

Description of the example for exercise

(Implementation and Simulation of Mixed Signal Models in VHDL-AMS)

2 Sigma delta convertor

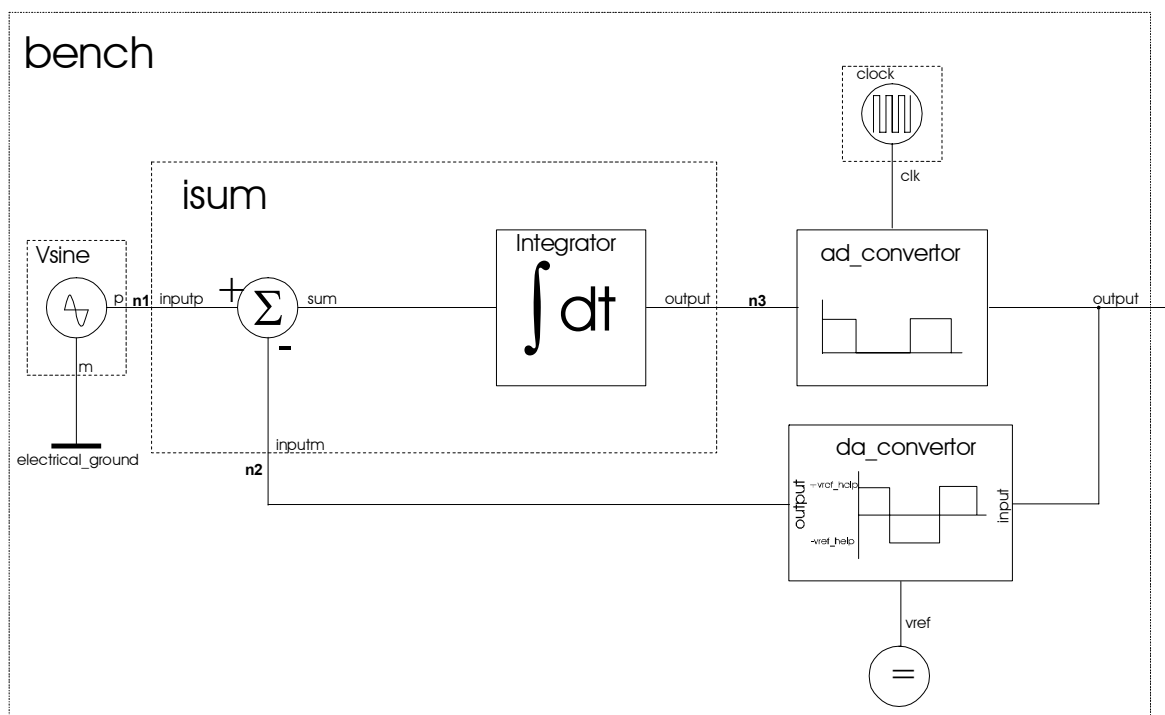
The simulation of AD or DA convertors is a typical example for the mixed signal modelling. Therefore you shall now start with the modelling of a sigma delta modulator as basic part of sigma delta ADC in VHDL-AMS.

2.1 Basics of the sigma delta ADC

In a sigma delta convertor the analogue signal first passes a sigma delta pulse density modulator to be converted into a high-frequency bit sequence with resolution of 1 (typically). The output signal of the modulator is then changed by a digital low pass filter into high-resolution parallel words with a considerably lower sampling rate.

2.2 Sigma delta modulator

Block circuit diagram of a sigma delta modulator with 1 bit resolution:

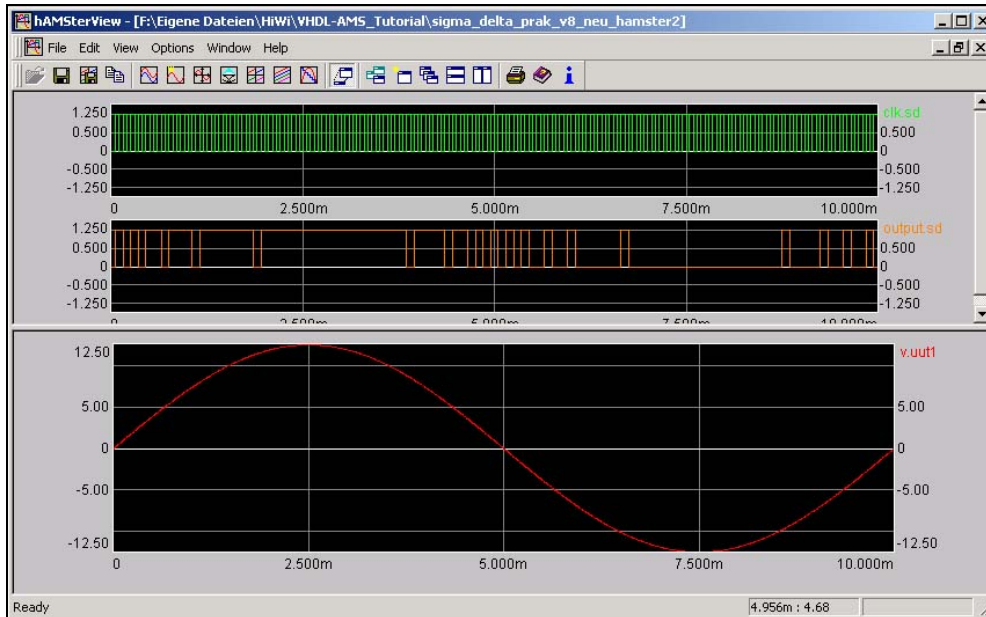


A sigma delta modulator changes an analogue input signal into a pulse sequence. The pulse density of the output signal (node *output*) is proportional to the amplitude of the input signal. It is a feedback loop including an analogue adder, an integrator and a quantizer with a digital as well as quasi analogue output. The quantizer is usually realized by an AD convertor of low resolution whose output signal is converted back to quasi analogue signal by a simple DA convertor. For most applications it is sufficient to use, a 1 bit ADC. A simple technical realization arises through that. As ADC merely a comparator (mostly a timed D flipflop) is then used, so the conversion can be realized by a simple threshold switch.

This roughly quantized analogue signal (approximation signal) is then compared with the input signal in the closed-loop control circuit. The difference is integrated and brought to the quantizer. As represented above, in the simplest case it is decided on the output voltage of the integrator (node *n3*) whether the quantized analogue voltage is positive or negative. If the integrator provides a negative voltage, the ADC switches to '0' at positive voltage on '1'. The DAC changes this bit sequence into a quasi analogue signal. It's output (node *n2*) is switched to $-v_{ref}$ if logical '0' appears at the input and changes to $+v_{ref}$ if there is a logical '1'. That means an input voltage between $+v_{ref}$ and $-v_{ref}$ can be converted. By the closed-loop control circuit the arithmetical mean average value of the output signal is held proportionally to the mean average value of the input voltage. To get parallel words with high resolution, the output signal must pass a digital decimation circuit which forms the temporal mean average value on a digital way. Parallel words result for lower repeating rate and higher resolution (> 1 bit) at the output of this filter.

A basic framework for the sigma delta modulator is provided in the file *sigdel.vhd*. It contains every component of the above model as prototypes without functionality. They must only be filled up with the corresponding functionality and have to be connected among themselves. After modelling the corresponding functions in VHDL-AMS and translating the model successfully, please load the file *test_sigdel.vhd*. A test environment for the sigma delta modulator is implemented here. Translate this source text and then simulate the model *bench*.

Respect: Clock in the Testbench is prepared for a period duration of 100 us, therefore it is required to **keep minimal simulation step size shorter than 50 us**.



Simulation Result of the sigma delta modulator

2.3 The Decimation filter

To get a high resolution digital value out of the pulse density information it is required to form the temporal mean average over the pulses of the sigma delta modulator. If logical '1' appears at the output, the modulation method loops back the reference voltage and for logical '0' the negative value of reference. That means, logical '1' at the output stands for maximum value of the desired resolution (2^n). So within 8 Bit ADC a logical '1' at the output of the modulator means the output value 128 (for bipolar conversion).

