

Министерство науки и высшего образования РФ  
Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
Кафедра «Информационная безопасность систем и технологий»

ОТЧЁТ

о выполнении лабораторной работы №4

«Аналогово-цифровой преобразователь микроконтроллера STM32 »

Дисциплина: Программирование  
микроконтроллеров

Группа: 21ПТ2

Выполнил студент: Юдин Н.М.

Отметка о сдаче (количество баллов):

Дата сдачи:

Принял: к.н.д Хворостухин С.П.

2023

## 1 Цель работы:

1.1 Ознакомиться с программными средствами работы с контроллером аналогово-цифрового преобразования в микроконтроллерах STM32.

## 2 Задания к лабораторной работе:

2.1 Создать проект в среде Keil uVision5.

2.2 Выполнить настройку режима отладки для проекта.

2.3 Разработать программу для микроконтроллера STM32F103RB, реализующую аналогово-цифровую обработку входного сигнала, показаний датчика температуры и канала опорного напряжения с использованием первого контроллера АЦП. Полученные значения выводить в терминал последовательного интерфейса один раз в секунду. Выводимая строка должна содержать:

- уровень входного сигнала, В;
- значение температуры, °С;
- уровень опорного напряжения, В.

Подачу входного аналогового сигнала и показаний датчика температуры реализовать через файл сценария с сигнальной функцией.

2.4 Выполнить симуляцию разработанной программы.

## 3 Результат выполнения:

3.1 Был создан проект в среде Keil uVision5.

3.2 Была выполнена настройка режима отладки для проекта.

3.3 Были реализованы согласно варианту задания программа и сигнальная функция. Код сигнальной функции приведен в приложении А. Код программы приведен в приложении Б. Ниже на рисунке 1 показан вариант задания.

Номер варианта	Номер канала АЦП	Закон изменения входного сигнала	Начальное значение температуры	Закон изменения температуры
3	3	Шумоподобный	+5°C	+0,3°C/с

Рисунок 1 – Вариант задания

В данной программе происходит инициализация портов для USART1 – PA9 и PA10. Прерывания по USART1 реализованы с помощью NVIC в функции void initNVIC(). Инициализируется USART1 инициализируется.

Реализована функция void Delay(uint32\_t takts) для произведения задержки через цикл for в функции инициализации ADC1 – void initADC1(). В функции void initADC1() происходит настройка ADC1 для принятия входного сигнала.

Для отправления данных строками реализована функция void sendToUSART(char \* string). Реализован обработчик USART - void USART1\_IRQHandler(). При нажатии в USART1 кнопки Enter обработчик принимает и преобразует сигнал, вольтаж и температуру и выводит их в окне USART1.

С помощью формулы на рисунке 2 из сигнала получается вольтаж.

$$V = \frac{DR \times V_{ref}}{2^{12}} \quad (1)$$

где:

- V - напряжение, В;
- DR - значение в регистре результата преобразования (ADC\_DR или ADC\_JDRx);
- Vref - значение опорного напряжения, В.

Рисунок 2 – Формула преобразования входящего сигнала в вольты

После получения вольтаж происходит расчёт температуры, с помощью формулы, приведенной ниже на рисунке 3.

$$T = \frac{(V_{25} - V_{SENSE})}{Avg\_Slope} + 25 \quad (2)$$

где:

- T - температура, ° C;
- V<sub>SENSE</sub> - напряжение, полученное контроллером АЦП с датчика температуры, В;
- V<sub>25</sub> - напряжение, соответствующее температуре +25°C, определяется электрическими характеристиками микроконтроллера, для STM32F103RB составляет 1,43 В;
- Avg\_Slope - среднее приращение температуры, определяется электрическими характеристиками микроконтроллера, для STM32F103RB составляет 4,3 мВ / °C.

Рисунок 3 – Формула преобразования вольтажа в температуру

3.4 Была выполнена симуляцию разработанной программы с использованием функций отладки и сигнальной функцией. Результат работы приведен на рисунке 4 и 5.

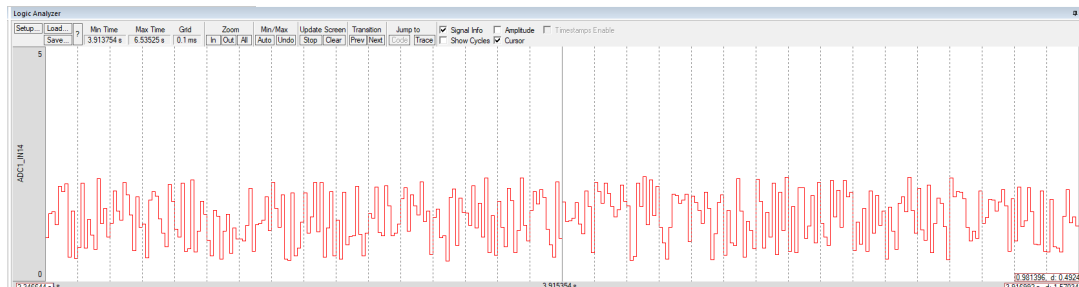
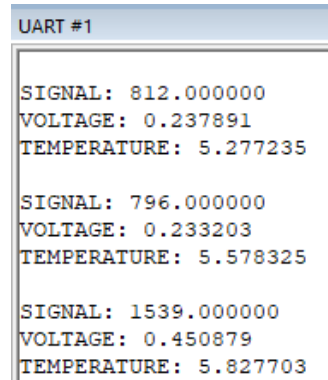


Рисунок 4 — Результат работы часть 1



```
UART #1  
  
SIGNAL: 812.000000  
VOLTAGE: 0.237891  
TEMPERATURE: 5.277235  
  
SIGNAL: 796.000000  
VOLTAGE: 0.233203  
TEMPERATURE: 5.578325  
  
SIGNAL: 1539.000000  
VOLTAGE: 0.450879  
TEMPERATURE: 5.827703
```

Рисунок 5 — Результат работы часть 2

4 Вывод: Было произведено ознакомление с программными средствами работы с контроллером аналогово-цифрового преобразования в микроконтроллерах STM32.

