Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический

университет имени В.Ф. Уткина»

Кафедра «ЭВМ»

Отчет о практической работе №1

«Программирование микроконтроллеров. Настройка среды Keil uVision. Выводы общего назначения. Параллельные порты»

по дисциплине

«Специализированные ЭВМ»

Выполнили:

Студенты группы 045

Вашкулатов Н.А.

Анохин В.А.

Проверил:

доц. каф. ЭВМ Устюков Д.И.

**Цель работы**: закрепить общие представления о выводах общего назначения, как интерфейсе периферийных устройств микропроцессорных систем, а также получить навыки настройки и создания проектов в среде Keil uVision для разработки программ на языке C.

**Ход работы**

**Задание 1.1.** Настройка проекта. Мигание диодом

После создания и настройки проекта был создан файл main.c, скомпилирован и запущен на контроллере. Светодиод начал мигать. При уменьшении задержки Delay светодиод мигает чаще.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit; //объявление переменной типа структура

void Delay (int del)//объявление процедуры "задержки"(пустой цикл с задаваемым числом повторений

{

for (int i=0; i<del; i++);//описание цикла в соответствии с синтаксисом языка С

}

void LedPinCfg (void) //описание процедуры инициализации порта

{

//инициализация порта осуществляется через структуру типа PORT\_InitTypeDef

PortInit.PORT\_Pin = (PORT\_Pin\_0);//выбор вывода порта, в данном случае 0 пин порта С

PortInit.PORT\_OE = PORT\_OE\_OUT;//порт для вывода информации

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от одного уровня к другому - медленно

PORT\_Init(MDR\_PORTC, &PortInit);//инициализация порта

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

LedPinCfg();//выполнение функции инициализации порта

while (1) {//бесконечный цикл

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0);//установить единицу в 0 бит порта C

Delay (100000); //Задержка в 100 000 операций процессора

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);//сбросить младший бит порта C

Delay (100000); //Задержка в 100 000 операций процессора

}

}

**Задание 1.2.** Инициализация нескольких бит одного порта

Поочередное мигание светодиодами. Для этого было проинициализировано два бита порта для управления светодиодами.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit; //объявление переменной типа структура

void Delay (int del)

{

for (int i=0; i<del; i++);

}

void LedPinCfg (void) //описание процедуры инициализации порта

{

//инициализация порта осуществляется через структуру типа PORT\_InitTypeDef

PortInit.PORT\_Pin = (PORT\_Pin\_0|PORT\_Pin\_1);

PortInit.PORT\_OE = PORT\_OE\_OUT;//порт для вывода информации

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;

PORT\_Init(MDR\_PORTC, &PortInit);//инициализация порта

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

LedPinCfg();//выполнение функции инициализации порта

while (1) {//бесконечный цикл

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_1);

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0);//установить младший бит порта C

Delay (1000000); //Задержка в 1 000 000 операций процессора

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);//сбросить младший бит порта C

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_1);

Delay (1000000); //Задержка в 1 000 000 операций процессора

}

}

**Задание 1.3.** Вывод информации через параллельный порт

На порт А выводим значение счетчика. На рисунке 1 показаны значения на выводах, полученные при помощи логического анализатора.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit; //объявление переменной типа структура

void portSetUp(){

PortInit.PORT\_Pin = PORT\_Pin\_All;

PortInit.PORT\_OE = PORT\_OE\_OUT;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от

PORT\_Init(MDR\_PORTA, &PortInit);

}

int main (void){

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);

portSetUp();

uint32\_t count = 0;

while(1){

PORT\_Write(MDR\_PORTA, count++);

