Programming Phase

Objectives:

The objective of this task is to test and debug your donut board connections, test all ports and components, and make sure that the board is fully functioning.

Main Tasks:

Program and test all components in your donut board.

Subtask 1:

Program your 3 programmable LEDs to blink with 5Hz frequency all at the same time.

Subtask 2:

Program your 2 programmable push buttons to switch the LEDs ON and OFF. Each button should control one LED, based on the state of the button:

- If the button is pressed, → LED **ON.**
- If the button is released, → LED **OFF**.

Subtask 3:

Write a program where every 500ms increases the servo angle by **45°**. Continue increasing the angle until the maximum rotation angle (180°) is reached. After reaching 180°, the next step should reset the servo back to 0° and continue the sequence as follow:
0° → 45° → 90° → 135° → 180° → 0° → 45° … .

Subtask 4:

Read the analog value from the **potentiometer** using the **ADC** and use it to adjust the blink speed of one of the **LEDs.** Repeat this for all 4 potentiometers.

Subtask 5:

Program your STM32 bluepill to transmit the message: “`Hello world!`” every time you reset your board using the UART interface, then receive the message from your phone or laptop.

Optional Task:

Fully AI documentations won’t be accepted. **Support your documentation with visualized content** and list all resources used (websites, youtube videos, …).

Write a documentation about the concept of RTOS, and more details about the Free RTOS, make sure to include the following topics:

- The main concept of RTOS, definition and its functionality.
- Why RTOS is needed compared to a general-purpose OS.
- Explain RTOS Architecture: Kernel and scheduler.
- Explain RTOS components: Tasks, Queues and Semaphores and mutexes?
- What is FreeRTOS?
- Practical Example (optional but very useful). Simple FreeRTOS code: LED blinking with two tasks.

RTOS and FreeRTOS are very wide concepts, and have a lot of components and functionalities. No need to dive deep in it, just write about the overview and general concept.

Hints:

Introduction to RTOS concept:

- https://youtu.be/F321087yYy4?si=6BXowolJrpAnqaYa
- https://www.freertos.org/Documentation/01-FreeRTOS-quick-start/01-Beginners-guide/01-RTOS-fundamentals

Introduction to Free RTOS concept:

- https://youtu.be/kP-pP6FEu8I?si=3tWDxCL9-QHslnDp
- https://youtu.be/JIr7Xm_riRs?list=PLXyB2ILBXW5FLc7j2hLcX6sAGbmH0JxX8

Servo motor safety rules :

- **Avoid Manually Forcing the Servo Shaft:** Do not try to manually rotate or force the servo motor shaft or gears. Servo motors have delicate internal gears that can break easily when forced by hand.
- **Do Not Block the Servo While Moving:** Avoid physically blocking the servo motor shaft during operation. This can cause the servo to draw its maximum current, potentially overheating and damaging the servo, and in some cases causing overcurrent that may harm the STM32 board.

Submission:

- Record a video of each subtask and send it to your mentor.
- Complete and submit this task in one PDF file for both main and optional (if done) tasks.
- Add a picture of your circuit connections to the PDF file.
- Add a screenshot of the configurations of IOC to the PDF file.
- Add a screenshot of the code of each subtask to the PDF file (only what you have modified in the generated code).
- Name the PDF files with task11_groupx_your_name, (replace x with your group number).
- This task should be submitted before 18th Sept 10:00pm (Malaysia time), 05:00pm (Makkah time).
- Your mentor must approve your task answer file before submission.
- Submit the PDF file to the Google form.