[TUT] [C] Newbie's Guide to the AVR ADC

Topic locked Go To Last Post

326 posts / 0 new

Author Message

penquissciguy

Level: New Member
Joined: Sat. Sep 9, 2006
Posts: 7 View posts
Location: Maine

Posted by penquissciguy: Fri. Nov 16, 2007 - 08:45 PM #1

Fivestar widget

Total votes: 0

I wanted to learn more about the ADC on my ATMega128, so I dug into the docs and taught myself a couple things. I thought since there wasn't a tutorial on this topic I'd write one. Hope this helps someone out. Comments, clarifications and constructive criticism welcomed!

Newbie's Guide to the AVR ADC
© Ken Worster

What is an ADC?

An ADC, or Analog to Digital Converter, allows one to convert an analog voltage to a digital value that can be used by a microcontroller. There are many sources of analog signals that one might like to measure. There are analog sensors available that measure temperature, light intensity, distance, position, and force, just to name a few.

Introduction – The AVR ADC

The AVR ADC allows the AVR microcontroller to convert analog voltages to digital values with few to no external parts. The author wrote this tutorial with the ATMega128 in mind, though other AVRs use similar hardware. The ADC built into the ATMega128 is capable of 10 bit resolution. The ATMega128 microcontroller has 8 ADC channels, allowing up to 8 analog sources to be attached to the microcontroller. The 8 ADC channels are connected to the internal DAC through a device called a multiplexer. The multiplexer connects the 8 ADC channels (the 8 pins of Port F on the ATMega128) to the internal ADC. One channel at a time is passed through the multiplexer to the ADC. The ADC has its own power supply, labeled AVCC, on the ATMega128. This pin needs to be connected to a power source within .3 volts of the chip's VCC supply. Most of the time, you would wire this to VCC. With the 10 bit DAC, this allows measuring voltages from 0 to 5 volts with a resolution of 5/1024 volts, or 4.88 mV.

The ADC channels in the ATMega128 can be configured in several different ways. In this tutorial, the channels are being used in "single-ended" mode. In this mode, the analog voltages presented on the ADC channels are compared to ground. There are several selectable voltage references, which determine the range of the ADC conversion. In this tutorial, AVCC is used as the voltage reference. The ADC can also be set to run continuously (the free-running mode) or to do only one conversion. The first example in this tutorial uses the free-running mode to continuously update the ADC reading.

Part 1 – A Simple Free-Running ADC Example

This example uses the simplest variable voltage source I could think of – a potentiometer. I wired up a 10k potentiometer on a breadboard as in the example below.

The two outside terminals were attached to 5 volts and ground, with the center terminal attached to the first ADC channel. The two 330 ohm resistors protect the microcontroller pin from being shorted to ground or to 5 volts at the edges of the potentiometer's travel. With this setup, turning the potentiometer will give you a range of .15 volts to 4.85 volts between ground and ADC0. In order to read the voltage of this circuit, its ground and the ground of the microcontroller need to be connected.

To give an indication of the value the ADC is reading, two LEDs are hooked to the microcontroller. We can toggle these to give us a "high" or "low" indication. Here is the pseudocode for this example:

```
Set up output LEDs
Configure ADC Hardware
Enable ADC
Start A2D Conversions

WHILE Forever
       IF ADC Value High, Turn on LED1
       ELSE Turn on LED2
END WHILE
```

To simplify this example, we will set up the ADC to continuously measure the voltage on ADC0. We will then poll the value in an endless loop and change the LEDs' statuses as we need to. The skeleton code for our example would then be

```
#include
```

```
int main (void)
{
        DDRE |= (1 << 2); // Set LED1 as output
        DDRG |= (1 << 0); // Set LED2 as output

        // TODO:  Configure ADC Hardware
        // TODO: Enable ADC
        // TODO: Start A2D Conversions

for(;;)  // Loop Forever
        {
                // TODO: Test ADC Value and set LEDs
        }
}
```

Setting up the LEDs is outside the topic of this tutorial, so the code to set them up is shown above without explanation. You can use any unused i/o line for the LEDs. Check out the “Programming 101” tutorial on the AVRFreaks forum for more information on this if you need it.

The next step is to configure the ADC hardware. This is done through setting bits in the control registers for the ADC. First, let's set the prescalar for the ADC. According to the datasheet, this prescalar needs to be set so that the ADC input frequency is between 50 KHz and 200 KHz. The ADC clock is derived from the system clock. With a system frequency of 16 MHz, a prescaler of 128 will result in an ADC frequency of 125 Khz. The prescaling is set by the ADPS bits in the ADCSRA register. According to the datasheet, all three ADPS bits must be set to get the 128 prescaler.

```
ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0);
```

Next, let's set the ADC reference voltage. This is controlled by the REFS bits in the ADMUX register. The following sets the reference voltage to AVCC.

```
ADMUX |= (1 << REFS0);
```

To set the channel passed through the multiplexer to the ADC, the MUX bits in the ADMUX register need to be set accordingly. Since we are using ADC0 here, which corresponds with all 5 MUX bits being zero, we don't need to set anything here.

