Analog to Digital Converters

ELECTGON
www.electgon.com
ma_ext@gmx.net

26.11.2017

Contents

Chapter 1

Theory Behind ADC

In RF systems an important block is needed to transfer received signal from the receiving stage to the processing stage, i.e. a gate that connects real world (which contains analog signal), with modern systems world (which contains digital signals) is needed. This gate is the ADC which is needed then for any application based on signal processing. Most ADC applications today can be classified into four broad market segments: Data acquisition – Precision industrial measurement – voice and audio applications – high speed (implying sampling rates greater than 5 MSPS).

Figure 1.1: ADC Symbol

1.1 How ADC works

Analog signal is time continuous signal. i.e. at any time analog signal has a value. To convert this signal to digital world it should match digital rules. Digital world is working on clocking. So if this signal shall be subjected to any digital manipulation, only values of the signal at certain times will be considered. This means not all values of the signal will be handled, digital world will take samples of the signal at specified time. This is first procedure of Analog to Digital conversion. Second procedure is to take right value of the sample. Finally a code should be generated for this sample, digital processing will work on this code of the sample. That is the idea of analog to digital conversion (to take samples of signal ”Sampling”, record values of samples ”Quantization”, generate code for these values ”Coding”). Figure 1.2 shows main procedure done in ADC. What this figure depicts is analog signal is entering the ADC block, after sampling process samples will be generated. After that quantization process will work on these samples. According to each quantized value, a code will be generated for it so

finally it will appear as binary digits that digital world can work with.

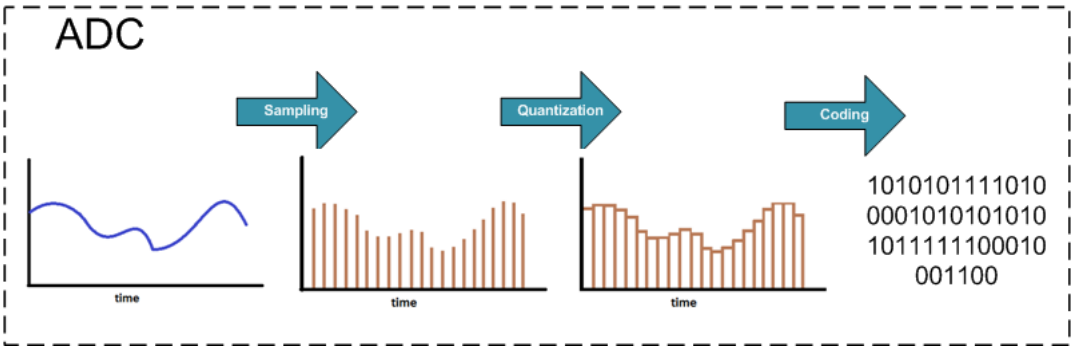

Figure 1.2: ADC Main Operations.

1.2 Sampling

By sampling a signal we are attempting to represent the time dependence of the signal by a discrete set of samples. As a practical convenience we typically depend on a fixed sample interval.

The conventional ADC process transforms an analog input signal x(t) into a sequence of digital codes x(n) at a sampling rate of $f_S = \frac{1}{T}$, where T denotes the sampling interval. The sampling function is equivalent to modulating the input signal by a set of carrier signals having frequencies of $0,\ f_S,\ 2f_S,\ 3f_S, \ldots$, see figure 1.3.

The sampling theorem simply says that one must sample a signal at a minimum of twice the highest frequency of interest. Thus to represent a signal with frequency components between zero and 1 KHz one must sample the signal at 2 KHz or above. As a practical matter where one is observing the time dependence of a signal directly, one may choose to sample at a rate of four times the highest frequency or more so as to have multiple samples over the period of the highest frequency.

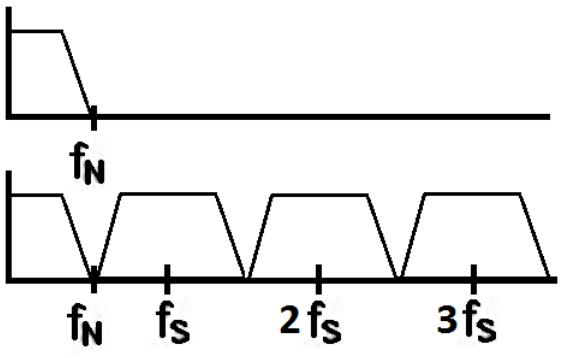

Figure 1.3: Sampling Frequencies.

