Report of

Intelli-Bin-The Smart Dustbin

A report to be submitted by

Pranav Gupta (210953244)
Anish Singh (2109532958)
Garakipati Abhinav (210953262)
Yashika Goyal (210953268)
Batch No. CCE B2

Bachelor of Technology

in

Computer and Communication Engineering

Submitted To

Dr. Manoj Tolani
Assistant Professor

Dr. Pankaj Kumar
Assistant Professor

MANIPAL
ACADEMY *of* HIGHER EDUCATION
(Institution of Eminence Deemed to be University)
INSPIRED BY LIFE

Manipal Institute of Technology
Manipal Academy of Higher Education
Manipal, Karnataka-576104, India

April, 2024

Contents

1 Introduction

Our Smart Dustbin is a cutting-edge way to make throwing away trash hassle-free. It's not your typical trash can. Imagine this: it detects when you or your garbage are close by, and then it uses a little magic to open the lid smoothly without the need for buttons or foot pedals. All of this is made possible by intelligent sensors that communicate with a tiny internal computer to control a motor. However, it's about more than just a cool bin—it's about simplifying life and maintaining organization without the typical commotion. Imagine if these were found in offices and parks everywhere. That would be a step towards a more straightforward and clean method of waste management for all.

Objective: This project aims to conceptualize, construct, and showcase a Smart Dustbin prototype that employs sensor-based technology coupled with a motorized lid mechanism. The primary goal is to elevate user convenience, promote cleanliness, and streamline waste management processes through an automated lid system triggered by proximity detection. By demonstrating the feasibility and practicality of integrating advanced technology into everyday waste disposal, this endeavor aims to redefine conventional trash bins, setting a precedent for enhanced efficiency and user-centric innovation.

Scope: The project encompasses the design, development, and implementation of a Smart Dustbin utilizing an LPC1768 microcontroller programmed with embedded C on Keil uVision. The scope involves integrating sensor technology, specifically the HC-SR04 proximity sensor, with the microcontroller to enable automatic lid operation upon detecting user proximity or waste presence. The project includes the design and assembly of the physical components, software development, testing procedures, and the utilization of Flash Magic for microcontroller programming. Additionally, the scope involves evaluating the system's functionality, reliability, and feasibility in real-world applications, emphasizing user-friendliness, efficiency in waste management, and the potential for scalability or future enhancements.

Project Description: The Smart Dustbin project employs an LPC1768 microcontroller programmed with embedded C on Keil uVision, utilizing Flash Magic for programming. Equipped with an HC-SR04 ultrasonic sensor, this innovation transforms a regular trash bin into a responsive system.

Upon detecting user proximity or waste, the sensor signals the microcontroller, prompting the stepper motor to rotate clockwise, automatically opening the lid. Subsequently, in the absence of a signal, the lid closes by an anticlockwise motor rotation, ensuring containment.

This integration optimizes hygiene and efficiency by autonomously closing the lid when no presence is detected or after a preset duration, culminating in a user-friendly and hygienic waste disposal solution.

2 Components Required

- **LPC1768:** The NXP (founded by Philips) LPC1768 is an ARM 32-bit Cortex-M3 Microcontroller. The code will be written in Embedded C and will be tested on an ALS evaluation board. A 16x2 LCD model has been used to generate the output produced by the sensor along with the 5V DC supply.

- **HC-SR04 sensor:** The HC-SR04 ultrasonic sensor is a popular choice for distance measurement applications. It operates by emitting ultrasonic pulses and measuring the time taken for the pulses to reflect back from nearby objects, allowing for accurate distance measurement.

Figure 1: HC-SR04 Sensor

- **Jumper cable wires:** Jumper cable wires are used to establish connections between various components in the circuit. They provide flexibility and ease of connection, enabling quick prototyping and testing.

- **Stepper Motor:** A Stepper Motor moves in precise steps through controlled electrical pulses. This allows for accurate rotation, essential for tasks like lid control in our project. To control the stepper motor's movement, connect it to the LPC1768 microcontroller via digital pins. To enable precise rotation, the motor's coils are typically controlled by two pairs of pins. The stepper motor rotates step-by-step by energising coils in a particular order. The direction and angle of the motor's rotation are set by this regulated activation pattern.