In order to put the ADC into free-running mode, set the aptly-named ADFR bit in the ADCSRA register:

```
ADCSRA |= (1 << ADFR);
```

One last settings change will be made to make reading the ADC value simpler. Though the ADC has a resolution of 10 bits, this much information is often not necessary. This 10 bit value is split across two 8 bit registers, ADCH and ADCL. By default, the lowest 8 bits of the ADC value are found in ADCL, with the upper two being the lowest two bits of ADCH. By setting the ADLAR bit in the ADMUX register, we can left align the ADC value. This puts the highest 8 bits of the measurement in the ADCH register, with the rest in the ADCL register. If we then read the ADCH register, we get an 8 bit value that represents our 0 to 5 volt measurement as a number from 0 to 255. We're basically turning our 10 bit ADC measurement into an 8 bit one. Here's the code to set the ADLAR bit:

```
ADMUX |= (1 << ADLAR);
```

That completes the setup of the ADC hardware for this example. Two more bits need to be set before the ADC will start taking measurements. To enable the ADC, set the ADEN bit in ADCSRA:

```
ADCSRA |= (1 << ADEN);
```

To start the ADC measurements, the ADSC bit in ADCSRA needs to be set:

```
ADCSRA |= (1 << ADSC);
```

At this point, the ADC would begin continuously sampling the voltage presented on ADC0. The code to this point would look like this:

```
#include

int main (void)
{
   DDRE |= (1 << 2); // Set LED1 as output
   DDRG |= (1 << 0); // Set LED2 as output

   ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // Set

   ADMUX |= (1 << REFS0); // Set ADC reference to AVCC
   ADMUX |= (1 << ADLAR); // Left adjust ADC result to allow eas

   // No MUX values needed to be changed to use ADC0

   ADCSRA |= (1 << ADFR);  // Set ADC to Free-Running Mode

   ADCSRA |= (1 << ADEN);  // Enable ADC
   ADCSRA |= (1 << ADSC);  // Start A2D Conversions
   for(;;)  // Loop Forever
       {
               // TODO: Test ADC Value and set LEDs
       }
}
```

The only thing left to do is test the ADC value and set the LEDs to display a high / low indication. Since the ADC reading in ADCH has a maximum value of 255, a test value of 128 was chosen to determine whether the voltage was high or low. A simple IF/ELSE statement in the FOR loop will allow us to turn the correct LED on:

```
if(ADCH < 128)
               {
                       PORTE |= (1 << 2); // Turn on LED1
                       PORTG &= ~(1 << 0); // Turn off LED2
               }

               else
               {
                       PORTE &= ~(1 << 2); // Turn off LED1
                       PORTG |= (1 << 0); // Turn on LED2
               }
```

Again, if the notation used above is unclear, the “Programming 101” tutorial at AVRFreaks forum gives a great explanation.

Here's the finished program with comments. When compiled and downloaded to an ATMega128, LED1 will be lit for roughly half the rotation of the potentiometer, indicating a “low” voltage reading. Near the halfway point of the potentiometer's rotation, LED1 will go out and LED2 will light. Indicating a “high” voltage reading. By changing the tests in the FOR loop, one could get different voltage indications with two or more LEDs.

```
#include

int main (void)
{
   DDRE |= (1 << 2); // Set LED1 as output
   DDRG |= (1 << 0); // Set LED2 as output

   ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // Set

   ADMUX |= (1 << REFS0); // Set ADC reference to AVCC
   ADMUX |= (1 << ADLAR); // Left adjust ADC result to allow eas

   // No MUX values needed to be changed to use ADC0

   ADCSRA |= (1 << ADFR);  // Set ADC to Free-Running Mode
   ADCSRA |= (1 << ADEN);  // Enable ADC
```

```
        ADCSRA |= (1 << ADSC);  // Start A2D Conversions

        for(;;)  // Loop Forever
        {
                    if(ADCH < 128)
                    {
                            PORTE |= (1 << 2); // Turn on LED1
                            PORTG &= ~(1 << 0); // Turn off LED2
                    }

                    else
                    {
                            PORTE &= ~(1 << 2); // Turn off LED1
                            PORTG |= (1 << 0); // Turn on LED2
                    }

             }

    }
```

Part 2 – An Interrupt-Driven Example

Let's improve the first example so that we can run the LED IF loop “in the background”. It takes 13 cycles of the ADC clock to perform one A2D conversion in free-running mode according to the datasheet. With a prescaler of 128 as in the previous example, there are 13x128, or 1664, system clock cycles between each A2D conversions. If our short IF loop can be run only after an A2D conversion, this allows considerable processing time to be dedicated to other tasks.

Microcontrollers allow this kind of program execution using something called interrupts. Certain pieces of hardware within the microcontroller can signal that a certain task has been completed. Normal program execution can be “interrupted” when one of these signals (the interrupt) is asserted. Depending on the interrupt signal, different user-defined programs, called interrupt service routines or ISRs, can be run. After the ISR completes, normal program execution resumes.

The AVR ADC has an interrupt associated with it that is asserted when an A2D conversion completes. There are several changes that need to make to the first example to utilize this interrupt. First, let's write the ISR that will be run when the interrupt is asserted.

The first step in using interrupts in our application is to add the standard library header avr/interrupt.h. This file defines functions and macros needed to utilize interrupts on the AVR. The following line should be added below the io.h define in our original program.

```
    #include
```

Next, we'll define the ISR itself. To do this, we need the name of the interrupt we are connecting to the ISR. Referring to the datasheet, we find the name of the interrupt we want to use – ADC, which is asserted when an

A2C conversion completes. Here is the proper format for an ISR using the ADC interrupt:

```
ISR(ADC_vect)
{
        // Code to be executed when ISR fires
}
```

Now, we place the IF statement originally in the infinite loop inside the ISR, so it will only be run when the ADC interrupt indicates a conversion has been completed:

```
ISR(ADC_vect)
{
        if(ADCH < 128)
                {
                        PORTE |= (1 << 2); // Turn on LED1
                        PORTG &= ~(1 << 0); // Turn off LED2
                }

                else
                {
                        PORTE &= ~(1 << 2); // Turn off LED1
                        PORTG |= (1 << 0); // Turn on LED2
                }
}
```

At this point, the program has an ISR defined. However, the ISR will never execute. Interrupts on the AVR need to be enabled before they will run. This is done in two steps. First, the interrupt capability of the microprocessor needs to be enabled. This is done with the sei() function call, defined in interrupt.h to simplify this process. Next, the ADC interrupt needs to be enabled. This is done by setting the ADIE bit in the ADCSRA register. The following two lines enable the ADC interrupt:

```
ADCSRA |= (1 << ADIE);
sei();
```

