ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра вычислительной техники



Лабораторная работа №3

по дисциплине «Периферийные устройства информационных систем»

Изучение системы аналого-цифрового преобразователя и

системы прямого доступа к памяти.

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

Студент: Ванин К.Е.

Новосибирск 2021

**Цель работы:**

Получение студентами основных понятий об аналого-цифровых преобразователях современных микроконтроллеров с ядром Cortex M3 и системе прямого доступа к памяти. В ходе работы требуется изучить:

- базовые принципы аналого-цифрового преобразования;

- режимы работы аналого-цифрового преобразования: однократный с

программным запуском, непрерывный, непрерывные с автоматическим сканирование каналов и использованием прямого доступа к памяти;

- способы конфигурирования выводов GPIO микроконтроллера в качестве входов

аналого-цифрового преобразователя

**Задание:**

- изучить теоретический материал по аналого-цифровому преобразователю ADC и

системе прямого доступа к памяти DMA микроконтроллера STM32F103C8;

- доработать проект из Лабораторной работы 1 и 2, либо создать новый добавив

функции инициализации и управления преобразованием ADC1, использовать канал для

преобразования, подключенный к выводу PA1 или PA2, программный запуск

преобразования;

- откомпилировать проект. В режиме отладки поставить точку останова после

функции чтения из регистра данных ADC1. Изменяя положение джойстика, убедиться в

работоспособности проекта. Значения, считанные из регистра данных должны быть около

1960~2040 в среднем положении джойстика, 3~15 или 4070~4094 в крайних положениях

- реализовать то же действие, иcпользуя режим DMA

- доработать проект, увеличив число каналов ADC до 4-х, используя в качестве

источника данных выводы PA1-PA3 (PA3 подключен к кнопке джойстика) и канал 16

внутреннего температурного сенсора. Для чтения данных иcпользовать режим DMA;

- отладить встраиваемое приложение, используя точки останова для верификации

корректной оцифровки аналогового сигнала.

- оформить отчет с использованием скриншотов экрана работающего в режиме

отладки приложения.

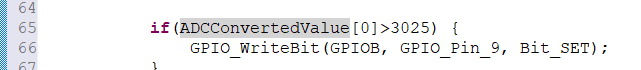
**Ход работы:**

Был изучен теоретический материал по аналого-цифровому преобразователю ADC и системе прямого доступа к памяти DMA микроконтроллера STM32F103C8.

Создан новый проект с добавлением функции инициализации и управления преобразованием ADC1, использован канал для преобразования, подключенный к выводу PA1 или PA2, произведен программный запуск преобразования;

При условии ADCConvertedValue[0]>3025, джойстик будет направлен вниз:

Рис. 1 - ADCConvertedValue[0]>3025



При условии ADCConvertedValue[0]<95, джойстик будет направлен вверх:

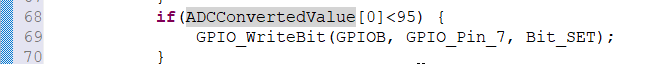


Рис. 2 - ADCConvertedValue[0]<95

При условии ADCConvertedValue[1]>3025, джойстик будет направлен влево:



Рис. 3 - ADCConvertedValue[1]>3025

При условии ADCConvertedValue[1]<95, джойстик будет направлен вправо:

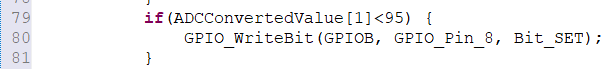


Рис. 4 - ADCConvertedValue[1]<95

При условии ADCConvertedValue[2]<800, джойстик будет в нажатии:

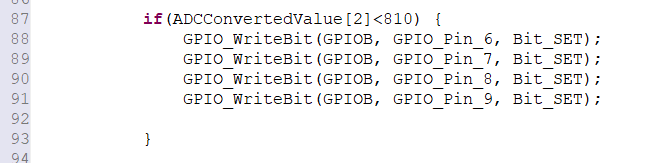


Рис. 5 - ADCConvertedValue[2]<810

**Листинг программы:**

**#include** "stm32f10x.h"

**#include** "stm32f10x\_adc.h"

**#include** "stm32f10x\_dma.h"

**#include** "stm32f10x\_gpio.h"

**#include** "stm32f10x\_rcc.h"

**#include** "stm32f10x\_exti.h"

**#include** "misc.h"

\_\_IO uint16\_t ADCConvertedValue[4];

**void** **GPIO\_Blink**(**void**)

{

**int** i;

GPIO\_InitTypeDef GPIO\_InitStructure;

EXTI\_InitTypeDef EXTI\_InitStructure;

NVIC\_InitTypeDef NVIC\_InitStructure;

ADC\_InitTypeDef ADC\_InitStructure;

