Table of Contents

Introduction to project and software APP developed in CENG 322;

In response to the critical concern of infant safety, we've created a non-invasive, precise biosensor bracelet featuring a TMP006 temperature sensor. This device combines comfort with safety and offers real-time infant temperature monitoring, efficiently designed for low power consumption and digital communication.

Our innovation extends to the software—a parallel Android app linked via Firebase. It processes temperature data and delivers instant caregiver alerts, forming a robust approach to infant safety.

This report explores hardware development, key components, and their significance in infant care, alongside software intricacies and seamless hardware-software integration, ultimately providing caregivers with peace of mind.

Project Proposal:

We propose a Monitoring and Alert System for Infant Safety, addressing Sudden Infant Death Syndrome (SIDS). Key components include a biosensor bracelet and an Android app for real-time data processing and alerts.

Objectives:
Hardware Development: Create the biosensor bracelet.
Software Development: Design an Android app to interface with the bracelet.
Data Transmission: Establish secure data transfer via Firebase.
User Interface: Develop a user-friendly interface for caregivers.

Timeline:
Weeks 1-2: Research and requirements.
Weeks 3-8: Hardware and software development.
Weeks 9-10: Testing and calibration.
Weeks 11-12: Integration, testing, and troubleshooting.
Weeks 13-14: Finalization and documentation.
Benefits: Enhanced infant safety, SIDS prevention, technology application in healthcare, and caregiver confidence.

Approval:
We seek approval and support for this project, aiming to contribute significantly to infant safety.

Introduction to CPU board, the sensors/components, and their functions;.

The CPU Board (Raspberry Pi):

Central to our initiative is the Raspberry Pi, our CPU board of choice. Serving as the brains behind the operation, it orchestrates a symphony of functions, ranging from data acquisition to real-time alerts. The Raspberry Pi's essential components include:

Microcontroller Unit (MCU):
As the project's orchestrator, the MCU is tasked with executing instructions and managing data flow. It interfaces with the chosen sensors, processes temperature data, and facilitates communication with the software application.

Connectivity Interfaces:
Ethernet, Wi-Fi, or Bluetooth interfaces empower the Raspberry Pi to establish seamless connections with external systems. This allows for the swift transmission of data to platforms such as Firebase or mobile applications, ensuring real-time alerting.

Power Supply Unit (PSU):
The PSU plays a critical role in maintaining stability. By regulating and distributing power, it ensures that the system operates reliably.

TMP006 Temperature Sensor:
A cornerstone of our design, this non-contact infrared sensor accurately measures an infant's temperature without physical contact. It excels in capturing the infrared radiation emitted by the infant, converting it into an electrical signal proportional to the temperature.

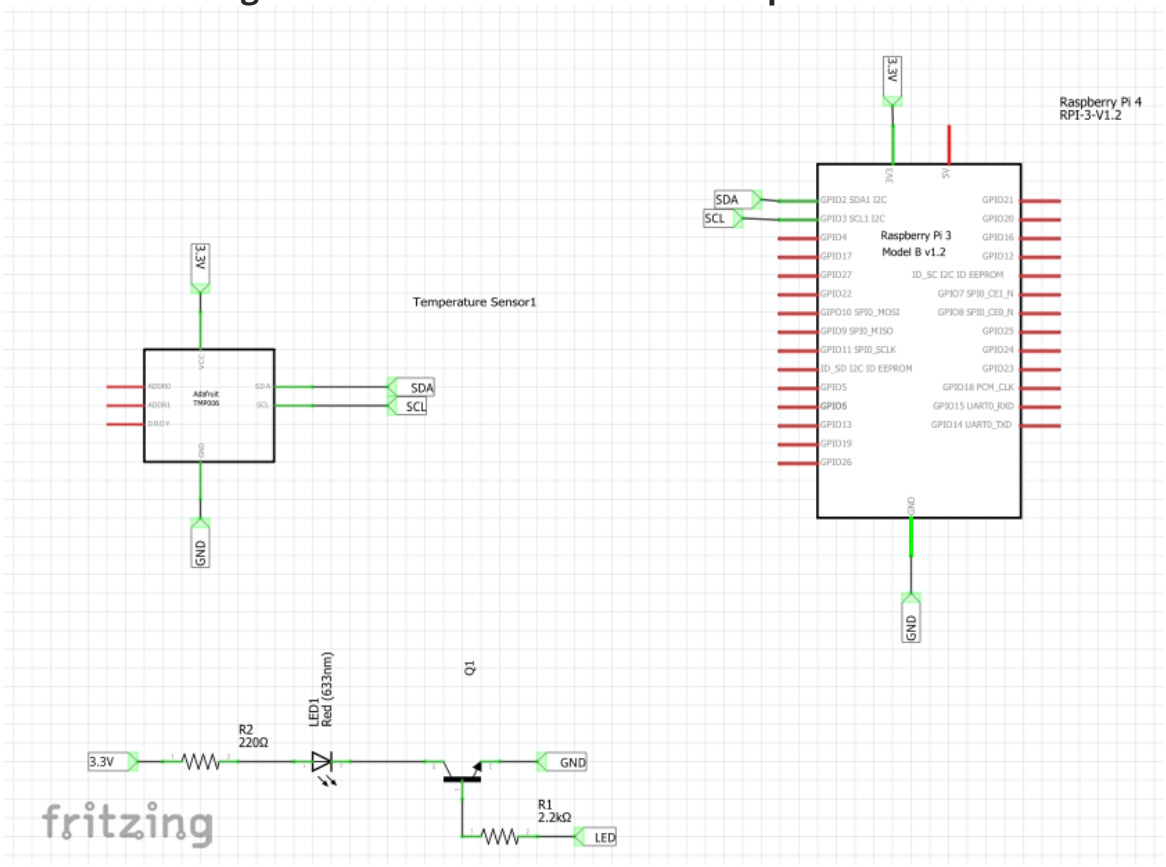
Transistor:
This component acts as a switch, controlled by the Raspberry Pi, to manage current flow to the LED for temperature alerts.

