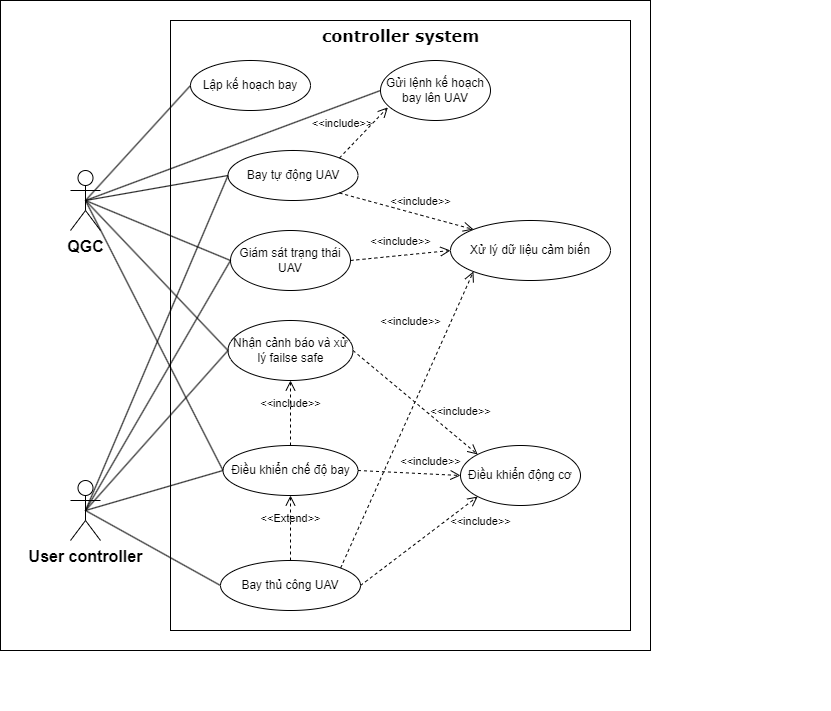
SYSTEM ARCHITECTURE DOCUMENT

# Giới thiệu và mục tiêu

Tài liệu này mô tả hệ thống điều khiển cho UAV, được coi như là một trung tâm diều khiển của UAV. Hệ thống sử dụng bao gồm khối điều khiển trung tâm (STM32H743) và khối cảm biến STM32F103.

