SMART DUSTBIN

Project Report Submitted

to

MANIPAL ACADEMY OF HIGHER EDUCATION

For Partial Fulfillment of the Requirement for the

Award of the Degree

Of

Bachelor of Technology

in

Information Technology

by

Anishka

(210911328)

Utkarsh Singhal

(210911356)

Under the guidance of

Mr. Raviraj Holla
Assistant Professor – Senior Scale
Department of I&CT
Manipal Institute of Technology
Manipal, Karnataka, India

Dr. Sameena Begum Pathan
Assistant Professor
Department of I&CT
Manipal Institute of Technology
Manipal, Karnataka, India

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Unit of MAHE, Manipal

November 2023

Index

1. Abstract

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 organisation 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.

2. Introduction

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.

Hardware Requirements:

LPC1768 Development board
Power Supply
Ultrasonic Sensor (HC-SR04)
Stepper Motor (for opening the lid)
Jumper Cable

Software Requirements:

Language: Embedded C
Software: Keil uVision, Flash Magic

3. Working Principle:

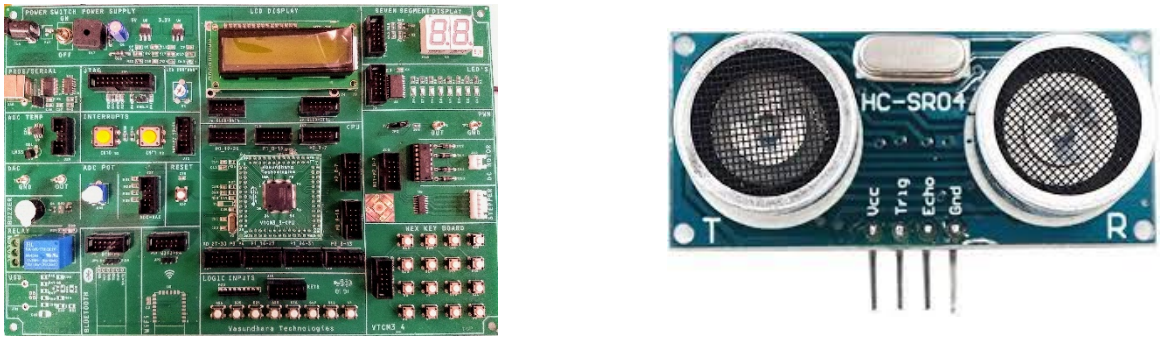


Fig 1. Lpc1768 kit and Ultrasonic Sensor

Lpc1768 Microcontroller:

The LPC1768 is a microcontroller based on the ARM Cortex-M3 architecture, and it is manufactured by NXP Semiconductors. It is part of the LPC1700 series of microcontrollers, which are designed for embedded applications.

Processor Core:
Utilizes an ARM Cortex-M3 processor core for high performance and low power consumption, handling the execution of instructions.

Peripherals:
Integrates on-chip peripherals (e.g., UART, SPI, I2C, GPIO, timers) for communication with external devices and general-purpose input/output operations.

Memory:
Utilizes Flash memory for program storage and SRAM for data storage during program execution.

Clock System:
Features a sophisticated clock system for precise control of the processor clock frequency, crucial for meeting performance requirements and minimizing power consumption.

Communication Interfaces and Development Tools:
Supports various communication interfaces (e.g., UART, SPI, I2C) for serial communication and is programmed and debugged using development tools like Keil uVision and Flash Magic.

HC-SR04 Ultrasonic Sensor:

The ultrasonic sensor contains a transducer that converts electrical energy into ultrasonic waves (sound waves with a frequency higher than the human audible range, typically above 20 kHz). These waves are often referred to as ultrasonic pulses.

Wave Propagation:

The sensor emits ultrasonic pulses in a specific direction, usually by using a piezoelectric crystal that vibrates at the desired frequency. The ultrasonic waves travel through the air and reflect off objects in their path.

Object Detection:

When the ultrasonic waves encounter an object in their path, they reflect back towards the sensor. The time taken for the waves to travel to the object and back is measured.

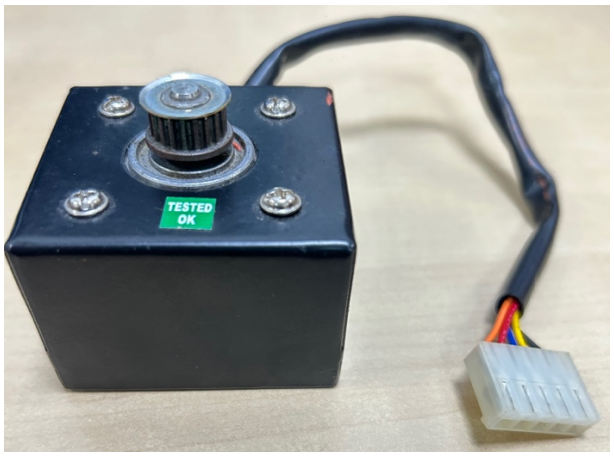
- **Ultrasonic Sensor Setup:**
  Connect the ultrasonic sensor to the LPC1768 microcontroller, typically using digital pins for trigger and echo signals. The trigger pin initiates the ultrasonic pulse, and the echo pin receives the reflected signal.
- **Pulse Generation**:
  The LPC1768 microcontroller sends a short pulse (trigger signal) to the ultrasonic sensor through the trigger pin. This pulse triggers the ultrasonic sensor to emit a burst of ultrasonic waves.
- **Echo Reception:**
  The ultrasonic waves travel through the air and, upon encountering an object, reflect back towards the sensor. The ultrasonic sensor detects the reflected waves and generates an echo signal on the echo pin.
- **Time Measurement:**
  The LPC1768 microcontroller measures the time it takes for the ultrasonic pulse to travel to the object and back. This time measurement is typically done by recording the duration between sending the trigger pulse and receiving the echo pulse.
- **Distance Calculation:**
  Using the known speed of sound in air, the microcontroller calculates the distance to the object using the formula:
  Distance= 2 x Speed of Sound Round/trip Time
  where the speed of sound is approximately 343 meters per second in air at room temperature.
- **Control Logic:**
  Implement control logic in the microcontroller based on the calculated distance. For example, if the calculated distance is below a certain threshold, it can trigger actions like rotating the Stepper motor which is responsible for opening and closing the lid.
- **Integration with Other Components:**
  Depending on the project requirements, integrate other components of the LPC1768 microcontroller, such as GPIO pins, communication interfaces (e.g., Flash Magic,), or external devices, to enhance the functionality or provide additional control features.