2.4 Realization as a counter

The simplest variant for a decimation filter would be a counter with the breadth of the desired resolution. The counter is clocked from the same source as the sigma delta modulator and simply counts the impulses in a period of time. For this realization no additional adders or multipliers and only few memory is required.

To obtain a resolution of n bit, it would be necessary to sum up 2^n times with this method. Very exact output values result from this procedure but the sampling rate of the system is reduced by the factor 2^n comparing to the sampling rate of the modulator.

This decimation circuit was already implemented in the Testbench and must only be called. To use the counter variant, the architecture to be taken advantage of, *counter*, must be indicated in brackets.

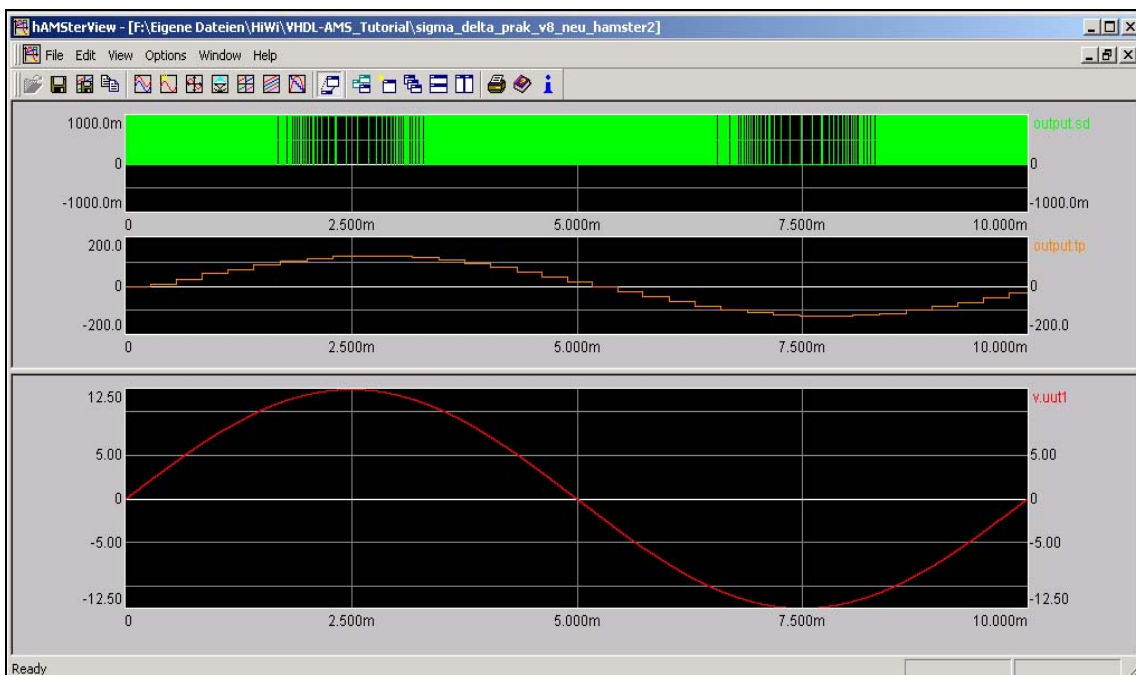
To start the simulation with the counter for the averaging, change the Testbench *test_sigdel.vhd* by inserting a new component in the *ARCHITECTURE*:

```
TP: ENTITY Decimation (counter) PORT MAP (clk, output, TPout);
```

Furthermore the signal *Tpout* must be declared in the interface description. A look in the model *Decimation* tells that this is a signal of the type *INTEGER*:

```
SIGNAL: TPout INTEGER:= 0;
```