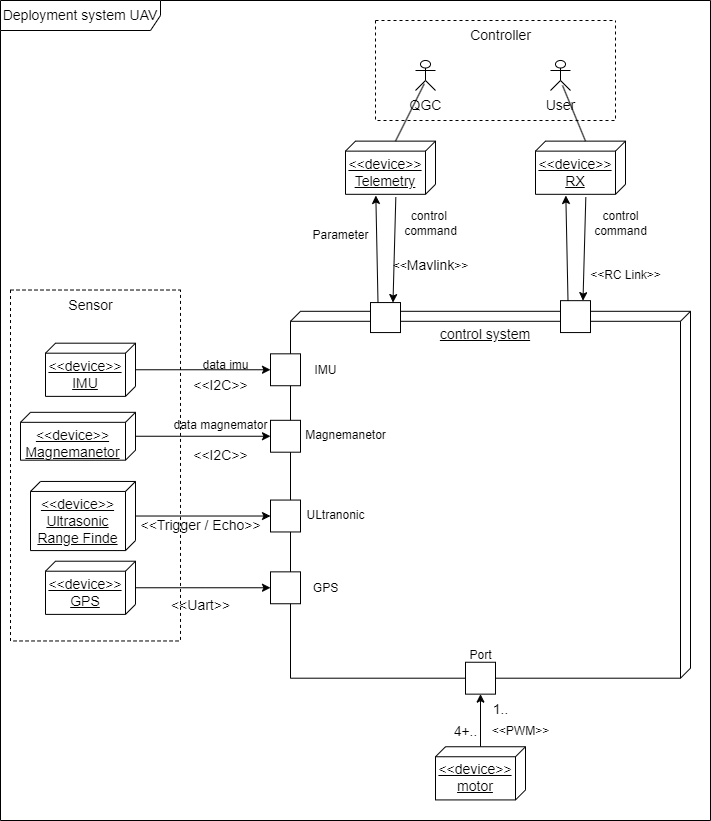
## Yêu cầu



Mô tả hoạt động của hệ thống

|  |  |  |
| --- | --- | --- |
| ID | Yêu cầu | Giải thích |
| R1: | Cơ chế thể điều khiển thủ công | Sử dụng RX điều khiển UAV thủ công, có thể tùy chỉnh sang chế độ bay tự động bằng cơ chế trên tay điều khiển, có cơ chế nhận thông báo false safe. |
| R2: | Cơ chế điều khiển tự động | Sử dụng phần mềm GRoundControl để điều khiển UAV, nhận về thông số trạng thái bay, cảnh báo false safe và đưa ra lệnh, kế hoạch điều khiển UAV. |
| R2.1: | In log dữ liệu |  |
| R3: | Xử lý cảm biến, nhận tín hiệu điều khiển và điều khiển động cơ. | Hệ thống điều khiển nhận và xử lý dữ liệu từ các cảm biến, kết hợp với lệnh điều khiển và điều chế xung cho động cơ. |