We can now combine the new interrupt code with our first example. We will insert the ISR after the main loop. The interrupt enable lines will be inserted before we start the A2D conversions. The FOR loop is now empty, as the code we had there originally has been moved to the ISR. Other code could be inserted here to run between ISR calls. The full code is shown below.

```
#include
#include
```

```
int main (void)
{
        DDRE |= (1 << 2); // Set LED1 as output
        DDRG |= (1 << 0); // Set LED2 as output

        ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); //

        ADMUX |= (1 << REFS0); // Set ADC reference to AVCC
        ADMUX |= (1 << ADLAR); // Left adjust ADC result to allow

        // No MUX values needed to be changed to use ADC0

        ADCSRA |= (1 << ADFR);  // Set ADC to Free-Running Mode
        ADCSRA |= (1 << ADEN);  // Enable ADC

        ADCSRA |= (1 << ADIE);  // Enable ADC Interrupt
        sei();  // Enable Global Interrupts

        ADCSRA |= (1 << ADSC);  // Start A2D Conversions

        for(;;)  // Loop Forever
        {
        }
}

ISR(ADC_vect)
{
        if(ADCH < 128)
        {
                PORTE |= (1 << 2); // Turn on LED1
                PORTG &= ~(1 << 0); // Turn off LED2
        }

        else
        {
                PORTE &= ~(1 << 2); // Turn off LED1
                PORTG |= (1 << 0); // Turn on LED2
        }
}
```

Tags:
Learning and Information, **Tutorials**

Last Edited: Tue. Jun 23, 2015 - 12:22 PM

Top

ttownfire

Posted by **ttownfire**: Fri. Nov 16, 2007 - 09:53 PM

#2

Level: Rookie
Joined: Wed. Oct 3, 2007
Posts: 37 **View posts**
Location: Tucson, AZ

Total votes: 0

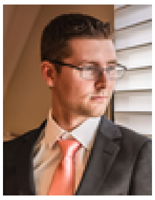
Awesome! Mind if I add this to the starter guide?

Michael

Dragon Slayer... no not that one...**This one! (http://www.tucsonfirefighters.com/images/home_1_04.png)**

Top

abcminiuser



Level: Raving Lunatic
Joined: Fri. Jan 23, 2004
Posts: 9855 **View posts**
Location: Melbourne, Australia

Posted by **abcminiuser**: Fri. Nov 16, 2007 - 10:00 PM **#3**

Total votes: 0

Very well done - I do like your writing and example style. It reminds me heavily of my own tutorials, which many people like.

Looking forward to more tutorials from you in the future!

- Dean :twisted:

Make Atmel Studio better with my **free extensions**. Open source and feedback welcome!

Top

penquissciguy

Level: New Member
Joined: Sat. Sep 9, 2006
Posts: 7 **View posts**
Location: Maine

Posted by **penquissciguy**: Fri. Nov 16, 2007 - 11:01 PM **#4**

Total votes: 0

ttownfire wrote:

> *Awesome! Mind if I add this to the starter guide?*

Please do. Thanks!

abcminiuser wrote:

> *Very well done - I do like your writing and example style. It reminds me heavily of my own tutorials, which many people like.*

:D

I intentionally copied the structure of your tutorials. I really like how yours flow from one topic to the next. Having a common tutorial structure might also help some know where to find certain information in a new tutorial and make code examples easier to follow.

abcminiuser wrote:

> *Looking forward to more tutorials from you in the future!*

Thanks, I appreciate that. I'm hoping to expand this one in the near future.

Ken

Top

penquissciguy

Level: New Member
Joined: Sat. Sep 9, 2006
Posts: 7 **View posts**
Location: Maine

Posted by **penquissciguy**: Sat. Nov 17, 2007 - 05:56 AM **#5**

Total votes: 0

I've added a simple interrupt-driven example. Comments welcome!

Ken

Top

starwarts

Level: New Member
Joined: Fri. Nov 23, 2007
Posts: 1 **View posts**

Posted by **starwarts**: Tue. Nov 27, 2007 - 09:27 AM **#6**

Total votes: 0

I have only starting dabbling with AVR coding.. this tutorial was very useful. Thanks.

Top

kioanakos

Posted by **kioanakos**: Tue. Nov 27, 2007 - 12:51 PM **#7**

Total votes: 0

Level: New Member
Joined: Fri. Jan 28, 2005
Posts: 1 **View posts**

Thanks for your tutorial!

I can't see your images...
Maybe the image hosting service you used is down.

Top

penquissciguy

Level: New Member
Joined: Sat. Sep 9, 2006
Posts: 7 **View posts**
Location: Maine

Posted by **penquissciguy**: Tue. Nov 27, 2007 - 07:05 PM **#8**

Total votes: 0

kioanakos wrote:

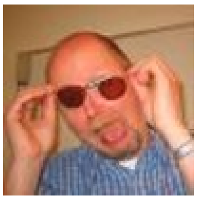
> *Thanks for your tutorial!*
>
> *I can't see your images...*
> *Maybe the image hosting service you used is down.*

The image is hosted on my personal webspace, so I think it should be OK. I reloaded the page on my machine and everything seems to be OK.

Ken

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Tue. Nov 27, 2007 - 07:27 PM **#9**

Total votes: 0

No dice here. The progress bar in the browser takes a while, and then it fails to load the images, shown by with a small frame with a red X in IE, a dimmed image icon in Firefox. Trying the direct url to one of the images (**http://www.intergate.com/~scienc... (http://www.intergate.com/~scienceguy/avr/varvolts.png)**) results in a non-responsive server error.

