Лабораторная работа №2

Цель работы:

Получение навыков настройки тактовых частот микроконтроллера.

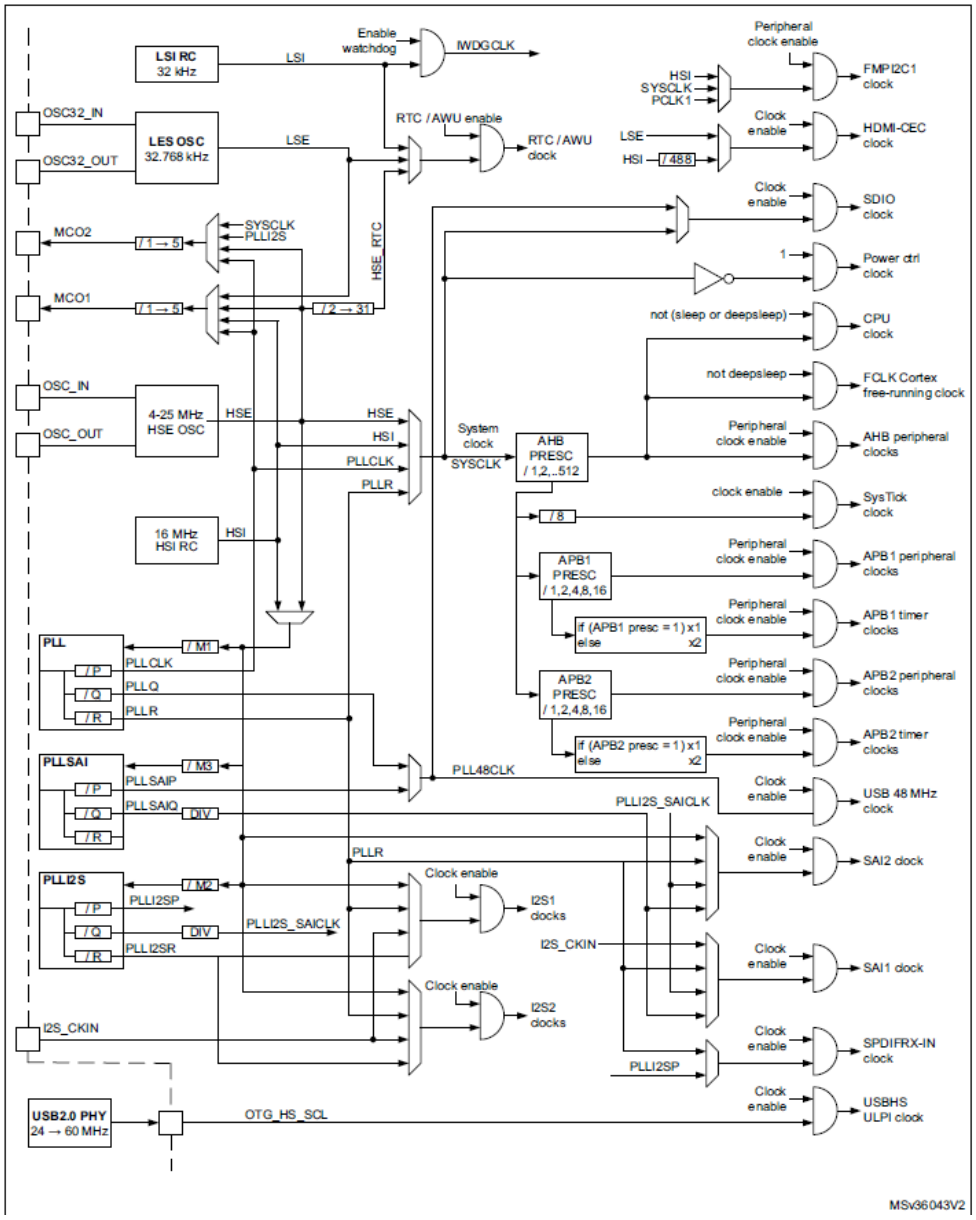
Программное обеспечение:

STM32CubeIDE.

Общие сведения:

Общая схема тактирования микроконтроллера:

Figure 14. Clock tree



Работа с библиотеками

Вся периферия микроконтроллера и все регистры определены в библиотеках аналогично тому, как требовалось сделать это в первой работе.

