МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Информационная безопасность систем и технологий»

Отчет

по практической работе №4

на тему «Функции управления таймерами в среде STM32CubeIDE»

Дисциплина: ПМК

Группа: 21ПИ1

Выполнил: Гусев Д. А.

Количество баллов:

Дата сдачи:

Принял: Хворостухин С. П.

2024

1 Цель работы: изучить функции управления таймерами средствами библиотеки HAL в среде разработки STM32CubeIDE.

2 Задание на практическую работу.

2.1 Выполните настройку таймеров в графическом интерфейсе среды разработки STM32CubeIDE в соответствии с вариантом 8, приведенным на рисунке 1. Период работы таймеров и направление счета выберите самостоятельно.

8	*TIM1*	*TIM3*	*TIM2*	*16*

Рисунок 1 — Вариант задания

2.2 Сгенерируйте программный код, соответствующий заданной конфигурации.

2.3 Проанализируйте программный код библиотеки HAL, приведенный в файлах:

– stm32f4xx_it.c - обработчики прерываний таймера;

– stm32f4xx_hal_tim.c - функции работы с таймерами.

2.4 Проанализируйте настройки таймеров в файле main.c.

2.5 Сформируйте описание программного кода:

– обработчиков прерываний таймеров;

– функций инициализации таймеров;

– функций работы с таймерами.

3 Выполнение практической работы:

3.1 Была выполнена настройка внешних линий прерываний в графическом интерфейсе среды разработки STM32CubeIDE в соответствии с вариантом 8. Результат приведен на рисунке 2.

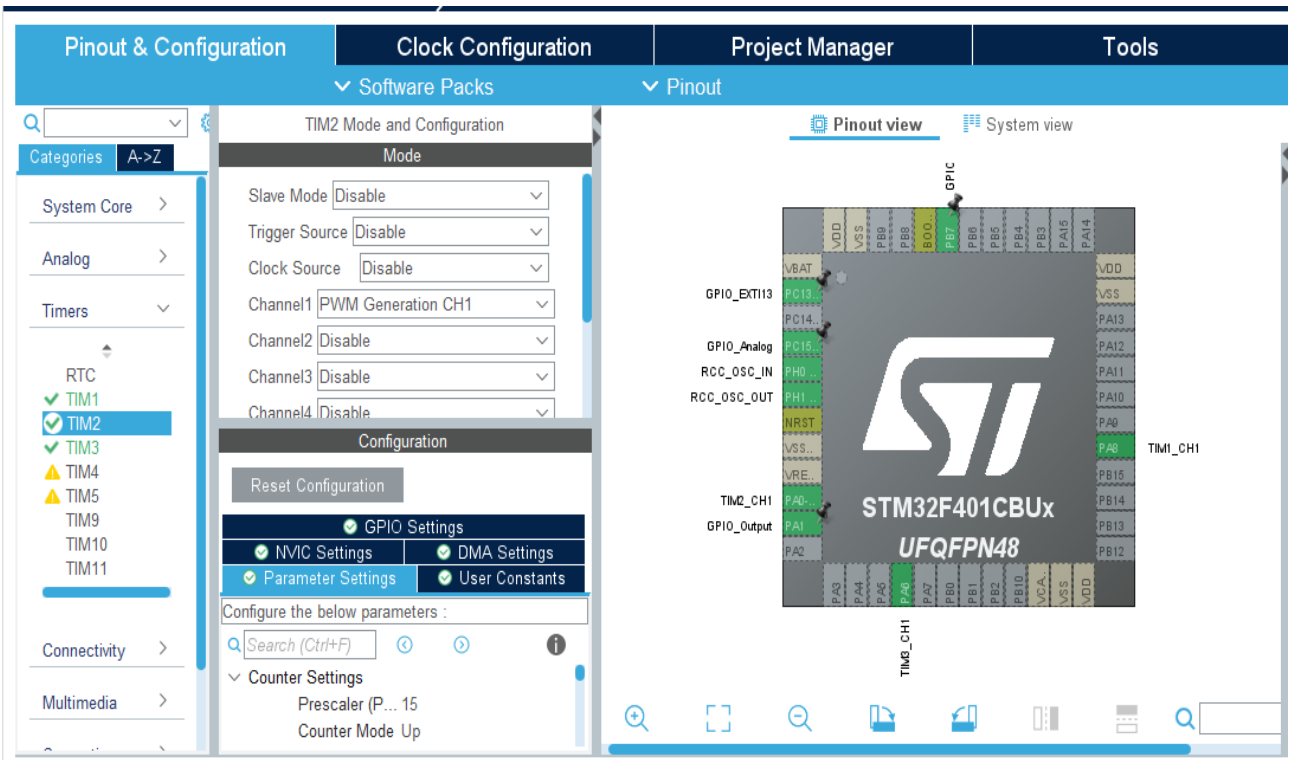

Рисунок 2 - Настройка внешних линий прерываний

3.2 Был сгенерирован программный код, соответствующий заданной конфигурации.

