

ATmega2560 ADC channels

Asked 10 years, 9 months ago Modified 2 years, 2 months ago

Viewed 3k times



0



I have been using ATmega128 and I am now looking a bit into ATmega2560. I have tried ADC in 128 already. When I tried it in 2560 I don't have problem except when I am changing it. I wrote this code in codevision. When using only single channel it works fine but when changing channels there is erratic behaviour. What is the problem ?

Here the code

```
void ADC_init(void)
{
    PINF=0;
    PORTF=0;
    PINK=0;
    PORTK=0;
    ADMUX  = 0b01000000;
    ADCSRA =0x87; //ADEN=1,ADIE=0
    ADCSRB=0x00;
}

void ADC1_read(void)// current - ADC1
{
    ADMUX=0x41;
    ADCSRA|=0x40; //ASSC=1 (ad_start)
    ADCSRB=0x00;
    while((ADCSRA&(1<<ADSC)));
    //while((ADCSRA&0x10)==0);
    ADCSRA |= (1<<ADIF);
    cur_l=ADCL;
    cur_h=ADCH;
    cur_buf=cur_h;//0x03;
```

```

        cur_buf=(cur_buf<<8)|cur_l;
        cur_vol = (unsigned long)(((unsigned
long)cur_buf*500)/1023);
    }

void ADC3_read(void) //temp sensor - ADC3
{
    unsigned long temp_volt;
    ADMUX = 0xC3;
    ADCSRA|=0x40; //ASSC=1 (ad_start)
    ADCSRB=0x00;
    while((ADCSRA&0x10)==0);
    ADCSRA |= (1<<ADIF);
    temp_l=ADCL;
    temp_h=ADCH;
}

```

The process function inside Main is like this :

```

void process(void)
{
    static unsigned char tick_2sec=0, tick_3sec=0;
    unsigned char rx2;

    if (tick_1sec){
        tick_2sec++;
        tick_3sec++;
        tick_1sec = 0;
    }
    if (tick_2sec == 2){
        tick_2sec = 0;
        if (t2 == 1){
            t2 = 0;
            ADC3_read(); //temp
        }
    }
    if (tick_3sec == 3){
        tick_3sec = 0;
        if (t3 == 1){
            t3 = 0;
            ADC1_read(); //current
        }
    }
}

```

```
}  
}
```

I used temperature sensor TMP36 to input ADC3 and at 20 C , output voltage is about .7 V. When only ADC 3 is used with 2.56 V reference voltage, the value acquired from ADCH and ADCL is 0x118. Likewise, I connected ADC1 to voltage source that outputs 2.48 volt at normal condition. When I used only ADC1 with reference voltage 5 volt, the output from ADCH and ADCL is 509. And with change in voltage, the ADCH and ADCL registers also change correspondingly. When I called the functions to read ADC1 and ADC3 at intervals of 2 and 3 seconds, as shown in code, at same voltage the ADCH and ADCL contents were different from above. ADC3 had ADCH,L as 137-140 and ADC1 had ADCH,L as 340-352.

avr

adc

Share Follow

edited Mar 28, 2014 at 0:08

asked Mar 27, 2014 at 14:50



avr_rookie

57 ● 3 ● 10

Can you provide more information about the "erratic behavior"? – Glenn Mar 27, 2014 at 15:00

I have posted them above. – [avr_rookie](#) Mar 28, 2014 at 0:46

3 Answers

Sorted by:

Highest score (default)



1



I have run into similar problems in the past when reading from different ADC pins. It turned out the last reading affected the current reading. The current value gets skewed higher or lower towards what the last reading was.



I solved my problem at the time by taking readings from the same pin twice, and only using the second one.



I have since learned that the problem probably comes from capacitance in the circuits. Some residual charge stays in the circuit from the previous voltage which affects the voltage on the next reading. I suspect adding capacitors to the devices being read might overcome this, but I never had a chance to test it out.

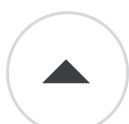
Share Follow

answered Mar 28, 2014 at 19:58



[UncleO](#)

8,449 ● 23 ● 30



0

As any ADC of this kind, the AVR ADC use a "Sample and Hold" input configuration. The "Hold" function is implemented using a capacitor. When you switch from one channel to another, this "Hold" capacitor needs time



to settle to the new value. This time is depending of your hardware/software configuration (Gain setting, input source impedance and so on.) This is why reading the value twice solves your issue, some additional time is given to the "Hold" capacitor to settle to the new value. Selecting the channel then waiting a bit should provide the same result.

Share Follow

answered Apr 27, 2017 at 15:34



Zeboss60

1



0




You can use this:

```
uint16_t read_ADC(uint8_t ADCchannel)
{
    if(ADCchannel<8){
        ADCSRB &= ~(1<<MUX5);
        ADMUX = (ADMUX & 0xF0) | (ADCchannel &
0x0F);
    }
    if(ADCchannel==8){
        ADCSRB |= (1<<MUX5);
        ADCSRB |= (uint16_t)ADCchannel;
    }
    //single conversion mode
    ADCSRA |= (1<<ADSC);
    // wait until ADC conversion is complete
    while( ADCSRA & (1<<ADSC) );
    // 0-1023
    return ADC;
}
```

Share Follow

edited Oct 20, 2022 at 6:29

 **Yunnosch**
26.6k ● 9 ● 44 ● 62

answered Oct 20, 2022 at 6:27

 **antango**
1

Welcome to Stack Overflow! While this code may solve the question, [including an explanation](#) of how and why this solves the problem would really help to improve the quality of your post, and probably result in more up-votes. Remember that you are answering the question for readers in the future, not just the person asking now. Please [edit](#) your answer to add explanations and give an indication of what limitations and assumptions apply. – **Yunnosch** Oct 20, 2022 at 6:29

Your answer could be improved with additional supporting information. Please [edit](#) to add further details, such as citations or documentation, so that others can confirm that your answer is correct. You can find more information on how to write good answers [in the help center](#). – **Community Bot** Oct 24, 2022 at 11:19
