### *#include "stm32f10x\_it.h"*

### *#include "exs03.h" /\* Expkits EXS-03 Development Kit Header File \*/*Ex03.c

### *#include "delay.h"*

### int main()

### {

int i=0;

EXS03\_initGPIO(void)

GPIO\_InitTypeDef GPIO\_InitStructure;

/\* GPIOA-GPIOC Periph clock enable \*/

#ifdef STM32F10X\_MD

### /\* GPIOA-GPIOC Periph clock enable \*/

### RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE); // LED, BUTTON GPIO

### RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB, ENABLE); // LCD GPIO

### RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOC, ENABLE); // LED, BUTTON, BUZZER GPIO

### #elif STM32F2XX

### RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOA, ENABLE); // LED, BUTTON GPIO

### RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOB, ENABLE); // LCD GPIO

### RCC\_AHB1PeriphClockCmd(RCC\_AHB1Periph\_GPIOC, ENABLE); // LED, BUTTON, BUZZER GPIO

### #endif

### /\*

### \*\*\*\*\*\*\*\*\*LED1->PA0, LED2->PA1, LED3-> PC2, LED4 -> PC3 \*\*\*\*\*\*\*\*\*

### \*/

### #ifdef STM32F2XX

### GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_0 | GPIO\_Pin\_1 ;

### GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

### GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_OUT; //Output

### GPIO\_InitStructure.GPIO\_OType = GPIO\_OType\_PP; // ve PushPull

### GPIO\_InitStructure.GPIO\_PuPd = GPIO\_PuPd\_UP;

### GPIO\_Init(GPIOA, &GPIO\_InitStructure);

### GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_2 | GPIO\_Pin\_3;

### GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

### GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_OUT; //Output

### GPIO\_InitStructure.GPIO\_OType = GPIO\_OType\_PP; // ve PushPull

### GPIO\_InitStructure.GPIO\_PuPd = GPIO\_PuPd\_UP;

### GPIO\_Init(GPIOC, &GPIO\_InitStructure);

### }

### #define HSE\_VALUE ((uint32\_t)8000000)

### delay\_init(void)

### RCC\_ClocksTypeDef RCC\_ClocksStatus;

### RCC\_GetClocksFreq(&RCC\_ClocksStatus);

### SysTick\_CLKSourceConfig(SysTick\_CLKSource\_HCLK\_Div8); /\* HCLK/8 \*/

### SysTick\_ITConfig(DISABLE);

### delay\_fac\_us = RCC\_ClocksStatus.HCLK\_Frequency / 8000000; // 8Mhz Kristal için

### delay\_fac\_ms = RCC\_ClocksStatus.HCLK\_Frequency / 8000; // 8Mhz Kristal için

### while(1)

### {

switch(i)

{

case 0:

GPIO\_SetBits(GPIOA , GPIO\_Pin\_0);

GPIO\_ResetBits(GPIOA , GPIO\_Pin\_1);

GPIO\_ResetBits(GPIOC , GPIO\_Pin\_2);

GPIO\_ResetBits(GPIOC , GPIO\_Pin\_3);

break;

### /\*\*

### \* @brief Sets the selected data port bits.

### \* @param GPIOx: where x can be (A..G) to select the GPIO peripheral.

### \* @param GPIO\_Pin: specifies the port bits to be written.

### \* This parameter can be any combination of GPIO\_Pin\_x where x can be (0..15).

### \* @retval None

### \*/

### void GPIO\_SetBits(GPIO\_TypeDef\* GPIOx, uint16\_t GPIO\_Pin)

### {

### /\* Check the parameters \*/

### assert\_param(IS\_GPIO\_ALL\_PERIPH(GPIOx));

### assert\_param(IS\_GPIO\_PIN(GPIO\_Pin));

### GPIOx->BSRR = GPIO\_Pin;

### }

}

### delay\_ms(250 🡪uint16\_t nms)

### uint32\_t temp = delay\_fac\_ms \* nms;

### if (temp > 0x00ffffff)

### {

### temp = 0x00ffffff;

### }

### SysTick\_SetReload(temp);

### SysTick\_CounterCmd(SysTick\_Counter\_Clear);

### SysTick\_CounterCmd(SysTick\_Counter\_Enable);

### do

### {

### Status = SysTick\_GetFlagStatus(SysTick\_FLAG\_COUNT);

### }while (Status != SET);

### SysTick\_CounterCmd(SysTick\_Counter\_Disable);

### SysTick\_CounterCmd(SysTick\_Counter\_Clear);

}

i++;

if(i==4)

i=0;

}