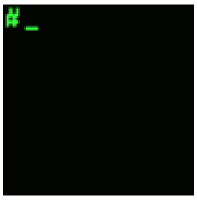
As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent concepts. Use the wrong words, communicate the wrong concept." [J Morin] "Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

clawson



Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**
Location: (using avr-gcc in) Finchingfield, Essex, England

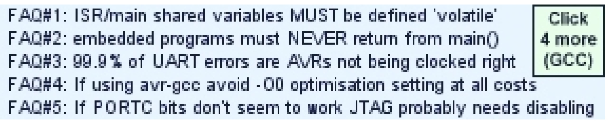
Posted by **clawson**: Tue. Nov 27, 2007 - 09:10 PM **#10**

Total votes: 0

The only image above is this isn't it?

Attachment(s):

varvolts.png



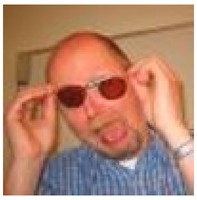
FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Tue. Nov 27, 2007 - 09:29 PM **#11**

Total votes: 0

Well Cliff, either the problem is not global or you have just proved that your cache is working. Still can't see it in the OP.

As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent concepts. Use the wrong words, communicate the wrong concept." [J Morin] "Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

bloody-orc

Level: Posting Freak
Joined: Sat. Dec 17, 2005
Posts: 1663 **View posts**
Location: Europe- Estonia- Tallinn

Posted by **bloody-orc**: Tue. Nov 27, 2007 - 09:56 PM **#12**

Total votes: 0

I can see it in OP post with FF and XP. No problems (And it's my first time here so no cache).

And nice tutorial indeed... At first I though it was Dean, who wrote it, but then I saw his name on the first comment and had to look again and surprise- surprise, it was penquissciguy. hehe ;)

Top

cgilson33

Level: New Member
Joined: Mon. Oct 8, 2007
Posts: 19 **View posts**
Location: Omaha, ne, usa

Posted by **cgilson33**: Fri. Dec 7, 2007 - 01:42 AM **#13**

Total votes: 0

great tutorial very helpful...

i'm actually using a similar functionality in a project. should i turn the ADC off somehow when i'm not using it? if so whats the best way...

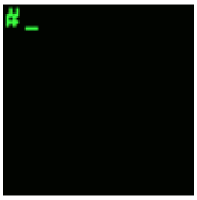
i am curious because i am trying to save power. is this a problem?

basic idea of project: user control =
1) remotely turn leds on or off
2) use the adc (via rssi) to control the leds

if the user uses the adc then goes back to manual mode the adc will still be running correct? what about interrupting? is it constantly interrupting if it's turned on (the ADC that is)?

Top

clawson



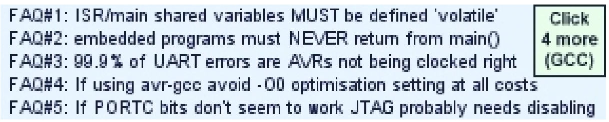
Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**
Location: (using avr-gcc in) Finchingfield, Essex, England

Posted by **clawson**: Fri. Dec 7, 2007 - 10:42 AM **#14**

Total votes: 0

Well it's the ADEN bit that turns it on, so clearing that will turn it off.

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)


FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

Top

albertob3

Level: New Member
Joined: Wed. Jan 31, 2007
Posts: 18 **View posts**

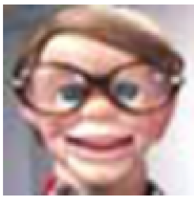
Posted by **albertob3**: Mon. Dec 17, 2007 - 11:54 PM **#15**

Total votes: 0

Thanks, this tutorial is very usefull.

Top

MR BLONDE



Level: Hangaround
Joined: Thu. May 13, 2004
Posts: 377 **View posts**
Location: north west uk

Posted by **MR BLONDE**: Wed. Dec 19, 2007 - 06:20 PM **#16**

Total votes: 0

Hello

Easy to follow very instructive with well placed useable material.

Thank you
Stephen :wink:

Codevisionavr & Avrstudio 4.18
Easyavr5A-Jtagicemk1

Call me Pedantic, But not after 9.

if Milk_Brilliant
else Codevision_Avrs==Better

Top

vicky3413

Posted by **vicky3413**: Mon. Jan 14, 2008 - 07:11 AM **#17**

Total votes: 0

mega 16 adc chanels are multiplexed.it means at a given time there is only one conversion.

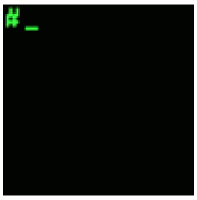
Level: Wannabe
Joined: Mon. Mar 12, 2007
Posts: 54 **View posts**

what in case we are using several sensors to guide a robot motion?

Top

---

clawson

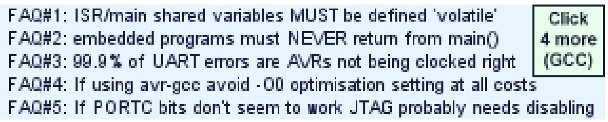


Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**
Location: (using avr-gcc in) Finchingfield, Essex, England

Posted by **clawson**: Mon. Jan 14, 2008 - 11:48 AM **#18**

Total votes: 0

Does the rate really need to be faster than the "round robin" rate you can achieve using the multiplexer? If so then use a number of Tiny's/small Mega's in parallel and "network" them using I2C or SPI or similar.



FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)

Top

MakeeK

Level: Rookie
Joined: Sat. Dec 1, 2007
Posts: 45 **View posts**

Posted by **MakeeK**: Tue. Jan 22, 2008 - 09:18 PM **#19**

Total votes: 0

Thanks for the tutorial. It only destroyed my atmega8 :)

I'm a Newbie! ... and it's not my fault!!

Top

TeonHarasymiv

Level: New Member
Joined: Tue. Jan 1, 2008
Posts: 2 **View posts**

Posted by **TeonHarasymiv**: Fri. Jan 25, 2008 - 06:24 AM **#20**

Total votes: 0

Thanks for the tutorial! Much appreciated!

