Introduction to Operating System Final Project

FreeRTOS Source Code Tracing and Analysis

Two students per group

Each group must briefly describe the division of labor (2–3 sentences) in your report.

Submission Format:

1. Written Report (PDF, Font size requirement: 12 pt. Page limit: 15 pages.)

   At the end of the report, please include each group member's individual reflection on the project and on the course (written separately by each person).

2. Video Presentation (Upload to YouTube and put the link in your PDF file, Video Length limitation: 20 minutes)

3. You can use Chinese or English for repot and video.

Deadline: 2026/1/9 23:59

FreeRTOS Source code (Please download the latest version and specify the version you used at the beginning of your report.): https://www.freertos.org/

Part 1 — Written Report

Answer the following questions in your PDF report. This assignment is based on the ARM Cortex-M4 architecture and uses the corresponding FreeRTOS port as the reference. Architecture-related implementation details (e.g., stack initialization, exception return behavior, and context switching) can be found in the files under FreeRTOS/Source/portable/GCC/ARM_CM4F/.

Using online articles, blogs, FreeRTOS tutorials, or StackOverflow explanations **as direct answers** is prohibited.
You may read them, but your submission must be written *entirely in your own words* and based on your own code tracing.

For each question in Part 1, your answer should include:

- A **code tracing flow** (function call sequence)
- **Key code snippets** (with file and function names)
- **Explanatory diagrams** (stack layout, list structure, scheduling flow, etc.)
- A **short summary paragraph** explaining your understanding

Each answer must be based on actual FreeRTOS code and may include file names, functions, diagrams, or code snippets where appropriate.

Q1 — Describe how FreeRTOS allocates and initializes a new task, inserts it into the ready list, and prepares it for scheduling.

Q2 —Describe how FreeRTOS moves tasks between Ready and Blocked states.

Q3 — Analyze how the scheduler selects the next task and in Task Control Block (TCB): Which fields control scheduling behavior?.

Q4 — How does FreeRTOS prevent race conditions when modifying task lists?

Q5 — Please trace the Heap Implementations (heap_1 ~ heap_5) of FreeRTOS, and compare the five heap allocators and summarize their differences.

Bonus Question (Optional, +10)

Q6. Trace the FreeRTOS context switch implementation.

Explain:

1. Which registers are saved/restored by hardware and which by software
2. How the PendSV handler interacts with vTaskSwitchContext()
3. How the new task's stack pointer is restored before returning to Thread mode

Part 2 — Video Presentation (15–20 minutes)

The video should demonstrate how you traced the FreeRTOS source code and how they reached their conclusions.

You only need to demonstrate the tracing process for 1–2 questions.

- Code tracing demonstration

- Explanation of the answer of the questions.

<u>The video must focus on the process, not only the final answer.</u>

The presentation should include:

- How you navigated the source code
- How you located key functions using search tools (which tool?)
- How you connected different files and logic
- Why you believe this function is the correct one
- Simply reading the written report will **not** be counted as a valid video submission.

Video Requirements:

- Duration: 15–20 minutes
- Must show real FreeRTOS code tracing
- Voice narration required

Number of Questions Answered	Score Range
1 questions + Video	40–54 points
2 questions + Video	55–69 points
3 questions + Video	70–79 points
4 questions + Video	80–89 points
5 questions + Video	90–100 points

✧ **Grading:**

Criterion	Description
Technical Accuracy	Correctness of code tracing, explanations, and understanding of FreeRTOS behavior.
Depth of Code Analysis	Appropriate use of function call flows, references to relevant files, and meaningful interpretation of the source code.
Use of Diagrams and Flowcharts	Clear and self-made diagrams (stack layout, scheduler flow, list structure, etc.) that enhance understanding.
Clarity and Structure	Well-organized writing, coherent explanation, and concise summaries.
Originality	Answers must reflect the student's own tracing work and cannot rely on AI-generated analyses.
Reflection	Each student must provide an individual reflection on the project and on the course. Reflections should show genuine insight rather than generic statements.

AI Usage Policy (AI Allowed with Disclosure + Mandatory Code Tracing)

General Principle

AI tools **can be used** as part of your learning process, **but they cannot replace your own understanding or source-code tracing.**

You must still:

1. Read FreeRTOS source code yourself
2. Perform the tracing work yourself
3. Produce your own explanations
4. Demonstrate the tracing workflow in your video (with your own voice and materials)

1. Allowed AI Usage (with disclosure)

Students **may** use AI for the following purposes:

(A) Asking conceptual or high-level questions

Examples:

“What is a task control block (TCB) in general OS design?”

“What does a scheduler typically do?”

“What is stack initialization in a typical RTOS?”

(B) Getting hints or directions

Examples:

Asking where to start reading FreeRTOS

Asking for clarification of terms (e.g., PendSV, xPSR)

(C) Grammar and writing refinement

Examples:

Polishing the English in your report

Improving clarity of sentences

Fixing typos and formatting

(D) Explaining how AI was used

You must clearly state:

“AI was used for X and Y purposes, but all code tracing and analysis were done manually.”

2. Mandatory Requirement: You must still trace the source code yourself

Regardless of AI usage, the following must be **your own work**:

1. All code tracing
2. All explanations of FreeRTOS behavior
3. All diagrams (stack layout, scheduler flow, ready list structure…)
4. All interpretations of tasks.c, list.c, port.c, heap_x.c
5. All reasoning steps shown in the video
6. The video must show your real tracing process, not AI output
7. AI **cannot** be used to generate these parts.

3. Prohibited AI Usage

AI tools may **not** be used for:

(A) Generating direct answers to the assignment questions

- No AI-summaries of FreeRTOS code
- No AI-produced diagrams
- No AI-generated call flows
- No AI-written explanations

(B) Generating or assisting video content

- No AI-generated voice
- No AI-generated avatars or presentations
- No AI-generated screen recordings
- Video must feature **your real screen + your real voice**

4. Required AI Usage Disclosure Section

At the end of your written report, include a short section titled:

“AI Usage Statement”

Example format:

AI Usage Statement:
I used AI tools to ask general conceptual questions about real-time operating systems and to help with grammar polishing in my final report.
All FreeRTOS source-code tracing, technical explanations, diagrams, and the video demonstration were completed manually by our group.

Failure to include this section → **−10 points**.