M.Kumarasamy College of Engineering
NAAC Accredited Autonomous Institution
Approved by AICTE & Affiliated to Anna University
ISO 9001:2015 Certified Institution
Thalavapalayam, Karur - 639 113, TAMILNADU.

A Project Report

on

MEDI TRAY

ECB1223-MICROCONTROLLERS AND INTERFACING

Submitted by

BHARATHI S **(927623BEC018)**

BRINDHA SRI I **(927623BEC019)**

DEEPIKAA V **(927623BEC024)**

DHAARANI T **(927623BEC026)**

BACHELOR OF ENGINEERING

in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

M.KUMARASAMY COLLEGE OF ENGINEERING

(Autonomous)

KARUR – 639 113

MAY 2025

M.KUMARASAMY COLLEGE OF ENGINEERING, KARUR

BONAFIDE CERTIFICATE

Certified that this report **"MEDI-TRAY"** is the Bonafide work of **BHARATHI.S (927623BEC018), BRINDHA SRI.I (927623BEC019), DEEPIKAA.V (927623BEC024), DHAARANI.T (927623BEC026)** who carried out the project work under my supervision in the academic year 2024 - 2025 **EVEN**.

SIGNATURE

Dr.A.KAVITHA B.E., M.E., Ph.D.,

HEAD OF THE DEPARTMENT,

Professor,

Department of Electronics and Communication Engineering,

M.Kumarasamy College of Engineering,

Thalavapalayam,

Karur-639113.

SIGNATURE

Mrs.M.SENTAMILSELVI, M.E., (Ph.D).,

SUPERVISOR,

Assistant Professor,

Department of Electronics and Communication Engineering,

M.Kumarasamy College of Engineering,

Thalavapalayam,

Karur-639113.

This report has been submitted for the **ECB1223 - MICROCONTROLLERS AND INTERFACING** final review held at M.Kumarasamy College of Engineering, Karur on _____

PROJECT COORDINATOR

INSTITUTION VISION AND MISSION

Vision

To emerge as a leader among the top institutions in the field of technical education.

Mission

M1: Produce smart technocrats with empirical knowledge who can surmount the global challenges.

M2: Create a diverse, fully -engaged, learner -centric campus environment to provide quality education to the students.

M3: Maintain mutually beneficial partnerships with our alumni, industry and professional associations

DEPARTMENT VISION, MISSION, PEO, PO AND PSO

Vision

To empower the Electronics and Communication Engineering students with emerging technologies, professionalism, innovative research and social responsibility.

Mission

M1: Attain the academic excellence through innovative teaching learning process, research areas & laboratories and Consultancy projects.

M2: Inculcate the students in problem solving and lifelong learning ability.

M3: Provide entrepreneurial skills and leadership qualities.

M4: Render the technical knowledge and skills of faculty members.

Program Educational Objectives

PEO1: **Core Competence:** Graduates will have a successful career in academia or industry associated with Electronics and Communication Engineering

PEO2: **Professionalism:** Graduates will provide feasible solutions for the challenging problems through comprehensive research and innovation in the allied areas of Electronics and Communication Engineering.

PEO3: **Lifelong Learning:** Graduates will contribute to the social needs through lifelong learning, practicing professional ethics and leadership quality

Program Outcomes

PO 1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO 2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO 3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO 4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO 5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO 6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO 7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO 8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO 9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO 10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO 11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO 12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes

PSO1: Applying knowledge in various areas, like Electronics, Communications, Signal processing, VLSI, Embedded systems etc., in the design and implementation of Engineering application.

PSO2: Able to solve complex problems in Electronics and Communication Engineering with analytical and managerial skills either independently or in team using latest hardware and software tools to fulfil the industrial expectations.

Abstract	Matching with POs, PSOs
Temperature-tuned fans improve cooling system efficiency by adjusting fan speed based on ambient temperature. This method reduces energy consumption and enhances system performance. It is especially effective in HVAC and data center applications. The approach supports sustainability by lowering operational costs and carbon emissions	PO-1, PO-2, PO-4, PO-5, PO-7, PO-10 PSO-1,PSO-2

ACKNOWLEDGEMENT

We gratefully remember our beloved **Founder Chairman, (Late) Thiru. M. Kumarasamy**, whose vision and legacy laid the foundation for our education and inspired us to successfully complete this project.

We extend our sincere thanks to **Dr. K. Ramakrishnan, Chairman**, and **Mr. K. R. Charun Kumar, Joint Secretary**, for providing excellent infrastructure and continuous support throughout our academic journey.

We are privileged to extend our heartfelt thanks to our respected Principal, **Dr. B. S. Murugan, B.Tech., M.Tech., Ph.D.,** for providing us with a conducive environment and constant encouragement to pursue this project work.