Many A/D converters such as successive approximation register (SAR) and flash converters operate at the Nyquist rate (f_N). These converters typically sample the analog signal at a sample frequency (f_S) approximately twice the maximum frequency of the input signal.

Sigma delta A/D (Sigma-Delta modulator will be discussed in next chapter) converters do not instantly digitize the incoming analog signal into a digital sample of n-bit precision at the Nyquist rate. Instead, a sigma delta ADC over samples the analog signal by an over sample ratio of (N) resulting in $f_N << f_S$ (over sample rates of 16, 32, 64, 128 are common) [1].

1.3 Coding

To understand what is meant by coding, have a look on figure 1.4.This figure depicts a signal that is sampled uniformly. The value of the sampled voltages should be converted to a digital code. To assign a code for a voltage value, your code should be defined before. For instance, signal in figure 1.4 a code should be defined initially i.e. say if we have a voltage value of 10 volts, this will be assigned a code of 1111. voltage values of 5 volts will be assigned as 1100. voltage values of 0 volts will be assigned as 1000, voltage values of -5 volts will be assigned as 0100 voltage values of -10 volts will be assigned as 0000.

So we can create coding table for sampled values as shown in table 1.1

Sampled Value (volts)	Code
-10	0000
-8.75	0001
-7.5	0010
-6.25	0011
-5	0100
-3.75	0101
-2.5	0110
-1.25	0111
0	1000
1.25	1001
2.5	1010
3.75	1011
5	1100
6.25	1101
7.5	1110
8.75	1111

Table 1.1: Coding Example Values

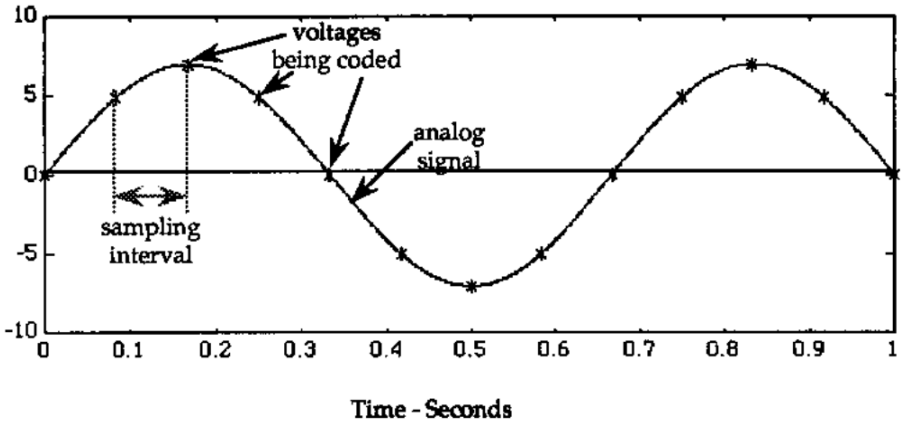

Figure 1.4: Analog Signal Example

So you will have bit stream at the output that is representing the value of the sampled voltage. This output of the ADC is digital signal:

- in Serial or Parallel form
- encoding of the output signal shall be considered (Unipolar/Manchester/...)

1.3.1 Dynamic Range

one of the characteristics that determines an ADC, is its dynamic range. It is simply how many bits are representing the analog signal samples. In previous example we had 4 bits output from the ADC, so its dynamic range is 4 bits.

1.4 Quantization

It wasn't easy to illustrate quantization without understanding coding and sampling first. Since quantization is intermediate stage between sampling and coding, so its function is to take samples values from sampling stage and represent it in right format for coding. Since analog signal is not pre-defined, we can't expect which value will be sampled. In previous example in coding section, it can happen that sampled voltage is 6.7 volts, we don't have code for this value. In this case comes importance of quantization. Quantization will represent this values to another defined value in our example here it will be considered as 7 volts. See figure 1.5 to understand how undefined values will be considered.

