Curtin University

School of Electrical Engineering, Computing and Mathematical Sciences

Design Assignment Report

CMPE2003

MICROCOMPUTERS

Submitted to:

Dr Nimsiri Abhayasinghe

Submission Date: 26/10 /2025 Time : 23:59 AWST

Group: Bentley (C3 Thu 7:00 AM)

Submitted by:

Name1 : Dhrubo Jouti Das Troyee Student ID:22663281
Name 2 : Asher Bailye Student ID : 22195755
Name 3 : Nazanin Rahimi Student ID : 21510234

29 January 2026

Table of contents

Table of Tables

Table of Figures

1. Introduction

This report presents the design, implementation, and testing of an embedded system based on the MSP430F5308 microcontroller. The system sees switch S3's ON time and compares it to a specified by the user threshold, working as a threshold-based stopwatch. While functioning effectively in low-power modes, it incorporates seven-segment output, LCD user feedback, keypad input, interrupt-driven control, and real-time clocking.

The key objectives of the project were:

- To configure the MSP430's core subsystems, including its clock, power management, and input/output interfaces.
- To communicate with legacy CLIC3 peripherals by implementing BusRead and BusWrite strategies.
- To build a real-time stopwatch counter (0–99) using timer interrupts and low-power operating modes.
- To connect with a 4×4 keypad, capturing a two-digit threshold value using interrupts.
- To display both the current stopwatch value and threshold on seven-segment displays (SSDs) and an I²C LCD.
- To activate LED D0 (threshold alarm) and LED D7 (S3 status) for displaying the system condition.
- To ensure low energy consumption by using LPM0 during operation and LPM3 when idle.

This work proves how modular assembly routines – covering initialization, bus control, lookup tables, interrupt handling (ISR), and application logic can be combined to produce a reliable and energy-efficient embedded design.

2. Problem Statement

This project's goal is to develop an interrupt-based embedded stopwatch satisfying the functional criteria listed below:

1. **System Initialization**
   On power-up, the system must configure the Power Management Module (PMM), Unified Clock System (UCS), and I/O ports. It must also initialize the I²C LCD and display the prompt *"Threshold:"*.

2. **Threshold Input (keypad-Based)**
   A keypad is continuously checked. A Port 2 interrupt detects key presses and allows entry of a two-digit threshold (00–99). This value is shown on the LCD.

3. **Stopwatch Operation using S3**

   - When S3 is pressed (ON), the LCD and SSDs display the elapsed-time counter, which increases once every second.

- When S3 is released (OFF) the timer ends and last value remains on the LED . Whenever S3 button is pressed, a new timing cycle begins, and the timer is reset to zero.

4. **S3 Status Indicator**
   The current condition of S3 must be reflected by LED D7, which should be ON when S3 is pushed and OFF when it is released.

5. **Threshold Alert (D0)**
   If the elapsed time exceeds the specified threshold, LED D0 blinks at about 2 Hz to alert the user. If not, D0 turns off.

6. **LCD User Input**
   LED highlights prompts and status like "Enter Threshold," “stopped” “Elapsed: XXs," and "Threshold Exceeded” to assist the user.

7. **Power Efficiency**:
   Two low-power operation modes are used by the system to decrease energy consumption**:** LPM0 is used for stopwatch timing, When timing is finished, LPM3 is utilized in idle mode, wake on PORT2 interrupts or TimerA1.

8. **Modularity and Scalability**
   All assembly routines must be modular, allowing reuse and potential system extension in future designs.

3. Hardware Configuration

The system is built around the MSP430F5308 microcontroller which controls:

- On-chip peripherals such as TimerA and the I²C module.
- External peripherals accessed through a software defined pseudo bus using BusRead and BusWrite.

3.1 Pin Assignments

Pin(s)	*Function*	*Connected Component*	*Reference*
P1.7	TA1.0 Output	TimerA1 output (500 Hz toggle)	[TIMSP430F530x Datasheet, TimerA Section]
P2.0	GPIO Input (Interrupt)	Keypad input trigger for PORT2_ISR.asm	[MSP430F530x User Guide, Port Interrupts]
P4.1(UCB1SDA)	I²C Data	LCD serial data line	[MSP430F530x User Guide, eUSCI_B I²C Mode]
P4.2(UCB1SCL)	I²C Clock	LCD clock signal	[MSP430F530x User Guide, eUSCI_B I²C Mode]