We sincerely thank **Dr. A. Kavitha, B.E., M.E., Ph.D.,** Professor and **Head, *Department of Electronics and Communication Engineering***, for her continuous support, valuable guidance, and motivation throughout the course of this project.

Our special thanks and deep sense of appreciation go to our **Project Supervisor, Mrs.M.Sentamilselvi, B.E., M.E., (Ph.D).,** Assistant Professor, ***Department of Electronics and Communication Engineering,*** for his exceptional guidance, continuous supervision, constructive suggestions, and unwavering support, all of which have been instrumental in the successful execution of this project.

We would also like to acknowledge **Dr.K,Karthikeyan, B.E., M.Tech., Ph.D., Associate Professor, our Class Advisor** for their constant encouragement and coordination that contributed to the smooth progress and completion of our project work.

We gratefully thank all the **faculty members of the Department of Electronics and Communication Engineering** for their timely assistance, valuable insights, and constant support during various phases of the project.

Finally, we extend our profound gratitude to our **parents and friends** for their encouragement, moral support, and motivation, without which the successful completion of this project would not have been possible.

ABSTRACT

The Medi Tray is an intelligent, microcontroller-based system designed to automate and streamline medication administration in hospitals. It addresses the common issues of human error, missed doses, and improper timing associated with manual medication tracking by integrating RFID technology, a Real-Time Clock (RTC) module, and audio-visual alerts. Each medicine container is tagged with an RFID tag, allowing for accurate identification when placed in the tray. The RTC logs the exact time a dose is given or missed, while LEDs and buzzers alert healthcare staff to upcoming or missed dosages. A microcontroller like Arduino or ESP32 manages the entire process, ensuring timely drug delivery and improving patient safety. Optional integration with cloud storage or a mobile/web app allows remote monitoring and medication log access. This system reduces the workload of nursing staff, enhances healthcare management, and ensures a reliable, efficient, and error-free medication dispensing process. By combining modern embedded technologies, Medi Tray presents a smart solution for safer and more effective hospital operations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- **RTC** – Real-Time Clock
- **LCD** – Liquid Crystal Display
- **I2C** – Inter-Integrated Circuit
- **UNO** – Universal Number One (Arduino UNO)
- **USB** – Universal Serial Bus
- **MCU** – Microcontroller Unit
- **PWM** – Pulse Width Modulation
- **Vcc** – Voltage at the Common Collector
- **PCB** – Printed Circuit Board
- **LED** – Light Emitting Diode

CHAPTER 1

INTRODUCTION

1.1 Objective

In hospitals and healthcare facilities, accurate and timely medication administration is critical to ensuring patient safety and effective treatment outcomes. However, traditional methods of tracking and dispensing medication rely heavily on manual processes, which are prone to human errors such as missed doses, incorrect timings, or wrong medications being administered. These errors can compromise patient health and increase the burden on medical staff.

To address these challenges, the Medi Tray project introduces an automated medication tracking and dispensing system. It integrates RFID technology for identifying medicines, a Real-Time Clock (RTC) for accurate timing, and visual/audio alerts to assist healthcare providers in ensuring timely and correct medication delivery. By using a microcontroller to coordinate the system's functions, Medi Tray offers a reliable, efficient, and intelligent solution that reduces manual workload and enhances the overall quality of healthcare services. The system can also be extended with cloud connectivity and mobile application support, enabling remote monitoring and real-time alerts for improved medication adherence.

1.2 Project Details

The Smart Medi Tray is an Arduino-based electronic pill reminder and monitoring system designed to assist patients, especially the elderly and chronically ill, in managing their medication schedule efficiently. The system uses a combination of LEDs, a buzzer, an LCD display, and push buttons to indicate

and record the status of medicine intake. At preset times, the buzzer alerts the user, and a blue LED lights up to indicate that it is time to take the pill. The LCD displays a message to guide the user, such as “Take Pill Now.” When the user confirms the intake by pressing a button, the blue LED switches to green, showing that the medicine has been taken. If the button is not pressed within a set duration, the system marks the dose as missed and activates a red LED. This ensures that the user, or caregivers, can easily track whether doses have been taken, missed, or are pending. The Smart Medi Tray is a simple, low-cost solution that promotes timely medication and helps improve treatment outcomes by reducing human error and forgetfulness.

1.3 Description

The Medi Tray is a smart medication tracking and dispensing system developed to automate the process of medicine administration in hospitals. It combines key technologies such as RFID, Real-Time Clock (RTC), microcontrollers, and alert mechanisms to ensure accurate and timely delivery of medications. Each medicine container is equipped with an RFID tag, which allows the system to uniquely identify and verify the medication when placed in the tray.

A microcontroller, such as an Arduino or ESP32, acts as the brain of the system. It reads data from the RFID module, checks the current time using the RTC module, and triggers appropriate actions based on the medication schedule. Visual alerts using LEDs and audio alerts using buzzers are used to notify healthcare staff about upcoming or missed doses. An LCD display provides real-time feedback about the medication status and timing.