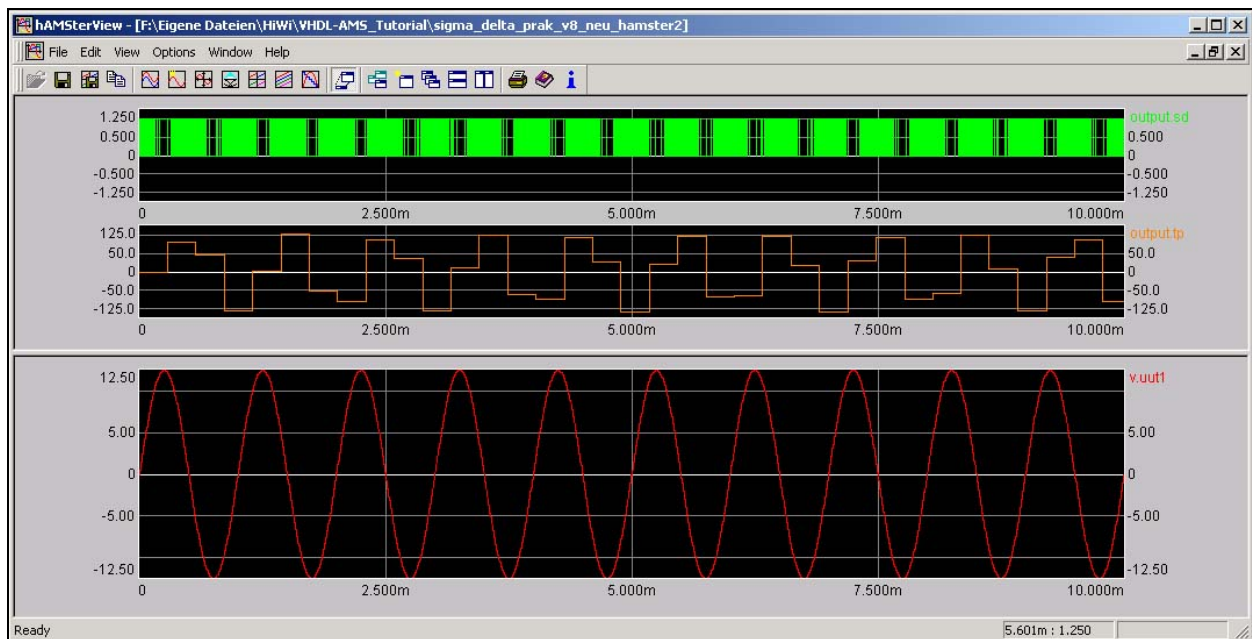
To get an acceptable result, the clock frequency of the sigma delta modulator has to be increased now. Take 1us here, choose a minimal step size of 10 ns and finally start the simulation.



Simulation results with counter for the averaging

2.5 Digital FIR filter

The variant with using a counter for the averaging is very exactly but unfortunately just as slow too. 256 clock cycles are needed for a result with 8 bits of resolution (like in the example). With that the sampling rate of 1 MHz in the modulator degrades to $1000 \text{ kHz}/256 = 3.9 \text{ kHz}$. Change the Testbench so that 1 kHz ($frequency = 1000.0$) sine appears at the input of the sigma delta convertor instead of 100Hz and watch the results.

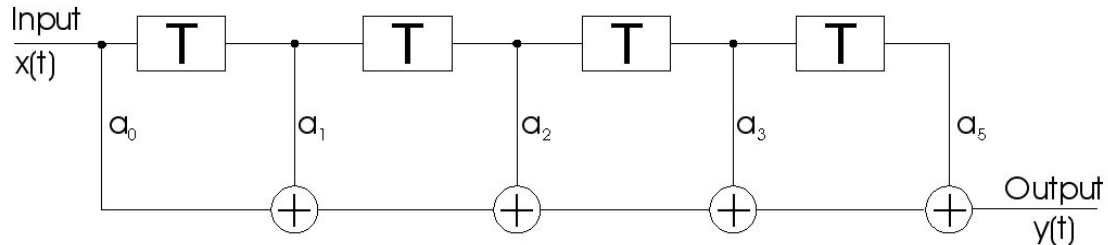


Simulation result at 1 kHz

In analogue circuit design a lowpass may be used to form a continuous signal from a pulse density. In the digital technique the behaviour of a lowpass circuit can be reproduced.

The simplest possibility to reproduce a lowpass digitally is the design of a FIR filter (Finite Impulse Response). To do so, a digital circuit which copies the impulse response of a lowpass in a digital way must be designed. A continuous stream of digital words appears at the input of the filter. Every word is stored in row after being delayed in time and weighted. The sum of all weighted digital words is at the output of the filter.

