BÁO CÁO BÀI TẬP

(LINK DRIVE VIDEO DEMO)

(LINK YOUTUBE VIDEO DEMO)

1. Tóm tắt yêu cầu bài toán:

Sử dụng STM32:

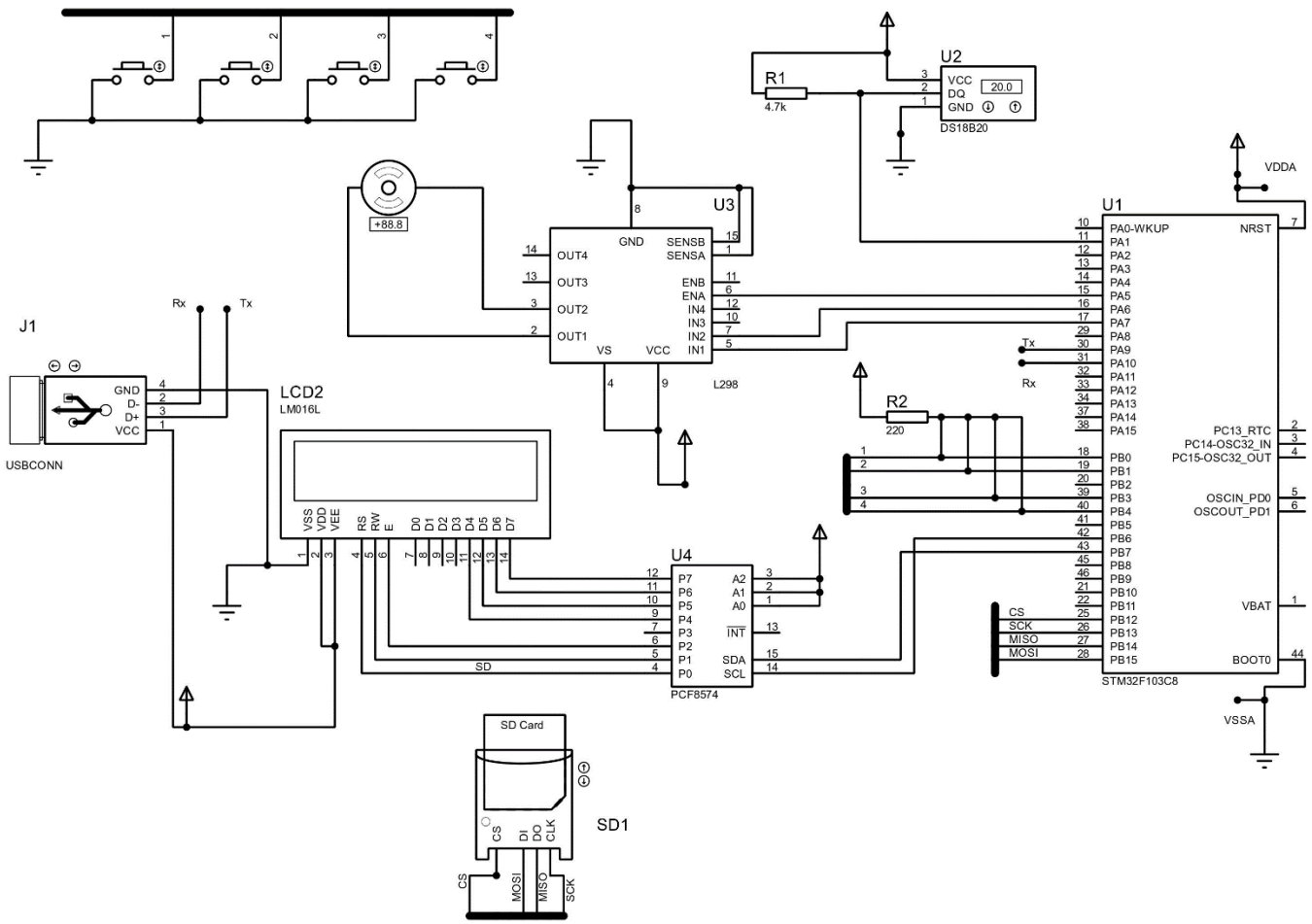
- Đọc nhiệt độ
- Giao tiếp với 4 nút nhấn để điều khiển động cơ
- Điều khiển tốc độ động cơ 1 chiều
- Hiển thị LCD sử dụng truyền nối tiếp I2C
- Truyền dữ liệu vào máy tính bằng giao tiếp UART
- Viết dữ liệu vào thẻ microSD với chu kỳ 10Hz (0.1s). NOTE: Do 0.1s khiến cho file sẽ bị ghi rất nhiều nên em sẽ để chu kỳ là 10s.