Top

ALIBE

Level: Wannabe
Joined: Sat. Sep 1, 2007
Posts: 53 **View posts**
Location: Redmond, WA, USA

Posted by **ALIBE**: Tue. Jan 29, 2008 - 02:45 PM **#21**

Total votes: 0

very well written tutorial. Simple and Descriptive.

Thank You

Nagi

Top

magnusrt

Level: Hangaround
Joined: Sun. Mar 4, 2007
Posts: 120 **View posts**

Posted by **magnusrt**: Mon. Feb 4, 2008 - 01:14 AM **#22**

Total votes: 0

Hi
WhenI try to compile The last example in this tutorial (with AVR Studio) I get the message

Quote:

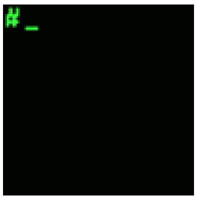
> *error; 'ADFR' undeclared (first use in this function)*

How can i correct this..

Magnus

Top

clawson



Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**

Posted by **clawson**: Mon. Feb 4, 2008 - 10:28 AM **#23**

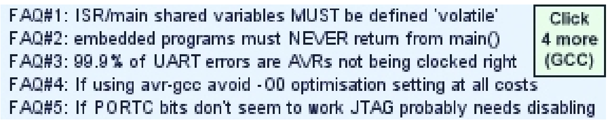
Total votes: 0

magnusrt wrote:

> *How can i correct this..*

Location: (using avr-gcc in) Finchingfield, Essex, England

Maybe start by identifying which AVR you are using?



FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)

Top

magnusrt

Level: Hangaround
Joined: Sun. Mar 4, 2007
Posts: 120 **View posts**

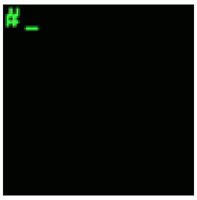
Posted by **magnusrt**: Mon. Feb 4, 2008 - 10:34 AM **#24**

Total votes: 0

Sorry!
Mega88

Top

clawson



Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**
Location: (using avr-gcc in) Finchingfield, Essex, England

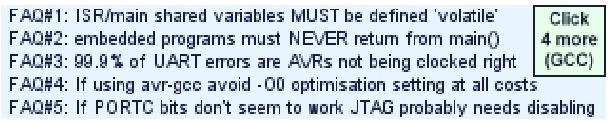
Posted by **clawson**: Mon. Feb 4, 2008 - 10:41 AM **#25**

Total votes: 0

Well if you go to the ADC chapter of the 48/88/168 datasheet and search for "free" you get SOME clues - though it may not be entirely obvious that they now also refer to it as "auto triggering" as well and the control bit for this is ADATE - bit 5 in ADCSRA

See the second paragraph in section 22.3 - Starting a Conversion

Also read about ADTS2/1/0 and note what 000 selects.



FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)

Top

bootstrap

Level: Hangaround
Joined: Mon. Sep 24, 2007
Posts: 181 **View posts**
Location: india

Posted by **bootstrap**: Thu. Feb 14, 2008 - 05:36 PM **#26**

Total votes: 0

hi

Last Edited: Sun. Mar 2, 2008 - 03:27 PM

Top

vizard356

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Posted by **vizard356**: Thu. Feb 21, 2008 - 02:19 PM **#27**

Total votes: 0

gr8 tutorial

vicky3413 wrote:

> *mega 16 adc chanels are multiplexed.it means at a given time there is only one conversion.*
> *what in case we are using several sensors to guide a robot motion?*

Whichever sensor u want to read u have to select that channel through the ADMUX register.

Top

vizard356

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Posted by **vizard356**: Thu. Feb 28, 2008 - 08:50 AM **#28**

Total votes: 0

I am little bit confuse :? about Interrupt Driven Example.

In mega32 data sheet it says
"Switching to Free Running
mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC
Interrupt Flag is set(Page 218)"

So is it necessary to use other trigger source if we are using interrupt?

I mean logically in this mode it will be continuously converting Analog to Digital so it will continuously call the interrupt routing on the completion.
Still i m not sure.:(
Please help me here.

Thank You

Last Edited: Thu. Feb 28, 2008 - 03:24 PM

Top

mikaelo

Level: Rookie
Joined: Thu. Oct 11, 2007
Posts: 40 **View posts**
Location: Atlanta, GA

Posted by **mikaelo**: Thu. Feb 28, 2008 - 02:53 PM

#29

Total votes: 0

vizard356 wrote:

> *gr8 tutorial*
>
> *Whichever sensor u want to read u have to select that channel through the ADMUX register.*

So, if you need say 3 ADC's and want to use interupt driven code, can you change the ADMUX inside the ISR and the next trigger will be the next channel, or will the next sampling already have started even though you haven't read the value yet?

Top

vizard356

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Posted by **vizard356**: Thu. Feb 28, 2008 - 03:21 PM

#30

Total votes: 0

mikaelo wrote:

> *So, if you need say 3 ADC's and want to use interupt driven code, can you change the ADMUX inside the ISR*

Yes, u can change ADMUX inside the ISR.

Quote:

> *and the next trigger will be the next channel, or will the next sampling already have started even though you haven't read the value yet?*

you have to careful because after changing ADMUX next conversation might be based of the previous value of ADMUX.
So you should put small delay after changing ADMUX and before reading the value.

Top

bootstrap

Level: Hangaround
Joined: Mon. Sep 24, 2007
Posts: 181 **View posts**
Location: india

Posted by **bootstrap**: Fri. Mar 7, 2008 - 02:24 PM **#31**

Total votes: 0

Is it necessary that we use a 10 k potentiometer?What is the potentiometer with the smallest possible value that one can use for the ADC?