Additionally, the system can be extended with cloud-based storage or a mobile/web application, enabling remote monitoring, logging of medication history, and generating alerts or reports. This smart integration helps in reducing

workload for medical staff, minimizing human errors, and improving overall patient care and safety.

CHAPTER-2

LITERATURE SURVEY

2.1 RHINO: An Autonomous Robot for Mapping Underground Mine Environments

Authors: C. Tatsch, J. A. Bredu Jnr, D. Covell, I. B. Tulu, and Y. Gu (2023)

Development of 'Rhino', a skid-steer, four-wheeled unmanned ground vehicle (UGV) equipped with LiDAR and IMU sensors, utilizing the LIO-SAM framework for long-duration autonomous navigation and 3D mapping in underground mines. Challenges include sensor degradation, GPS-denied localization, and operational durability.

2.2 IoT-Based Smart Pill Dispenser with Remote Monitoring and Alerts

Authors: R. Kumar and S. Sharma (2020)

Proposed an IoT-based system for automated pill dispensing and remote monitoring. The system includes sensors, GSM modules, and alert mechanisms to ensure patients follow medication schedules.

2.3 Smart Medication Dispenser: A Review of Automated Pill Dispensers

Authors: M. Patel, R. Singh, and A. Desai (2019)

Survey of automated pill dispensers for elderly and chronically ill patients. Discusses technologies like RFID, GSM, and Wi-Fi, and evaluates their effectiveness in improving medication adherence.

2.4 Design and Implementation of a Smart Pill Dispenser for Elderly Patients

Authors: A. Gupta and S. Verma (2021)

Describes the development of a microcontroller-based smart pill dispenser with buzzer and LED alerts. It focuses on reducing medication errors and ensuring timely dosage for elderly patients.

2.5 Smart Medication Reminder System using Arduino and GSM

Authors: R. Nair and V. K. Pillai (2020)

Implemented a simple medication reminder system using Arduino and GSM modules. SMS alerts are sent to patients and caregivers based on the dosage schedule stored in the microcontroller.

2.6 IoT Based Pill Reminder and Dispenser for Elderly People

Authors: P. Joshi and M. Shah (2022)

Developed a cost-effective IoT-enabled pill dispenser that sends alerts via a mobile app. It uses a servo-controlled mechanism to dispense pills at scheduled times.

2.7 Real-Time Health Monitoring and Medication Reminder System

Authors: L. Zhang and Y. Zhao (2018)

Combined wearable health monitoring with an automated medication reminder system. Sensors collect patient vitals, and dosage reminders are generated accordingly.

2.8 Automation in Healthcare: Smart Medicine Dispenser with Mobile App

Authors: K. Thomas and N. David (2021)

Presents a Bluetooth-enabled smart medicine dispenser integrated with a mobile app for schedule management, alerts, and manual override in emergencies.

2.9 Smart Pill Box with Cloud Connectivity for Remote Monitoring

Authors: S. Banerjee and A. Mukherjee (2019)

Created a pill box that logs data to the cloud and allows doctors and family members to monitor medication adherence remotely. Includes an LCD and alert system.

2.10 Medication Management System Using RFID and NFC

Authors: H. Lin and M. Chen (2020)

Explores RFID and NFC technologies for tracking medication intake. The system is designed to work in hospitals for accurate patient-medication matching and real-time alerts.

CHAPTER - 3

EXISTING SYSTEM

3.1. Manual Medication Tracking

In many hospitals, nurses or caregivers manually track and administer medications. This involves maintaining written records or relying on memory, which increases the risk of errors such as missed doses or incorrect timings.

3.2. Lack of Real-Time Monitoring

Existing manual systems do not provide real-time alerts or notifications. As a result, healthcare providers may overlook scheduled dosages, especially in busy hospital environments.

3.3. High Risk of Human Error

Due to fatigue or workload, staff may accidentally administer the wrong medication or at the wrong time. This compromises patient safety and may delay recovery.

3.4. Limited Record-Keeping and Accessibility

Traditional systems do not maintain digital records or logs, making it difficult to track medication history, generate reports, or allow remote monitoring by doctors or caregivers.

CHAPTER - 4

PROPOSED SYSTEM

4.1 Proposed System Description

The proposed system architecture for the **Smart Medi Tray** is designed to offer a structured and efficient approach to medication reminders over a three-day cycle with caregiver integration. At the core of the system is an **Arduino UNO** microcontroller, which serves as the central control unit. The Arduino is programmed to keep track of time and manage alerts across three consecutive days, with each day having its own dedicated **buzzer**—Buzzer 1 for Day 1, Buzzer 2 for Day 2, and Buzzer 3 for Day 3. These buzzers are triggered at scheduled intervals to alert the user when it is time to take a medication.