P4.6	GPIO Output	Bus Enable (E) signal for CLIC3 legacy bus	[Bus Interface Notes]
P4.7	GPIO Output	Bus 'R/W' (Read/Write) control for CLIC3 legacy bus	[Bus Interface Notes]
P5 (Port5)	GPIO Output/Input	4-bit data/address bus for legacy bus	[Bus Interface Notes]
PJ (PortJ)	GPIO Output	4-bit latch enabled for SSD and LED selection.	[Bus Interface Notes]
Via BusWrite	Peripheral Output	DIS1/DIS2 7 seg Displays	Digital.Lookup.asm
Via BusWrite	Peripheral Output	LED D7 (S3 status Warning) LED D0(Threshold Alarm.)	asm. s43

Table 1 – Hardware configuration – Pin Assignments

3.2 Schematic overview:

The MSP430F530 is connected to :

- a legacy bus for SSDs, keypads, LEDs via P4.6, P4.7, P5 and PJ.
- the LCD through the I²C pins (P4.1/P4.2)
- the input of the S3 switch (accessible via BusRead)
- TimerA and logic for interrupts
- LCD slave address is 0x3E.

3.3 Software-Defined Pseudo-Bus Device Codes:

Actual MSP430 memory-mapped I/O is not used to access external peripherals on the CLIC3 board; instead, a software-implemented legacy bus is used . To choose devices using GPIO-controlled bus signals, logical 16-bit codes are created in **asm.s43** and transmitted to **BusRead** and **BusWrite**

Serial no	*Address*	*Symbol*	*Address details*
1	**0x4000**	switchAddress	S3 Switch Input Port
2	**0x4002**	ledAddress	LED Output Port
3.	**0x4004**	segLowAdd	DIS1 -Lower 7-Segment Display(units)
4	**0x4006**	segHighAddr	D1S2 Upper 7-Segment Display(tens)
5	**0x4008**	KeyPadAddr	Keypad Input Port

Table 2 : software-Defined Pseudo-Bus Device Codes

3.4 Internal Hardware Selections

- **Watchdog Timer** :Disabled at startup to avoid resets.

- **Power management (PMM)**: By upgrading Vcore to Level 3, higher-frequency performance (~25 MHz) is possible.
- **Clock system (UCS):** The DCO is set to about 25 MHz. For CPU functioning, MCLK is obtained via DCO. Additionally, I²C communication and TimerA functionality depend on SMCLK, which is sourced from DCO.
- **TimerA1 (time base) :** ACLK is used by TimerA1 to create a 1-second time base. When the threshold exceeds, it sets D0 blinking (~2 Hz) and provides elapsed-time updates.
- **eUSCI_B1 (I²C)** : Configured as master for LCD communication.
- **Keypad Interrupt (Port 2 ISR):** A rising-edge event on P2.0 (the keypad input pin) activates the Port 2 interrupt service routine (ISR). To guarantee accurate key detection, the ISR debounces the signal upon activation. It then converts the key press to the appropriate index using a lookup table. A two-digit threshold value between 00 and 99 is created using this index and immediately displayed to the LCD for user input.

4. Software Configuration

The system's software is written in assembly and divided into multiple files, each with a specific role. A specific function, such as hardware configuration, bus communication, input processing, display logic, or time control, is handled by each module. Additionally, this modular approach makes the code easier to debug, reuse, and integrate.

4.1 Initialization (Initial.asm)

After a reset, this module runs once and gets the microcontroller ready for use. It does the following things:

- To avoid accidental resets, turn off the watchdog timer.
- Sets the pseudo-bus pins' I/O directions (P4.6, P4.7, P5, PJ).
- Facilitates handling of global interrupts.
- MCLK and SMCLK sources are initially set up for the clock system (UCS) utilizing DCO ≈ 25 MHz.
- Raises the Vcore level and sets up the power system (PMM) for a reliable operation.

4.2 Legacy Bus Interface (BusRead2.asm, BusWrite2.asm)

This mechanism simulates a full 16-bit address/data bus using only GPIO pins. By developing a software-driven I/O protocol over GPIO, these modules allow the MSP430 to access a 4-bit pseudo-parallel bus to operate legacy peripherals on the CLIC3 board.

- BusWrite: uses PJ latch to toggle P4.6 (Enable) and P4.7 (R/W) while transmitting a 16-bit value as four nibbles on P5
- BusRead: chooses the device, converts a 16-bit word into BusData, runs four reads, and sets P5 as input.

Only GPIO pins are used by this software-based bus to represent a memory-mapped I/O interface.

