B. Tech ECE BECE204L COURSE PROJECT Final Review

STOPWATCH USING 8051 MICROCONTROLLER

April 2025

Introduction

- The objective of this project is to design and implement a **digital stopwatch using the 8051 microcontroller.**

- The stopwatch features start, pause, reset, and lap timing functionalities, displayed on an **LCD screen**. Designed for power efficiency with minimal energy consumption

Problem Statement

- Timing devices are widely used in sports, laboratories, and industrial automation. A digital stopwatch provides precise timing measurements with user-friendly controls.
- This project demonstrates the practical implementation of **interrupts, timers, and LCD interfacing** in the **8051 microcontroller**, making it a valuable educational tool

Key Contributions

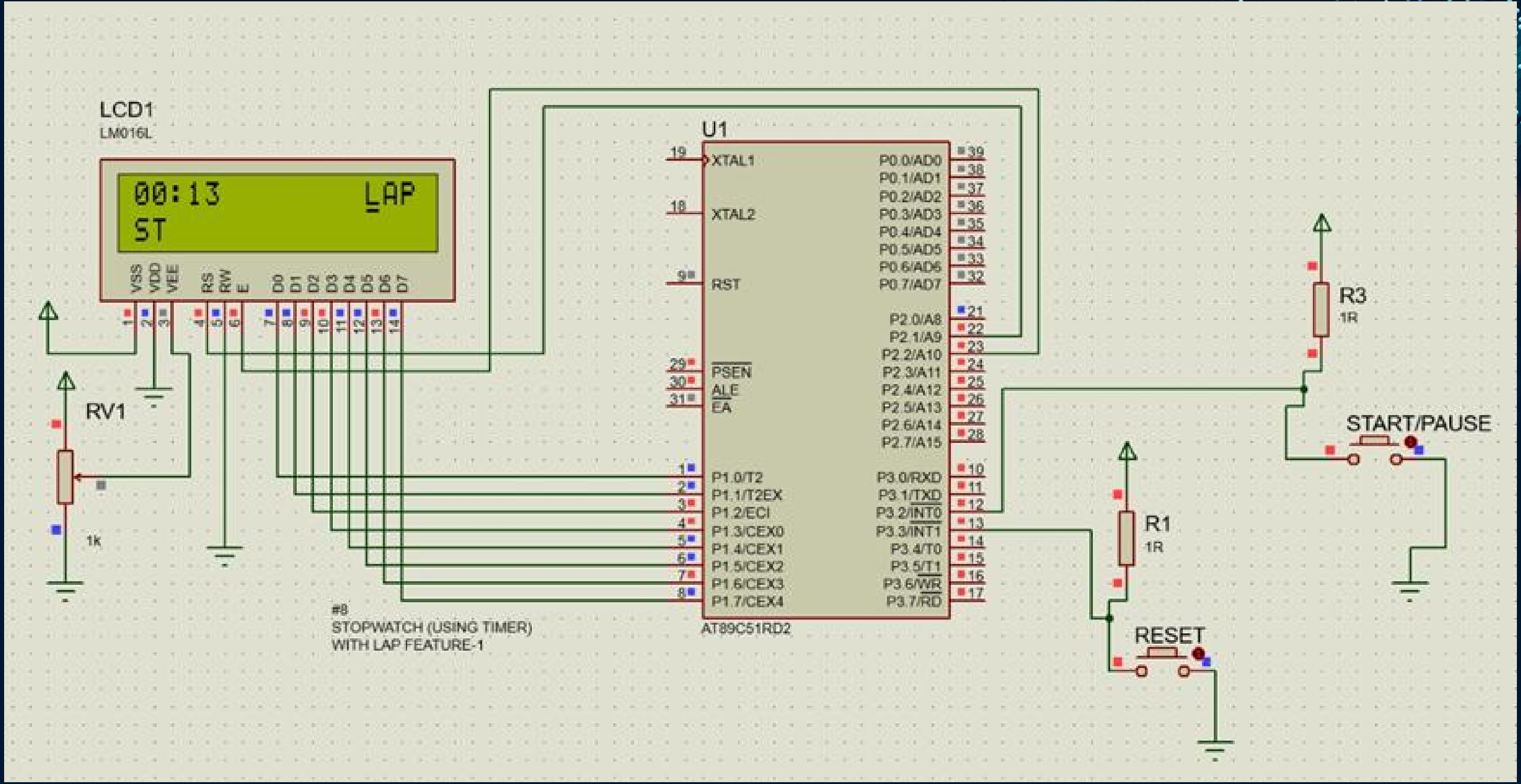
- Implemented state-based control: Running, Paused, Lap, Reset.
- Used external interrupts (INT0 & INT1) for user inputs.
- Designed an LCD interface to display real-time updates.
- Enhanced power efficiency using the PCON Register.

Relevance to Sustainable Development Goals (SDG)

This project aligns with **Sustainable Development Goal (SDG) 9 - Industry, Innovation, and Infrastructure**, as it focuses on **embedded systems development**, a key area in modern technology, helped us learn microcontroller programming.

