|  |  |
| --- | --- |
| Học trực tuyến CNTT, học lập trình từ cơ bản đến nâng cao |  |

**IOT302 – Tổng quan về IOT & Lập trình C nhúng cho vi điều khiển**

**THUYẾT MINH ASSIGNMENT 1**

**MÔ PHỎNG THIẾT BỊ CÔNG TẮC CẢM ỨNG CỦA LUMI SMART HOME**

|  |  |
| --- | --- |
| Tên học viên:  Mã học viên:  Ngày báo cáo: | **Nguyễn Hoàng Dũng**  **FX19603**  **29/11/2023** |

**HÀ NỘI, 11/2023**

**MỤC LỤC**

[1. Danh sách thư viện 4](#_Toc156468422)

[2. Danh sách các biến toàn cục và các macro 4](#_Toc156468423)

[3. Danh sách các hàm 6](#_Toc156468424)

[4. Hàm main ( ) 6](#_Toc156468425)

[4.1. Mô tả 6](#_Toc156468426)

[4.2. Tham số truyền vào 7](#_Toc156468427)

[4.3. Giá trị trả về 7](#_Toc156468428)

[5. Hàm AppInitCommon ( ) 7](#_Toc156468429)

[5.1. Mô tả 7](#_Toc156468430)

[5.2. Tham số truyền vào 7](#_Toc156468431)

[5.3. Giá trị trả về 8](#_Toc156468432)

[6. Hàm SetStateApp ( ) 8](#_Toc156468433)

[6.1. Mô tả 8](#_Toc156468434)

[6.2. Tham số truyền vào 8](#_Toc156468435)

[6.3. Giá trị trả về 8](#_Toc156468436)

[7. Hàm GetStateApp ( ) 8](#_Toc156468437)

[7.1. Mô tả 8](#_Toc156468438)

[7.2. Tham số truyền vào 8](#_Toc156468439)

[7.3. Giá trị trả về 8](#_Toc156468440)

[8. Hàm AppStateManager ( ) 8](#_Toc156468441)

[8.1. Mô tả 8](#_Toc156468442)

[8.2. Tham số truyền vào 8](#_Toc156468443)

[8.3. Giá trị trả về 8](#_Toc156468444)

[9. Hàm LoadConfiguration ( ) 8](#_Toc156468445)

[9.1. Mô tả 8](#_Toc156468446)

[9.2. Tham số truyền vào 8](#_Toc156468447)

[9.3. Giá trị trả về 8](#_Toc156468448)

[10. Hàm DeviceStateMachine ( ) 9](#_Toc156468449)

[10.1. Mô tả 9](#_Toc156468450)

[10.2. Tham số truyền vào 9](#_Toc156468451)

[10.3. Giá trị trả về 9](#_Toc156468452)

[11. Hàm Clamp ( ) 9](#_Toc156468453)

[11.1. Mô tả 9](#_Toc156468454)

[11.2. Tham số truyền vào 9](#_Toc156468455)

[11.3. Giá trị trả về 9](#_Toc156468456)

[12. Hàm Increase\_LedLevel ( ) 10](#_Toc156468457)

[12.1. Mô tả 10](#_Toc156468458)

[12.2. Tham số truyền vào 10](#_Toc156468459)

[12.3. Giá trị trả về 10](#_Toc156468460)

[13. Hàm Decrease\_LedLevel ( ) 10](#_Toc156468461)

[13.1. Mô tả 10](#_Toc156468462)

[13.2. Tham số truyền vào 10](#_Toc156468463)

[13.3. Giá trị trả về 10](#_Toc156468464)

[14. Hàm MultiSensorScan ( ) 10](#_Toc156468465)

[14.1. Mô tả 10](#_Toc156468466)

[14.2. Tham số truyền vào 10](#_Toc156468467)

[14.3. Giá trị trả về 10](#_Toc156468468)