# Bối cảnh kỹ thuật



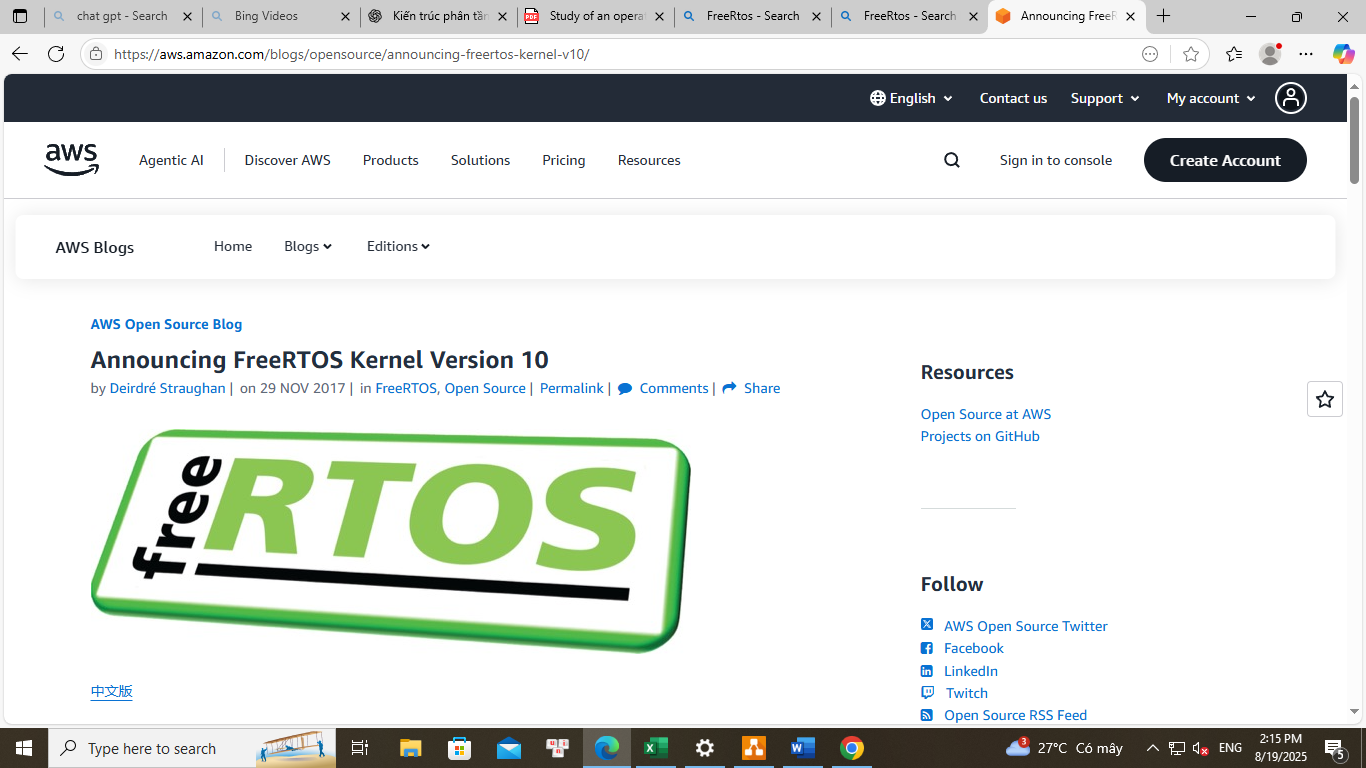
Các thành phần tương tác với hệ thống

|  |  |
| --- | --- |
| Phần ngoài control system | Mô tả |
| Controler | Nhận dữ liệu điều khiển từ controller và trả về các thông số trạng thái. |
| Sensor | Đọc dữ liệu từ cảm biến để xử lý |
| Motor | Điều chế xung điều khiển động cơ |

# Kiến trúc phần mềm

## Giới thiệu về FreeRTOS

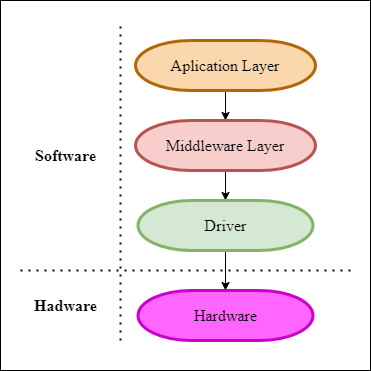
FreeRTOS là một hệ điều hành thời gian thực (Real-Time Operating System – RTOS) được ứng dụng nhiều trong hệ thống nhúng. FreeRTOS cho phép vi điều khiển chay nhiều tác vụ (task) song song với nhau một các độc lập, các task có thể chạy định kì thông qua Timer software gọi hàm callback hoặc kích hoạt các task theo chu kỳ [1]. Hệ điều hành cũng hỗ trợ quản lý task, queue, semaphore, mutex, timer và các cơ chế đồng bộ khác, gúp trao đổi dữ liệu giữa các task một cách an toàn và hiệu quả[1].



Logo hệ điều hành FreeRTOS

Ứng dụng thực tế, FreeRTOS được sử dụng để tổ chức các vụ song song trong hệ thống, giúp tăng tính ổn định và đáp ứng thời gian thực cho các chức năng trong khối cảm biến và khối điều khiển chính. Do cấu trúc firmware được thiết kế theo kiểu module do đó dễ dàng them bớt các task nhiệm vụ mà không ảnh hưởng đến hệ thống tổng thể, giúp mã nguồn dễ dàng tái sử dụng và mở rộng cho nhiều ứng dụng khác nhau.

## Kiến trúc chung của cà 2 khối phần mềm



Kiến trúc phân tầng hệ thống

Kiến trúc chung của hệ thống phần mềm được chia thành bốn tầng: Application, Middleware, Driver và Hardware.

|  |  |
| --- | --- |
| Tầng | Chức năng |
| Tầng Hardware | Bao gồm các phần cứng trực tiếp giao tiếp với hệ thống, như IMU, Magnetometer, Barometer… |
| Tầng Driver | Tách biệt phần mềm khỏi phần cứng cụ thể bằng cách cung cấp các API chuẩn, cho phép truy cập và điều khiển phần cứng một cách thống nhất. |
| Tầng Middleware | Cung cấp các dịch vụ chung, làm cầu nối giữa tầng Application và tầng Driver. Trong FreeRTOS, tầng Middleware chịu trách nhiệm quản lý các task, lập lịch, giao tiếp giữa các task, đồng thời cung cấp API giúp tầng Application triển khai chức năng dễ dàng và mở rộng. |
| Tầng Application | Xử lý các chức năng chính của hệ thống. Các chức năng này được triển khai dưới dạng các task cụ thể. |