Scheme of a FIR filter:



The present input is multiplied by a_0 . The stored value of previous clock which got weighted with a_1 is added up etc. So T stands for one clock cycle in this design. If output is called $y(T)$ and the input is $x(T)$ the output results in:

$$y(T) = x(T) * a_0 + x(T-1) * a_1 + x(T-2) * a_2 + x(T-3) * a_3 + x(T-4) * a_4$$

For this example filter, one single impulse with duration of one clock cycle causes a sequence of 6 resulting values at the output. In first clock cycle the value of the pulse multiplied by a_0 arises at the output, in second clock cycle the pulse is multiplied by a_1 . In this case a digital word larger than Zero is understood as an impulse.

So the impulse response of the filter can be adjusted directly by changing the factors a_0 to a_5 . If $g(t)$ is the desired pulse answer equation $a_n = g(T * n)$ is valid.

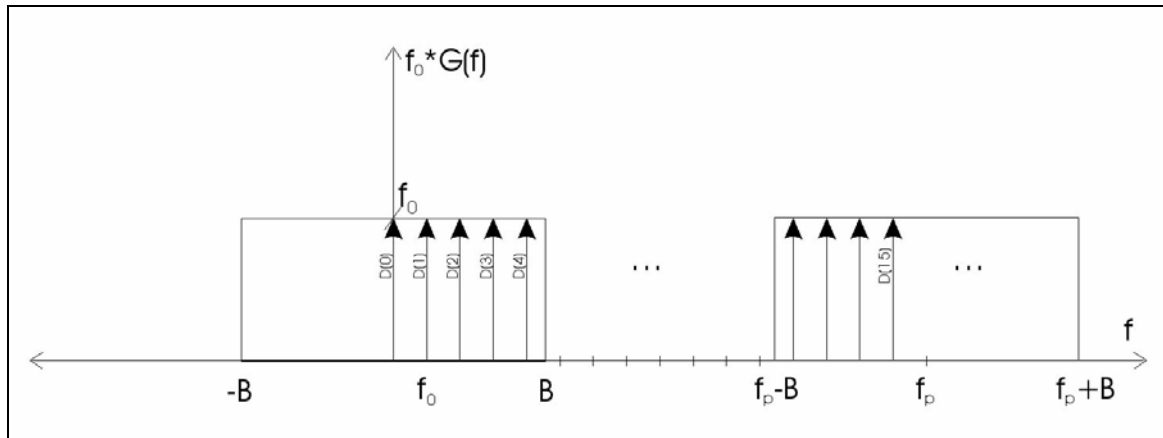
The course of the impulse response $g(t)$ can be found out with the help of a continuous or discrete inverse Fourier transform from the desired frequency response.

By the lowpass filtering the mean average value of the output signal of the Sigma delta modulator is formed. It is therefore possible to achieve a high-resolution output signal from the 1 bit input signal. If the input signal is 'High' this corresponds to the maximum value of the conversion, if it is 'Low' this corresponds to the minimum value (according to the feedback's block circuit diagram).

For example:

The desired frequency response shall be rectangular transfer function with the breadth $B = 100 \text{ kHz}$:

For the analysis by the DFT the transfer function must be periodical:



The sampling rate is given in the time domain by the clock of the filter, so the period in the frequency domain (f_p) results in:

$$t_0 = 1\mu s = 10^{-6} s$$

$$f_p = \frac{1}{t_0} = 1MHz = 10^6 s^{-1}$$

After specification of the desired number of supporting values ($N = 16$) the sampling rate in the frequency domain ($= f_0$) can be investigated:

$$t_p = N * t_0 = 16 * 10^{-6} s$$

$$f_0 = \frac{1}{N * t_0} = \frac{1}{16 * 10^{-6} s} = 62,5 * 10^3 s^{-1}$$

Assuming now that it is a straight (axially symmetric) transfer function and an even number of supporting values, a simplified form of the DFT can be used:

$$D(\mu) = f_0 * G(\mu * f_0)$$

$$d(n) = \sum_{\mu=0}^{N-1} D(\mu) * \cos\left(\frac{2\pi n \mu}{N}\right) = f_0 * \sum_{\mu=0}^{N-1} G(\mu * f_0) * \cos\left(\frac{2\pi n \mu}{N}\right)$$

