TRƯỜNG ĐẠI HỌC BÁCH KHOA - ĐẠI HỌC ĐÀ NẴNG
KHOA ĐIỆN TỬ - VIỄN THÔNG

BÁO CÁO NHÓM

Đề tài: HỆ ĐIỀU HÀNH THỜI GIAN THỰC RTOS CẢNH BÁO NGẬP LỤT

GV hướng dẫn: TS. Đào Duy Tuấn

Sinh viên thực hiện: Nhóm 5

Tôn Thất Hải

Lê Bảo

Nguyễn Thanh Thanh

Hoàng Ngọc Lộc

Trần Đặng Thành

Đà nẵng, ngày 18 tháng 6 năm 2024

Bảng phân công nhiệm vụ

Công việc	Tỉ lệ	Hải	Bảo	Thanh	Lộc	Thành
Tìm tài liệu	**10%**	2%	3%	2%	1%	2%
Thiết kế hệ thống	**20%**	8%	3%	4%	2%	3%
Dự báo lũ	**5%**	3%	0%	2%	0%	0%
Frontend Backend	**15%**	0%	0%	0%	15%	0%
Phần cứng	**30%**	1%	13%	1%	1%	14%
Các sơ đồ	**10%**	5%	0%	5%	0%	0%
Báo cáo	**10%**	1%	1%	6%	1%	1%
Tổng cộng	**100%**	20%	20%	20%	20%	20%

Nội dung thực hiện

- **Đặt vấn đề**
- **Giới thiệu**
- **Phương pháp**
- **Kiến trúc hệ thống**
- **Demo**
- **Hướng phát triển**
- **Kết luận**

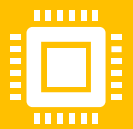
1. Đặt vấn đề

Thiệt hại do thiên tai trong tháng 10 năm 2022

2. Giới thiệu

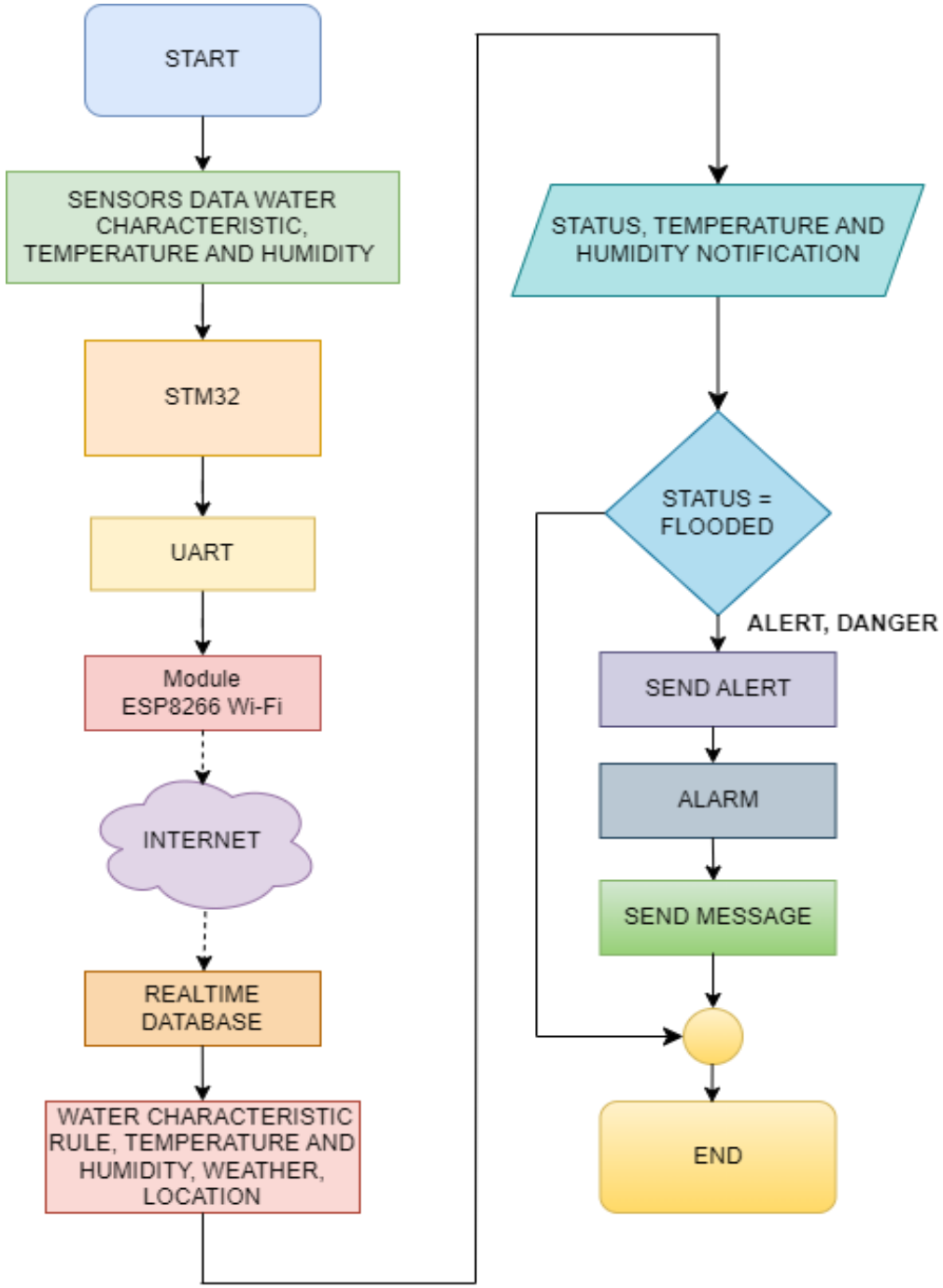
IoT(Internet of Things) đã trở thành một yếu tố quan trọng trong việc cải thiện và hỗ trợ các hệ thống ở nhiều ngành công nghiệp như: dự báo, y tế, sản xuất và nhiều lĩnh vực khác.

Hiện nay, thiên tai ảnh hưởng rất là lớn tới đời sống kinh tế-xã hội của người dân như là :mưa lớn, bão... và ảnh hướng lớn nhất là lũ lụt.

