**Лабораторная работа №4**

**Тема:** Использование АЦП для обработки данных от клавиатуры

**Задание:** 1. Реализовать опрос 8 клавиш с помощью одного канала АЦП (ADC0). Реализовать процедуру антидребезга (частота опроса 100-200 Гц, количество опросов в одном цикле антидребезга - 10).

2. Реализовать индикацию номера нажатой клавиши с помощью матричного индикатора, строки которого управляются выводами PC7-PC0, а столбцы - выводами PB7-PB0. Тактовая частота МК - 8 МГц.

**Ход работы**

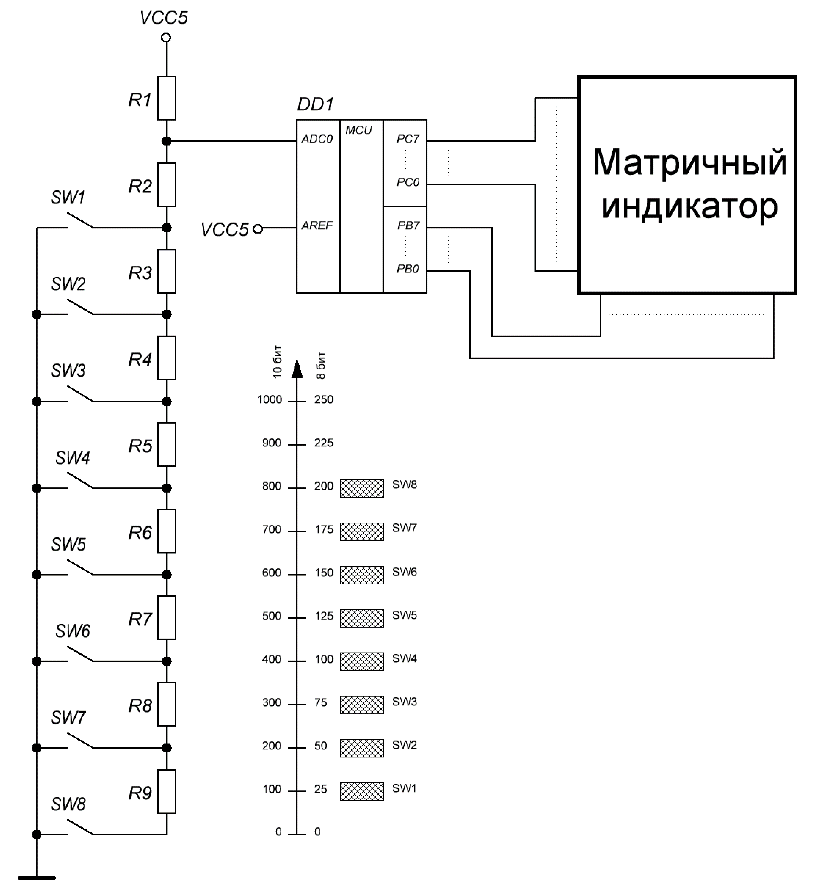
****

Рис.1 Схема матричного индикатора

Согласно схеме при нажатии клавиш изменяется напряжение на выводе ADC0. При отсутствии нажатых клавиш напряжение будет около напряжения питания CC5. Сопротивление резисторов подобраны таким образом, чтобы получить равномерное распределение получающихся напряжений по шкале. Если одновременно нажато 2 или более клавиш, то определяющей является нажатая клавиша с меньшим номером, потому что часть резистивного делителя ниже этой клавиши никак не влияет на получающееся напряжение.

Для опроса клавиатуры надо будет настроить АЦП и какой-нибудь таймер. АЦП можно использовать в режиме одиночного преобразования. Частота прерывания таймера - согласно частоте опроса. Т.е. по прерыванию таймера сначала получаем результат предыдущего измерения, а потом запускаем очередное измерение. Причем можно использовать как 10-битный результат, так и "округление" этого результата до 8 бит (т.е. младшие 2 бита отбрасываются). На картинке показаны цифры для обоих вариантов (10 бит и 8 бит). Результат каждого измерения сравнивается с пороговыми значениями напряжения (на картинке зоны возможных значений напряжения при нажатии той или иной клавиши показаны заштриховаными прямоугольниками). Если напряжение попало в одну из зон, то фиксируем номер клавиши, если не попало ни в одну из зон, то фиксируем ошибку (для простоты примем, что в таком случае ни одна клавиша не нажата). По этим данным должна отработать процедура антидребезга. Т.е. должно быть не менее 10 одинаковых значений для того, чтобы это значение можно было считать стабильным и использовать далее.