so the a_n can be found out by the following equation (as described in [1]):

$$a_n = t_0 * d(-n + N / 2)$$

For the example this results in the following characteristics:

$$\begin{aligned}
d(0) = & 62,5 \cdot 10^3 \cdot [G(0) + G(1 \cdot f_0) + G(2 \cdot f_0) + G(3 \cdot f_0) \\
& + G(4 \cdot f_0) + G(5 \cdot f_0) + G(6 \cdot f_0) + G(7 \cdot f_0) \\
& + G(8 \cdot f_0) + G(9 \cdot f_0) + G(10 \cdot f_0) + G(11 \cdot f_0) \\
& + G(12 \cdot f_0) + G(13 \cdot f_0) + G(14 \cdot f_0) + G(15 \cdot f_0)]
\end{aligned}$$

At a rectangular transfer function with a bandwidth of $2 \cdot 100$ kHz, all values between 100 kHz and $1000 \text{ kHz} - 100 \text{ kHz} = 900$ kHz results in zero. That's why all members for which $(100/62,5) < \mu < (900/62,5)$ need not to be calculated. All others are equal to 1:

$$\begin{aligned}
d(0) &= 62,5 \cdot 10^3 \cdot [G(0) + G(f_0) + G(15 \cdot f_0)] \\
&= 3 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}
\end{aligned}$$

$$\begin{aligned}
d(1) &= 62,5 \cdot 10^3 \cdot [1 + \cos(2 \cdot \pi \cdot 1 \cdot 1/16) + \cos(2 \cdot \pi \cdot 1 \cdot 15/16)] \\
&= 2,848 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}
\end{aligned}$$

$$\begin{aligned}
d(2) &= 62,5 \cdot 10^3 \cdot [1 + \cos(2 \cdot \pi \cdot 2 \cdot 1/16) + \cos(2 \cdot \pi \cdot 2 \cdot 15/16)] \\
&= 2,414 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}
\end{aligned}$$

$$d(3) = 1,765 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(4) = 1,000 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(5) = 0,2346 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(6) = -0,4142 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(7) = -0,8478 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(8) = -1,000 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(9) = -0,8478 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(10) = -0,4142 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(11) = 0,2346 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

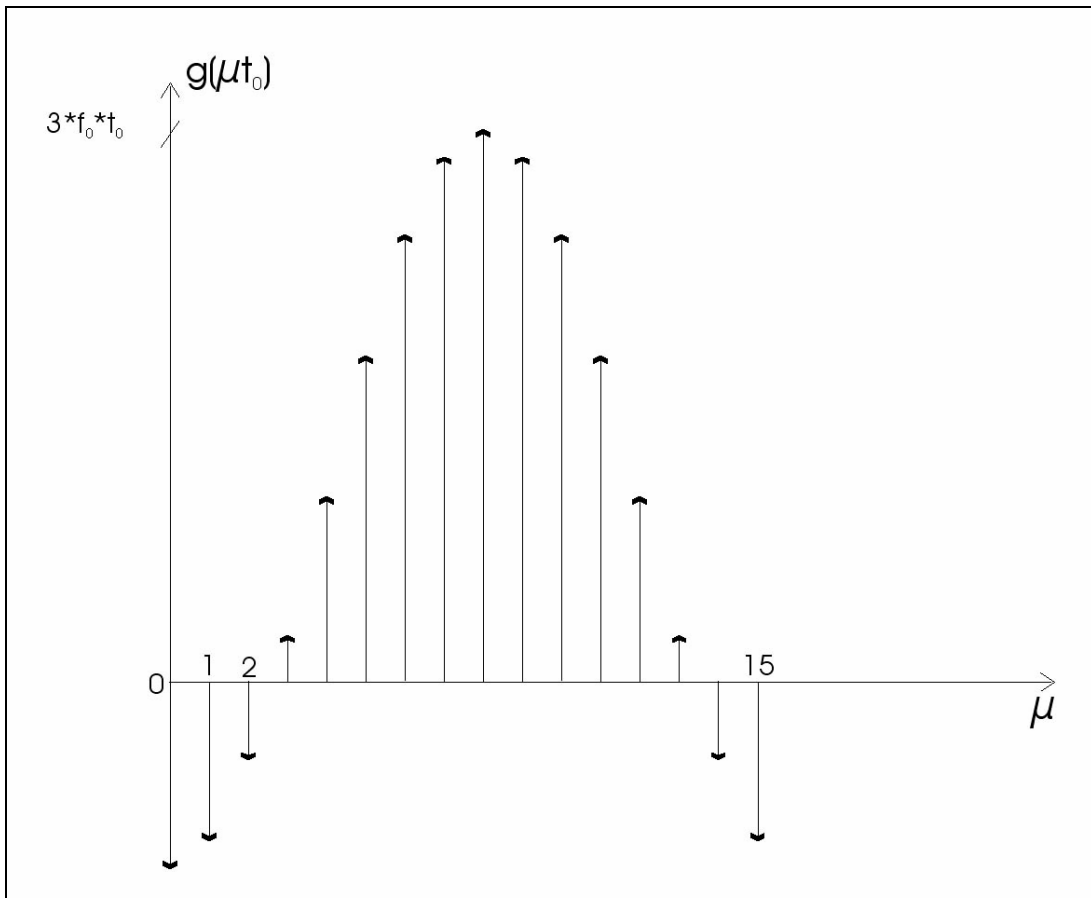
$$d(12) = 1,000 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(13) = 1,765 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(14) = 2,414 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