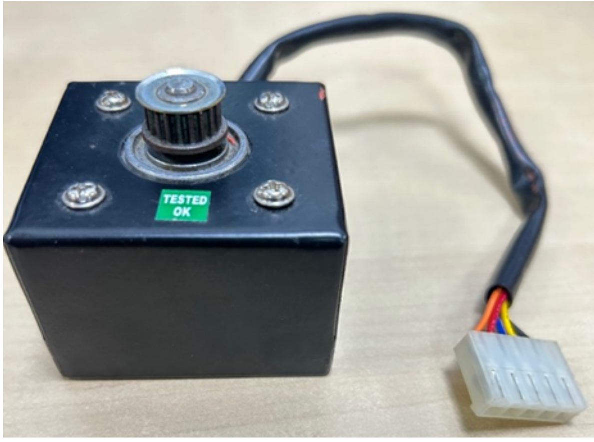

Figure 2: Stepper Motor used

- **Connector:** Connectors are used to establish electrical connections between different components or devices. They come in various types and configurations, such as pin headers, female headers, and terminal blocks, providing versatility in circuit design and assembly.
- **Power Supply:** A stable power supply is essential for the proper functioning of electronic circuits. In this project, a 5V DC power supply is used to power the components, ensuring consistent operation and accurate measurement.

3 Working Principle

The working principle of the smart car parking system involves the integration of various components to achieve accurate distance measurement, parking duration tracking, and dynamic fee calculation. Here's a detailed explanation of the working principle:

1. **Initialization**: The system is initialized when an object is detected near the dustbin. The LPC1768 microcontroller, serving as the brain of the system, is powered up along with the HC-SR04 ultrasonic sensor.
2. **Distance Measurement**: The HC-SR04 sensor emits ultrasonic pulses towards the parking space. These pulses travel through the air and bounce back when they encounter an obstacle. The sensor calculates the time taken for the pulses to return, which is directly proportional to the distance between the sensor.

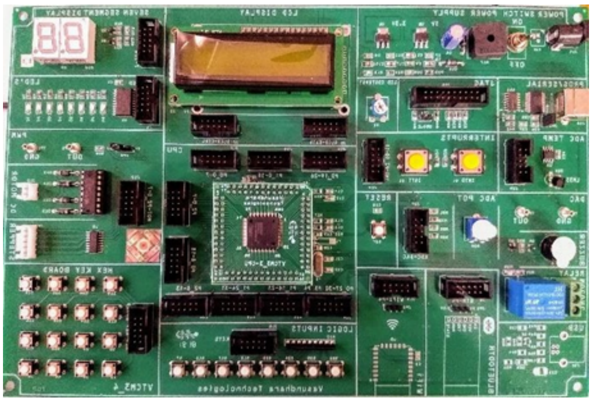

Figure 3: Lpc1768 kit

3. **Data Processing**: The LPC1768 microcontroller receives the distance measurements from the HC-SR04 sensor and processes this data in real-time. Using embedded C programming, the microcontroller analyzes the distance information to detect object(trash) entry and exit events. Additionally, it tracks the duration of session by continuously monitoring changes in distance.

4. **Stepper Motor Working**: A Stepper Motor moves in precise steps through controlled electrical pulses. This allows for accurate rotation, essential for tasks like lid control in our project. To control the stepper motor's movement, connect it to the LPC1768 microcontroller via digital pins. To enable precise rotation, the motor's coils are typically controlled by two pairs of pins. Through these pins, the microcontroller communicates with the stepper motor, controlling the order and direction of coil activations necessary for rotation. Use a microcontroller control algorithm to regulate the stepper motor's motion in response to particular inputs or circumstances, like the ultrasonic sensor's distance readings. Ensure synchronization with other components by integrating the control of the stepper motor into the microcontroller's overall operation. This could involve using external devices, such as GPIO pins or communication interfaces like Flash Magic, to enable precise lid opening/closing based on distance calculations from the sensor and coordinated operation.

5. **Feedback and Adjustment:** If necessary, put feedback mechanisms in place to guarantee precise motor control and lid movement in response to changes in the surrounding environment or user interactions.

6. **System Maintenance**: The system undergoes regular maintenance and calibration to ensure accurate distance measurements and reliable operation. This includes periodic checks of sensor functionality, microcontroller performance, and user interface responsiveness.