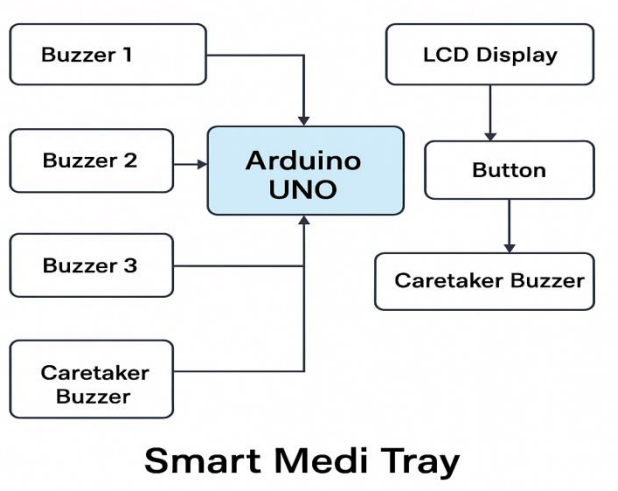

Figure.4.1 Proposed System block diagram

An **LCD display** is used to visually guide the user by showing real-time messages such as “Take Day 1 Pill” or “Press Button After Pill.” Once the user takes the medicine, they press a **confirmation button**, which stops the buzzer and marks the dose as completed. If the user does not press the button, the buzzer continues to sound, indicating a missed dose.

In addition to user interaction, the system includes a **fourth buzzer** meant for the **caretaker**. This buzzer can be activated by the caregiver using a separate push button, allowing them to confirm supervision or respond to missed doses. This ensures caregiver involvement and adds a layer of accountability.

The entire system is mounted on a board for hardware integration and powered via USB or external power supply. The architecture supports modular expansion and can be enhanced in the future with real-time clocks, wireless modules (like Wi-Fi or GSM), or mobile app integration.

4.2 Components and Their Functions

Table.4.2 Components Details

Component Name	Quantity	Range / Specification	Price (Approx.)
Arduino UNO	1	ATmega328P Microcontroller	₹500
16x2 LCD Display (I2C)	1	HD44780 Driver, Green Backlit	₹120
Piezo Buzzer	4	5V Operating Voltage	₹20 x 4 = ₹80
Push Button	4	Momentary Type, 5V	₹5 x 4 = ₹20
RTC Module (DS3231)	1	I2C Interface, CR2032 Battery Backup	₹80
Resistors	4	10kΩ for Pull-down	₹2 x 4 = ₹8
Perforated Board	1	For mounting components	₹40
Jumper Wires	20+	Male-to-Male/Male-to-Female	₹50
USB Cable / Power Supply	1	USB Type-B or 9V Adapter	₹50

1. Arduino UNO

Role: Main microcontroller unit

Details: Based on the ATmega328P, the Arduino UNO handles all processing tasks such as controlling the LCD, checking button inputs, and triggering buzzers. It acts as the brain of the system.

2. 16x2 LCD Display (I2C Interface)

Role: User interface

Details: Displays essential information like pill schedules and confirmation messages. The I2C module reduces pin usage and simplifies wiring.

3. Piezo Buzzers (4 Units)

Role: Audio alert system

Details:

Buzzer 1 – Alerts for Day 1 medication

Buzzer 2 – Alerts for Day 2 medication

Buzzer 3 – Alerts for Day 3 medication

Buzzer 4 – Alerts the caretaker in case of missed doses or for acknowledgment

4. Push Buttons (4 Units)

Role: User and caretaker interaction

Details:

Buttons 1–3 – Pressed by the user to confirm pill intake for Days 1 to 3

Button 4 – Pressed by the caretaker to acknowledge an alert

5. Real-Time Clock (RTC) Module – DS3231

Role: Timekeeping

Details: Maintains accurate time to ensure timely pill reminders. Includes a battery backup to function even when power is off.

6. Resistors

Role: Electrical stability

Details: Used as pull-down resistors (typically 10kΩ) to prevent false triggering of the push buttons.

7. Perforated Board

Role: Hardware assembly

Details: Provides a durable base to mount and solder all components for a neat and permanent setup.

8. Jumper Wires

Role: Wiring and connections

Details: Connect various components to the Arduino and among themselves to allow signal and power flow.

9. Power Supply (USB or 9V Adapter)

Role: Power source

Details: Provides stable 5V operating voltage to the Arduino and other components.

4.3 System Architecture

The **Smart Medi Tray** system architecture is designed to offer a reliable and user-friendly medication reminder and monitoring solution. At the core of the architecture is the **Arduino UNO**, which acts as the central processing unit. It receives real-time data from the **RTC (Real-Time Clock) module**, which ensures precise timing for daily medication reminders.

The system includes **three piezo buzzers**, each corresponding to a specific day (Day 1, Day 2, and Day 3). These buzzers are triggered at pre-set times based on RTC inputs, alerting the user to take their medication. Once the user takes the pill, they press the corresponding **push button** to confirm intake, which is registered by the Arduino and logged in the system.

