This document describes various factors to consider when using the ADC peripheral for the nRF51 series. The ADC is identical on all nRF51 series chip versions, which are:

- nRF51822 (all versions)
- nRF51422 (all versions)
- nRF51922 (all versions)

Introduction

This document describes the following:

- ADC saturation points
- Maximum input voltage
- How to sample from multiple input pins
- What to consider when using the ADC together with the softdevice
- Sampling frequency limitations
- Input impedance
- Connect voltage divider to lower voltage on ADC input pin
- Measure Lithium-Ion battery voltage with dedicated voltage divider

The ADC samples connected input with either 8-bit, 9-bit or 10-bit resolution for the voltage range it is configured for. The ADC should be configured in a way that is appropriate for the intended application. It is important to match the voltage range of the ADC with the connected input voltage range in order to obtain the best results.

ADC configuration

When measuring battery voltage, configure the INPSEL as VDD, either with 1/3 prescaling or 2/3 prescaling, depending on the maximum voltage of the connected battery. Also configure the REFSEL to VBG, which is 1.2V fixed voltage reference. For batteries that have voltage range exceeding the maximum 3.6V, e.g. Lithium-Ion batteries, see chapter ‘Measuring Lithium battery voltage with connected voltage divider’

To configure the ADC for an input pin, the first thing is to consider what voltage range is for the connected input. Configure the ADC to have its voltage range match the input voltage range in order to utilize the full resolution of the ADC (8-bit, 9-bit or 10-bit) and to avoid saturation. Also make sure the input pin voltage is never higher than VDD, i.e. the supply voltage for the chip. Also make sure that the input impedance seen by the ADC input pin is small (<1kohm) to ensure accurate reading.

ADC saturation points

Point of ADC saturation really depends on the configured ADC reference voltage and the chosen prescaling. If the 1.2V VBG internal reference voltage is used the ADC range will be 0-1.2V with a saturation of 1.2V. This means that your AIN signal with 1/1 prescaling should be in the range 0-1.2V in order to obtain proper conversion. Input above 1.2V will just be converted to the maximum ADC value. However if you use for example 1/3 prescaling for your AIN input the input is scaled down by 1/3. The effect is that your AIN

voltage range is 0-3.6V, because the 3.6V input voltage is scaled down to 3.6/3=1.2V.

Examples of saturation points for ADC AIN inputs are:

1.2VBG reference, AIN 1/1 prescaling: AIN 1.2V
1.2VBG reference, AIN 2/3 prescaling: AIN 1.8V
1.2VBG reference, AIN 1/3 prescaling: AIN 3.6V
1.0V AREF, AIN 1/1 prescaling: AIN 1.0V
1.0V AREF, AIN 2/3 prescaling: AIN 1.5V
1.0V AREF, AIN 1/3 prescaling: AIN 3.0V
VDD is 3.0V, VDD 1/2 reference, AIN 1/1 prescaling: AIN 1.5V

ADC maximum input voltage

There are basically two rules to follow when deciding max voltage on AIN pins:

1. The ADC should not be exposed to higher voltage than 2.4V on an input pin AIN after prescaling
2. GPIO pin should not be exposed to higher voltage than VDD+0.3V, see Aboslute maximum ratings in the nRF51822 PS

For example, when having 2/3 prescaling you can expose 2.4V/(2/3)=3.6V to an AIN pin. For not to violate rule 2, VDD should be 3.3V or higher in this case.

These are the maximum voltages that can be exposed to an ADC AIN pins, depending on supply voltage and your prescale settings:
VDD 3.6, prescaling 1/1: AIN max 2.4V (rule 1 limitation)
VDD 3.6, prescaling 2/3: AIN max 3.6V (rule 1 limitation)
VDD 3.6, prescaling 1/3: AIN max 3.9V (rule 2 limitation)
VDD 3.3, prescaling 1/1: AIN max 2.4V (rule 1 limitation)
VDD 3.3, prescaling 2/3: AIN max 3.6V (rule 1 and rule 2 limitation)
VDD 3.3, prescaling 1/3: AIN max 3.6V (rule 2 limitation)
VDD 1.8, prescaling 1/1: AIN max 2.1V (rule 2 limitation)
VDD 1.8, prescaling 2/3: AIN max 2.1V (rule 2 limitation)
VDD 1.8, prescaling 1/3: AIN max 2.1V (rule 2 limitation)

ADC input impedance limitations

The impedance of the input voltage source, i.e. the device that generates the voltage to be measured by the ADC input pin, must be low, preferably within 1 kohm. If the impedance of the source is below 1 kohm the ADC error specification in the Product specification (nRF51822 PS) is valid and choosing different prescale settings for the ADC input will have close to no effect on the accuracy of the ADC. On the other hand,

when applying voltage source to the ADC input pin with high impedance, additional gain and offset error is introduced which is different for different prescalers.

