5

Mixed-Signal Circuits Modelling and Simulations Using Matlab

Drago Strle
University of Ljubljana, Faculty for electrical engineering,
Tržaška 25, Ljubljana,
Slovenia

1. Introduction

Continuous advances in IC (integrated circuits) processing technologies offer the possibility to produce integrated circuits with increased complexity and capability at a reduced cost. In addition, integrated circuits are composed more and more of heterogeneous embedded systems with various kinds of digital, analogue, and mixed-signal circuits and sensors that are integrated on the same IC. In the future, the number of different embedded systems integrated on the same IC, as well as the number of all modules and the complexity of all modules, will increase. In addition, more and more heterogeneous modules (including analogue and mixed-signal circuits, sensors and actuators, micro-electro-mechanical-systems) will be integrated on the same IC.

Currently, well-established circuit design tools exist for the circuit level design of analogue circuits; even better design tools exist for the systematic and hierarchical design of large digital circuits, including an important aspect of any reliable digital circuit, that is, testability, which is also well covered by existing digital EDA tools (EDA stands for Electronic Design Automation tools). There are no such tools available at the moment for analogue and mixed-signal circuits except maybe Saber. Saber can help to model, design, and simulate a complete mixed signal system in a short amount of time. It may also be used in several analogue and mixed signal blocks and different kinds of integrated sensors. With Saber (Synopsis, 2004) it is possible to model such circuits; however, Matlab/Simulink is much more convenient because of many different toolboxes and libraries available. Matlab, together with Simulink, offers a hierarchical environment and computation engines that make it possible to model, simulate and design complex circuits in a very efficient way on a high hierarchical level. In addition, modules may be described, modelled, simulated, and designed at different abstraction levels and at different levels of detail. For example, analogue circuits could be modelled in a very simple way, that is, taking into consideration only system level aspects like transfer functions; conversely, more details about the implementation, such as offset voltage, gain bandwidth product, slew rate, thermal and 1/f noise, and parameters such as nonlinearity, could be added, when necessary. For digital circuits, it might be necessary to produce a bit-true model early in the design to take all aspects of the design into consideration. Appropriate decisions early in the design can improve the efficiency of the design process. Since large modern IC's usually contain many mixed-signal functions, it is necessary to model and simulate the analogue and the digital

part of the mixed-signal circuits, and possibly also the sensors, with most important non-ideal effects included.
Good examples of complex mixed-signal circuits are high-resolution Σ-Δ AD and/or DA converters. These mixed-signal circuits trade time with accuracy. To get some samples of digital results, many clock samples of analogue circuits must be processed. Using circuit simulator for the verification of such a module would require enormous amount of time. For example, to be able to simulate and calculate the performance of a fifth order modulator using normal EDA tools, more than 1000 hours on a 3GHz computer would be required; this amounts only to some simulations, which of course is not acceptable. The simulation of a high-level model of the same circuit, with most of the important non-ideal effects included, takes only a couple of minutes if the circuit is modelled on a high hierarchical level using the Matlab. In addition, because of the speed of simulation, depending on the specifications, such a methodology could be used to optimize certain parameters and determine the minimum requirements for the building blocks.
This chapter will explain how to model, design, and simulate mixed-signal circuits efficiently by using Matlab and Simulink. The design, modelling and simulation of a fifth order Σ-Δ A/D converter will be used as an example for high-resolution mixed-signal circuit. Generally, analogue and digital parts, and possibly even the sensors, are modelled according to needs. At the beginning of the design process, only high-level models are used; non-ideal effects are not yet implemented and only basic functionality is checked. For example, a transfer function for linear circuits can be used to verify the concept and later on, non-ideal effects are added to the model to verify parameters of analogue circuit implementation. For the digital part, only a discrete time model with double precision of variables are used at the beginning, while later on, a bit-true model is used. In this way, it is possible to verify the correctness of the design at an early stage of the design process. Since a complete mixed-signal circuit is verified on a high hierarchical level, it is possible to simulate all parts of the circuit at the same time and in a very short time. In this way, the efficiency of the design process is highly improved and the design time is shortened. The main objective of this chapter is to explain how we can model and simulate mixed-signal circuits in an efficient way, using a fifth order Σ-Δ A/D converter as an example.
For digital design, one of the trends is to integrate Matlab/Simulink tools into EDA CAD tools (CAD stands for computer aided design) for IC design (Perelroyzen, 2007) in order to facilitate and speed up the design of the digital integrated circuits and more specifically, to simplify and speed-up the design of the digital signal processing modules. For digital DSP (digital signal processing) designs, the Matlab/Simulink environment provides the opportunity to design and use DSP IP blocks and their models, to simulate those models on a functional level. Some companies (Xilinx, Altera, Synplicity) use Simulink for the implementation of IP-based design flow. The design starts with Matlab with the design of the signal processing algorithm; when the algorithm is developed, the corresponding IP blocks are selected from the library and then the RTL (register transfer logic) description of the system is synthesized (Moris, 2004). RTL description is then used in the synthesis of FPGA (field programmable gate array) content. For VLSI (very large scale integrated circuits) design, the net-list is synthesized from the RTL or HDL (hardware description language) code.
The methodology and tools for analogue and mixed-signal circuits that are currently used are based on a Spice-like simulation environment; that is good for simple analogue circuits. As soon as many modules are involved, or, as in modern heterogeneous designs, many

different modules are integrated, no tool exists that can handle the complex nature of such circuits and no tool is currently available for the design of such heterogeneous and complex systems. For such designs, many different specialists must work together. Because of that, there is a real need for heterogeneous design environment and simulation tool that can be used on a high hierarchical level and that integrates the design solutions of many different specialists. We believe that the Matlab/Simulink environment provides the appropriate tools where representations at the high hierarchical level enables the efficient design of digital, analogue, and mixed-signal circuits/systems.
In this chapter, we will present a methodology for the efficient design, modelling, and simulation of mixed-signal integrated circuits using the Matlab/Simulink environment; specifically and as our example, we shall design and model a fifth order Σ-Δ analogue to digital converter module. In Section 2, a short description of a top-down approach for design of a mixed-signal circuit is given, together with a short description of what is needed to design such circuits/systems successfully. It is assumed that the readers know all the basic tips and tricks of the Matlab/Simulink environment. In section 3, the definition of a mixed-signal circuit is given, together with the list of some of the libraries and elements used for modelling such circuits. In section 4, a system level design procedure is given for the Σ-Δ modulator that is used as our example together with the description of the Simulink blocks that are used to model such nonlinear system. In section 5, the basic behaviour, the mathematical model, and the Simulink model of the discrete-time integrator is presented, together with the mathematical and Simulink models of the most important non-ideal effects. Two-simulation results will be presented in this section: the Spectrum of a BS (bit-stream) including non-ideal effects and a "scatter" plot of the SnR that results from the Monte Carlo analysis. Section 6 presents the decimator bit-true modelling and design, and a complete mixed-signal circuit/system simulation result.

2. Top down design approach

To verify a functional property of a complex system early in the design phase, the design starts from a simplified description at a high hierarchical level. This then proceeds to the modelling of each constituent building block in a more detailed way. Eventually, such a modelling is applied to the transistor level. Later on, the description and modelling of each constituent module is improved by adding lower level detailed information that comes either from experience or from the detailed design process that use other design tools and simulation results. For example, for analogue interface circuits, the following might be important parameters: thermal noise, 1/f noise, open loop gain, gain bandwidth, slew rate, nonlinearities, sampling, kT/C noise; for the DSP, on the other hand, a bit-true model might be important to be able to see the influences. For example, one circuit level simulation of a simple 2nd order Δ-Σ modulator using Spectre simulator requires more than 160 hours of the processor time; such simulation makes sense when the design is finished, that is, the simulation acts as a final check to see the effect of parasitic capacitances, for example. If such simulation tools are used throughout the entire design process, the simulation should be repeated several times, so the time needed for the complete design cycle would be prohibitively long. When such a modulator is modelled on a high hierarchical level that includes the important non-ideal effects, the simulation needs only a couple of minutes. This means that such a tool could really be used very efficiently: once the model fulfils the specifications, including the specifications of the most important non-ideal effects, the model could be taken as a specification limit for the detailed circuit design. Once the detailed circuit and layout design is finished, the results of the "Spice" simulations of each module are taken and added into the Matlab model, and the whole simulation cycle could be repeated again on a high hierarchical level. This methodology is very efficient; it shortens the design time for mixed-signal circuits considerably.

3. Mixed-signal integrated circuits

Mixed-signal integrated circuits contain analogue and digital modules that are integrated on a single semiconductor chip; these circuits are closely interconnected and interrelated. It is difficult or not appropriate to design, model, and simulate analogue and digital modules separately. A very simple example from a complexity point of view are the Δ-Σ A/D or D/A converters that include low noise analogue signal processing blocks as well as digital blocks that are used for dynamic element matching, decimation, clock-form generation, and others. Furthermore, in modern heterogeneous systems, different kinds of sensors are integrated and their operation quality is closely related to the operation of analogue and digital interface circuits for data acquisition, for controlling of the properties and for driving the sensors. Digital signal processing is included as the integral part to extract useful information and/or to prepare/correct the driving signals; an example of such a complex system is the integrated inertial measurement system, which, besides the analogue and the digital signal processing modules, also includes the MEMS accelerometers and Gyro sensors (Strle & Kempe, 2007).

For a mixed-signal heterogeneous system, there are no synthesis algorithms available and most of the work must be done "by-hand" by specialists, that is, by a system designer, a mixed-signal and/or analogue designer together with a digital designer. Therefore, it is very important for the whole design team to use a common design environment where everybody could participate efficiently in the design procedure. In addition to the design of a mixed signal circuit on a high hierarchical level, the design of functional tests for a mixed-signal system can be performed and verified on a high hierarchical level by using Matlab/Simulink. This makes the design procedure even more efficient. Last but not least, during the design of a mixed signal circuit, a Monte Carlo simulation might be needed to verify efficiently whether the specifications are fulfilled for the whole spread of certain process parameters.

3.1 Basic building blocks of mixed-signal circuits

Analogue circuits are composed of a variety of modules built of CT (continuous time) and/or DT (discrete time) circuits. The operation of CT circuits (linear or nonlinear) is best described by a system of linear or nonlinear algebraic and/or differential equations, while the discrete time systems are best described by a system of linear and/or nonlinear difference equations. Linear systems of both kinds can be described by corresponding S-domain (for CT circuits) or Z-domain (for DT circuits) transfer functions. Usually the System on Chip (SoC) contains a mixture of some or all of the mentioned modules that are implemented either in an analogue and/or in a digital way. Some of the most popular high-level building blocks that are used in mixed-signal systems are the following:

- Analogue signal processing blocks
    - Amplifiers, comparators, switches, trans-conductors,
    - CT filters, S-C filters, S-I filters

    - Mixers
    - Signal generators
    - A/D and D/A converters
    - And many others
- Digital signal processing blocks
    - Adders, multipliers, CPU
    - Filters, mixers
    - Memory, Ram, counters, shifters
    - I/O

Many other blocks not mentioned above might be needed for a complete description of a SoC (System on Chip).