A **fourth buzzer** is designated for the **caretaker**. If any medication is missed or needs acknowledgment, the system activates this buzzer, alerting the caretaker. The caretaker confirms the alert by pressing their dedicated **push button**.

The system uses a **16x2 LCD Display** with I2C support to provide users with real-time feedback and instructions, such as which day's medicine is due and confirmation messages after button presses.

All components are interconnected using **jumper wires** and mounted on a **perforated board** for stability and compactness. The entire setup is powered through a **USB cable** or a **9V adapter**, regulated by the onboard Arduino circuitry.

This architecture ensures timely alerts, user confirmation, and caretaker monitoring—making it ideal for elderly or chronically ill patients who require regular medication adherence.

4.4 Key Features of the System:

The Smart Medi Tray is a cost-effective and user-friendly medication reminder system that incorporates several essential features to ensure timely pill intake and caretaker involvement. It includes three individual buzzers that activate on specific days (Day 1, Day 2, and Day 3) to audibly remind the user to take their medication. A fourth buzzer is reserved for the caretaker, which gets triggered if a dose is missed or requires acknowledgment. The system enables manual confirmation through dedicated push buttons—three for the user to confirm daily medication and one for the caretaker to acknowledge alerts. An integrated Real-Time Clock (RTC) module ensures accurate scheduling of reminders, maintaining time even during power outages. A 16x2 LCD display with I2C interface provides real-time information, displaying instructions and confirmation messages to guide the user. Powered via a USB cable or a 9V adapter, the entire setup is controlled by an Arduino UNO, which manages all interactions between components. The modular design makes the system easy to assemble, maintain, and upgrade, making it ideal for home use, elderly care, and healthcare facilities.

CHAPTER - 5

METHODOLOGY

1. RFID-Based Identification

Each medicine container is tagged with a unique RFID tag. The RFID reader detects and verifies the medicine when placed in the tray.

2. Real-Time Tracking

A Real-Time Clock (RTC) module keeps track of the current time to ensure accurate logging of dose timings.

3. Microcontroller Control

A microcontroller (e.g., Arduino or ESP32) processes data from the RFID and RTC modules and manages the system's operation.

4. Alert Mechanism

LEDs and a buzzer provide visual and audio alerts to indicate when a dose is due or missed.

5. Display Unit

An LCD displays key information such as patient ID, medication status, and scheduled timing.

6. Optional Cloud/App Integration

The system can be extended with a cloud database or mobile/web app for remote monitoring and notifications.

5.2 System Design

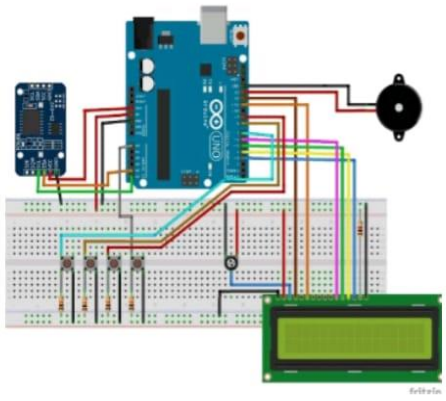

Figure.5.2 System circuit design

Explanation of the Circuit:

1. **Arduino Uno** acts as the central controller, reading inputs and driving outputs.
2. **Push-buttons** (with pull-down resistors) feed digital input pins on the Uno for user interactions.
3. **16×2 LCD** is wired in parallel to digital pins on the Uno, displaying text or data.
4. **Buzzer** connects to a digital output pin, producing sound for alerts or feedback.
5. **I²C module** (e.g. DS3231 RTC or a sensor) connects via SDA (A4) and SCL (A5), adding time-keeping or sensor data.
6. **Breadboard** organizes all power (5 V/GND) and signal lines, enabling plug-and-play prototyping.

CHAPTER - 6

RESULT AND DISCUSSION

6.1 Prototype Description

The **Smart Medi Tray** is a microcontroller-based medicine reminder system designed to assist patients in taking their medication on time over a three-day cycle. This system eliminates the use of colored LED indicators and instead uses **three dedicated buzzers**—each corresponding to a specific day (Day 1, Day 2, Day 3). When it's time for a dose on a particular day, the corresponding buzzer is activated to alert the user.

.