Приложение А

(обязательное)

Сигнальная функция

```
signal void sign(void) {
float volts, frequency, offset, val, steps;
long i;
volts = 2.0;
offset = 0.2;
frequency = 1400;
i = 0;
steps = (100000 * (1/frequency));
while (1)
{
val = ((float) rand(0)) / 32767.0;
ADC1_IN14 = (val * volts) + offset;
i++;
swatch (0.00001);
}
}
```

Приложение Б  
(обязательное)  
Реализация программы

```
#include "stm32f10x.h"
#include "stm32f10x_gpio.h"
#include "stm32f10x_rcc.h"
#include "stm32f10x_tim.h"
#include "stm32f10x_usart.h"

float S, V, T;
uint8_t temp_arr[10]; uint8_t temp; uint8_t temp_char;
float temperature_delta = 0.0;

void Delay(uint32_t takts)
{
    volatile uint32_t i;
    for (i = 0; i < takts; i++) {};
}

void initGPIO() {
    GPIO_InitTypeDef port;
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA
RCC_APB2Periph_USART1, ENABLE);
    port.GPIO_Pin = GPIO_Pin_9;
    port.GPIO_Speed = GPIO_Speed_50MHz;

    port.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOA, &port);
    port.GPIO_Pin = GPIO_Pin_10;
    port.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOA, &port);
}

void initNVIC() {
    NVIC_InitTypeDef nvic;
    nvic.NVIC_IRQChannel = USART1_IRQn;
    nvic.NVIC_IRQChannelPreemptionPriority = 0;
    nvic.NVIC_IRQChannelCmd = ENABLE;
    nvic.NVIC_IRQChannelSubPriority = 0;
    NVIC_Init(&nvic);
}

void initUSART1() {
    USART_InitTypeDef usart;
    usart.USART_BaudRate = 9600;
    usart.USART_WordLength = USART_WordLength_8b;
    usart.USART_StopBits = USART_StopBits_1;
    usart.USART_Parity = USART_Parity_No;
```

```

    usart.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    usart.USART_Mode = USART_Mode_Tx | USART_Mode_Rx;
    USART_Init(USART1, &usart);
    USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);
    USART_Cmd(USART1, ENABLE);
}

void initADC1(void)
{
    RCC->APB2ENR |= RCC_APB2ENR_IOPCEN;

    GPIOC->CRL &= ~(GPIO_CRL_MODE4 | GPIO_CRL_CNF4);

    RCC->APB2ENR |= RCC_APB2ENR_ADC1EN;

    ADC1->SMPR1 |= ADC_SMPR1_SMP14;

    ADC1->CR2 |= ADC_CR2_TSVREFE;
    ADC1->CR2 |= ADC_CR2_EXTSEL;
    ADC1->CR2 |= ADC_CR2_EXTTRIG;
    ADC1->CR2 |= ADC_CR2_ADON;

    Delay(5);
    ADC1->CR2 |= ADC_CR2_CAL;
    while (!(ADC1->CR2 & ADC_CR2_CAL)){};
}

uint16_t Read_ADC(uint8_t n)
{
    ADC1->SQR3 = n;
    ADC1->CR2 |= ADC_CR2_SWSTART;
    while(!(ADC1->SR & ADC_SR_EOC));
    return ADC1->DR;
}

void sendToUSART(char * string)
{
    uint8_t i = 0;
    while(string[i]) {

        USART_SendData(USART1, string[i]);
        while(!USART_GetFlagStatus(USART1, USART_FLAG_TXE)) {}
        i++;
    }
}

void USART1_IRQHandler() {
    if(USART_GetFlagStatus(USART1, USART_FLAG_RXNE)) {
        USART_ClearITPendingBit(USART1, USART_FLAG_RXNE);
        temp_arr[temp] = USART_ReceiveData(USART1);
    }
}

```



```

        temp_char = temp_arr[temp];
        temp++;
        if( temp_char != 0x0D ) {
            USART_SendData(USART1, temp_char);
        }
        if( temp_char == 0x0D ) {
            sendToUSART("\n");
            S = Read_ADC(14);
            char sss[20];
            sprintf(sss, "%f", S);
            sendToUSART("SIGNAL:      ");      sendToUSART(sss);
sendToUSART("\n");

            V = S * 1.2 / 4096;
            char vvv[20];
            sprintf(vvv, "%f", V);
            sendToUSART("VOLTAGE:      ");      sendToUSART(vvv);
sendToUSART("\n");

            T = (1.43 - V) / 4.3 + 5 + temperature_delta;
            temperature_delta += 0.3;
            char ttt[20];
            sprintf(ttt, "%f", T);
            sendToUSART("TEMPERATURE:  ");      sendToUSART(ttt);
sendToUSART("\n");
        }
    }
}

int main(void)
{
    initNVIC();
    initGPIO();
    initUSART1();
    initADC1();
    while(1){}
}

```