[15. Hàm Task\_multiSensorScan ( ) 10](#_Toc156468469)

[15.1. Mô tả 10](#_Toc156468470)

[15.2. Tham số truyền vào 10](#_Toc156468471)

[15.3. Giá trị trả về 10](#_Toc156468472)

[16. Luồng xử lý code. 10](#_Toc156468473)

[16.1. Các bước thực xử lý 10](#_Toc156468474)

* [Câu 1: Hiển thị dòng text “IOT Programming by Lumi Smarthome” lên màn hình LCD của Kit mở rộng khi bắt đầu cắm nguồn cho thiết bị. 10](#_Toc156468475)
* [Câu 2: Nhấn nút B3 năm lần khi đó tất cả các Led trên Kit mở rộng sẽ nháy năm lần màu GREEN và hiển thị thông tin sau thiết bị lên màn hình LCD: 11](#_Toc156468476)
* [Câu 3: Nhấn các nút B1, B2, B4, B5 một lần để điều khiển bật/tắt màu tương ứng RED, GREEN, WHITE, BLUE của tất cả các led RGB trên Kit mở rộng và còi sẽ kêu một bíp một lần 13](#_Toc156468477)
* [Câu 4: Nhấn giữ B1/B5 để điều khiển tăng/giảm cường độ sáng của led RGB: 13](#_Toc156468478)
* [Câu 5: Cập nhật thông số nhiệt độ, độ ẩm và cường độ ánh sáng của các cảm biến lên màn hình LCD 14](#_Toc156468479)

[16.2. Lưu đồ thuật toán 16](#_Toc156468480)

**NỘI DUNG THUYẾT MINH**

# Danh sách thư viện

|  |  |  |
| --- | --- | --- |
| **STT** | **Thư viện** | **Mô tả** |
| 1 | stdio.h | Thư viện có các hàm nhập xuất:  + sprintf: Gửi output đã được định dạng tới một chuỗi |
| 2 | stdint.h | Thư viện lưu trữ và làm việc với các dữ liệu ở cấp độ bit  + uint8\_t: 8 bit  +uint16\_t: 16 bit  +uint32\_t: 32 bit |
| 3 | string.h | Thư viện cung cấp hàm để xử lý chuỗi |
| 4 | system\_stm32f4xx.h | Thư viện chứa thông tin hệ thống |
| 5 | timer.h | Thư viện cho phép quản lý các tác vụ theo thời gian |
| 6 | eventman.h | Thư viện chứa thông tin hệ thống |
| 7 | led.h | Thư viện cung cấp hàm để thao tác với LED |
| 8 | melody.h | Thư viện cung cấp hàm để bật còi |
| 9 | lightsensor.h | Thư viện cung cấp hàm để sử dụng cảm biến ánh sáng |
| 10 | temhumsensor.h | Thư viện cung cấp hàm để sử dụng cảm biến nhiệt độ và độ ẩm |
| 11 | eventbutton.h | Thư viện cung cấp hàm để sử dụng các sự kiện của nút nhấn |
| 12 | button.h | Thư viện cung cấp hàm để sử dụng nút nhấn |
| 13 | Ucglib.h | Thư viện cung cấp hàm để cấu hình các chân giao tiếp SPI của STM32 |

