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

**IOT304 – MẠNG TRUYỀN THÔNG KHÔNG DÂY**

**THUYẾT MINH ASSIGNMENT 1**

**GIAO TIẾP VỚI CÁC CẢM BIẾN MÔI TRƯỜNG VÀ HIỂN THỊ CÁC THÔNG SỐ LÊN MÀN HÌNH PC**

|  |  |
| --- | --- |
| Tên học viên:  Mã học viên:  Ngày báo cáo: | Dương Nhật Huy  FX29455  13/4/2024 |

HÀ NỘI, 2/2024

MỤC LỤC

[**1. Danh sách thư viện** 1](#_Toc164895999)

[**2. Các sự kiện và macro được khai báo** 1](#_Toc164896000)

[**3. Danh sách các hàm** 2](#_Toc164896001)

[**4 . Hàm initI2C**() 3](#_Toc164896002)

[**5. Hàm I2C\_LeaderRead**() 3](#_Toc164896003)

[**6. Hàm** **TempHumiSensor\_getTemp**() 3](#_Toc164896004)

[**7. Hàm TempHumiSensor\_getHumi**() 4](#_Toc164896005)

[**8. Hàm initIADC** (**void**) 4](#_Toc164896006)

[**9. float** **LightSensor\_getLight**() 4](#_Toc164896007)

[**11. Hàm TempHumi\_LightEventHandler**() 7](#_Toc164896008)

[**12. Luồng xử lý code** 7](#_Toc164896009)

NỘI DUNG THUYẾT MINH

# **1. Danh sách thư viện**

|  |  |  |
| --- | --- | --- |
| STT | Thư viện | Mô tả |
| 1 | af.h | - Cung cấp struct EmberEventControl để khai báo các hàm sự kiện.  - Cung cấp các hàm xử lý sử kiện:  + emberEventControlSetActive : Hàm khởi động sự kiện.  + emberEventControlSetInActive : Hàm vô hiệu hóa sự kiện  + emberEventControlSetDelayMS : Hàm khởi động sự kiện sau một thời gian nhất định/  - Cung cấp hàm emberAfCorePrintln để ghi ra màn hình. |
| 2 | em\_iadc.h | Thư viện cung cấp các hàm xử lý iadc |
| 3 | em\_cmu.h | Thư viện cung cấp các hàm xử lý xung clock |
| 4 | em\_i2c.h | Thư viện cung cấp các hàm giao tiếp i2c |
| 5 | math.h | Thư viện cung cấp các hàm xử lý toán học |
| 6 | stdio.h | Thư viện cung cấp hàm sprintf để chuyển số thành chuỗi. |

# **2. Các sự kiện và macro được khai báo**

|  |  |  |
| --- | --- | --- |
| STT | Sự kiện và Macro | Mô tả |
| 1 | TempHumi\_LightEventControl | Sự kiện đọc nhiệt độ, độ ẩm, cường độ ánh sáng rồi ghi lên PC theo chu kỳ |

# 

# **3. Danh sách các hàm**

|  |  |  |
| --- | --- | --- |
| STT | Danh sách hàm | Mô tả |
| 1 | **void emberAfMainInitCallback(void)** | Hàm khởi động khi mới chạy chương trình |
| 2 | **Void initI2C**(**void**) | Hàm cấu hình I2C để giao tiếp với cảm biến nhiệt độ, độ ẩm theo phương pháp Polling |
| 3 | **I2C\_LeaderRead**(  **uint16\_t followerAddress,**  **uint8\_t command,**  **uint8\_t numBytes**  ) | Hàm đọc dữ liệu từ cảm biến |
| 4 | **float** **TempHumiSensor\_getTemp**() | Hàm lấy giá trị nhiệt độ |
| 5 | **float** **TempHumiSensor\_getHumi**() | Hàm lấy giá trị độ ẩm |
| 6 | **void** **initIADC** (**void**) | Hàm cấu hình iadc để giao tiếp với cảm biến ánh sáng |
| 7 | **uint\_16** **LightSensor\_getLight**() | Hàm lấy giá trị cường độ ánh sáng |
| 8 | **void TempHumi\_LightEventHandler**() | Hàm sự kiện có nhiệm vụ đọc và ghi giá trị nhiệt độ, độ ẩm và cường độ ánh sáng ra màn hình theo chu kỳ |
| **9** | **double KalmanFilter**  **(**  **double mea**  **int i**  **)** | Bộ lọc Kalman xử lý nhiễu từ các cảm biến. |

# **4 . Hàm initI2C**()

*a. Mô tả*

Hàm cấu hình I2C để giao tiếp với cảm biến nhiệt độ, độ ẩm Si7020 theo phương pháp Polling.