2. Giải quyết vấn đề:

Các ngoại vi được sử dụng:

- Ngắt ngoài – Nút nhấn
- PWM từ Timer 3 – Tốc độ động cơ
- Timer 2 – Delay microsecond
- I2C1 - LCD
- UART1 – Truyền dữ liệu máy tính
- SPI2 – Module microSD
- Middleware FATFS – Đọc/ghi file trong SD
- FreeRTOS – Lập lịch, đảm bảo tính thời gian thực
- IWDG – Tránh TH chương trình bị treo (Sau 3 giây bị treo, chương trình sẽ được reset)
- Đọc/ghi Flash – Tránh trường hợp mất điện

3. Sơ đồ ghép nối phần cứng (Sử dụng file pdf để nhìn rõ hơn)



4. Sơ đồ khối bài toán

4.1. FreeRTOS

- Sử dụng 7 task với mức ưu tiên như bảng dưới:

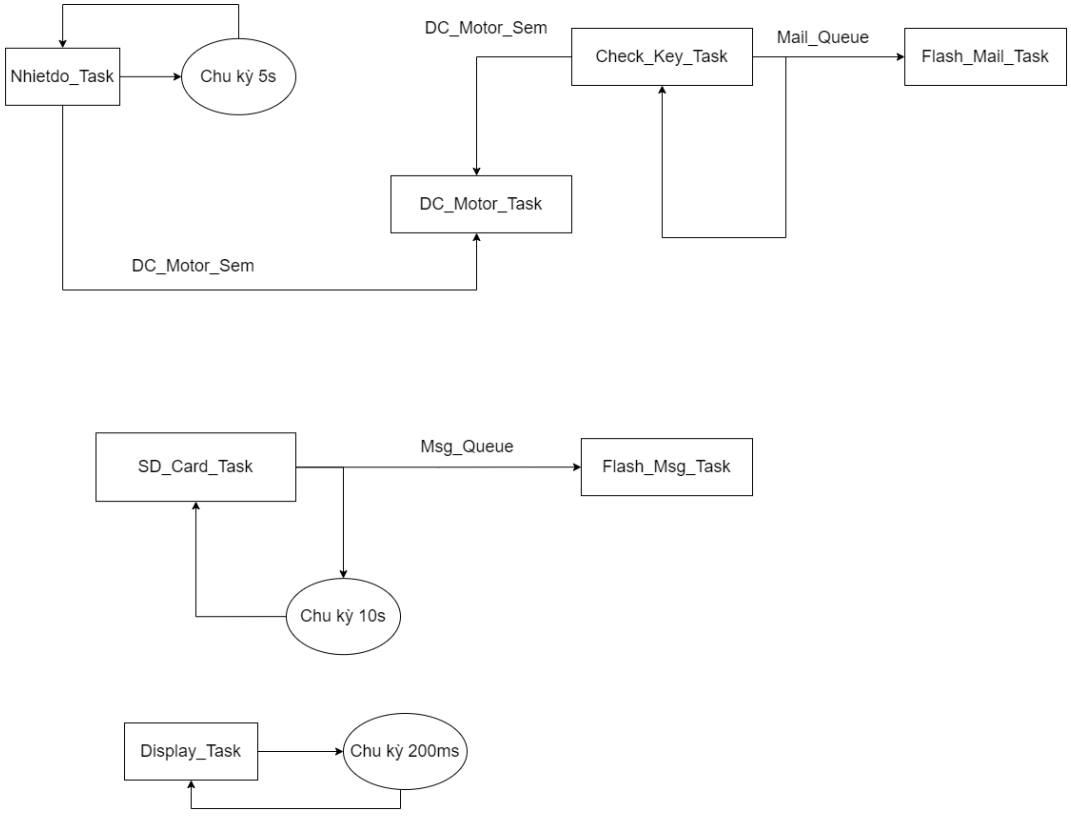
Các task	Mức ưu tiên
Check_Key_Task	Cao nhất
DC_Motor_Task	Cao nhất
Nhietdo_Task	Trên bình thường
SD_Card_Task	Bình thường
Flash_Mail_Task	Bình thường
Flash_Msg_Task	Bình thường
Display_Task	Thấp nhất