Figure 1 shows the nRF51 ADC input model when the ADC is sampling and the value of R_AIN for different prescale settings. The internal VBG reference voltage is 1.2V so the ADC internal voltage source is VBG/2=0.6V. When the ADC is not sampling the AIN input pin has very high impedance and can be regarded as open circuit. Table 1 shows the statistics for the internal impedance for different prescale settings. 99.7% of devices (+- 3 sigma) are expected to be within 6.3%, for example for 1/1 prescale => [121.5, 137.9]kOhm.

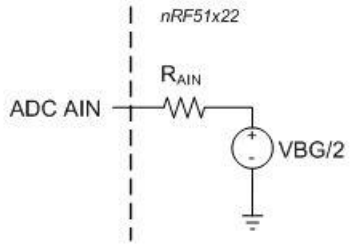

Figure 1. nRF51 ADC input model

Input prescaler	Mean resistance [kOhm]	Standard deviation [kOhm]
1/3	389.2	8.2
2/3	194.6	4.1
1/1	129.7	2.74

Table 1. Input impedance statistics for R_{AIN} on nRF51 ADC input

Selecting timers to trigger ADC sampling

The application timers use the RTC1 timer in the background. When using application timers, you need to share your timing with other periodic tasks that are also using application timers. Also, the application timers are setting software interrupts that may introduce latency. For real time sampling it would be best to use TIMER1 or TIMER2 directly with PPI channels to trigger the ADC sampling. You could also use RTC1 directly if you are not enabling the application timers. RTC0 and TIMER0 are blocked by the softdevice.

Selecting the right clock source for the ADC

It is chosen in the ble_app_hrs_with_adc_sample_ain example to use the external 16MHz crystal for the ADC sampling because that will guarantee that the accuracy of the ADC will be within specification in the nRF51x22 Product Specification. You can use the internal 16MHz RC clock but then the accuracy is no longer guaranteed. The accuracy of the internal RC is much lower than of the crystal. However, when using the external crystal, current consumption will be higher since it takes longer to start up the crystal than the RC, see tables 16 and 18 in the nRF51822 PS v1.3. In short, use the external crystal for maximum accuracy, use the internal RC for lowest current consumption.

Sampling on multiple input pins simultaneously

The ADC can actually only sample from one input pin at each time. To sample from multiple input pins is achieved with time division multiplexing by configuring the ADC to sample from one input pin, sample that pin, reconfigure the ADC to sample from a second input pin, and then sample from the second pin. The adc_example_from_AIN_8bit_10bit code example samples from two input pins. Explanation on how to operate the example is given at the top of the main file.

Lowering voltage on ADC input pin

If a sensor or battery has output voltage which is above the ADC voltage range it is necessary to divide that voltage before exposing it to an ADC input pin. This can be achieved with a voltage divider. An example of a voltage divider is shown in figure Figure 2 for lowering voltage from a Lithium-Ion battery. Because of internal impedance of the ADC, having a voltage divider with large resistor values will introduce error in ADC output. If the impedance of the voltage divider is less than 1kohm, the error is very small and can be neglected. However, as the impedance of the voltage divider is increased, more error will be present. It may on the other hand be desirable to have resistor value of the voltage divider high in order to limit current leak through the voltage divider.

A valid method for lowering voltage to an ADC input pin is to connect a voltage divider with relatively high resistance values and a capacitor connected between the ADC input pin (AIN) and ground. Connecting relatively large capacitor will decrease the ADC output error but will limit the sampling frequency.

To make the ADC output values with highest accuracy and with high sampling frequency for input voltages higher than the ADC voltage range, a voltage buffer is needed.

Another possible method is to connect a FET transistor between the power supply and the voltage divider which will open for current through the voltage divider momentarily before sampling. The voltage divider would then have low resistor values (<1kohm) and

no capacitor connected. The voltage divider would then consume high current when sampling but will not consume any current when not sampling.

When not sampling, the R_AIN (referring to Figure 2) value will have very high value and you can consider it to be an open circuit. The moment you are sampling, R_AIN is 120k-400k and therefore lowers the U_AIN voltage when a voltage divider is connected. On the other hand, when a capacitor is also connected between AIN and ground, it should keep the U_AIN voltage at previous level for an adequate time period while sampling, therefore minimizing the effect of high resistance value of R_2. The capacitor should be large enough to hold voltage up for the required time period, i.e. 20 us for 8 bit sampling or 68 us for 10 bit sampling. On the other hand, the capacitor should be small enough in order for it to fully charge before the next sample is taken. So when a capacitor is connected, it's size is a tradeoff between accuracy and sampling frequency.

What actually happens during the sampling period is that multiple samples are made and the ADC output value is the mean value from the sample pool. As noted in table 38 in the nRF51822 PS v1.2 the sample pool is created during 20 us period for 8 bit sampling, 36 us period for 9 bit sampling and 68 bit period for 10 bit sampling.