4.3 Main Application Logic (asm.s43)

This is the main control application that uses threshold logic and stopwatches. It handles these tasks:

1. Configures the threshold by entering two-digit numbers on a keypad.
2. Using LCD interaction to provide results ("Elapsed: XXs") and prompts ("Threshold:").
3. Stopwatch operation:
   - When S3 is turned on, it refreshes the SSD and LCD, jumps up the counter every second (TimerA1 ISR), and compares with the threshold.
   - When S3 is off, time remains frozen, SSDs are cleared, and LPM3 low-power mode is activated.
4. LED warning logic**:**
   - S3 status is displayed by LED D7.
   - If the threshold exceeds the limit, LED D0 blinks (~2 Hz).

4.4 Interrupt Handling

To preserve real-time response and lower power processing, interrupts are implemented.

- TIMER1_A0_ISR: uses TimerA1 CCR0 to create 1-second ticks and clears LPM0 to activate the CPU.
- PORT2_ISR When the P2.0 keypad is pressed, debounces the signal, reads it using BusRead, maps the key using DigitalLookup.asm, converts it to BCD, updates the threshold, and refreshes the LCD.

5. Testing & Result:

To ensure effective operation as required with the project criteria, the system was tested on the CLIC3 board. Validation tests were conducted for LCD user feedback, LED indicators D0 and D7, seven-segment display output, threshold entering, and stopwatch timing via switch S3. Below is evidence from the actual process.

5.1 Test outcomes

The system after setup is highlighted in Figure 1 below. The LCD validates the stored threshold after a threshold value of 20 seconds was entered correctly using the keypad.

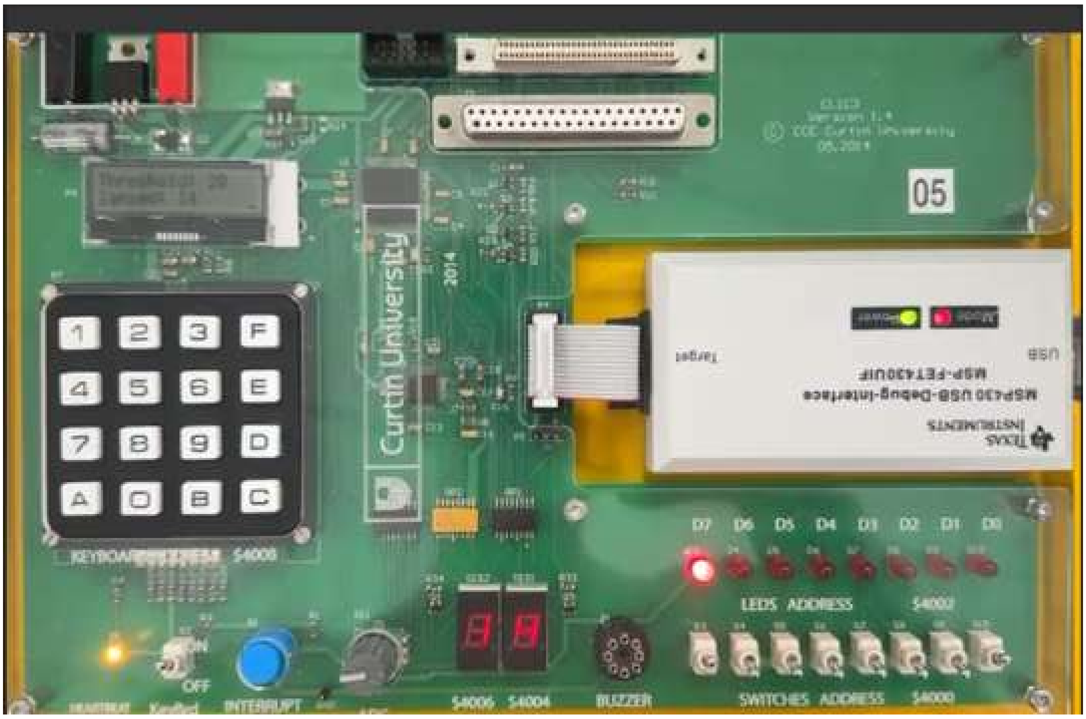

Figure 1: The threshold =20 s entered via keypad and showed on the LCD.

Pressing switch S3 activates the stopwatch's timing. On the dual seven-segment displays (SSD), the elapsed time is displayed and increased once every second. When S3 is being pressed, LED D7 turns on.

Figure 2: Stopwatch running, D7 ON indicating S3 active, elapsed time shown SSDs.

To satisfy the threshold alert requirement, LED D0 automatically starts blinking (figure3), as a visual alarm to notify the user when the elapsed time reaches the threshold value (20 seconds).

