Design Assignment

Design and Implement an Embedded System on CLIC3 Board

Learning outcomes

By completing this assignment, you will be able to:

- Design, implement and test an embedded solution.
- Interface user inputs (keyboard), displays (seven-segment and LCD) and indicator LEDs.
- Produce a formal technical report and demonstrate/debug your system.
- Apply interrupt-driven programming and timer peripherals in embedded systems.
- Apply energy-aware programming / low-power design considerations to reduce device energy consumption.

Objective

Design and implement an interrupt-based embedded system on the CLIC3 board using Assembly or C. Assembly solutions that meet the requirements will receive full credit; C implementations are permitted but may be graded with a smaller premium.

Problem statement / Functional requirements

Implement a system on CLIC3 board with the following behaviour (CLIC3 is shown in Figure 1):

1. **Measure ON time of S3**
   - Each time S3 is turned ON, start timing using the internal timer(s). When S3 goes OFF, stop timing. Measure the elapsed time in **seconds**.
2. **Display elapsed time**
   - Show the ON time in seconds on the two-digit seven-segment displays: **DIS2 (tens)** and **DIS1 (units)**. DIS1 is the least significant digit. Format: 00 to 99 seconds. If elapsed >= 100 s, display 99 (or handle overflow per your design — document choice).

3. **Threshold entry**
   - At program start, request the user to enter a **threshold time (in seconds)** via the keyboard (numeric keypad). Display the prompt on the LCD and echo the entered value. Store the threshold for comparison.

4. **Threshold alarm**
   - If the measured ON time **exceeds the threshold**, the LED **D0** must **start blinking** at a visible rate (e.g., 2 Hz). The blinking should continue while the condition holds.

5. **S3 status indicator**
   - The LED **D7** must reflect the instantaneous state of S3: ON → D7 = ON; OFF → D7 = OFF.

6. **Behaviour on repeated presses**
   - **Every time S3 goes ON**, the timer for this activation should reset and begin timing that activation (i.e., measure each turn on duration independently).

7. **Robustness**
   - Implement **hardware/software debouncing** for S3. Use interrupts for switch ON/OFF detection and use hardware timer interrupts for accurate timing.

8. **User feedback on LCD**
   - Provide useful messages on the LCD: prompt for threshold entry, current threshold value, status messages (e.g., "Timing…", "Elapsed: xx s", "THRESHOLD EXCEEDED"), and brief error messages if needed.

Non-functional / implementation notes

- Use the internal timers and interrupt facilities of the microcontroller. Document which timer(s) and interrupt sources you used and why.
- Aim to measure elapsed time to ±1 second accuracy. If you use sub-second timing (encouraged), explain your method.
- Prefer Assembly programming (full marks). If using C, indicate which low-level routines are implemented in Assembly (if any).
- Keep code modular and well-commented. Provide a README that explains build and run steps.

Sustainability

To encourage sustainable design thinking, include one or more of the following and document them:

- Use low-power modes between events (e.g., sleep/idle when waiting for S3/keypad).
- Minimise active CPU time (efficient polling avoidance; use interrupts).
- Use minimal peripheral activation (turn off unused modules).
  If you implement these, describe the approach and estimate (qualitatively) the expected energy savings. You will be awarded additional credit within the Implementation/Quality rubric.

Deliverables

1. **Code** (source + build instructions). Name files clearly.
2. **Executable / hex** to flash on the CLIC3 board.
3. **Demonstration** showing the system working through the main behaviours (threshold entry, ON/OFF S3 short/long, D0 blinking when exceeded, D7 state, displays).
4. **Final report (formal report format)** — max **10 pages** (excluding appendices and references). Report must include:
   - Title, authors, student IDs, date.
   - Brief introduction & objectives.
   - Design & architecture (timers, interrupts, I/O mapping).
   - Key code snippets and explanation (appendix for full code).
   - Testing and results (include screenshots or photos and observations).
   - Sustainability considerations implemented (if any).
   - Short reflection and limitations.

Mark Breakdown

- Functionality & Requirements Met — **30%**
  - Correct operation of all required behaviours (timing, displays, threshold entry/display, D0 blinking, D7 indicator, debouncing, interrupts).
- Implementation Quality & Robustness — **45%**
  - Efficient use of timers/interrupts, clean modular code, correctness, reliability, and handling of edge cases. Although C language programming may be used, usage of Assembly language will be best appreciated.
- Sustainability Considerations — **15%**
  - Explicitly identify and implement sustainability aspects in your design (e.g., low-power operation, efficient interrupt-driven design, minimised resource use, reusability of code). Must be documented and demonstrated.
- Report & Documentation — **10%**
  - Formal report in correct format, clear explanations, diagrams, references, Turnitin report attached. Format will be available with Blackboard report submission link.