}

}

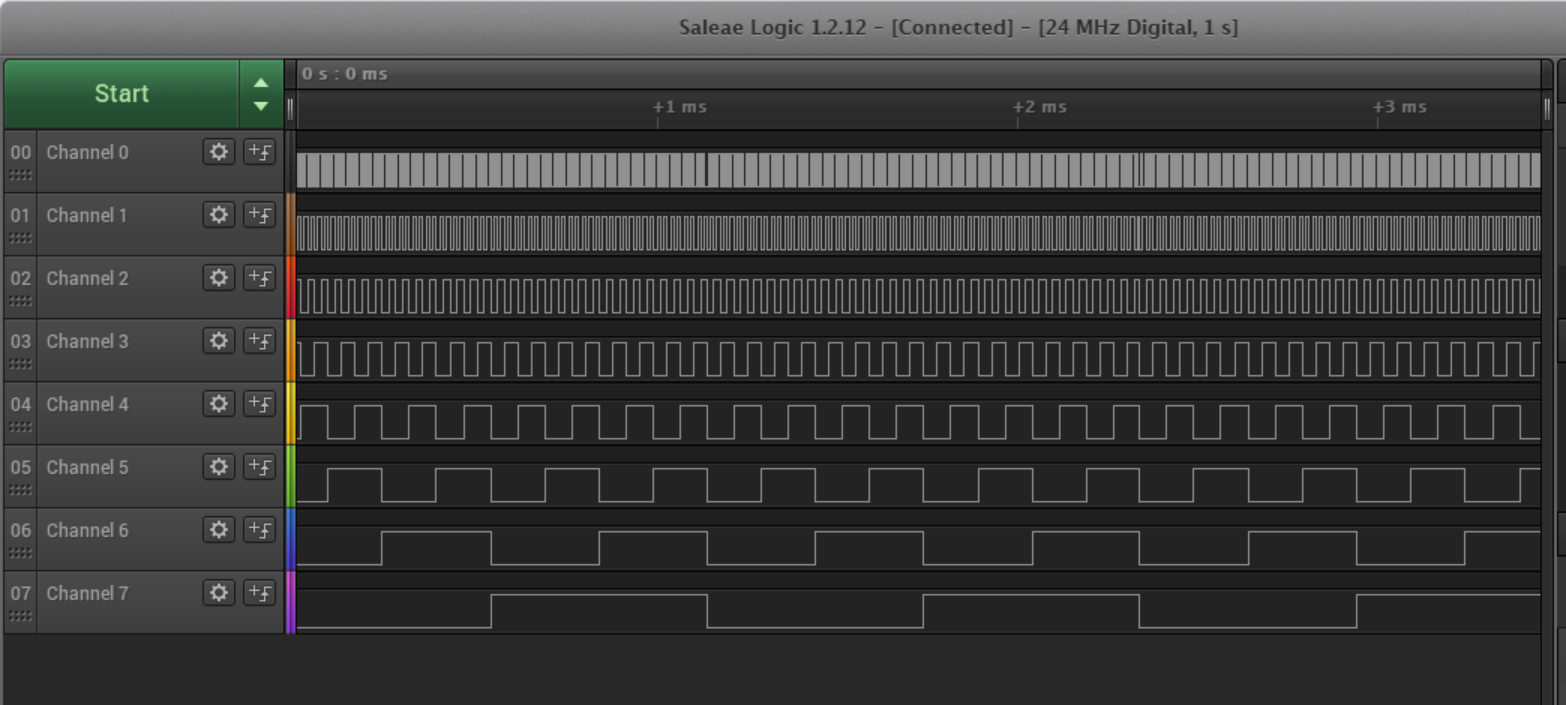
****

Рисунок 1 – Вывод на порт А

**Задание 2.1.** Настройка проекта. Подбор параметров задержки, пошаговая отладка.

В файле system\_MDR32F9Qx.c был создан пустой цикл в процедуре инициализации контроллера. Он необходим для задержки перед началом основной программы, чтобы избежать случайной блокировки контроллера. При цикле в 12 000 000 повторов наблюдается задержка в 12 секунд.