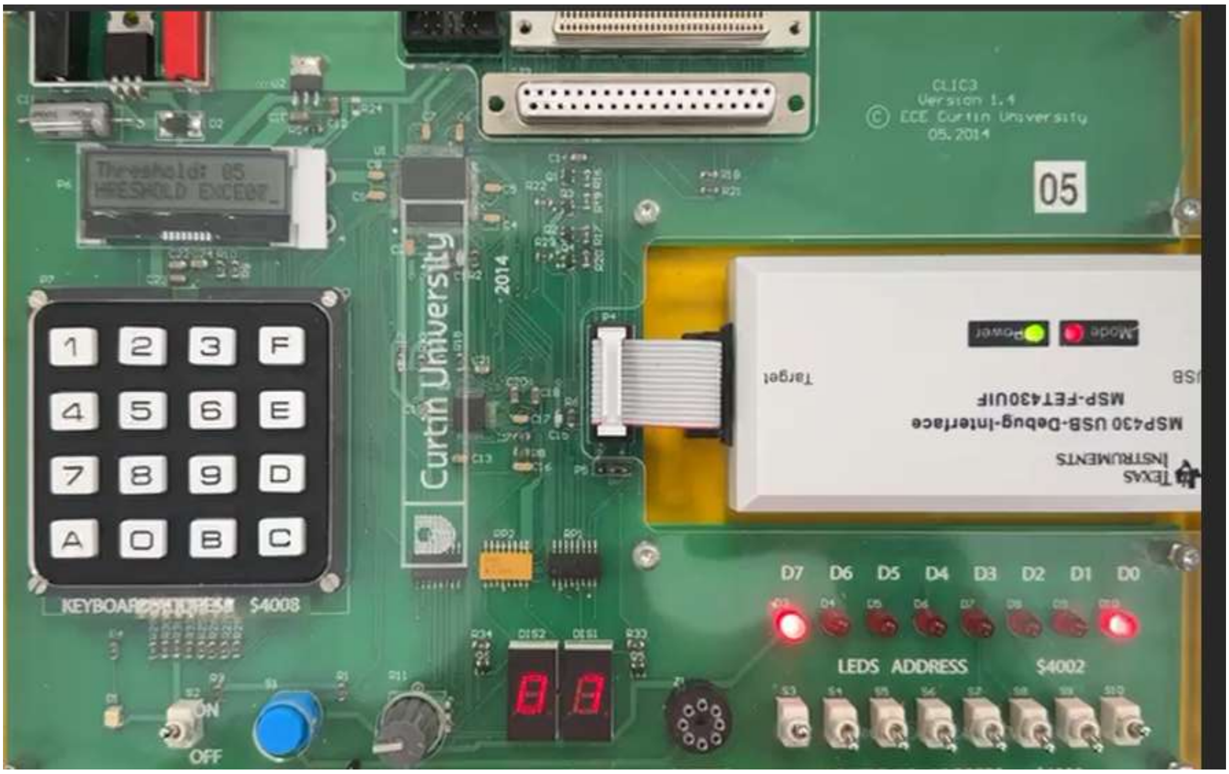

Figure 3:LED D0 blinking alarm when threshold is exceeded.

5.2 Test summery table:

Test No	Short Description	Expected Outcome	Observed Outcome	Pass/Fail
1	System setting up	LCD screen requests for threshold input	LCD working and ready to go	**Pass**
2	A threshold entry via keypad	2-digit threshold stored and viewed	Threshold=20 displayed on LCD screen	**Pass**
3	Operation of S3 triggering	Counter activity starts when S3 pressed	SSD increasing successfully	**Pass**
4	S3 release behaviour	Timing finishes and holds the last value .	Time operated correctly	**Pass**
5	Threshold alarm	The alarm activates above the threshold.	D0 blinking after 20s	**Pass**
6	Notification of S3 status	LED signifies switch condition	D7 ON if S3 activated	**Pass**

Table 3: Test summery table.

5.3 Analysing the Results

According to all test findings, this system satisfies the functional criteria. Keypad input handling via PORT2 interrupt, threshold storage, stopwatch control via TimerA, and output display via LCD (I^2C) and seven-segment display were part of the hardware and software interactions that were successfully validated. LEDs D0 and D7 for the alarm logic and warning lights responded exactly as planned.