3.2 Simulink blocks that can be used for mixed-signal modules

Matlab/Simulink offers many different libraries. In this chapter we will use only the most common libraries and the most common elements that are available in many versions of Matlab/Simulink. Some of the libraries and some of the library elements are listed below. The meanings and the settings of the important parameters of each individual block will be explained when we will use a particular block in our Σ-Δ modulator design example. The libraries (bold face list) and the most useful blocks within a library for modelling mixed signal circuits are the following:

- **Commonly used blocks**
    - Constant
    - Gain
    - In, Out, Terminator
    - Sum, Product
    - Scope
    - Data Type Conversion
    - Subsystem definition
    - Saturation
    - Etcetera
- **Continuous time blocks**
    - Derivative
    - Integrator, Transfer function, Zero-Pole
    - State-Space
    - Variable Time Delay
- **Discontinuous**
    - Saturation, Relay
    - Dead Zone
    - Quantizer
- **Discrete time blocks**
    - Zero-Order Hold
    - Integer Delay, Unit delay
    - Discrete time Transfer function
    - Discrete time Zero-Pole
    - Discrete State-Space

  - Discrete time Integrator
- **Logic and Bit Operations**
- **Math operations**
  - Gain, Product, Slider Gain, Bias
  - Sum, Sum of Elements, Product, Divide, Dot Product
  - Sign, MinMax
  - Math Function
- **Ports and Subsystems**
  - In, Out
  - Subsystem
- **Signal attributes**
  - Data Type Conversion
  - Signal Conversion
- **Sinks**
  - Out, Terminator
  - Display, Scope
  - To File, To workspace
- **Sources**
  - Constant, In
  - From File
  - Sine Wave, Pulse generator, Repeating sequence
  - Signal generator, Step
  - Band-Limited White Noise, Uniform Random Number
- **User defined functions**
  - Fcn, MATLAB Fcn

Of course, for different systems, it might be necessary to use some additional libraries such as:

- Communication library
- Signal processing library
- Control system library

4. System level design

The design, modelling, and simulations of a complex mixed-signal module using Matlab/Simulink will be presented, using a fifth order Σ-Δ A/D converter as a design example. Σ-Δ converters are currently very popular because of the possibility to achieve high-resolution A/D conversion with great efficiency using the elements from VLSI process with characteristics that are not very good and stable. Σ-Δ converter is composed mostly of analogue modules and small digital circuit for generating clock signals, executing dynamic element matching algorithm and in addition, a digital decimation filter that is a fixed-point digital circuit. Many books and articles have been written about the subject; two of them are (Schrier & Temes, 2005) and (De la Rosa, 2011). Most of them deal with basic synthesis algorithms and properties. However, practical design recommendations of how to bring such designs into life in the shortest time possible are not presented in those books. Predicting final characteristics early on in the design, taking into considerations the most important non-ideal effects based on real circuit behaviour (kT/C noise, thermal and 1/f noise, offset voltages, finite gain-bandwidth product (GBW), finite open loop gain (A0), slew rate (SR), spread of parameters due to technology spread, stability issues, latency issues, meta-stability issues, etcetera), are usually not taken into considerations. In this chapter, we will try to model the additional characteristics that are essential for a successful design and use this information as a kind of specifications for the circuit design process.

4.1 The basics of a Σ-Δ A/D converter design

The design of an A/D converter usually starts with specifications. The simple specifications may be the following:

- Nyquist rate after decimation (BW=425kHz),
- Required signal to noise ratio (SnR>80dB),
- Required Harmonic distortions (THD≤80dB),
- Maximum signal level and its spectral contents,
- Reference level,
- Supply voltage and supply current, and others.

The data in brackets are the specifications of our sample A/D converter. Usually, the designer selects the architecture according to her/his experience. We will select a Σ-Δ A/D converter architecture with as small oversampling ratio as possible and as small order as possible in order to have as small power consumption as possible. Since the required HD is very demanding, we do not want to use any of the available linearization and/or dynamic-element-matching techniques. The natural choice in that case is to use a single bit internal quantizer and single bit internal D/A converter. It is difficult to design such an A/D converter correctly. The design presents many design challenges from the system level to the modelling and the circuit level because it needs a deep understanding of the system level design as well as the circuit level design issues. In addition, for a one bit internal quantizer, the circuit is highly nonlinear, which means that it is not possible to complete the design using simple AC design principles, transformations, and methods that are valid for the design of "linear" circuits. Despite the fact that linear models offer a simplified insight into the operation of the Σ-Δ modulator, the design is not complete without extensive simulations where influences of nonlinearity and other circuit parameters to the SnR (signal to noise ratio), BW (bandwidth), HD (harmonic distortions), noise, and stability may be checked. Modelling on a high hierarchical level makes it possible to explore the design space very efficiently and the results of high-level simulations could be used as inputs to the circuit design.

4.2 Synthesis of a discrete time noise transfer function

The NTF (noise-transfer-function) can be synthesized according to the definition given in subsection 4.1. The selected implementation might be an S-C (switched capacitor) loop transfer function implementation and therefore a discrete time model is selected. The simplest way to synthesize the noise transfer function is to use the appropriate tool; one such tool that is using Matlab is described in (Schrier & Temes, 2005). The link to download a latest version of toolbox called "delsig" is given in (Schreier, 2009). With the help of this toolbox, it is possible to synthesize the required NTF(z) in an efficient way; z-domain equation for the specifications given in subsection 4.1 is given in equation (1). Fig. 1 shows the plot of the |NTF(z)| and the pass-band detail.

$$NTF(z) = \frac{(z-1)(z^2 - 1.998 \cdot z + 1)(z^2 - 1.994 \cdot z + 1)}{(z - 0.8116)(z^2 - 1.676 \cdot z + 0.7128)(z^2 - 1.837 \cdot z + 0.8782)} \tag{1}$$

It is possible to implement such transfer function in many different ways. Based on additional requirements such as small power supply consumption and stability under different signal conditions, we selected feed-forward architecture with only one feedback, presented in Fig. 2. It has a one-bit internal quantizer, a one bit internal D/A converter, and a feed-forward structure with one feedback coefficient b_{ref1}.

Coefficients α_0, δ_0, $\beta_{1,2}$, $\gamma_{1,2}$, $\eta_{1,2}$ and $\lambda_{1,2}$ from (2) are related in a complex way to the feedback and feed-forward coefficients represented by vectors **a, g, b, c** of the circuit shown on Fig.2. They can be calculated using function: [**a,g,b,c**] = realizeNTF(NTF, FORM, STF) from package "delsig" (Schreier, 2009), where [**a,g,b,c**] are vectors of coefficients from Fig. 2. The NTF (noise transfer function) is given by (1); STF (signal transfer function) STF=1 and FORM refers to the form of the architecture and are selected as the feed-forward structure.

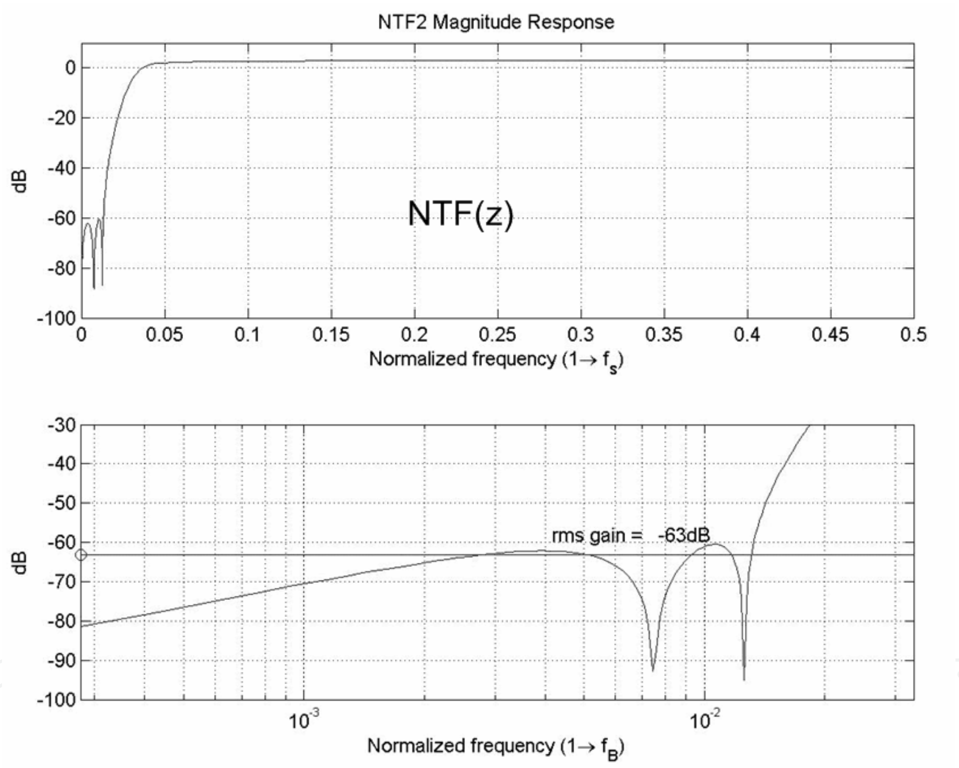


Fig. 1. Noise transfer function plot of (1). Upper plot: |NTF(z)| from 0 to 0.5*fs. Lower plot: |NTF(z)| pass-band detail

Input signal is added to the first integrator through coefficient b_{in1} and to the quantizer input through coefficient b6. The state variables $x_1(z)$ through $x_5(z)$ are the outputs of discrete time integrators. The coefficients g$_1$ and g$_2$ determine the positions of complex conjugate poles, while weighted state variables with weights a_1 through a_5 that are added together, with weighted input signal (b_6), contribute to the signal at the input of one bit internal quantizer. In this way, a complex conjugate zeros of (1) are formed. A linear model is formed if the internal quantizer is replaced by an adder that adds signal $x_6(z)$ with quantization noise $Q(z)$ and use gain $k = 1$. One can determine the poles and the zeroes of noise transfer function $NTF_x(z)$ of the block diagram on Fig. 2. using symbolic and optimization toolbox of the Matlab and some "hand" written state space equations. The result is given in (2).

$$NTF_C(z) = \frac{V(z)}{Q(z)} = \frac{(z-\alpha_0)(z^2-\beta_1 z+\gamma_1)(z^2-\beta_2 z+\gamma_2)}{(z-\delta_0)(z^2-\eta_1 z+\lambda_1)(z^2-\eta_2 z+\lambda_2)} \tag{2}$$

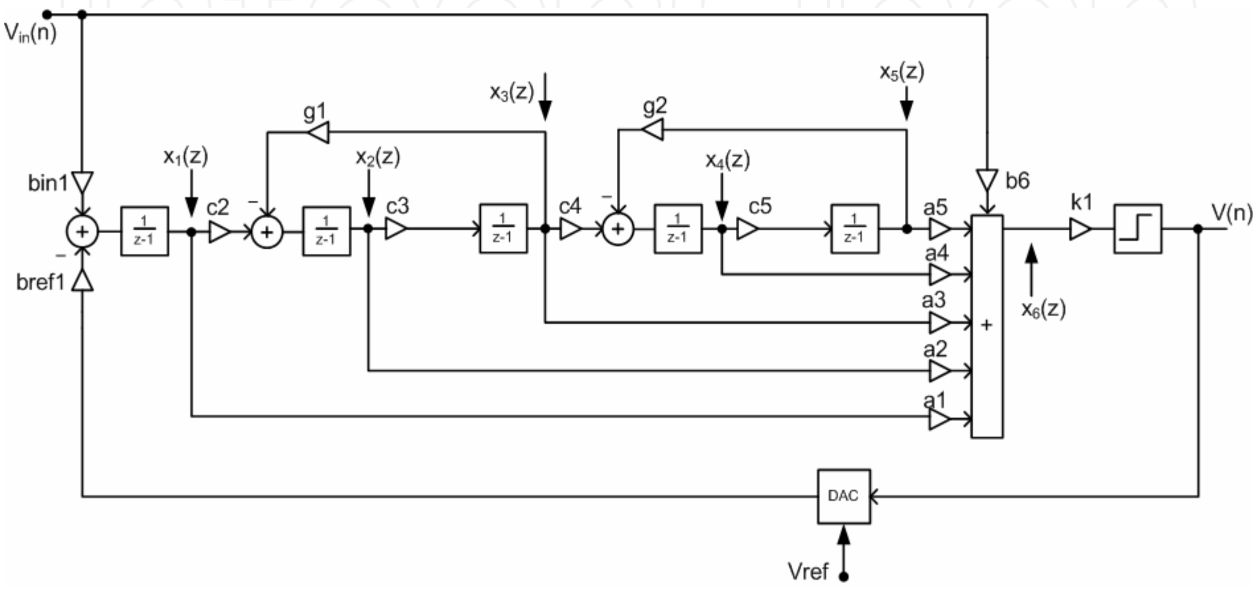


Fig. 2. Block diagram of a fifth order Σ-Δ modulator with feed-forward architecture. State variables $x_i(z)$ are outputs of the integrators

All coefficients are calculated by assuming the linear model of the Σ-Δ modulator. Of course, coefficients obtained in this way do not produce appropriately scaled state variables of the modulators' loop transfer function and further scaling steps will be needed; these are explained in the following subsections. For NTF(z) from (1), STF(z)=1, and for the selected feed-forward architecture the coefficients are (3):

$$\begin{aligned}\mathbf{a}^T &= [1.43\,1.35\;0.98\;0.53\;0.067] \\ \mathbf{g}^T &= [0.0090\;\;0.0633] \\ \mathbf{b}^T &= [0.23\;0\;0\;0\;0\;0.5] \\ \mathbf{c}^T &= [0.23\;0.33\;0.25\;0.18\;0.099]\end{aligned} \tag{3}$$

To check the validity of the synthesis algorithm, we can calculate the spectrum of the linear model of the modulators' bit-stream *V(n)*. The state space description of the linear model of the circuit from Fig. *2* is defined in (4).

$$\begin{aligned}y(n) &= \mathbf{a}^T\mathbf{x}(n) + b_6 v_{in}(n) \\ v(n) &= k_1 y(n) + q(n) \\ x(n) &= \mathbf{A}\mathbf{x}(n) + \mathbf{b}_{in} v_{in}(n) + b_{ref} v(n) v_{ref}\end{aligned} \tag{4}$$

