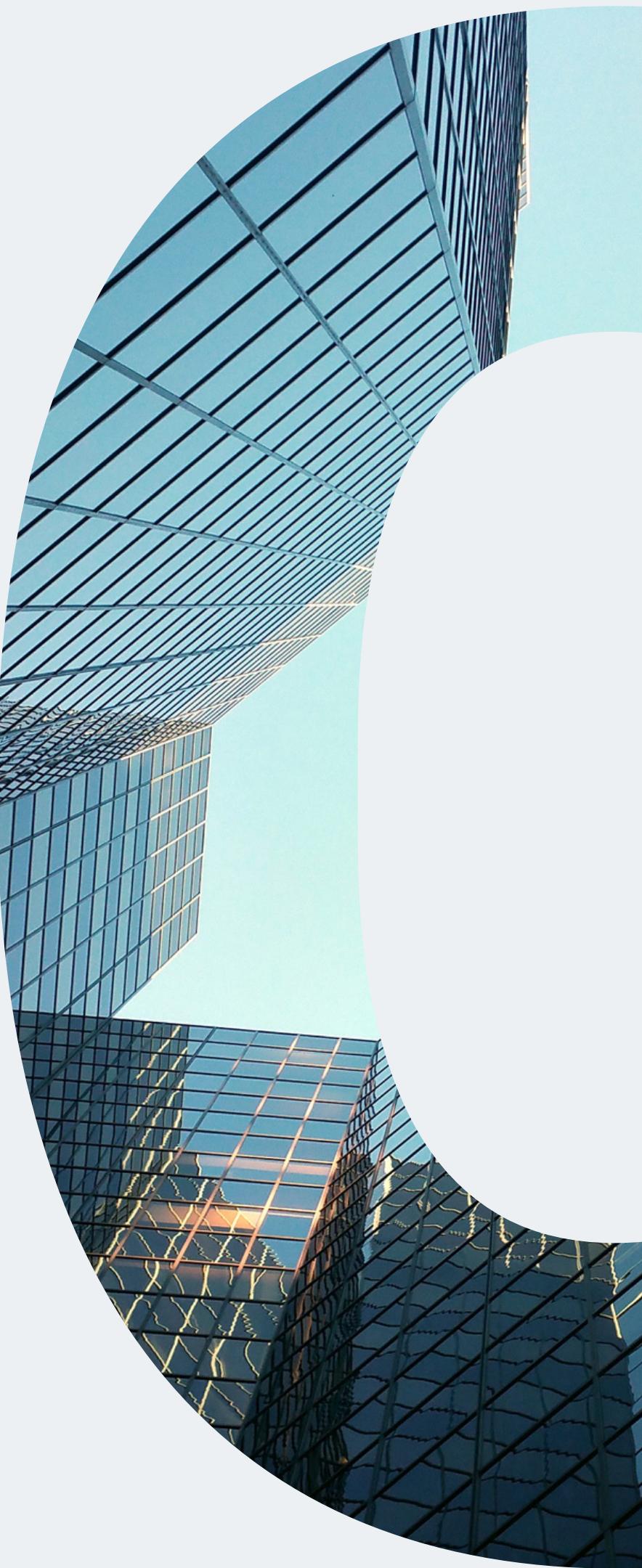


TESTING AND EVALUATION

Testing and Results

High Water Level & Low Temperature

During this fourth condition test, the expected output from Thermosmart is as follows. The LED indicator turns OFF, indicating the high water level that requires prompt action. Since the temperature is already within the desired range, the AC remains OFF as cooling is not necessary and stopping the water leakage from the container.



