ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра вычислительной техники



Лабораторная работа №4

по дисциплине «Периферийные устройства информационных систем»

Таймерная подсистема микроконтроллера

STM32F103C8T6.

Группа: АВТ-819 Преподаватель: Токарев В. Г.

Студент: Ванин К.Е.

Новосибирск

2021

**Цель работы:**

Получить основные понятия о таймерных подсистемах современных микроконтроллеров с ядром Cortex M3. В ходе работы требуется изучить:

- способы формирования тактовых сигналов базовых таймеров и таймеров с

расширенным функционалом;

-методику расчета выходных частот и длительностей импульсов;

-способы конфигурирования выводов GPIO микроконтроллера в качестве

входных/выходных выводов таймеров.

-способы формирования широто-импульсно модулированных сигналов.

-способы обработки прерываний от таймеров.

**Задание:**

- в среде CooCoxIDE создать встраиваемое приложение.

- запрограммировать таймер TIM4 для генерации ШИМ сигналов частотой 50 Гц и

изменяемой длительностью импульса от 1 до 2 мс. на выводах PB6, PB7, PB8, PB9.

(проверить логическим анализатором). Регулировать длительность можно с помощью

ADC из предыдущего проекта.

- используя прерывания от таймера TIM4 сформировать «плавное» переключение

светодиодов. LED3-LED6. Возможно использование и других таймеров.

**Ход работы:**

1. Было создано встраиваемое приложение
2. Был запрограммирован таймер TIM4 для генерации сигналов частотой 50 Гц
3. Рассмотрена зависимость канала захвата/сравнения и регистра захвата/сравнения

Листинг программы:

#include "stm32f10x.h"

#include "stm32f10x\_tim.h"

#include "stm32f10x\_rcc.h"

#include "stm32f10x\_gpio.h"

#include "misc.h"

#define maxval 1000

TIM\_TimeBaseInitTypeDef TIM\_TimeBaseStructure;

TIM\_OCInitTypeDef TIM\_OCInitStructure;

uint16\_t CCR1\_Val = 664;

uint16\_t CCR2\_Val = 664; //249

uint16\_t CCR3\_Val = 664; //166

uint16\_t CCR4\_Val = 664; //83

uint16\_t PrescalerValue = 0;

uint16\_t EventCnt = 0;

void RCC\_Configuration(void);

void GPIO\_Configuration(void);

void NVIC\_Configuration(void);

void NVIC\_Configuration(void) {

NVIC\_InitTypeDef NVIC\_InitStructure;

NVIC\_InitStructure.NVIC\_IRQChannel = TIM4\_IRQn;

NVIC\_InitStructure.NVIC\_IRQChannelPreemptionPriority = 0;

NVIC\_InitStructure.NVIC\_IRQChannelSubPriority = 1;

NVIC\_InitStructure.NVIC\_IRQChannelCmd = ENABLE;

NVIC\_Init(&NVIC\_InitStructure);

}

void TIM4\_IRQHandler(void)

{

if (TIM\_GetITStatus(TIM4, TIM\_IT\_CC1) != RESET) {

/\* Очищаем TIM4 Capture Compare1 interrupt pending bit\*/

TIM\_ClearITPendingBit(TIM4, TIM\_IT\_CC1);

}

else if (TIM\_GetITStatus(TIM4, TIM\_IT\_CC2) != RESET) {

TIM\_ClearITPendingBit(TIM4, TIM\_IT\_CC2);

}

else if (TIM\_GetITStatus(TIM4, TIM\_IT\_CC3) != RESET) {

TIM\_ClearITPendingBit(TIM4, TIM\_IT\_CC3);

}

else if (TIM\_GetITStatus(TIM4, TIM\_IT\_CC4) != RESET) {

TIM\_ClearITPendingBit(TIM4, TIM\_IT\_CC4);

}

else {

TIM\_ClearITPendingBit(TIM4, TIM\_IT\_Update);

if(EventCnt++ > maxval) {

EventCnt=0;

TIM\_SetCompare1(TIM4, 0);

TIM\_SetCompare2(TIM4, 0);

TIM\_SetCompare3(TIM4, 0);

TIM\_SetCompare4(TIM4, 0);

}

else {

if(EventCnt < maxval\*0.25)

TIM\_SetCompare1(TIM4, EventCnt);

else if(EventCnt < maxval\*0.5)

TIM\_SetCompare2(TIM4, EventCnt-maxval\*0.25);

else if(EventCnt < maxval\*0.75)

TIM\_SetCompare3(TIM4, EventCnt-maxval\*0.5);

else if(EventCnt < maxval)

TIM\_SetCompare4(TIM4, EventCnt-maxval\*0.75);

}

}

}

void TIM\_PWMOutput(void) {

RCC\_Configuration();

GPIO\_Configuration();