The elements of the (4) are defined in (5) and (6). The meanings of the symbols are the following: $\mathbf{x}(n)$ is a vector of state variables, $v_{in}(n)$ is input voltage, $y(n)$ is the output of the loop-filter, $q(n)$ is quantization noise, and $v(n)$ is a bit-stream output. The spectrum obtained by the FFT function from Matlab is given in Fig. 3.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ c_2 & 1 & -g_1 & 0 & 0 \\ 0 & c_3 & 1 & 0 & 0 \\ 0 & 0 & c_4 & 1 & -g_2 \\ 0 & 0 & 0 & c_5 & 1 \end{bmatrix} \tag{5}$$

$$\mathbf{b}_{in}^T = \mathbf{b}(1:5); \quad b_6 = \mathbf{b}(6); \quad b_{ref} = \mathbf{c}(1) \tag{6}$$

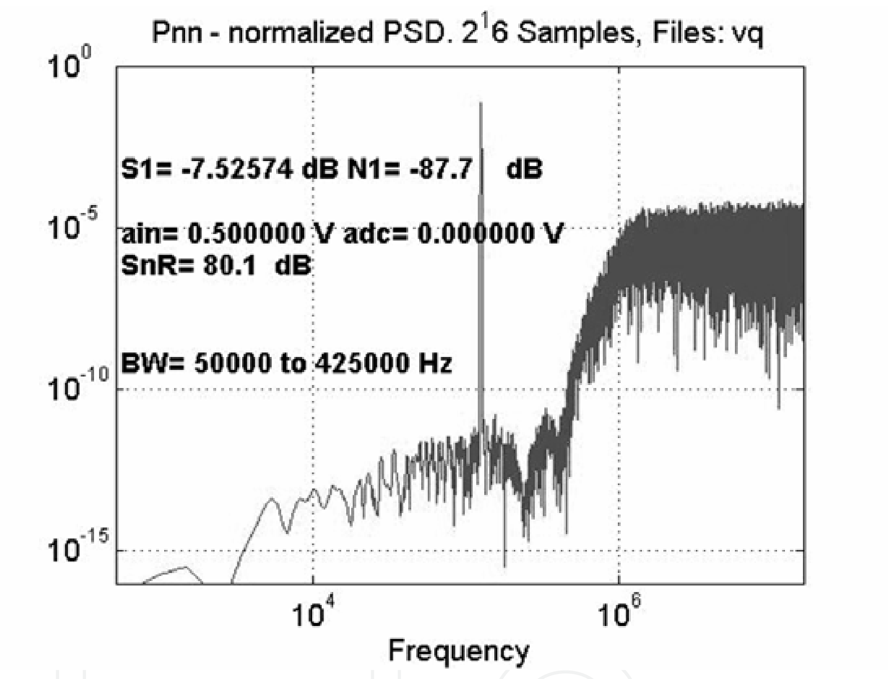


Fig. 3. Power spectral density of the bit-stream of ideal linear model of Sigma-Delta modulator. The output signal v(n) is a consequence of input signal v_{in} (a sine-wave) with frequency 125kHz, amplitude $0.5V_{rms}$ and sampling frequency f_s=32 MHz

Further checks require a nonlinear model of a one bit quantizer. For the first tests, a "signum" function instead of linear model of a quantizer can be used. In that case, the description is similar to (4) except that v*(n)* is modelled by a *sign()* function. The Matlab simulation result using nonlinear state-space description (7), with sine-wave input at a frequency of 125 kHz and an amplitude of 0.5 V_{rms} , is presented in Fig. 4.

$$\begin{aligned} y(n) &= \mathbf{a}^T \mathbf{x}(n) + b_6 v_{in}(n) \\ v(n) &= sign\left[k_1 y(n)\right] \\ x(n) &= \mathbf{A}\mathbf{x}(n) + \mathbf{b}_{in} v_{in}(n) + b_{ref} v(n) v_{ref} \end{aligned} \tag{7}$$

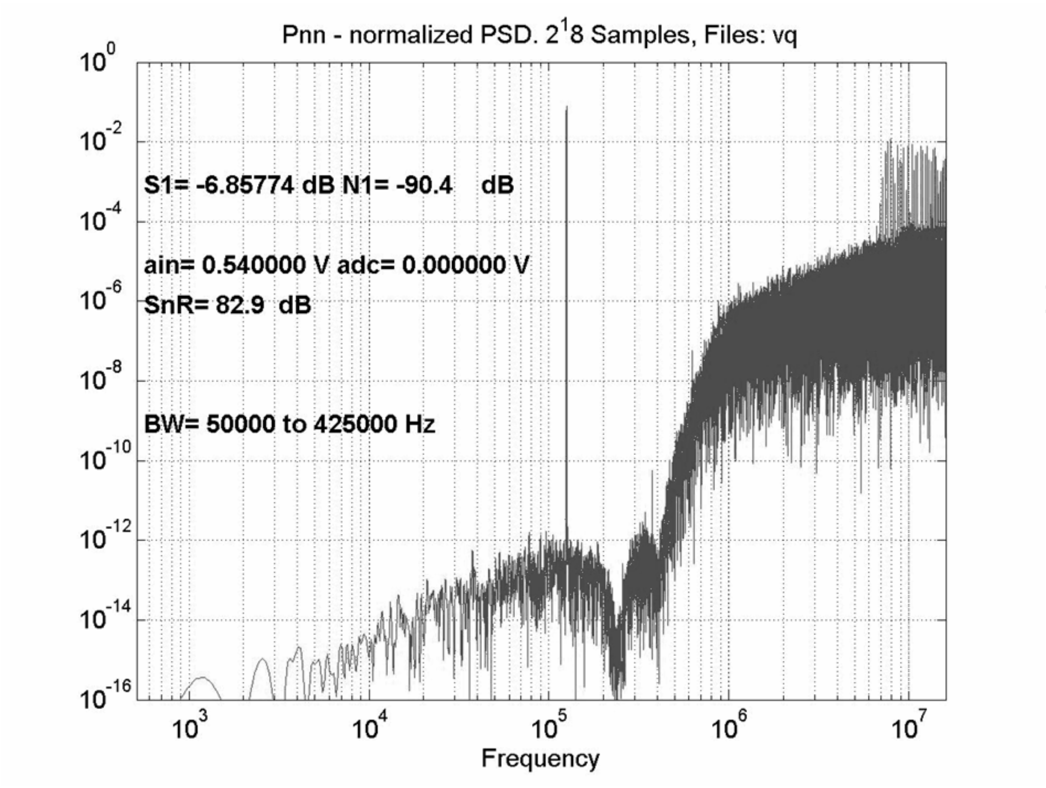


Fig. 4. Power spectral density of the bit-stream of a nonlinear model (7) at the output of a Sigma-Delta modulator v(n) with input signal v_{in} that is a sine-wave with a frequency 125kHz, an amplitude of $0.5V_{rms}$; sampling frequency is f_s=32 MHz

Simulation results in Fig. 3 and Fig. 4 show almost identical spectrums. The main spectral component at 125 kHz is due to the input signal and the noise spectrum is a consequence of noise shaping of quantization noise of 1 bit internal quantizer. The biggest difference between linear and nonlinear model appears at high frequency; the quantization noise spectrum of the nonlinear model is shaped differently for frequencies above 1MHz. In addition, for the nonlinear model, the “tones” can be observed close to fovs/2 as sharp peaks of the spectrum. “Tones” can be removed by adding the appropriate dither signal to the quantizer input. The procedure for detecting tones and a methodology for their removal is presented in subsection 4.4.

4.3 Simulink model

The Simulink model of ideal Σ-Δ modulator is equivalent to the block diagram presented in Fig. 2. All coefficients are available in the work-space and are set in a Matlab m-file according to the values given in (3). The Simulink symbols used and the settings of some of their parameters are the following:

- **Gain:** All coefficients are implemented with the symbol, “Gain” (Fig. 5). The main parameter is the gain factor that multiplies the input signal to get the output signal. The coefficients are defined in the m-file. After running the m-file, the coefficients are available in the Matlab workspace. In pane “Main”, the gain factor and the “Sample time” are defined. In the “Signal attributes” pane the min-max values for the output and the data type of the output are set. They are set to their default values. In pane “Parameter Attributes”, the parameter attributes are set as follows: min-max values and output data types are set to "Inherit: Same as input", so their data types are double. The rounding in this case is of no importance since the variables and constants are represented in a double floating-point format.

- **Zero-Order-Hold:** Each input connection of the modulator is assumed to have a continuous time signal; therefore, it is sampled and held by using the Zero-Order-Hold symbol, which closely follows the behaviour of the S-C circuits. Signals like In1 and Vref from the modulators block diagram are equipped with the Zero-Order-Hold element (Fig. 6). Sampling period of the block is set to T_s.

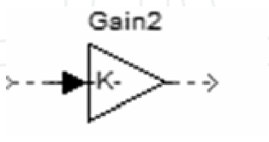


Fig. 5. Gain symbol

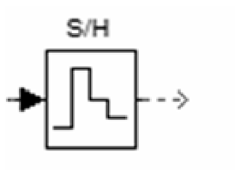


Fig. 6. Zero-Order-Hold symbol

- **Sum:** The block "Sum" (Fig. 7) adds input signals to get the output. The number of input signals is dependent on the number of + or – symbols that appear in the "list of signs". In our example, several differently shaped "sum" symbols are used: ++ , +− , and +++++++. Number of + or − symbols signify how many and what kind of inputs are used (these are present in the "List of signs"). In addition, the shape of the symbol can be selected (round or rectangle). For example, for feed-forward architecture, the sum of weighted state variables appears at the input of the quantizer. In the same pane, the sampling time is set as "-1", which means that it is "Inherited". In pane "Signal attributes", the "Accumulator data type" is selected as either "Inherit" or "double"; the output min-max values are not limited and the output data type is selected as "Inherit". Since input signal data type is double, all other signals are also double.

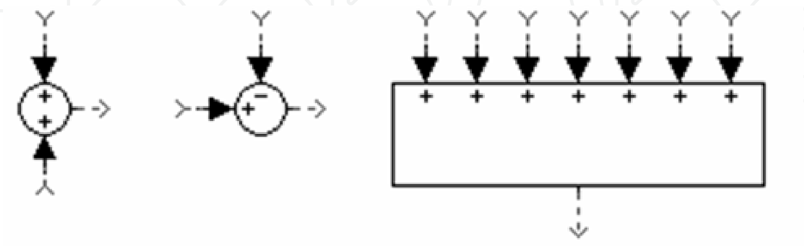


Fig. 7. The Sum symbols

- **Product:** The block which multiplies v(n) (bit-stream BS) with reference voltage is a "Product" shown in Fig. 8. It is possible to define the "Number of inputs" and the "type of multiplication". In our case two inputs need simple element-wise multiplication. In addition, Sample time is set to "-1=Inherited". In the pane "Signal Attributes", the output minimum and output maximum can be defined. The default value is "unlimited", while the output data type is set to "Inherited", and therefore the data type is double.

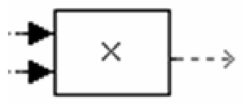


Fig. 8. The Product symbol

- **Relay:** The comparator can be in the simplest way modelled as a relay. The parameters that must be set are the following: Switch on point (VH), Switch off point (VL), Output when on (1), Output when off (-1), Enabled zero crossing, and sample time (-1=Inherited). The switching levels are defined in the main Matlab m-file. In our case, a small hysteresis is built-in and taken into consideration as follows: VH=1mV and VL=-1mV. The output levels (BS) are set to ±1.

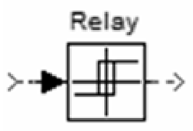


Fig. 9. The Relay symbol

- **DT integrator:** The last element used in the simplified model of our example modulator is a discrete time integrator shown in Fig. 10. In the ideal and simplest case, it can be modelled by a discrete transfer function with pole at z=1, followed by a Zero-Order-Hold block. The transfer function coefficients are defined in the "Main" pane of the parameter settings, with Numerator coefficients set to [1] and Denominator coefficients set to [1 -1]. In addition, the Sample time is set to T_s, and is defined in the main Matlab routine before running the Simulink simulation. The State attributes in this case are unimportant and default settings can be accepted. All five integrators are equivalent for the ideal Simulink model.

