Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

**”ADC”**

**Вариант №7**

|  |  |  |  |
| --- | --- | --- | --- |
| Преподаватель |  | подпись, дата | Яблонский А.П. |
| Студент КИ19-09Б | 031940750  номер зач. книжки | подпись, дата | Кудрявцев Н.М. |

Красноярск 2022 г.

**ВВЕДЕНИЕ**

**Цель работы**: изучить операции по работе с АЦП, режимы работы АЦП, настройка и выбор каналов АЦП, регистры настройки и управления АЦП.

**Задание**: необходимо разработать программу, в которой реализовать оцифровку данных, заданное количество раз, с седьмого канала АЦП (циклическое или по требованию, в зависимости от варианта). Данные из АЦП необходимо сохранить во внутреннюю память контроллера, и по требованию вывести поочередно на индикаторы. Временную задержку между отображением данных реализовать с помощью таймера (ТС0).

Таблица 1 ─ Вариант на работу.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Вариант** | **Режим АЦП** | **Запуск АЦП** | **Адрес памяти** | **Кол-во оцифровок** | **Вывод данных** |
| **7** | Однократный | INT1 | 0x95 | 10 | INT0 |

**Команды:**

Справка по Ассемблеру для Atmel AVR:

<https://dfe.karelia.ru/koi/posob/avrlab/avrasm-rus.htm>

**MCUCR** ─ регистр прерываний для INT0 и INT1;

**SPL и SPH** ─ регистры с адресом вершины стека;

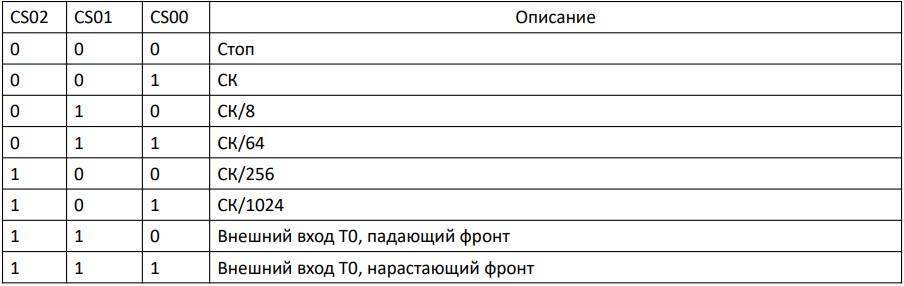
**TIMSK** ─ Регистр маски прерывания таймеров/счетчиков;

**TCNT0** ─ Регистр данных таймера/счетчика 0

**ADMUX** ─ Управление мультиплексором модуля АЦП;

**TCCR1B** ─ типо счётчик;

**TCCR0** ─ Управляющий регистр таймером/счетчиком 0;

****

**ADCSRA** ─ устанавливаются единицы в битах:

* **Бит ADEN** разрешает функционирование модуля АЦП. Если он установлен, то модуль активен, если сброшен, то, соответственно, неактивен.
* **Бит** **ADSC** запускает преобразование. В режиме одиночного преобразования именно установка этого бита в "1" стартует преобразование, и далее модуль АЦП ожидает очередной установки его в "1". В режиме непрерывного преобразования установка этого бита определяет старт первого преобразования, а все последующие уже не зависят от состояния бита ADSC.
* **Бит** **ADATE** как раз и определяет, в каком режиме будет работать модуль АЦП. Если он равен "0", то устанавливается режим одиночного преобразования, а если "1", то режим непрерывного преобразования.
* **Бит** **ADIE** разрешает генерацию прерывания по завершению цикла преобразования АЦП. Поскольку мы собираемся использовать именно это прерывание, то в программе мы и устанавливаем данный бит.
* **Биты** **ADPSx** (х = 0, 1, 2) Определяют коэффициент деления тактовой частоты контроллера для тактирования модуля АЦП.