Микроконтроллер имеет по несколько однообразных периферийных устройств, например 8 USART, 11 таймеров, 3 SPI и тд. Каждое устройство имеет свой номер (исключение составляет GPIO, они нумеруются буквами). Таким образом, работа с тем или иным периферийным устройством микроконтроллера, всегда необходимо указывать его сокращенное название и номер!

Для обращения к регистрам периферии вначале пишется имя периферии, затем указывается имя регистра! Например – «$USART1 -> DR$» - обращение к регистру данных периферии USART1. Имена всех регистров совпадают с наименованиями в документации, например Управляющей регистр 1(Control Register 1) любого таймера будет иметь имя CR1, таким образом обращение к нему будет иметь вид: «$TIMx -> CR1$». Работа с регистрами описана в предыдущей работе.

Offset	Register
0x28	TIMx_PSC
	Reset value
0x2C	TIMx_ARR
	Reset value
0x30	Reserved
0x34	TIMx_CCR1
	Reset value
0x38	TIMx_CCR2
	Reset value

```
87      TIM4->PSC = A
88      TIM4->ARR = 1
89      TIM4->CCER =
90
91
92
93      TIM4->CCMR1 =
94
95
96
97      TIM4->CCMR2 =
98
99
.00
.01     TIM4->CCR1 =
.02      TIM4->CCR2 =
```

Описание регистров в документации и работа с ними в программе.

Кроме того, разнообразные режимы работы/настройки периферии так же определены в библиотеках. Например: «$ADC1 -> CR1 = ADC\_CR1\_SCAN;$» - **ADC_CR1_SCAN** – включение сканирующего режима АЦП в управляющем регистре 1. Все определения формируются следующим образом – Переферия_Регистр_Бит. Каждое такое определение содержит в себе несколько дополнительных определений, которые также могут быть использованы в программе. Включение определенного режима работы означает запись «1» (или нескольких «1») в определенную область регистра. Например:

17.4.1 TIMx control register 1 (TIMx_CR1)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 4 **DIR**: Direction

0: Counter used as upcounter

1: Counter used as downcounter

Из документации мы видим, что для включения режима счета вниз таймера необходимо записать «1» в четвертый бит Управляющего регистра 1. Самое простое действие, для выполнений этой операции является сдвиг «1» на требуемое число бит, в

данном случае, на 4. В итоге получаем такую запись: «1 ≪ 4», которую можно использовать для настройки таймера. По такому принципы сделаны все определения настроек регистров в библиотеке. Для каждой настройки имеется определение отвечающее за положение данной настройки в регистре, с кодовым словом «$\_Pos$» в конце; маску содержащую «1» во всех битах настройки, с кодовым словом «$\_Msk$» в конце; и конечную строку для включения требуемой настройки. Возвращаясь к предыдущему примеру:

```
#define TIM_CR1_DIR_Pos                 (4U)
#define TIM_CR1_DIR_Msk                 (0x1UL << TIM_CR1_DIR_Pos)
#define TIM_CR1_DIR                     TIM_CR1_DIR_Msk
```

«TIM_CR1_DIR» - Включение режима счета вниз для таймера;
«$TIM\_CR1\_DIR\_Msk$» - Маска данного режима;
«$TIM\_CR1\_DIR\_Pos$» - Положение данного режима в регистре;

Если для включения определенного режима работы периферии требуется несколько бит, то определения будут сделаны аналогичным образом, плюс определение записи единицы в каждую часть данного режима. Например, выбор канала DMA:

9.5.5 DMA stream x configuration register (DMA_SxCR) (x = 0..7)

This register is used to configure the concerned stream.