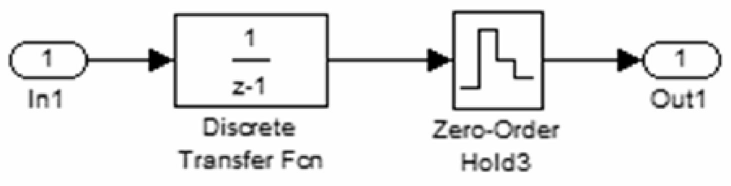


Fig. 10. Discrete time integrator model used in example modulator

- **Top level**: To run the simulation and to store some results, a top-level scheme is needed that consists of system to be simulated (Modulator_5th_order on Fig. 11), signal generators, and some elements that help store the selected results to the workspace or to the disk. The scope (osciloskope) is useful for observing some internal signals in time domain.The top-level Simulink scheme for our example modulator is presented in Fig. 11. It consists of a sine-wave generator, a constant for the reference input, the circuit to be modelled and simulated (in our case, a fifth order modulator), the scope, and two sinks with names bs_mod5.mat and Comp_in.mat that store the results to the disk.

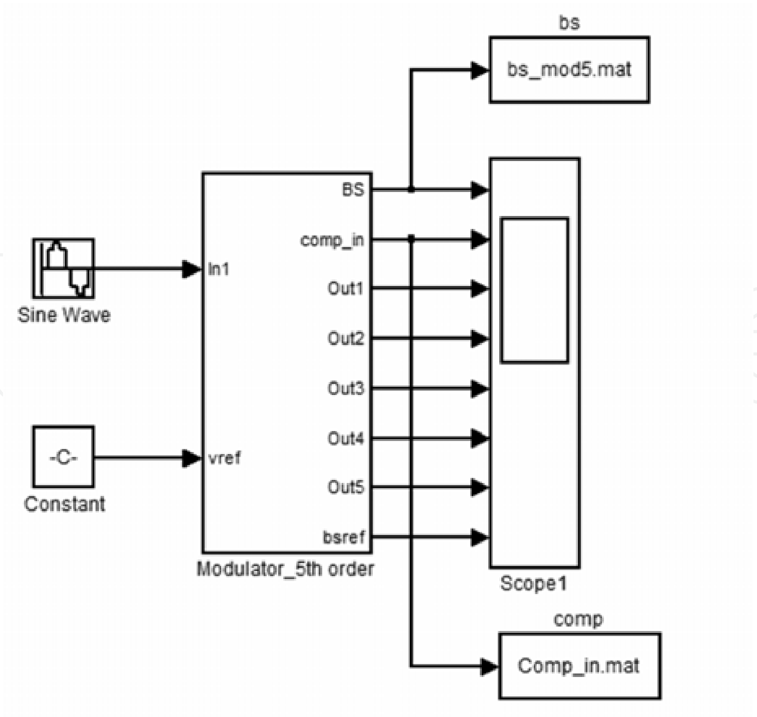


Fig. 11. Top level Simulink symbol with generators, constants, and sinks

- **Sine-Wave**: The definition of sine-wave signals needs many parameters with the following settings: "Sine Type" is Time based, "Amplitude = ain_real", "Bias = adc", "Frequency = 2*pi*fin", "Phase = 0", and "Sample Time= Ts". Variables ain_real, adc, fin and Ts are defined in the main Matlab routine and are available in the Matlab workspace.
- **Constant:** This is used to define the reference voltage of the modulator. The parameters are "Constant value = Vref_real" and "Sample Time = Ts"; both are defined in a Matlab workspace.
- **To file:** The data or variables are written to the disc. Two such blocks are available in Fig. 11: bs and comp. The data are stored to the files bs_mod5.mat and Comp_in.mat. The Parameters are the following: "Filename" that is set to, for example, bs_mod5.mat; "Variable name" is set to bs; "Decimation" is set to 1 (all samples are stored); and "Sample time" is set to Ts that is defined in the Matlab work-space. The stored data can be used for further analysis and plots.
- **Scope:** It is possible to observe the time-domain responses of the different connected signals. Fig. 12 shows time-domain signals of simulated modulator for: BS, Comp_in, and Out1 to Out5 that are state variables x(1) to x(5) and bsref that is a feedback signal BS multiplied by Vref.

The time-domain simulation results that use the ideal Simulink model, which is composed of the symbols defined above are presented in Fig. 12. The simulation parameters are specified in the Fig. 12 caption text. The signals observed are marked on the left hand side of the picture; BS stands for bit-stream, Comp_in for comparator input signal, X1 through X5 for state variables (outputs of the integrators), and BS_ref for the BS multiplied by the reference signal. The spectrum of a bit-stream signal for the same simulation is presented in Fig. 13; it is almost equal to the spectrum obtained by simulation of an ideal nonlinear Matlab model defined with (7) and presented on Fig. 4.

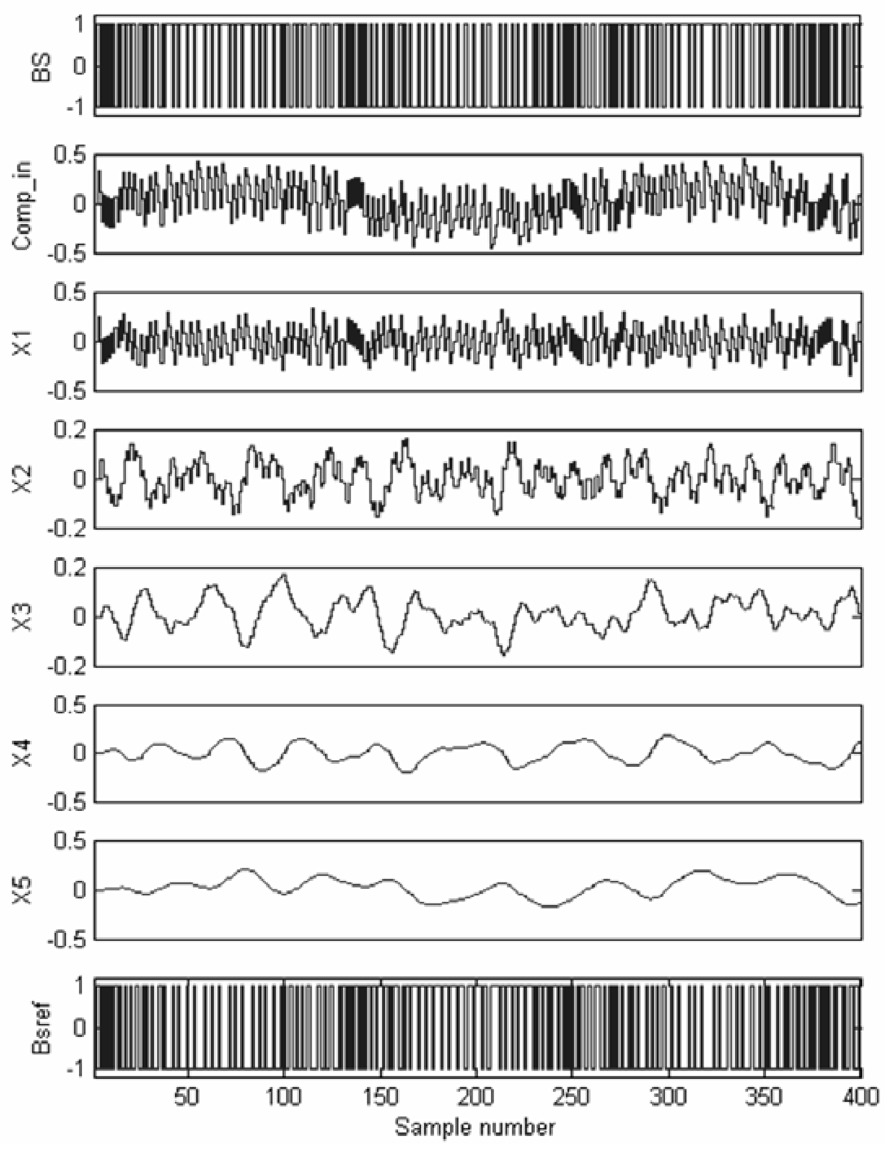


Fig. 12. Simulation results for 5th order modulator with vref_real=1, ain_real=0.54, fin=125kHz, fs=32MHz

4.4 Limit cycles analysis

Σ-Δ modulators are nonlinear devices/systems and therefore some characteristic problems related to such systems can happen. An important and difficult problem is related to "Limit-Cycles" (Mann, 1999) and (Strle, 2006). It is possible that under certain conditions, the spectral components appear in the base-band ("tones") that are not present in the input signal. This problem is severe in low order modulators but also not negligible in the high order modulators, where it is not so pronounced compared to low order modulators because of higher order noise shaping. Because of the non-linearity of the loop, it is very difficult to analyze the problem theoretically and no analytic result exists for high order modulators. A possible way to analyze the problem is to use a High level Simulink model and to perform many systematically prepared simulations at different conditions to locate eventual limit-cycles. The Matlab/Simulink environment is efficient because simulations run on a high hierarchical level. The results of a set of such simulations are presented in Fig.

14 as a 3D plot of spectrums, where the x-axis represents the frequency, y-axis represents DC input voltage, and the z-axis represents the PSD (power spectrum density) of a bit-stream. For our experiment a small AC signal with an amplitude of V_{inac}=68 uV and a frequency of 250 kHz is connected to the input of the modulator at different DC input voltages (V_{inDC}=-5mV to +5mV). On the left part of Fig. 14, the peaks ("tones") can be observed at a certain DC input voltages even in the base-band. A spectrum is presented for the original fifth order modulator from Fig. 2.

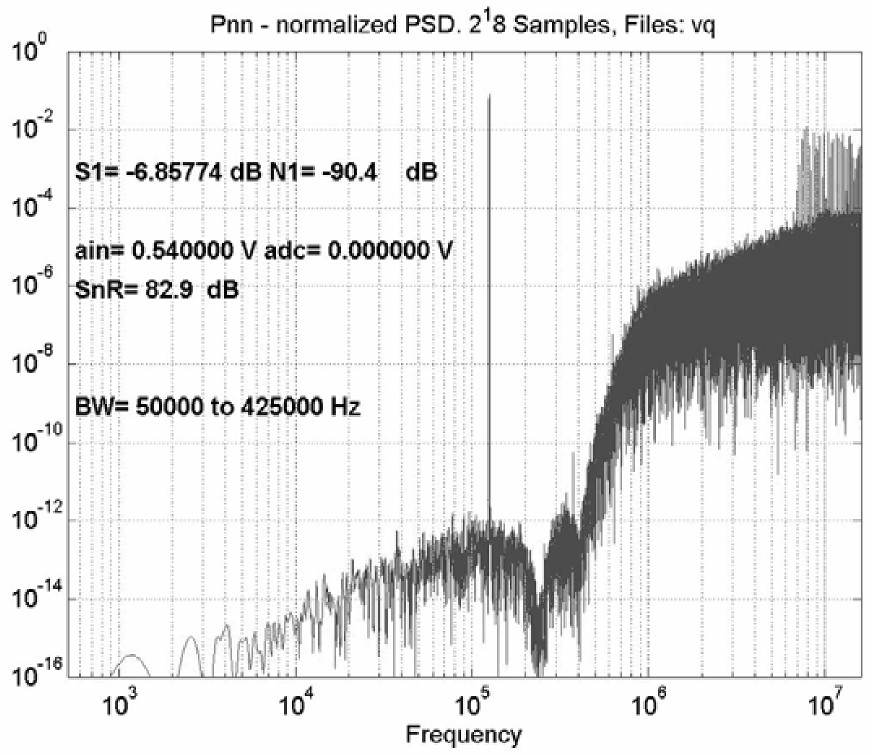


Fig. 13. Spectrum of a BS (bit-stream) for ideal Simulink model of a fifth order modulator

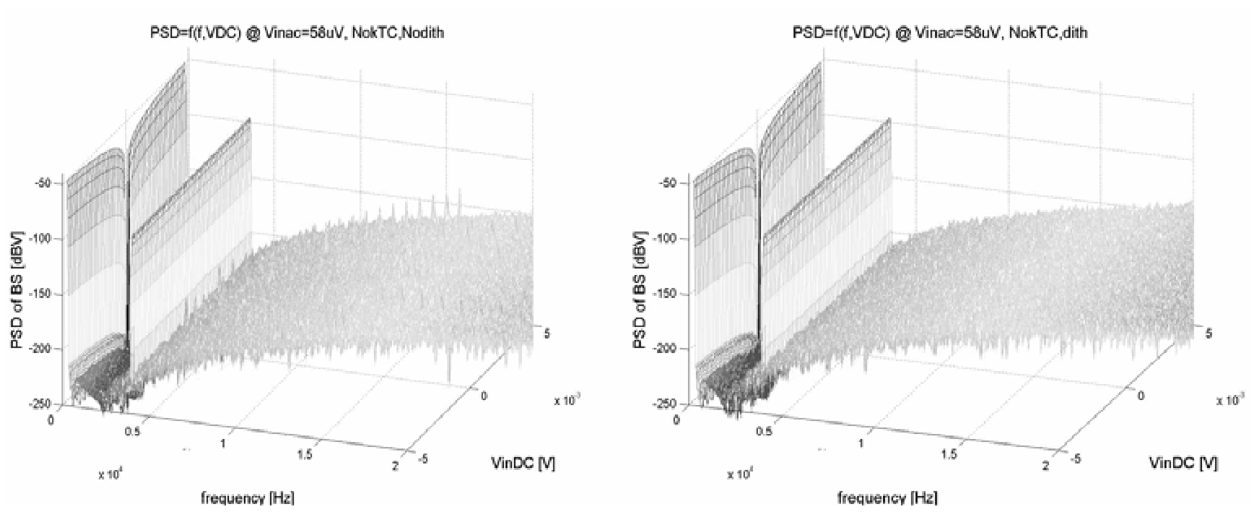


Fig. 14. Limit cycles of a modulator: left simulation without dither and right simulation with pseudo-random dither signal added to the input of the quantizer

The right side of Fig. 14 shows the spectrum after adding a dither signal to the quantizer input. A dither signal is a one bit pseudo-random signal connected to the input of the internal quantizer, according to the Simulink scheme on Fig. 15. The weight of the dither signal is 0.3*Vref. A pseudo-random-noise signal (PRS) (Zepernick, 2005) that is used as a "dither" is for the simulation purposes generated off-line and stored into the file. A dither signal de-correlates "tones" and spreads their energy over the entire band, which is then shaped by the noise transfer function of the modulator. A 3D plot of a bit-stream's PSD after adding the dither signal is presented on the right side of Fig. 14. The tones have disappeared but there is some small noise penalty in the base-band; the SnR is reduced by approximately 1.5 dB.