# Danh sách các biến toàn cục và các macro

|  |  |  |
| --- | --- | --- |
| **STT** | **Biến toàn cục và macro** | **Mô tả** |
|  | CYCLE\_LED\_CHANGE | Macro thể hiện chu kỳ thay đổi độ sáng LED |
|  | PERIOD\_SCAN\_MULTISENSOR | Macro thể hiện chu kỳ quét của cảm biến |
|  | event\_api\_t | Kiểu dữ liệu enum cho các sự kiện của chương trình |
|  | \*event\_api\_p | Con trỏ kiểu dữ liệu enum trỏ tới các sự kiện của chương trình |
|  | state\_app\_t | Kiểu dữ liệu enum cho các trạng thái của chương trình |
|  | ucg | Biến để điều khiển màn hình LCD |
|  | eCurrentState | Biến cho biết trạng thái hiện tại của chương trình |
|  | ledRed | Biến thể hiện trạng thái đèn LED, khi nhấn nút B1 |
|  | ledGreen | Biến thể hiện trạng thái đèn LED, khi nhấn nút B2 |
|  | ledWhite | Biến thể hiện trạng thái đèn LED, khi nhấn nút B4 |
|  | ledBlue | Biến thể hiện trạng thái đèn LED, khi nhấn nút B5 |
|  | ledChange | Biến dùng để thay đổi trạng thái LED khi nhấn giữ nút B1/B5 |
|  | idTimerStartIncrease | Biến lưu giá trị định danh ID của hàm TimerStart khi nhấn giữ nút B1 |
|  | idTimerStartDecrease | Biến lưu giá trị định danh ID của hàm TimerStart khi nhấn giữ nút B5 |
|  | idTimerDisplayLCD | Biến lưu giá trị định danh ID của hàm TimerStart khi gọi hàm MultiSensorScan |
|  | idTimerSensorUpdate | Biến lưu giá trị định danh ID của hàm TimerStart khi gọi hàm Task\_multiSensorScan |
|  | temperature | Biến lưu giá trị nhiệt độ được cập nhật từ cảm biến |
|  | humidity | Biến lưu giá trị độ ẩm được cập nhật từ cảm biến |
|  | light | Biến lưu giá trị cường độ ánh sáng được cập nhật từ cảm biến |
|  | strTemp[30] | Mảng lưu giá trị nhiệt độ |
|  | strHumi[30] | Mảng lưu giá trị độ ẩm |
|  | strLight[30] | Mảng lưu giá trị cường độ ánh sáng |

# Danh sách các hàm

|  |  |  |
| --- | --- | --- |
| **STT** | **Danh sách hàm** | **Mô tả** |
|  | int main () | Hàm chính của chương trình |
|  | void AppInitCommon (); | Hàm khởi tạo các tài nguyên cần sử dụng |
|  | static void SetStateApp (state\_app\_t state); | Hàm SET trạng thái của chương trình |
|  | static state\_app\_t GetStateApp (); | Hàm Return trạng thái của chương trình |
|  | static void AppStateManager (uint8\_t event); | Hàm xử lý các sự kiện của chương trình chính |
|  | void LoadConfiguration (); | Hàm hiển thị thông tin khi bắt đầu cắm nguồn cho thiết bị |
|  | void DeviceStateMachine (uint8\_t event); | Hàm điều khiển LED, Buzzer và hiển thị thông tin sau khi nhấn các nút trên bộ KIT |
|  | int Clamp (int value, int min , int max); | Hàm tạo khoảng giá trị |
|  | void Increase\_LedLevel (); | Hàm tăng độ sáng đèn LED |
|  | void Decrease\_LedLevel (); | Hàm giảm độ sáng đèn LED |
|  | void MultiSensorScan (); | Hàm cập nhật giá trị nhiệt độ, độ ẩm, cường độ ánh sáng |
|  | void Task\_multiSensorScan (); | Hàm lấy thông tin nhiệt độ, độ ẩm, cường độ ánh sáng từ cảm biến và hiển thị lên LCD |

# Hàm main ( )

## 4.1. Mô tả

Hàm main ( ) thực hiện việc gọi các hàm sau:

* AppInitCommon()
* SetStateApp(): Khởi tạo chương trình chính về trạng cũ của thiết bị trước khi tắt nguồn
* EventSchedulerAdd(): Thêm các sự kiện khởi tạo vào Queue
* Trong hàm while(1) gọi các hàm:
* processTimerScheduler(): Xử lý các sự kiện theo thời gian đã cài đặt trước
* processEventScheduler(): Xử lý các sự kiện khi có trạng thái nhấn nút

