Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Аппаратные платформы встраиваемых систем

Отчет по лабораторной работе №4
На тему **«Пьезоизлучатель (Buzzer)»**

Работу выполнили:
Студенты гр. 13541/1 Онищенко Д.И.
Шаменов А.А.
Преподаватель: Васильев А.Е.

Санкт – Петербург
2019

Содержание

Цель работы:

Получение навыков работы с пьезоизлучателем (buzzer) МК STM32.

Подготовка к работе:

1. Подготовить проект в IARWE, согласно документу IAR Project for IAR SK Board
2. Ознакомиться со схемой платы IAR SK (STM32F407ZG-board-schematic.pdf)
3. Ознакомиться с документацией МК STM32F407 (STM32F4xx_RM.pdf)

Теоретическая информация:

Пьезоизлучатель.

Пьезоэлектрический излучатель, пьезоизлучатель – электроакустическое устройство, способное воспроизводить звук, либо излучать ультразвук, благодаря обратному пьезоэлектрическому эффекту (т.е. эффекту возникновения поляризации диэлектрика под действием механических напряжений).

На плате STM32 расположен пьезоизлучатель PB-1221PQ, приведем его характеристики в нижеследующей таблице

Таблица 1 – Характеристики пьезоизлучателя PB-1221PQ

Частота колебаний (Гц)	2048
Рабочее напряжение (В)	1.25 ~ 2.5
Номинальное напряжение (В)	1.5
Максимальное потребление (mA)	18 / ном. напр.
Уровень звукового давления (dB/min)	8 / ном. напр.
Сопротивление катушки (Ω)	42±5

Рабочая температура, ^{0}C	-20~+70
Температура хранения, 0C	-30~+80

Схема подключения пьезоизлучателя к плате показана ниже. Отметим, что он подключен от источника 3.3В, управляющий сигнал - BUZ:

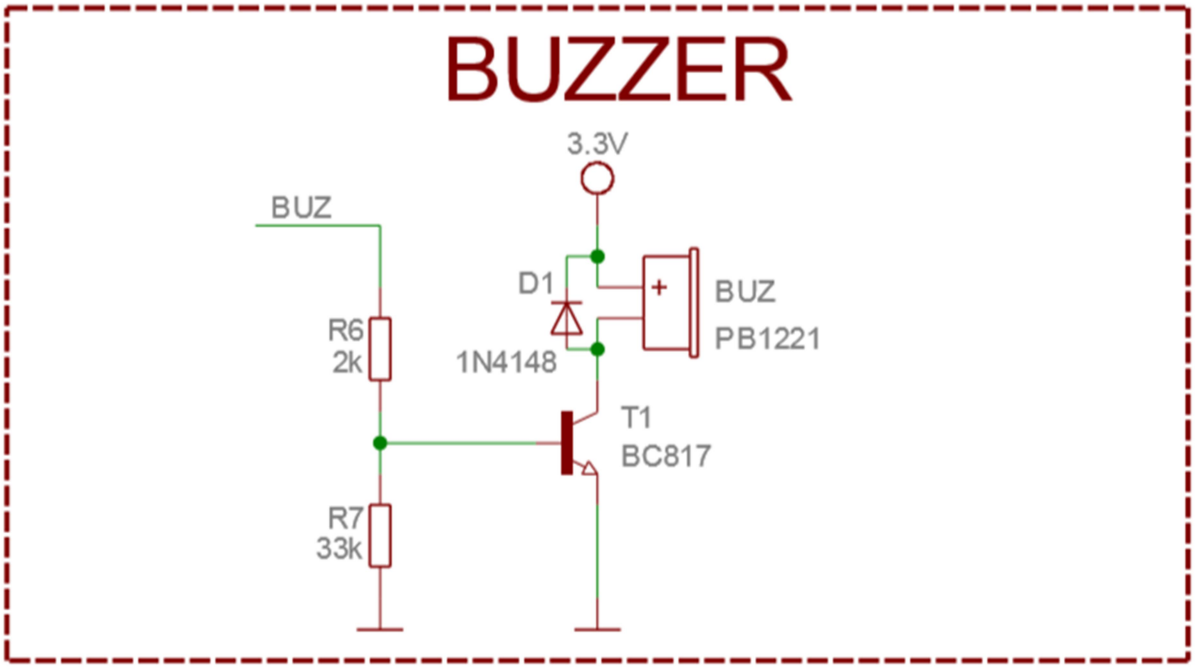

Рисунок 1 – Схема подключения пьезоизлучателя

Ход работы:

Задание состоит из следующих пунктов:

1. Изучить работу с пьезоизлучателем. Необходимо написать программу, в ней задать такие конфигурации ШИМа и таймера, чтобы пьезоизлучатель воспроизводил звук путем колебания звуковых волн. Модифицировать программу.
2. При нажатии кнопки WKUP_BTN необходимо, чтобы пьезоизлучатель воспроизводил звук, в противном случае звук не должен воспроизводится.

Практическая часть:

Продемонстрируем последовательность действий для задания №1.

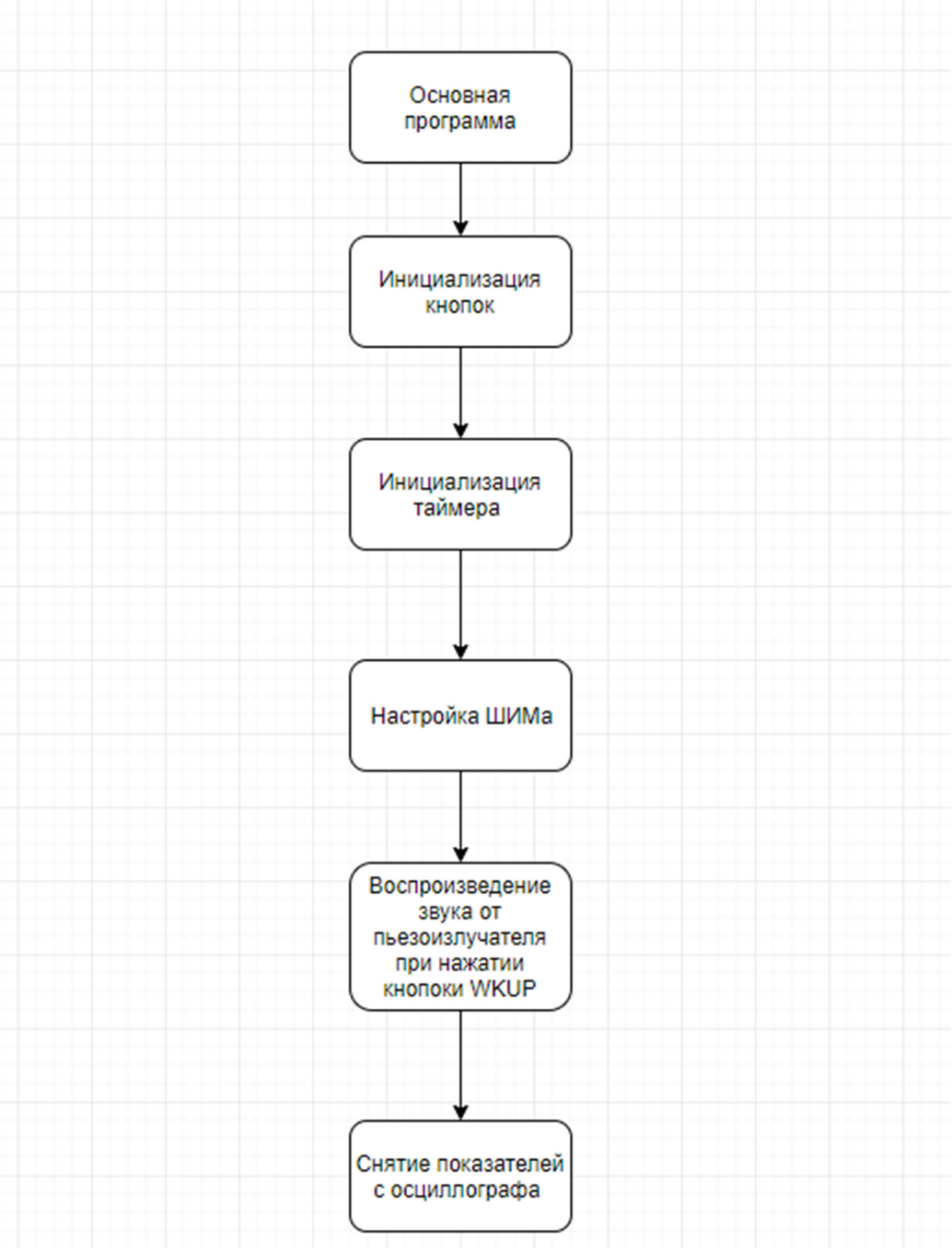

Рисунок 2 – Схема последовательности действии для первого задания

Проинициализируем кнопки и светодиоды:

```
1. void BTN_init (void) {
2.     /* FOR User Button */
3.     RCC_AHB1PeriphClockCmd(BUTTON_USER_RCC, ENABLE);
4.     GPIO_InitStruct.GPIO_Pin = BUTTON_USER_PIN;
5.     GPIO_InitStruct.GPIO_Mode = GPIO_Mode_IN;
6.     GPIO_Init(BUTTON_USER_PORT, &GPIO_InitStruct);
7.     /* FOR WKUP */
8.     RCC_AHB1PeriphClockCmd(BUTTON_WKUP_RCC, ENABLE);
9.     GPIO_InitStruct.GPIO_Pin = BUTTON_WKUP_PIN;
10.    GPIO_InitStruct.GPIO_Mode = GPIO_Mode_IN;
11.    GPIO_Init(BUTTON_WKUP_PORT, &GPIO_InitStruct);
12. }
```

Далее проинициализируем таймер и настроим ШИМ:

```
1. GPIO_InitTypeDef GPIO_InitStructure;
2. TIM_TimeBaseInitTypeDef  TIM_TimeBaseStructure;
3. TIM_OCInitTypeDef TIM_OCInitStructure;
4. RCC_ClocksTypeDef RCC_Clocks;
5. int pwm_state = 0;
6. /*!< At this stage the microcontroller clock setting is al
7. ready configured,
8. this is done through SystemInit() function which is called
9. from startup
10.file (startup_stm32f4xx.s) before to branch to application
11.main.
12.To reconfigure the default setting of SystemInit() function,
13.refer to
14.system_stm32f4xx.c file
15.*/
16.ENTR_CRT_SECTION();
17./* SysTick Config*/
18.if(SysTick_Config(SystemCoreClock/10000))
19.{
20.   /* Capture error */
21.   while (1);
22.}
23.EXT_CRT_SECTION();
24.//GLCD init
```

```
25. GLCD_PowerUpInit((pInt8U)IAR_Logo.pPicStream);
26. /*Turn on Backlight*/
27. GLCD_Backlight(BACKLIGHT_OFF);
28. /*TAMPER button init*/
29. STM_EVAL_PBInit(BUTTON_TAMPER,BUTTON_MODE_GPIO);