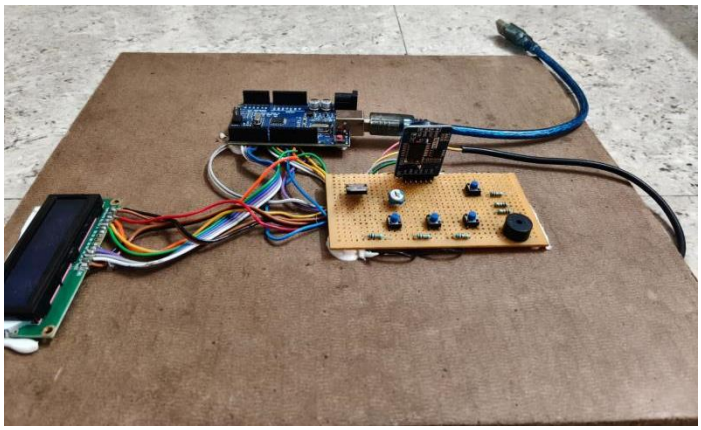

Figure.6.1 Prototype Model

The **Smart Medi Tray** is a microcontroller-based medicine reminder system designed to assist patients in taking their medication on time over a three-day cycle. This system eliminates the use of colored LED indicators and instead uses **three dedicated buzzers**—each corresponding to a specific day (Day 1, Day 2, Day 3). When it's time for a dose on a particular day, the corresponding buzzer is activated to alert the user.

The hardware setup includes an **Arduino Uno**, **push buttons**, **an LCD display**, and **four buzzers** (three for patient reminders and one for caretaker acknowledgment). The LCD display provides real-time information and pill timing notifications. After the pill is taken, the user confirms it by pressing a dedicated button, which deactivates the buzzer. If the dose is missed or not confirmed, the buzzer remains active until manual intervention.

Additionally, there is a **fourth buzzer** controlled by the caretaker. This ensures that the caretaker can monitor and verify if the patient is following the medicine schedule. The project is implemented on a perforated board and uses a neat arrangement of components for practical deployment and testing.

This system is highly useful for elderly patients or individuals who require regular medication but may have difficulty remembering or tracking doses. By providing daily alerts and involving caretakers, the system ensures better adherence to prescribed medication regimens.

CHAPTER - 7

CONCLUSION AND FUTURE WORK

Conclusion:

The circuit successfully integrates an Arduino Uno with a 16x2 LCD, push buttons, a buzzer, and an I2C module to create an interactive embedded system. It demonstrates basic principles of microcontroller-based design, including digital input/output, user interface, and peripheral communication. This setup can be used for applications such as timers, counters, clocks, or menu-driven systems.

Future Work:

1. **Add a Real-Time Clock (RTC):** If not already present, integrate a DS3231 RTC module to maintain accurate time tracking even during power loss.
2. **Implement EEPROM Storage:** Store settings or user preferences in EEPROM so they persist after power cycles.
3. **Enhance UI:** Upgrade to an I2C LCD to reduce wiring complexity or integrate a touchscreen for better interactivity.
4. **Wireless Connectivity:** Add a Wi-Fi or Bluetooth module (e.g., ESP8266, HC-05) to enable remote control or data monitoring.
5. **Sensor Integration:** Add temperature, humidity, or motion sensors to turn the system into a monitoring or automation device.
6. **Enclosure Design:** Design and 3D print a case to protect the circuit and give it a professional finish for practical deployment.
7. **Power Optimization:** Implement low-power modes or battery backup to make the device portable and energy-efficient.

REFERENCES

1. Banerjee S. Embedded Systems: A Contemporary Design Tool. Pearson Education; 2021:212–219.

2. Kumar R. Practical Arduino: Cool Projects for Open Source Hardware. Apress; 2017:95–102.

3. Singh M, Rani S. Internet of Things in Smart Healthcare Systems. CRC Press; 2020:145–153.

4. Raj P, Varalakshmi P. A smart medicine box for elderly people using IoT. Int J Eng Res Technol. 2018;7(4):185–188.

5. Nair R, Joseph A. Arduino based medicine reminder system. Int J Innov Res Comput Commun Eng. 2019;7(5):2305–2309.

6. Sharma A, Mehta K. IoT based health monitoring and medication system. Int J Sci Res Eng Manag. 2020;4(7):14–19.

7. Gupta P, Singh R. Smart pill reminder using Arduino UNO and GSM module. Int J Adv Res Electron Commun Eng. 2018;7(9):833–836.

8. Malarvizhi P, Santhosh D. Healthcare monitoring and alert system using Arduino. J Emerg Technol Innov Res. 2022;9(3):490–496.

9. Patil S, Kale R. Design and implementation of a digital medicine box with buzzer alert. Int J Res Electron Comput Eng. 2019;7(2):192–195.

10. Bhargava M, Kaur S. Real-time pill dispenser system using RTC and microcontroller. Int J Innov Technol Explor Eng. 2021;10(6):52–56.

11. Arduino.cc. Arduino UNO Technical Specifications. https://www.arduino.cc/en/Main/arduinoBoardUno. Published 2023. Accessed May 24, 2025.

12. World Health Organization. Medical Device Technical Series: Personal Health Monitoring Devices. WHO Press; 2019:20–35.