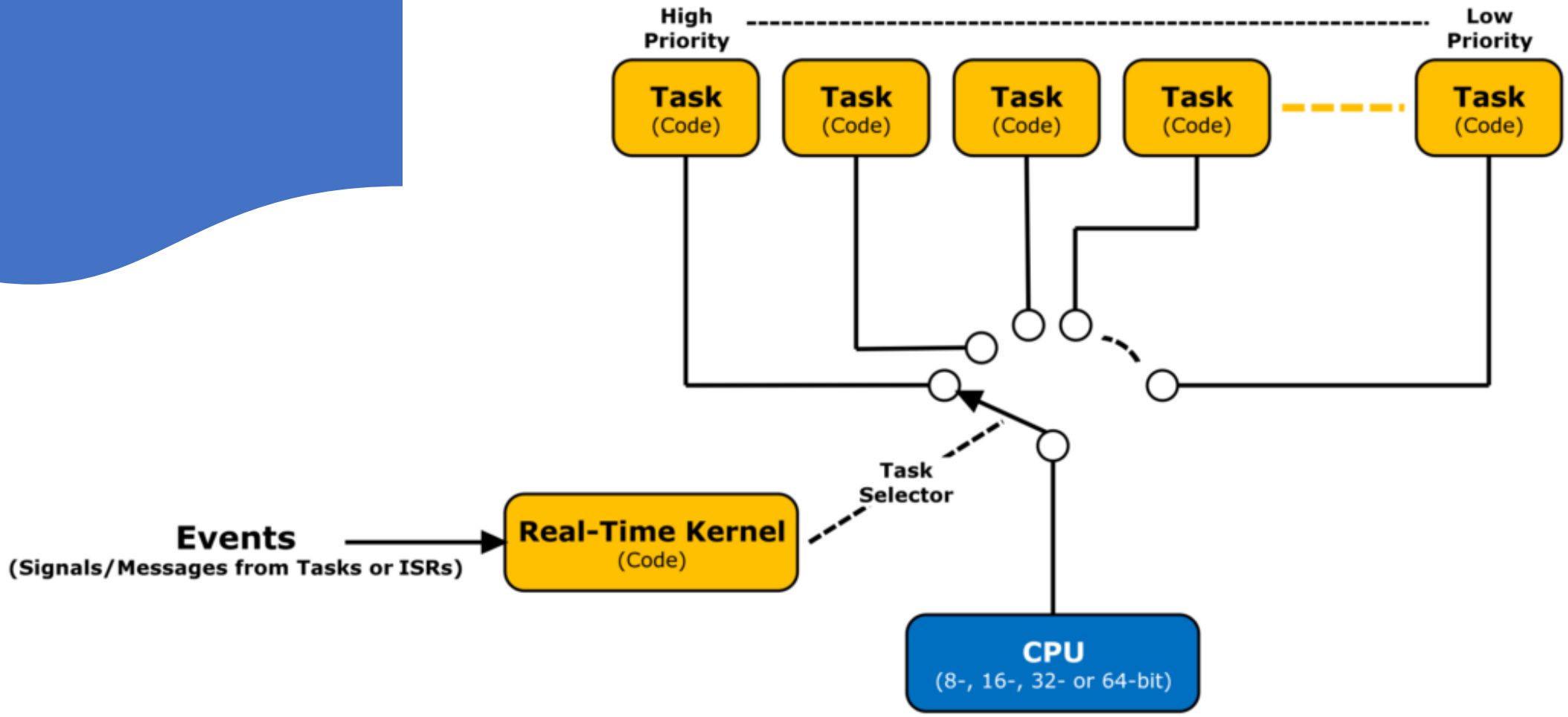


Với việc hệ thống RTOS có thể đáp ứng thời gian thực Realtime, quản lý tài nguyên hiệu quả trong việc sử dụng các tác vụ. RTOS trở thành lựa chọn lý tưởng cho hệ thống nhúng Realtime

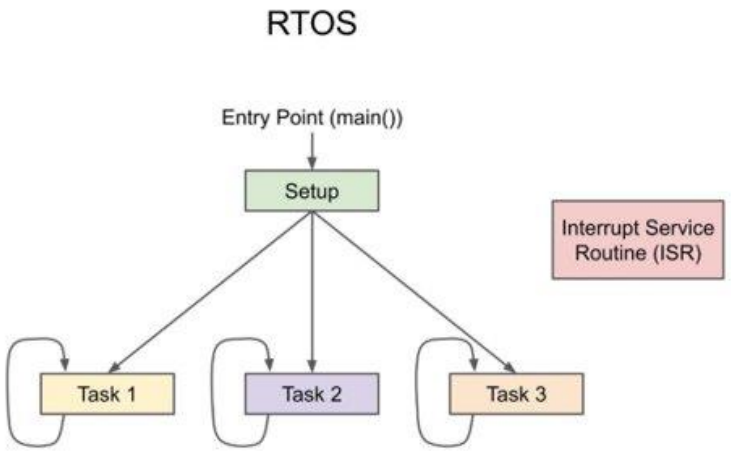
3. Phương pháp



RTOS


High Priority
Low Priority
Task (Code)
Task (Code)
Task (Code)
Task (Code)
Task (Code)
Task Selector
Events
(Signals/Messages from Tasks or ISRs)
Real-Time Kernel
(Code)
CPU
(8-, 16-, 32- or 64-bit)

Task



- - Task là một đoạn code được lập lịch bởi bộ scheduler để chạy và chương trình có thể có một hay nhiều task cùng chạy đồng thời.
- - Các task có thể trao đổi dữ liệu với nhau thông qua: Queue, biến Global và Static.
- - Đảm bảo Synchronization với nhau thông qua: Mutex, Semaphore, Notification.

Task Priority

- Mỗi một task đều được gán một giá trị từ 0->. Giá trị càng thấp thì Task có priority càng cao.

- Ở chế độ lập lịch preemptive thì task có priority cao hơn luôn được lập lịch để chạy trước task có priority thấp hơn. Điều này không đúng ở chế độ Cooperative.

- Task priority được lưu trữ trong Task Control Block của Task và được scheduler sử dụng để lập lịch.

Task STM32

```
TaskDHT11Handle = osThreadNew(StartTaskDHT11, NULL, &TaskDHT11_attributes);
TaskHCHandle = osThreadNew(StartTaskHC, NULL, &TaskHC_attributes);
TaskRainSensorHandle = osThreadNew(StartTaskRainSensor, NULL, &TaskRainSensor_attributes);
```

- TaskDHT11: Đọc dữ liệu từ cảm biến DHT11 và gửi qua UART.
- TaskHC: Đo khoảng cách bằng cảm biến siêu âm HC-SR04 và gửi qua UART.
- TaskRainSensor: Đọc dữ liệu từ cảm biến mưa và gửi qua UART.

```
/* Definitions for TaskDHT11 */
osThreadId_t TaskDHT11Handle;
const osThreadAttr_t TaskDHT11_attributes = {
  .name = "TaskDHT11",
  .stack_size = 128 * 4,
  .priority = (osPriority_t) osPriorityNormal,
};

/* Definitions for TaskHC */
osThreadId_t TaskHCHandle;
const osThreadAttr_t TaskHC_attributes = {
  .name = "TaskHC",
  .stack_size = 128 * 4,
  .priority = (osPriority_t) osPriorityHigh,
};

/* Definitions for TaskRainSensor */
osThreadId_t TaskRainSensorHandle;
const osThreadAttr_t TaskRainSensor_attributes = {
  .name = "TaskRainSensor",
  .stack_size = 128 * 4,
  .priority = (osPriority_t) osPriorityNormal1,
};
```

Task ESP32

- Task sendDataTask: Đọc dữ liệu từ UART, kiểm tra và phân loại dữ liệu nhận được từ các cảm biến (khoảng cách, mưa, nhiệt độ, độ ẩm).
- Khi tất cả các giá trị đều có, task sẽ kết hợp các giá trị này thành một chuỗi và gửi chúng tới server thông qua một yêu cầu HTTP POST.

```
xTaskCreate(
  sendDataTask,
  "SendDataTask",
  8192,
  NULL,
  1,
  NULL
);
```