4.5 Scaling

The initial implementation of a discrete-time loop filter with feed-forward structure does not take into consideration the real voltage levels of state-variables at maximum input voltage signal, as well as the real reference voltage. At the beginning of the design process, the state variables are almost arbitrary; their relationship is such that poles and zeroes are implemented, while the state variable levels are not defined. The scaling is a semi-automatic procedure where maximum levels of the state-variables are adjusted to the required values. Scaling is needed to optimize signal levels in comparison to the noise levels and to adjust maximum voltage levels of the integrator outputs. In this way the performance parameters like SnR (signal to noise ratio), HD (harmonic distortions), SR (slew-rate) requirements are optimized. In the first step the real reference voltage (vref_real=0.6V for our example modulator) and the maximum input signal voltage (vin_real=0.54V) are connected to the modulator. A long simulation is then performed to get the distribution of state-variables. Long simulations are needed because Σ-Δ modulator is a chaotic device and it is difficult to predict voltage levels of the state variables in a short simulation; the spectrum may contain some very low frequency components. The min/max values are obtained from the simulation results in the second step (see $|x_{i\_\max}|$ on the left side of Fig. 16). In the third step, the desired limit $L(i)$ of each state variable is compared to the $|x_{i\_\max}|$ and the vector of scaling coefficients $\mathbf{s}$ is calculated using (8).

$$s(i) = \frac{L(i)}{|xi\_\max|} \quad i = 1,2,.... \tag{8}$$

In the fourth step, the level of each integrator output is adjusted by multiplying all input coefficients that are connected to the integrator. For example, coefficients b_{in} and b_{ref} are multiplied by $s(1)$ to adjust the maximum level of $x(1)$; at the same time, the coefficient c_2 is divided by $s(1)$ to keep the positions of the poles unchanged. This scaling procedure is repeated for all state variables and for all coefficients using Matlab. After scaling, maximum state-variable voltage levels are adjusted to the desired level, as shown on the right side of Fig. 16. In a real circuit, the required voltage range of state variables is dependent on the architecture, technology, and design of the modules. Histogram plots for all state variables before and after the scaling procedure are, for our example modulator, shown in Fig. 16.
The maximum levels of the state variables for our example modulator are optimized to the following levels: $|x(1)|_{\max} = 0.4\,V$ down to $|x(5)|_{\max} = 0.2\,V$. All other state variables are scaled in such a way that the levels are linearly distributed between 0.4 V to 0.2 V. Such

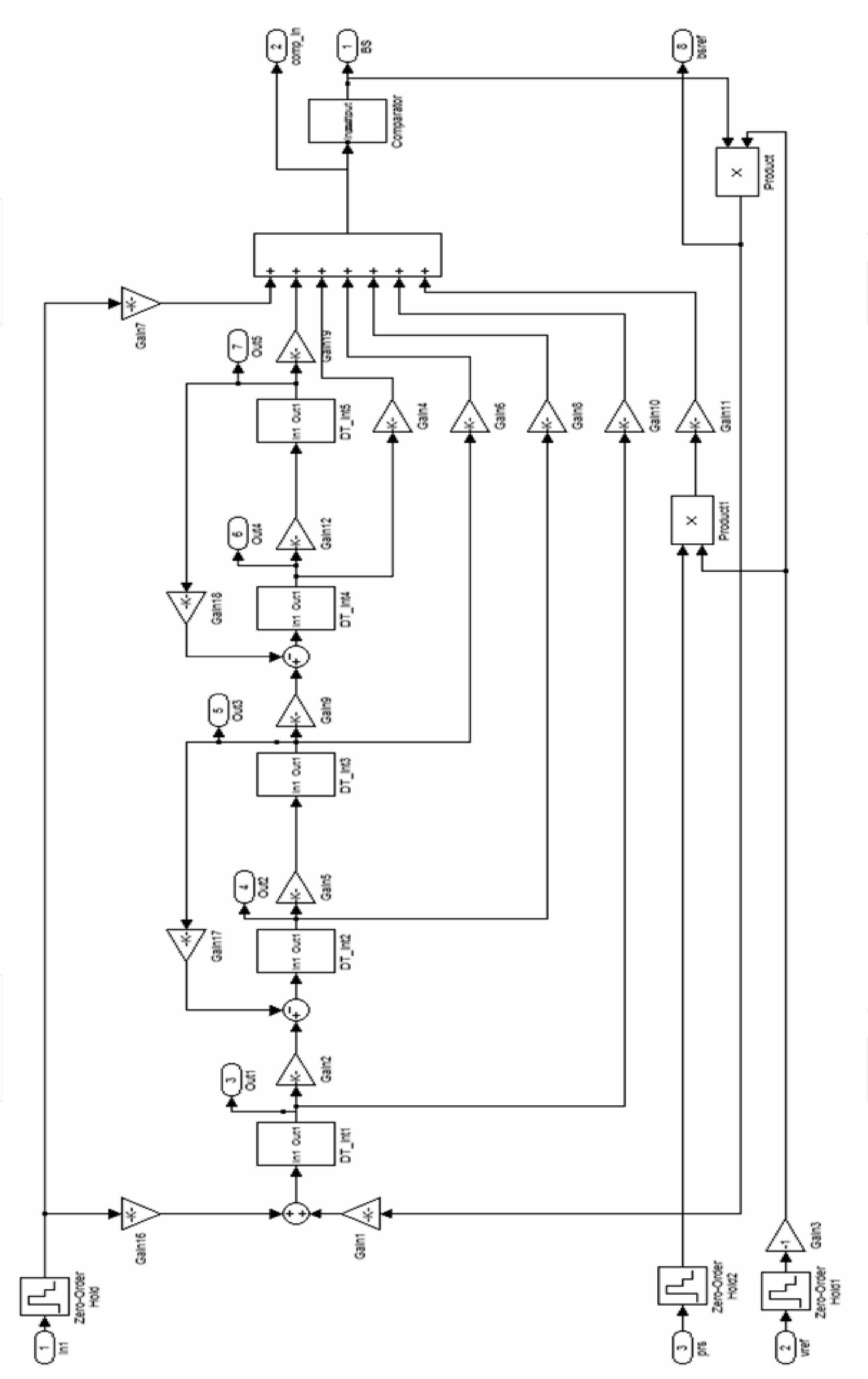


Fig. 15. Fifth order modulator model with dither added to the quantizer input signal

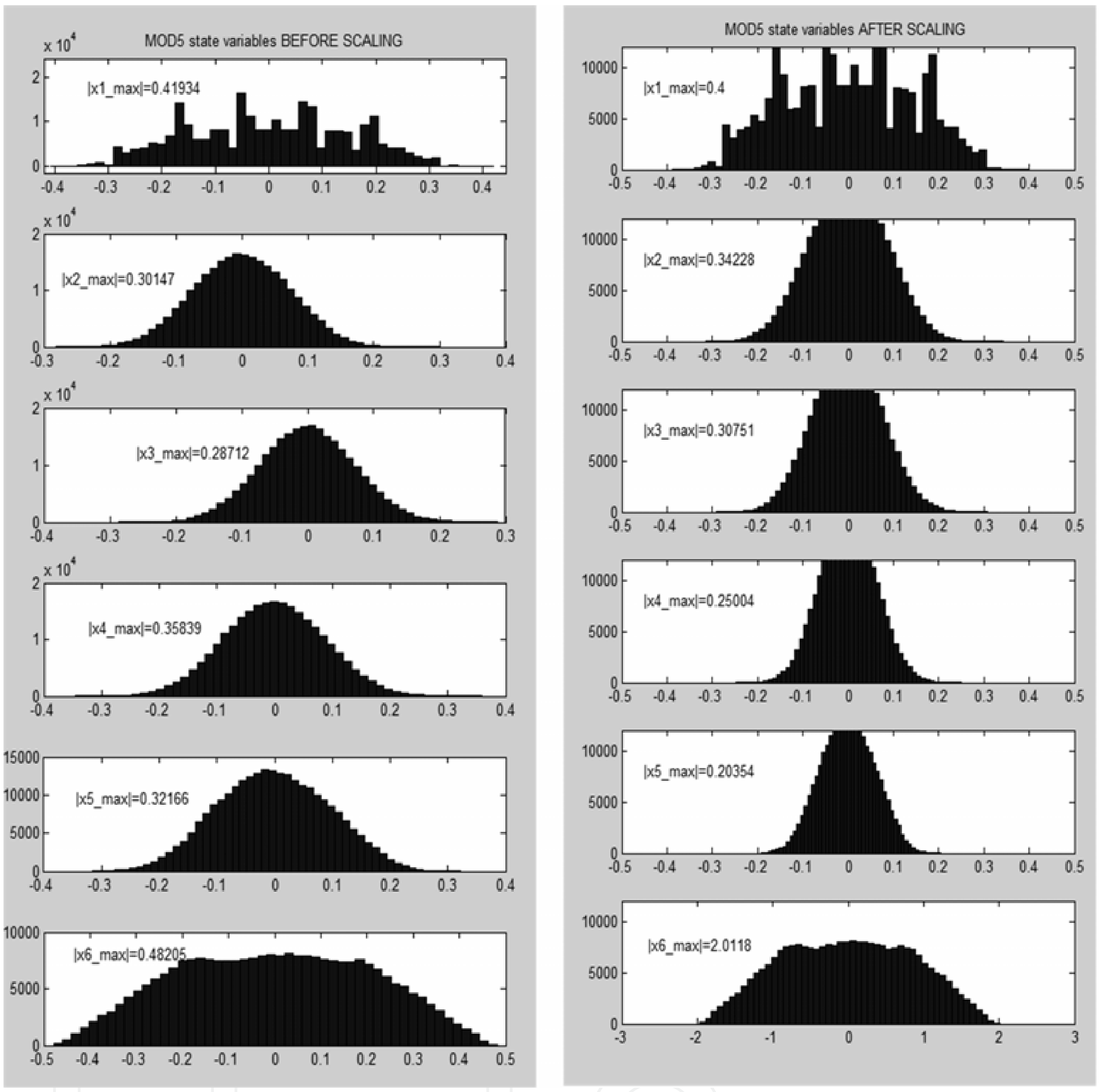


Fig. 16. Histograms of state variables for Vin=0.54 V and Vref=0.6V. Left figure: before scaling. Right histogram: after scaling

limits are selected because of the following stability considerations. With a large input voltage, the modulator could become unstable because of the high order loop filter and because the feedback cannot follow the input voltage. Therefore, for input voltage larger than the selected limit, the first state variable enters the saturation of the first integrator and reduces the loop filter order, and hence, the conditions for stability are improved. For slightly higher input voltage, the second integrating stage enters the saturation, and so on. Maximum voltage at the quantizer input ($\left|x_{6\_\max}\right|$) is not important for the stability reasons but for the proper operation of the quantizer; a big input voltage is allowed (2 V in our example modulator) since we know that a passive adder before the quantizer will contain some parasitic capacitances that will reduce the level. The only requirement for the input signal of a one bit quantizer is that it is big enough to reliably drive the internal quantizer.

