微處理機 LAB 2.2

Due : 兩週後 早上8:00

第21組 郭宗諺 林穎沛

PART 1. (10%)

1. Moder設定:

RCC -> AHB2ENR |= RCC\_AHB2ENR\_GPIOBEN;

gpio -> MODER &= A1; //清空PB0~PB3

gpio -> MODER |= A2; //寫入PB0~ PB3

請填入A1及A2 ?

A1 = 0xFFFFFCC

1111 1111 1111 1111 1111 1111 0000 0000

A2 = 0xFFFFF55

1111 1111 1111 1111 1111 1111 0101 0101

PART 2. (90%) 實作題

請完成實驗 錄影及截圖紀錄實驗結果並附上程式碼(main.s及include之pin.s檔案)

1. LED跑馬燈+按鍵控制  C語言版  
   功能: 按下版上的藍色按鈕(GPIOC 13)，讓跑馬燈停止/啟動 (30%)
2. 將GPIOB3更改為GPIOA7。程式和pin腳都要改才計分。(30%)
3. 然後修改成一顆，兩顆，三顆LED左右跑，按按鈕切換。(30%)

(HINT: led\_data)

一張含有 文字 的圖片

自動產生的描述

#include "stm32l476xx.h"

#include "helper\_functions.h"

#include "led\_button.h"

#include "7seg.h"

// Define pins for 4 leds

#define LED\_gpio GPIOA

#define LED1\_pin 7

#define LED2\_pin 6

#define LED3\_pin 5

#define LED4\_pin 4

// Define pins for button (default use on-board button PC13)

#define BUTTON\_gpio GPIOC

#define BUTTON\_pin 13

// Define pins for 7seg

//#define SEG\_gpio GPIOB

//#define DIN\_pin 3

//#define CS\_pin 4

//#define CLK\_pin 5

// Use to decide which part of the code will run

// Use define & ifdef to control

#define lab\_led

//#define lab\_led\_button

//#define lab\_7seg\_non\_decode

//#define lab\_7seg\_decode

int main(){

#ifdef lab\_led

    if(init\_led(LED\_gpio, LED1\_pin) != 0 || init\_led(LED\_gpio, LED2\_pin) != 0 || init\_led(LED\_gpio, LED3\_pin) != 0 || init\_led(LED\_gpio, LED4\_pin) != 0){

        // Fail to init LED

        return -1;

    }

    if(init\_button(BUTTON\_gpio, BUTTON\_pin) != 0 ){

        return -1;

    }

    // Direction to shift 0=left, 1=right

    int shift\_direction = 0;

    int led\_data = 0b000001;

    int leds[4] = {LED1\_pin, LED2\_pin, LED3\_pin, LED4\_pin};

    int state = 1;

    int ch\_state = 0;

    int button\_press\_persecond\_cycle = 10;

    int debounce\_cycle = 100;

    int debounce\_threshold = 70;

    int last\_botton\_pos =1;

    while(1){

        if(state == 1){//1 led

                // Shift led\_data

                if(shift\_direction==0){

                    led\_data = (led\_data<<1);

                }

                else{

                    led\_data = (led\_data>>1);

                }

                // Check to see if change shift direction

                if(led\_data==0b000001 || led\_data==0b100000){

                    shift\_direction = 1-shift\_direction;

                }

        }

        if(state == 3){//2 led

                // Shift led\_data

                if(shift\_direction==0){

                    led\_data = (led\_data<<1);

                }

                else{

                    led\_data = (led\_data>>1);

                }

                // Check to see if change shift direction

                if(led\_data==0b000011 || led\_data==0b110000){

                    shift\_direction = 1-shift\_direction;

                }

        }

        if(state == 5){//3 led

                // Shift led\_data

                if(shift\_direction==0){

                    led\_data = (led\_data<<1);

                }

                else{

                    led\_data = (led\_data>>1);

                }

                // Check to see if change shift direction

                if(led\_data==0b000111 || led\_data==0b111000){

                    shift\_direction = 1-shift\_direction;

                }

        }

        for(int a=0;a<4;a++){

            // Active

            if((led\_data>>(a+1))&0x1){

                set\_gpio(LED\_gpio, leds[a]);

            }

            else{

                reset\_gpio(LED\_gpio, leds[a]);

            }

        }

        for(int i=0; i<button\_press\_persecond\_cycle; i++){

                int pos\_cnt = 0; //count

                for(int a=0; a<debounce\_cycle; a++){

                    if(read\_gpio(BUTTON\_gpio, BUTTON\_pin) == 0){

                        pos\_cnt++;

                    }

                    delay\_without\_interrupt(1000/(button\_press\_persecond\_cycle\*debounce\_cycle));