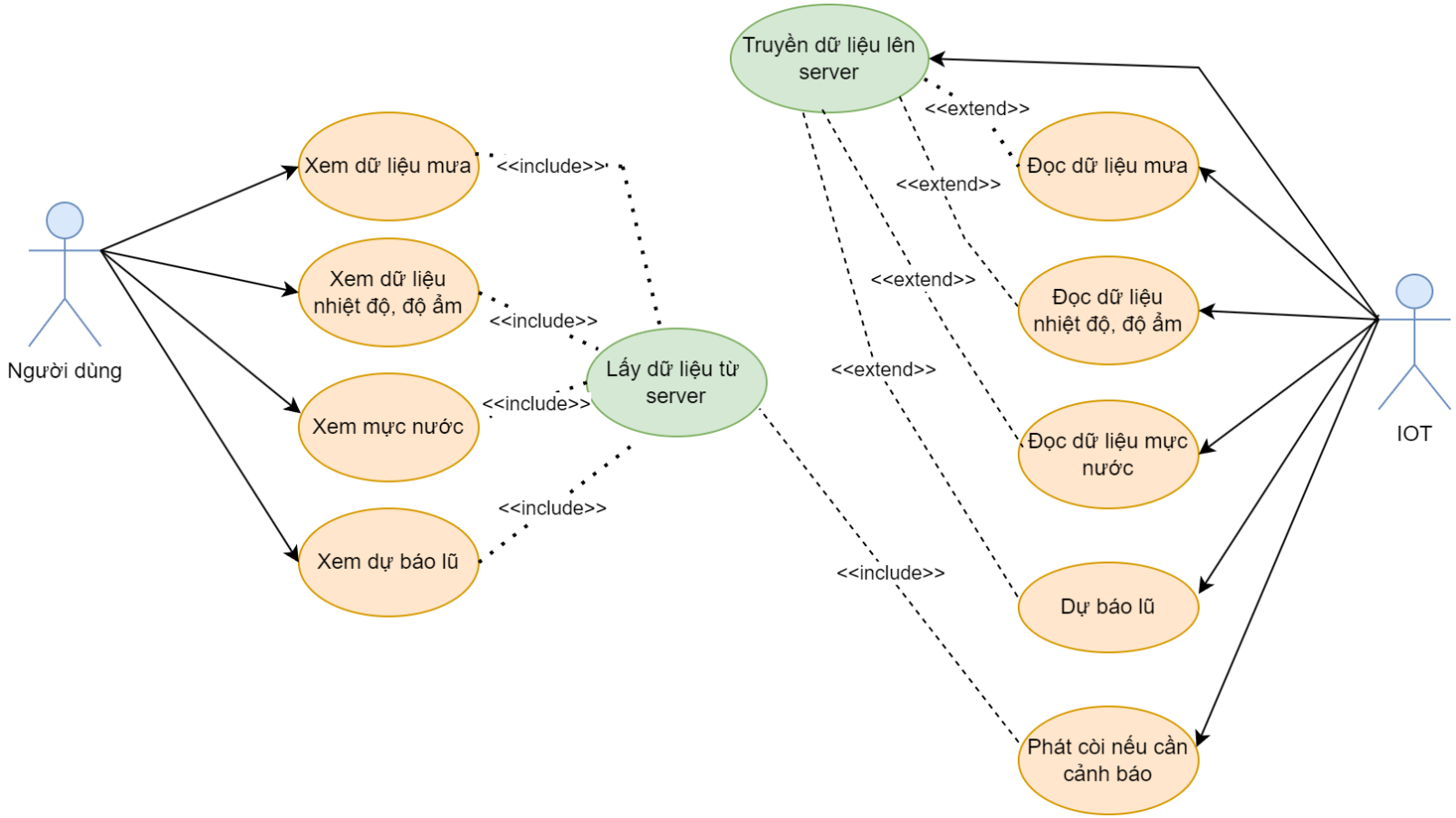
QUEUE

```
#define QUEUE_SIZE 5
#define RAIN_SENSOR_QUEUE_SIZE 5
#define HC_QUEUE_SIZE 5
#define DHT11_QUEUE_SIZE 5

osMessageQueueId_t DHT11QueueHandle;
osMessageQueueId_t HCQueueHandle;
osMessageQueueId_t RainSensorQueueHandle;
```

```
DHT11QueueHandle = osMessageQueueNew(DHT11_QUEUE_SIZE, sizeof(uint32_t[5]), NULL);
HCQueueHandle = osMessageQueueNew(HC_QUEUE_SIZE, sizeof(uint16_t), NULL);
RainSensorQueueHandle = osMessageQueueNew(RAIN_SENSOR_QUEUE_SIZE, sizeof(uint16_t), NULL);
```

