Университет ИТМО

Лабораторная работа №2
по предмету
“Встроенные системы”

Выполнил:
Студент группы
P3410
Глушков Дима

Санкт-Петербург,
2020 г.

Введение:

Данная лабораторная работа предназначена для получения практических навыков работы с инкрементируемыми таймерами TIM6 и TIM7 и системой прерываний на виртуальном стенде на базе микроконтроллера STM32F407c использованием HAL функций организовать изменение скорости изменения анимации согласно варианту.

Описание теоретической части:

Для выполнения лабораторной работы используем информацию из таблицы сравнения таймеров МК STM32 и таблицы с адресами переходов МК при вызове обработчика прерываний.

Timer feature comparison

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/ compare channels	Complementary output	Max interface clock (MHz)	Max timer clock (MHz)
Advanced -control	TIM1, TIM8	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	168
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	168
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	168
	TIM12	16-bit	Up	Any integer between 1 and 65536	No	2	No	42	84
	TIM13, TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	42	84
Basic	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	42	84

Рис. 1. Таблица сравнения таймеров МК STM32F407VGx.

Position	Priority	Type of priority	Acronym	Description	Address
40	47	settable	EXTI15_10	EXTI Line[15:10] interrupts	0x0000_00E0
41	48	settable	RTCAlarm	RTC alarm through EXTI line interrupt	0x0000_00E4
42	49	settable	OTG_FS_WKUP	USB On-The-Go FS Wakeup through EXTI line interrupt	0x0000_00E8
-	-	-	-	Reserved	0x0000_00EC - 0x0000_0104
50	57	settable	TIM5	TIM5 global interrupt	0x0000_0108
51	58	settable	SPI3	SPI3 global interrupt	0x0000_010C
52	59	settable	UART4	UART4 global interrupt	0x0000_0110
53	60	settable	UART5	UART5 global interrupt	0x0000_0114
54	61	settable	TIM6	TIM6 global interrupt	0x0000_0118
55	62	settable	TIM7	TIM7 global interrupt	0x0000_011C
56	63	settable	DMA2_Channel1	DMA2 Channel1 global interrupt	0x0000_0120
57	64	settable	DMA2_Channel2	DMA2 Channel2 global interrupt	0x0000_0124
58	65	settable	DMA2_Channel3	DMA2 Channel3 global interrupt	0x0000_0128
59	66	settable	DMA2_Channel4	DMA2 Channel4 global interrupt	0x0000_012C

Рис. 2. Фрагмент таблицы с адресами переходов МК при вызове обработчика прерываний.

Регистры стенда, используемые при выполнении ЛР:

Регистр	Описание
TIMx_CR1	регистр управления
TIMx_DIER	регистр разрешения прерываний
TIMx_CNT	счетный регистр
TIMx_PSC	регистр делителя тактового сигнала
TIMx_ARR	регистр, хранящий значение перезагрузки

Функции, которые могут потребоваться при выполнении лабораторной работы:

```
void WRITE_REG(uint reg_address, uint value);
uint READ_REG(uint reg_address);
void registerTIM6_IRQHandler(void* irqHandler);
void registerTIM7_IRQHandler(void* irqHandler);
__enable_irq();
__disable_irq();
```

Задание:
На светодиодные индикаторы LED1 … LED8 должна выводиться анимация согласно варианту задания.

Скорость анимации задается с помощью переключателей SW. Если на переключателях выставлен код 0x0, то кадры анимации меняются каждые 500 мс. С увеличением значения, выставленном на переключателях SW анимация замедляется на T мс. Значение T задается вариантом задания.

Например, если по варианту задано, что T = 100 мс, это означает, что при установке переключателей в состояние SW = 0x1, кадры начинают меняться каждые 500+1*100= 600 мс, если SW = 0x5, то кадры начинают меняться каждые 500+5*100 = 1000 мс и т.д.

Все задержки должны быть реализованы с использованием прерываний от базовых таймеров TIM6 или TIM7.
T = 250 мс

Кадр	LED1	LED2	LED3	LED4	LED5	LED6	LED7	LED8
1	■							
2	■	■						
3	■	■	■					
4	■	■	■	■				
5	■	■	■	■	■			
6	■	■	■	■	■	■		
7	■	■	■	■	■	■	■	
8	■	■	■	■	■	■	■	■
9	■	■	■	■	■	■	■	
10	■	■	■	■	■	■		
11	■	■	■	■	■			
12	■	■	■	■				
13	■	■	■					
14	■	■						

Исходный код программы:

```
#include "hal.h"

int frame = 0;
int max_frame = 13;
int init_speed = 500;
int speed = 500;
int step = 250;
int state[4];

int led_num[] = {
    GPIO_PIN_3,
    GPIO_PIN_4,
    GPIO_PIN_5,
    GPIO_PIN_6,
    GPIO_PIN_8,
    GPIO_PIN_9,
    GPIO_PIN_11,
    GPIO_PIN_12
};
unsigned int switch_num[] = {
    GPIO_PIN_4,
    GPIO_PIN_8,
    GPIO_PIN_10,
    GPIO_PIN_12
};

void set_state()
{
    for (int i = 0; i < 4; i++)
          state[i] = HAL_GPIO_ReadPin(GPIOE, switch_num[i]) == GPIO_PIN_SET ? 1 : 0;
    int general_state = state[0] + state[1] * 2 + state[2] * 4 + state[3] * 8;
    speed = init_speed + general_state * step;
    WRITE_REG(TIM6_ARR, speed);
    WRITE_REG(TIM7_ARR, speed);
}

void TIM6_IRQ_Handler()
{
    if (frame <= 7)
          HAL_GPIO_WritePin(GPIOD, led_num[frame], GPIO_PIN_SET);
}

void TIM7_IRQ_Handler()
{
    if (frame > 7)
          HAL_GPIO_WritePin(GPIOD, led_num[max_frame - frame + 2], GPIO_PIN_RESET);
    set_state();
    if (frame++ > max_frame)
          frame = 0;
}

int umain()
{
    set_state();

    registerTIM6_IRQHandler(TIM6_IRQ_Handler);
    registerTIM7_IRQHandler(TIM7_IRQ_Handler);
```

```
    __enable_irq();

    WRITE_REG(TIM6_ARR, speed);
    WRITE_REG(TIM6_DIER, TIM_DIER_UIE);
    WRITE_REG(TIM6_PSC, 0);

    WRITE_REG(TIM7_ARR, speed);
    WRITE_REG(TIM7_DIER, TIM_DIER_UIE);
    WRITE_REG(TIM7_PSC, 1);

    WRITE_REG(TIM6_CR1, TIM_CR1_CEN);
    WRITE_REG(TIM7_CR1, TIM_CR1_CEN);

    return 0;
}
```