FATIMA JINNAH WOMEN UNIVERSITY

GROUP MEMBERS:

- MOOMNA ATIQ (2023-BCS-065)
- AMNA PERVEZ (2023-BCS-010)
- ZAHRA BASHIR (2023-BCS-092)
- INSA SALEEM (2023-BCS-043)

PROJECT TITLE:

WEATHER MONITORING SYSTEM

SUBMITTED TO :

SIR MAJID SHAFIQUE

SECTION:

BCS 4 A

PROJECT REPORT:

Weather Monitoring System Using Arduino UNO and RS232 Communication

Project Overview

This project creates a simple weather monitoring system using Arduino UNO without real sensors. Instead, it simulates temperature and humidity data and shows the temperature range using three LEDs (Green, White, Red). The system uses the MAX232 module to convert Arduino's TTL serial data to RS232 levels, allowing data transfer to a computer.

Components Required

- **Arduino UNO**: Main controller generating simulated data.
- **MAX232 Module**: Converts Arduino's TTL serial output to RS232 voltage levels.
- **Blue LED**: Indicates high temperature (< 30°C).
- **White LED**: Indicates low temperature (>30°C).
- **Resistors (470Ω)**: Current limiting for LEDs.
- **Breadboard and Jumper Wires**: For circuit assembly.
- **RS232 Cable**: Connects MAX232 to PC serial port.

RS232 Communication

The MAX232 module converts the Arduino's TTL serial signals (0-5V) into RS232 standard voltage levels suitable for longer cable lengths and connection to PC serial ports. This allows the Arduino to send simulated weather data to a computer terminal program like PuTTY or HyperTerminal for monitoring.

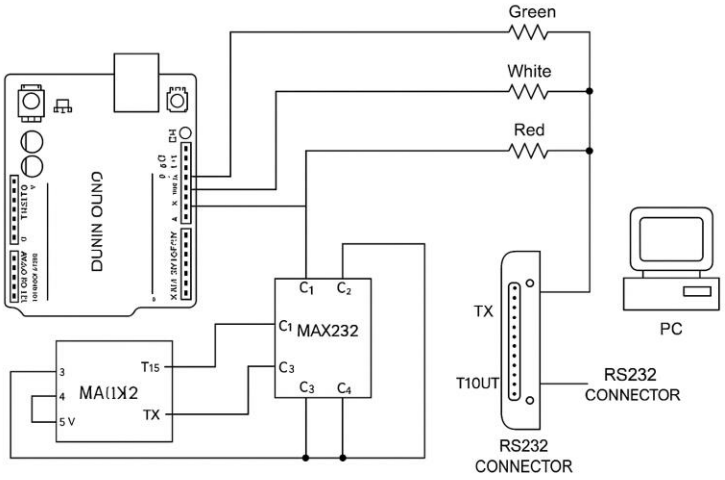
Testing and Operation

1. Connect components according to the circuit diagram.
2. Upload the Arduino code using Arduino IDE.
3. Open a terminal program on PC, set baud rate to 9600.
4. View simulated temperature and humidity data streamed via RS232.
5. Observe LEDs light up based on temperature conditions.

Troubleshooting Tips

- Verify all wiring, especially LED polarity and resistor connections.
- Check MAX232 power and capacitor connections per its datasheet.
- Use correct COM port and baud rate in your PC terminal program.
- Ensure Arduino is powered and code is uploaded successfully.
-

Circuit Diagram



Arduino Code

```
#define LED1 13  // LED for temperature >= 30°C

#define LED2 12  // LED for temperature < 30°C

String inputString = "";  // to hold incoming data

boolean stringComplete = false;

void setup() {

  Serial.begin(9600);

  pinMode(LED1, OUTPUT);

  pinMode(LED2, OUTPUT);

  Serial.println("Temperature Monitoring System Ready.");

  Serial.println("Enter temperature value (e.g., 28 or 35):");

}

void loop() {
```

```
  // Read complete serial input as string
  if (stringComplete) {
    int temperature = inputString.toInt();  // Convert input to integer
    Serial.print("Received Temperature: ");
    Serial.println(temperature);
    if (temperature >= 30) {
      digitalWrite(LED1, HIGH);  // LED1 ON (High Temp)
      digitalWrite(LED2, LOW);   // LED2 OFF
      Serial.println("Status: HIGH TEMPERATURE (>= 30°C)");
    }
    else {
      digitalWrite(LED1, LOW);   // LED1 OFF
      digitalWrite(LED2, HIGH);  // LED2 ON (Low Temp)
      Serial.println("Status: NORMAL TEMPERATURE (< 30°C)");
    }
    // Reset for next input
    inputString = "";
    stringComplete = false;
    Serial.println("Enter next temperature value:");
  }
  }
// SerialEvent occurs whenever new data comes
void serialEvent() {
```

```
  while (Serial.available()) {
    char inChar = (char)Serial.read();
    if (inChar == '\n') {
      stringComplete = true;
    } else {
      inputString += inChar;
    }
  }
}
```