3.3 Был проанализирован программный код библиотеки HAL, приведенный в файлах:

– stm32f4xx_it.c - обработчики прерываний таймера;

– stm32f4xx_hal_tim.c - функции работы с таймерами.

3.3.1 Программный код в файле stm32f4xx_it.c содержит обработчики прерываний для микроконтроллера STM32F4xx. В этом файле содержаться следующие функции:

– NMI_Handler(void) - обрабатывает Non-maskable interrupt (NMI). Содержит бесконечный цикл для блокировки выполнения программы.

– HardFault_Handler(void) - обрабатывает жесткую ошибку (Hard Fault). Содержит бесконечный цикл для блокировки выполнения программы.

– MemManage_Handler(void) - обрабатывает ошибку управления памятью (Memory Management Fault). Содержит бесконечный цикл для блокировки выполнения программы.

– BusFault_Handler(void) - обрабатывает ошибку шины (Bus Fault). Содержит бесконечный цикл для блокировки выполнения программы.

– UsageFault_Handler(void) - обрабатывает ошибку использования (Usage Fault). Содержит бесконечный цикл для блокировки выполнения программы.

– SVC_Handler(void) - обрабатывает вызов службы системы через инструкцию SWI. Содержит комментарии без реального кода обработки прерывания.

– DebugMon_Handler(void) - обрабатывает монитор отладки. содержит комментарии без реального кода обработки прерывания.

– PendSV_Handler(void) - обрабатывает запрос на обслуживание системы. содержит комментарии без реального кода обработки прерывания.

– SysTick_Handler(void) - обрабатывает таймер системного такта. вызывает функцию HAL_IncTick(), которая увеличивает системное время HAL.

3.3.2 Программный код в файле stm32f4xx_hal_tim.c содержит следующие функции для работы с таймерами:

– Функции установки конфигурации для режима сравнения вывода (Output Compare) для соответствующих каналов TIM_OC1_SetConfig(), TIM_OC3_SetConfig(), TIM_OC4_SetConfig().

– Функции для инициализации, деинициализации и управления генерацией базового времени HAL_TIM_Base_MspInit(), HAL_TIM_Base_MspDeInit().

– HAL_TIM_IRQHandler(): Обработчик прерываний TIM.

– HAL_TIM_OC_DelayElapsedCallback(): Обратный вызов для обработки событий задержки порогового значения сравнения вывода TIM (Output Compare).

– HAL_TIM_PeriodElapsedCallback(): Обратный вызов для обработки событий завершения периода TIM.

– HAL_TIM_PWM_PulseFinishedCallback(): Обратный вызов для обработки событий завершения периода для ШИМ сигнала TIM.

– HAL_TIM_OnePulse_Start(), HAL_TIM_PWM_Start(), HAL_TIM_IC_Start(), HAL_TIM_Encoder_Start(): Функции для запуска одноимпульсного режима, ШИМ, ввода захвата и интерфейса энкодера TIM соответственно.

– HAL_TIM_StateTypeDef HAL_TIM_GetState(): Возвращает текущее состояние TIM и его компонентов (Base, Output Compare, PWM, Input Capture, One Pulse, Encoder).

– HAL_TIM_ConfigOCrefClear(), HAL_TIM_ConfigClockSource(), HAL_TIM_PWM_Init(), HAL_TIM_OC_Init(): Функции для настройки различных параметров TIM, таких как очистка сравнения вывода, источник тактового сигнала, инициализация ШИМ и Output Compare для TIM соответственно.

– HAL_TIM_IRQHandler(): Функция обработки прерываний TIM.

3.4 Были проанализированы настройки таймеров в файле main.c. В файле main.c настраиваются три таймера (TIM1, TIM2, TIM3) с помощью функций MX_TIM1_Init(), MX_TIM2_Init(), и MX_TIM3_Init() соответственно.