Top

vizard356

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Posted by **vizard356**: Fri. Mar 7, 2008 - 07:46 PM **#32**

Total votes: 0

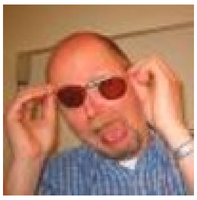
bootstrap wrote:

> *Is it necessary that we use a 10 k potentiometer?What is the potentiometer with the smallest possible value that one can use for the ADC?*

Well, potentiometer is there because you can change voltage by changing resistance so that you can test sample program.
Actually there is no specific value of potentiometer should be used.
But 10K works great. At 0 ohm u can get min voltage and at 10k u can get max voltage.

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Fri. Mar 7, 2008 - 08:54 PM **#33**

Total votes: 0

Quote:

> *Actually there is no specific value of potentiometer should be used.*

Maybe not, but a too low value on the pot will work bad. Heres why:

A potentiometer is essentially a resistor with the two ends connected to the supply voltage and ground respectively, and with a tap that you can move from one end of it to the other. Thus it forms a voltage divider and with the tap at one end you the voltage at it will be the supply voltage and at the other end it will be zero volts. Thats all fine, and a broad range of pot values will make this scheme work. But (and thats a BIG BUT) the pot is also a resistor between the supply rail and ground. So now, if you use say a 47 Ohm pot you will more or less short the supply rail to ground. Ohms law will tell you how much current will flow through the pot: V = R * I, and moved around a bit you have I = V / R, and assuming a 5 Volt supply you get 5 / 47 amps which is approximately 100 milliamps. Compare that to using a 10KOhm pot through which will flow only 5/10000 = 0.5 milliamps.

Quote:

> *What is the potentiometer with the smallest possible value that one can use for the ADC?*

What is the maximum current that the pot can take? (look it up in the specs)
How much current can your power supply generate?
How much heat do you need? :wink:

As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent concepts. Use the wrong words, communicate the wrong concept." [J Morin] "Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

bootstrap

Level: Hangaround
Joined: Mon. Sep 24, 2007
Posts: 181 **View posts**
Location: india

Posted by **bootstrap**: Sat. Mar 8, 2008 - 03:33 AM **#34**

Total votes: 0

I have a 1k pot.The I/O pin of the atmega 8 can take in 40 milliamp.Power supply can give a max of 500 ma.And the current through the pot should be 5/1000 = 5 mA.This 5mA doesn't seem to damage anything

Top

vizard356

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Posted by **vizard356**: Sat. Mar 8, 2008 - 08:45 AM **#35**

Total votes: 0

bootstrap wrote:

> *I have a 1k pot.The I/O pin of the atmega 8 can take in 40 milliamp.Power supply can give a max of 500 ma.And the current through the pot should be 5/1000 = 5 mA.This 5mA doesn't seem to damage anything*

5mA doesn't do any damage.But it is current when resistance across pot is 1k, what if you turn the pot and make resistance 0 ohm. Its short circuit. So you have to be careful and connect some fix resistance as shown in sample figure.

Top

bootstrap

Level: Hangaround
Joined: Mon. Sep 24, 2007
Posts: 181 **View posts**

Posted by **bootstrap**: Sat. Mar 8, 2008 - 09:42 AM **#36**

Total votes: 0

vizard356 wrote:

> *5mA doesn't do any damage.But it is current when resistance*

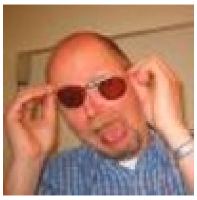
Location: india

> *across pot is 1k, what if you turn the pot and make resistance 0 ohm. Its short circuit.*
> *So you have to be careful and connect some fix resistance as shown in sample figure.*

So if independent of the value of the pot if I have the 330 ohm resistors connected to the Vcc and Gnd legs of the pot the micro wont be damaged? So if the we have the 330 ohm does it matter what value of POT we use (as in the sample figure?)

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Sat. Mar 8, 2008 - 04:07 PM **#37**

Total votes: 0

Quote:

> *Its short circuit.*

Only if the impedance of the AVR input pin is 0 Ohms. I took a quick look in th ATmega88 data sheet to find the impedance of a pin in input mode, but couldnt find it. The only thing I found was this(p 299):

Input Leakage Current I/O Pin: 1 microampere (for pin both high and low, @ Vcc 5.5 volts). If I interpret this correctly there is no problem with a short through the AVR. (It actually suggests an input impedance of 5.5 MOhms. Take this with a grain of salt though, maybe I'm misreading the sheet.)

Anyhow, the problem with using a pot with a too low value is the shorting from supply rail to ground through the pot.

Now I'm curious: What made you pose the question, bootstrap? Why the need to use a low ohm pot? (Weekend and no supply of 10K resistors, maybe?)

As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent

concepts. Use the wrong words, communicate the wrong concept." [J Morin]
"Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

microcarl

Level: Raving Lunatic
Joined: Sun. May 30, 2004
Posts: 8363 **View posts**
Location: Cincinnati, Ohio

Posted by **microcarl**: Sat. Mar 8, 2008 - 05:03 PM **#38**

Total votes: 0

JohanEkdahl wrote:

> *Quote:*
>
> > *Anyhow, the problem with using a pot with a too low value is the shorting from supply rail to ground through the pot.*
>
> *If the OP is concerned about input impedance of the ADC channel, a buffer amp will provide high input, and quite low output impedance.*

```
VCC
---
 |
 |
 |
 /
 \  10K Ohm            |\
 /                      |  \
 \<-----------------|+  \
 /                      |   \
 \            LF355   |    +-------+-------> ADC Input
 /                      |   /         |
 |             --------|- /        |
 |             |        | /         |
 |             |        |/          |
---           |                     |
GND           |                     |
               +--------------------+
```

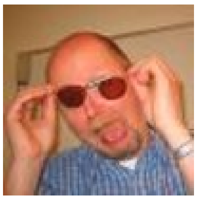
I have used the above circuit in several designs. But there, the usage was for isolation and not necessarily impedance control.