                }

                if(pos\_cnt > debounce\_threshold){

                    if(last\_botton\_pos == 1){

                        last\_botton\_pos = 0;

                    }

                }else{

                    if(last\_botton\_pos == 0){

                        state++;// state mode: no, 1, no, 2, no, 3

                        if(state == 1){

                            led\_data = 0b000001;

                            shift\_direction = 0;

                        }else if(state == 3){

                            led\_data = 0b000011;

                            shift\_direction = 0;

                        }else if(state == 5){

                            led\_data = 0b000111;

                            shift\_direction = 0;

                        }else if(state == 6){

                            state = 0;

                        }

                        last\_botton\_pos = 1;

                    }

                }

        }

    }

#endif

    while(1){}

    return 0;

}

PART 3. 加分題(10%)

將PART2-1的跑馬燈從按下按鈕停止改為按下按鈕切換為以下跑馬燈:

●為熄滅 ○為亮起

LED燈腳為由右至左依序為PB3~PB6

○○○●-> ○○●●-> ○●●○-> ●●○○->●○○○-> ●●○○->

○●●○-> ○○●●-> 循環

#include "stm32l476xx.h"

#include "helper\_functions.h"

#include "led\_button.h"

#include "7seg.h"

// Define pins for 4 leds

#define LED\_gpio GPIOA

#define LED1\_pin 7

#define LED2\_pin 6

#define LED3\_pin 5

#define LED4\_pin 4

// Define pins for button (default use on-board button PC13)

#define BUTTON\_gpio GPIOC

#define BUTTON\_pin 13

// Define pins for 7seg

//#define SEG\_gpio GPIOB

//#define DIN\_pin 3

//#define CS\_pin 4

//#define CLK\_pin 5

// Use to decide which part of the code will run

// Use define & ifdef to control

#define lab\_led

//#define lab\_led\_button

//#define lab\_7seg\_non\_decode

//#define lab\_7seg\_decode

int main(){

#ifdef lab\_led

    if(init\_led(LED\_gpio, LED1\_pin) != 0 || init\_led(LED\_gpio, LED2\_pin) != 0 || init\_led(LED\_gpio, LED3\_pin) != 0 || init\_led(LED\_gpio, LED4\_pin) != 0){

        // Fail to init LED

        return -1;

    }

    if(init\_button(BUTTON\_gpio, BUTTON\_pin) != 0 ){

        return -1;

    }

    // Direction to shift 0=left, 1=right

    int shift\_direction = 0;

    int led\_data = 0b00011;

    int leds[4] = {LED1\_pin, LED2\_pin, LED3\_pin, LED4\_pin};

    int state = 1;

    int ch\_state = 0;

    int button\_press\_persecond\_cycle = 10;

    int debounce\_cycle = 100;

    int debounce\_threshold = 70;

    int last\_botton\_pos =1;

    while(1){

        if(state == 1){//normal mode

                // Shift led\_data

                if(shift\_direction==0){

                    led\_data = (led\_data<<1);

                }

                else{

                    led\_data = (led\_data>>1);

                }

                // Check to see if change shift direction

                if(led\_data==0b000011 || led\_data==0b110000){

                    shift\_direction = 1-shift\_direction;

                }

        }else{//new mode

            // Shift led\_data

                if(shift\_direction==0){

                    led\_data = (led\_data<<1);

                }

                else{

                    led\_data = (led\_data>>1);

                }

                // Check to see if change shift direction

                if(led\_data==0b000011 || led\_data==0b11000){//reduce one side's data

                    shift\_direction = 1-shift\_direction;

                }

        }

        for(int a=0;a<4;a++){

            // Active

            if((led\_data>>(a+1))&0x1){

                set\_gpio(LED\_gpio, leds[a]);

            }

            else{

                reset\_gpio(LED\_gpio, leds[a]);

            }

        }

        for(int i=0; i<button\_press\_persecond\_cycle; i++){

                int pos\_cnt = 0; //count

                for(int a=0; a<debounce\_cycle; a++){

                    if(read\_gpio(BUTTON\_gpio, BUTTON\_pin) == 0){

                        pos\_cnt++;

                    }

                    delay\_without\_interrupt(1000/(button\_press\_persecond\_cycle\*debounce\_cycle));

                }

                if(pos\_cnt > debounce\_threshold){

                    if(last\_botton\_pos == 1){

                        last\_botton\_pos = 0;

                    }

                }else{

                    if(last\_botton\_pos == 0){

                        state = 1 - state;

                        led\_data = 0b001100;

                        last\_botton\_pos = 1;

                    }

                }

        }

    }

#endif

    while(1){}

    return 0;

}