– TIM1 настраивается в режиме Output Capture (Захват): CounterMode установлен в UP (вверх); Period (период) установлен на 0; OCMode установлен в TIM_OCMODE_FORCED_ACTIVE. Prescaler установлен на 16.

– TIM2 настраивается в режиме PWM: CounterMode установлен в UP (вверх); Prescaler установлен на 16; Period установлен на максимальное значение 4294967295 (это максимальное 32-битное значение); OCMode установлен в TIM_OCMODE_PWM1.

– TIM3 настраивается в режиме Output Compare (Сравнение): CounterMode установлен в UP (вверх); Period установлен на 65535; OCMode установлен в TIM_OCMODE_TIMING; Prescaler установлен на 16;

3.5 Было сформировано описание программного кода:

– обработчиков прерываний таймеров;

– функций инициализации таймеров;

– функций работы с таймерами.

3.5.1 Функция обработки прерываний HAL_TIM_IRQHandler() находится в файле stm32f4xx_hal_tim.c.

Параметры: указатель на структуру TIM_HandleTypeDef, представляющую обработчик таймера TIM.

Возвращаемое значение функции: void - нет возвращаемого значения.

Краткий алгоритм работы:

1) Проверяется флаг события захвата / сравнения (Capture/Compare) для каждого канала (CC1, CC2, CC3, CC4).

2) Если событие произошло и прерывание для этого события разрешено, то выполняются соответствующие действия:

3) Для события захвата: вызывается обработчик захвата или обработчик завершения импульса ШИМ (pulse).

4) Для события сравнения: вызывается обработчик задержки окончания импульса (delay) или обработчик завершения импульса ШИМ.

5) После выполнения соответствующих действий для канала, текущий канал обнуляется.

6) Далее проверяется событие обновления таймера и, если разрешено прерывание этого события, вызывается соответствующий обработчик.

7) Проверяются и обрабатываются другие возможные события, такие как Break input, Trigger detection, Commutation.

3.5.2 Функция обработчика прерывания системного таймера SysTick_Handler находится в файле stm32f4xx_it.c.

Параметры: функция не принимает параметры.

Возвращаемое значение: функция не возвращает значений.

Краткий алгоритм работы:

1) Выполняется пользовательский код, помещенный между комментариями USER CODE BEGIN SysTick_IRQn 0 и USER CODE END SysTick_IRQn 0.

2) Вызывается функция HAL_IncTick(), которая увеличивает значение системного счетчика времени. Это функция из HAL (Hardware Abstraction Layer), предоставляемой производителем микроконтроллера для работы с аппаратными ресурсами.

3) Выполняется пользовательский код, помещенный между комментариями USER CODE BEGIN SysTick_IRQn 1 и USER CODE END SysTick_IRQn 1.

3.5.3 Функции инициализации таймеров MX_TIM1_Init, MX_TIM2_Init, MX_TIM1_Init находятся в файле main.c

Параметры: функции не принимают параметры.

Возвращаемое значение: функции не возвращают значение

Краткий алгоритм работы:

1) Объявление и инициализация структур для конфигурации таймера и его каналов.

2) Конфигурация параметров таймера, таких как делитель частоты (Prescaler), режим счетчика (CounterMode), период (Period) и другие параметры.

3) Инициализация режима работы таймера с использованием соответствующей HAL функции (например, HAL_TIM_OC_Init для TIM1 и HAL_TIM_PWM_Init для TIM2).

4) Конфигурация мастер-режима работы таймера, если необходимо.

5) Конфигурация режима работы канала таймера, например, режима ШИМ (PWM) или других режимов.

3.5.4 Функции установки конфигурации для режима сравнения вывода (Output Compare) для соответствующих каналов TIM_OC1_SetConfig(), TIM_OC3_SetConfig(), TIM_OC4_SetConfig() находятся в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_TypeDef *TIMx - указатель на структуру TIM_TypeDef, представляющую периферийное устройство TIM.

2) TIM_OC_InitTypeDef *OC_Config - указатель на структуру TIM_OC_InitTypeDef, которая содержит параметры конфигурации.

Возвращаемое значение: функции не возвращают значение.

Краткий алгоритм работы:

1) Проверить корректность переданных параметров TIMx и OC_Config.