Resistors:
(2.2k ohms and 220 ohms):
The 2.2k-ohm resistor scales voltage for precise temperature measurements, while the 220-ohm resistor protects the LED by regulating its current.

LED (Light Emitting Diode):
This real-time visual indicator responds to TMP006 temperature readings, displaying various colors or patterns for immediate caregiver alerts.

Together, these components and the Raspberry Pi synergize to create a comprehensive monitoring and alert system dedicated to infant safety. In the subsequent sections, we delve deeper into the roles and interactions of these components, revealing their intricate contributions to our innovative project.

Schematic design of the hardware and its description.



TMP006 Sensor with Raspberry Pi, LED, Transistor, and Resistors

In my setup, I've integrated a TMP006 temperature sensor, Raspberry Pi (as the CPU), LED, transistor, and resistors to create a robust monitoring and alert system.

TMP006 Sensor:

- V+ (Power Supply Voltage): I've connected it to the Raspberry Pi's 3.3V supply.
- V- (Ground): It's linked to the Raspberry Pi's ground (GND).
- SDA (Serial Data Line): I've connected it to the Raspberry Pi's GPIO2 for data.
- SCL (Serial Clock Line): It's linked to the Raspberry Pi's GPIO3 for clock synchronization.
- ALERT (Alert Output): Optionally, it can connect to the Raspberry Pi's GPIO17 for temperature alerts.

Transistor:

- Transistor Base: I've connected it to the Raspberry Pi's GPIO18 for transistor control.
- Transistor Emitter: It's linked to the Raspberry Pi's ground (GND).
- Transistor Collector: It connects to the LED's anode.

Resistors:
- 2.2k-ohm Resistor: It limits current to the transistor's base and connects between the TMP006's SDA/SCL junction and the transistor base.
- 220-ohm Resistor: This resistor safeguards the LED from overcurrent and connects between the TMP006's ALERT/SCL junction and the LED's anode.

LED (Light Emitting Diode):
- LED Cathode: I've connected it to the transistor's collector.
- LED Anode: It links to the junction between the 220-ohm resistor and the transistor collector.

This setup enables temperature monitoring by the TMP006 sensor. When specific conditions are met, the Raspberry Pi activates the transistor, controlling the LED for real-time visual alerts. This schematic forms the foundation of our effective monitoring and alert system.

The PCB design and the testing procedures;

PCB Design:

In the hardware development phase, I designed a customized Printed Circuit Board (PCB) to integrate components efficiently. The PCB serves as the system's backbone, enhancing reliability and compactness.

Key PCB features:

Component Placement:
Components like the TMP006 sensor, Raspberry Pi headers, transistor, resistors, and the LED were carefully positioned for optimal functionality and space efficiency.

Signal Routing:
The PCB efficiently routes data and power connections, minimizing interference and maximizing efficiency.

Power Distribution:
Power distribution circuits ensure stable and efficient power supply to each component.

Size Optimization:
The PCB's compact design suits environments with limited space.

Regarding testing procedures:

Functional Testing:
Comprehensive tests confirmed proper data acquisition, processing, and LED functionality in response to temperature thresholds.

Temperature Calibration:
Calibration procedures validated accurate temperature readings from the TMP006 sensor across various ranges.

Transistor Control:
Testing ensured the transistor effectively controlled the LED in response to temperature conditions.

Voltage and Current Measurement:
Measurements verified proper voltage supply and safe component operation.

