

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт компьютерных наук и технологий

Высшая школа искусственного интеллекта

Дисциплина:
ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ

ОТЧЕТ
По лабораторной работе №8
Тема: Использование таймеров STM32F200 для
генерирования сложных форм волн

Обучающийся гр. 3530201/10001

Нгуен Куок Дат

Руководитель _____ Вербова Наталья Михайловна

Санкт-Петербург 2022

Содержание

1	Цель и постановка задачи	2
1.1	Цель работы	2
1.2	Постановка задачи	2
2	Выполнение задания	3

1 Цель и постановка задачи

1.1 Цель работы

Ознакомиться с основными приемами изучения предметной области программируемой задачи. Ознакомиться с генерированием модулированных колебаний. Закрепить навыки работы с низкоуровневыми библиотеками и промежуточным программным обеспечением микроконтроллера. Закрепить навыки отладки программ.

1.2 Постановка задачи

Используя библиотеки Keil μ Vision5, разработать программу для генерирования одного из несущих колебаний для частотной манипуляции.

2 Выполнение задания

Код программы

```
#include "math.h"
```

```
#include "stm32f2xx_hal.h" // Keil::Device:STM32Cube HAL:Common
```

```
uint16_t sinetable[] = {  
    127,130,133,136,139,143,146,149,152,155,158,161,164,167,170,173,176,178,181,  
    184,187,190,192,195,198,200,203,205,208,210,212,215,217,219,221,223,225,227,  
    229,231,233,234,236,238,239,240,242,243,244,245,247,248,249,249,250,251,252,  
    252,253,253,253,254,254,254,254,254,254,254,253,253,253,252,252,251,250,249,
```

```

249,248,247,245,244,243,242,240,239,238,236,234,233,231,229,227,225,223,221,
219,217,215,212,210,208,205,203,200,198,195,192,190,187,184,181,178,176,173,
170,167,164,161,158,155,152,149,146,143,139,136,133,130,127,124,121,118,115,
111,108,105,102,99,96,93,90,87,84,81,78,76,73,70,67,64,62,59,56,54,51,49,46,
44,42,39,37,35,33,31,29,25,23,21,20,18,16,15,14,12,11,10,9,7,6,5,5,4,3,2,
2,1,1,1,0,0,0,0,0,0,0,0,1,1,1,2,2,3,4,5,5,6,7,9,10,11,12,14,15,16,18,20,21,23,
25,27,29,31,33,35,37,39,42,44,46,49,51,54,56,59,62,64,67,70,73,76,78,81,84,
87,90,93,96,99,102,105,108,111,115,118,121,124};

```

```

#define TIMER_PERIOD 1000

```

```

#define WAVE_ZERO 1180

```

```

#define WAVE_ONE 980

```

```

GPIO_InitTypeDef  GPIO_InitStruct;

```

```

TIM_HandleTypeDef htim;

```

```

uint32_t R = (256 *(WAVE_ONE/WAVE_ZERO));

```

```

uint16_t pulse_width;

```

```

uint32_t phase_accumulator;

```

```

uint8_t angle = 0;

```

```

unsigned int i;

```

```

void PWM_SetDC(uint16_t channel,uint16_t dutycycle)

```

```

{

```

```

    if (channel == 1)

```

```

{
    TIM2->CCR1 = dutycycle;
}

else if (channel == 2)
{
    TIM2->CCR2 = dutycycle;
}
}

void TIM2_IRQHandler(void)
{
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_7, GPIO_PIN_SET);
    PWM_SetDC(1,pulse_width);
    PWM_SetDC(2,pulse_width);

    // Calculate a new pulse width
    phase_accumulator += R;
    angle = phase_accumulator;
    pulse_width = sinetable[angle];
    TIM2->SR = ~(htim.State);
    HAL_GPIO_WritePin(GPIOG, GPIO_PIN_7, GPIO_PIN_RESET);
}

void InitializeLED()
{
    RCC->AHB1ENR |= RCC_AHB1ENR_GPIOGEN;