But a 10K Ohm pot should not be of any concern to the ADC. I've used 10K, ten turn wire-wound pots for years with no ill effects.

You can avoid reality, for a while. But you can't avoid the consequences of reality! - C.W. Livingston

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Sun. Mar 9, 2008 - 09:24 AM **#39**

Total votes: 0

Carl, the eye-opener! My brain must have gone bzzzzztt! yesterday.

Of course we are discussing the ADC input, and on page 306 of the ATmega88 data sheet it is in black and white:

Analog input resistance: Typically 100 MOhm.

Quote:

> *But a 10K Ohm pot should not be of any concern to the ADC.*

No, but OP is asking "how low can we go".

Again, as I see it this has nothing to do with the value of the pot (as *any* pot will be able to be set to act as a 0 Ohm resistor between the supply rail and the ADC pin).

As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent concepts. Use the wrong words, communicate the wrong concept." [J Morin] "Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

vizard356

Posted by **vizard356**: Sun. Mar 9, 2008 - 06:32 PM **#40**

Level: New Member
Joined: Fri. Oct 5, 2007
Posts: 10 **View posts**

Total votes: 0

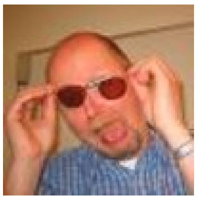
Quote:

> *Again, as I see it this has nothing to do with the value of the pot (as any pot will be able to be set to act as a 0 Ohm resistor between the supply rail and the ADC pin).*

Yep, exactly. That's why some fix resistance **must** be used. Otherwise power supply might get damage if kept for long time :(due to heavy current (if fuse is not used).

Top

JohanEkdahl



Level: 10k+ Postman
Joined: Wed. Mar 27, 2002
Posts: 26988 **View posts**
Location: Lund, Sweden

Posted by **JohanEkdahl**: Sun. Mar 9, 2008 - 08:53 PM **#41**

Total votes: 0

Quote:

> *That's why some fix resistance must be used.*

And if I read the data sheet correctly it is - internal in the AVR. And the value of it is 1 MOhm (nominally)! So you do not need to supply any fixed resistance external to the AVR.

As of January 15, 2018, **Site fix-up** work has begun! Now **do your part** and **report** any bugs or deficiencies **here**.

No guarantees, but if we don't report problems they won't get much of a chance to be fixed! Details/discussions at link given just above.

"Some questions have no answers."[C Baird] "There comes a point where the spoon-feeding has to stop and the independent thinking has to start." [C Lawson] "There are always ways to disagree, without being disagreeable."[E Weddington] "Words represent concepts. Use the wrong words, communicate the wrong concept." [J Morin] "Persistence only goes so far if you set yourself up for failure." [Kartman]

Top

JohnPsmith

Level: Rookie
Joined: Thu. Nov 29, 2007
Posts: 29 **View posts**

Posted by **JohnPsmith**: Mon. Mar 10, 2008 - 05:12 PM **#42**

Total votes: 0

As a newbie (we often struggle with nuances), I feel compelled to point out that while the NON-interrupt example works as it should, if one is taking baby steps at getting the ADC's to work, it should be observed that the first conversion will *not* be completed when

```
if(ADCH < 128)
```

is executed. Thus, if one simply captures ADCH at that point (as a baby stepping newbie might),say, with

```
Temp = ADCH;
```

it will not valid. To do so, the ADCSR bit, ADSC, must be polled for a transition to 0 to indicate that the conversion is complete.

John

Top

Ravender

Level: New Member
Joined: Sun. Mar 2, 2008
Posts: 4 **View posts**

Posted by **Ravender**: Sat. Mar 15, 2008 - 01:30 PM **#43**

Total votes: 0

I have a problem. The tutorial was great so I decide to make a thermometer with LM35 on attiny26.

When I measure voltage on PA0, there is 0,27V (so is't ok - 27 celsius). BUT!!! when I set for egzample PB4 (ADC7) (PORTB=(1<<4)), my voltmeter shows 0,67V !!! I thought, that if PA0 is an input, so i can make everything with other pins, but it seems I'm wrong! What is going on?

```
DDRB = 0xff; // portb as output

// ADC setup
    setb(ADCSR,ADPS0); setb(ADCSR,ADPS1);    // prescaler 8 for 1M
    clrb(ADMUX, REFS0); clrb(ADMUX, REFS1); // voltage reference -
    setb(ADCSR, ADFR); // free running mode
```

```
        setb(ADCSR, ADEN); // adc enable
        setb(ADCSR, ADSC); // start conversion
        //setb(ADCSR, ADIE); // interrupt for ADC on

        sei();


        while(1)
        {
            setb(PORTB, 4); // <--- problem!!!
        }
```

Top

●

davef

Level: Resident
Joined: Sat. Sep 3, 2005
Posts: 867 **View posts**
Location: Christchurch, NZ

Posted by **davef**: Sun. Mar 16, 2008 - 08:46 AM **#44**

Total votes: 0

Are you enabling the pull-up on this port, when you don't want to?

Top

●

gicutabacaru

Level: New Member
Joined: Sun. Mar 19, 2006
Posts: 13 **View posts**
Location: ROU

Posted by **gicutabacaru**: Thu. Mar 20, 2008 - 06:49 PM **#45**

Total votes: 0

How do you select the ADC channel? in the atmega8535 datasheet they write that this is done by setting some bits in ADMUX, but i can't figure out which. Thanks

Top

●

SteveN

Posted by **SteveN**: Thu. Mar 20, 2008 - 08:02 PM **#46**

Total votes: 0

Level: Raving Lunatic
Joined: Wed. Nov 14, 2001
Posts: 3635 **View posts**
Location: Charlottesville, VA USA

I can certainly see how that would be difficult to find (seriously, I am not being a smart ass :-)). The Acrobat "Bookmark" takes you to page 213 in the datasheet for the "Changing Channel or Reference Selection" section. You have to skip all the way to page 220 to see Table 85. All is revealed there :-).

