**TỔNG QUAN CÁC BƯỚC**

**Bước 1:** Khai báo các thư viện; định nghĩa macro để sử dụng ngoại vi SPI, I2S.

**Bước 2:** Khởi tạo **System Clock** và các ngoại vi: **TIMER, LCD, I2C, SPI, cảm biến nhiệt độ, độ ẩm.**

**Bước 3:** Xây dựng các hàm I2C phục vụ giao tiếp với cảm biến nhiệt độ, độ ẩm.

**Bước 4:** Xây dựng hàm I2C lấy dữ liệu cảm biến nhiệt độ, độ ẩm.

**Bước 5:** Xây dựng hàm xử lý dữ liệu nhiệt độ, độ ẩm và hiển thị.

**THUYẾT MINH**

**Bước 1:** Khai báo các thư viện; định nghĩa macro để sử dụng ngoại vi SPI, I2S.

**Bước 2:** Khởi tạo **System Clock** và các ngoại vi: **TIMER, LCD, I2C, SPI, cảm biến nhiệt độ, độ ẩm.**

**1. Khởi tạo System Clock:** SystemCoreClockUpdate().

**2. Khởi tạo SPI**

/\*

\* @func Soft\_SPI\_Init

\* @brief To configure all specifications of the soft SPI protocol

\* @param None

\* @retval None

\*/

**static** **void** **Soft\_SPI\_Init**(**void**)

**3. Khởi tạo I2C**

/\*

\* @func I2C1\_Init

\* @brief To configure all specifications of I2C protocol

\* @param None

\* @retval None

\*/

**static** **void** **I2C1\_Init**(**void**)

**4. Khởi tạo LCD**

/\*

\* @func I2C1\_Init

\* @brief To configure all specifications of I2C protocol

\* @param None

\* @retval None

\*/

**static** **void** **LCD\_Init**(**void**)

**5. Khởi tạo chung**

/\*

\* @func AppInitCommon

\* @brief To initialize all peripherals

\* @param None

\* @retval None

\*/

**static** **void** **AppInitCommon**(**void**)

**Bước 3:** Xây dựng các hàm I2C phục vụ giao tiếp với cảm biến nhiệt độ, độ ẩm.

**1. Hàm tạo bit START**

/\*

\* @func I2C1\_Start

\* @brief To generate start condition

\* @param None

\* @retval None

\*/

**static** **void** **I2C1\_Start**(**void**)

**\* Thuật toán:**

B1: Kiểm tra giao tiếp I2C có đang giao tiếp hay không.

B2: Tạo bit start (Set bit START trong thanh ghi I2C\_CR1).

B3: Đợi thiết lập cờ SB trong thanh ghi I2C\_SR1.

**2. Hàm truyền địa chỉ cảm biến**

/\*

\* @func I2C1\_TransmitAddressSensor

\* @brief To transmit the address of sensor and select the mode is read or write

\* @param

\* Sensor\_Address - address of sensor

\* R\_W - action: read of write

\* @retval None

\*/

**static** **void** **I2C1\_TransmitAddressSensor**(uint8\_t Sensor\_Address, uint8\_t R\_W)

**\* Thuật toán:**

B1: Truyền địa chỉ cảm biến + Bit Read or Write.

B2: Đợi thiết lập cờ ADDR trong thanh ghi I2C\_SR1.

**3. Hàm truyền địa chỉ thanh ghi của cảm biến**

/\*

\* @func I2C1\_TransmitSensorRegisterAddress

\* @brief To write the address of sensor registers into DR to request data

\* @param Sensor\_Register\_Address - The address of sensor register

\* @retval None

\*/

**static** **void** **I2C1\_TransmitSensorRegisterAddress**(uint8\_t Sensor\_Register\_Address)

**\* Thuật toán:**

B1: Truyền địa chỉ thanh ghi cảm biến.

B2: Đợi thiết lập cờ TxE, BTF trong thanh ghi I2C\_SR1.

