Temperature-Initiated Object Detection Using TM4C123G: Real-Time Sensing and Feedback System

Elif Ceren Yılmaz
Electrical – Electronics Engineering
Middle East Technical University
Ankara, Turkey
elifcerenn.yilmaz@gmail.com

Çetincan Önelge
Electrical – Electronics Engineering
Middle East Technical University
Ankara, Turkey
cetincanonelge@gmail.com

***Abstract*— The Temperature-Initiated Object Detection project demonstrates a robust system capable of detecting environmental changes and reacting in real-time. Utilizing the TM4C123G microcontroller, the system integrates multiple sensors and peripherals, including LM35, BMP280, and HC-SR04, to achieve precise monitoring and object detection. By leveraging advanced communication protocols and an interrupt-driven design, the project fulfills its objectives efficiently and reliably.**

Keywords*— *Temperature detection, object detection, TM4C123G, I2C, SPI, real-time processing

I. INTRODUCTION

The Temperature-Initiated Object Detection project integrates various hardware components and software modules, demonstrating the capabilities of the TM4C123G microcontroller. This intelligent system detects temperature changes and scans the environment for objects, ensuring functionality in real-time applications. The key objectives of the project include:

1. Efficient Deep Sleep Mode:

The system minimizes power consumption by entering a deep sleep state and utilizes an analog comparator to wake up when temperature thresholds are exceeded.

2. Accurate Heat Sensing:

Integration of the LM35 analog and BMP280 digital temperature sensors ensures precise temperature monitoring, enabling a reliable wakeup mechanism.

3. Comprehensive Object Detection:

An ultrasonic sensor (HC-SR04) mounted on a stepper motor scans the environment within a specified angle range to detect objects.

4. Interactive User Interface:

A Nokia 5110 LCD, RGB LEDs, keypad, and speaker enhance the user interaction by displaying relevant data and providing audio-visual feedback.

5. Real-Time Processing:

Leveraging the I2C and SPI protocols along with timer modules ensures real-time data acquisition and processing.

The project adheres to specified requirements, utilizing modular design and interrupt-driven programming to achieve a robust and efficient system. This report elaborates on the project's architecture, implementation, and its alignment with the stated goals.

II. SYSTEM ARCHITECTURE AND DESIGN

The system architecture of the Temperature-Initiated Object Detection project is designed to ensure seamless interaction between hardware components and software modules. It integrates sensors, actuators, and user interface elements with the TM4C123G microcontroller to achieve reliable and efficient operation.

A. Hardware Components

1. Microcontroller: The TM4C123G serves as the central processing unit, orchestrating all system operations. It manages communication between peripherals and handles the core logic for temperature monitoring and object detection.
2. Temperature Sensors:
   - 2.1. LM35: An analog sensor connected to the analog comparator module. It provides a continuous voltage output proportional to the sensed temperature.
   - 2.2. BMP280: A digital temperature sensor interfaced via I2C. It offers high-precision measurements and acts as a secondary validation sensor.
3. Ultrasonic Distance Sensor: The HC-SR04 sensor detects object proximity within a 1-meter range. It operates by emitting ultrasonic pulses and measuring the time delay of the echo.
4. Stepper Motor: A stepper motor rotates the ultrasonic sensor, allowing the system to scan an angle range of -90° to 90° for object detection.
5. Display and Indicators:
   - 5.1. Nokia 5110 LCD: Displays real-time data, including temperature, object distance, and angle.
   - 5.2. RGB LEDs and Power LED: Provide visual feedback based on object distance.
   - 5.3. Speaker: Alerts the user when the temperature exceeds the threshold.
6. Input Devices:
   - 6.1. Keypad: Allows digital temperature threshold adjustment.
   - 6.2. Trimpot: Sets the analog temperature threshold for the LM35 sensor.
   - 6.3. Pushbuttons: Control system states, including entering deep sleep mode.