4 Flow Diagram

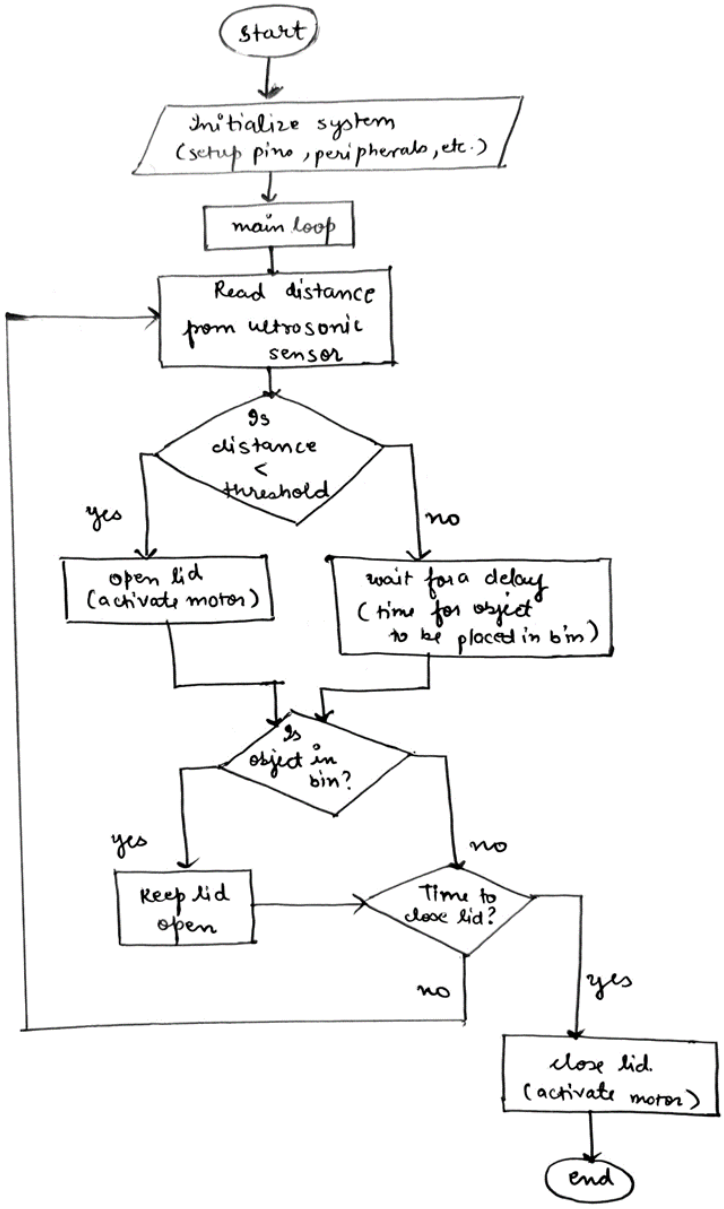

Figure 4: Flow Diagram

5 Block Diagram

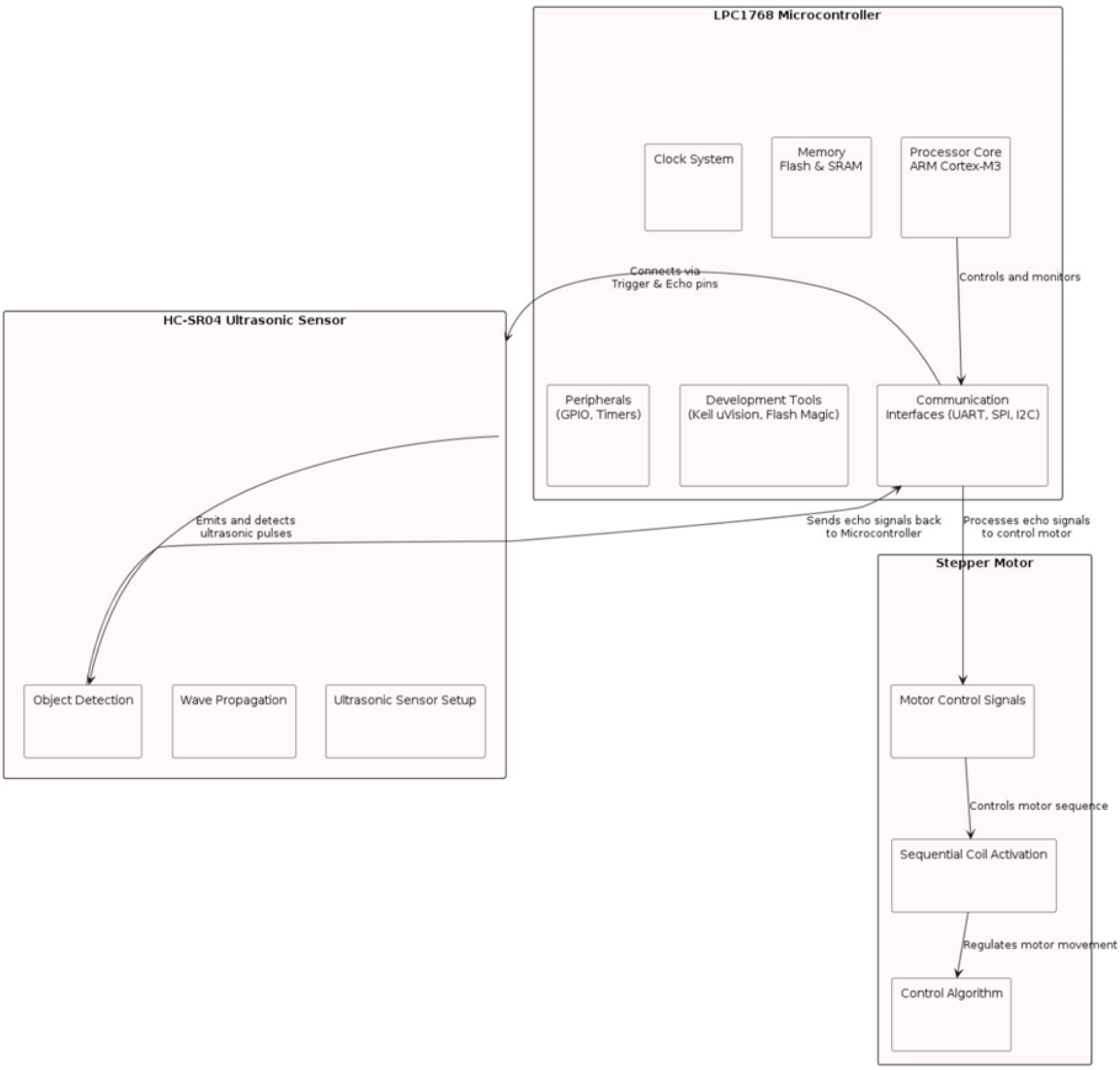

