#include "STM32L1xx.h" /\* Microcontroller information \*/

#define UP 1;

#define DOWN 0;

#define TRUE 1;

#define FALSE 0;

struct {

unsigned char count;

unsigned char direction;

unsigned char disable;

}

typedef counter;

void delay(void);

void count(counter\* count);

void setup\_pins(void);

void update\_leds(void);

void setup\_interupts(void);

void counter\_factory(counter\* c);

counter count\_one;

counter count\_two;

unsigned char LED8;

unsigned char LED9;

//Delay function - do nothing for about 1/2 second

void delay () {

int i,j;

for (i=0; i<10; i++) { //outer loop

for (j=0; j<20000; j++) { //inner loop

asm("nop"); //dummy operation for single-step test

} //do nothing

}

}

//Incriment or decriment toggle mod 10 either up

or down.

void count(counter\* count) {

if (count->disable) return;

if (count->direction) {

count->count = (count->count + 1) % 10; //UP

}

else {

count->count = (count->count + (10 - 1)) % 10; //DOWN

}

}

//Update the LEDS used in the lab

void update\_leds(void) {

unsigned short leds = 0;

leds = (count\_two.count & 0x0F) << 4; // PC[7:4]

leds += (count\_one.count & 0x0F); // PC[3:0]

SET\_BIT(GPIOC->BSRR, (~leds & 0xFF) << 16); // turn off LEDs

SET\_BIT(GPIOC->BSRR, leds); // turn on LEDs

}

//Initialize GPIO pins used in the program

void setup\_pins () {

// Configure PA0 and PA1 as input pin to read push buttons

SET\_BIT(RCC->AHBENR, RCC\_AHBENR\_GPIOAEN); // Enable GPIOA clock (bit 0)

CLEAR\_BIT(GPIOA->MODER, (GPIO\_MODER\_MODER0 | GPIO\_MODER\_MODER1)); // set to input mode

// Configure PC[0,9] as output pins to drive LEDs

SET\_BIT(RCC->AHBENR, RCC\_AHBENR\_GPIOCEN); // Enable GPIOC clock (bit 2) \*/

CLEAR\_BIT(GPIOC->MODER, 0x000FFFFF); /\* Clear PC[0,9] mode bits \*/

SET\_BIT(GPIOC->MODER, 0x00055555); /\* General purpose output mode\*/

}

void setup\_interupts() {

SYSCFG->EXTICR[0] &= 0xFFF0; //clear EXTL0 field

SYSCFG->EXTICR[0] &= 0xFF0F; //clear EXTL1 field

SYSCFG->EXTICR[0] |= 0x0001; //set to PA0

SYSCFG->EXTICR[0] |= 0x0002; //set to PA1

SET\_BIT(EXTI->IMR, 0x0001);//enable bit

SET\_BIT(EXTI->IMR, 0x0002);//enable bit

SET\_BIT(EXTI->RTSR, 0x0001);//rising edge

SET\_BIT(EXTI->RTSR, 0x0002);//rising edge

NVIC\_EnableIRQ(EXTI0\_IRQn);// enable external interrupt EXTI0

NVIC\_EnableIRQ(EXTI1\_IRQn);// enable external interrupt EXTI1

NVIC\_ClearPendingIRQ(EXTI0\_IRQn); //avoiding duplicate service

NVIC\_ClearPendingIRQ(EXTI1\_IRQn); //avoiding duplicate service

}

void small\_delay () {

int i;

for (i=0; i<100000; i++) { //outer loop

asm("nop"); //dummy operation for single-step test

}

}

void EXTI0\_IRQHandler() {

// Hanndle Pushbutton 0

// Acknowledge interupt

count\_two.direction = DOWN;

if (LED8) {

GPIOC->BSRR = GPIO\_BSRR\_BR\_8;

LED8 = 0;

} else {

GPIOC->BSRR = GPIO\_BSRR\_BS\_8; //Set PC8=1 to turn ON blue LED

LED8 = 1;

}

//GPIOC->BSRR = GPIO\_BSRR\_BR\_9; //Reset PC9=0 to turn OFF green LED

small\_delay();

SET\_BIT(EXTI->PR, EXTI\_PR\_PR0);

NVIC\_ClearPendingIRQ(EXTI0\_IRQn);

}

void EXTI1\_IRQHandler() {

/\* Hanndle Pushbutton 1

Acknowledge interupt \*/

SET\_BIT(EXTI->PR, EXTI\_PR\_PR1);

count\_two.direction = UP;

if (LED9) {

GPIOC->BSRR = GPIO\_BSRR\_BR\_9;

LED9 = 0;

} else {

GPIOC->BSRR = GPIO\_BSRR\_BS\_9; //Set PC9=1 to turn ON blue LED

LED9 = 1;

}

//GPIOC->BSRR = GPIO\_BSRR\_BR\_8; //Reset PC8=0 to turn OFF blue LED

small\_delay();

NVIC\_ClearPendingIRQ(EXTI1\_IRQn);

}

void counter\_factory(counter\* c) {

c->count = 0;

c->direction = UP;

c->disable = FALSE;

}

int main() {

setup\_pins();

setup\_interupts();

counter\_factory(&count\_one);

counter\_factory(&count\_two);

GPIOC->BSRR = GPIO\_BSRR\_BS\_9;

LED8 = 0;

LED9 = 1;

\_\_enable\_irq();

while(1) {

delay();

count\_two.disable = ~count\_two.disable;

count(&count\_one);

count(&count\_two);

update\_leds();

}

}