B. *Software Design*

1. Modular Programming: The software is structured into multiple modules to ensure reusability and clarity. Key modules include:
   - 1.1. main.c: Integrates all functionalities.
   - 1.2. lm35_control.h: Manages LM35 sensor and analog comparator.
   - 1.3. printHelper.h: Serial terminal print module via UART for debugging. This module utilizes OutChar.s inside the directory.
   - 1.4. bmp280.h: Handles BMP280 data acquisition and processing.
   - 1.5. Stepper_Scan.h: Controls stepper motor and object detection logic.
   - 1.6. DistanceSensor.h: Manages ultrasonic sensor operations.
   - 1.7. LedSpeaker.h: Controls Power LED and speaker.
   - 1.8. plot.h: LCD object mapping module after the scan is completed.
   - 1.9. Nokia5110.c: LCD source file created by Prof. Daniel Valvano [1]
   - 1.10. Nokia5110.h: LCD header file created by Prof. Daniel Valvano [1]
2. Interrupt-Driven Design: Interrupts are used extensively to ensure efficient handling of sensor data and system states. For example, the analog comparator interrupt wakes the system from deep sleep, while the timer and keypad interrupts manage precise timing and user input
3. Communication Protocols:
   - 3.1. I2C: Used for communication with the BMP280 sensor.
   - 3.2. SPI: Used to drive the Nokia 5110 LCD.
4. Control Logic:
   - 4.1. In deep sleep mode, the system monitors the LM35 sensor for threshold violations.
   - 4.2. Upon waking, it verifies the temperature using BMP280, activates the ultrasonic sensor for object scanning, and provides visual and audio feedback based on the results.

C. *System Workflow:*

1. Deep Sleep Monitoring:

The system stays in low-power mode, monitoring temperature via the LM35 sensor. If a threshold violation is detected, the analog comparator triggers an interrupt.

2. Temperature Verification:

The BMP280 sensor verifies the temperature reading to prevent false positives.

3. Object Detection:

The ultrasonic sensor scans the environment, and data is processed to identify objects and their distances.

4. User Feedback:

Results are displayed on the LCD, and RGB LEDs and speaker provide additional alerts.

This systematic integration ensures that the project meets all functional and performance requirements, providing a robust and user-friendly temperature-initiated object detection system.

The overall workflow of the system is illustrated in Figure 1, showcasing the interactions between sensors, the TM4C123G microcontroller, and feedback mechanisms.

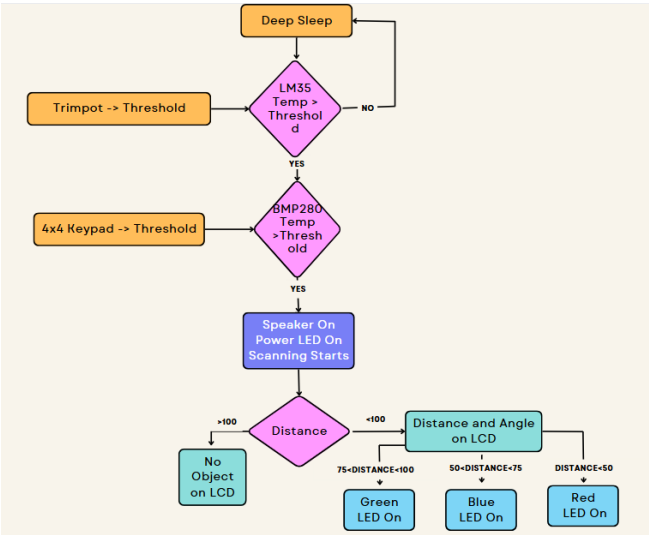

FIGURE 1: SYSTEM WORKFLOW

III. HARDWARE IMPLEMENTATION

The hardware implementation of the Temperature-Initiated Object Detection project involved integrating multiple components to create a cohesive system. The following subsections detail the setup and configuration of each hardware element.

A. *Microcontroller Setup*

The TM4C123G microcontroller serves as the central unit for managing all components. Key aspects of its configuration include:

- 4.1. Enabling necessary GPIO ports for sensor and actuator interfacing.
- 4.2. Configuring analog and digital peripherals for accurate data acquisition.
- 4.3. Initializing communication protocols such as I2C and SPI for sensor and LCD interfacing.
- 4.4. All utilized IO pins can be seen in Table 1.