Semaphore
DC_Motor_Sem

Queue
Mail_Queue
Msg_Queue

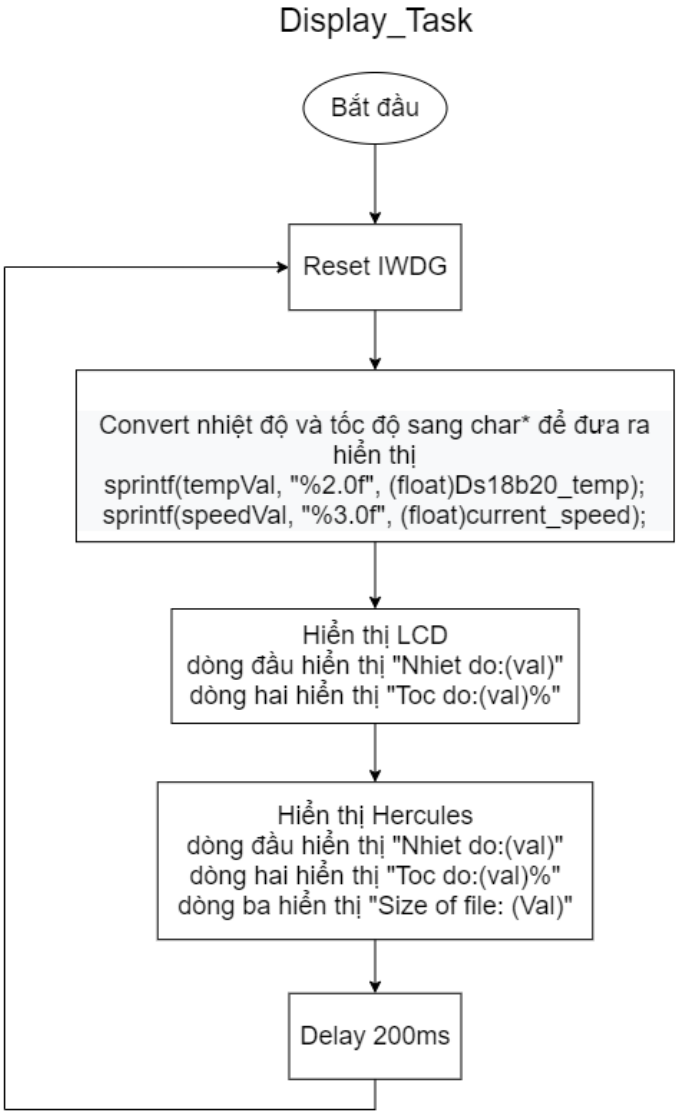
- Sử dụng Semaphore từ 2 task: Nhiệt độ được cập nhật hoặc Nút nhấn được ấn làm tín hiệu cho phép task tốc độ động cơ được hoạt động.
- Sử dụng Mail queue và Message queue để đảm bảo giá trị được lưu trữ chính xác nhất khi Task ghi dữ liệu vào flash hoạt động
- Các task còn lại sẽ có chu kỳ hoạt động riêng