Индикацию можно взять из лабы 3. Откуда нужна только динамическая индикация. Эффект бегущей строки не нужен. Для того, чтобы можно было пользоваться имеющейся таблицей перекодировки, нужно к получающемуся знаяению номера нажатой клавиши (1-8) добавлять 48 и только потом перекодировать. Если ни одна клавиша не нажата, то выводится пробел или прочерк

**Код программы:**

#include <avr/io.h> //подключение стандартной библиотеки ввода/вывода

int main(void)

{

// инициализация таймера 0

TCCR0 = 0;

TCCR0 = (0<<WGM01)|(0<<WGM00); //Timer Mode of Operation = Normal

TCCR0 |= (0<<CS02)|(1<<CS01)|(1<<CS00); //предделитель 64, f(T0)=490 Гц

TIFR |= (1<<TOV0); //Clear Timer0 Overflow Flag

TIMSK |= (1<<TOIE0); //Timer0 Overflow Interrupt Enable

//инициализация АЦП

SFIOR |= (0<<ADTS0)|(0<<ADTS1)|(1<<ADTS2); //Выбор прерывания по переполнению таймера Т0

ADMUX |= (0<<REFS1)|(0<<REFS0) //выбор источника опорного напряжения

ADMUX |= (1<<ADLAR) //Настройка выравнивания результата влево

ADCSR |= (1<<ADPS2)|(1<<ADPS1)|(0<<ADPS0) //Настройка предделителя 64, f(ADC) = 125 кГц

ADCSRA |= (1<<ADEN) //Разрешение АЦП 1-включено

//инициализация остальной периферии

DDRB |= 0xFF;

DDRC |= 0xFF;

unsigned char dreb\_count=0;

\_\_enable\_interrupt();

while(1);

return 0;

}

unsigned int adc\_value; //переменная для хранения результата преобразования

ISR(TIMER0\_OVF\_vect) //обработчик прерывания Таймера 0

{

TCNT0 = 0; //перезапись счетного регистра

indication;

}

ISR(ADC\_vect) //обработчик прерывания по преобразованию

{

//После завершения преобразования (при установке в «1» флага ADIF регистра ADCSR)

//его результат сохраняется в регистре данных АЦП

adc\_value = ADCL; adc\_value=ADCH; // Считываем ADCH, откидывая 2 младших бита из ADCL

//u = (ADCL|ADCH << 8); // Считываем ADC

if (adc\_value==!0)

{

if ((adc\_value >= 1) && (adc\_value <= 15)) //no SW

{

n\_ind=0;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 20) && (adc\_value <= 30)) //SW1

{

n\_ind=1;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 45) && (adc\_value <= 60)) //SW2

{

n\_ind=2;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 70) && (adc\_value <= 85)) //SW3

{

n\_ind=3;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 95) && (adc\_value <= 110)) //SW4

{

n\_ind=4;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 120) && (adc\_value <= 135)) //SW5

{

n\_ind=5;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 140) && (adc\_value <= 160)) //SW6

{

n\_ind=6;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 170) && (adc\_value <= 185)) //SW7

{

n\_ind=7;

if (!buttoncount(n\_ind))

n\_ind=0;

}

elseif ((adc\_value >= 195) && (adc\_value <= 215)) //SW8

{

n\_ind=8;

if (!buttoncount(n\_ind))

n\_ind=0;

}

else

{

n\_ind=0;

}

// }

}

//достаточно точности 8 разрядного значения, поэтому для получения результата

//можно прочитать только содержимое регистра ADCH

//ADCSRA |= (1<<ADSC) //Запуск преобразования 1-начать преобразование

}

unsigned short RecodingArray[2048];

#define ROWS PORTC //порт «рядов» дисплея

#define COLS PORTB //порт управления «столбцами»

int indication; //процедура индикации

{

{

COLS = 0x00;

ROWS = RecodingArray[pos+n\_ind\*8+48\*8];

COLS = (1 << pos);

If (++pos == 8) pos = 0;

}

}

byte count=0;

byte n\_ind=0; //номер нажатой клавиши

byte n\_last; //номер предыдущей клавиши

int buttoncount(byte n\_ind) //процедура антидребезга

{

if (n\_ind==n\_last)

{

if (count<10)

{

count++;

return 0;

}

else return 1;

}

else

{

count=0;

n\_last=n\_ind;

return 0;

}

}

**Вывод:** в данной лабораторной работе мы научились реализовать опрос 8 клавиш с помощью одного канала АЦП (ADC0), процедуру антидребезга, индикацию номера нажатой клавиши с помощью матричного индикатора.