## 4.2. Tham số truyền vào

Không có

## 4.3. Giá trị trả về

Không có

# Hàm AppInitCommon ( )

## 5.1. Mô tả

Hàm AppInitCommon ( ) thực hiện việc khởi tạo các tài nguyên cần sử dụng:

* SystemCoreClockUpdate(): Cấu hình clock của hệ thống là 84 Mhz
* TimerInit(): Khởi tạo timer system tick để xử lý các sự kiện thời gian
* EventSchedulerInit(AppStateManager): Khởi tạo bộ đệm lưu trữ danh sách sự kiện của chương trình
* EventButton\_Init(): Cấu hình chân GPIO của các nút nhấn trên mạch
* BuzzerControl\_Init(): Cấu hình chân GPIO của còi trên mạch
* LedControl\_Init(): Cấu hình chân GPIO của các led RGB trên mạch
* LightSensor\_Init(ADC\_READ\_MODE\_DMA): Cấu hình ngoại vi ADC hoạt động ở chế độ DMA để đọc dữ liệu của cảm biến ánh sáng trên mạch.
* TemHumSensor\_Init(): Cấu hình ngoại vi I2C để giao tiếp với cảm biến nhiệt độ - độ ẩm
* Ucglib4WireSWSPI\_begin(&ucg, UCG\_FONT\_MODE\_SOLID): Cấu hình các chân giao tiếp SPI của STM32 với màn hình LCD
* ucg\_ClearScreen(&ucg): Xóa màn hình hiển thị LCD
* ucg\_SetFont(&ucg, ucg\_font\_ncenR10\_hf): Cài đặt font chữ kiểu ncenR10\_hr để hiển thị
* ucg\_SetColor(&ucg, 0, 255, 255, 255): SET màu trắng cho text hiển thị
* ucg\_SetColor(&ucg, 1, 0, 0, 0): SET màu đen cho nền hiển thị (backgroup)
* ucg\_SetRotate180(&ucg): Xoay màn hình 1 góc 180 độ

## 5.2. Tham số truyền vào

Không có

## 5.3. Giá trị trả về

Không có

# Hàm SetStateApp ( )

## 6.1. Mô tả

Hàm SetStateApp ( ) thực hiện việc đặt trạng thái của chương trình khi bật thiết bị

## 6.2. Tham số truyền vào

Biến state (kiểu dữ liệu enum: state\_app\_t)

## 6.3. Giá trị trả về

Không có

# Hàm GetStateApp ( )

## 7.1. Mô tả

Hàm GetStateApp ( ) thực hiện việc lấy trạng thái hiện tại của chương trình

## 7.2. Tham số truyền vào

Không có

## 7.3. Giá trị trả về

Trả về giá trị eCurrentState là trạng thái hiện tại của chương trình.

# Hàm AppStateManager ( )

## 8.1. Mô tả

Hàm AppStateManager ( ) thực hiện việc xử lý các sự kiện của chương trình chính

## 8.2. Tham số truyền vào

Biến event (kiểu dữ liệu uint8\_t): Sự kiện khi bấm nút trên bộ KIT

## 8.3. Giá trị trả về

Không có

# Hàm LoadConfiguration ( )

## 9.1. Mô tả

Hàm LoadConfiguration ( ) thực hiện việc hiển thị dòng text “IOT Programming by Lumi Smarthome” lên màn hình LCD của Kit mở rộng khi bắt đầu cắm nguồn cho thiết bị.