LẬP LỊCH RTOS

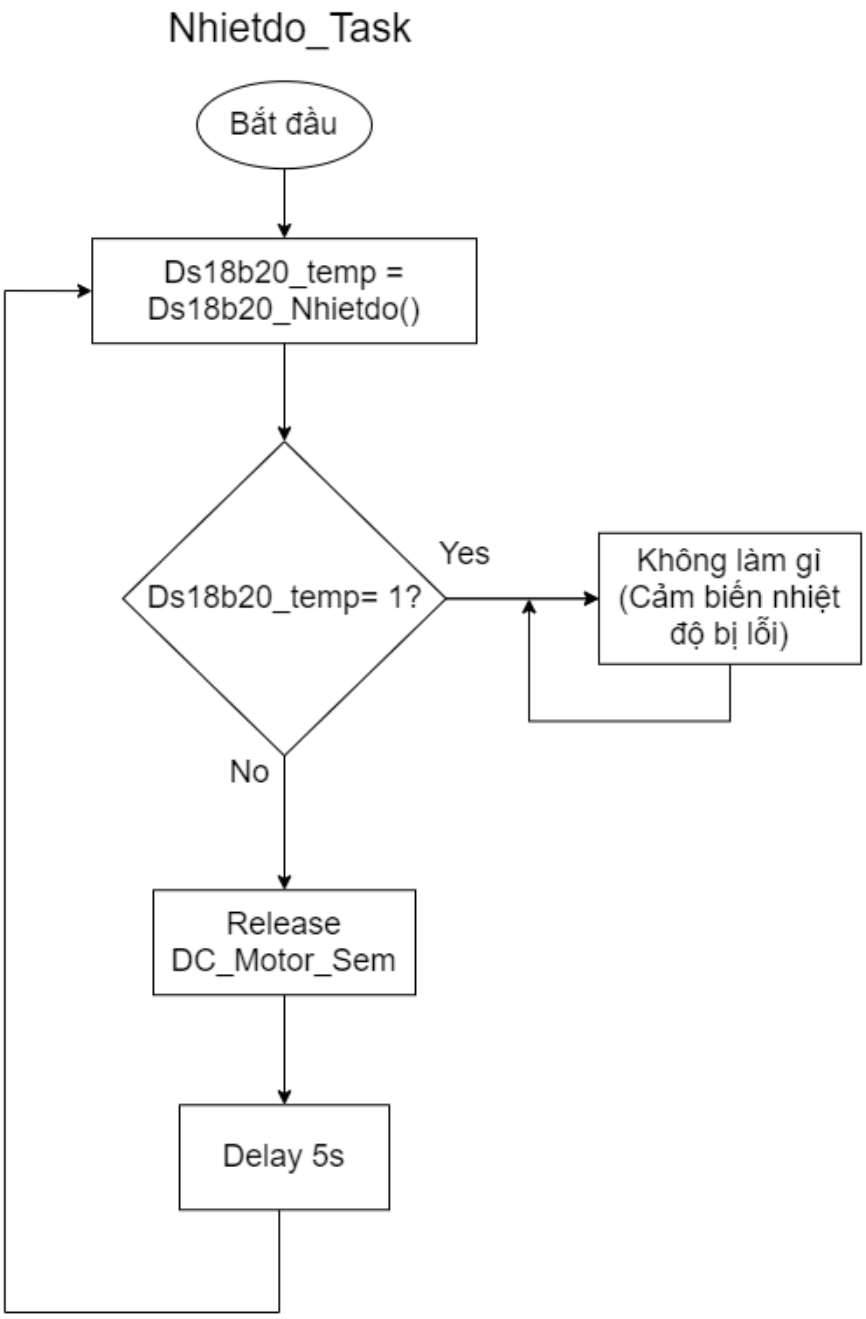


4.2. Chi tiết các task:

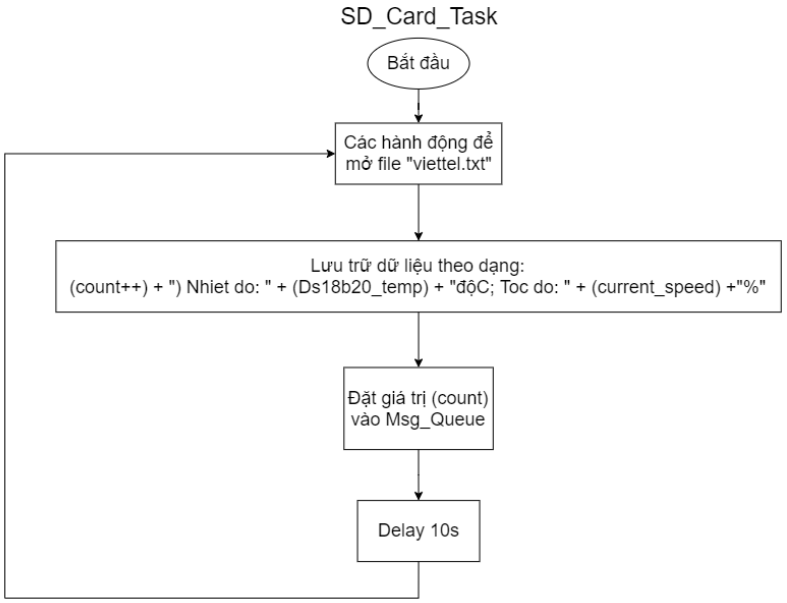
- Task Check_Key và Task Display có chu kỳ nhanh => Reload bộ đếm IWDG (Task Check_Key dài nên xem file FlowDiagram)
- Task Check_Key kiểm tra 4 nút nhấn, thay đổi giá trị tương ứng với 4 nút nhấn đó. Ngoài ra còn Release DC_Motor_Sem và Gửi structure vào Mail queue để lưu trữ vào bộ nhớ flash mỗi lần nút nhấn được ấn.
- Task Display ngoài việc hiển thị ra LCD và UART nhiệt độ và độ ẩm, ta còn truyền lên UART dung lượng của file được ghi trong thẻ SD, cho thấy file vẫn đang được cập nhật liên tục