5. High level modelling of the mixed-signal circuits

Up to now, a discrete time loop-filter was implemented using ideal discrete time integrator transfer functions (see Fig. 2). The model is good enough if we are not interested in real implementation and circuit parameter influences. In real designs, it is beneficial if we can predict the influence of circuit parameters at a high hierarchical level; it is even better if we can determine the required circuit parameters from a system level model and simulation results.
The first step in including circuit non-ideal effects is to select the possible implementation of the discrete time integrators. In our case, we selected a Switched-Capacitor (S-C) implementation of the loop transfer function, which is composed of switched capacitor integrators, and therefore, it is necessary to develop the models of their behaviour. The procedure is defined in the following subsections.

5.1 Modelling Switched-Capacitor (S-C) circuits

Switched capacitor circuits are used in analogue and mixed signal circuits for a very long time. One of the first books that treated the subject systematically was "*Analog MOS Integrated Circuits for signal processing*" written by R. Gregorian and G.C. Temes (Gregorian, 1986); many other books about the subject were published until today. The most important block used extensively in most of the switched-capacitor circuits is the parasitic-capacitance insensitive discrete-time S-C integrator with a simplified electrical scheme presented in Fig. 17. The circuit is composed of switches, capacitors, and operational amplifier. The difference between left and right S-C integrator is the connection of phase *Φ1* and *Φ2* and how they drive the switches. For the circuit to function properly, the phases should be non-overlapping signals.
Assuming that input and output signals are discrete-time signals that change their states at the end of signal *Φ2*, we can describe the operation of the S-C integrator on the left (non-inverting integrator) by writing the difference equation that relate the output signal of the integrator and the input signal at discrete times according to (9). The C_1 and C_{int} are input and integrating capacitors respectively, while $V_1(n)$ and $V_{out}(n)$ are input and output voltages at time slot n, (see left part of Fig. 17).

$$-C_{\text{int}}\left[V_{out}(n)-V_{out}(n-1)\right]=-C_1V_1(n-1) \qquad (9)$$

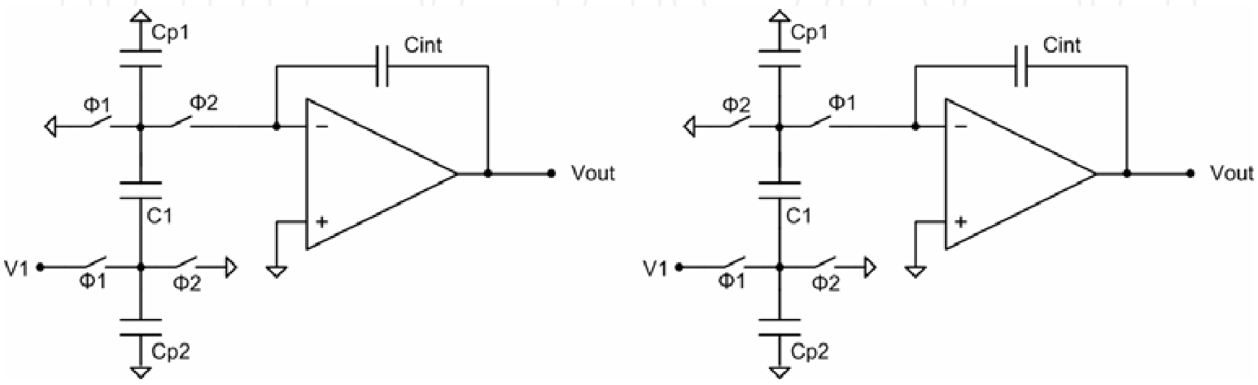


Fig. 17. Simplified circuit diagrams of S-C integrators. Left: non-inverting S-C integrator, Right: inverting S-C integrator

The output voltage for this simple integrator is, in time domain equal to (10).

$$V_{out}(n) = V_{out}(n-1) + \frac{C_1}{C_{\text{int}}} V_1(n-1) \tag{10}$$

Capacitor ratio determines the constant of the integrator. In z domain, the description becomes (11), so the transfer function of ideal **non-inverting S-C integrator** is (12).

$$V_{out}(z) = z^{-1} V_{out}(z) + \frac{C_1}{C_{\text{int}}} z^{-1} V_1(z) \tag{11}$$

$$H(z) = \frac{V_{out}(z)}{V_1(z)} = \frac{C_1}{C_{\text{int}}} \frac{z^{-1}}{1-z^{-1}} = \frac{C_1}{C_{\text{int}}} \frac{1}{z-1} \tag{12}$$

If switches are driven according to the right part of Fig. 17, the ideal z-domain transfer function is (13). Therefore, it implements an **inverting S-C integrator**.

$$H(z) = \frac{V_{out}(z)}{V_1(z)} = -\frac{C_1}{C_{\text{int}}} \frac{1}{1-z^{-1}} = -\frac{C_1}{C_{\text{int}}} \frac{z}{z-1} \tag{13}$$

Both integrators can be modelled by discrete time models where all voltages are described by double floating point numbers to be able to represent the continuum of analogue variables available in any analogue or mixed signal circuits. The models of discrete-time integrators are presented in Fig. 18. The left side shows the non-inverting S-C integrator and the right side shows inverting S-C integrator. Usually, this is the level of abstraction that is used in modelling mixed signal circuits. However, real circuit influences must be taken into consideration if one wants to see the contributions of these influences. The discrete time transfer function of the integrator $1/(z-1)$ is implemented with non-inverting S-C integrator, which is modelled on the left part of Fig. 18. A complete DT integrator that is used in the simplified model of our example modulator is presented in Fig. 19.

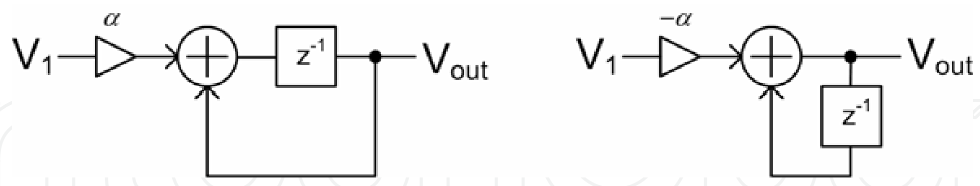


Fig. 18. Simulink model of ideal discrete-time (DT) integrator. Left: non-inverting; Right: inverting

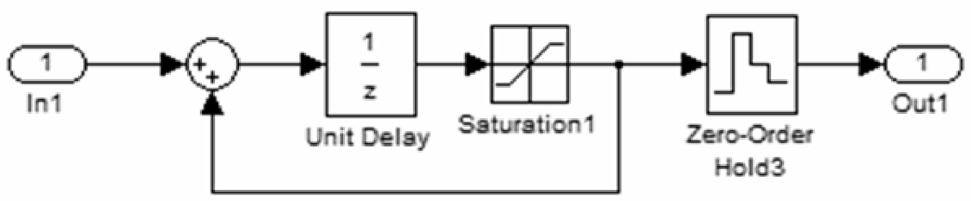


Fig. 19. DT non-inverting integrator Simulink model including Saturation and Zero-Order-Hold blocks

The architecture of the example modulator remains as it was defined in Fig. 15, except that all the blocks that implement the DT integrators are replaced with the model from Fig. 19; coefficients remain exactly the same and are shown in Fig. 15.
Only non-inverting integrators are used in our example circuit because of power consumption reasons. Two Simulink blocks in Fig. 19 were not described before: "Unit Delay" and "Saturation". "Unit delay" needs the settings of two Block Parameters: "Initial conditions" that are set to 0 and "Sampling time" that is set to Ts, as defined in a Matlab work-space. The block "saturation" is defined in the subsection 5.3.

5.2 Capacitor ratios

The coefficients are implemented with capacitor ratios as defined in (12). Each capacitor ratio defines one coefficient. For example, coefficient $b_{in}(1) = c_u(1)/C_I(1)$ is the ratio of input switched capacitor and the integrating capacitor of the first integrator. Assuming at the beginning that $b_{in}(1) \le b_{ref}(1)$ (see Fig. 2), then the input switched capacitor is selected as a unit $c_u(1) = 1$, while all other capacitors connected to the first integrator are bigger than $c_u(1)$. It is easy then to calculate the relative capacitance of, for example, $C_I(1)$. In this way, all coefficients can be replaced by corresponding capacitor ratios. For each integrator, the smallest coefficient determines the unit capacitor and all other capacitors in the same integrator stage are just multiples of the unit corresponding to that. How big the unit capacitance of each integrator stage must be is dependent on the required noise level allowed for each integrator stage i. and the architecture of the circuit. To reflect that, all coefficients implemented by the "Gain" blocks in Fig. 15 are changed; they now contain appropriate capacitor ratios expressed as the explicit quotient of two capacitors, where capacitance in the denominator is always corresponding to the integrating capacitor.

5.3 Adding circuit design parameters to the model

In a real S-C integrator, the charges are transferred from input capacitors to the integrating capacitors ideally without loss of charge, while the time of charge transfer is negligible. We can model that by difference equations (9) and the DT model in Fig. 18. In reality, this transfer is not complete and several additional non-ideal effects contribute to the results. It is beneficial if one is able to predict the influence of such effects on a high hierarchical level before the start of the circuit design. To be able to take into consideration the important circuit parameters of a mixed-signal circuit, the model of the S-C integrator must be improved by adding circuit design parameters such as: nonlinearity of the opamp characteristics, kT/C noise, thermal and 1/f noise, open loop gain of the amplifier (A0), offset voltage (V_{off}), unity gain bandwidth (GB), slew rate (SR), and others. In our improved model of the DT integrator, some discrete time processes will be influenced by the continuous time processes inside the sampling period. In addition, random signals will be added to model the kT/C noise and other circuit noise sources. The model parameters that give acceptable simulation results are the result of optimisation of power consumption, unit capacitor size, the architecture of the circuit and optimum selection of other parameters. Optimized parameters define the limits for the circuit design. The possible non-ideal effects that we modelled in our example modulator are the following:

- **Nonlinearities of the DT integrators** are modelled using the block "Saturation" on Fig. 19. Parameters that must be set in the block are the following: "Upper Limit" and "Lower limit"; they are set to +LIMIT and –LIMIT respectively and are defined in the

Matlab workspace. The element is treated as a gain when voltage is increasing and the "zero-crossing" detection is enabled. Sampling rate is "inherited", therefore Ts. The value of the LIMIT is obtained from the knowledge of the circuit behaviour of the opamp used in the S-C integrator. If the modulator is appropriately scaled as is presented on the right side of Fig. 16, we can be sure that under all normal conditions, the state variables will always remain within the linear region of the integrator. For our example design, the first integrator saturation level and limit is: $LIMIT(1) = |L(1)| = 0.4V$. Other limits can be obtained from the same figure.

- **kT/C noise:** Each switched-capacitor, in addition to the charge transfer, produces noise as a consequence of a thermal noise generated due to finite ON resistance of the switch (Gregorian & Temes, 1986). The noise power of the switched capacitor is independent of the switch ON resistance because the switch and the capacitor form a low-pass filter; with the increase of resistance comes the increase in noise power density and the decrease in bandwidth. Therefore, a smaller part of the noise gets under-sampled. Consequently, the noise power becomes independent of the resistance but inversely proportional to the capacitance. The noise-power of a switched-capacitor is distributed in the band from 0 to fs/2, which can be calculated according to (14), where $k = 1.38 \cdot 10^{-23} \left[J \cdot K^{-1} \right]$, T is absolute temperature in $°K$, c_i is relative capacitance of the switched capacitor, and $c_{unit}(i)$ is the absolute capacitance of the unit capacitor that corresponds to the integrator stage.

$$P_{n,i} = \frac{k\,T}{c_i \cdot c_{unit}(i)} [W] \tag{14}$$

Each coefficient implemented by the switched-capacitor generates a noise that is modelled in Simulink as a noise source by using a block "Random source" (see Fig. 20 for the 2nd integrator). The parameters are set as follows: "Source type" to Gaussian, "Mean" to 0, "Variance" to noise power according to (14), "Sample mode" to discrete, "Sample time" to Ts, "Samples per frame" to 1, "Output data type" to Double, and "Complexity" to Real.