Figure 5: Block diagram used for the project.

6 Circuit Diagram

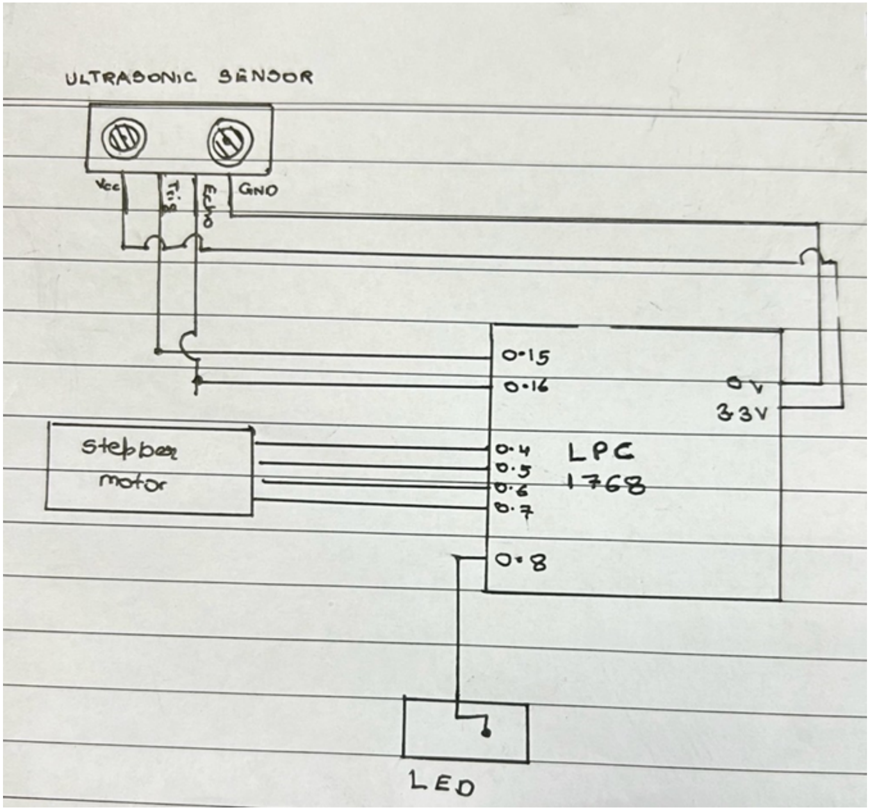

Figure 6: Circuit diagram of the Smart Dustbin.