Hints & tips

- Details of addresses of CLIC3 board would be found in the library files you have used in Lab 1.
- Use edge-triggered interrupts for S3 (ON and OFF). If hardware lacks separate edges, use a single interrupt and sample pin to detect state.
- Use a timer in periodic mode (e.g., 1 ms or 100 ms tick) and count ticks to compute seconds, but keep CPU usage low by doing the timekeeping in an interrupt service routine.
- Implement a small state machine: IDLE -> WAIT_DEBOUNCE -> TIMING -> STOP -> DISPLAY -> IDLE. This improves clarity and debugging.
- When threshold is exceeded, toggle D0 in a timer ISR or using a second timer to get reliable blinking.
- Test debouncing thoroughly — contact bounce will ruin timing if not handled.
- Validate keypad input (no negative numbers, reasonable upper bound, e.g., 99 s). Echo each digit on the LCD as it is entered.

Submission instructions & deadline

-

- Demonstrations of your work will be conducted during scheduled lab classes during last two teaching weeks of the semester (from 6th October 2025).

- Submit the group report (Turnitin similarity report) and all codes via the Blackboard assignment link by **11:59 pm on 26th October 2025**.

- Late submissions will incur penalties as per unit policy unless an approved extension is provided.

- Instructions on the submissions are available on Blackboard.

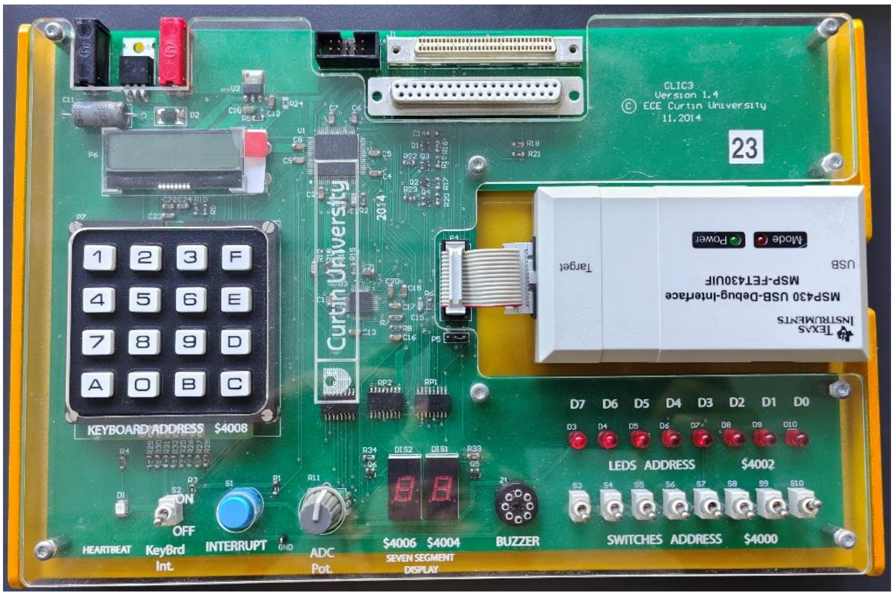

Figure 1 – CLIC3 Board

Rubric

Criterion	Unsatisfactory (<50%)	Good (50–64%)	Very Good (65–79%)	Excellent (80–100%)	Weight
Functionality & Requirements Met	Major requirements missing; system unreliable or fails to run.	Some requirements met; partial functionality demonstrated.	Most requirements correctly implemented; minor errors in operation.	All requirements fully met and system functions smoothly and robustly.	**30%**
Implementation Quality & Robustness	Poor code quality; polling-based, inefficient, or highly error-prone. ***OR*** Some use of timers/interrupts; limited modularity; moderate reliability using **C**.	Some use of timers/interrupts; limited modularity; moderate reliability using **Assembly**. ***OR*** Interrupt driven design with the usage of counter modules implemented in **C**.	Good implementation with correct use of timers/interrupts; mostly efficient and reliable using **Assembly**.	Highly efficient, interrupt-driven, robust, and professional-quality implementation using **Assembly**.	**45%**
Sustainability Considerations	No evidence of addressing sustainability.	Mentions sustainability superficially but not implemented.	Some sustainability features implemented (e.g., partial low-power operation, efficient coding), with reasonable discussion.	Strong sustainability focus: efficient interrupt-driven design, power-aware implementation, minimised resource use, clearly documented.	**15%**
Report & Documentation	Report missing, poorly structured, or incorrect format.	Report present but lacks clarity, formatting, or key details.	Clear and mostly professional report with minor issues.	Highly professional, well-structured formal report; clear visuals, correct referencing, Turnitin included.	**10%**