Biểu đồ Uses case



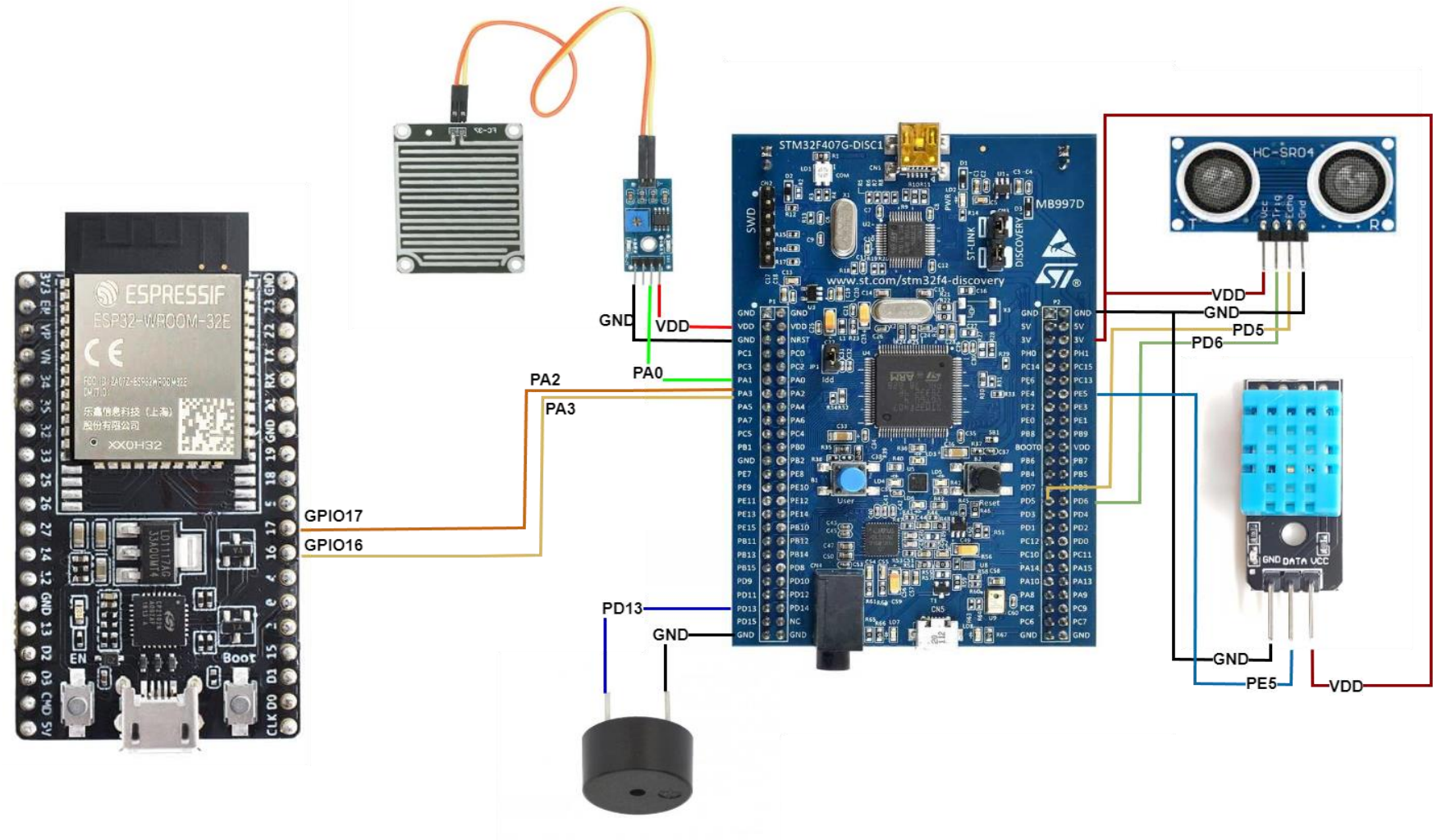
4. Kiến trúc hệ thống

- Hệ thống thu thập dữ liệu từ các thiết bị và cảm biến kết nối với nhau để giám sát thời gian thực các thông số như mức độ mưa, mực nước, nhiệt độ độ ẩm, thời tiết … thông qua các cảm biến như DHT11, cảm biến siêu âm HC-SR04, cảm biến mưa, API OpenWeahterMap.

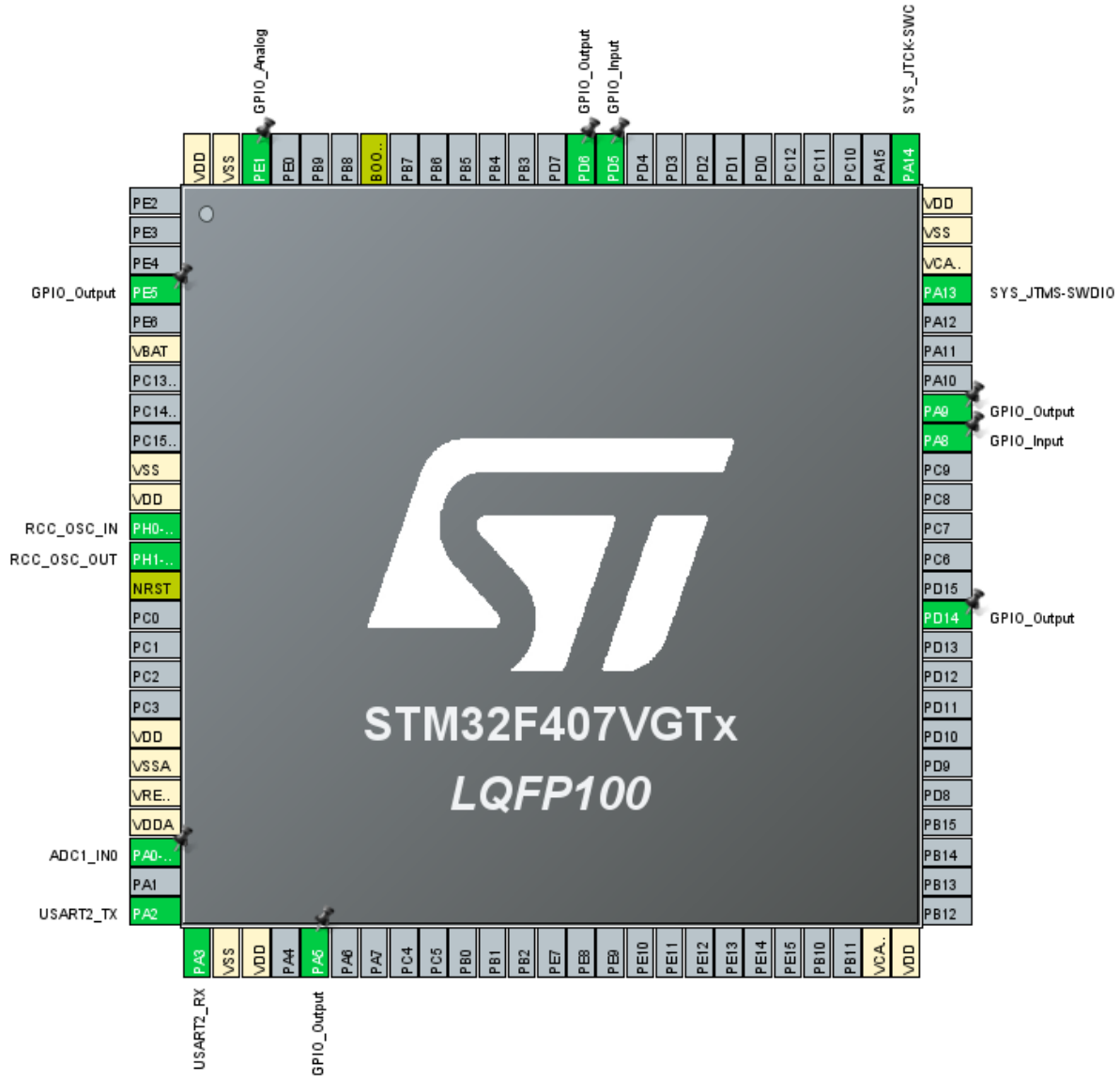
- Giám sát thời gian thực, tự động việc cảnh báo cho người dân đồng thời hệ thống cũng cung cấp dữ liệu thời tiết nhiệt độ, độ ẩm cho người dùng giúp cho người dùng có thể theo dõi tình hình lũ lụt.