30. /* Enable the BUZZER_GPIO Clock */
31. RCC_AHB1PeriphClockCmd(BUZZER_GPIO_CLK, ENABLE);
32. /* Configure Buzzer pin*/
33. GPIO_InitStructure.GPIO_Pin = BUZZER_GPIO_PIN;
34. GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
35. GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
36. GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
37. GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
38. GPIO_Init(BUZZER_GPIO_PORT, &GPIO_InitStructure);
39. /*Select AF*/
40. GPIO_PinAFConfig(BUZZER_GPIO_PORT,BUZZER_PIN_SOURCE,
41. BUZZER_PIN_AF);
42. /* Buzzer Timer enable*/
43. RCC_APB2PeriphClockCmd(RCC_APB2PERIPH_BUZZER_TIM,ENABLE);
44. RCC_APB2PeriphResetCmd(RCC_APB2PERIPH_BUZZER_TIM,DISABLE);
45. RCC_APB2PeriphClockCmd(RCC_APB2PERIPH_BUZZER_TIM, ENABLE);
46. RCC_GetClocksFreq(&RCC_Clocks);
47. /*Time base configuration*/
48. TIM_TimeBaseStructure.TIM_Period = 0xFF;
49. /*8 bit resolution*/
50. TIM_TimeBaseStructure.TIM_Prescaler =
51.  RCC_Clocks.HCLK_Frequency/(256*BUZZER_FREQ); /*2kHz PWM
52. periotd*/
53. TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
54. TIM_TimeBaseStructure.TIM_ClockDivision = 0;
55. TIM_TimeBaseStructure.TIM_RepetitionCounter = 0;
56. TIM_TimeBaseInit(BUZZER_TIM,&TIM_TimeBaseStructure);
57. /*Output Compare init*/
58. TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM1;
59. TIM_OCInitStructure.TIM_OutputState =
60. TIM_OutputState_Disable;
61. TIM_OCInitStructure.TIM_OutputNState =
62. TIM_OutputNState_Disable;
63. /*50%*/
64. TIM_OCInitStructure.TIM_Pulse = 0x80;
65. TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High;
66. TIM_OCInitStructure.TIM_OCNPolarity = TIM_OCNPolarity_High;
67. TIM_OCInitStructure.TIM_OCIdleState = TIM_OCIdleState_Reset;
68. TIM_OCInitStructure.TIM_OCNIdleState =
69. TIM_OCIdleState_Reset;
70. TIM_OC3Init(BUZZER_TIM,&TIM_OCInitStructure);
71. TIM_ARRPreloadConfig(BUZZER_TIM,ENABLE);
72. /*Timer counter enable*/
73. TIM_Cmd(BUZZER_TIM,ENABLE);
74. /*Enables the TIM peripheral Main Outputs*/
75. TIM_CtrlPWMOutputs(BUZZER_TIM,ENABLE);
```

