федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

|  |  |
| --- | --- |
| / | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | Информатики и систем управления |
|  |  |
| КАФЕДРА | Проектирования и технологии производства ЭА |

**Отчет по лабораторной работе:**

по курсу          Проектирование микропроцессорных систем

на тему        Пять режимов работы светодиода с помощью таймеров и прерываний

|  |  |  |
| --- | --- | --- |
| Студент |  | Ж.О. Исроилов |
|  | (Подпись, дата) | (И.О.Фамилия) |
| Консультант |  | В.В. Леонидов |
|  | (Подпись, дата) | (И.О.Фамилия) |
|  |  |  |
|  |  |  |
| Отметки о сдачи лабораторной работы №2: |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Москва, 2021

**Исходные элементы.**

Отладочный комплект, светодиод, две кнопки.

**Задание.**

Необходимо запрограммировать микроконтроллер таким образом, чтобы при нажатии на одну кнопку светодиод горел ярче, а при нажатии на другую тусклее, при этом возможно только пять яркостей свечения светодиода. При последующие нажатиях схема никак не должна реагировать. Яркость регулируется с помощью таймеров, кнопки обрабатываются с помощью прерываний. Реализовать данную задачу на языках программирования Си.

**Реализация программы на Cи**

Файл main.c

**#include "main.h"**

**int w=2;**

**bool led\_flag=false;**

**void TIM2\_IRQHandler(void)**

**{**

**TIM2->SR &= ~TIM\_SR\_UIF; //Сброс флага переполнения**

**if((w>0)&&(w<12))**

**{**

**if(led\_flag)**

**{**

**LED\_SWAP();**

**TIM2->ARR = (12-w+1)-1;**

**led\_flag = false;**

**}**

**else**

**{**

**LED\_SWAP();**

**TIM2->ARR = (w+1)-1;**

**led\_flag = true;**

**}**

**}**

**else if(w==12)**

**{**

**LED\_ON();**

**led\_flag = true;**

**}**

**else**

**{**

**LED\_OFF();**

**led\_flag = false;**

**}**

**//LED\_SWAP();**

**}**

**void EXTI0\_IRQHandler(void)**

**{**

**if (EXTI->PR & EXTI\_PR\_PR0) // нас интересует EXTI0**

**{**

**EXTI->PR |= EXTI\_PR\_PR0;**

**delay(10000); //Задержка для защиты от дребезга контактов**

**w+=2;**

**if(w>12)**

**{**

**w-=2;**

**}**

**TIM2->CNT=0;**

**//TIM2->CR1 ^= TIM\_CR1\_CEN; //Инвертируем состояние таймера**

**}**

**}**

**void EXTI1\_IRQHandler(void)**

**{**

**// нужно проверить, кто из них его вызвал.**

**if (EXTI->PR & EXTI\_PR\_PR1) // нас интересует EXTI13**

**{**

**EXTI->PR |= EXTI\_PR\_PR1;**

**delay(10000); //Задержка для защиты от дребезга контактов**

**w-=2;**

**if(w<0)**

**{**

**w+=2;**

**}**

**TIM2->CNT=0;**

**//TIM2->CR1 ^= TIM\_CR1\_CEN; //Инвертируем состояние таймера**

**}**

**}**

**void initPorts(void)**

**{**

**RCC->APB2ENR |= RCC\_APB2ENR\_IOPBEN; //включить тактирование GPIOB**

**//очистка полей**

**GPIOB->CRL &= ~(GPIO\_CRL\_CNF0 | GPIO\_CRL\_MODE0);**

**//и конфигурация**

**GPIOB->CRL |= GPIO\_CRL\_MODE0\_1; //PB0, выход 2МГц**

**}**

**void initButton(void)**

**{**

**//Включить тактирование порта GPIOA и альтернативных функций**

**RCC->APB2ENR |= RCC\_APB2ENR\_IOPAEN | RCC\_APB2ENR\_AFIOEN;**

**//PA0**

**GPIOA->CRL &= ~(GPIO\_CRL\_MODE0 | GPIO\_CRL\_CNF0);**

**GPIOA->CRL |= GPIO\_CRL\_CNF0\_1;**

**GPIOA->BSRR |= GPIO\_BSRR\_BS0;**

**//PA1**

**GPIOA->CRL &= ~(GPIO\_CRL\_MODE1 | GPIO\_CRL\_CNF1);**

