**VI XỬ LÝ – VI ĐIỀU KHIỂN**

**LAB 3 - Lớp: L03**

**Họ và tên:** Nguyễn Đình Đạt

**Mssv:** 1811869

**4.1.3**

Đối với cả xử lý đầu vào và đầu ra, chúng ta có một mẫu tương tự để làm việc. Thông thường, chúng ta có một trình điều khiển có tên mô-đun hoạt động trực tiếp với phần cứng. Chúng tôi cũng có một bộ đệm để lưu trữ các giá trị tạm thời. Trong trường hợp xử lý đầu vào, trình điều khiển sẽ lưu trữ giá trị của trạng thái phần cứng vào bộ đệm để xử lý thêm. Trong trường hợp đầu ra xử lý, trình điều khiển sử dụng dữ liệu đệm để xuất ra phần cứng

Hình 1.6 cho thấy rằng chúng ta nên có một mô- đun input\_reading để xử lý các nút, sau đó lưu trữ dữ liệu đã xử lý vào bộ đệm. Sau đó, một mô-đun của input\_output\_processing\_fsm sẽ xử lý dữ liệu đầu vào và cập nhật bộ đệm đầu ra. Trình điều khiển đầu ra nhận giá trị từ bộ đệm đầu ra để chuyển đến phần cứng.

**Problem 1:**

* Có bộ hẹn giờ ngắt sau mỗi 10 mili giây
* Đọc giá trị của nút 1 và 2 sau 10 mili giây. Chức năng nút đọc nên được gọi bên trong quy trình dịch vụ ngắt bộ định thời.
* Tăng giá trị của đèn LED được kết nối PORTC khi nhấn nút 1
* Tự động tăng giá trị của PORTC sau mỗi 0.5 giây, nếu nút 1 được nhấn trong hơn 1 giây
* Tự động tăng giá trị của PORTC sau mỗi 1 giây, nếu nút 1 được nhấn trong hơn 3 giây
* Giảm giá trị của PORTC khi nhấn nút 2
* Tự động giảm giá trị của PORTC sau mỗi 0.5 giây, nếu nút 2 là được nhấn trong hơn 1 giây.
* Tự động giảm giá trị của PORTC sau mỗi 1 giây, nếu nút 2 là được nhấn trong hơn 3 giây.
* Các giá trị trên như 10ms, 0.5s, 0.1s, 1s và 3s là các giá trị cố định làm ví dụ. Tuy nhiên, chương trình của bạn cần cung cấp một cách dễ dàng để thay đổi các giá trị đó. Ví dụ, bạn nên sử dụng DEFINE để xác định các giá trị đó.
* Nếu nhấn cả hai nút 1 và 2, nút 1 có mức ưu tiên cao hơn

**Problem 2:**

* Đồng hồ bình thường hiển thị giờ, phút, giây trên 6 đèn LED bảy đoạn
* Đồng hồ bấm giờ hiển thị phút, giây và 1/100 giây
* Chúng tôi sử dụng nút 1 để thay đổi các chế độ của đồng hồ kỹ thuật số. Mỗi lần nhấn nút 1, đồng hồ sẽ chuyến sang chế độ tiếp theo
* Chế độ 0: chạy đồng hồ bình thường (mặc định)
* Chế độ 1: sửa đổi giờ. Ở chế độ này, số giờ nhấp nháy và nút 2 là được sử dụng để tăng số giờ. Nếu nút 2 được nhấn hơn 1 giây, số giờ sẽ tự động tăng lên, tức là 5 lần mỗi giây. Xin lưu ý rằng số giờ sẽ được trả về 0 khi nó đạt đến 23
* Chế độ 2: sửa đổi phút. Tương tự như chế độ 1, số phút nhấp nháy và tăng khi nhấn nút 2
* Chế độ 3: sửa đổi giây. Tương tự như chế độ 1, số giây nhấp nháy và tăng khi nhấn nút 2
* Chế độ 4: chạy đồng hồ bấm giờ. Sử dụng nút 2 để khởi động và dung đồng hồ

Xin lưu ý rằng trong khi đồng hồ bấm giờ đang chạy, đồng hồ bình thường vẫn chạy trong lai lịch

**Example:**

**void** **clear\_all**(){

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_0\_Pin|RED\_LED\_1\_Pin|

RED\_LED\_2\_Pin|RED\_LED\_3\_Pin|RED\_LED\_4\_Pin|RED\_LED\_5\_Pin|

RED\_LED\_6\_Pin|RED\_LED\_7\_Pin, 0);