In brief, the LPC1768 microcontroller, paired with an ultrasonic sensor, precisely detects objects by measuring sound reflection to control the lid mechanism. This integration showcases a streamlined solution for efficient waste management, with the potential for further enhancements through additional component integration.

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.

- **Connection to LPC1768 Microcontroller**:
  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.
- **Motor Control Signals:**
  Through these pins, the microcontroller communicates with the stepper motor, controlling the order and direction of coil activations necessary for rotation.
- **Sequential Coil Activation:**
  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.
- **Control Algorithm:**
  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.
- **Interface and Integration:**
  Ensure synchronisation 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.
- **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.



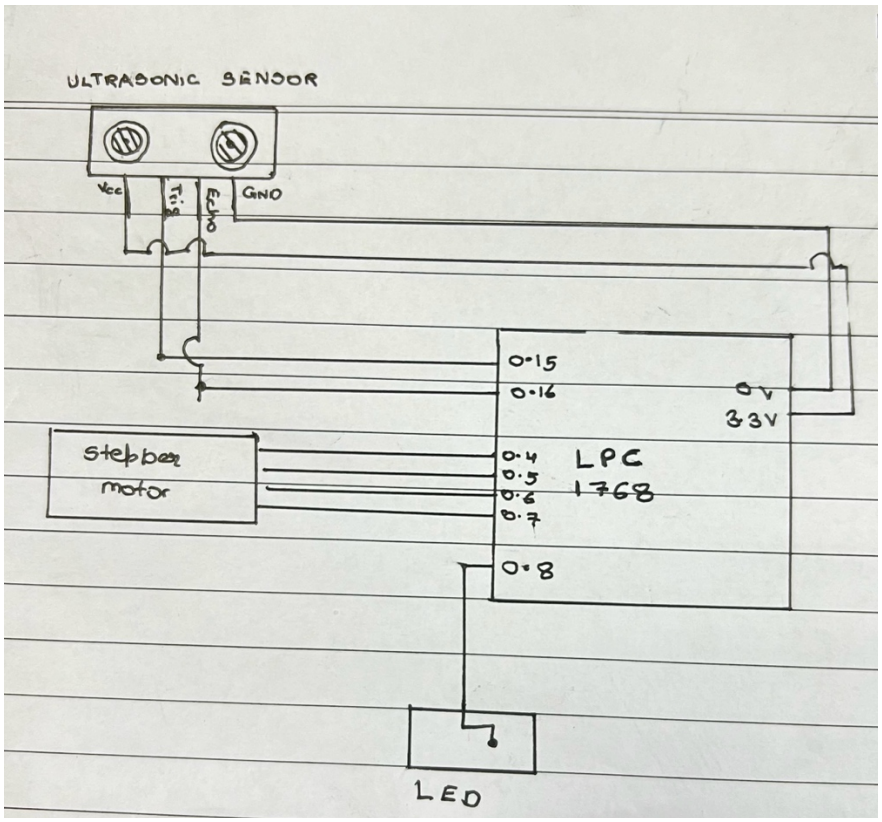


Fig 2. Circuit Diagram used for the project

4. Embedded C Working 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

	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
        }
}

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 MR 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++);
        }
}
```

5. Demonstration Of the Smart Dustbin using Lpc1768:

Fig 3. Working Project

6. Conclusion:

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.

7. Future Scope:

- **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.
- **Connectivity & Monitoring:** To facilitate the monitoring of waste levels and maintenance alerts, think about connecting the bin to cellphones or a central system.
- **Waste Sorting Technology:** Provide technology to assist the bin in sorting various waste kinds, making it easier for users to recycle.
- **Sustainability Features:** To lessen your impact on the environment and to promote sustainability, include eco-friendly components like solar power.
- **Interfaces that are User-Friendly:** For everyone's convenience, use interfaces that are easy to use, like touch sensors or voice commands.
- **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.
- **Data Utilisation for Efficiency:** Make better decisions, manage waste, and plan routes by utilising the data that has been gathered.
- **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. References:

[1] 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

[2] 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

9. Submitted by:

Name	Registration number	Roll Number	Semester&Branch	Section
Anishka	210911328	61	V (IT)	C
Utkarsh Singhal	210911356	63	V (IT)	C