A. Phát triển phần cứng

- STM32F4, ESP32
- Cảm biến siêu âm HC-SR04
- Cảm biến nhiệt độ, độ ẩm DHT11
- Cảm biến mưa
- Còi Buzzer

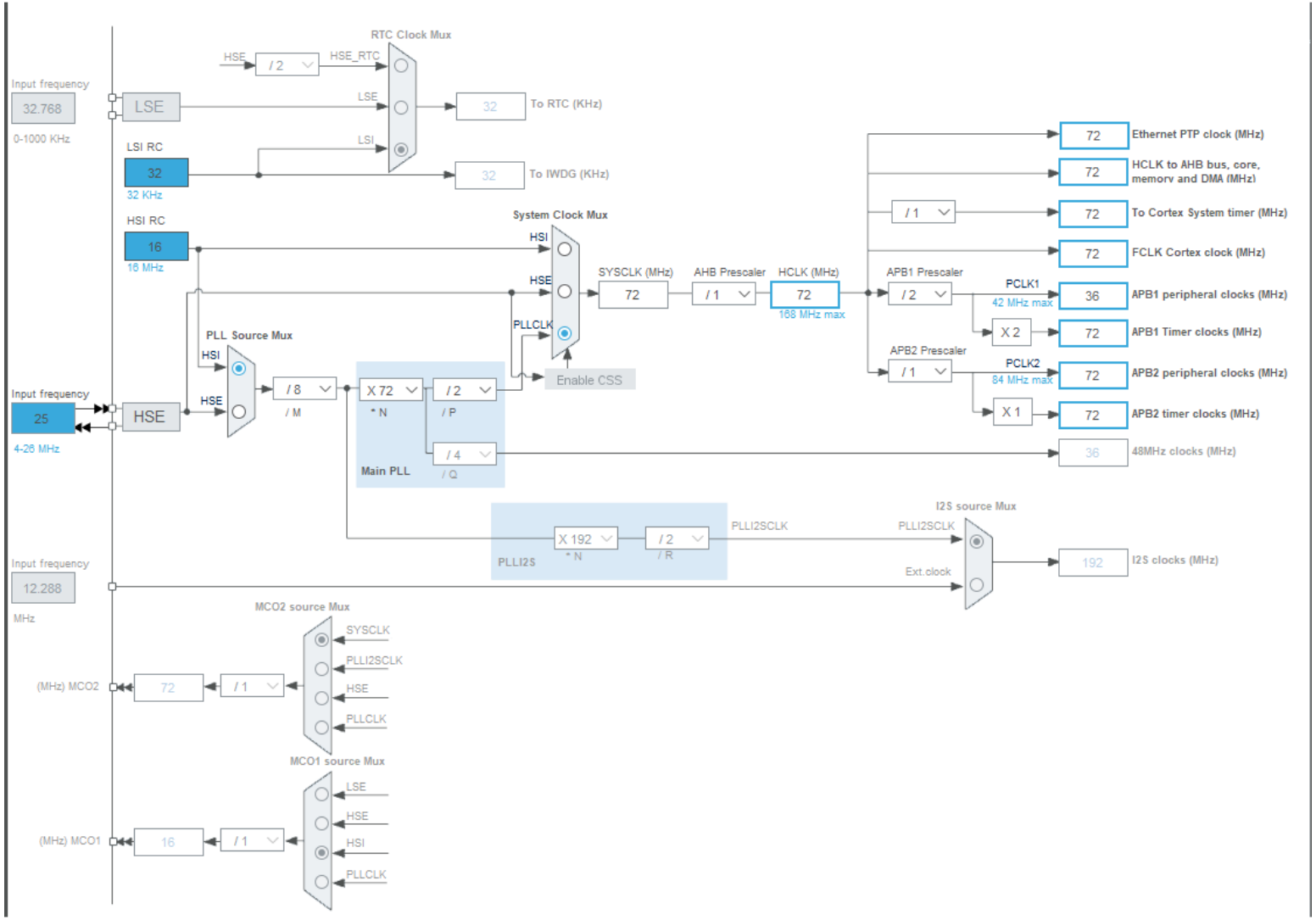


A. Phát triển phần cứng



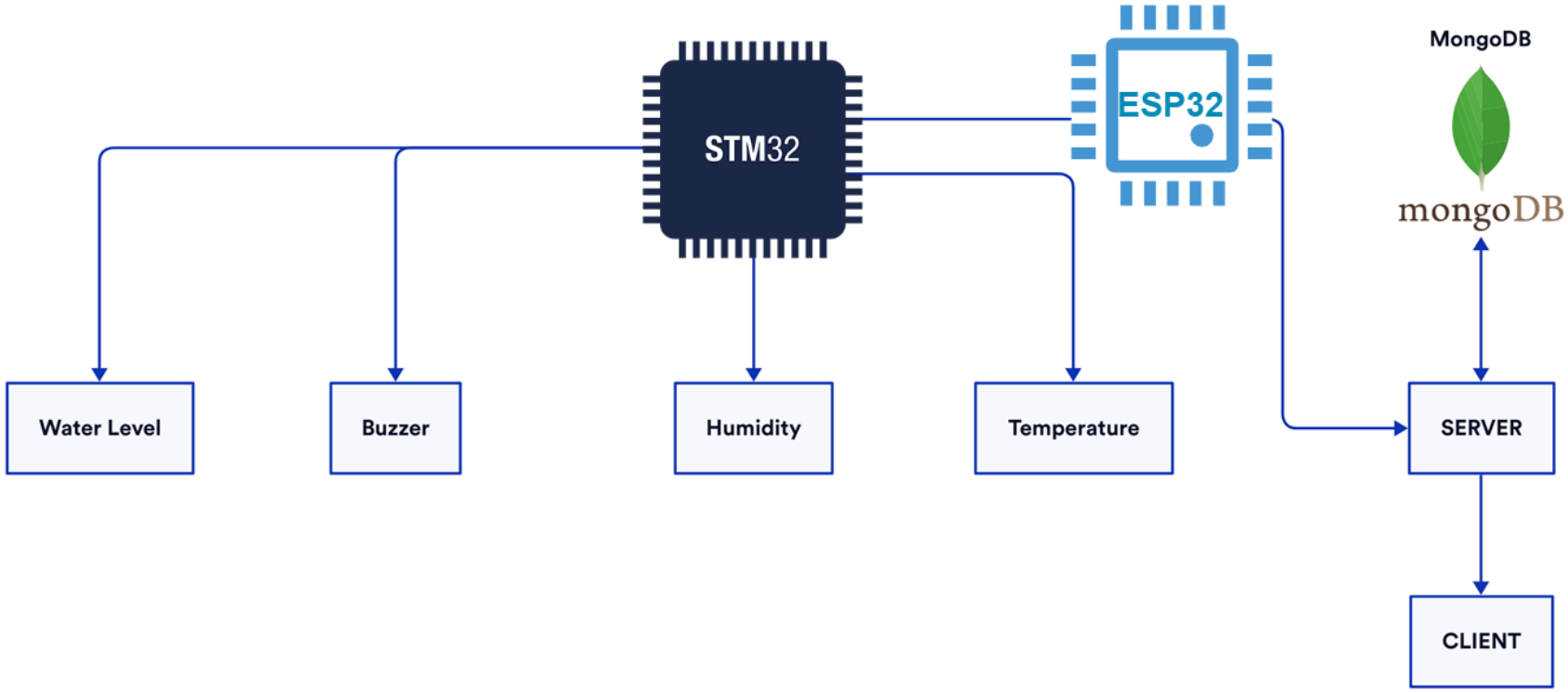
A. Phát triển phần cứng

Block Diagram



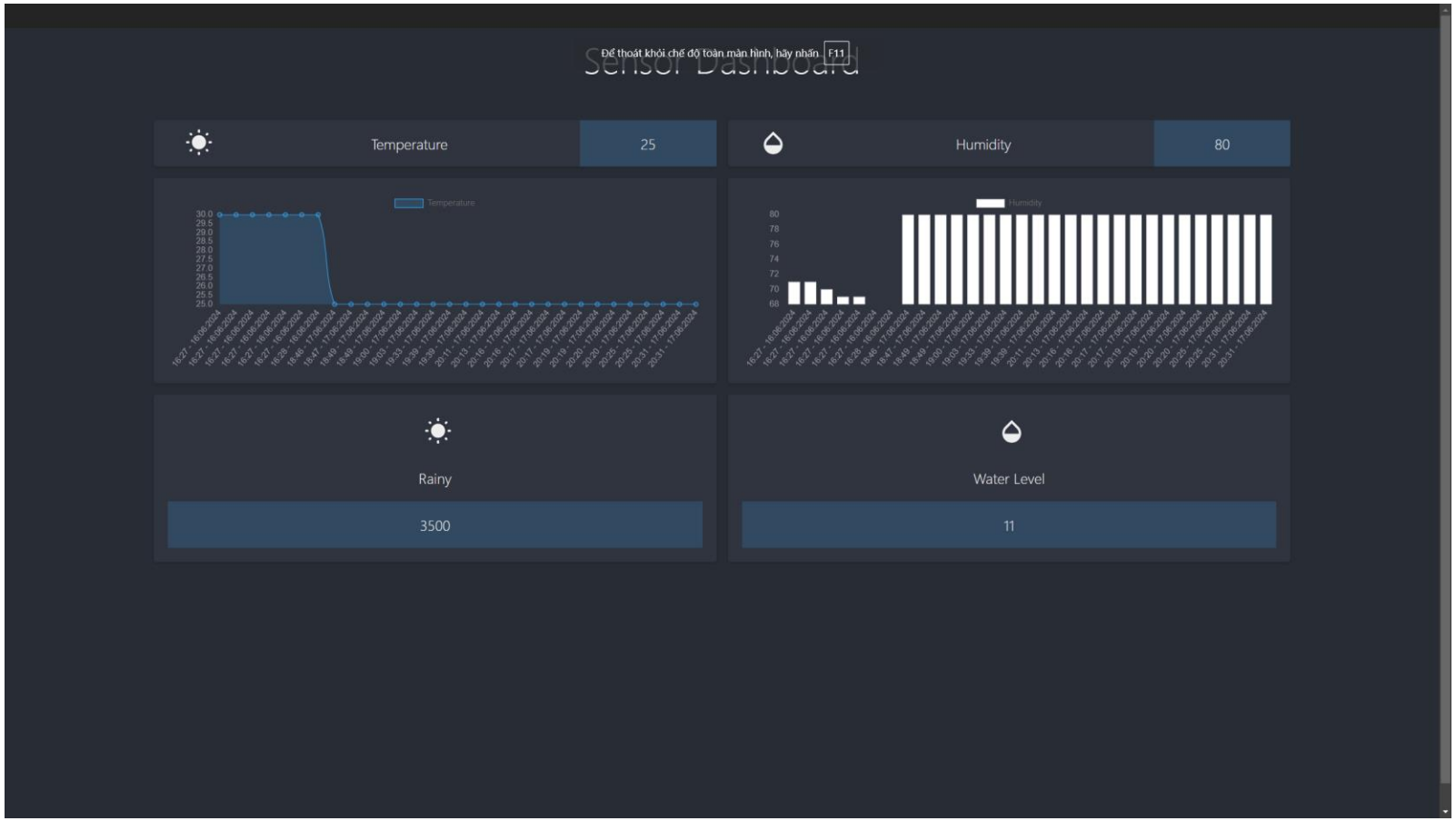
B. Phát triển phần mềm

