Name	**Sujit Kumar Sethi**
REG NO.	**2101020692**
Branch	**CSE(Software Engg.)**

Project-Report

Problem Statement:

Implement a menu driven utility using uart Rx, where based on what is entered it should control various operations as listed below.

1. Should invoke led blinking but should keep changing the delay of the LED blinking by press on Button (PC13), and after 3 or 4 press's it should come back to its starting delay.

2. Should invoke a PWM and varying intensity of light based on button pressed (min 3 intensities required and should come back to its original intensity when button is pressed 4rth time).

3. Exit the utility..

Algorithm:

STEP 1->Select all the pins required for the project.

- *Initialize system clock.*
- *Initialize UART for communication.*
- *Initialize GPIO for LED and button.*
- *Initialize PWM for LED intensity control.*
- *Enable external interrupt for button press.*
- *Display menu options.*
- *Read user input via UART.*
- *Perform action based on user input.*

STEP 2->Initialize all the GPIO & Peripheral pins

- *Set up the system clock for appropriate speed and timing.*
- *Configure UART for communication.*
- *Set baud rate, data bits, stop bits, and parity.*
- *Enable UART receive interrupt.*
- *Configure GPIO pin for LED as output.*
- *Configure GPIO pin for button as input with pull-up/pull-down resistors if necessary.*
- *Set up external interrupt on the button pin.*
- *Configure GPIO pin for PWM output.*
- *Set up PWM timer with appropriate prescaler and auto-reload value.*

- *Enable PWM output on the configured pin.*
- *Enable NVIC interrupt for external button press.*
- *Set priority and enable the interrupt.*

STEP 3->Wait for user request to INPUT data.

- *Send menu options over UART:*
  - *Option 1: LED Blinking*
  - *Option 2: PWM Intensity Control*
  - *Option 3: Exit*
- *Wait for user input via UART.*
- *Parse input to determine the selected option.*
- Perform Action Based on Input:
  - ***If Option 1 (LED Blinking)****:*
  - *Invoke LED blinking function.*
  - ***If Option 2 (PWM Intensity Control)****:*
  - *Invoke PWM intensity control function.*
  - ***If Option 3 (Exit)****:*
  - *Exit the utility and halt the system.*

STEP4-> LED Blinking Function

- Initialize LED Blinking Variables:
  - *Set initial delay value.*
  - *Define an array of delay values.*
  - *Toggle LED state.*
  - *Delay based on the current delay value.*
  - *Check if the button was pressed:*
  - *Increment delay index.*
  - *Wrap around to the initial delay after reaching the last delay value.*

STEP5-> PWM Intensity Control Function

- *Set initial duty cycle value.*
- *Define an array of duty cycle values.*
- *Set PWM duty cycle based on the current duty cycle value.*
- *Check if the button was pressed:*
  - *Increment duty cycle index.*
  - *Wrap around to the initial duty cycle after reaching the last value.*

STEP6-> Button Press Interrupt Handler*:*

- *Check if the interrupt was triggered by the button pin.*

- *Clear the interrupt flag.*
- *Set a flag indicating the button was pressed.*

Connection details:

Sl No	Peripheral	Port & Pin No	Type	Description	Mode
1	LED1	PA5	OUTPUT	Indicates status of data capture	Input
2	LED2	PA6	OUTPUT	Indicates data transmission	Output
3	EXT_13	PC13	INPUT	Call the Interrupt function	Alternate function
4					Analog

Conclusion / Future scope:

Future Scope:

1. Enhanced User Interface:
    - Implement a more sophisticated user interface with feedback messages and error handling to improve user experience.
2. Additional Functionalities:
    - Introduce new options in the menu for controlling other peripherals or performing additional tasks, such as motor control or sensor data acquisition.
3. Remote Control:
    - Extend the UART communication to support remote control via Bluetooth or Wi-Fi, allowing the utility to be operated from a mobile device or computer.
4. Data Logging and Monitoring:
    - Add functionality to log and monitor the status and performance of the LED and PWM control, which can be useful for debugging and analysis.
5. Interrupt Debouncing:
    - Implement more sophisticated debouncing algorithms for the button press to ensure reliable operation.

6. Low Power Modes:
    - Incorporate low power modes to enhance the energy efficiency of the system, especially for battery-powered applications.
7. Scalability:
    - Make the code more modular and scalable to support more complex applications and additional hardware components.
8. Safety and Reliability:
    - Add safety checks and fault tolerance mechanisms to make the system more robust and reliable in various operating conditions.

Conclusion:

This project lays a solid foundation for creating interactive and responsive embedded systems with UART communication, GPIO control, and PWM signal generation. By building on this foundation, future enhancements and features can be added to develop more advanced and versatile applications.