Лабораторная работа №3. «Использование таймеров и системы прерываний»

Задание

Разработать и реализовать драйверы для электромагнитного излучателя звука, управления яркостью светодиода, обработки внешних прерываний от кнопки. Написать программу с использованием разработанных драйверов по алгоритму, соответствующему варианту задания.

Общие сведения

Для использования внешних прерываний с кнопки достаточно лишь настроить соответствующий порт в режим GPIO_EXTI (Рисунок 1), а также включить прерывания (System Core->GPIO->NVIC).

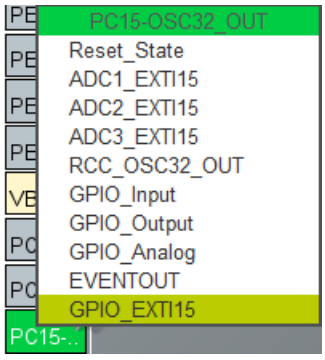

Рис. 1. Настройка внешних прерываний

Библиотека HAL предлагает два варианта обработки прерываний: с помощью обработчика (Handler) и с помощью функции обратного вызова (Callback), которая вызывается в конце обработчика. Обработчики находятся в файле stm32f4xx_it.c. Код функции Callback для кнопки выглядит следующим образом:

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if (GPIO_Pin == GPIO_PIN_15)
        HAL_GPIO_TogglePin(LED1_GPIO_Port, LED1_Pin);
}
```

В STM32F407VG есть три типа таймеров: Advanced-control (TIM1, TIM8), General-purpose timers (TIM2 – TIM5) и Basic (все остальные). Первые два типа позволяют генерировать ШИМ.

Основными параметрами таймера являются:

- Prescaler – предделитель частоты таймера;
- Counter Mode – направление отсчета (Up – от 0 до переполнения, Down – наоборот);
- Counter Period – значение переполнения счетчика;

Для генерации прерываний подходит любой базовый таймер. Пример настройки показан на рисунке 2.

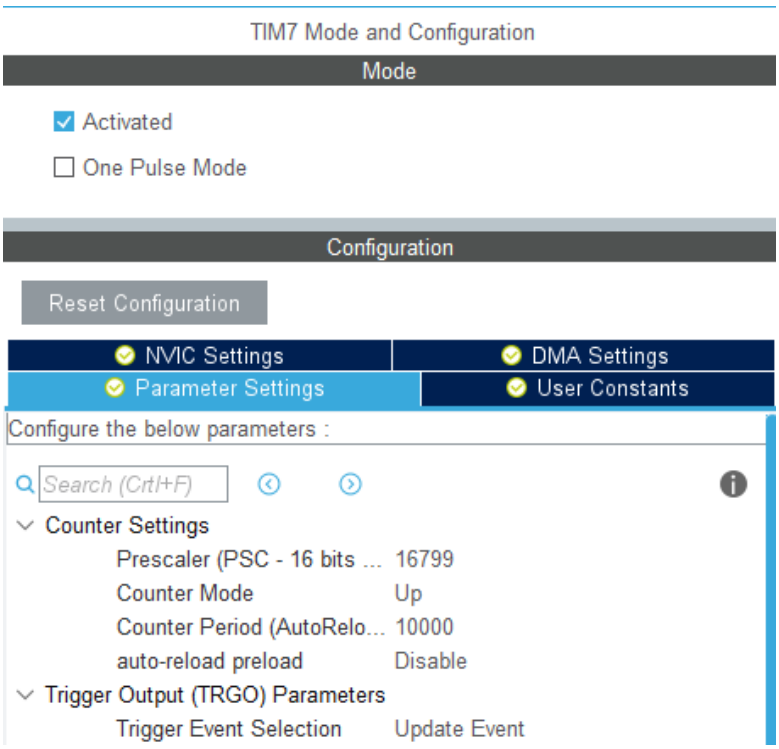

Рис. 2. Параметры TIM7

Для генерации прерываний по таймеру достаточно установить галочку в System Core->NVIC->TIM7 Global Interrupt. Чтобы таймер начал работу, в код программы необходимо добавить функцию HAL_TIM_Base_Start_IT(&htim7);

Функция Callback выглядит следующим образом:

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
     if (htim == &htim7)
          HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
}
```

С помощью ШИМа можно регулировать уровень напряжения на порте микроконтроллера, что позволяет изменять яркость светодиода, а также генерировать звуки разной частоты на электромагнитном звукоизлучателе. В репозитории на github выложены проекты SDK_TIM и SDK_Buzzer, в которых показаны примеры кода для управления яркостью светодиодов и электромагнитным звукоизлучателем соответственно.

Варианты заданий

Вариант 1.

Доработать программу «Светофор», добавив обработку прерывания с кнопки, во время работы зеленого сигнала светофора должен воспроизводиться звуковой сигнал. Переключение светофора реализовать с помощью таймера.

Вариант 2.

Доработать программу «Гирлянда», добавив обработку прерывания с кнопки, а также изменение яркости светодиодов. При нажатии кнопки помимо переключения режима должен воспроизводиться звуковой сигнал.

Вариант 3.

Доработать программу «Передатчик» азбуки Морзе, добавив обработку прерываний с кнопки. При нажатии кнопки должен воспроизводиться звуковой сигнал длительность которого соответствует длительности нажатия кнопки (короткое и длинное).

Вариант 4.

Реализовать программу с использованием кнопки, таймера и светодиодов, которая проигрывает восходящую гамму с длительностью нот 1с пока нажата кнопка. Если кнопку отжать, последовательность нот сбрасывается и при нажатии кнопки начинается заново. Каждую секунду нажатия кнопки происходит смена состояния зеленого светодиода.

Вариант 5.

Реализовать «Музыкальный плеер» с использованием кнопки, светодиодов, таймера и излучателя звука. При подаче питания начинается последовательное воспроизведение мелодий, сопровождающихся мигающими в такт светодиодами. Переключение мелодий осуществляется прерыванием от боковой кнопки SDK-1.1M.

Требования к выполнению работы

Проект должен быть в виде отдельных файлов, каждый из которых содержит отдельный драйвер или цельную, логически законченную часть программы.

Код должен быть структурированным, хорошо читаемым. Названия переменных и функций должны быть осмысленными и поясняющими назначение данных переменных и функций. На «верхнем уровне» программы (функция main) должна быть только прикладная часть, вся логика работы с аппаратурой скрывается от пользователя «верхнего уровня» и реализуется на уровне драйверов.

Не должно быть одинаковых или сильно похожих блоков кода, желательно реализовывать их с помощью отдельных функций. Крайне желательно минимизировать количество вложенных циклов и свести их к необходимому минимуму.

Требования к отчёту

Отчёт должен содержать:

1. Титульный лист с номером лабораторной работы, её названием, ФИО и группой исполнителей.
2. Номер варианта, задание.
3. Блок-схема прикладного алгоритма.
4. Подробное описание инструментария, который использовался в работе.
5. Исходные коды прикладной программы и использованных библиотек.
6. Выводы, в которых описываются проблемы, которые возникли в ходе работы, и способы их решения.

Защита лабораторной работы

На защите задаются вопросы по:

1. Теории.
2. Инструментальным средствам, которые использовались для работы.
3. Принципиальной схеме стенда SDK.
4. Исходным кодам, включая библиотеки.