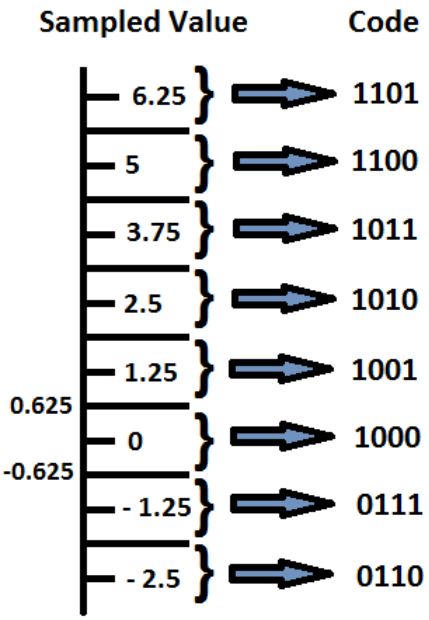

Figure 1.5: Quantization Example Values

In this figure any value between -0.625 and 0.625 will be considered as 0. The reason is we assumed our ADC can generate 4 bits digital symbol (i.e. Dynamic Range). This means that we can generate $2^4 = 16$ codes. We assumed also that our signal will be between -10 to 10 volts, so we have voltage range (10 - (-10)) = 20 volts. This voltage range shall be distributed over 16 codes, so we have to consider that each 1.25 volts can have a code. Now if we have a value in between, it shall be approximated to nearest value. So any value between 0.625 and 1.875 will be considered as 1.25 and consequently it will get a code 1001. A full map of how each value should be approximated is shown in figure 1.6

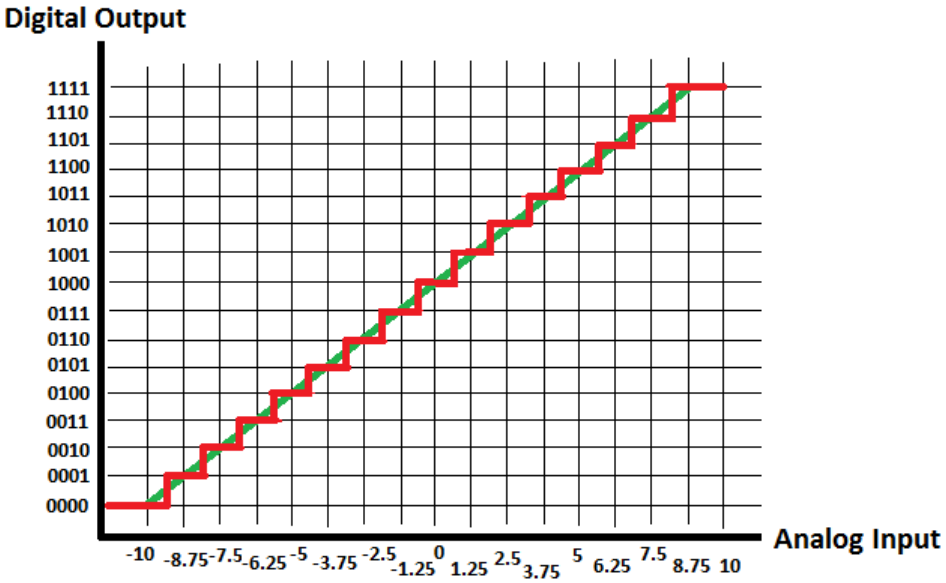

Figure 1.6: Quantization Map

1.4.1 Quantization Error

Due to this approximation operation, coded values is not representing accurately the real value. You can guess that easily. That is why we have quantization error. This error is shown in figure 1.7.

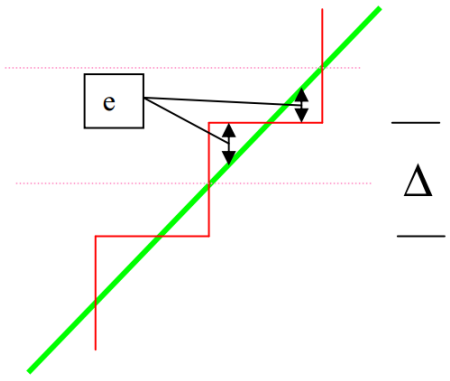

Figure 1.7: Quantization Error

Since we have a step of 1.25 volts (i.e. Δ = 1.25 volts), we have then maximum quantization error as +0.625 or –0.625. Normally quantization error (sometimes is called quantization noise) is $e = \pm\frac{1}{2}\Delta$ where $\Delta = \frac{V}{2^N}$ (V is fullscale voltage, N is dynamic range).