## 9.2. Tham số truyền vào

Không có

## 9.3. Giá trị trả về

Không có

# Hàm DeviceStateMachine ( )

## 10.1. Mô tả

Hàm DeviceStateMachine ( ) thực hiện việc điều khiển LED, Buzzer và hiển thị thông tin sau khi nhấn các nút trên bộ KIT:

* Nhấn nút B3 năm lần khi đó tất cả các Led trên Kit mở rộng sẽ nháy năm lần màu GREEN và hiển thị thông tin sau thiết bị lên màn hình LCD:

*Device: Board STM32 Nucleo*

*Code: STM32F401RE\_NUCLEO*

*Manufacturer: STMicroelectronics*

*Kit expansion: Lumi Smarthome*

*Project: Simulator touch switch*

Sau khi hiển thị các thông tin trên, thì hiển thị thông tin nhiệt độ, độ ẩm, cường độ ánh sáng được cập nhật từ cảm biến lên màn hình LCD

* Nhấn các nút B1, B2, B4, B5 một lần để điều khiển bật/tắt màu tương ứng RED, GREEN, WHITE, BLUE của tất cả các led RGB trên Kit mở rộng và còi sẽ kêu một bíp một lần.
* Nhấn giữ B1/B5 để điều khiển tăng/giảm cường độ sáng của led RGB:
* Nhấn giữ nút B1 với thời gian T lớn hơn một giây để điều khiển tăng độ sáng của Led và nhả ra thì dừng điều khiển.
* Nhấn giữ nút B5 với thời gian T lớn hơn một giây để điều khiển giảm độ sáng của Led và nhả ra thì dừng điều khiển.

## 10.2. Tham số truyền vào

Biến event (kiểu dữ liệu uint8\_t): Sự kiện khi bấm nút trên bộ KIT

## 10.3. Giá trị trả về

Không có

# Hàm Clamp ( )

## 11.1. Mô tả

Hàm Clamp ( ) thực hiện việc tạo khoảng giá trị của biến

## 11.2. Tham số truyền vào

* Biến value (kiểu dữ liệu int): Giá trị hiện tại
* Biến min (kiểu dữ liệu int): Giá trị thấp nhất
* Biến max (kiểu dữ liệu int): Giá trị cao nhất

## 11.3. Giá trị trả về

Trả về giá trị value

# Hàm Increase\_LedLevel ( )

## 12.1. Mô tả

Hàm Increase\_LedLevel ( ) thực hiện việc tăng độ sáng đèn LED khi nhấn giữ nút B1

## 12.2. Tham số truyền vào

Không có

## 12.3. Giá trị trả về

Không có

# Hàm Decrease\_LedLevel ( )

## 13.1. Mô tả

Hàm Decrease\_LedLevel ( ) thực hiện việc giảm độ sáng đèn LED khi nhấn giữ nút B5

## 13.2. Tham số truyền vào

Không có

## 13.3. Giá trị trả về

Không có

# Hàm MultiSensorScan ( )

## 14.1. Mô tả

Hàm MultiSensorScan ( ) thực hiện việc cập nhật giá trị nhiệt độ, độ ẩm, cường độ ánh sáng

## 14.2. Tham số truyền vào

Không có

## 14.3. Giá trị trả về

Không có

# Hàm Task\_multiSensorScan ( )

## 15.1. Mô tả

Hàm Task\_multiSensorScan ( ) thực hiện việc lấy giá trị nhiệt độ, độ ẩm, cường độ ánh sáng từ cảm biến và hiển thị lên LCD