TIM4 Конфигурация: генерируем 4 PWM сигнала с 4 различными скважностями:

TIM4CLK частота устанавливается переменной SystemCoreClock (Hz),

для работы таймера TIM4 на частоте 24 MHz предделитель рассчитывается по выражению:

- Prescaler = (TIM4CLK / TIM4 counter clock) - 1

SystemCoreClock устанавливается в 72 MHz для Low-density, Medium-density,

High-density и Connectivity line устройств (STM32F103xxx) и в 24 MHz для

Low-Density Value line и Medium-Density Value line устройств

(STM32F100xxx)

TIM4 работает на частоте 36 KHz, рассчитывается по выражению:

TIM4 Frequency = TIM4 counter clock/(ARR + 1)

или = 24 MHz / 666 = 36 KHz

TIM4 Channel1 duty cycle = (TIM4\_CCR1/ TIM4\_ARR)\* 100 = 50%

TIM4 Channel2 duty cycle = (TIM4\_CCR2/ TIM4\_ARR)\* 100 = 37.5%

TIM4 Channel3 duty cycle = (TIM4\_CCR3/ TIM4\_ARR)\* 100 = 25%

TIM4 Channel4 duty cycle = (TIM4\_CCR4/ TIM4\_ARR)\* 100 = 12.5%

----------------------------------------------------------------------- \*/

/\* Расчет значения предделителя \*/

PrescalerValue = (uint16\_t) (SystemCoreClock / 10000000) - 1;

/\* Базовая конфигурация таймера \*/

TIM\_TimeBaseStructure.TIM\_Period = maxval;

TIM\_TimeBaseStructure.TIM\_Prescaler = PrescalerValue;

TIM\_TimeBaseStructure.TIM\_ClockDivision = 0;

TIM\_TimeBaseStructure.TIM\_CounterMode = TIM\_CounterMode\_Up;

/\* Инициализируем таймер \*/

TIM\_TimeBaseInit(TIM4, &TIM\_TimeBaseStructure);

/\* PWM1 Mode конфигурация: Channel1 \*/

TIM\_OCInitStructure.TIM\_OCMode = TIM\_OCMode\_PWM1;

TIM\_OCInitStructure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = CCR1\_Val;

TIM\_OCInitStructure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

/\* Инициализируем канал сравнения \*/

TIM\_OC1Init(TIM4, &TIM\_OCInitStructure);

/\* Задействуем теневой регистр канала \*/

TIM\_OC1PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

/\* PWM1 Mode конфигурация: Channel2 \*/

TIM\_OCInitStructure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = CCR2\_Val;

/\* Инициализируем канал сравнения \*/

TIM\_OC2Init(TIM4, &TIM\_OCInitStructure);

/\* Задействуем теневой регистр канала \*/

TIM\_OC2PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

/\* PWM1 Mode конфигурация: Channel3 \*/

TIM\_OCInitStructure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = CCR3\_Val;

/\* Инициализируем канал сравнения \*/

TIM\_OC3Init(TIM4, &TIM\_OCInitStructure);

/\* Задействуем теневой регистр канала \*/

TIM\_OC3PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

/\* PWM1 Mode конфигурация: Channel4 \*/

TIM\_OCInitStructure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OCInitStructure.TIM\_Pulse = CCR4\_Val;

/\* Инициализируем канал сравнения \*/

TIM\_OC4Init(TIM4, &TIM\_OCInitStructure);

/\* Задействуем теневой регистр канала \*/

TIM\_OC4PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

/\* Задействуем теневой регистр таймера \*/

TIM\_ARRPreloadConfig(TIM4, ENABLE);

NVIC\_Configuration();

/\* TIM4 включение таймера \*/

TIM\_Cmd(TIM4, ENABLE);

TIM\_ITConfig(TIM4, TIM\_IT\_Update, ENABLE);

while (1)

{

}

}

void RCC\_Configuration(void) {

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM4, ENABLE);

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA | RCC\_APB2Periph\_GPIOB |

RCC\_APB2Periph\_GPIOC | RCC\_APB2Periph\_AFIO, ENABLE);

}

void GPIO\_Configuration(void) {

GPIO\_InitTypeDef GPIO\_InitStructure;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_6 | GPIO\_Pin\_7 | GPIO\_Pin\_8 | GPIO\_Pin\_9;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, &GPIO\_InitStructure);

}

**Вывод:**

В ходе лабораторной работы были изучены основные понятия о таймерных подсистемах современных микроконтроллеров с ядром Cortex M3. Также были изучены:

* способы формирования тактовых сигналов базовых таймеров и таймеров с

расширенным функционалом;

* методику расчета выходных частот и длительностей импульсов;
* способы конфигурирования выводов GPIO микроконтроллера в качестве

входных/выходных выводов таймеров.

* способы формирования широто-импульсно модулированных сигналов.
* способы обработки прерываний от таймеров.