Chapter 2

Sigma-Delta Architecture

Before showing which architectures are used currently to build an ADC, it is important to remember that based on theory introduced before, ADC's suffers from errors like quantization error. Too many architectures were developed then to optimize operation of ADC as much as possible. General speaking, to improve quality of the ADC there are 2 possible options: increase sampling rate or increase dynamic range (no. of sampling bits).

Based on that many architectures were introduced like Flash ADC, Successive approximation, pipelined and Sigma-Delta. We will focus in this document on Sigma delta modulator. It worth mentioning here that each architecture of these ADC is designed to meet specific requirements in terms of speed and resolution. Figure 2.1 shows each architecture and which applications it fits more.

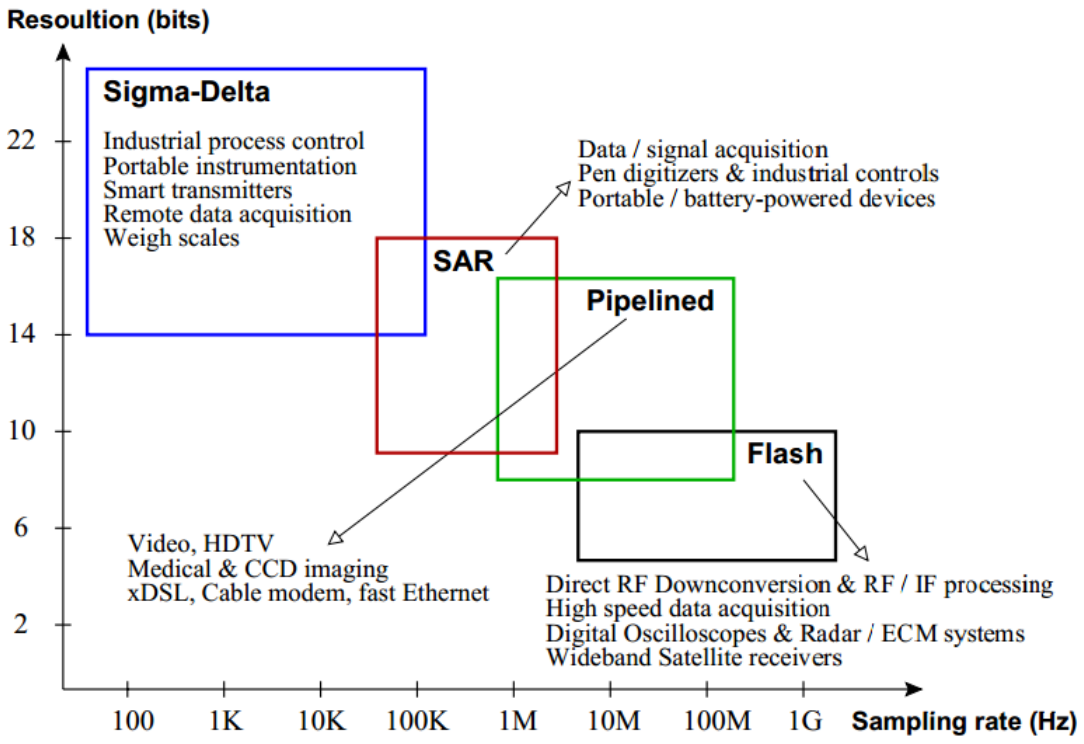

Figure 2.1: ADC architectures and its Applications[2].

2.1 Sigma-Delta ADC

This type of ADCs is based on ”Sigma-Delta modulator” as this modulator in its origin was used as pulse code modulator (PCM). It has been developed later to be used in ADC. Architecture of this ADC is shown in figure 2.2.