# **5. Hàm I2C\_LeaderRead**()

*a. Mô tả*

Đọc dữ liệu từ cảm biến nhiệt độ, độ ẩm bằng giao tiếp I2C

*b. Tham số truyền vào*

uint16\_t followerAddress : Địa chỉ Slave của cảm biến cần giao tiếp I2C

uint8\_t command: : Mệnh lệnh đọc dữ liệu từ Cảm biến.

0xE0: Để đọc giá trị nhiệt độ

0xE5: Để đọc giá trị độ ẩm

uint8\_t numBytes : : Kích thước của giữ liệu trả về ( Tính bằng byte)

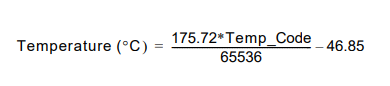
*c. Tham số trả về*

Dữ liệu đọc được từ cảm biến

# **6. Hàm** **TempHumiSensor\_getTemp**()

*a. Mô tả*

Hàm lấy giá trị nhiệt độ. Áp dụng công thức.



Trong đó, Temperature(oC) là giá trị nhiệt độ cần tính toán.

Temp\_Code là dữ liệu 16-bit đọc được từ cảm biến Si7020

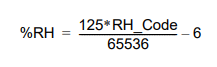
*b. Giá trị trả về*

Giá trị nhiệt độ.

# **7. Hàm TempHumiSensor\_getHumi**()

*a. Mô tả*

Hàm lấy giá trị độ ẩm



Trong đó %RH là giá trị độ ẩm.

RH\_Code là dữ liệu 16-bit đọc được từ cảm biến Si7020

*b. Giá trị trả về*

Giá trị độ ẩm.

# **8. Hàm initIADC** (**void**)

*a. Mô tả*

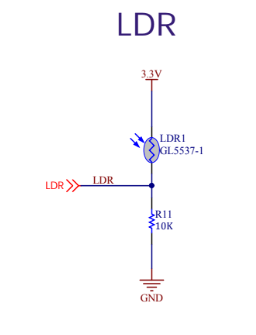
Hàm cấu hình IADC0 giao tiếp với cảm biến ánh sáng GL5537-1 theo phương pháp Polling, chế độ chuyển đổi một lần, độ phân giải 12-bit.

# **9. Hàm** **LightSensor\_getLight**()

*a. Mô tả*

Hàm lấy giá trị cường độ ánh sáng.

Bước 1: Tính điện trở hiện tại của GL5537-1



RLDR = 10 x (4096 - VLDR) / VLDR

Trong đó VLDR là kết quả thu được từ IADC0 với độ phân giải 12bit.

RLDR là điện trở của GL5537-1(KΩ)

Bước 2:

Thiết lập công thức liên hệ giữa RLDR và Lux.

Tra datasheet GL5537-1, ta có:

Tại RLDR = 25 KΩ, Lux = 10

Lg(R10/R100) = 0.6 (Trong đó R10 và R100 là điện trở của cảm biến khi cường độ ánh sáng đạt 10 Lux và 100 Lux)

A graph of a line graph

Description automatically generated

Sau các bước tính toán, ta tìm ra công thức liên hệ:

Log(Lux) = - Log(RLDR) \* 5 / 3 + 3.3299;

*b. Giá trị trả về*

Giá trị cường độ ánh sáng đo bằng lux.

**10. double KalmanFilter()**

*a. Mô tả*

Bộ lọc nhiễu Kalman

measurement equation

Trong đó:

là sai số ước lượng

là hệ số Kalman

là nhiễu cảm biến

là nhiễu quá trình

là giá trị được lọc

là giá trị đo được

*b. Tham số truyền vào*

double mea : là giá trị đo được từ cảm biến

uint8\_t i :

- i = 0 xử lý giá trị nhiệt độ

- i = 1 xử lý giá trị độ ẩm

- i = 2 xử lý giá trị cường độ ánh sáng

*c. Tham số trả về*

Giá trị sau khi đã lọc nhiễu.

# **11. Hàm TempHumi\_LightEventHandler**()

*a. Mô tả*

Lấy giữ liệu từ các cảm biến, sau đó ghi ra màn hình giá trị nhiệt độ độ ẩm cứ sau mỗi 5s và giá trị cường độ ánh sáng cứ sau mỗi 10s.

# **12. Luồng xử lý code**

Các bước thực xử lý:

*Bước 1:* Khi mới khởi động chương trình, gọi các hàm initI2C0(), initIADC0() để cấu hình giao tiếp I2C với

*Bước 2:* Khởi động sự kiện TempHumi\_LightEventControl để tiến hành gửi các giá trị nhiệt độ, độ ẩm, ánh sáng lên màn hình máy tính.