**GPIOA->CRL |= GPIO\_CRL\_CNF1\_1;**

**GPIOA->BSRR |= GPIO\_BSRR\_BS1;**

**//AFIO**

**AFIO->EXTICR[0] |= (AFIO\_EXTICR1\_EXTI0\_PA | AFIO\_EXTICR1\_EXTI1\_PA);**

**EXTI->FTSR |= (EXTI\_FTSR\_TR0 | EXTI\_FTSR\_TR1);**

**EXTI->IMR |= (EXTI\_IMR\_MR0 | EXTI\_IMR\_MR1);**

**//NVIC PA0**

**NVIC\_EnableIRQ(EXTI0\_IRQn);**

**NVIC\_SetPriority(EXTI0\_IRQn, 0);**

**//NVIC PA1**

**NVIC\_EnableIRQ(EXTI1\_IRQn);**

**NVIC\_SetPriority(EXTI1\_IRQn, 0);**

**}**

**void initTIM2(void)**

**{**

**RCC->APB1ENR |= RCC\_APB1ENR\_TIM2EN; //Включить тактирование TIM2**

**//Частота APB1 для таймеров = APB1Clk \* 2 = 32МГц \* 2 = 64МГц**

**TIM2->PSC = 64000-1; //Предделитель частоты (64МГц/64000 = 1кГц)**

**TIM2->ARR = (500)-1; //Модуль счёта таймера (1кГц/1000 = 1с)**

**TIM2->DIER |= TIM\_DIER\_UIE; //Разрешить прерывание по переполнению таймера**

**TIM2->CR1 |= TIM\_CR1\_CEN; //Включить таймер**

**if((w>0)&&(w<12))**

**{**

**TIM2->CNT=0;**

**TIM2->ARR = (w+1)-1; //Модуль счёта таймера (1кГц/1000 = 1с)**

**LED\_ON();**

**led\_flag = true;**

**}**

**else if(w==12)**

**{**

**LED\_ON();**

**led\_flag=true;**

**}**

**else**

**{**

**LED\_OFF();**

**led\_flag=false;**

**}**

**NVIC\_EnableIRQ(TIM2\_IRQn); //Рарзрешить прерывание от TIM2**

**NVIC\_SetPriority(TIM2\_IRQn, 1); //Выставляем приоритет**

**}**

**void initClk(void)**

**{**

**// Enable HSI**

**RCC->CR |= RCC\_CR\_HSION;**

**while(!(RCC->CR & RCC\_CR\_HSIRDY)){};**

**// Enable Prefetch Buffer**

**FLASH->ACR |= FLASH\_ACR\_PRFTBE;**

**// Flash 2 wait state**

**FLASH->ACR &= ~FLASH\_ACR\_LATENCY;**

**FLASH->ACR |= FLASH\_ACR\_LATENCY\_2;**

**// HCLK = SYSCLK**

**RCC->CFGR |= RCC\_CFGR\_HPRE\_DIV1;**

**// PCLK2 = HCLK**

**RCC->CFGR |= RCC\_CFGR\_PPRE2\_DIV1;**

**// PCLK1 = HCLK**

**RCC->CFGR |= RCC\_CFGR\_PPRE1\_DIV2;**

**// PLL configuration: PLLCLK = HSI/2 \* 16 = 64 MHz**

**RCC->CFGR &= ~RCC\_CFGR\_PLLSRC;**

**RCC->CFGR |= RCC\_CFGR\_PLLMULL16;**

**// Enable PLL**

**RCC->CR |= RCC\_CR\_PLLON;**

**// Wait till PLL is ready**

**while((RCC->CR & RCC\_CR\_PLLRDY) == 0) {};**

**// Select PLL as system clock source**

**RCC->CFGR &= ~RCC\_CFGR\_SW;**

**RCC->CFGR |= RCC\_CFGR\_SW\_PLL;**

**// Wait till PLL is used as system clock source**

**while ((RCC->CFGR & RCC\_CFGR\_SWS) != RCC\_CFGR\_SWS\_PLL){};**

**}**

**int main(void)**

**{**

**/\*Инициализации всякие\*/**

**initClk();**

**initPorts();**

**initButton();**

**initTIM2();**

**/\*Основной цикл\*/**

**while(true){};**

**}**

**void delay(uint32\_t takts)**

**{**

**for (uint32\_t i = 0; i < takts; i++) {};**

**}**

Файл main.h

**#ifndef** \_\_MAIN\_H

**#define** \_\_MAIN\_H

**#include** "stm32f1xx.h"

**#include** "stdbool.h"

**#define** DELAY\_VAL 1000000

**#define** LED\_ON() GPIOB->BSRR = GPIO\_BSRR\_BS0

**#define** LED\_OFF() GPIOB->BSRR = GPIO\_BSRR\_BR0

**#define** LED\_SWAP() GPIOB->ODR ^= GPIO\_ODR\_ODR0

**void** **initClk**(**void**);

**void** **initPorts**(**void**);

**void** **initButton**(**void**);

**void** **initTIM2**(**void**);

**void** **delay**(uint32\_t takts);

**#endif**

**Графики**

Полученные графики соответствуют каждому из пяти режимов яркости светодиода

 