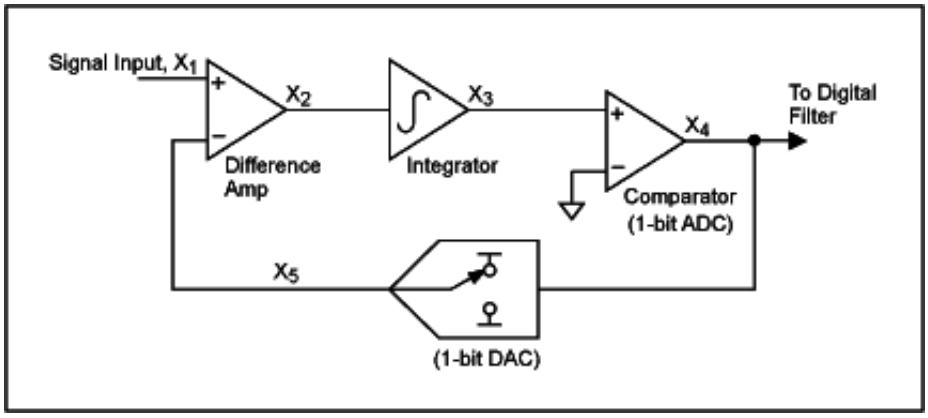

Figure 2.2: Sigma-Delta Architecture [3].

Input signal X1 is the signal needed to be converted. Subtraction operation between this signal and a reference voltage X5 is done. This difference is integrated to give signal X3 which is compared with reference voltage (usually 0 volt). So output of the comparator is high or low voltage X4. This X4 is PCM signal. This will be converted back to analog signal X5 in order to repeat the operation again. To visualize this operation, you can follow it in figure 2.3.

Important is to note that output of Sigma-Delta is fed to decimater filter in order to produce needed code.