$$d(15) = 2,848 \cdot 62,5 \cdot 10^3 \text{ s}^{-1}$$

The impulse response of the calculated filter looks as follows:



The numerical values only must be scaled for the sigma delta convertor. With a step at the input, the output of the filter must even out on the highest conversion value after a finite time. The sum of all impulses must therefore correspond to the highest digital numerical value:

A resolution of 8 bits results in a theoretical change area of -127... 127:

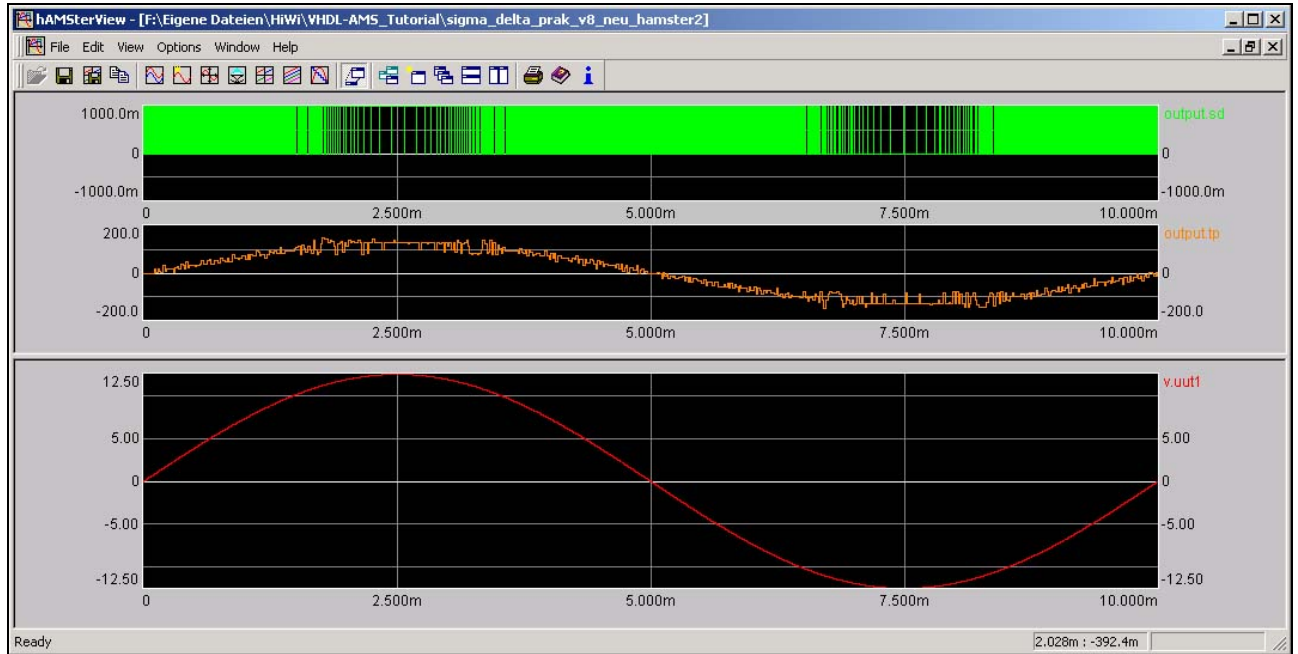
$$k = \frac{127}{\sum_{n=0}^{N-1} a_n} = \frac{127}{3 + 2 * (2,848 + 2,414 + 1,765 + 1 + 0,2346 - 0,4142 - 0,8478) - 1} = 7,93$$

