Санкт-Петербургский Национальный Исследовательский Университет ИТМО

Факультет программной инженерии и компьтерной техники

Лабораторная работа №3 по Проектированию вычислительных систем

Вариант 1

Роман Юнусов

Алексей Лапин  
Группа P34102

Преподаватель Василий Пинкевич

Санкт-Петербург 2024

Оглавление

[Задание 3](#_Toc184064148)

[Схемы 3](#_Toc184064149)

[Решение 4](#_Toc184064150)

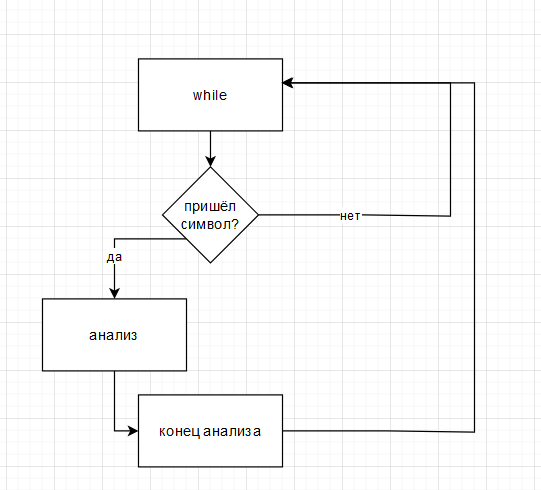
[Выводы 5](#_Toc184064151)

# Задание

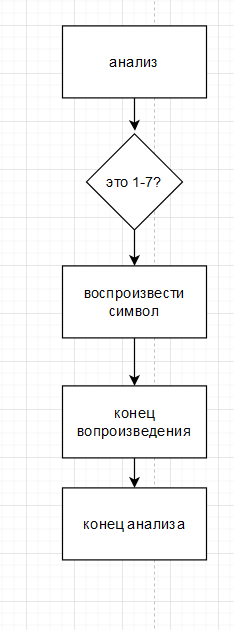
Реализовать «музыкальную клавиатуру» с помощью излучателя звука.  
Должна быть возможность выбора октавы и длительности воспроизведения. В каждой октаве можно либо выбрать нужную ноту(от до, до си), либо при нажатии проиграть все последовательно.

# Схемы

Основной алгоритм



Алгоритм анализа на воспроизведение ноты по цифре



Воспроизведение мелодии

# Решение

Глобальные переменные, которые соответствуют нашей системе. Номер октавы, длительность мелодии, и размер регистра для сочетания октавы и ноты.

**int** now\_oktav = 4;

**int** music\_time = 1000;

uint32\_t oktava\_info[9][7] = {

{36781, 32690, 29124, 27489, 24490, 21818, 19438},

{18391, 16345, 14562, 13745, 12245, 10909, 9719},

{9195, 8172, 7281, 6872, 6122, 5455, 4859},

{4598, 4086, 3640, 3436, 3061, 2727, 2430},

{2299, 2043, 1820, 1718, 1531, 1364, 1215},

{1149, 1022, 910, 859, 765, 682, 607},

{575, 511, 455, 430, 383, 341, 304},

{287, 255, 228, 215, 191, 170, 152},

{144, 128, 114, 107, 96, 85, 76}

};

Код анализа символа. Просто смотрим, и выполняем команды, которые либо меняют наши параметры, либо выполняют мелодию

**void** **analize\_symb**(**char** el){

**if**(el >= '1' && el <= '7'){

**int** note\_to\_ring = el - '1';

execute\_one\_oktav(note\_to\_ring);

}

**else**

**if**(el == '+' || el == '-'){

**if**(el == '+')

change\_now\_oktav(1);

**else**

change\_now\_oktav(-1);

**char** out\_buffer[100];

**int** transmitted\_data\_len;

transmitted\_data\_len = **sprintf**(out\_buffer, "change oktav\_num, now is %d\r", now\_oktav);

HAL\_UART\_Transmit( &huart6, (uint8\_t \*) out\_buffer, transmitted\_data\_len, 30 );

}

**else**

**if**(el == 'A' || el == 'a') {

**if**(el == 'A')

change\_music\_time(10);

**else**

change\_music\_time(-10);

**char** out\_buffer[100];

**int** transmitted\_data\_len;

transmitted\_data\_len = **sprintf**(out\_buffer, "change time, now is %d\r", music\_time);

HAL\_UART\_Transmit( &huart6, (uint8\_t \*) out\_buffer, transmitted\_data\_len, 30 );

}

**else**

**if**(el == '\r'){

execute\_melody();

}

**else**

{

**char** out\_buffer[100];

**int** transmitted\_data\_len;

transmitted\_data\_len = **sprintf**(out\_buffer, "wrong symbol %c\r", el);

HAL\_UART\_Transmit( &huart6, (uint8\_t \*) out\_buffer, transmitted\_data\_len, 30 );

}

}

Код исполнения одной октавы. Для смены параметров прибор не выключается, мы напрямую лезем в характеристики, изменяя регистр в соответствии с расчётным значением, а пульсовое значение как его половину для максимальной громкости. В этот момент времени мы находимся в блокирующем режиме и не можем принимать команды через UART. После исполнения ноты, мы заглушаем звук через установку размера пульсации в 0.

**void** **execute\_one\_oktav**(**int** note\_to\_ring){

**char** out\_buffer[100];

**int** transmitted\_data\_len;

transmitted\_data\_len = **sprintf**(out\_buffer, "note %s is ringning on %d oktav, time is %d\r", noteNames[note\_to\_ring], now\_oktav, music\_time);

HAL\_UART\_Transmit( &huart6, (uint8\_t \*) out\_buffer, transmitted\_data\_len, 30 );

htim1.Instance->ARR = oktava\_info[now\_oktav][note\_to\_ring] - 1;

htim1.Instance->CCR1 = (oktava\_info[now\_oktav][note\_to\_ring] - 1) / 2;

**int** start\_time = HAL\_GetTick();

**while**(HAL\_GetTick() - start\_time < music\_time){

}

htim1.Instance->CCR1 = 0;

}

# Выводы

Мы использовали механизмы таймеров для задания ШИМ сигналов, управляющих генератором звука. Также мы получили практические изменения частоты сигналов во время работы, и определение начальных значений для наиболее точного воспроизведения необходимой частоты.