**Код программы:**

...

void SystemInit (void)

{

/\* Reset the RST clock configuration to the default reset state \*/

for (int i=0; i<12000000; i++);

/\* Reset all clock but RST\_CLK & BKP\_CLC bits \*/

MDR\_RST\_CLK->PER\_CLOCK = (uint32\_t)0x8000010;

...

**Задание 2.2.** Обработка сигнала от кнопок. Выполнение действия по нажатию

При нажатии на кнопку UP загорается светодиод 1, а при нажатии на кнопку DOWN загорается светодиод 2. Светодиоды горят пока кнопки удерживаются в нажатом состоянии.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit;

void LedPinCfg (void) //описание процедуры инициализации порта

{

//инициализация порта осуществляется через структуру типа PORT\_InitTypeDef

PortInit.PORT\_Pin = (PORT\_Pin\_0| PORT\_Pin\_1);//выбор вывода порта

PortInit.PORT\_OE = PORT\_OE\_OUT;//порт для вывода информации

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала

PORT\_Init(MDR\_PORTC, &PortInit);//инициализация порта

}

void ButtonsPinCfg(void){

PORT\_StructInit(&PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_5;

PortInit.PORT\_OE = PORT\_OE\_IN;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала

PortInit.PORT\_PULL\_DOWN = PORT\_PULL\_DOWN\_OFF;

PortInit.PORT\_PULL\_UP = PORT\_PULL\_UP\_ON;

PortInit.PORT\_GFEN = PORT\_GFEN\_ON;

PORT\_Init(MDR\_PORTB, &PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_1;

PORT\_Init(MDR\_PORTE, &PortInit);

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTE, ENABLE);

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTB, ENABLE);

LedPinCfg();//выполнение функции инициализации порта

ButtonsPinCfg();

//бесконечный цикл

while(1){

if(PORT\_ReadInputDataBit(MDR\_PORTB, PORT\_Pin\_5)){

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

} else {

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0);

}

if(PORT\_ReadInputDataBit(MDR\_PORTE, PORT\_Pin\_1)){

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_1);

} else {

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_1);

}

}

}

**Задание 2.3.** Обработка сигнала от кнопок. Изменение состояния по нажатию.

По нажатию на кнопку LEFT светодиод меняет своё состояние на противоположное.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit;

void LedPinCfg (void) //описание процедуры инициализации порта

{

//инициализация порта осуществляется через структуру типа PORT\_InitTypeDef

PortInit.PORT\_Pin = (PORT\_Pin\_0);//выбор вывода порта, в данном случае

PortInit.PORT\_OE = PORT\_OE\_OUT;//порт для вывода информации

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от

PORT\_Init(MDR\_PORTC, &PortInit);//инициализация порта

}

void ButtonsPinCfg(void){

PORT\_StructInit(&PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_3;

PortInit.PORT\_OE = PORT\_OE\_IN;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от

PortInit.PORT\_PULL\_DOWN = PORT\_PULL\_DOWN\_OFF;

PortInit.PORT\_PULL\_UP = PORT\_PULL\_UP\_ON;

PortInit.PORT\_GFEN = PORT\_GFEN\_ON;

PORT\_Init(MDR\_PORTE, &PortInit);

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTE, ENABLE);

LedPinCfg();//выполнение функции инициализации порта

ButtonsPinCfg();

int isOn = 0;

int isPressed = 0;

//бесконечный цикл

while(1){

if(isOn == 1){

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0);

} else {

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

}

if(!PORT\_ReadInputDataBit(MDR\_PORTE, PORT\_Pin\_3)){

if(isPressed == 0){

isOn = !isOn;

isPressed = 1;

}

} else {

isPressed = 0;

}

}

}

На рисунках 2 – 7 изображена отладка программы. На рисунке 8 показана работа программы в рабочем режиме с использованием логического анализатора.