Danh sách chức năng của các tầng hệ thống

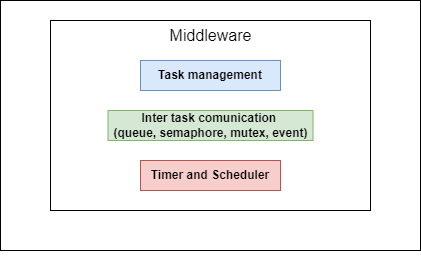
Trong dự án này, trọng tâm nghiên cứu và phát triển là hai tầng Middleware và Application, nơi đảm nhận việc quản lý tài nguyên, điều phối task và xử lý logic chính của hệ thống [2].

## Middleware

### FreeRTOS trong middleware

Middleware là tầng trung gian giữa tầng application và tầng Driver, cho phép tầng application tách biệt với phần drive, cung cấp các phương thức giao tiếp làm đơn giản hóa cho tầng application.

FreeRTOS được đơn giản hóa trong lớp middleware, có nhiệm vụ tạo và quản lý task, giao tiếp giữa các task, định thời, trừu tượng hóa phần cứng.



Mô tả chức năng trong Middleware

Task management cung cấp giao diện tạo và quản lý task, bao gồm: đặt độ ưu tiên, stack size từng task, quán lý danh sách stack và trạng thái.

Inter task communication: khởi tạo và cung cấp các cơ chế giao tiếp an toàn giữa các task queue; tạo API cho phép đọc dữ liệu đơn giản hóa trong tầng application.

Timer và Scheduler định thời cho các task theo chu kỳ cố định, cung cấp chế độ một lần hoặc chu kỳ để callback hoặc kích hoạt task.

## Chi tiết lớp Application của khối cảm biến

|  |  |  |
| --- | --- | --- |
| Tên Task | Chức năng | Tần số |
| readIMUTask() | Đọc dữ liệu gyro và accelerometer theo 3 trục của cảm biến và xử lý dữ liệu thô thành dữ liệu thực tế | 400hz |
| readMagnometerTask() | Đọc giá trị từ kế theo 3 trục của cảm biến | 50hz |
| readUltraSonicTask() | Đo khoảng cách đến vật thể | 5hz |
| readGPSTask() | Lấy dữ liệu từ GPS và xử lý dữ liệu GPS | 5hz |
| powerMonitorTask() | Đo điện áp nguồn điện | 5hz |
| PWMMotorTask() | Điều chế xung cho động cơ | 200hz |
| comunicationTask() | Giao tiếp với khối điều khiển trung tâm | 100hz |
| readPressureTask() | Đọc dữ liệu áp suất tính độ cao | 10hz |

Các task trong khối cảm biến

## Chi tiết lớp Application của khối điều khiển chính

|  |  |  |
| --- | --- | --- |
| Tên task | Chức năng | Tần số |
| sensorComunicationTask() | Lấy dữ liệu từ khối cảm biến | 200hz |
| GCScomunicationTask() | Giao tiếp với trạm mặt đất | 50hz |
| controlRXTask() | Giao tiếp với tay điều khiển | 250h |
| externKalmanFilterTask() | Thuật toán EKF tính EHRS | 200hz |
| PIDTask() | Dùng trạng thái EKF để tính trạng thái điều khiển | 200hz |
| flightControlTask() | Điều khiển hoạt động của Drone và xử lý fail safe | 200hz |

Các task trong khối điều khiển chính

**Tiếng Anh**

[1] Tomas Docekal and Zdenek Slanina, “Control System Based on FreeRTOS for Data Acquisition and Distribution on Swarm Robotics Platform”, ICCC, 2017, pp. 435-438.

[2] Tammy Noergaard, *Embedded Systems Architecture,* Embedded Technology Series, 2005, pp. 445-447