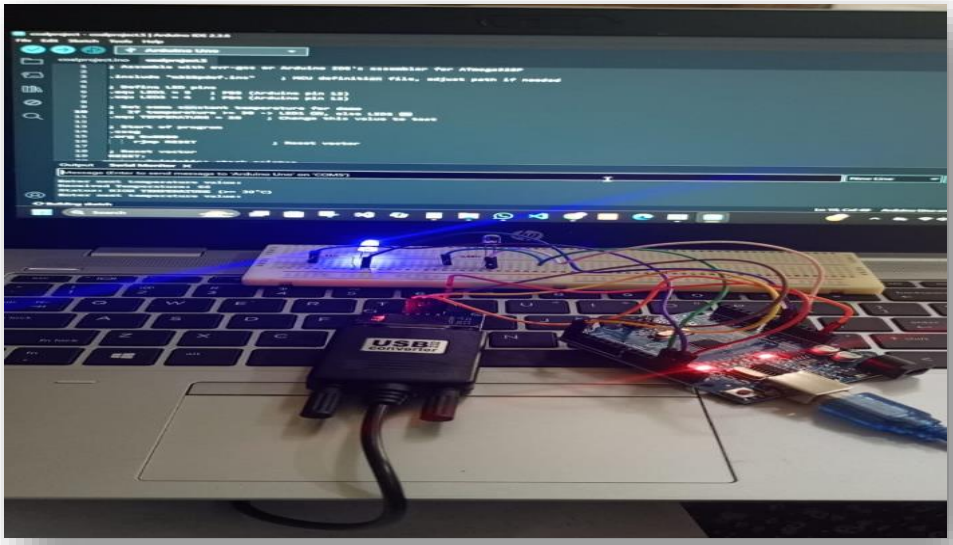
Integrate both code assembly language and C code:

```
.include "m328pdef.inc"    ; MCU definition file, adjust path if needed
; Define LED pins
.equ LED1 = 5   ; PB5 (Arduino pin 13)
.equ LED2 = 4   ; PB4 (Arduino pin 12)
; Set some constant temperature for demo
;  If temperature >= 30 -> LED1 ON, else LED2 ON
.equ TEMPERATURE = 28    ; Change this value to test
; Start of program
.cseg
.org 0x0000
    rjmp RESET            ; Reset vector
; Reset vector
RESET:
```

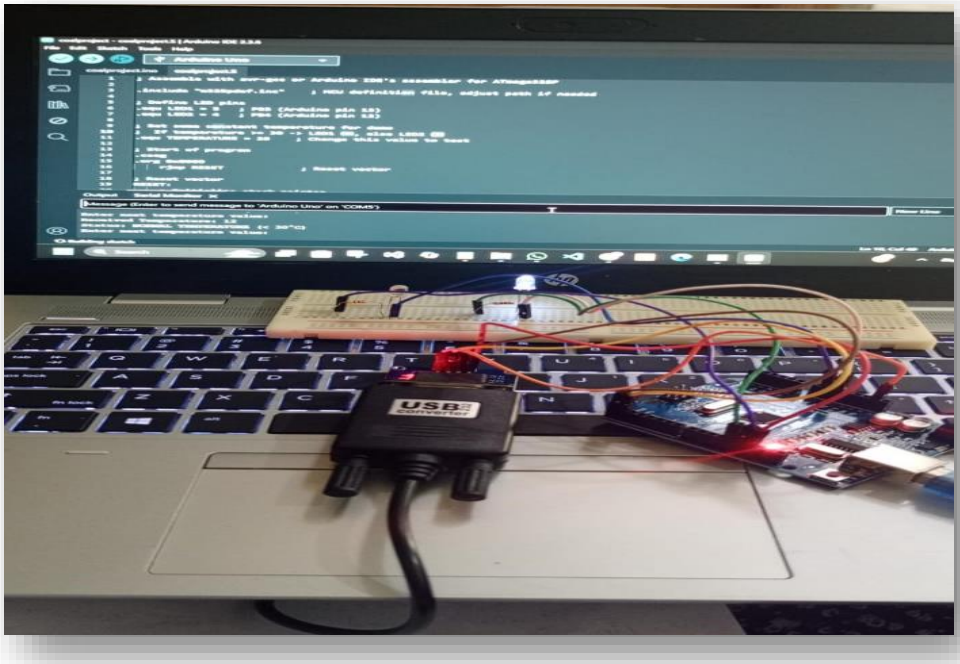
```
    ; Initialize stack pointer
    ldi r16, low(RAMEND)
    out SPL, r16
    ldi r16, high(RAMEND)
    out SPH, r16
    ; Configure PB4 and PB5 as outputs
    sbi DDRB, LED1       ; Set PB5 as output
    sbi DDRB, LED2       ; Set PB4 as output
    ; Check temperature and control LEDs
    ldi r16, TEMPERATURE
    cpi r16, 30          ; Compare temperature with 30
    brlt TURN_ON_LED2    ; If less than 30, jump to TURN_ON_LED2
TURN_ON_LED1:
    sbi PORTB, LED1      ; Turn ON LED1 (PB5)
    cbi PORTB, LED2      ; Turn OFF LED2 (PB4)
    rjmp LOOP
TURN_ON_LED2:
    cbi PORTB, LED1      ; Turn OFF LED1 (PB5)
    sbi PORTB, LED2      ; Turn ON LED2 (PB4)
LOOP:
    rjmp LOOP            ; Infinite loop to keep program running
```

Output:

When temperature is high:



When temperature is low:



Conclusion

This project demonstrates a basic weather monitoring simulation using Arduino, LED indicators, and RS232 communication with the MAX232 module. It is an excellent beginner-level introduction to serial communication and hardware interfacing without requiring complex sensors or LCDs. The setup can be expanded by adding real sensors or more output options in the future.

References

1. **Arduino Official Website — https://www.arduino.cc**
2. **RS232 to TTL Conversion https://www.max232.com/**