TABLE 1: I/O PIN CONFIGURATION OF MCU

I/O Pins	***Submodule***	***Special Function***
PA2,3,5	LCD	SSI0
PA6,7	LCD	-
PB0,1	Power LED & Speaker	-
PB2,3	BMP280	I2C
PC4	HC-SR04	WT0CCP0

I/O Pins	*Submodule*	*Special Function*
PC6,7	Analog Comparator (LM35 & Trimpot)	C0+ & C0-
PD0,1,2,3	Step Motor	-
PE1,2,3,4	Push buttons	-
PF2	HC-SR04	-

B. Temperature Sensors

1. LM35 Sensor:
   1.1. Connected to the analog comparator module for continuous monitoring of temperature.
   1.2. Configured to trigger interrupts upon exceeding the threshold set by the trimpot.
   1.3. Placed close to BMP280 in order to get accurate readings from the two sensors.
2. BMP280 Sensor:
   2.1. Communicates via I2C to provide high-precision temperature readings.
   2.2. Averages 128 samples to ensure stability and accuracy.
   2.3. Connected two pull-up resistors to both SDA and SCL lines.

Figure 2 illustrates the LM35 and BMP280 sensors configuration for precise temperature measurement.

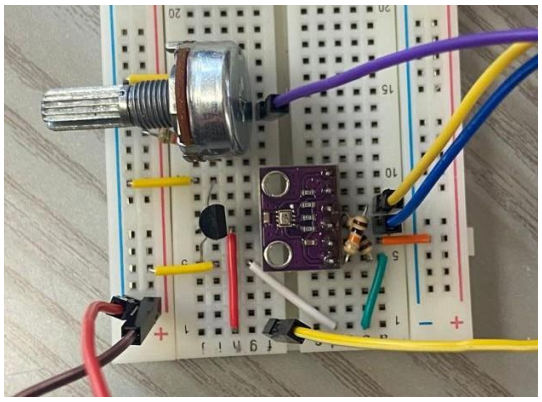

FIGURE 2: TEMPERATURE SENSORS

C. Ultrasonic Distance Sensor

The HC-SR04 ultrasonic sensor is interfaced with the TM4C123G to measure object distances. Key implementation steps include:

1. Configuring trigger and echo pins.
2. Using GPTM (General Purpose Timer Module) to calculate the time delay of the echo signal.
3. Specifically connected to Wide GPTM to utilize 32 bit buffer for correct calculation of distance.

D. Stepper Motor

The stepper motor rotates the ultrasonic sensor for scanning a range of -90° to 90°. Key configurations:

1. Setting up control pins on Port D.
2. Using a predefined step sequence for smooth and precise motor movement.

As shown in Figure 3, the stepper motor enables the ultrasonic sensor to scan the environment for object detection.

FIGURE 3: DISTANCE SENSOR AND STEPPER MOTOR

E. Display and Indicators

1. Nokia 5110 LCD:
   1.1. Displays real-time data such as temperature, object distance, and angle.
   1.2. Driven using the SPI protocol and Daniel Valvano's driver files [1].

Distance and angle data is displayed on the Nokia 5110 LCD, as depicted in Figure 4.

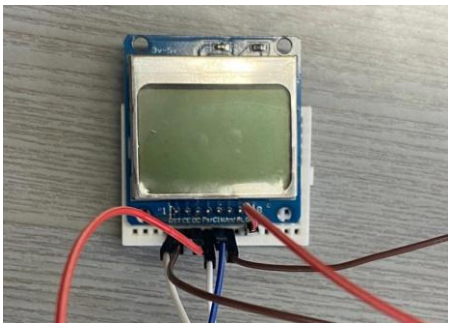

FIGURE 4: NOKIA5110 LCD DISPLAY

2. RGB LEDs and Power LED:
   2.1. Provide visual feedback based on detected object distances.
   2.2. RGB leds are on the MCU and controlled through GPIO pins on Port F.
   2.3. Power LED controlled via a GPIO pin on Port B using square wave signals.
3. Speaker:
   3.1. Generates audio alerts when the BMP280 sensor detects a high temperature.
   3.2. Controlled via a GPIO pin on Port B using square wave signals.

