Лабораторная работа № 3: Настройка АЦП в микроконтроллере STM32.

Цель работы: научиться настраивать АЦП в микроконтроллере STM32 для измерения аналоговых сигналов.

Оборудование и программное обеспечение: перемычки для подключения источника напряжения (батарейки) ко входам АЦП

Теоретический материал

Для работы с АЦП микроконтроллера STM32 необходимо написать программу на языке программирования С. Рассмотрим основные команды и функции, которые используются для настройки и работы с АЦП.

1. ADC_Init(): функция инициализации АЦП, которая определяет режим работы АЦП, источник опорного напряжения, разрешение, режим запуска, входной канал АЦП и другие параметры.

2. ADC_RegularChannelConfig(): функция выбора входного канала АЦП, который будет использоваться для измерения аналоговых сигналов. В качестве аргумента функции передается номер входного канала.

3. ADC_Cmd(): функция запуска АЦП, которая запускает измерение сигнала, когда это необходимо.

4. ADC_GetConversionValue(): функция получения значения АЦП, которая возвращает цифровой код, представляющий значение измеренного аналогового сигнала.

5. ADC_InjectedChannelConfig(): функция выбора канала АЦП, который будет использоваться для измерения внешних опорных напряжений.

6. ADC_VoltageScaleConfig(): функция выбора источника опорного напряжения, который может быть внутренним или внешним.

7. ADC_InitTypeDef: структура, используемая для определения параметров АЦП, таких как режим работы, разрешение, режим запуска, входной канал АЦП.

Пример программы настройки АЦП в микроконтроллере STM32:

```
#include "stm32f10x.h"
  void ADC_Config(void)
  {
    ADC_InitTypeDef ADC_InitStructure;
    GPIO_InitTypeDef GPIO_InitStructure;

    // включение тактирования порта A
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    // настройка порта A.0 для подключения к АЦП
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AIN;
    GPIO_Init(GPIOA, &GPIO_InitStructure);
```

```
    // включение тактирования АЦП
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);
    // настройка АЦП
    ADC_InitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = DISABLE;
    ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_None;
    ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
    ADC_InitStructure.ADC_NbrOfChannel = 1;
    ADC_Init(ADC1, &ADC_InitStructure);
    // настройка входного канала АЦП (канал A.0)
    ADC_RegularChannelConfig(ADC1, ADC_Channel_0, 1, ADC_SampleTime_1Cycles5);
    // запуск АЦП
    ADC_Cmd(ADC1, ENABLE);
}

int main(void)
{
    // инициализация системы тактирования
    RCC_Configuration();
    // настройка АЦП
    ADC_Config();
    while(1)
    {
        // запуск преобразования АЦП
        ADC_SoftwareStartConvCmd(ADC1, ENABLE);
        // ожидание завершения преобразования
        while(ADC_GetFlagStatus(ADC1, ADC_FLAG_EOC) == RESET);
        // получение результата преобразования
        int value = ADC_GetConversionValue(ADC1);
        // обработка результата преобразования
        ...
    }
}
```

Это базовый пример программы для настройки и использования АЦП в микроконтроллере STM32. Можно добавить дополнительные функции и условия в соответствии с требованиями приложения и настраивать АЦП для измерения разных аналоговых сигналов.

Ход работы

1. Подготовка необходимых компонентов - микроконтроллер STM32, датчик температуры, переменный резистор, плату для подключения компонентов.

2. Подключение датчика температуры к входу АЦП микроконтроллера.

3. Написание программы на языке программирования C для настройки АЦП и измерения температуры.

4. Настройка АЦП на работу в режиме одиночного измерения.

5. Выбор внутреннего источника опорного напряжения.

6. Настройка разрешения АЦП на 12 бит.

7. Настройка входного канала АЦП на подключенный датчик температуры.

8. Определение формулы для преобразования значения АЦП в температуру.

9. Запуск АЦП и получение значения температуры с датчика.

10. Отображение полученного значения температуры на LCD дисплее или экране ПК.

11. Проведение нескольких измерений и сравнение результатов.

Индивидуальные задания

1. Написать программу, которая будет использовать 6 каналов АЦП микроконтроллера STM32 для измерения аналоговых сигналов. Каждый канал должен быть настроен на разрешение 12 бит, а режим работы АЦП должен быть выставлен в режим одиночного преобразования.

2. Написать программу, которая будет использовать встроенный в STM32 опорный источник напряжения для калибровки АЦП. Программа должна измерять опорное напряжение с помощью внешнего вольтметра, затем устанавливать измеренное значение как значение опорного напряжения и запускать АЦП для измерения других аналоговых сигналов.

3. Написать программу, которая будет использовать внешний источник опорного напряжения для АЦП. Программа должна измерять аналоговый сигнал, а затем использовать опорное напряжение с внешнего источника для преобразования аналогового сигнала в цифровое значение. Разрешение АЦП должно быть не менее 10 бит.

4. Написать программу, которая будет использовать встроенный компаратор STM32 для сравнения значений, полученных от нескольких входов АЦП. Если какой-либо вход АЦП будет считаться "высоким" (то есть значение больше заданного порога), то компаратор должен выдать сигнал на соответствующем выходном порту, или наоборот.