## 15.2. Tham số truyền vào

Không có

## 15.3. Giá trị trả về

Không có

# Luồng xử lý code.

## 16.1. Các bước thực xử lý

* **Câu 1:** Hiển thị dòng text “**IOT Programming by Lumi Smarthome**” lên màn hình LCD của Kit mở rộng khi bắt đầu cắm nguồn cho thiết bị.
* Bước 1: Chương trình chạy hàm main
* Gọi hàm AppInitCommon để khởi tạo các tài nguyên cần sử dụng
* Gọi hàm SetStateApp với tham số truyền vào là STATE\_APP\_STARTUP: Khởi tạo chương trình chính về trạng thái cũ của thiết bị trước khi tắt nguồn. Hàm trả về giá trị biến eCurrentState = state với state = 0
* Gọi hàm EventSchedulerAdd với tham số truyền vào là EVENT\_APP\_INIT để thêm các sự kiện khởi tạo EVENT\_APP\_INIT vào Queue
* Trong vòng lặp while(1) gọi các hàm:
* processTimerScheduler () để xử lý các sự kiện theo thời gian đã cài đặt trước
* processEventScheduler () để xử lý các sự kiện khi có trạng thái nhấn nút
* Bước 2: Trong hàm AppStateManager
* Thực hiện kiểm tra eCurrentState lấy từ hàm GetStateApp với các trường hợp:
* Trường hợp eCurrentState bằng STATE\_APP\_STARTUP chuyển sang bước 3
* Trường hợp eCurrentState bằng STATE\_APP\_IDLE chuyển sang câu 2 bước 1
* Trường hợp eCurrentState bằng STATE\_APP\_RESET chuyển về câu 1 bước 1
* Bước 3: Trường hợp eCurrentState bằng STATE\_APP\_STARTUP
* Kiểm tra event có bằng EVENT\_APP\_INIT hay không:
* Nếu đúng:
* Gọi hàm LoadConfiguration để hiển thị dòng chữ **“IOT Programming by Lumi Smarthome”** lên màn hình LCD
* Gọi hàm SetStateApp với tham số truyền vào là STATE\_APP\_IDLE
* Thoát khỏi câu điều kiện và quay trở lại bước 2
* Nếu sai: Quay trở lại bước 2
* **Câu 2:** Nhấn nút B3 năm lần khi đó tất cả các Led trên Kit mở rộng sẽ nháy năm lần màu GREEN và hiển thị thông tin sau thiết bị lên màn hình LCD:

*Device: Board STM32 Nucleo*

*Code: STM32F401RE\_NUCLEO*

*Manufacturer: STMicroelectronics*

*Kit expansion: Lumi Smarthome*

*Project: Simulator touch switch*

* Bước 1: Trường hợp eCurrentState bằng STATE\_APP\_IDLE

Gọi hàm DeviceStateMachine với tham số truyền vào là event.