- Data sẽ được truyền UART từ STM32 đến ESP32 để gửi dữ liệu lên server
- Dữ liệu gửi lên Server sẽ được lưu trữ ở MongoDataBase
- Giao diện Client sẽ được POST/GET với Server



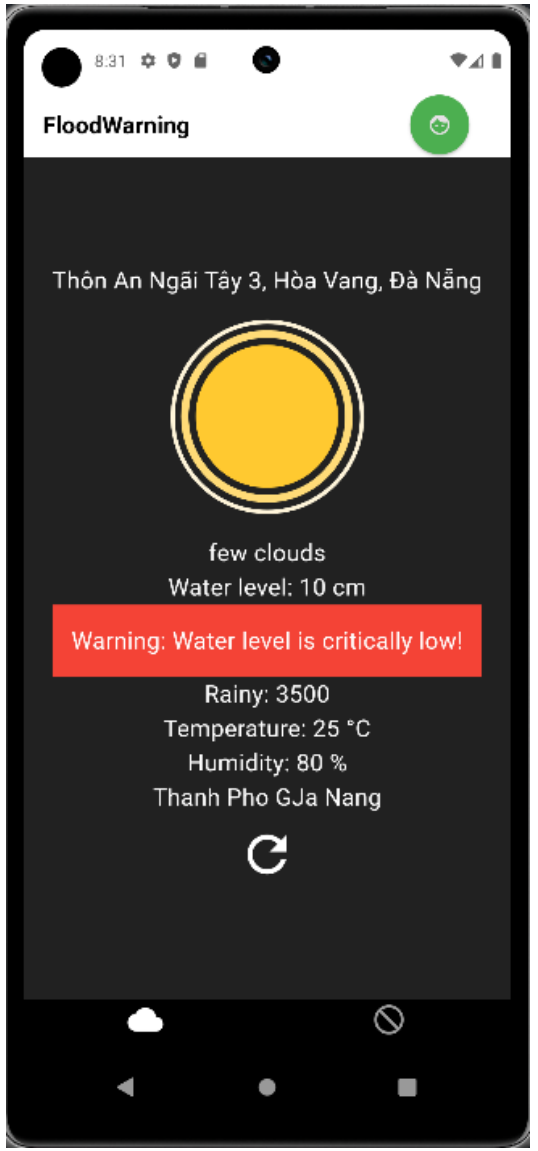
B. Phát triển phần mềm

- Client
  - Xây dựng giao diện Web để hiển thị các thông số đã thu thập
  - Công nghệ sử dụng: React JS, React-Chart-JS