}

**void** **increa\_led**(**int** i){

**switch**(i){

**case** 0:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_0\_Pin, 1);

**break**;

**case** 1:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_1\_Pin, 1);

**break**;

**case** 2:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_2\_Pin, 1);

**break**;

**case** 3:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_3\_Pin, 1);

**break**;

**case** 4:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_4\_Pin, 1);

**break**;

**case** 5:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_5\_Pin, 1);

**break**;

**case** 6:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_6\_Pin, 1);

**break**;

**case** 7:

HAL\_GPIO\_WritePin(GPIOA, RED\_LED\_7\_Pin, 1);

**break**;

**default**:

clear\_all();

**break**;

}

}

/\* USER CODE BEGIN 4 \*/

**int** i = -1;

**int** c = 0;

**int** count = 100;

**int** count\_tr = 50;

**void** **HAL\_TIM\_PeriodElapsedCallback** ( TIM\_HandleTypeDef \* htim ){

increa\_led(i);

**if**(HAL\_GPIO\_ReadPin(GPIOB, BUTTON\_1\_Pin) == 1){

c = 1;

count--;

**if**(count <= 0){

count\_tr--;

**if**(count\_tr <= 0){

i++;

clear\_all();

**if**(i > 7){

i = 0;

}

count\_tr = 50;

}

}

}

**if**(c == 1 && HAL\_GPIO\_ReadPin(GPIOB, BUTTON\_1\_Pin) == 0){

count = 100;

count\_tr = 50;

c = 0;

i++;

clear\_all();

**if**(i > 7){

i = 0;

}

}

}

**Problem 1:** /\*

\* input\_processing.c

\*

\* Created on: Sep 22, 2021

\* Author: HH

\*/

**#include** "main.h"

**#include** "input\_reading.h"

**#include** "led\_display.h"

**#define** NUMBER\_LED 7

**enum** ButtonState{*BUTTON\_RELEASED*, *BUTTON\_PRESSED*, *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND*};

**enum** ButtonState buttonState = *BUTTON\_RELEASED*;

**enum** ButtonState1{*BUTTON\_RELEASED\_1*, *BUTTON\_PRESSED\_1*, *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND\_1*};

**enum** ButtonState1 buttonState1 = *BUTTON\_RELEASED\_1*;

**int** c = 0;

**int** d = 0;

**void** **fsm\_for\_input\_processing**(**void**){

**switch**(buttonState){

**case** *BUTTON\_RELEASED*:

**if**(get\_button\_value(0) == *GPIO\_PIN\_SET*){

buttonState = *BUTTON\_PRESSED*;

c = 1;

}

**break**;

**case** *BUTTON\_PRESSED*:

**if**(get\_button\_value(0) == *GPIO\_PIN\_RESET*){

buttonState = *BUTTON\_PRESSED*;

**if**(c == 1){

increases\_led();

c=0;

}

}

**else**{

**if**(get\_flag\_for\_button\_press\_1s(0) == 1){

buttonState = *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND*;

}

c = 1;

}

**break**;

**case** *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND*:

**if**(get\_button\_value(0) == *GPIO\_PIN\_RESET*){

buttonState = *BUTTON\_RELEASED*;

}

//to do

**else**{

auto\_increases\_led();

}

**break**;

}

**if**(get\_button\_value(0) == *GPIO\_PIN\_RESET*){

**switch**(buttonState1){

**case** *BUTTON\_RELEASED\_1*:

**if**(get\_button\_value(1) == *GPIO\_PIN\_SET*){

buttonState1 = *BUTTON\_PRESSED\_1*;

d = 1;

}

**break**;

**case** *BUTTON\_PRESSED\_1*:

**if**(get\_button\_value(1) == *GPIO\_PIN\_RESET*){

buttonState1 = *BUTTON\_PRESSED\_1*;

**if**(d == 1){

decreases\_led();

d=0;

}

}

**else**{

**if**(get\_flag\_for\_button\_press\_1s(1) == 1){

buttonState1 = *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND\_1*;

}

d = 1;

}

**break**;

**case** *BUTTON\_PRESSED\_MORE\_THAN\_1\_SECOND\_1*:

**if**(get\_button\_value(1) == *GPIO\_PIN\_RESET*){

buttonState1 = *BUTTON\_RELEASED\_1*;

}

//to do

**else**{

auto\_decreases\_led();

}

**break**;

}

}

}