UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+ ACCREDITED UNIVERSITY

DIGITAL WATCH:

Project Report

Submitted to:

Navdeep Singh Sodhi

Submitted by:

Mahak Rana

24 MCA20156

in partial fulfilment for the award of the degree of

Master of Computer Application

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Chandigarh University

Aug 2024 – Nov 2024

Digital Watch Project Report

1. Introduction

- Objective: The primary goal of this project is to create a digital watch application that runs on a Linux platform, utilizing a Raspberry Pi or similar device to display the current time, set alarms, and provide additional timekeeping features.
- Scope: The project includes features such as:
  - Real-time clock display
  - Alarm settings with notifications
  - Stopwatch and timer functions
  - User-friendly interface for settings adjustment

2. System Requirements

Hardware Requirements

- Raspberry Pi: Any model with GPIO support (e.g., Raspberry Pi 3, 4).
- RTC Module: DS3231 or similar for accurate timekeeping.
- Display: LCD (e.g., 16x2) or OLED display for visual output.
- Buttons: Push buttons for user input (e.g., setting the time or alarms).

Software Requirements

- Linux Distribution: Raspberry Pi OS (formerly Raspbian) or any compatible Linux distro.
- Libraries:
  - WiringPi: For GPIO control.
  - I2C Tools: For communicating with the RTC module.
  -LCD Library: To interface with the LCD display.

3. Design and Architecture

System Architecture

- Diagram: Create a block diagram showing:
  - Raspberry Pi as the central controller
  - RTC module connected via I2C
  - LCD/OLED display for output
  - Buttons connected to GPIO pins for input

Flowchart

- Logic Flow:
- Start program
- Initialize RTC and display
- Enter main loop:
- Update display with current time
- Check for button presses
- Handle user inputs (set time, alarms, etc.)

User Interface Design

- Display should show:
  - Current time (HH:MM:SS)
  - Date (optional)
  - Alarm status (set/active)
- Button functionalities:
  - Button 1: Set time
  - Button 2: Set alarm
  - Button 3: Start/stop stopwatch

4. Implementation

Development Environment

- Set up a Raspberry Pi with the latest version of Raspberry Pi OS.
- Install necessary packages and libraries using terminal commands:

```bash
sudo apt-get update
sudo apt-get install wiringpi i2c-tools
```

Configuration

- RTC Setup: Connect the RTC module to the Raspberry Pi via I2C. Use the following commands to check the RTC:

```bash
sudo i2cdetect -y 1
sudo hwclock -r
```

- Display Configuration: Set up the LCD using the appropriate library. This may involve writing a simple test program to confirm functionality.

Key Features

- Current Time Display: Use a loop in a shell script to update the display every second.
- Alarm Functionality:
  - Store alarm time in memory.
  - Trigger a notification (e.g., sound, flashing display) when the alarm time is reached.

- User Settings: Implement functions to change time and alarm settings using buttons.

Sample Bash Script for Time Display

```bash
#!/bin/bash
# Loop to continuously display current time
while true; do
    clear
    current_time=$(date +"%H:%M:%S")
    echo "Current Time: $current_time"
    sleep 1
done
```

5. Testing and Evaluation

Testing Methodology

- Unit Testing: Test individual components (RTC, display, buttons) to ensure they function correctly.
- Integration Testing: Verify that all components work together seamlessly (e.g., setting an alarm triggers the correct output).
- User Testing: Conduct tests with potential users to evaluate the interface and usability.

Results

- Document any issues encountered during testing, such as:
- RTC not updating correctly

- Display glitches or miscommunication with the library
- Provide solutions implemented to resolve these issues.

6. Conclusion

- Summary: The project successfully created a functional digital watch on a Linux platform. All intended features were implemented, including accurate time display and alarm functionalities.
- Future Work: Consider adding:
  - More advanced timekeeping features (e.g., multiple alarms, snooze function).
  - A graphical user interface (GUI) for easier interaction.
  - Integration with network time protocol (NTP) for internet-based time syncing.

7. References

- Books/Articles: List any relevant literature or documentation consulted during the project.
- Online Resources: Include links to tutorials, forums, or GitHub repositories that provided guidance.

8. Appendices

- Wiring Diagrams: Include diagrams showing how components are connected.
- Detailed Code Listings: Provide the complete source code used in the project.
- User Manual: Instructions on how to use the digital watch, including setting the time and alarms.