So every value is multiplied by 7.93 now:

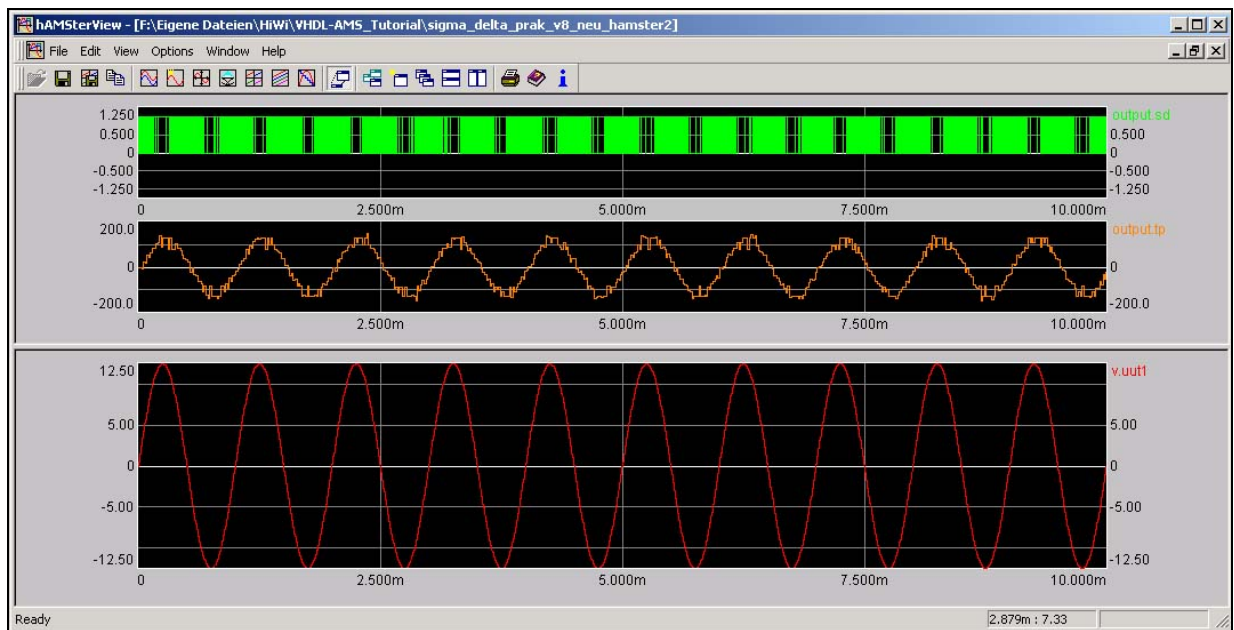
$$a_0 = -1,000 * 7,93 = -7,93 = -8$$

$$a_1 = -0,8478 * 7,93 = -6,72 = -7$$

a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	Σ
-8	-7	-3	2	8	14	19	23	23	23	19	14	8	2	-3	-7	127



Simulation result with example filter and 100 Hz test signal



Simulation result with example filter and 1 kHz of test signal

The results of the simulation show that it is still very inaccurate after the filtration by this very simple filter with 16 coefficients. To get an exact result it is necessary to connect more decimation filters in series or use one with more coefficients.

This should be only a small introduction to outline the thematic of digital filter behaviour.