The system's visual and audio feedback mechanisms, including the speaker and power LED, are shown in Figure 5.

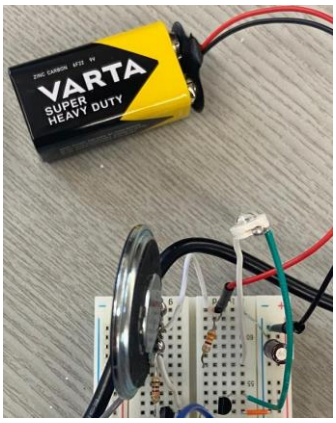

FIGURE 5: SPEAKER AND POWER LED

F. *Input Devices*

1. Trimpot: Connected to C0- pin of analog comparator utilizing simple voltage division from 3.3V.
2. Pushbuttons: Connected to GPIO pin on Port E utilizing keypads own pull-up resistors.

The keypad used for threshold adjustments is shown in Figure 6.

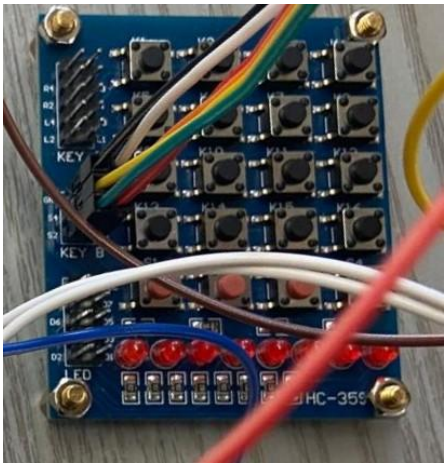

FIGURE 6: 4x4 KEYPAD

G. *Additional Hardware Considerations*

1. Capacitors for Noise Reduction: Decoupling capacitors (e.g., 0.1µF) are placed between the VCC and GND lines of the microcontroller and sensors to reduce electrical noise and ensure stable operation.
2. Transistor Drivers: NPN transistors are used to drive the power LED and speaker to avoid excessive current draw from GPIO pins. External power source (9V) utilized for Power LED & Speaker.
3. Pull-Up Resistors for I2C Lines: As mentioned, 4.7kΩ pull-up resistors are placed on the SDA and SCL lines to ensure proper logic levels for I2C communication.
4. Stepper Motor Driver Circuit: A ULN2003A driver is used to control the stepper motor, providing the necessary current amplification for smooth operation.

The complete hardware setup is summarized in Figure 7.

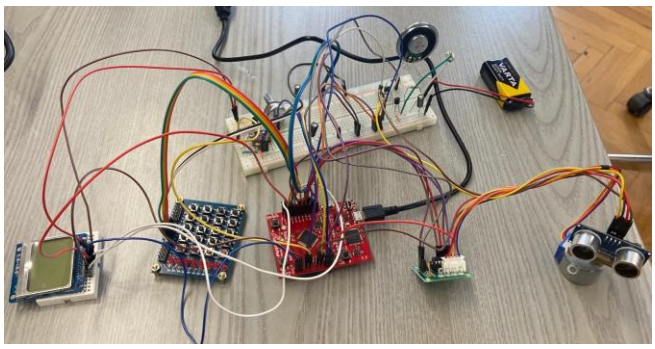

FIGURE 7: OVERALL HARDWARE IMPLEMENTATION

IV. SOFTWARE IMPLEMENTATION

The software implementation for the Temperature-Initiated Object Detection system involves integrating various hardware components with efficient software modules to ensure seamless functionality. Each module is designed to handle specific tasks, from sensor data acquisition to motor control and user interaction, thereby achieving a robust and scalable system.