6. Sustainability Considerations

The design of the system emphasizes sustainability in terms of social utility, economic efficiency, and environmental effect

6.1 Environmental

An essential part of this design is its power efficiency. Several low-power modes that dramatically reduce energy usage are supported by the MSP430F5308:

- The CPU disables itself while supporting system clocks during stopwatch operation by switching to Low-Power Mode 0 (LPM0) in between timer interruptions ((MSP430F530x Datasheet, User Guide). By doing this, less energy is lost during idle cycles.
- Low-Power Mode 3 (LPM3) is engaged while the stopwatch is not in use (S3 released). Only necessary peripherals are still operational in this mode, which results in low current consumption because the DCO is turned off.
- Active polling is not used since peripheral activity is interrupt driven.

These features extend battery life and reduce the overall energy footprint of the device.

6.2 Economic

The design emphasizes cost efficiency by maximising reuse and avoiding unnecessary hardware:

- The system reuses existing CLIC3 board peripherals such as the keypad, seven-segment displays, and LEDs without requiring more modules.
- The software-driven pseudo bus replaces separate decoder or driver ICs, reducing hardware cost and PCB complexity.
- A single microcontroller handles processing, display control, timing, and communication. This reduces the bill of materials (BOM) while still achieving full functionality.

6.3 Social

The design emphasizes security, simplicity, and accessibility:

- Safety awareness is increased by visual alerts; LED D0 blinks to provide an instant threshold-warning alert, and LED D7 displays the system condition in real time.
- Usability is enhanced by clear feedback mechanisms: the LCD offers straightforward messages and instructions, and the SSDs provide readable numerical values.

- For those who are new to embedded systems, the system is accessible as well as educationally helpful because it is straightforward to use and interact with using a keypad.
- By highlighting bus connections, interrupts, low-power design, and modular assembly programming, the design helps engineering education and benefits upcoming students.

7. Discussion and Reflection

7.1 Achievements

The project's fundamental functional criteria were effectively fulfilled.

- Custom assembly methods were used to combine the MSP430F5308 with legacy CLIC3 peripherals.
- The pseudo memory-mapped bus supported secure communication because the BusRead and BusWrite modules operated effectively.
- The implementation of a dual-display output featured the LCD generating threshold feedback and system notifications, while the SSDs handled numerical output.
- Using a TimerA1 interrupt-based approach in combination with low-power operation by LPM0 and LPM3, the system also performed precise time monitoring.

Overall, both displays and keypad input presented a straightforward, efficient user layout

7.2 Challenges

During development, some sort of technical obstacles appeared:

- **ISR Complexity**:
  the keypad interrupt service routine (ISR) needed to handle both LCD updates and debounce logic, leading to it expensive and increased latency. Simplifying the ISR and transferring display updates to the main loop enhanced the architecture.
- **Bus Timing Problems:**
  Incorrect keypad reads sometimes occurred because of insufficient trigger pulses. The problem was resolved by adding tiny delays.
- **Stack Overflows**:
  Pushing too many registers without balance caused errors. Debugging required reviewing subroutine calls and register usage, emphasizing the importance of careful stack management.

7.3 Graduate Capabilities

- **Discipline Knowledge**: The report enhanced the practical application of low-level embedded concepts such bus interface, interrupts, low-power modes, and finite-state timing.
- **Problem Solving**: Debugging BusWrite required oscilloscope checks; stack issues required restructuring.

- **Teamwork:** Group members worked together on tasks. Asher performed the hardware testing. Moreover, he converted the C code into assembly and integrated routines from previous labs to ensure everything worked correctly. Troyee helped with validation and debugging, while Nazanin helped with LCD control logic and code transfer from C to assembly. Continuous progress and the project's successful completion were guaranteed by regular collaboration between teams.

8. Individual Contribution

Member Name	Student ID	Specific Contributions
Dhrubo Jouti Das Troyee	22663281	Contributed to report development by formatting, organizing, and completing sections that were missing. Ensured the report was consistent and ready for submission. Assisted in early C coding and provided support during testing.
Asher Bailye	2219575	Converted C code into assembly and ensured all modules(BusRead/BusWrite, interrupts, display drivers) worked together. Debugged hardware/software interactions and finalized the working design.
Nazanin Rahimi	21510234	Worked on the first two stopwatch tasks in C (time counting and basic counter functions). Helped with testing on hardware and took the lead in documenting hardware/software design and writing the draft report.

Table 4: Individual Contribution

9. References

[1] Texas Instruments, *MSP430F530x Mixed-Signal Microcontroller Datasheet*, 2014.
[2] Texas Instruments, *MSP430x5xx and MSP430x6xx Family User's Guide*, 2013.
[3] Curtin University, Initial.asm, BusRead2.asm, BusWrite2.asm, Lab Resources, 2025.
[4] LCD Module Datasheet, I²C Interface, 2024.