B. Phát triển phần mềm

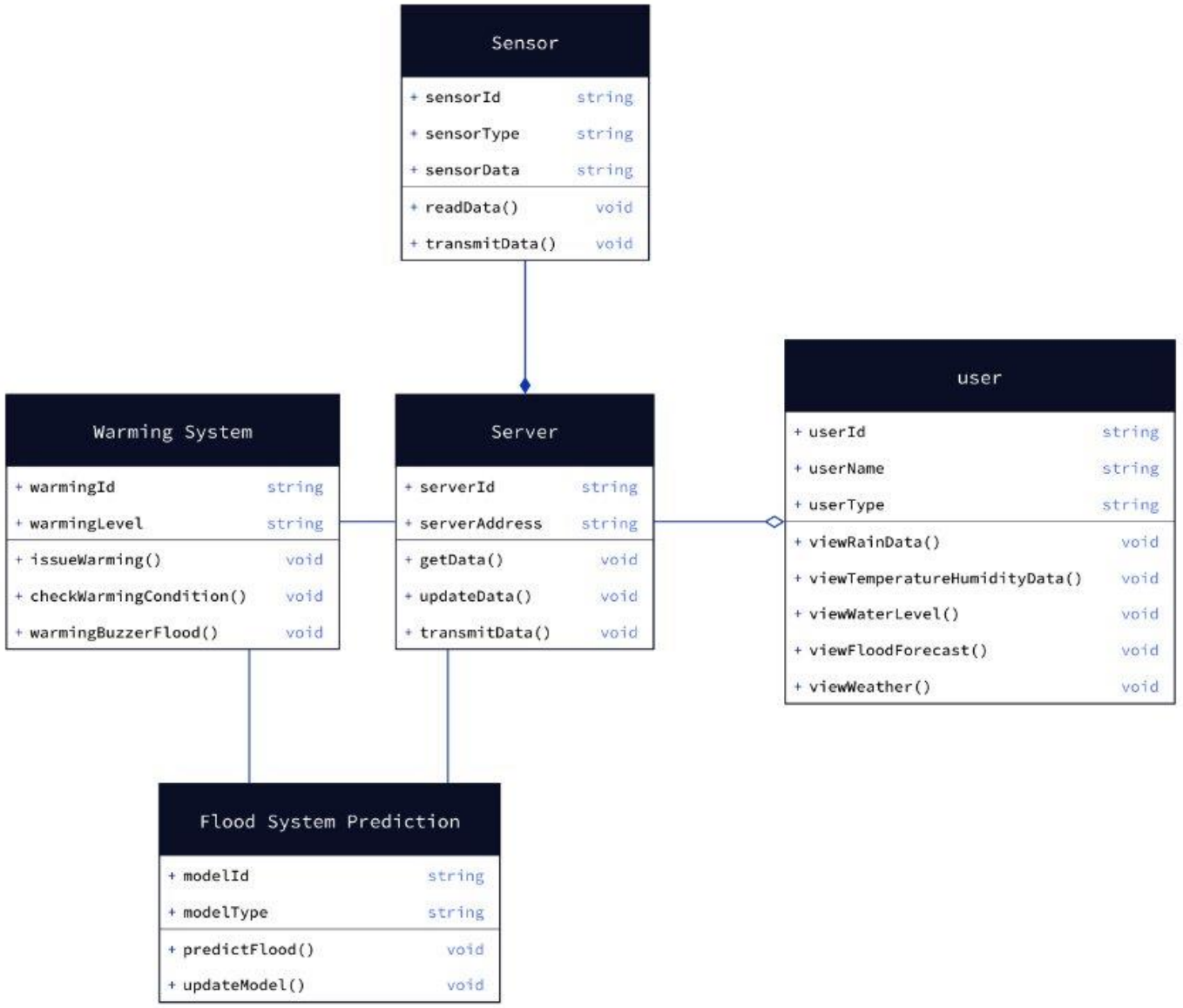
- Client
  - Xây dựng giao diện Web để hiển thị các thông số đã thu thập
  - Công nghệ sử dụng: Flutter/Dart



B. Phát triển phần mềm

- Server
  - Xây dựng các API để để tương tác với ESP 32 và Cơ sở dữ liệu
  - Công nghệ sử dụng: Express JS, Mongo DB