A. *Programming Structure*

The system is organized into distinct modules to enhance readability, maintainability, and scalability. Each module handles specific functionalities and communicates with others through well-defined interfaces.

The overall structure consists of:

1. *Control Logic (main.c)*
   - 1.1. Initializes all hardware components and software modules. The initialization of hardware components follows the guidelines provided in [2].
   - 1.2. Contains the main control loop and manages the system's operational states, such as deep sleep, scanning, and data processing.
2. *Sensor Management Modules*
   - 2.1. lm35_control.h: Monitors temperature using the LM35 sensor via an analog comparator.
   - 2.2. bmp280.h: Reads and processes temperature data from the BMP280 sensor using I2C communication.
   - 2.3. DistanceSensor.h: Interfaces with the HC-SR04 ultrasonic sensor to measure object distances.
3. *Actuator and Display Modules*
   - 3.1. Stepper_Scan.h: Controls the stepper motor for scanning the environment and integrates object detection logic.
   - 3.2. LedSpeaker.h: Provides visual and audio feedback through LEDs and a speaker.
   - 3.3. Nokia5110.c and Nokia5110.h: Drive the Nokia 5110 LCD for displaying real-time data and visual plots.
4. *Utility and Interaction Modules*
   - 4.1. plot.h: Dynamically plots distances and angles on the LCD.

4.2. keypad.h: Handles user input via a 4x4 keypad for system settings adjustments.

B. Key Algorithms and Functions

1. Analog Comparator Setup (lm35_control.h):

1.1 Monitors the LM35 sensor output to detect temperature threshold violations.

1.2 AnalogComparator_Init() configures the comparator for external input comparison.

1.3 EnterDeepSleep() puts the system in a low-power state, awakening only when temperature thresholds are exceeded.

2. BMP280 Data Acquisition (bmp280.h):

2.1. Reads and compensates temperature data from the BMP280 sensor with high precision.

2.2. Key functions:

2.2.1. BMP280_Init(): Initializes the sensor.

2.2.2. BMP280_ReadTemperature(): Acquires raw temperature data.

2.2.3. BMP280_FilteredTemperature(): Applies a moving average filter for accurate readings.

3. Stepper Motor Control (Stepper_Scan.h):

3.1. Executes a 180-degree scan using a stepper motor to detect objects.

3.2. The motor steps incrementally by a predefined angle. At each step, the TriggerPulse function sends a pulse to the ultrasonic sensor. The Timer captures the Echo signal, and its pulse width and distance is calculated.

4. Ultrasonic Sensor Distance Measurement (DistanceSensor.h):

4.1. Measures object distances using echo pulse timing.

4.2. DistanceSensor_Init() configures the sensor, while WTIMER0A_Handler() processes edge detection to calculate distances.

5. Dynamic Plotting on LCD (plot.h):

5.1. Visualizes data on the Nokia 5110 LCD by mapping distances and angles to screen coordinates.

5.2. Functions like mapAngleToX() and mapDistanceToY() ensure precise graphical representation.

6. Nokia 5110 LCD Display (Nokia5110.c, Nokia5110.h):

6.1. Enables text and graphical output on the LCD.

6.2. Key functions:

6.2.1. Nokia5110_OutString(): Display text.

7. Keypad Interaction (keypad.h):

7.1. Reads user inputs from a 4x4 keypad to adjust system thresholds and modes.

7.2. Keypad_GetPressedButton() identifies and debounces button presses for reliable input.

C. Interrupt Handling

1. I2C0 Interrupt for BMP280 Temperature Sensor

The I2C0 interrupt is used to facilitate real-time temperature readings from the BMP280 sensor. The I2C0_Handler is triggered when the sensor completes a temperature reading. This interrupt enables the system to update the filtered temperature value promptly by calling a filtering function. It ensures smooth and accurate temperature monitoring for further processing.

2. Wide Timer 0 Interrupt for Distance Measurement:

The Wide Timer 0 interrupt, handled by `WTIMER0A_Handler`, is essential for calculating distances using the HC-SR04 ultrasonic sensor. It captures the pulse width of the echo signal, with the rising edge marking the start and the falling edge marking the end. This pulse width is then converted into a distance measurement in centimeters, providing precise real-time distance data.