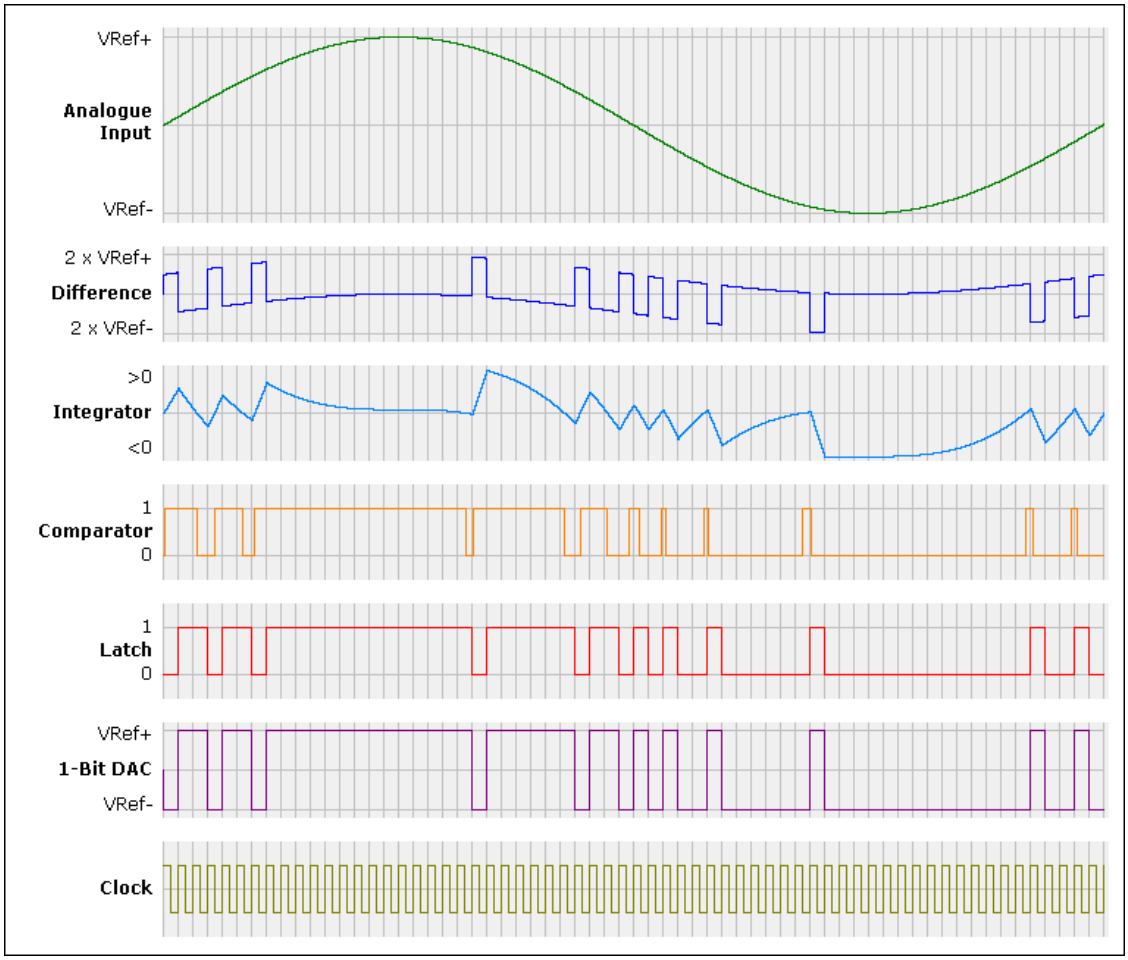

Figure 2.3: Signal flow in Sigma-Delta [4].

2.2 Sampling in Sigma-Delta ADC

The driving force behind delta modulation and differential PCM was to achieve higher transmission efficiency by transmitting the changes (delta) in value between consecutive samples rather than the actual samples themselves. So sampling here is directly reflected as difference between input signal and reference voltage.

2.3 Quantization in Sigma-Delta ADC

Quantization here is done by the comparator. It was discussed before that quantization step is needed to set the sampled voltage to a defined value. Here you can consider that our defined values are the high and low voltages. That is because idea of Sigma-Delta is to produce PCM signal that will be coded to appropriate digital values.

2.4 Coding in Sigma-Delta ADC

In sigma-Delta modulator, input analog signal is converted into digital bit streams (1s and 0s). One can observe a bit, either 1'b1 or 1'b0 coming at every clock edge of the modulator. A

decimation filter after the Delta-Sigma modulator receives the input bit streams and, depending on the over sampling ratio (OSR) value, it gives one N-bit digital output per OSR clock edge. For example, if we consider OSR to be 64, then the Filter gives one N-bit output for every 64 clock edges (64 data outputs of the modulator). Here N is the resolution of the Sigma-Delta modulator.

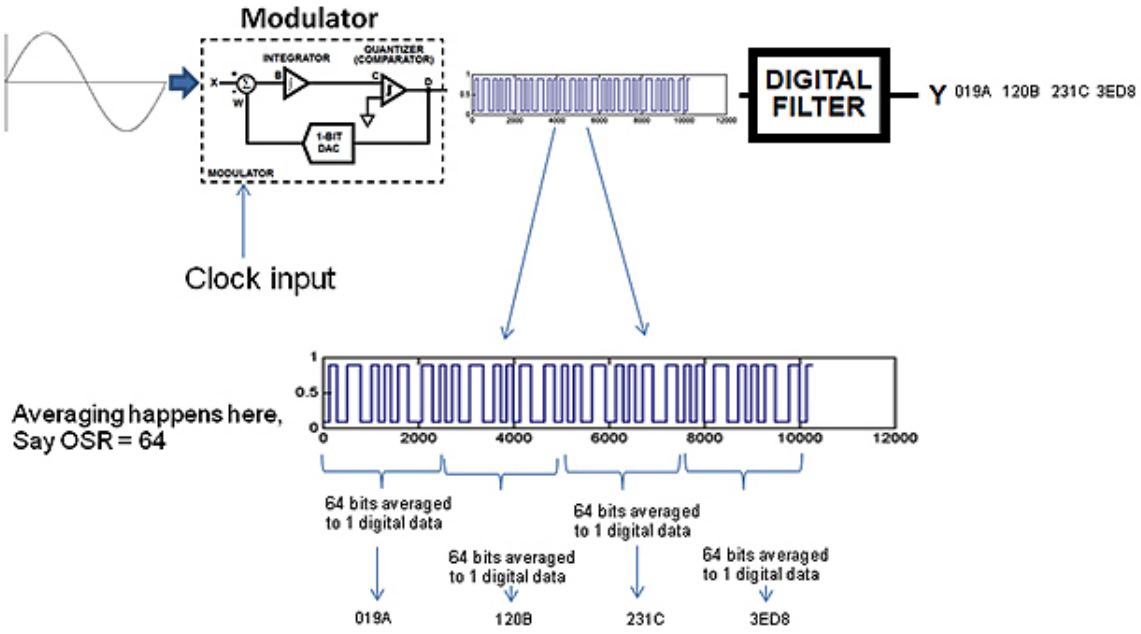

Figure 2.4: PCM of Sigma-Delta [5]