**Ход работы:**

.def temp = r16

.def temp2 = r17

.def diods = r18

.def mode = r19

.def countADC = r20

.def countWait = r21

.set number\_of\_samples = 10 // 10 оцифровок

.dseg

.org 0x95

mas\_ADC: .byte number\_of\_samples

.cseg

.org $000 rjmp INIT

.org INT0addr rjmp INT\_0

.org INT1addr rjmp INT\_1

.org OVF1addr rjmp T\_OWF1

.org OVF0addr rjmp T\_OWF0

.org ADCCaddr rjmp Int\_ADC

.macro mas

ldi @0, low(@2)

ldi @1, high(@2)

.endmacro

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

INIT:

ldi temp, low(RAMEND) ; настройка указателя стека

out SPL, temp

ldi temp, high(RAMEND)

out SPH, temp

clr mode

ldi temp,0x7F ; 7 пин АЦП

out DDRA,temp ; на ввод

; настройка порта B на вывод светодиода

ser temp

out DDRB, temp

out PORTB, temp

; настройка прерываний

; настраиваем на срабатывание INT0 и INT1 по переднему фронту

ldi temp, (1<<ISC11)|(1<<ISC10)|(1<<ISC01)|(1<<ISC00)

out MCUCR, temp

; разрешение внешних (глобальных) прерываний

ldi temp, (1<<INT1)|(1<<INT0)

out MCUCR, temp

; настройка таймера 0

; Разрешение прерываний по переполнению счета

ldi temp, (1<<TOIE0)|(1<<TOIE1)

out TIMSK, temp

ldi temp, 0 ; задать число начального счета

out TCNT0,temp

; настройка ADC на 7 канал (MUX), выравнивание по левому (ADLAR) краю включить

; REFS1:REFS0. Биты выбора опорного напряжения.

; ADLAR - выравнивания результата по какому-то краю

; MUX0, MUX1, MUX2 - биты регистра ADMUX

ldi temp, (0<<REFS1)|(0<<REFS0)|(1<<ADLAR)|(1<<MUX2)|(1<<MUX1)|(1<<MUX0)

out ADMUX, temp

ldi temp,(1<<ADEN)|(0<<ADSC)|(0<<ADATE)|(0<<ADIF)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)

out ADCSRA,temp

sei ; разрешить прерывания

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

main:

SBRC mode, 0 ; проверка на 1 режим работы

call mode0 ; вызов АЦП

SBRC mode, 1 ; проверка на 2 режим работы

call mode1 ; вызов вывода массива ADC

rjmp main ; бесконечный цикл

INT\_0:

ldi mode, 0b00000010

reti

INT\_1:

ldi mode, 0b00000001

reti

mode0:

clr mode ; выключение, чтобы режим отработал 1 раз

; настройка массива

mas xl, xh, mas\_ADC

ldi countADC, number\_of\_samples

sbi ADCSRA,ADSC ; старт ADC

ldi temp,(0<<CS12)|(1<<CS11)|(0<<CS10)

out TCCR1B,temp

RET

T\_OWF1:

sbi ADCSRA,ADSC ; старт ADC

reti

Int\_ADC:

; вывод отработки ADC

in diods, ADCL ; Считываем сперва младший, а затем старший, даже если младший не нужен

in diods, ADCH

; ввод числа в массив

st x+, diods

;вывод

com diods

out PORTB, diods

; счет элементов массива

dec countADC

BRNE END\_Int\_ADC ; если заполнили массив, закончить выполнения

; закончить выполнения

cbi ADCSRA,ADSC ; выкл ADC

clr temp

out TCCR1B,temp

ser diods

out PORTB, diods

END\_Int\_ADC:

reti

mode1:

clr mode ; выключение, чтобы режим отработал 1 раз

; настройка массива

mas xl, xh, mas\_ADC

ldi countADC, number\_of\_samples

; включаем таймер...

ldi temp, (1<<CS02)|(0<<CS01)|(1<<CS00)

out TCCR0, temp ; ...частотой СК/1024

ldi countWait, 4 ; счетчик чтобы обеспечить дополнительную задержку

RET

T\_OWF0:

; отчитывание для секунды

dec countWait

BRNE END\_T\_OWF

ldi countWait, 4

; вывод числа масива на диоды

ld diods, x+

com diods

out PORTB, diods

dec countADC ; счет элементов массива

BRNE END\_T\_OWF ; если прошли масив, закончить выполнения

ser diods ; выключаем светодиоды

out PORTB, diods

clr temp ; выключить Timer 0

out TCCR0,temp ; стоп таймер

END\_T\_OWF:

reti

//inf:

// rjmp inf