Measuring Lithium battery voltage with connected voltage divider

Attached is a hardware setup for the ADC which is a voltage divider with a connected capacitor. It is meant for measurement of Lithium battery with voltage range of 2.7-4.2V. For this purpose, we recommend to divide your input voltage with two resistors, R1 = 10Mohm and R2 = 2.2Mohm. Then you need to connect a 22nF capacitor (C2 in the attached schematic) from your ADC analog input pin (AIN) to ground. With this configuration you can sample with up to 20Hz for 8-bit sampling and up to 8Hz for 10-bit sampling without noticable ADC output error. You should use the internal VBG voltage as reference which is 1.2V fixed, and no input prescaling (1/1).

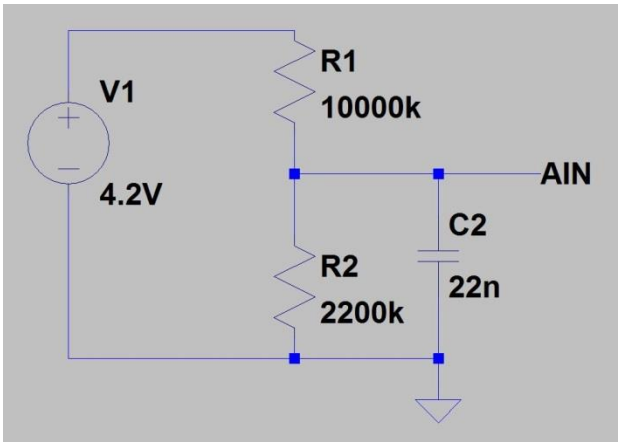

Figure 2. Voltage divider setup to measure Lithium battery

I have attached ADC example code that samples from analog intput pin 6 and is set up

for the hardware configuration mentioned above. The example outputs the ADC sampled result to port 1, i.e. pins 8-15. It is tested for nRF51822 SDK 4.4.2.

With the setup mentioned above the voltage divider consumes current of 4.2V/(12,2Mohm)=0.35 uA.

Using the ADC with the S110 softdevice

The ble_app_hrs_with_adc_sample_ain code example samples voltage on an input pin while the S110 softdevice is enabled. You should be aware of that you need to set the priority of the ADC interrupt explicitly to LOW. Also, PPI channels 8-15 are blocked by the softdevice, as stated in table 6 in the S110 Softdevice Specification v1.1, so using them for the application will cause conflicts with the softdevice. Use only PPI channels 0-7 for the application. You can configure your PPI channels directly before initializing the softdevice or use the sd_ppi_* wrapper functions if you choose to initialize the PPI channels after you initialize the softdevice. Documentation for the wrapper function can be found in the SoC library API in the nRF51 SDK documentation.

Sampling frequency limitations

You should also be aware of that during BLE transmission, the CPU will be blocked, therefore limiting the sampling frequency of the ADC. The ADC itself will be able to process up to 50k samples per second but the CPU will be blocked for ca 0.8 - 6.0 ms when the softdevice is enabled, depending on the amount of data you are sending over the BLE link.

Table 12 in the S110_Softdevice_Specification_v1.1 states that throughput is up to 120kpbs for sending data from a client. However, when you send 32-40 kbps you would have to select 7.5ms connection interval and send 2 packets per connection interval. The CPU will be blocked for the whole transmission time of the 2 packets and that will block the CPU for about 1.6 ms. Sampling with 1kHz will not work because the ADC only buffers one sample and overwrites it when sampling again. The CPU is simply not available to process the ADC data every 1 millisecond.

Table 10 in the S110_Softdevice_Specification v1.1 indicates how long the CPU is blocked in respect to what you are sending many packets in each connection interval. The thing is that the length of the blocking period also depends on several factors:

Packets sent per connection interval
Number of bytes in each packet
Master 32kHz clock accuracy
Slave 32kHz clock accuracy
Connection interval
Slave latency

The CPU blocking time in table 10 is therefore not very accurate because you really need

to know the value of the above parameters, so the table is more of an estimate. However, when you want to obtain the shortest CPU blocking period of the softdevice you should set connection interval to 7.5ms, slave latency to zero and transfer as few packets as possible in each connection interval.

When connection interval is 7.5ms the master and slave 32kHz clock accuracy has very little effect on the lenght of the CPU blocking period. However, it has great effect when connection interval is e.g. 4000 ms and/or when slave latency is high.

So, to be brief, if you send or receive only one packet per connection interval, you should be able to process ADC samples with 1kHz frequency, given that you choose connection interval of 7,5ms and no slave latency. However, if you send or receive 2 packets per connection interval, you will most likely need to decrease you sampling frequency to 500 Hz.