Top

gicutabacaru

Level: New Member
Joined: Sun. Mar 19, 2006
Posts: 13 **View posts**
Location: ROU

Posted by **gicutabacaru**: Thu. Mar 20, 2008 - 08:21 PM **#47**

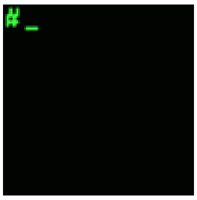
Total votes: 0

So, for changing chanels i need to add some instructions like these:
ADMUX |= (1 << REFS0);
ADMUX |= (1 <<ADLAR)|(0<<MUX4)|(0<<MUX3)|(0<<MUX2)|(0<<MUX1)|(0<<MUX0)

?

Top

clawson



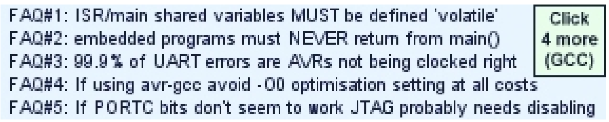
Level: Moderator
Joined: Mon. Jul 18, 2005
Posts: 92966 **View posts**
Location: (using avr-gcc in) Finchingfield, Essex, England

Posted by **clawson**: Thu. Mar 20, 2008 - 11:05 PM **#48**

Total votes: 0

For changing to channel n use:

```
ADMUX &= 0xF8; // clear bottom 3 bits
ADMUX |= n; // then set bottom 3 bits to channel n
```



FAQ#1: ISR/main shared variables MUST be defined 'volatile'
FAQ#2: embedded programs must NEVER return from main()
FAQ#3: 99.9% of UART errors are AVRs not being clocked right
FAQ#4: If using avr-gcc avoid -O0 optimisation setting at all costs
FAQ#5: If PORTC bits don't seem to work JTAG probably needs disabling

Click
4 more
(GCC)

(http://www.nongnu.org/avr-libc/user-manual/FAQ.html)

Top

lmc222

Level: Rookie
Joined: Sun. Sep 23, 2007
Posts: 22 **View posts**
Location: Brisbane, Australia

Posted by **lmc222**: Fri. Mar 21, 2008 - 02:02 AM **#49**

Total votes: 0

@penquissciguy
Fantastic TuT loved the style and pacing, easy to follow even for a newb like me. SOOO much easier to understand when there is a working example to guide you and only one variable is changed at a time with each progression. Once again thanks for your effort!!! :mrgreen: (wish there was an emoticon with a hat tip because that would be more appropriate).

Lachlan

What we need to learn,
we learn by doing.

Top

penquissciguy

Level: New Member
Joined: Sat. Sep 9, 2006
Posts: 7 **View posts**
Location: Maine

Posted by **penquissciguy**: Sat. Mar 22, 2008 - 04:39 PM **#50**

Total votes: 0

lmc222 wrote:

> *@penquissciguy
> Fantastic TuT loved the style and pacing, easy to follow even for a newb like me. SOOO much easier to understand when there is a working example to guide you and only one variable is changed at a time with each progression. Once again thanks for your effort!!! :mrgreen: (wish there was an emoticon with a hat tip because that would be more appropriate).*

Thanks! I appreciate the compliment. I basically copied abcminiuser's pattern for a tutorial. I like the whole step-by-step flow myself.

Ken

Last Edited: Tue. Mar 25, 2008 - 08:47 PM

Top

Posted by **pepper_bg**: Mon. Mar 24, 2008 - 06:39 AM **#51** pepper_bg

Level: New Member
Joined: Mon. Mar 24, 2008
Posts: 9 **View posts**

Total votes: 0

Hi,

Excellent tutorial, thanks for posting it!

I am trying to run the same examples with a attiny45. It works but not quite. Looked at the data sheet and converted the registers:

```
#include
#include

int main(void) {
    // Set Port B pins for 3 and 4 as outputs
    DDRB = ( 1 << DDB3 ) | ( 1 << DDB4 );

    //set the reference voltage for the ATTINY45 ADC to be VCC
    ADMUX &= ~((1 << REFS0) | (1 << REFS1));

    //set pin #1 as ADC0
    ADMUX &= ~( ( 1 << MUX0 ) | ( 1 << MUX1 ) | ( 1 << MUX2 )

    //left allign the adc value
    ADMUX |= (1 << ADLAR);

    //set the division factor to 128 (see the datasheet)
    ADCSRA |= (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0);

    //auto triger enable
    ADCSRA |= (1 << ADATE);

    //enable the ADC
    ADCSRA |= (1 << ADEN);

    //start the adc measurments
    ADCSRA |= (1 << ADSC);

    for ( ; 1==1 ; ) {

        if(ADCH < 128) {
            PORTB = 8;
        } else {
            PORTB = 16;
        }

        //optional
        PORTB = 0;

    }

    return 1;
}
```

What this does is untill around 2.1V both LEDs are off, at 2.1V LED1 turns on, at around 2.5V LED1 is off and LED2 is on and stays this way to the end (4.82V).

What beats me is that with the if/else in the endless loop above both LEDs can be dark. My guess is that there is some sort of interrupt/triggering happening but I can't quite get it. Without the

```
    ADCSRA |= (1 << ADATE);
```

line it was behaving even more strangely (the behavior was dependent on the speed with which I move the potentiometer - if I move fast LED1 turns on, if I move slow LED2 turns on, both at around 2.5V).

Hope someone can explain what is happening and/or how to fix it.
Appreciate your help...

Top

Jump To

--Tutorials ▼

Topic locked

- **Privacy**
- **Contact**
- **Site Use Terms**