APPENDIX

```
//Medicine Reminder using Arduino Uno
// Reminds to take medicine at 8am, 2pm, 8pm
/*  The circuit:
   LCD RS pin to digital pin 12
   LCD Enable pin to digital pin 11
   LCD D4 pin to digital pin 5
   LCD D5 pin to digital pin 4
   LCD D6 pin to digital pin 3
   LCD D7 pin to digital pin 2
   LCD R/W pin to ground
   LCD VSS pin to ground
   LCD VCC pin to 5V
   10K resistor:
   ends to +5V and ground
   wiper to LCD VO pin (pin 3)*/
#include <LiquidCrystal.h>
#include <Wire.h>
#include <RTClib.h>
#include <EEPROM.h>
int pushVal = 0;
int val;
int val2;
int addr = 0;
RTC_DS3231 rtc;
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;              // lcd pins
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
#define getWellsoon 0
```

```
int buzz2pmHH = 14;          //    HH - hours
int buzz2pmMM = 00;          //    MM - Minute
int buzz2pmSS = 00;          //    SS - Seconds
int buzz8pmHH = 20;          //    HH - hours
int buzz8pmMM = 00;          //    MM - Minute
int buzz8pmSS = 00;          //    SS - Seconds
int nowHr, nowMin, nowSec;                  // to show current mm,hh,ss
// All messeges
void gwsMessege(){              // print get well soon messege
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("Stay Healthy :)");     // Give some cheers
   lcd.setCursor(0, 1);
   lcd.print("Get Well Soon :)");    // wish
}
void helpScreen() {             // function to display 1st screen in LCD
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("Press Buttons");
   lcd.setCursor(0, 1);
   lcd.print("for Reminder...!");
 }
void timeScreen() {             // function to display Date and time in LCD screen
  DateTime now = rtc.now();             // take rtc time and print in display
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("Time:");
   lcd.setCursor(6, 0);
```

```
    lcd.print(nowHr = now.hour(), DEC);
    lcd.print(":");
    lcd.print(nowMin = now.minute(), DEC);
    lcd.print(":");
    lcd.print(nowSec = now.second(), DEC);
    lcd.setCursor(0, 1);
    lcd.print("Date: ");
    lcd.print(now.day(), DEC);
    lcd.print("/");
    lcd.print(now.month(), DEC);
    lcd.print("/");
    lcd.print(now.year(), DEC);
}
void setup() {
  Serial.begin(9600);                  // start serial debugging
  if (! rtc.begin()) {                 // check if rtc is connected
    Serial.println("Couldn't find RTC");
    while (1);
  }
  if (rtc.lostPower()) {
    Serial.println("RTC lost power, lets set the time!");
  }
//    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));          // uncomment this
to set the current time and then comment in next upload when u set the time
  rtc.adjust(DateTime(2025, 4, 26, 06, 47, 15));             // manual time set
  lcd.begin(16, 2);
  lcd.clear();
  lcd.setCursor(0, 0);
```

```
  lcd.print("smart ");                          // print a messege at startup
  lcd.setCursor(0, 1);
  lcd.print("medi tray");
  delay(1000);
  pinMode(push1pin, INPUT);                     // define push button pins type
  pinMode(push2pin, INPUT);
  pinMode(push3pin, INPUT);
  pinMode(stopPin, INPUT);
  pinMode(ledPin, OUTPUT);
  delay(200);
  Serial.println(EEPROM.read(addr));
  val2 = EEPROM.read(addr);                     // read previosuly saved value of
push button to start from where it was left previously
  switch (val2) {
    case 1:
      Serial.println("Set for 1/day");
      push1state = 1;
      push2state = 0;
      push3state = 0;
      pushVal = 1;
      break;
    case 2:
      Serial.println("Set for 2/day");
      push1state = 0;
      push2state = 1;
      push3state = 0;
      pushVal = 2;
      break;
```

```
    case 3:
      Serial.println("Set for 3/day");
      push1state = 0;
      push2state = 0;
      push3state = 1;
      pushVal = 3;
      break;
  }
}
void loop() {
  push1();                          //call to set once/day
  push2();                          //call to set twice/day
  push3();                          //call to set thrice/day
    if (pushVal == 1) {                   // if push button 1 pressed then remind at
8am
    at8am();                          //function to start uzzing at 8am
  }
  else if (pushVal == 2) {                  // if push button 2 pressed then remind
at 8am and 8pm
    at8am();
    at8pm();                          //function to start uzzing at 8mm
  }
  else if (pushVal == 3) {                  // if push button 3 pressed then remind
at 8am and 8pm
    at8am();
    at2pm();                           //function to start uzzing at 8mm
    at8pm();
  }
```

```
  currentMillisLCD = millis();                    // start millis for LCD screen
switching at defined interval of time
  push1state = digitalRead(push1pin);             // start reading all push button pins
  push2state = digitalRead(push2pin);
  push3state = digitalRead(push3pin);
  stopinState = digitalRead(stopPin);
  stopPins();                                     // call to stop buzzing