В бесконечном цикле считываем кнопку WKUP_BTN, проверяем, нажата ли она, и, если нажата, воспроизводим звук при помощи функции TIM_CCxCmd():

```
1. while(1)
2. {
3.    if (Bit_RESET != STM_EVAL_PBGetState(BUTTON_WAKEUP))
4.    {
5.      if(0 == pwm_state)
6.      {
7.         pwm_state = 1;
8.         /*Enabel PWM Output*/
9.         TIM_CCxCmd(BUZZER_TIM,TIM_Channel_3,TIM_CCx_Enable);
10.     }
11.   }
12.   else
13.   {
14.     if(1 == pwm_state)
15.     {
16.        pwm_state = 0;
17.        /*Delay*/
18.        TIM_CCxCmd(BUZZER_TIM,TIM_Channel_3,TIM_CCx_Disable);
19.     }
20.   }
21. }
```

Снимем показания с осциллографа:

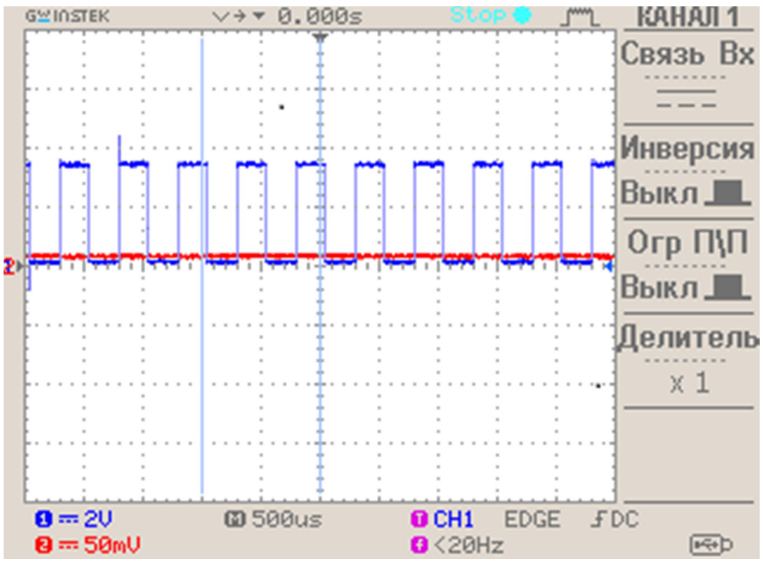

Рисунок 3 – Показания осциллографа при нажатой кнопке

Вывод

В данной лабораторной работе были приобретены практические навыки работы с пьезоизлучателем. На практике реализовали воспроизведение звука при помощи пьезоизлучателя. Конфигурация таймера и настройки ШИМа имеют некоторые особенности: для работы с пьезоизлучателем необходимо использовать первый таймер и третий канал. Их грамотная конфигурация и является основной особенностью работы с пьезоизлучателем.