DMA\_InitTypeDef DMA\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA|RCC\_APB2Periph\_GPIOB|RCC\_APB2Periph\_GPIOC, *ENABLE*);

RCC\_ADCCLKConfig(RCC\_PCLK2\_Div4);

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_ADC1 | RCC\_APB2Periph\_GPIOA, *ENABLE*);

RCC\_AHBPeriphClockCmd(RCC\_AHBPeriph\_DMA1, *ENABLE*);

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_1|GPIO\_Pin\_2|GPIO\_Pin\_3;

GPIO\_InitStructure.GPIO\_Mode = *GPIO\_Mode\_AIN*;

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

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

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

GPIO\_InitStructure.GPIO\_Mode = *GPIO\_Mode\_Out\_PP*;

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

ADC\_InitStructure.ADC\_Mode = ADC\_Mode\_Independent;

ADC\_InitStructure.ADC\_ScanConvMode = *ENABLE*; ADC\_InitStructure.ADC\_ContinuousConvMode = *ENABLE* ADC\_InitStructure.ADC\_ExternalTrigConv = ADC\_ExternalTrigConv\_None;

ADC\_InitStructure.ADC\_DataAlign = ADC\_DataAlign\_Right; ADC\_InitStructure.ADC\_NbrOfChannel = 4;

ADC\_Init(ADC1, &ADC\_InitStructure);

DMA\_DeInit(DMA1\_Channel1); DMA\_InitStructure.DMA\_PeripheralBaseAddr = ((uint32\_t)0x4001244C);

DMA\_InitStructure.DMA\_MemoryBaseAddr = (uint32\_t)&ADCConvertedValue;

DMA\_InitStructure.DMA\_DIR = DMA\_DIR\_PeripheralSRC; DMA\_InitStructure.DMA\_BufferSize = 4; DMA\_InitStructure.DMA\_PeripheralInc = DMA\_PeripheralInc\_Disable;

DMA\_InitStructure.DMA\_MemoryInc = DMA\_MemoryInc\_Enable;

DMA\_InitStructure.DMA\_PeripheralDataSize = DMA\_PeripheralDataSize\_HalfWord;

DMA\_InitStructure.DMA\_MemoryDataSize = DMA\_MemoryDataSize\_HalfWord;

DMA\_InitStructure.DMA\_Mode = DMA\_Mode\_Circular;

DMA\_InitStructure.DMA\_Priority = DMA\_Priority\_High;

DMA\_InitStructure.DMA\_M2M = DMA\_M2M\_Disable;

DMA\_Init(DMA1\_Channel1, &DMA\_InitStructure);

DMA\_Cmd(DMA1\_Channel1, *ENABLE*);

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_1, 1, ADC\_SampleTime\_55Cycles5);

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_2, 2, ADC\_SampleTime\_55Cycles5);

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_3, 3, ADC\_SampleTime\_55Cycles5);

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_TempSensor, 4, ADC\_SampleTime\_55Cycles5);

ADC\_DMACmd(ADC1, *ENABLE*);

ADC\_Cmd(ADC1, *ENABLE*);

ADC\_SoftwareStartConvCmd(ADC1, *ENABLE*);

**while** (1) {

**if**(ADCConvertedValue[0]>3025) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_9, *Bit\_SET*);

}

**if**(ADCConvertedValue[0]<95) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_7, *Bit\_SET*);

}

**if**(ADCConvertedValue[0]>950&ADCConvertedValue[0]<2000) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_9, *Bit\_RESET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_7, *Bit\_RESET*);

}

**if**(ADCConvertedValue[1]>3025) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_6, *Bit\_SET*);

}

**if**(ADCConvertedValue[1]<95) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_8, *Bit\_SET*);

}

**if**(ADCConvertedValue[1]>950&ADCConvertedValue[1]<2000) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_6, *Bit\_RESET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_8, *Bit\_RESET*);

}

**if**(ADCConvertedValue[2]<800) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_6, *Bit\_SET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_7, *Bit\_SET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_8, *Bit\_SET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_9, *Bit\_SET*);

}

**if**(ADCConvertedValue[2]>950) {

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_6, *Bit\_RESET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_7, *Bit\_RESET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_8, *Bit\_RESET*);

GPIO\_WriteBit(GPIOB, GPIO\_Pin\_9, *Bit\_RESET*);

}

}

}

**Вывод:**

Был изучен теоретический материал по аналого-цифровому преобразователю ADC и системе прямого доступа к памяти DMA микроконтроллера STM32F103C8.

Был создан проект с добавлением функции инициализации и управления преобразованием ADC1.

Был написан код для обработки каждого направления наклона джойстика с последующей индикацией с помощью светодиодов.

Произведена отладка встраиваемого приложения с использованием точек останов для верификации корректной оцифровки аналогового сигнала.