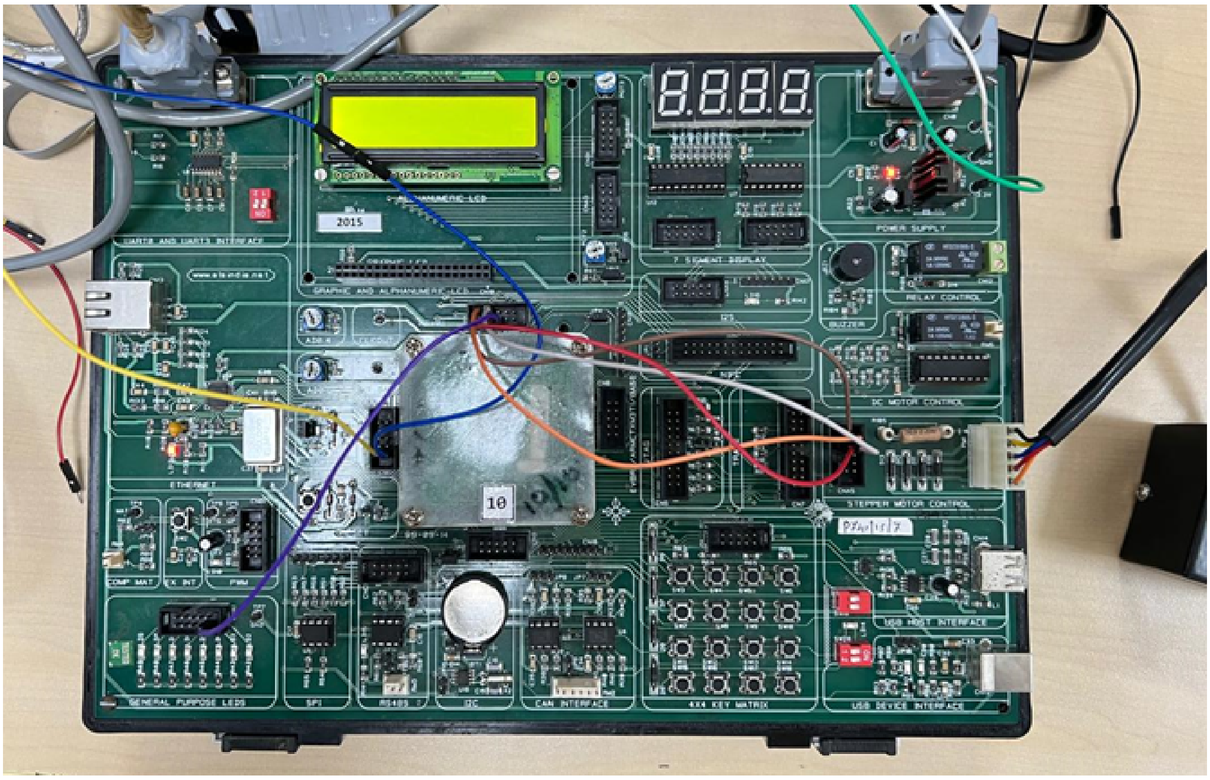

Figure 7: Demonstration on LPC

7 Outcome

Our Smart Dustbin project combines everyday convenience with technological know-how. Together, the stepper motor, ultrasonic sensor, and LPC1768 microcontroller enable the bin to open and close on its own accord in response to waste or people in the vicinity. It's about streamlining our daily tasks and maintaining organisation, not just about technology. This is not a one-time pilot project. It's a peek of what lies ahead: waste disposal handled intelligently by our bins will make life easier and cleaner. This project demonstrates how technology can genuinely improve our lives by providing more ingenious solutions for our everyday tasks.

Future Scope

1. **Intelligent Sensors :** Investigate enhanced sensors to enhance the bin's ability to sense when to open and close, guaranteeing that it is always prepared when required.

2. **Connectivity Monitoring:** To facilitate the monitoring of waste levels and maintenance alerts, think about connecting the bin to cellphones or a central system.