Address offset: 0x10 + 0x18 * stream number

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CHSEL[2:0]			MBURST [1:0]		PBURST[1:0]		Res.	CT	DBM	PL[1:0]	
				rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw

Bits 27:25 **CHSEL[2:0]**: channel selection
These bits are set and cleared by software.
000: channel 0 selected
001: channel 1 selected
010: channel 2 selected
011: channel 3 selected
100: channel 4 selected
101: channel 5 selected
110: channel 6 selected
111: channel 7 selected

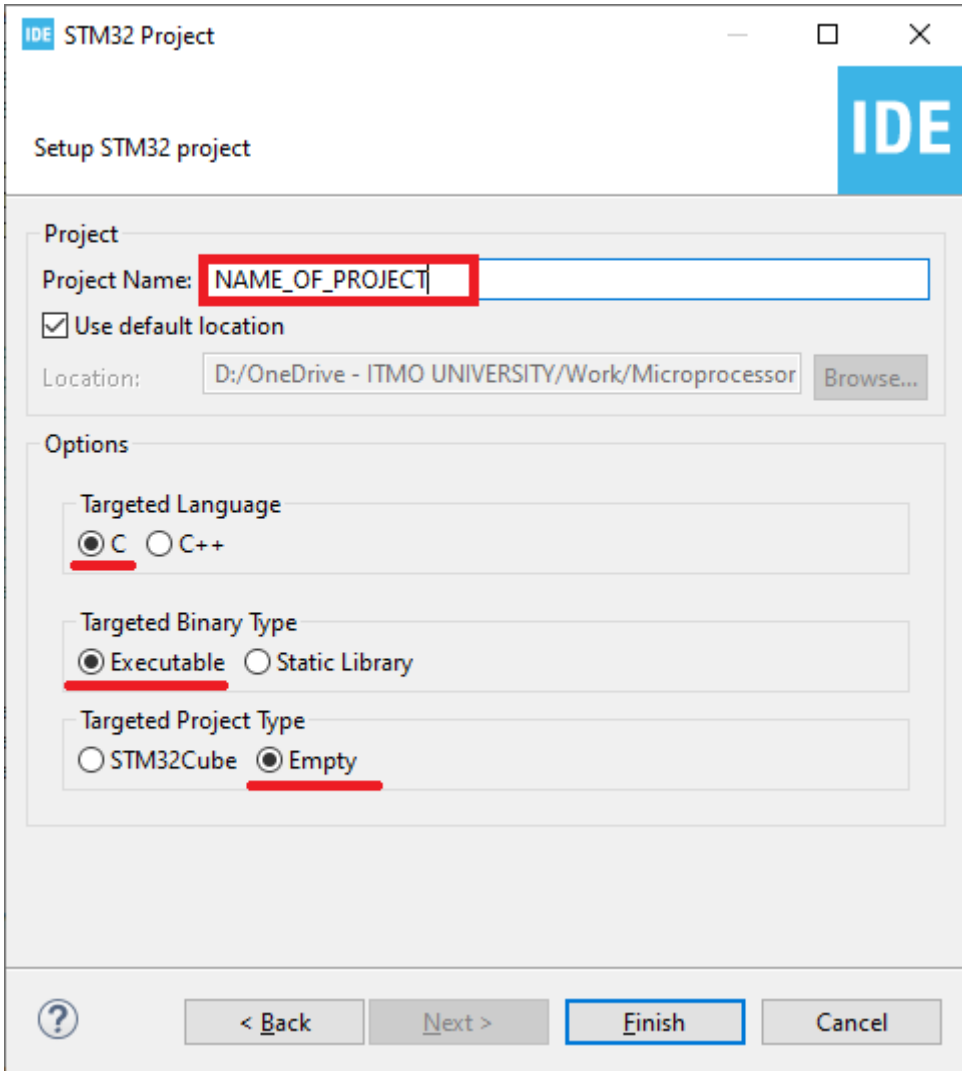
Для выбора канала DMA требуется 3 бита, соответственно определение в библиотеке будет выглядеть следующим образом:

```
#define DMA_SxCR_CHSEL_Pos              (25U)
#define DMA_SxCR_CHSEL_Msk              (0x7UL << DMA_SxCR_CHSEL_Pos)
#define DMA_SxCR_CHSEL                  DMA_SxCR_CHSEL_Msk
#define DMA_SxCR_CHSEL_0                0x02000000U
#define DMA_SxCR_CHSEL_1                0x04000000U
#define DMA_SxCR_CHSEL_2                0x08000000U
```