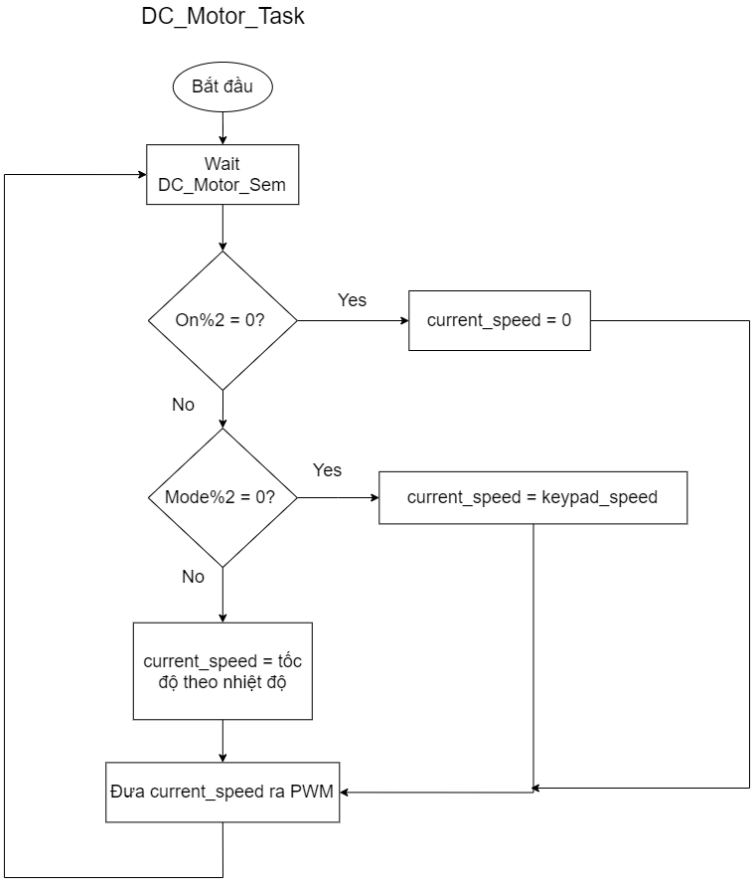
- Tại task nhiệt độ sẽ rơi vào vòng lặp vô hạn nếu đọc cảm biến bị lỗi
- Sau khi đọc nhiệt độ thành công (Hoàn thành task) sẽ Release DC_Motor_Sem
- Lưu ý task nhiệt độ , do sử dụng giao tiếp 1 Wire với cảm biến nên cần sử dụng hàm delay mức us => Sử dụng bộ đếm Timer 2 với tần số hoạt động 1MHz



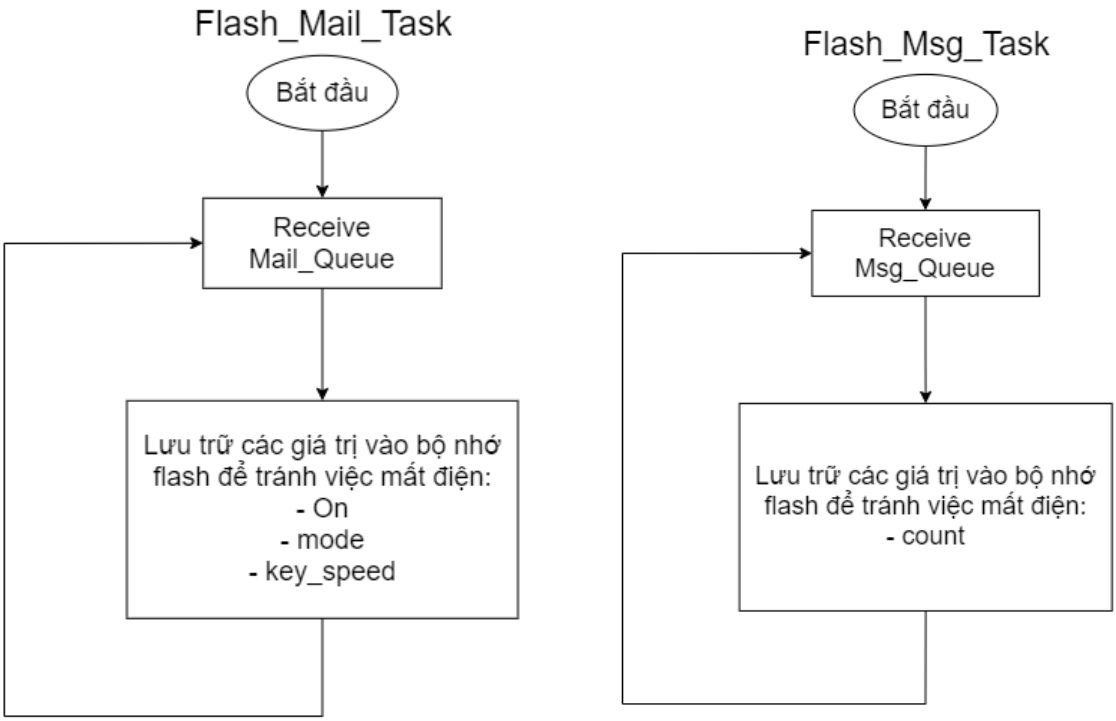
- Task 4, khi lưu trữ dữ liệu vào thẻ SD sẽ luôn có 1 biến đếm ở mỗi dòng. Biến đếm đó sẽ được truyền vào Message Queue để lưu vào flash mỗi lần tăng, đảm bảo việc lưu trữ được thực hiện liên tục.



