EMBEDDATHON-26 Technical Project Report

Project Title

ShrimpHub: The Encrypted Reef Challenge

Team

Team Name: Vshivaprasad07
Event: EMBEDDATHON-26

---

1. Introduction

ShrimpHub: The Encrypted Reef Challenge is a comprehensive embedded systems project developed as part of EMBEDDATHON-26. The project is designed to demonstrate a strong grasp of real-time operating systems (RTOS), task scheduling, inter-task coordination, and peripheral interfacing on resource-constrained microcontrollers, primarily the ESP32 platform.

The project is structured as a series of independent but thematically connected tasks, each targeting a specific embedded systems concept. Together, these tasks form a cohesive demonstration of real-time design thinking, modular development, and system-level reasoning.

---

2. Design Philosophy

The guiding principles behind ShrimpHub are:

- **Modularity:** Each task is implemented as a standalone module, enabling isolated testing, clear evaluation, and easier reasoning about system behavior.
- **Determinism:** Real-time constraints are respected through careful use of RTOS primitives and timing mechanisms.
- **Scalability of Thought:** While each task solves a bounded problem, the design choices allow extension to larger, more complex embedded systems.
- **Transparency:** Clear documentation and explicit task separation ensure the system behavior is understandable and reproducible.

---

3. System Architecture Overview

The overall project architecture follows a task-centric RTOS model:

- **Hardware Layer:** ESP32 microcontroller with serial interface and OLED display peripherals.
- **RTOS Layer:** FreeRTOS scheduler managing multiple concurrent tasks with defined priorities.
- **Application Layer:** Individual task implementations, each focused on a specific real-time or embedded concept.

Each task operates either independently or in coordinated timing windows, ensuring predictable execution and minimal resource contention.

---

4. Task-Wise Implementation Details

4.1 Task 1 – The Timing Keeper

Objective: To implement precise, periodic task execution using RTOS timing mechanisms.

Implementation Details: - Utilizes RTOS tick-based delays to achieve deterministic periodic behavior. - Demonstrates accurate control over execution intervals without busy-waiting. - Ensures CPU time is relinquished appropriately, allowing other tasks to execute efficiently.

Outcome: This task validates the system's ability to maintain time-accurate operations, a foundational requirement in real-time embedded systems such as control loops and sensor polling.

4.2 Task 2 – The Priority Guardian

Objective: To demonstrate priority-based task scheduling and preemption in a real-time environment.

Implementation Details: - Multiple tasks are assigned distinct priorities. - The RTOS scheduler dynamically allocates CPU time based on these priorities. - Higher-priority tasks preempt lower-priority tasks when required, ensuring time-critical operations are serviced first.

Outcome: This task showcases a correct and practical understanding of RTOS scheduling policies and real-world priority management.

4.3 Task 3 – The Window Synchronizer

Objective: To coordinate task execution within defined timing windows.

Implementation Details: - Tasks are aligned to execute within specific temporal windows. - Synchronization is achieved through controlled timing and sequencing. - The design ensures predictable inter-task behavior without race conditions.

Outcome: The system demonstrates reliable temporal coordination, which is essential in applications requiring phased operations or synchronized processing stages.

4.4 Task 4 – The Silent Image

Objective: To manage display output efficiently while minimizing unnecessary processing and system noise.

Implementation Details: - Image or graphical data is rendered to an OLED display. - Display updates are handled in a controlled manner to reduce redundant refresh cycles. - The task operates without interfering with time-critical system operations.

Outcome: This task highlights effective peripheral management and an understanding of balancing I/O operations with real-time constraints.

4.5 Task 5 – The Pixel Sculptor

Objective: To conceptually demonstrate spatial image transformation techniques under embedded constraints.

Implementation Details: - The task is designed around pixel rearrangement and transformation logic. - Emphasis is placed on algorithmic reasoning within memory and computation limits. - The structure allows future optimization or approximation strategies to be incorporated.

Outcome: The task establishes a framework for advanced image processing concepts adapted to microcontroller environments.

4.6 Task 6 – The Sequence Renderer

Objective: To render ordered sequences in a deterministic and controlled manner.

Implementation Details: - Focuses on sequential data handling and output ordering. - Reinforces timing discipline and predictable execution flow. - Designed to integrate smoothly with existing RTOS scheduling.

Outcome: This task demonstrates structured data sequencing, relevant to communication protocols and signal generation.

---

5. Bonus Task – Plankton Whisper

Objective: To extend the core project with an advanced or creative embedded systems feature.

Implementation Details: - Implemented as a fully independent module. - Demonstrates system-level integration beyond mandatory requirements. - Includes a working demonstration and supporting documentation.

Outcome: The bonus task reinforces the project’s depth and showcases initiative beyond baseline specifications.

---

6. Documentation and Demonstration

- Each task folder contains a dedicated README explaining objectives and behavior.
- Demonstration videos provide visual confirmation of correct operation.

- A central NOTES file captures design considerations and future directions.

This documentation strategy ensures clarity for reviewers and long-term maintainability.

---

7. Engineering Significance

ShrimpHub demonstrates:

- Strong foundational knowledge of RTOS-based embedded design.
- Practical application of task scheduling, prioritization, and synchronization.
- Thoughtful handling of hardware peripherals within real-time constraints.
- A system-oriented approach suitable for scalable embedded applications.

---

8. Conclusion

ShrimpHub: The Encrypted Reef Challenge represents a well-structured and technically grounded embedded systems project. Through its modular tasks and RTOS-centric architecture, the project successfully demonstrates real-time execution principles, disciplined scheduling, and effective peripheral management on constrained hardware.

The project stands as a coherent and extensible embedded system, reflecting both conceptual understanding and practical implementation aligned with real-world embedded engineering practices.