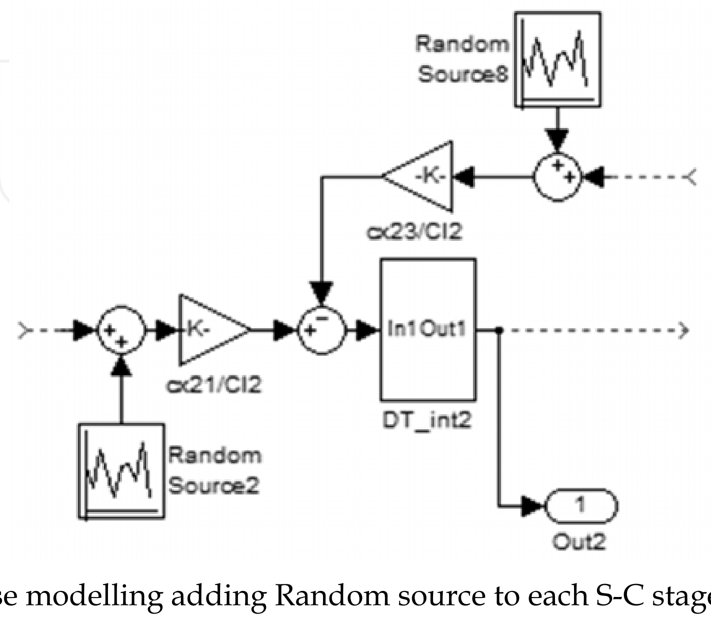


Fig. 20. kT/C noise modelling adding Random source to each S-C stage

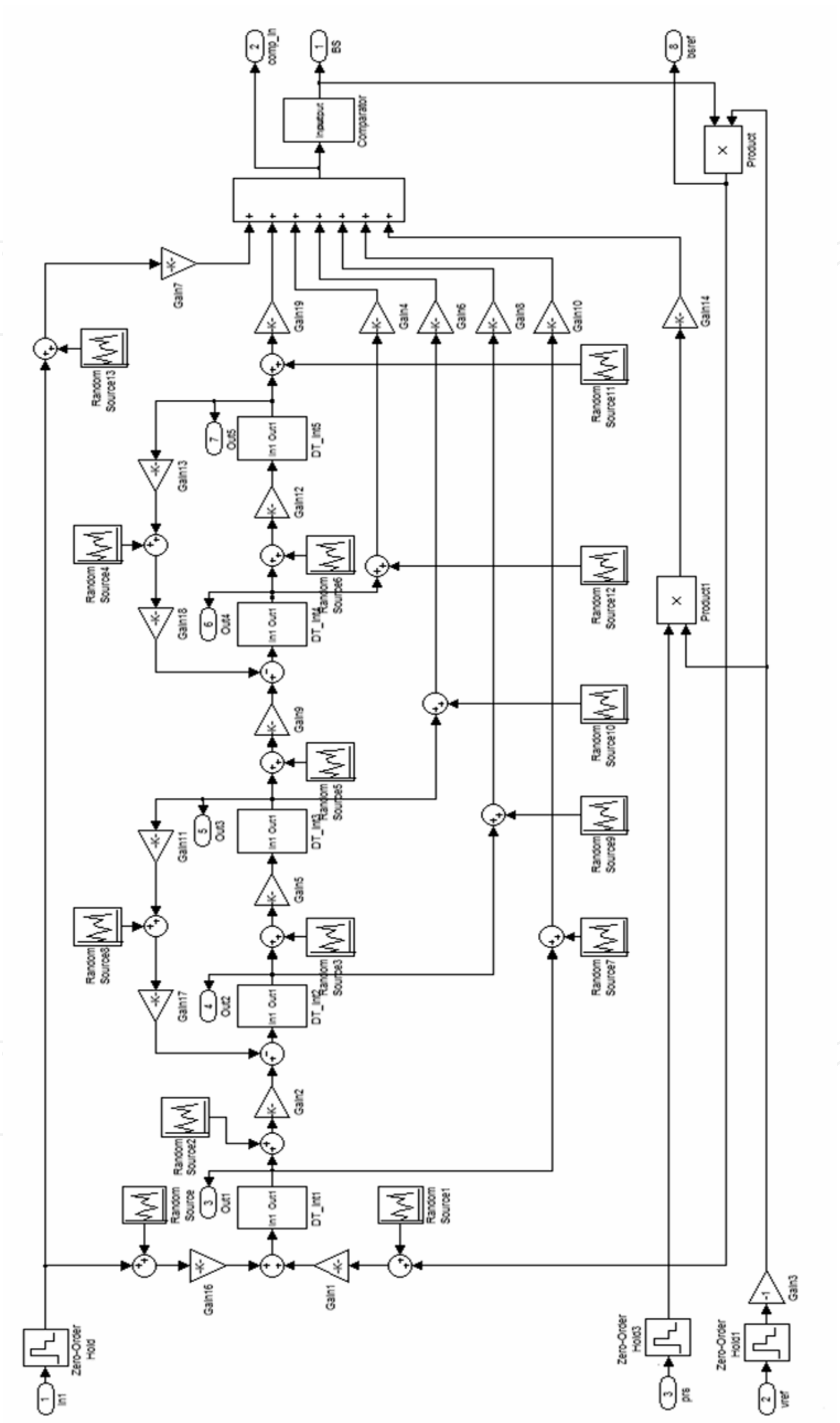


Fig. 21. Simulink model of the fifth order modulator, with kT/C noise sources included

Our fifth order modulator schematics is improved (Fig. 21) by adding appropriate noise sources, with noise power inversely proportional to the absolute value of the corresponding switched capacitance. The contribution of each noise source to the final noise level at the BS output is shaped by a particular noise transfer function. The biggest contributions (not attenuated) come from the input and reference switched capacitors. For high order circuits, the contribution of each k/TC noise source is hard to predict and optimize; hence, the Simulink model can be used to optimize all unit capacitor sizes, which may be different for each integrator in the loop. The power-consumption of each integrator is proportional to the absolute capacitance on one side, while kT/C noise of each S-C stage is inversely proportional to its capacitance, so the smallest power consumption could be achieved by optimizing the absolute value of the unit capacitor of each integrator stage.

- **Open loop gain A0:** Ideal charge transfer requires infinite open loop gain of the opamp that was used in the S-C integrator. In a real integrator, part of the charge that is supposed to be transferred to the integrating capacitor is lost. The consequences are that the pole is not anymore at z=1 and the coefficient is different than the ratio of input and integrating capacitance. The real transfer function of DT S-C integrator is given by (15). The model of DT integrator is changed to Fig. 22, where α can be calculated according to (16). The pole frequency is moved from an ideal location at $z_p = 1$ to $z_p = \alpha$, while the gain of the integrating stage is moved from k to $k \cdot \alpha$. In this expression, A_0 is low-frequency open loop gain of the amplifier, $\sum C_{x,i}$ is the sum of all switched capacitors connected to the integrator i and $C_{I,i}$ is the integrating capacitance of the integrator i.

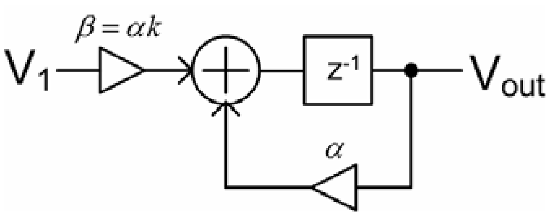


Fig. 22. DT integrator with a model of A0 influence

$$H(z) = \frac{k \cdot \alpha_i}{1 - \alpha_i z} \tag{15}$$

$$\alpha_i = \frac{\left[C_{I,i}(1 + A_0)\right]}{\left[C_{I,i}(1 + A_0)\right] + \sum C_{x,i}} \tag{16}$$

- **Dynamic properties of the integrator's opamp:** The response of each integrator during charge transfer depends on dynamic opamp characteristics: Slew-Rate (SR) and Gain Bandwidth Product (GB); in addition, it also depends on the finite open loop gain of the opamp (Maloberti, 2007). During charge transfer (phase Φ2 on the left part of Fig. 17), the opamp may enter 2 different modes of operation: SR limited mode at the first part of the transient, if the input charge is big enough, and the "linear" settling mode in the second part. The whole transient might follow a "linear" settling mode if input charge is small. For $G_{R0} \le SR$, where $G_{R0} = \left[dv_o(t)/dt\right]_{t=0}$ the time domain response of the output voltage can be modelled using (17), assuming the dominant pole model of the opamp with $1/\omega_p = \tau = 1/(2\pi \cdot GB)$; α is defined in (16), $k = C_{x,i}/C_{I,i}$ is the capacitor ratio, v_x is the voltage on the input capacitor slightly before the start of the charge transfer, and $v_{o1}(nT_s - T_s/2)$ is the output voltage of the integrator slightly before the start of the charge transfer. For $G_{R0} > SR$, the first part of the transient follows the SR limited behaviour according to (18), while the second part follows the "linear" settling behaviour according to (19). T_0 is the time where the slopes of the SR limited behaviour $v_{o1}(t)$ and settling limited behaviour $v_{o2}(t)$ are equal according to (20). The complete behaviour in that case is (21).

$$v_o(t) = v_o\left(nT_s - \frac{T_s}{2}\right) + \alpha k v_x \left(1 - e^{-\frac{t}{\tau}}\right) \text{ for } (nT_s - T_s/2) \le \text{t} < \text{n}T_s \tag{17}$$

$$v_{o1}(t) = v_o\left(nT_s - \frac{T_s}{2}\right) + SR * t \text{ for } t \le T_0 \tag{18}$$

$$v_{o2}(T_0) = (k \cdot \alpha \cdot v_x - SR * T_0)\left(1 - e^{-\frac{t - T_0}{\tau}}\right) \text{ for } t > T_0 \tag{19}$$

$$\left.\frac{dv_{o1}(t)}{dt}\right|_{t=T_0} = \left.\frac{dv_{o2}(t)}{dt}\right|_{t=T_0} \Rightarrow T_0 = \frac{\alpha k v_x}{SR} - \tau \tag{20}$$

$$v_o(t) = v_{o1}(T_0) + (\alpha \cdot k \cdot v_x - SR \cdot T_0)\left(1 - e^{-\frac{t - T_0}{\tau}}\right) \text{ for } T_0 < t \le nT_s \tag{21}$$

A complete Simulink model of the S-C integrator stage, excluding input coefficients (they are defined on the top level of the Simulink model shown in Fig. 21), is presented in Fig. 23. The model includes dynamic contributions due to SR and GB and the model of static error cause by A_0. The result of continuous time output voltage at the end of charge transfer that is at time nT_s is calculated by using an "Embedded" Matlab function, GBW_SR1, as shown on the left part of the S-C integrator Simulink model on Fig. 23. The constants for all integrators from Fig. 21 (constants for second integrator: Constant=tau(2), Constant1=alpha(2), Constant2=SR(2) and Constant3=Ts) are defined in the m-file and are available in the Matlab work-space. The code that evaluates (18) to (21) is presented in Fig. 24. Thermal noise generator (random Source8 on Fig. 23) is added at the output of the integrator and is thus multiplied by appropriate capacitor ratio. It models the contribution of the thermal noise generated in the opamp; input referred opamp noise power is multiplied by the appropriate coefficient $\sum(C_{x,i}/C_{I,i})$ so, that it can be added to the output of the integrator model.