and to understand how the average of bit stream is calculated, assume a single bit Sigma-Delta ADC. The 1-bit A/D stream is digitally filtered to obtain an n-bit representation of the analog input. In simplified terms the 1-bit A/D stream is accumulated over (N) sampling cycles and divided by (N). This yields a decimated value which is the average value of bit stream from the modulator as shown in figure 2.5.

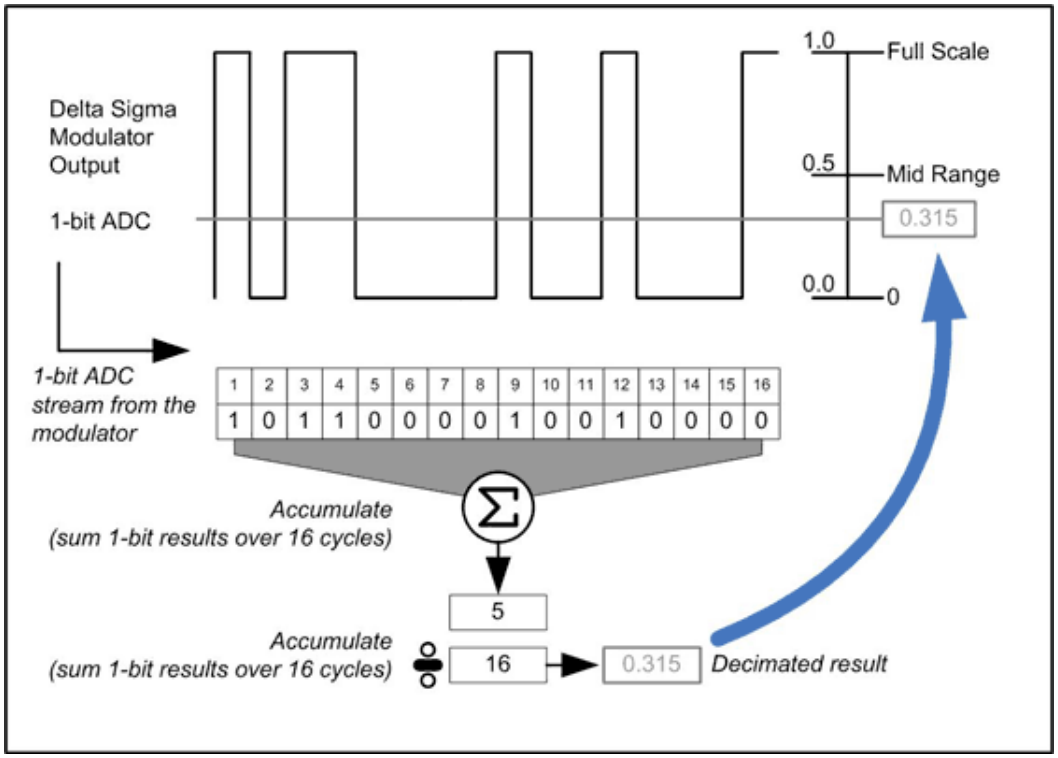

Figure 2.5: Average Calculation of PCM Signal [6].

Bibliography

[1] http://www.triadsemi.com/2007/01/25/how-to-design-a-16-bit-sigma-delta-analog-to-digital-converter/

[2] A TIQ BASED CMOS FLASH A/D CONVERTER FOR SYSTEM-ON-CHIP APPLICATIONS, Jincheol Yoo, 2003, PhD Thesis.

[3] http://www.maximintegrated.com/en/images/appnotes/1870/1870Fig04.gif

[4] http://www.beis.de/Elektronik/DeltaSigma/DeltaSigma.html

[5] http://www.triadsemi.com/2007/01/25/how-to-design-a-16-bit-sigma-delta-analog-to-digital-converter/

[6] http://www.embedded.com/design/debug-and-optimization/4406844/The-basics-of-sigma-delta-analog-to-digital-converters-