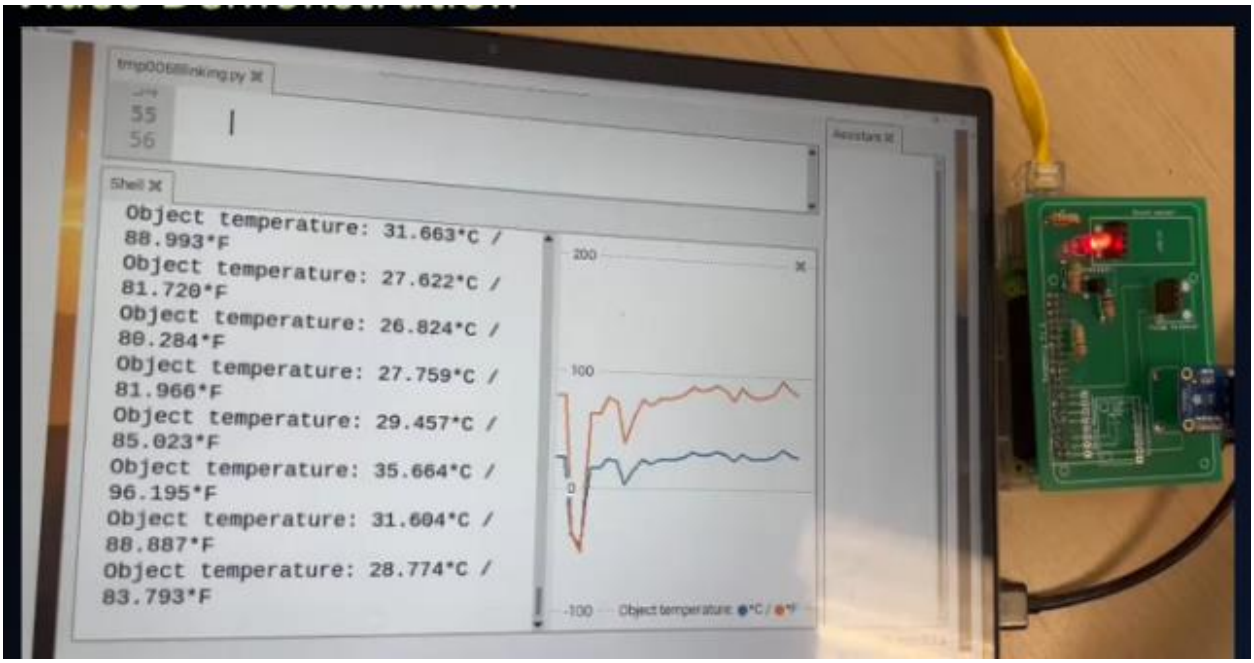
Alert Triggering:
The system reliably activated the LED when temperature anomalies were detected.

These tests ensured the system's reliability and accuracy under varying conditions, contributing to its overall robustness.

GitHub website for the project.

https://github.com/ZoyebaMahbub5837/InfantHealthMonitor.git

Testing results on the PCB boards and the screenshots of the outputs



The troubleshooting procedure
Troubleshooting Procedures:

Component Isolation: I isolated components to identify issues, testing each one independently.

Signal Checks: I ensured data and power connections' integrity, inspecting for loose or damaged connections.

Data Analysis: Data logs and error messages were analyzed to pinpoint communication issues.

Voltage and Current Measurements: I measured these parameters to identify power-related problems.

Temperature Sensor Calibration: Calibration parameters were reviewed to ensure accurate temperature measurements.

Code Inspection: I scrutinized code for errors and logic flaws.

Lessons Learned:

Component Selection: Prioritize reliable, well-documented components.

Robust Testing: Extensive testing, including stress tests, is essential.

Communication Reliability: Ensure reliable inter-component communication with error handling.

Documentation: Clear documentation aids in quicker issue resolution.

Collaboration: Effective teamwork and communication are vital in troubleshooting.

These experiences have emphasized rigorous testing, documentation, and collaboration for project success

Code:

```
import time
import board
import busio
import adafruit_tmp006
import RPi.GPIO as GPIO
import threading

# Define a function to convert Celsius to Fahrenheit.
def c_to_f(c):
    return c * 9.0 / 5.0 + 32.0

# Set up the LED and GPIO pins
led_pin = 21  # Change this to the GPIO pin number you used for the LED circuit
threshold_temp = 30.0  # Set the threshold temperature in Celsius

GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)

# Create library object using our Bus I2C port
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_tmp006.TMP006(i2c, address=0x40)

blinking = True

# Function for LED blinking
def blink_led():
    while blinking:
        GPIO.output(led_pin, GPIO.HIGH)
        time.sleep(0.2)
        GPIO.output(led_pin, GPIO.LOW)
        time.sleep(0.2)

# Start the LED blinking function in a separate thread
blink_thread = threading.Thread(target=blink_led)
blink_thread.start()

try:
```

```
    while True:
        obj_temp = sensor.temperature
        print("Object temperature: {:.3F}*C / {:.3F}*F".format(obj_temp,
c_to_f(obj_temp)))

        if obj_temp > threshold_temp:
            blinking = False
            GPIO.output(led_pin, GPIO.HIGH)  # Turn on the LED (solid light)
        else:
            blinking = True

        time.sleep(0.2)

except KeyboardInterrupt:
    blinking = False
    GPIO.cleanup()
```