**4. Hàm tạo bit STOP**

/\*

\* @func I2C1\_Stop

\* @brief To generate stop condition

\* @param None

\* @retval None

\*/

**static** **void** **I2C1\_Stop**(**void**)

**5. Hàm đọc dữ liệu sử dụng ACK và NACK**

/\*

\* @func I2C1\_Receive\_ACK

\* @brief To receive data byte from I2C bus, then return ACK

\* @param None

\* @retval Received data byte

\*/

**static** uint8\_t **I2C1\_Receive\_ACK**(**void**)

**\* Thuật toán:**

B1: Set bit ACK trong thanh ghi I2C\_CR1.

B2: Đợi thiết lập cờ RxNE trong thanh ghi I2C\_SR1.

B3: Đọc dữ liệu.

/\*

\* @func I2C1\_Receive\_NACK

\* @brief To receive data byte from I2C bus, then return NACK

\* @param None

\* @retval Received data byte

\*/

**static** uint8\_t **I2C1\_Receive\_NACK**(**void**)

**\* Thuật toán:**

B1: Reset bit ACK trong thanh ghi I2C\_CR1.

B2: Đợi thiết lập cờ RxNE trong thanh ghi I2C\_SR1.

B3: Đọc dữ liệu.

**Bước 4:** Xây dựng hàm I2C lấy dữ liệu cảm biến nhiệt độ, độ ẩm.

/\*

\* @func TemHumSensor\_readRegister

\* @brief To get the sensor value

\* @param

- Sensor\_Address - The address of I2C\_Slave

- Temp\_Humi\_Register\_Address - The TEMP, HUM register address of I2c\_Slave

\* @retval None

\*/

**static** uint16\_t **TemHumSensor\_readRegister**(uint8\_t Sensor\_Address, uint8\_t Temp\_Humi\_Register\_Address)

**\* Thuật toán:**

B1: Tạo bit START.

B2: Truyền địa chỉ của cảm biến + Bit ghi.

B3: Truyền địa chỉ thanh ghi của cảm biến.

B4: Tạo bit STOP.

B5: Tạo bit RESTART.

B6: Truyền địa chỉ thanh ghi của cảm biến + Bit đọc.

B7: Sử dụng ACK, đọc dữ liệu, chuyển dữ liệu đó lên byte cao.

B8: Sử dụng NACK, đọc dữ liệu.

B9: Tạo bit STOP.

B10: Trả lại giá trị cảm biến cho hàm.

**Bước 5:** Xây dựng hàm xử lý dữ liệu nhiệt độ, độ ẩm và hiển thị.

**1. Hàm xử lý dữ liệu nhiệt độ, độ ẩm**

/\*

\* @func TemHumSensor\_Get\_ProcessTemperature

\* @brief To process the temperature data

\* @param None

\* @retval 0-255

\*/

**static** uint8\_t **TemHumSensor\_Get\_ProcessTemperature**(**void**)

/\*

\* @func TemHumSensor\_Get\_ProcessHumidity

\* @brief To process the humidity data

\* @param None

\* @retval 0-255

\*/

**static** uint8\_t **TemHumSensor\_Get\_ProcessHumidity**(**void**)

**\* Thuật toán:**

B1: Lấy dữ liệu nhiệt độ, độ ẩm và xử lý.

B2: Đưa dữ liệu cảm biến vào mảng và hiển thị.

**2. Hàm hiển thị dữ liệu**

/\*

\* @func DisplayLCD

\* @brief To get the processed sensor value and display into LCD

\* @param None

\* @retval None

\*/

**static** **void** **DisplayLCD**(**void**)

/\*

\* @func createDisplayingTime

\* @brief To create a fixed period to display temp, hum into LCD

\* @param None

\* @retval None

\*/

**static** **void** **createDisplayingTime**(**void**)

**\* Thuật toán:**

B1: Lấy thời gian hiện tại và so sánh.

B2: Nếu thỏa mãn chu kỳ hiển thị, hiển thị dữ liệu.