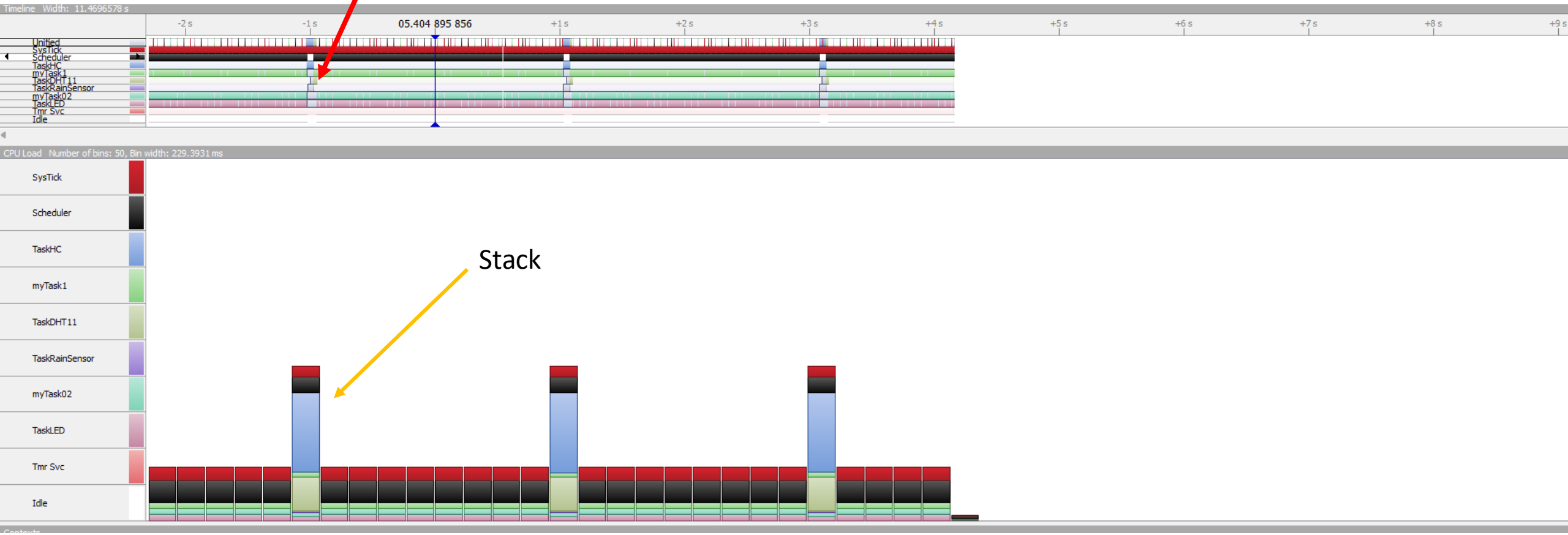
Sơ đồ Class Diagram



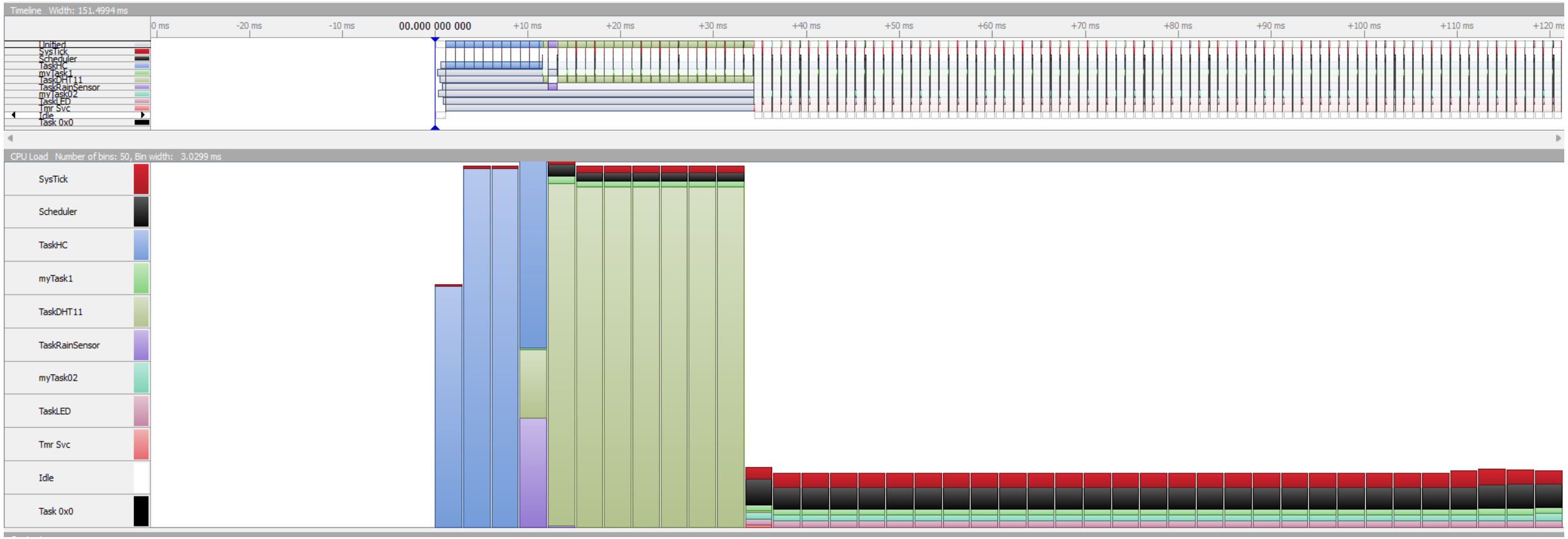
5. Demo

Segger

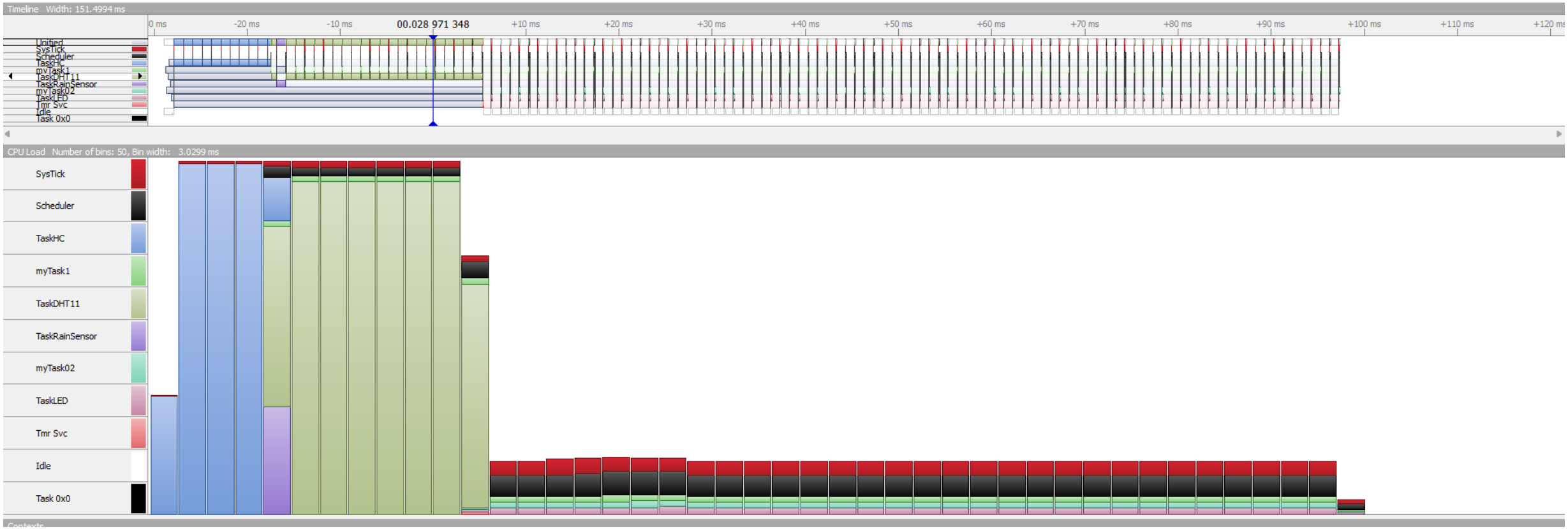
Priority:
TaskHC: High
TaskLED: High
TaskRainSenSor: Normal1
TaskDHT11: Normal



Segger


Timeline Width: 151.4994 ms
0 ms
-20 ms
-10 ms
00.000 000 000
+10 ms
+20 ms
+30 ms
+40 ms
+50 ms
+60 ms
+70 ms
+80 ms
+90 ms
+100 ms
+110 ms
+120 ms
Unified
SysTick
Scheduler
TaskHC
myTask1
TaskDHT11
TaskRainSensor
myTask02
TaskLED
Tmr Svc
Idle
Task 0x0
CPU Load Number of bins: 50, Bin width: 3.0299 ms
SysTick
Scheduler
TaskHC
myTask1
TaskDHT11
TaskRainSensor
myTask02
TaskLED
Tmr Svc
Idle
Task 0x0

Segger


Timeline Width: 151.4994 ms
0 ms
-20 ms
-10 ms
00.028 971 348
+10 ms
+20 ms
+30 ms
+40 ms
+50 ms
+60 ms
+70 ms
+80 ms
+90 ms
+100 ms
+110 ms
+120 ms
Unified
SysTick
Scheduler
TaskHC
myTask1
TaskDHT11
TaskRainSensor
myTask02
TaskLED
Tmr Svc
Idle
Task 0x0
CPU Load Number of bins: 50, Bin width: 3.0299 ms
SysTick
Scheduler
TaskHC
myTask1
TaskDHT11
TaskRainSensor
myTask02
TaskLED
Tmr Svc
Idle
Task 0x0

6.Hướng phát triển

- Sử dụng model AI để dự báo lũ kết hợp với nhiều thông tin hơn từ các cảm biến đo từ nhiều địa điểm
- Mở rộng phạm vi hệ thống, đo ở sườn núi, sông … để cảnh báo cho nhiều người dân hơn

7.Kết luận

Nhóm đã thực hiện được hệ thống thử nghiệm sử dụng FreeRTOS, tiền đề để có thể phát triển đề tài có thể áp dụng thực tế

Hệ thống cảnh báo lũ sớm sử dụng FreeRTOS là một giải pháp tiềm năng để giám sát mực nước và cảnh báo lũ lụt.

Tài liệu tham khảo

[1] Marzukhi, Syahaneim & Sidik, Mohd & Nasir, Haidawati & Zainol, Zuraini & Ismail, Mohd Nazri. (2018). Flood Detection and Warning System (FLoWS). 1-4. 10.1145/3164541.3164623.

[2] Keoduangsine, Saysoth & Robert, Robert & Gardner-Stephen, Paul. (2014). A Review of Flood Warning Systems in Developed and Developing Countries. International Journal of Future Computer and Communication. 3. 172-176. 10.7763/IJFCC.2014.V3.290.


THANKS FOR LISTENING