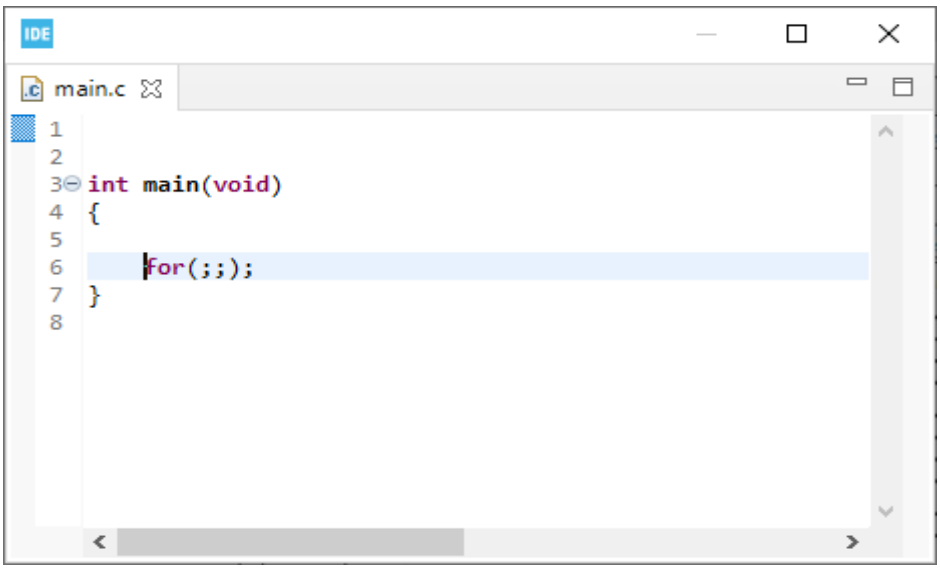
«$DMA\_SxCR\_CHSEL\_Pos$» - Положение данного режима в регистре;
«$DMA\_SxCR\_CHSEL\_Msk$» - Маска данного режима (Содержит «1» во всех 3х битах режима);
«$DMA\_SxCR\_CHSEL\_0$» - Запись «1» в нулевой бит данного режима;
«$DMA\_SxCR\_CHSEL\_1$» - Запись «1» в первый бит данного режима;
«$DMA\_SxCR\_CHSEL\_2$» - Запись «1» в второй бит данного режима;

Порядок выполнения работы.

1. Запустите STM32CubeIDE, в качестве workspace выберете папку для лабораторной работы. Создайте новый проект «Start new STM32 project», в поле выбора микроконтроллера выберете STM32F446RE. Нажмите Next, задайте имя проекта, выберете язык – *C*, типы выходного файла – *Executable*, тип проекта – *Empty*; нажмите «Finish». Обратите внимание, в пути и названии проекта не должно быть русских букв!



2. Откройте основной файл программы – «Src/main.c». Удалите весь код, кроме функции «main».



3. Скопируйте папку с библиотеками в workspace, папку выбранную на 1 шаге работы. Откройте настройки проекта – «Project->Properties», перейдите в «C/C++ General->Path and Symbols». Добавьте в проект 2 библиотеки во вкладке «Includes»:

- ..\CMSIS\Include;
- ..\CMSIS\Device\ST\STM32F4xx\Include.

Для использования библиотек, подключите библиотеку «stm32f446xx.h».

4. Используя информацию из лекции, документацию на микроконтроллер и схему стенда, выполнить задание в соответствии с вариантом.

Задание:

1. Рассчитать и определить(define) учитывая все ограничения:
   - значения делителей AHB1, APB1, APB2, APB1_timers, APB2_timers;
   - значения коэффициентов PLL;
   - значение частоты HCLK на основе значений коэффициентов PLL;
   - значения частот всех шин, на основе значений делителей;
   - частоту системного таймера;
   - значение делителя системного таймера.
2. Настроить задержку для памяти.
3. Настроить частоту работы микроконтроллера в соответствии с вариантом, используя определенные заранее значения коэффициентов.
4. Запустить системный таймер с требуемой частотой работы.
5. В прерывании системного таймера изменять светодиода с частотой 1 Гц.

Содержание отчета:

- Титульный лист.
- Цель работы.
- Листинг разработанной программы.
- Вывод.

Варианты:

№ варианта	Частота CPU, МГц	Частота прерывания от системного таймера, кГЦ
1	100	5
2	110	10
3	120	20
4	130	9
5	140	15
6	150	8
7	160	17
8	170	7
9	180	25
10	105	6