DESN2000 (Computer Engineering) 2026 T2
Lab sheet 4 (weeks 8 and 9)

Last edited: 11/07/2026

Make reasonable assumptions if not explicitly stated and state such assumptions to your demonstrator when getting marked.

Task 1 (10%)

Write a programme that slowly increases the brightness of LEDs D1-D4 on the coast board and LD2 on the NUCLEO board. Once 100% brightness is reached, slowly decrease the brightness back to 0% and repeat the process. You must use hardware PWM if the corresponding microcontroller pins have such hardware PWM support. For pins without such hardware PWM, you should implement the PWM signal using software.

Task 2 (25%)

Write a programme that lets the user enter a *sequence* of musical notes through the keypad and then plays that sequence back through the buzzer using hardware PWM.

Each note is entered as a **frequency value (integer, in Hz)** followed by the # key to confirm and append it to the sequence. The * key is a backspace on the current number being typed; if the current number is empty, * deletes the most recently confirmed note from the sequence. The LCD must interactively display both the number (frequency value) currently being typed **and** the count of notes stored so far (e.g. `Freq: 440 Notes:` 3).

Pressing **SW1** stops accepting keypresses and plays the whole sequence back once as a series of short beeps, with each note held for a fixed duration with sufficient silence between beeps, so it is not unpleasant (no continuous tone).

Pressing **SW2** stops playback and returns to input mode with the existing sequence preserved, so the user can edit and extend it rather than starting from scratch.

In addition, pressing **SW3** will replace the current sequence with a tune of your choice (must be longer than 5 notes/values). It should still require **SW1** to commence playing it. It should play a recognisable tune rather than random pitches

You must use **PWM** for the buzzer. Your programme must reject invalid input gracefully: frequencies outside a sensible audible range (roughly 20 Hz–20 kHz) should not be accepted, and the LCD should indicate the rejection rather than just ignoring it.

Task 3 (25%)

Write a programme that reads ambient light from **one LDR of your choice** and displays the light level simultaneously on two different outputs driven from that single sensor reading.

Fine display: LEDs D1–D4 must fade smoothly in brightness, proportional to the level of **darkness** with darker surroundings produce brighter LEDs. These LEDs should use **hardware PWM**, and the brightness must respond convincingly across the *full range* of ambient light in the room (it will be tested with 3 main cases, phone torch, ambient light and full covering).

Coarse display (shift-register bar): the D5–D20 shift-register LED bar must act as a level meter, lighting a number of LEDs proportional to the light level, with **more darkness lighting more** of the bar.

Both displays must be driven from the **same** LDR reading and update continuously in real time. You must use hardware PWM for the pins that support it.

Task 4 (20%)

Write a programme that rotates the motor on the board when the SW1 button is pressed. The motor should stop when the SW2 button is pressed. The SW3 button should double the motor's current speed, while SW4 should halve it. These speed adjustments should be repeatable as long as they remain within practical limits. The motor's current speed, in revolutions per minute (RPM), must be displayed on the LCD screen.

Task 5 (20%)

Write an ARM assembly function that delays the number of milliseconds specified as an argument using busy waiting. Note that the delay should be very accurate (it will be tested with a stopwatch), which means you should consider the clock frequency of the processor and the number of instructions in your function. Once the assembly delay function is implemented, call this function from a C programme (which you may use HAL) that takes a user input from the keypad (an integer) and waits for the requested number of seconds after the '#' key on the keypad is pressed. As soon as the timeout is reached, indicate it by lighting LEDs D5-D20 on coaST board. The user should be able to clear the LEDs and start another round when the SW1 button is pressed.