3. **Waste Sorting Techniques:**Provide technology to assist the bin in sorting various waste kinds, making it easier for users to recycle.

4. **Sustainability Features:** To lessen your impact on the environment and to promote sustainability, include eco-friendly components like solar power.

5. **Interfaces that are User-Friendly:** For everyone's convenience, use interfaces that are easy to use, like touch sensors or voice commands.

6. **Scaling for public use**: Examine the possibility of using these smart bins for widespread waste management advantages in larger environments, such as office buildings or cities.

7. **Data Utilisation for Efficiency** : Make better decisions, manage waste, and plan routes by utilising the data that has been gathered.

8. **Collaboration for impact**: Work with waste management agencies to incorporate these smart bins into the current system for a citywide waste collection that is better organised.

Waste management could become easier, more effective, and environmentally friendly with each of these approaches, which would help both private citizens and larger communities.

8 Appendix

Embedded C Program Code:

```
#include <LPC17xx.h>
#include <system_LPC17xx.h>

#define PRESCALE 29999999
#define TRIG (1 << 15)       // P0.15
#define ECHO (1 << 16)       // P0.16
#define LED_PIN (1 << 8)     // P0.8
#define STEPPER_CTRL_MASK 0xF << 4  // P0.4 to P0.7 for stepper
    motor controls

void timer21(int mr){
LPC_TIM0->TCR=0X2;
LPC_TIM0->CTCR=0;
LPC_TIM0->MCR=0X4;
LPC_TIM0->EMR=0X20;
LPC_TIM0->MR0=mr;
LPC_TIM0->PR=7999;
LPC_TIM0->TCR=0X1;

while(!(LPC_TIM0->EMR&0X1));
}

unsigned long int var1;
unsigned long int i, j, k;
int row, x, col, key;
int echoTime = 5000;
float distance = 0;

enum State {
    IDLE,
    ROTATE_CLOCKWISE,
    WAIT,
    ROTATE_ANTICLOCKWISE
```

```
};

enum State currentState = IDLE;
int stepCounter = 0;
int waitCounter = 0;

void initTimer0(void);
void startTimer0(void);
float stopTimer0(void);
void delayUS(unsigned int microseconds);
void delay(unsigned int r1);
void controlStepperMotor(int enable);
void rotateOnceClockwise(void);
void rotateOnceAnticlockwise(void);


int main() {
    SystemInit();
    SystemCoreClockUpdate();
    initTimer0();

    // Configuring trigger, echo, light, fan, buzzer
    LPC_PINCON->PINSEL0 &= 0x3fffc0ff;
    LPC_PINCON->PINSEL1 &= 0xfffffffc;

    // Configuring keyboard
    LPC_PINCON->PINSEL3 &= 0;
    LPC_PINCON->PINSEL4 &= 0;
    LPC_GPIO1->FIODIR |= 0 << 16 | 0 << 23; // Direction for
       ECHO PIN and keyboard
    LPC_GPIO0->FIODIR |= TRIG;
    LPC_GPIO2->FIODIR |= 0xf << 10;
    LPC_GPIO0->FIODIR |= 0x00000070; // Direction for Light, fan
       , and buzzer
    LPC_GPIO0->FIODIR |= LED_PIN;     // Direction for the LED
    LPC_GPIO0->FIODIR |= STEPPER_CTRL_MASK;  // Direction for
       stepper motor controls
```

```
LPC_GPIO0->FIOCLR = TRIG; // Ensure TRIG is initially low

while (1) {
    LPC_GPIO0->FIOSET = TRIG; // Output 10us HIGH on TRIG
        pin
    delayUS(10);
    LPC_GPIO0->FIOCLR = TRIG;

    while (!(LPC_GPIO0->FIOPIN & ECHO)) {
        // Wait for a HIGH on ECHO pin
    }
    startTimer0();
    while (LPC_GPIO0->FIOPIN & ECHO)
        ; // Wait for a LOW on ECHO pin
    echoTime = stopTimer0();        // Stop Counting
    distance = (0.0343 * echoTime) / 40;

   if (distance < 40) {
        if (currentState == IDLE) {
            LPC_GPIO0->FIOSET = LED_PIN;   // Turn on the LED
            currentState = ROTATE_CLOCKWISE;
            stepCounter = 0;
        }
    } else {
        if (currentState == ROTATE_CLOCKWISE) {
            // Finished opening, move to WAIT state
            currentState = WAIT;
            waitCounter = 0;
        }