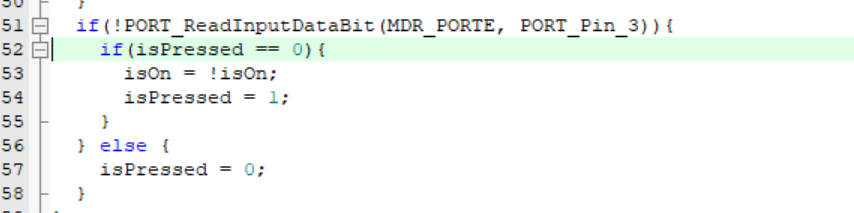


Рисунок 2 – Нажатие на кнопку

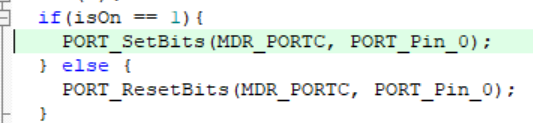


Рисунок 3 – Включение светодиода



Рисунок 4 – Светодиод включен

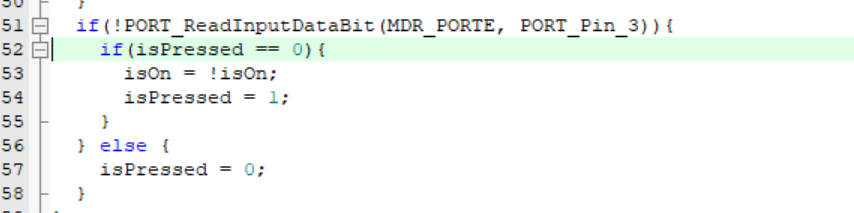


Рисунок 5 – Повторное нажатие на кнопку

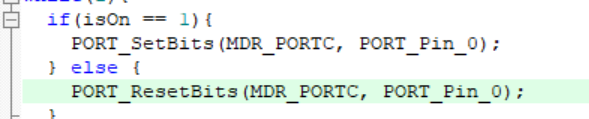


Рисунок 6 – Выключение светодиода

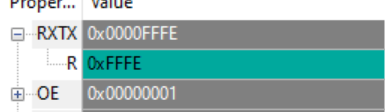


Рисунок 7 – Светодиод выключен

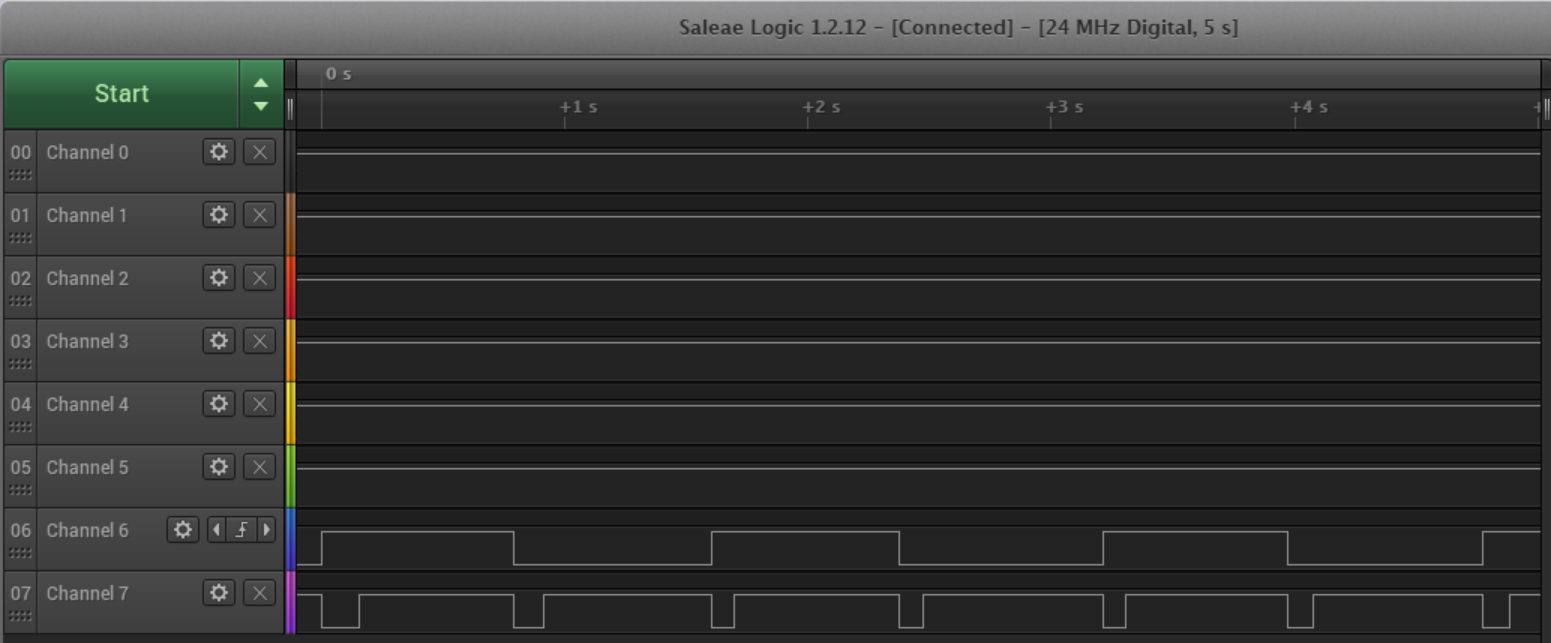


Рисунок 8 – Изменение состояния светодиода

**Задание 2.4.** Программная генерация импульсов определенной длины.

На рисунке 9 показана программная генерация импульсов.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit;

void Delay(int del){

for(int i = 0; i < del; i++);

}

void LedPinConfig(void){

PortInit.PORT\_Pin = PORT\_Pin\_0;

PortInit.PORT\_OE = PORT\_OE\_OUT;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;

PORT\_Init(MDR\_PORTB, &PortInit);

}

int main(void){

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTB, ENABLE);

LedPinConfig();

int T = 1000;//ПЕриод

int w = 100; //Длина импульса

while(1){

MDR\_PORTB->RXTX |= 0x01;

Delay(w);

MDR\_PORTB->RXTX &=~0x01;

Delay(T-w);

}

}



Рисунок 9 – Программная генерация импульсов

