Университет ИТМО

Лабораторная работа № 2

по дисциплине «Встроенные системы»

Студент:

P3410 Нгу Фыонг Ань

Преподаватель:

Санкт-Петербург

2020 г.

**Цель работы** – освоение работы с таймерами микроконтроллера в режиме ШИМ.

**ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ**

1. Изучите структуру стенда SDK 1.1M;
2. Скачать репозиторий с проектом по ссылке - https://github.com/lmtspbru/SDK\_cLAB;
3. Открыть проект в STM32CubeIDE;
4. Произвести настройку векторов прерываний и компоновщика;
5. Написать программу с помощью фреймовка HAL, работающую с таймером из номера варианта. Программа должна реализовывать генерацию периодического сигнала с заданной частотой и скважностью. Генерируемый периодический сигнал необходимо выводить в трассировочный график.

Дополнительно:

* 1. Генерация периодического сигнала должна происходить с помощью аппаратных возможностей контроллера, а именно на основе таймера в PWM режиме. Для этого стоит рассмотреть настройку таймера в PWM режиме и функцию HAL HAL\_TIM\_PWM\_Start\_IT. Обратным вызовом функции будет являться функции HAL\_TIM\_PWM\_PulseFinishedCallback.

1. Собрать проект в Debug режиме;
2. Загрузить сгенерированный бинарный файл на ITMO.cLAB для проверки правильности выполнения задания.
3. По итогам работы написать и защитить отчет. Отчет должен содержать описание теоретической и практической частей, а также основной код программы. В процессе защиты работы требуется демонстрация реализованного функционала, поэтому .bin файл для загрузки на ITMO.cLAB должен быть готов к началу сдачи.

|  |  |  |  |
| --- | --- | --- | --- |
| **№ Варианта** | **Задание** | | |
| Таймер | Частота (Гц) | Скважность (%) |
| 2 | TIM5 | 200 | 20 |

Генерация периодического сигнала: T = 1/f = 1000/200 = 5 ms

Ширина импульса: Tw = T \* S = 5 \* 20% = 1 ms

1. **Исходный код**

**int** **main**(**void**)

{

/\* USER CODE BEGIN 1 \*/

/\* USER CODE END 1 \*/

/\* MCU Configuration--------------------------------------------------------\*/

/\* Reset of all peripherals, Initializes the Flash interface and the Systick. \*/

HAL\_Init();

/\* USER CODE BEGIN Init \*/

/\* USER CODE END Init \*/

/\* Configure the system clock \*/

SystemClock\_Config();

/\* USER CODE BEGIN SysInit \*/

/\* USER CODE END SysInit \*/

/\* Initialize all configured peripherals \*/

MX\_GPIO\_Init();

MX\_IWDG\_Init();

MX\_I2C1\_Init();

MX\_UART4\_Init();

MX\_UART5\_Init();

MX\_USART1\_UART\_Init();

MX\_USART2\_UART\_Init();

MX\_USART3\_UART\_Init();

MX\_USART6\_UART\_Init();

MX\_I2C2\_Init();

MX\_I2C3\_Init();

MX\_TIM2\_Init();

MX\_TIM9\_Init();

MX\_TIM5\_Init();

/\* USER CODE BEGIN 2 \*/

/\* Do not remove this code below \*/

MX\_TRACE\_Init();

SDK\_TRACE\_Start();

HAL\_TIM\_Base\_Start\_IT(&htim5);

HAL\_TIM\_PWM\_Start\_IT(&htim5, TIM\_CHANNEL\_1);

i2c\_detect();

HAL\_TIM\_PWM\_Stop\_IT(&htim5, TIM\_CHANNEL\_1);

/\* Place your code before here \*/

/\* Do not remove this code below \*/

SDK\_TRACE\_Stop();

/\* Do not remove this code from above \*/

/\* Place your code before here \*/

/\* Do not remove this code below \*/

SDK\_TRACE\_Stop();

/\* Do not remove this code from above \*/

/\* USER CODE END 2 \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

**while** (1)

{

/\* USER CODE END WHILE \*/

/\* USER CODE BEGIN 3 \*/

}

/\* USER CODE END 3 \*/

}

/\* TIM5 init function \*/

**void** **MX\_TIM5\_Init**(**void**)

{

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

TIM\_OC\_InitTypeDef sConfigOC = {0};

htim5.Instance = TIM5;

htim5.Init.Prescaler = 8399;

htim5.Init.CounterMode = TIM\_COUNTERMODE\_UP;

//period

htim5.Init.Period = 50;

htim5.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim5.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

**if** (HAL\_TIM\_Base\_Init(&htim5) != *HAL\_OK*)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

**if** (HAL\_TIM\_ConfigClockSource(&htim5, &sClockSourceConfig) != *HAL\_OK*)

{

Error\_Handler();

}

**if** (HAL\_TIM\_PWM\_Init(&htim5) != *HAL\_OK*)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

**if** (HAL\_TIMEx\_MasterConfigSynchronization(&htim5, &sMasterConfig) != *HAL\_OK*)

{

Error\_Handler();

}

sConfigOC.OCMode = TIM\_OCMODE\_PWM1;

//Duty cycle

sConfigOC.Pulse = 10;

sConfigOC.OCPolarity = TIM\_OCPOLARITY\_HIGH;

sConfigOC.OCFastMode = TIM\_OCFAST\_DISABLE;

**if** (HAL\_TIM\_PWM\_ConfigChannel(&htim5, &sConfigOC, TIM\_CHANNEL\_1) != *HAL\_OK*)

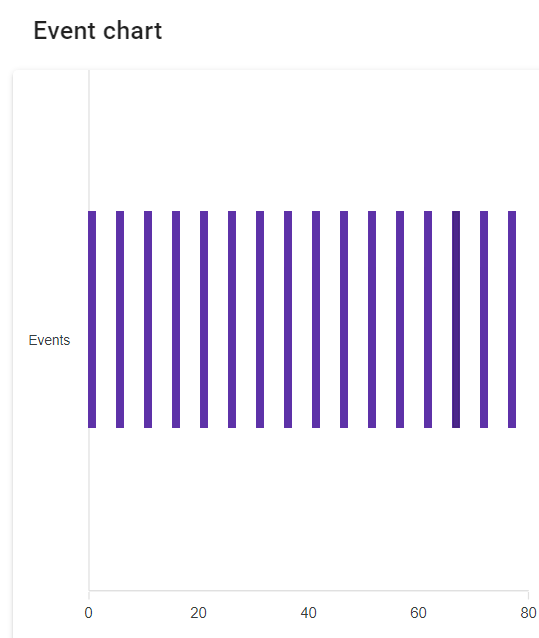
{

Error\_Handler();

}

}

1. **Результат**



1. **Вывод**

Во время работы в лаборатории я ознакомилась с освоением работы с таймерами микроконтроллера в режиме ШИМ.