2) Инициализировать вывод сравнения первого канала таймера TIMx с помощью параметров из структуры OC_Config.

3) Настроить режим работы и регистры сравнения для первого канала таймера TIMx.

4) Применить заданные параметры конфигурации к выводу сравнения первого канала таймера TIMx.

Завершить работу функции.

3.5.5 Функция обратного вызова для обработки событий задержки порогового значения сравнения вывода TIM (Output Compare) HAL_TIM_OC_DelayElapsedCallback() находится в файле stm32f4xx_hal_tim.c.

Параметры: TIM_HandleTypeDef *htim представляет указатель на структуру данных типа TIM_HandleTypeDef.

Возвращаемое значение: функции не возвращают значение.

Краткий алгоритм работы:

1) Позволяет пользовательскому коду обрабатывать событие завершения задержки сравнения для таймера TIM.

3.5.6 Функция обратного вызова для обработки событий завершения периода TIM HAL_TIM_PeriodElapsedCallback() находится в файле stm32f4xx_hal_tim.c.

Параметры: TIM_HandleTypeDef *htim представляет указатель на структуру данных типа TIM_HandleTypeDef.

Возвращаемое значение: функции не возвращают значение.

Краткий алгоритм работы:

1) Позволяет пользовательскому коду реагировать на событие завершения периода таймера TIM.

3.5.7 Функция обратного вызова для обработки событий завершения периода для ШИМ сигнала TIM HAL_TIM_PWM_PulseFinishedCallback() находится в файле stm32f4xx_hal_tim.c.

Параметры:

Возвращаемое значение: функции не возвращают значение.

Краткий алгоритм работы:

3.5.8 Функция для запуска одноимпульсного режима HAL_TIM_OnePulse_Start() находится в файле stm32f4xx_hal_tim.c.

Параметры:

Возвращаемое значение: функции не возвращают значение.

Краткий алгоритм работы:

3.5.9 Функция для запуска ШИМ HAL_TIM_PWM_Start() находится в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_HandleTypeDef *htim: Указатель на структуру данных типа TIM_HandleTypeDef, которая содержит информацию о выбранном таймере TIM.

2) uint32_t Channel: Номер канала TIM, на котором нужно запустить генерацию PWM.

Возвращаемое значение: возвращает статус операции в виде значения типа HAL_StatusTypeDef (HAL_OK - успешно, HAL_ERROR - ошибка).

Краткий алгоритм работы:

1) Проверяются параметры (проверка на допустимость значений параметров).

2) Проверяется состояние канала TIM.

3) Устанавливается состояние канала TIM в режим занятости.

4) Включается канал сравнения (Capture Compare channel).

При необходимости (если TIM поддерживает BREAK) включается основной вывод TIM.

5) Включается периферийное устройство TIM, за исключением случая, когда таймер находится в режиме слейва и управление производится через триггер.

3.5.10 Функция для запуска ввода захвата HAL_TIM_IC_Start() находится в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_HandleTypeDef *htim: Указатель на структуру данных типа TIM_HandleTypeDef, которая содержит информацию о выбранном таймере (TIM).

2) uint32_t Channel: Номер канала TIM, на котором нужно запустить ввод захвата (Input Capture).

Возвращаемое значение: возвращает HAL_StatusTypeDef статус выполнения операции - HAL_OK в случае успешного запуска ввода захвата или HAL_ERROR в случае ошибки.

Краткий алгоритм работы:

1) Проверяет параметры функции и состояния каналов TIM.

2) Устанавливает состояние канала TIM в режим занятости (BUSY).

3) Включает канал для захвата ввода (Input Capture channel).

4) Включает периферию TIM в режиме Slave, если она не находится в режиме триггера.

3.5.11 Функция для запуска интерфейса энкодера HAL_TIM_Encoder_Start() находится в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_HandleTypeDef *htim - указатель на структуру TIM handle.

2) uint32_t Channel - указывает канал TIM, который следует запустить.

Возвращаемое значение: возвращает значение типа HAL_StatusTypeDef, которое может быть HAL_OK в случае успешного запуска энкодера или HAL_ERROR в случае ошибки.

Краткий алгоритм работы:

1) Получает состояния каналов 1 и 2, а также их комплементарных каналов.

2) Проверяет параметры и устанавливает состояния каналов TIM в зависимости от выбранного канала.