**Задание 2.5.** Программная генерация ШИМ

Реализовано увеличение ширины импульса при нажатии кнопки SELECT и уменьшение ширины импульса при нажатии кнопки RIGHT (рисунок 10).

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit;

void Delay(int del){

for(int i = 0; i < del; i++);

}

void LedPinConfig(void){

PortInit.PORT\_Pin = PORT\_Pin\_0;

PortInit.PORT\_OE = PORT\_OE\_OUT;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;

PORT\_Init(MDR\_PORTB, &PortInit);

}

void ButtonsPinCfg(void){

PORT\_StructInit(&PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_2;

PortInit.PORT\_OE = PORT\_OE\_IN;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала

PortInit.PORT\_PULL\_DOWN = PORT\_PULL\_DOWN\_OFF;

PortInit.PORT\_PULL\_UP = PORT\_PULL\_UP\_ON;

PortInit.PORT\_GFEN = PORT\_GFEN\_ON;

PORT\_Init(MDR\_PORTC, &PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_6;

PORT\_Init(MDR\_PORTB, &PortInit);

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTB, ENABLE);

LedPinConfig();

ButtonsPinCfg();

int isPressedSelect = 0;

int isPressedRight = 0;

int T = 5;//Период

int w = 1; //Длина импульса int

int step = 1; //Шаг изменения;

int counter = 0;

while(1){

if(counter < w){

MDR\_PORTB->RXTX |= 0x01;

} else {

MDR\_PORTB->RXTX &=~0x01;