3. GPIO Port E Interrupt for Push Buttons:

The GPIO Port E interrupt, managed by GPIOE_Handler(), handles user interactions via push buttons. It detects falling edges on the buttons connected to Port E. This interrupt is used to adjust the temperature threshold, enter deep sleep mode

4. Comparator 0 Interrupt for LM35 Temperature Monitoring:

The Comparator 0 interrupt, handled by COMP0_Handler, monitors the LM35 temperature sensor. It triggers when the sensed temperature exceeds the predefined threshold, waking the system from deep sleep mode.

V. RESULTS AND DISCUSSION

A. System Performance

The system demonstrated reliable real-time sensing and feedback capabilities. The integration of multiple sensors with the TM4C123G microcontroller ensured smooth data acquisition and processing. The analog comparator effectively monitored temperature thresholds, and the stepper motor provided consistent and precise environmental scans. The ultrasonic sensor accurately measured distances, and the Nokia 5110 LCD displayed real-time data seamlessly.

B. Accuracy of Measurments

The accuracy of temperature measurements was verified by comparing LM35 and BMP280 sensor readings. The BMP280, with its built-in compensation, provided highly precise data, while the LM35 offered reliable threshold monitoring. The ultrasonic sensor's distance measurements were consistent within a 3% error margin for distances up to 1 meter. This accuracy was sufficient for the intended object detection application.

C. Energy Efficiency

The system optimized energy usage by leveraging the TM4C123G's deep sleep mode. During inactivity, the microcontroller consumed minimal power, waking only when the analog comparator detected a threshold breach.

The use of efficient communication protocols like I2C and SPI further contributed to reduced power consumption. Additionally, an external power source was used for the speaker and Power LED to reduce the load on the microcontroller's GPIO pins. Overall, the energy-saving mechanisms enhanced the system's operational longevity, making it suitable for battery-powered applications.

D. Limitations and Improvements

1. Limitations

1.1. The LM35 sensor is susceptible to slight variations due to ambient conditions, which could impact threshold accuracy.

1.2. The ultrasonic sensor's performance is affected by reflective and absorbent surfaces, leading to occasional inaccuracies in distance measurement.

1.3. The system does not scan the same area every time. The position of the stepper motor and the distance sensor is manually adjusted after each operation for scanning the same area.

2. Improvements

2.1. Enhanced Sensor Calibration: Implementing advanced calibration methods for the LM35 sensor can improve accuracy under varying environmental conditions.

2.2. Advanced Algorithms: Utilizing more sophisticated filtering and object detection algorithms could enhance data reliability and system responsiveness.

VI. CONCLUSION

The Temperature-Initiated Object Detection project demonstrates a sophisticated integration of advanced sensing, control, and feedback mechanisms using the TM4C123G microcontroller. Through the seamless incorporation of sensors like the LM35, BMP280, and HC-SR04, the system effectively achieves real-time temperature monitoring and precise object detection. By leveraging efficient communication protocols such as I2C and SPI, alongside modular programming and interrupt-driven design, the project delivers robust, responsive, and reliable performance.

The system successfully achieves its objectives, including accurate detection of temperature thresholds, comprehensive environmental scanning, and dynamic user feedback through the Nokia 5110 LCD, RGB LEDs, and audio signals. This integration ensures smooth operation across various scenarios.

The design provides a framework for future enhancements, paving the way for improved accuracy, advanced object recognition, and broader application potential.

Overall, this project sets a strong precedent for developing intelligent, sensor-based systems in embedded applications, offering a blend of innovation, efficiency, and practical utility.

REFERENCES

[1] J. Valvano, "Starter files for embedded systems." https://users.ece.utexas.edu/%7Evalvano/arm/.

[2] TI, "Tiva™ tm4c123gh6pm microcontroller data sheet." http://www.ti.com/lit/ds/spms376e/spms376e.pdf.