3) Включает интерфейс энкодера для выбранного канала или для обоих каналов, если выбран неопределенный или общий канал.

4) Результатом является запуск энкодера на выбранном канале TIM с соответствующими настройками.

3.5.12 Функции получения текущего состояние TIM и его компонентов HAL_TIM_IC_GetState, HAL_TIM_PWM_GetState, HAL_TIM_OC_GetState, HAL_TIM_Base_GetState находится в файле stm32f4xx_hal_tim.c.

Параметры: TIM_HandleTypeDef *htim, для которого нужно вернуть состояние.

Возвращаемое значение: возвращает состояние TIM из структуры HAL_TIM_StateTypeDef.

3.5.13 Функция для настройки очистки сравнения вывода HAL_TIM_ConfigOCrefClear() находится в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_HandleTypeDef *htim - указатель на структуру, представляющую TIM PWM handle.

2) TIM_ClearInputConfigTypeDef *sClearInputConfig - указатель на структуру, содержащую конфигурацию очистки OCRef и параметры для периферийного устройства TIM.

3) uint32_t Channel - указывает канал TIM, для которого применяется настройка очистки OCRef.

Возвращаемое значение: озвращает значение типа HAL_StatusTypeDef, которое может быть HAL_OK при успешной конфигурации очистки OCRef или HAL_ERROR в случае возникновения ошибки.

Краткий алгоритм работы:

1) Проверка переданных параметров. Блокировка процесса для предотвращения конфликтов доступа к ресурсам. Установка состояния TIM в занятое.

2) В зависимости от источника очистки OCRef. Отключение битов выбора очистки OCRef и битов ETR. Настройка параметров при использовании функции OCRef clear с источником ETR.

3) В зависимости от канала TIM: Включение или выключение функции очистки OCRef для соответствующего канала.

3.5.14 Функция для настройки источника тактового сигнала HAL_TIM_ConfigClockSource() находится в файле stm32f4xx_hal_tim.c.

Параметры:

1) TIM_HandleTypeDef *htim - указатель на структуру, представляющую TIM PWM handle.

2) TIM_ClockConfigTypeDef *sClockSourceConfig - указатель на структуру, содержащую конфигурацию источника тактового сигнала.

Возвращаемое значение: возвращает значение типа HAL_StatusTypeDef, которое может быть HAL_OK при успешной настройке источника тактирования или HAL_ERROR в случае возникновения ошибки.

Краткий алгоритм работы:

1) Блокировка процесса для предотвращения конфликтов доступа к ресурсам; Установка состояния TIM в занятое; Проверка переданных параметров; Сброс битов SMS, TS, ECE, ETPS и ETRF.

2) В зависимости от переданного источника тактового сигнала: Настройка внутреннего источника тактирования; Настройка внешнего источника тактирования в режиме ETRMODE1 или ETRMODE2; Настройка внешнего

источника тактирования через входы TI1, TI2 или TI1ED; Настройка внутреннего триггера ITR0-ITR3;

3) Установка состояния TIM в готовое; азблокировка процесса; Возврат статуса операции.

3.5.15 Функция для инициализации ШИМ HAL_TIM_PWM_Init() находится в файле stm32f4xx_hal_tim.c.

Параметры: указатель на структуру TIM_HandleTypeDef, представляющую TIM PWM handle.

Возвращаемое значение: возвращает значение типа HAL_StatusTypeDef, которое указывает на успешность выполнения операции и может быть HAL_OK или HAL_ERROR.

Краткий алгоритм работы:

1) Проверка наличия выделенного ресурса для TIM handle.

2) Проверка параметров инициализации с помощью макросов assert_param.

3) Если TIM находится в состоянии сброса, то: Выделение ресурса блокировки и его инициализация; В зависимости от макроса USE_HAL_TIM_REGISTER_CALLBACKS установка обработчиков прерываний на стандартные слабые обратные вызовы или кастомные обработчики; Инициализация низкоуровневого оборудования: GPIO, CLOCK, NVIC и возможно DMA.

4) Установка состояния TIM в занятое; Инициализация базового времени для PWM через вызов функци; TIM_Base_SetConfig; Инициализация состояния DMA-буфера; Инициализация состояния каналов TIM; Установка состояния TIM в готовое; Возврат успешного статуса HAL_OK.

4 Вывод: были изучены функции управления таймерами средствами библиотеки HAL в среде разработки STM32CubeIDE.