```

```

        /* GPIO base configuration */

        GPIO_InitStruct.Pin |= (GPIO_PIN_7);
        GPIO_InitStruct.Pin |= (GPIO_PIN_8);
        GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
        GPIO_InitStruct.Speed = GPIO_SPEED_LOW;
        HAL_GPIO_Init(GPIOG, &GPIO_InitStruct);

        HAL_GPIO_WritePin(GPIOG, GPIO_PIN_7, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(GPIOG, GPIO_PIN_8, GPIO_PIN_RESET);
    }

    void InitializeTimer()
    {
        //????????? ?????????????? ???????

        RCC->APB1ENR |= RCC_APB1ENR_TIM2EN ;

        /* Time base configuration */

        htim.Instance = TIM2;
        htim.Init.Period = TIMER_PERIOD;
        htim.Init.Prescaler = 40000;
        htim.Init.ClockDivision = 0;
        htim.Init.RepetitionCounter = 0;
        htim.Init.CounterMode = TIM_COUNTERMODE_UP;
        HAL_TIM_Base_Init(&htim);

        /* Enable TIM peripheral counter */
        HAL_TIM_Base_Start(& htim);
    }

```

```

        HAL_TIM_Base_Start_IT(&htim);
    }

void EnableTimerInterrupt(){
    NVIC_SetPriorityGrouping(0);
    NVIC_SetPriority(TIM2_IRQn,1);
    NVIC_EnableIRQ(TIM2_IRQn);
}

int main()
{
    InitializeLED();
    InitializeTimer();

    for (;;)
    {
        TIM2_IRQHandler();
    }
}

```

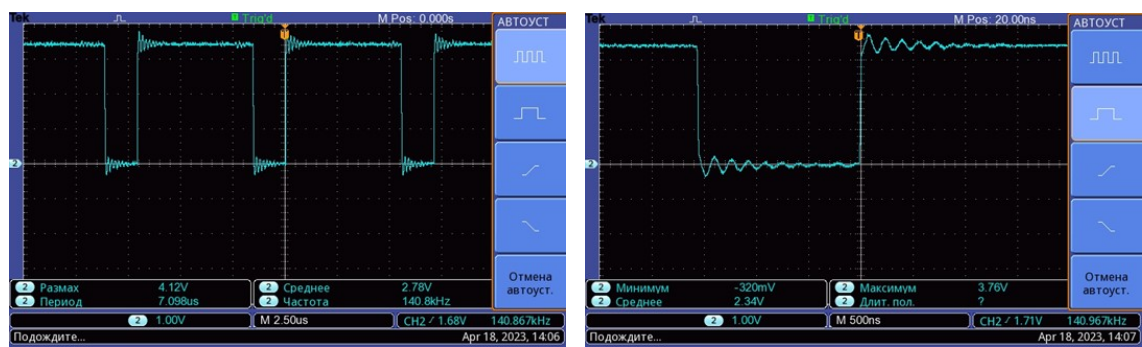


Рис. 1: Форма модулированной волны

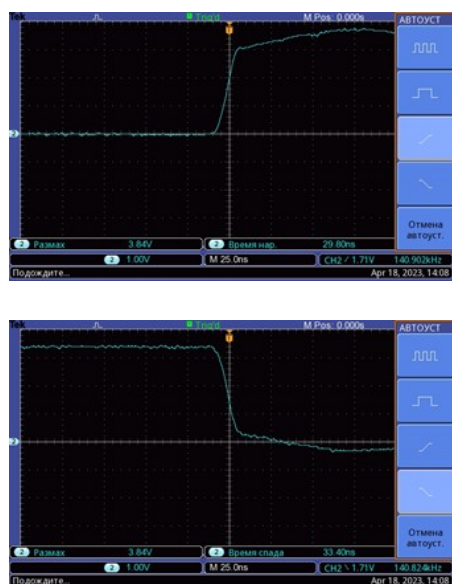


Рис. 2: Фронты модулированной волны