Power Efficiency for Sustainable Tech – Consumes minimal energy, supporting sustainability

Proposed design



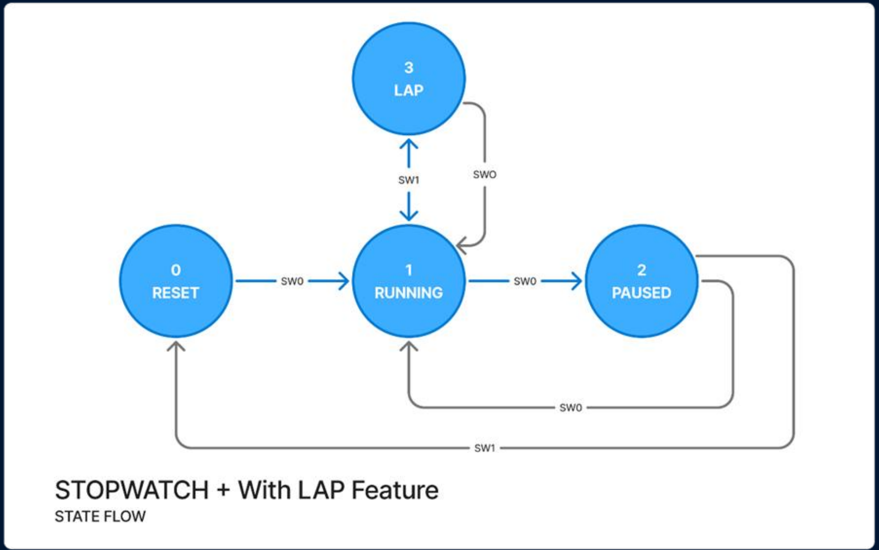
Theoretical Description

Defining the State Flow & Interrupt Handling

The state transitions are interrupt driven, the stopwatch operates in **four distinct states**:

- **Reset (State 0)** – The timer resets to **00:00**.
- **Running (State 1)** – The timer increments **every second**.
- **Paused (State 2)** – The timer stops incrementing but retains the displayed time.
- **Lap (State 3)** – The timer continues running, but the display temporarily shows a lap time.

Section 5



STOPWATCH + With LAP Feature

STATE FLOW

Firstly the state flow is verified using Memory Windows of Keil and the same is verified using Proteus simulations. (D: 30H holds the value of current state and the state changes when interrupt is pressed)

Theoretical Description

Counting 1 Second and Displaying on LCD

Timer is used in mode 1 (16 Bit Mode) to count generate 1 second delay. Since 1second is too large to be counted single cycle **(65536*1.085us=~71.1ms) .**
We are counting 50ms 20 times to generate 1s delay.

R1 keeps track of 20 overflows
R5,R6 stores the values of minutes and seconds respectively, since these cannot be displayed directly we need to convert them into ASCII before writing it to the LCD

(Subroutine for displaying time is given on the right).

It is made energy efficient by using the stopwatch in idle mode in reset and pause state since there is no need of CPU.(Current required in normal mode is 25 mA,while current required in Idle mode is 6.5mA.)*

```
DISPLAY_TIME:
  MOV A, #80H
  ACALL COMNWRT
  ; Tens of R5
  MOV A, R5
  MOV B, #10
  DIV AB
  ADD A, #'0'
  ACALL DATAWRT
  ; Ones of R5
  MOV A, B
  ADD A, #'0'
  ACALL DATAWRT
  MOV A, #':'
  ACALL DATAWRT
  ; Tens of R6
  MOV A, R6
  MOV B, #10
  DIV AB
  ADD A, #'0'
  ACALL DATAWRT
  ; Ones of R6
  MOV A, B
  ADD A, #'0'
  ACALL DATAWRT
  RET
```

TOOLS USED

HARDWARE:

- 8051 DEVELOPMENT BOARD
  PS-8051-EVB-V4
- LCD Module
  PS-LCD-ADDON-V2

SOFTWARE:

PROTEUS , KEIL UVISON , FLASH MAGIC

Simulation Results

https://drive.google.com/drive/folders/1cVPgsGd33i30Fe4njQjyXzObS76qPI97?usp=drive_link

Code:

https://drive.google.com/file/d/10HRlDJAWG4ZcGSeGmsuW90T5lJnCaWiJ/view?usp=sharing *(REQUEST ACCESS)*

Hardware Results

https://drive.google.com/file/d/1jedsUudRkQjvy4ArA5xv-Kfttqgb8yFi/view?usp=sharing

Future Work

Making the colon ":" blink in running state and static in pause state.

Using **CG RAM of LCD** for custom characters in Pause/Lap mode.

Code **optimization** for accurate timekeeping.

Altering state transition from 3 to 1 to match commercial stopwatches.(i.e., via INT0 then then INT1)

Conclusion

This project successfully implemented a **digital stopwatch using the 8051 microcontroller**. While doing this project we have learnt

- To efficiently use **interrupts for input handling**.
- To work with **timers for delay generation**.
- LCD interfacing for **real-time display updates**.
- **Microcontroller Simulation** using **Proteus** Software