    }

    // State machine logic
         switch (currentState) {
        case ROTATE_CLOCKWISE:
            if (stepCounter < 50) {
                controlStepperMotor(1); // Clockwise
```

```
                    rotation
                stepCounter++;
            } else {
                currentState = WAIT;
                stepCounter = 0;  // Reset the step counter
            }
            break;
        case WAIT:
            delay(7000000); // 1 second delay
            waitCounter++;
            if (waitCounter >= 4) { // 4 seconds elapsed
                currentState = ROTATE_ANTICLOCKWISE;
                stepCounter = 0;  // Reset the step counter
            }
            break;
        case ROTATE_ANTICLOCKWISE:
            if (stepCounter < 50) {
                controlStepperMotor(0); // Anticlockwise
                    rotation
                stepCounter++;
            } else {
                controlStepperMotor(-1); // Stop the motor
                currentState = IDLE;
                LPC_GPIO0->FIOCLR = LED_PIN;  // Turn off
                    the LED
                stepCounter = 0;  // Reset the step counter
            }
            break;
        default:
            break;
    }

    delay(88000); // Delay between sensor checks
  }
}
```

```c
void initTimer0(void) {
    // Timer for distance
    LPC_TIM0->CTCR = 0x0;
    LPC_TIM0->PR = 11999999;
    LPC_TIM0->TCR = 0x02; //
    LPC_TIM0->TCR = 0x02; // Reset Timer
}

void startTimer0(void) {
    LPC_TIM0->TCR = 0x02; // Reset Timer
    LPC_TIM0->TCR = 0x01; // Enable timer
}

float stopTimer0(void) {
    LPC_TIM0->TCR = 0x0;
    return LPC_TIM0->TC;
}

void delayUS(unsigned int microseconds) {
    LPC_SC->PCLKSEL0 &= ~(0x3 << 2); // Set PCLK_TIMER0 to
        divide by 1
    LPC_TIM0->TCR = 0x02;               // Reset timer
    LPC_TIM0->PR = 0;                   // Set prescaler to 0
    LPC_TIM0->MR0 = microseconds - 1; // Set match register for
        the specified microseconds
    LPC_TIM0->MCR = 0x01;               // Interrupt on match
    LPC_TIM0->TCR = 0x01;               // Enable timer

    while ((LPC_TIM0->IR & 0x01) == 0)
        ; // Wait for interrupt flag
    LPC_TIM0->TCR = 0x00; // Disable timer
    LPC_TIM0->IR = 0x01;
}

void delay(unsigned int r1) {
    volatile int r;
```

```
    for (r = 0; r < r1; r++)
        ;
}

void controlStepperMotor(int action) {
    if (action == 1) {
        // Clockwise rotation
        LPC_GPIO0->FIOPIN |= STEPPER_CTRL_MASK;
        for (j = 0; j < 150; j++) {
            rotateOnceClockwise();
        }
    } else if (action == 0) {
        // Anticlockwise rotation
        LPC_GPIO0->FIOPIN |= STEPPER_CTRL_MASK;
        for (j = 0; j < 3; j++) {
            rotateOnceAnticlockwise();
        }
    } else {
        // Stop the motor
        LPC_GPIO0->FIOPIN &= ~STEPPER_CTRL_MASK;
    }
}

void rotateOnceClockwise() {
    var1 = 0x8;
    for (i = 0; i < 4; i++) {
        var1 = var1 << 1;
        LPC_GPIO0->FIOPIN = ~var1;
        for (k = 0; k < 25000; k++)
            ;
    }
}

void rotateOnceAnticlockwise() {
    var1 = 0x100;
    for (i = 0; i < 4; i++) {
        var1 = var1 >> 1;
```

```
        LPC_GPIO0->FIOPIN = ~var1;
        for (k = 0; k < 25000; k++)
            ;
    }
}
```

9 References

Rokhsana Titlee and Muhibul Haque Bhuyan, "Design, Implementation and Testing of Ultrasonic High Precision Contactless Distance Measurement System Using Microcontroller", SEU Journal of Science and Engineering, vol. 10, no. 2, 2016

Li Zhengdong, Huang Shuai, Lin Zhaoyang, Luo Weifang and He Daxi, "The Ultrasonic Distance Alarm System Based on MSP430F449", Fifth Conference on Measuring Technology and Mechatronics Automation, 2013