* Bước 2: Trong hàm DeviceStateMachine
* Thực hiện so sánh giá trị của **event** với các trường hợp:
* EVENT\_OF\_BUTTON\_3\_PRESS\_5\_TIMES
* Nếu đúng: chuyển sang bước 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_1\_PRESS\_LOGIC
* Nếu đúng: chuyển sang câu 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_2\_PRESS\_LOGIC
* Nếu đúng: chuyển sang câu 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_4\_PRESS\_LOGIC
* Nếu đúng: chuyển sang câu 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_5\_PRESS\_LOGIC
* Nếu đúng: chuyển sang câu 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_1\_HOLD\_1S
* Nếu đúng: chuyển sang câu 4 bước 1
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_5\_HOLD\_1S
* Nếu đúng: chuyển sang câu 4 bước 1
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_1\_RELEASED\_1S
* Nếu đúng: chuyển sang câu 4 bước 3
* Nếu sai: quay trở lại bước 2
* EVENT\_OF\_BUTTON\_5\_RELEASED\_1S
* Nếu đúng: chuyển sang câu 4 bước 3
* Nếu sai: quay trở lại bước 2
* Bước 3: Trường hợp event bằng EVENT\_OF\_BUTTON\_3\_PRESS\_5\_TIMES
* Thực hiện hàm ucg\_ClearScreen, ucg\_SetFont, ucg\_DrawString để xóa màn hình, cài đặt font chữ hiển thị và hiển thị thông tin theo yêu cầu lên màn hình LCD
* Thực hiện hàm LedControl\_BlinkStart để điều khiển nháy đèn LED (màu GREEN) 5 lần
* Thực hiện hàm TimerStart để gọi hàm MultiSensorScan và chuyển sang câu 5
* **Câu 3:** Nhấn các nút B1, B2, B4, B5 một lần để điều khiển bật/tắt màu tương ứng RED, GREEN, WHITE, BLUE của tất cả các led RGB trên Kit mở rộng và còi sẽ kêu một bíp một lần
* Thực hiện kiểm tra trạng thái đèn LED theo các nút bấm tương ứng
* Trường hợp biến trạng thái đèn LED với nút nhấn tương ứng bằng 0 (đèn LED đang không sáng)
* Gán biến trạng thái đèn LED với nút nhấn tương ứng bằng 1 để thể hiện trạng thái đèn LED đang bật, đồng thời gán trạng thái các LED khác bằng 0 để đặt lại trạng thái các đèn LED đó
* Thực hiện hàm LedControl\_SetAllColor để bật đèn LED. Đối số truyền vào là màu đèn LED theo nút nhấn tương ứng và cường độ sáng là 50%
* Thực hiện hàm BuzzerControl\_SetMelody để bật còi kêu 1 tiếng bíp. Đối số truyền vào là pbeep
* Quay trở lại câu 2 bước 2
* Trường hợp trạng thái đèn LED với nút nhấn tương ứng bằng 1 (đèn LED đang sáng)
* Gán biến trạng thái đèn LED với nút nhấn tương ứng bằng 0 để thể hiện trạng thái đèn LED đang tắt
* Thực hiện hàm LedControl\_SetAllColor để tắt đèn LED. Đối số truyền vào là màu đèn LED theo nút nhấn tương ứng và cường độ sáng là 0%
* Thực hiện hàm BuzzerControl\_SetMelody để bật còi kêu 1 tiếng bíp. Đối số truyền vào là pbeep
* Quay trở lại câu 2 bước 2
* **Câu 4:** Nhấn giữ B1/B5 để điều khiển tăng/giảm cường độ sáng của led RGB:
* Nhấn giữ nút B1 với thời gian T lớn hơn một giây để điều khiển tăng độ sáng của Led và nhả ra thì dừng điều khiển.
* Nhấn giữ nút B5 với thời gian T lớn hơn một giây để điều khiển giảm độ sáng của Led và nhả ra thì dừng điều khiển.
* Bước 1: So sánh giá trị định danh ID (idTimerStartIncrease, idTimerStartDecrease) của sự kiện nút nhấn tương ứng với NO\_TIMER
* Nếu giá trị định danh ID không bằng NO\_TIMER:
* Thực hiện hàm TimerStop để dừng sự kiện. Đối số truyền vào là giá trị định danh ID của nút nhấn tương ứng
* Gán giá trị định danh ID của sự kiện nút nhấn tương ứng bằng NO\_TIMER
* Bước 2: Tăng / giảm độ sáng đèn LED
* Gán giá trị định danh ID của sự kiện nút nhấn tương ứng bằng giá trị trả về của hàm TimerStart
* Đối số truyền vào hàm TimerStart (trong trường hợp này):
* Đối số 1: Tên của sự kiện hẹn giờ tương ứng
* Đối số 2: Thời gian hẹn giờ (CYCLE\_LED\_CHANGE: 20 ms)
* Đối số 3: Số lần lặp lại (100 lần)
* Đối số 4: Con trỏ hàm thực hiện (Increase\_LedLevel tương ứng khi nhấn giữ nút B1, Decrease\_LedLevel tương ứng khi nhấn giữ nút B5)
* Đối số 5: Tham số truyền vào con trỏ hàm (NULL)
* Quay trở lại câu 2 bước 2
* Bước 3: Nhả nút nhấn, dừng tăng / giảm độ sáng đèn LED. Giữ nguyên cường độ ánh sáng của đèn LED khi nhả nút nhấn
* Nhả nút nhấn, tương ứng với các trường hợp:

EVENT\_OF\_BUTTON\_1\_RELEASED\_1S

EVENT\_OF\_BUTTON\_5\_RELEASED\_1S

* Thực hiện hàm TimerStop để dừng sự kiện tăng / giảm độ sáng đèn LED
* Quay trở lại câu 2 bước 2
* **Câu 5:** Cập nhật thông số nhiệt độ, độ ẩm và cường độ ánh sáng của các cảm biến lên màn hình LCD
* Bước 1: So sánh idTimerDisplayLCD với NO\_TIMER
* Nếu idTimerDisplayLCD không bằng NO\_TIMER
* Thực hiện hàm TimerStop để dừng sự kiện. Đối số truyền vào là idTimerDisplayLCD
* Gán idTimerDisplayLCD bằng NO\_TIMER
* Bước 2: Gọi hàm MultiSensorScan
* Gán idTimerDisplayLCD bằng giá trị trả về của hàm TimerStart
* Đối số truyền vào hàm TimerStart (trong trường hợp này):
* Đối số 1: Tên của sự kiện hẹn giờ (MultiSensorScan)
* Đối số 2: Thời gian hẹn giờ (7000 ms)
* Đối số 3: Số lần lặp lại (1 lần)
* Đối số 4: Con trỏ hàm thực hiện (hàm MultiSensorScan)
* Đối số 5: Tham số truyền vào con trỏ hàm (NULL)
* Bước 3: Thực hiện hàm MultiSensorScan
* Thực hiện hàm ucg\_ClearScreen, ucg\_SetFont để xóa màn hình và cài đặt font chữ hiển thị lên màn hình LCD
* So sánh idTimerSensorUpdate với NO\_TIMER
* Nếu idTimerSensorUpdate không bằng NO\_TIMER
* Thực hiện hàm TimerStop để dừng sự kiện. Đối số truyền vào là idTimerSensorUpdate
* Gán idTimerSensorUpdate bằng NO\_TIMER
* Bước 4: Gọi hàm Task\_multiSensorScan
* Gán idTimerSensorUpdate bằng giá trị trả về của hàm TimerStart
* Đối số truyền vào hàm TimerStart (trong trường hợp này):
* Đối số 1: Tên của sự kiện hẹn giờ (Task\_multiSensorScan)
* Đối số 2: Thời gian hẹn giờ (PERIOD\_SCAN\_MULTISENSOR: 1000 ms)
* Đối số 3: Số lần lặp lại (TIMER\_REPEAT\_FOREVER: lặp vô hạn)
* Đối số 4: Con trỏ hàm thực hiện (hàm Task\_multiSensorScan)
* Đối số 5: Tham số truyền vào con trỏ hàm (NULL)
* Bước 5: Thực hiện hàm Task\_multiSensorScan
* Lấy giá trị nhiệt độ, độ ẩm và cường độ ánh sáng từ cảm biến
* Thực hiện hàm TemHumSensor\_GetTemp để lấy giá trị nhiệt độ từ cảm biến, sau đó chia cho 100 để lấy giá trị nhiệt độ và gán vào biến temperature
* Thực hiện hàm TemHumSensor\_GetHumi để lấy giá trị độ ẩm từ cảm biến, sau đó chia cho 100 để lấy giá trị độ ẩm và gán vào biến humidity
* Thực hiện hàm LightSensor\_MeasureUseDMAMode để lấy giá trị cường độ ánh sáng từ cảm biến và gán vào biến light
* Lưu giá trị nhiệt độ, độ ẩm và cường độ ánh sáng vào các mảng tương ứng strTemp, strHumi, strLight
* Thực hiện hàm memset để gán giá trị của cả 3 mảng bằng 0
* Thực hiện hàm sprintf để ghi các giá trị đã được định dạng vào các mảng
* Thực hiện hàm ucg\_DrawString để hiển thị giá trị nhiệt độ, độ ẩm và cường độ ánh sáng ra màn hình LCD

## 16.2. Lưu đồ thuật toán