}

if(counter == T) counter = 0;

if(!PORT\_ReadInputDataBit(MDR\_PORTC, PORT\_Pin\_2)){

if(isPressedSelect == 0){

if(w <= (T - step)) w+=step;

isPressedSelect = 1;

}

} else {

isPressedSelect = 0;

}

if(!PORT\_ReadInputDataBit(MDR\_PORTB, PORT\_Pin\_6)){

if(isPressedRight == 0){

if(w >= step) w-=step;

isPressedRight = 1;

}

} else {

isPressedRight = 0;

}

counter++;

}

}

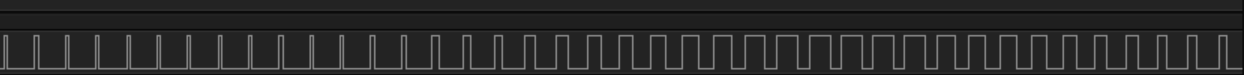


Рисунок 10 – Программная генерация ШИМ

**Задание 2.6.** Питание светодиода через ШИМ

При подаче питания светодиода через ШИМ наблюдается:

1. Периодическое мигание светодиода в соответствии с генерируемыми импульсами при большом периоде импульсов
2. Изменение яркости светодиода при изменении ширины импульса при малом периоде. Чем больше ширина импульса, тем выше яркость.

**Код программы:**

#include "MDR32F9Qx\_port.h"// подключение заголовочного файла библиотеки

#include "MDR32F9Qx\_rst\_clk.h" // подключение заголовочного файла библиотеки

static PORT\_InitTypeDef PortInit;

void Delay(int del){

for(int i = 0; i < del; i++);

}

void LedPinConfig(void){

//инициализация порта осуществляется через структуру типа PORT\_InitTypeDef

PortInit.PORT\_Pin = (PORT\_Pin\_0);//выбор вывода порта

PortInit.PORT\_OE = PORT\_OE\_OUT;//порт для вывода информации

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала

PORT\_Init(MDR\_PORTC, &PortInit);//инициализация порта

}

void ButtonsPinCfg(void){

PORT\_StructInit(&PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_2;

PortInit.PORT\_OE = PORT\_OE\_IN;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала

PortInit.PORT\_PULL\_DOWN = PORT\_PULL\_DOWN\_OFF;

PortInit.PORT\_PULL\_UP = PORT\_PULL\_UP\_ON;

PortInit.PORT\_GFEN = PORT\_GFEN\_ON;

PORT\_Init(MDR\_PORTC, &PortInit);

PortInit.PORT\_Pin = PORT\_Pin\_6;

PORT\_Init(MDR\_PORTB, &PortInit);

}

int main (void)//

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//включить тактирование

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTB, ENABLE);

LedPinConfig();

ButtonsPinCfg();

int isPressedSelect = 0;

int isPressedRight = 0;

int T = 10000;//Период

int w = 1000; //Длина импульса

int step = 300; //Шаг изменения;

int counter = 0;

while(1){

if(counter < w){

MDR\_PORTC->RXTX |= 0x01;

} else {

MDR\_PORTC->RXTX &=~0x01;

}

if(counter == T) counter = 0;

if(!PORT\_ReadInputDataBit(MDR\_PORTC, PORT\_Pin\_2)){

if(isPressedSelect == 0){

if(w <= (T - step)) w+=step;

isPressedSelect = 1;

}

} else {

isPressedSelect = 0;

}

if(!PORT\_ReadInputDataBit(MDR\_PORTB, PORT\_Pin\_6)){

if(isPressedRight == 0){

if(w >= step) w-=step;

isPressedRight = 1;

}

} else {

isPressedRight = 0;

}

counter++;

}

}

**Вывод**: в ходе работы были закреплены общие представления о выводах общего назначения, как интерфейсе периферийных устройств микропроцессорных систем, а также получены навыки настройки и создания проектов в среде Keil uVision для разработки программ на языке C.