  changeScreen();                                 // screen cycle function
}
// push buttons
void push1() {                  // function to set reminder once/day
  if (push1state == 1) {
    push1state = 0;
    push2state = 0;
    push3state = 0;
//    pushPressed = true;
    EEPROM.write(addr, 1);
    Serial.print("Push1  Written  :  "); Serial.println(EEPROM.read(addr));    //  for
debugging
    pushVal = 1;                                  //save the state of push button-1
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Reminder set ");
    lcd.setCursor(0, 1);
    lcd.print("for Once/day !");
    delay(1200);
    lcd.clear();
  }
```

```
}
void push2() {                    //function to set reminder twice/day
  if (push2state == 1) {
    push2state = 0;
    push1state = 0;
    push3state = 0;
//    pushPressed = true;
    EEPROM.write(addr, 2);
    Serial.print("Push2 Written : "); Serial.println(EEPROM.read(addr));
    pushVal = 2;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Reminder set ");
    lcd.setCursor(0, 1);
    lcd.print("for Twice/day !");
    delay(1200);
    lcd.clear();
  }
}
void push3() {                    //function to set reminder thrice/day
  if (push3state == 1) {
    push3state = 0;
    push1state = 0;
    push2state = 0;
//    pushPressed = true;
    EEPROM.write(addr, 3);
    Serial.print("Push3 Written : "); Serial.println(EEPROM.read(addr));
    pushVal = 3;
```

```
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Reminder set ");
    lcd.setCursor(0, 1);
    lcd.print("for Thrice/day !");
    delay(1200);
    lcd.clear();
  }
}
void stopPins() {                  //function to stop buzzing when user pushes stop push
button
  if (stopinState == 1) {
//    stopinState = 0;
//    pushPressed = true;
    pushpressed = 1;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Take Medicine  ");
    lcd.setCursor(0, 1);
    lcd.print("with Warm Water");
    delay(1200);
    lcd.clear();
  }
}
void startBuzz() {                    // function to start buzzing when time reaches to
defined interval
//  if (pushPressed == false) {
 if (pushpressed == 0) {
```

```
    Serial.println("pushpressed is false in blink");
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= interval) {
      previousMillis = currentMillis;        // save the last time you blinked the LED
      Serial.println("Start Buzzing");
      if (ledState == LOW) {                 // if the LED is off turn it on and vice-versa:
        ledState = HIGH;
      }  else {
        ledState = LOW;
      }
      digitalWrite(ledPin, ledState);
    }
  }
  else if (pushpressed == 1) {
    Serial.println("pushpressed is true");
    ledState = LOW;
    digitalWrite(ledPin, ledState);
  }
}
void at8am() {                   // function to start buzzing at 8am
  DateTime now = rtc.now();
  if (int(now.hour()) >= buzz8amHH) {
    if (int(now.minute()) >= buzz8amMM) {
      if (int(now.second()) > buzz8amSS) {
        ////////////////////////////////////////////////
        startBuzz();
        ////////////////////////////////////////////////
      }
```

```
    }
  }
}
void at2pm() {                    // function to start buzzing at 2pm
  DateTime now = rtc.now();
  if (int(now.hour()) >= buzz2pmHH) {
    if (int(now.minute()) >= buzz2pmMM) {
      if (int(now.second()) > buzz2pmSS) {
        ////////////////////////////////////////////////
        startBuzz();
        ////////////////////////////////////////////////
      }
    }
  }
}
void at8pm() {                    // function to start buzzing at 8pm
  DateTime now = rtc.now();
  if (int(now.hour()) >= buzz8pmHH) {
    if (int(now.minute()) >= buzz8pmMM) {
      if (int(now.second()) > buzz8pmSS) {
        ////////////////////////////////////////////////
        startBuzz();
        ////////////////////////////////////////////////
      }
    }
  }
}
//Screen Cycling
```

```
void changeScreen() {                //function for Screen Cycling
  // Start switching screen every defined intervalLCD
  if (currentMillisLCD - previousMillisLCD > intervalLCD)              // save the last
time you changed the display
  {
    previousMillisLCD = currentMillisLCD;
    screens++;
    if (screens > maxScreen) {
      screens = 0;  // all screens over -> start from 1st
    }
    isScreenChanged = true;
  }
  // Start displaying current screen
  if (isScreenChanged)   // only update the screen if the screen is changed.
  {
    isScreenChanged = false; // reset for next iteration
    switch (screens)
    {
      case getWellsoon:
        gwsMessege();               // get well soon message
        break;
      case HELP_SCREEN:
        helpScreen();              // instruction screen
        break;
      case TIME_SCREEN:
        timeScreen();               // to print date and time
        break;
      default:
```

```
            //NOT SET.
            break;
        }
    }
}
```