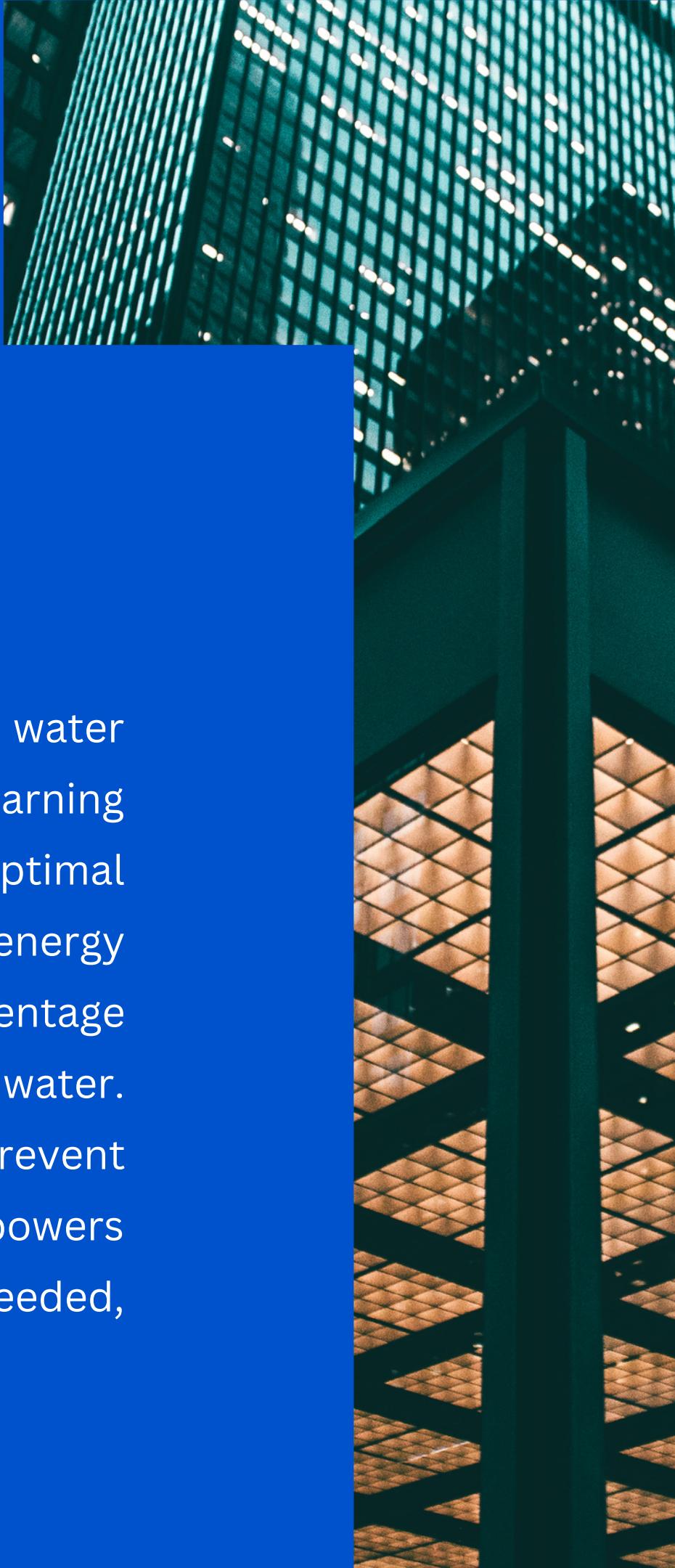


TESTING AND EVALUATION

Evaluation

An evaluation of the Serial Peripheral Interface (SPI) communication in both Proteus and physical circuit reveals a discrepancy in their behavior. In Proteus, the SPI communication can run smoothly, allowing multiple transmissions within a loop. However, in the physical circuit implementation, the SPI communication only sends data once, and subsequent transmissions do not occur as expected. To further enhance the efficiency and effectiveness of the smart control system, it is crucial to evaluate and improve the system's logic. By incorporating advanced algorithms or machine learning techniques, the system can adapt more accurately to occupants' preferences and maximize energy efficiency. Additionally, implementing a mechanism to display the water level in percentage using assembly language simplifies understanding and enables timely management of condensate water. Introducing an enhanced interrupt button allows users to forcefully activate the air conditioner when needed, regardless of system logic, while adjusting the settings to allow the AC to continue operating when the storage container reaches maximum capacity ensures uninterrupted cooling. These improvements enhance user control, comfort, and safety within the environment.





CONCLUSION

Conclusion

Thermosmart offers a comprehensive solution for air conditioner temperature and condensate water management. By evaluating and refining the system's logic, incorporating advanced algorithms, learning techniques, Thermosmart could achieve enhanced efficiency and effectiveness in maintaining optimal temperature control. This ensures a comfortable environment for occupants while maximizing energy efficiency. Additionally, the implementation of a mechanism to display the water level in percentage using assembly language enhances usability and facilitates timely management of condensate water. With Thermosmart, users can easily monitor the water level and take appropriate actions to prevent overflow or potential damage. Moreover, the integration of an enhanced interrupt button empowers users to have more control over the air conditioner. This allows for immediate cooling when needed, regardless of the system's predefined thresholds.



GROUP A6

Thank You