- Task 3 hoạt động khi Mutex được release từ Task 1 hoặc ngắt
- Kiểm tra bật tắt – Biến on
- Kiểm tra chế độ - Biến mode

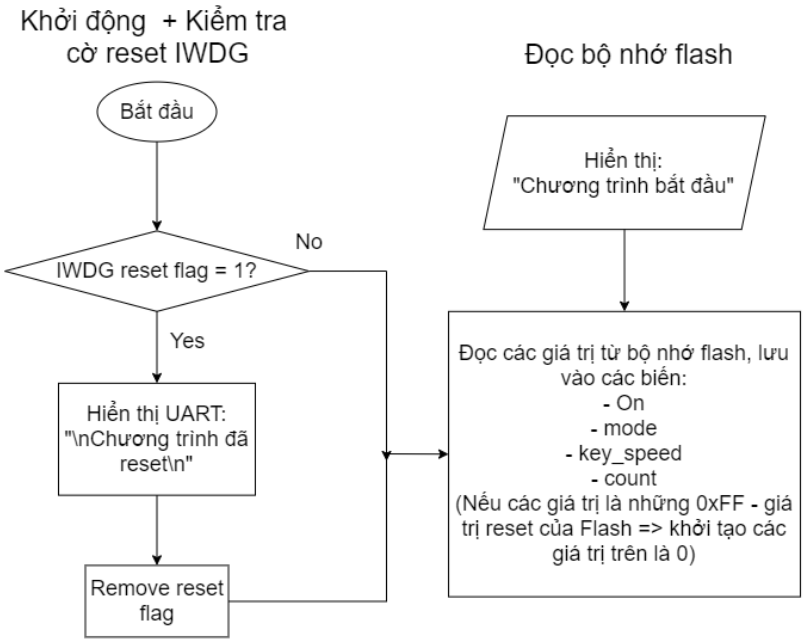


- 2 Task Flash được hoạt động khi nhận được dữ liệu từ Mail Queue và Msg Queue gửi về. Ta sẽ lưu trữ dữ liệu từ Msg Queue vào Page 126, từ Mail Queue vào Page 127 của bộ nhớ Flash



4.3. Các hàm quan trọng khác:

- Sau khi khởi động: Kiểm tra xem có phải reset do IWDG hay không; Đọc bộ nhớ flash lưu trữ vào các giá trị



5. Những khó khăn đã gặp phải

- Lần đầu tiếp xúc với FATFS và giao tiếp với microSD

=> Em dùng 5-6 ngày để tìm tài liệu, tập trung vào việc đọc tài liệu và đọc hiểu thư viện có sẵn

- Lần đầu tiếp cận các phương pháp quản lý task và dữ liệu: Semaphore, mutex, queue

=> Em đã tập trung vào thiết kế cách lập lịch, sử dụng các phương pháp sao cho hiệu quả

- Khi sử dụng nhiều task dẫn đến việc quá tải bộ nhớ => Sinh Hardfault bug

=> Em phải giảm thiểu các task, Semaphore hoặc Queue tránh sinh bug nhưng vẫn giữ được các chức năng cần thiết

- Em sử dụng ngắt ngoài giao tiếp với Nút nhấn, nhưng khi Release Semaphore thì bị Sinh Hardfault bug

=> Theo em nghĩ là do em sử dụng thư viện RTOS của CMSIS. Nhưng do không đủ thời gian tìm hiểu thư viện FreeRTOS, em đã phải sử dụng polling.