5.4 Quantizer model

In a nonlinear model defined by state-space equations (7), it is assumed that the quantizer is described by a sign() function. This function has an infinitely steep response to the input signal. In reality, the gain of the quantizer alone is finite, which means that for very small

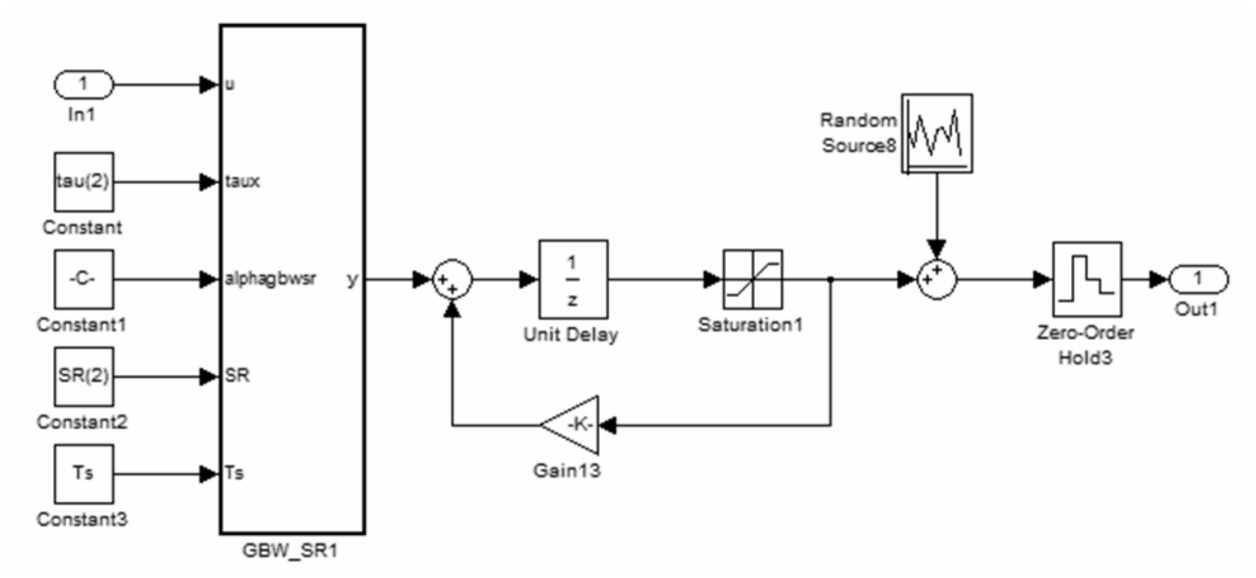


Fig. 23. Model of static and dynamic properties of non-inverting S-C integrator, including opamp thermal noise

```
function y     = gbwsr(u,taux,alpha,SR,Ts)

% this block calculates effects of GBW and Slew-Rate
% to the output voltage of an S-C integrator

Vs=u;
tau=taux;
if Vs<0
    SRx=-SR;
else
    SRx=SR;
end
GR0=alpha*Vs/tau;
if (abs(GR0)<=abs(SRx)) % gradient due to GBW smaller than SR
    y_Ts=alpha*Vs*(1-exp(-(Ts/(2*tau)))); % linear
else % GR0>SR % linear settling slower than SR
    T0=alpha*abs(Vs)/abs(SRx)-tau;
    if T0<Ts/2;
        y0=SRx*T0;
        y_Ts=y0+(alpha*Vs-SRx*T0)*(1-exp(-(Ts/2-T0)/tau));
    else
        y_Ts=SRx*Ts/2;
    end
end

y=y_Ts;
```

Fig. 24. Code that calculates output voltage of the S-C integrator, according to (18), (19), (20), and (21)

signals, the decision may be wrong. For low resolution devices, this is of no consequence but for high resolution devices many wrong decisions could have consequences by increasing the noise in the base-band. It is very difficult to determine the gain of a one bit quantizer from a system point of view, that is, by just observing the reaction to the input signals. The output signal is always limited, whatever the input signal might be. The intuitive explanation is that the system gain of the quantizer is dependent on the amplitude of the input signal: it is large for small signals and small for large input signals. Therefore, it would be necessary to check the NTF and the stability of the loop as a function of the quantizer's gain. Some authors have proposed complicated statistical models of the one bit quantizer (Boche & Monich, 2010); however, we prefer to use simple intuitive understanding and as simple a model as possible. It is possible to check the NTF for the linear model as a function of the large signal gain (system gain) of the quantizer, taking into consideration the different values of k from Fig. 2. Further discussions about the problem are presented in the continuation of this subsection, where the influence of a small system gain on a more realistic model of the quantizer is presented, together with its influence to the operation of the modulator.

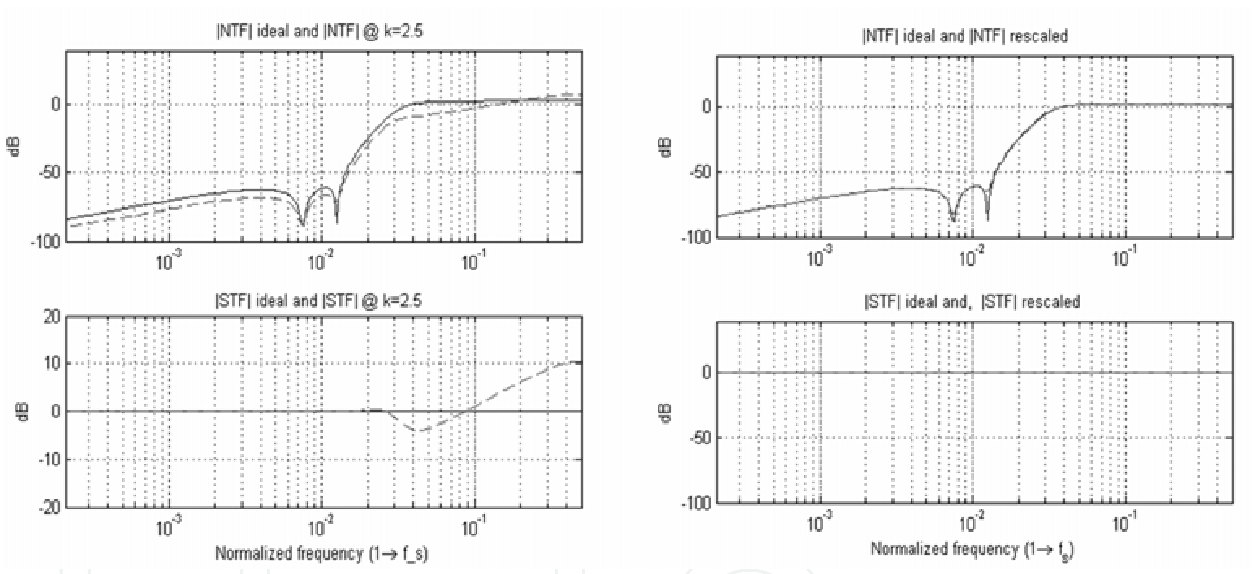


Fig. 25. |NTF| and |STF| plots for ideal quantizer and different system gain of the quantizer. Solid line: system gain of the quantizer k=1. Dashed line: system gain of the quantizer k=2.5. Upper plots: linear model of the |NTF|. Lower plots: |STF|. Left plots: un-scaled modulator; Right plots: rescaled modulator with system gain k=2.5 taken into considerations

Fig. 25 shows plots of the NTF and the STF for the ideal linear model of the fifth order modulator for different quantizers' gains. On both plots, the solid line presents the ideal linear model with system gain of the quantizer k=1, while the dashed lines represent plots of NTF and STF for system signal gain of a quantizer increased to k=2.5. The system gain of a linear model of a modulator (k=2.5) is deduced from extensive statistical simulations. Left plots are for an un-scaled modulator, while the right side of Fig. 25 plots the NTF and STF of rescaled linear model of the modulator. Rescaling is such that it compensates for the eventual change in the average system gain of the one bit quantizer in a linear model.

A simple nonlinear model of a one bit quantizer that includes the small signal gain of a differential stage, saturation block relay, offset voltage, input referred noise, and no latency is presented in Fig. 26. The model follows the implementation of the comparator, which is composed of the gain stage with gain G_comp, offset voltage Voff, and input referred noise power pncomp. The three blocks Fcn model the behaviour of the comparator in a good way: they calculate the decision under different conditions according to (22).

$$x=\begin{Bmatrix}1 & u\ge V_{th\_H}\\ 0 & u<V_{th\_H}\end{Bmatrix}\quad y=\begin{Bmatrix}-1 & u\le V_{th\_L}\\ 0 & u>V_{th\_L}\end{Bmatrix}\quad z=\begin{Bmatrix}1 & V_{th\_L}\le u\le V_{th\_H}\\ 0 & \text{otherwise}\end{Bmatrix}\tag{22}$$

The decision is stable and correct if the difference between input voltage and the offset voltage of the comparator, multiplied by the gain of differential stage, is bigger than the threshold voltage high (x), or if the difference is smaller than the threshold voltage low (y). In those two cases, the decision is +1 or -1, respectively. However, for small input voltage, the decision is difficult and may be arbitrary because of comparator noise (z). Using this model, we can study the influence of some of the characteristics of the internal quantizer (comparator) as a function of its performance. In Fig. 27, the simulation results for different small signal comparator gains are presented. The spectrum on the left is for a small signal comparator gain Gcomp=1000, which gives a correct result, while the spectrum on the right side of Fig. 27 for Gcomp=100 is not acceptable because the SnR has been degraded considerably. The explanation for the increased base-band noise level is the following: because of very small signal gain of the comparator’s differential stage, the decision was many times dictated by the random noise of the comparator instead of the signal, and this causes the increase of the baseband noise.

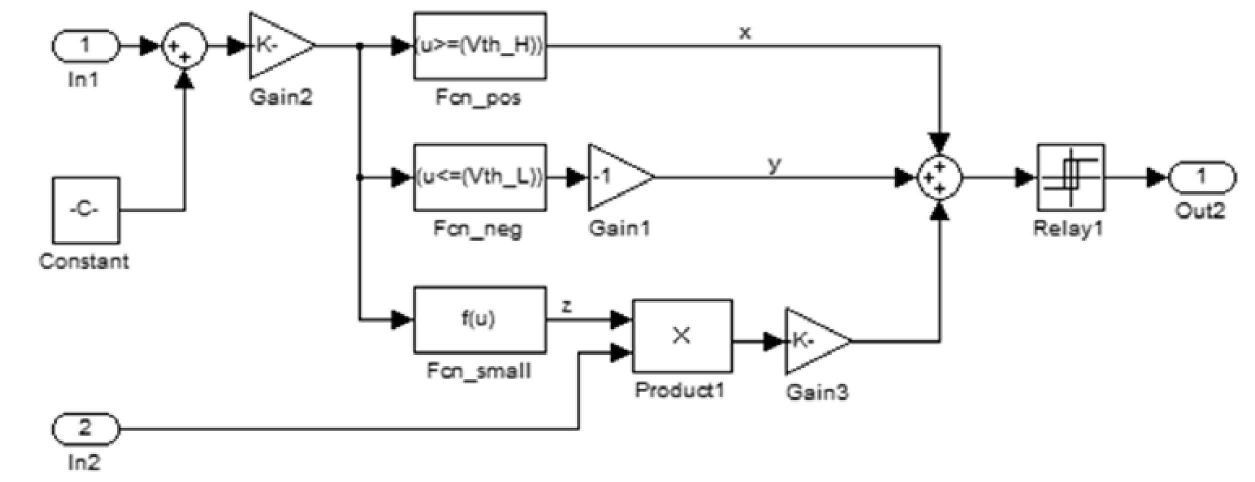


Fig. 26. Nonlinear model of the quantizer

5.5 Simulation and analysis

Using the models defined in the previous subsections, one can study the influence of many different effects due to non-ideal parameters. The complete model can also be used to optimize the main parameters of the circuit. Due to lack of space, only two simulation results will be presented for our fifth order modulator. In reality, many different simulations must be performed to verify the correctness of the design and to verify the influence of the non-ideal parameters on the behaviour of a mixed signal module.

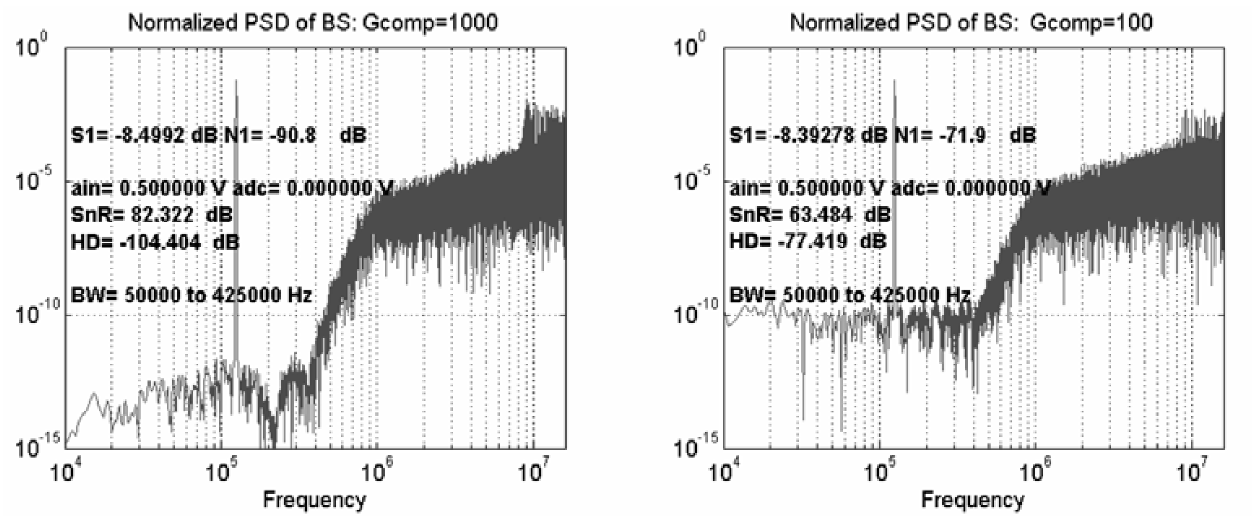


Fig. 27. BS spectrum of the example modulator for nonlinear Quantizer model. Left spectrum: Gcomp=1000, Right spectrum: Gcomp=100

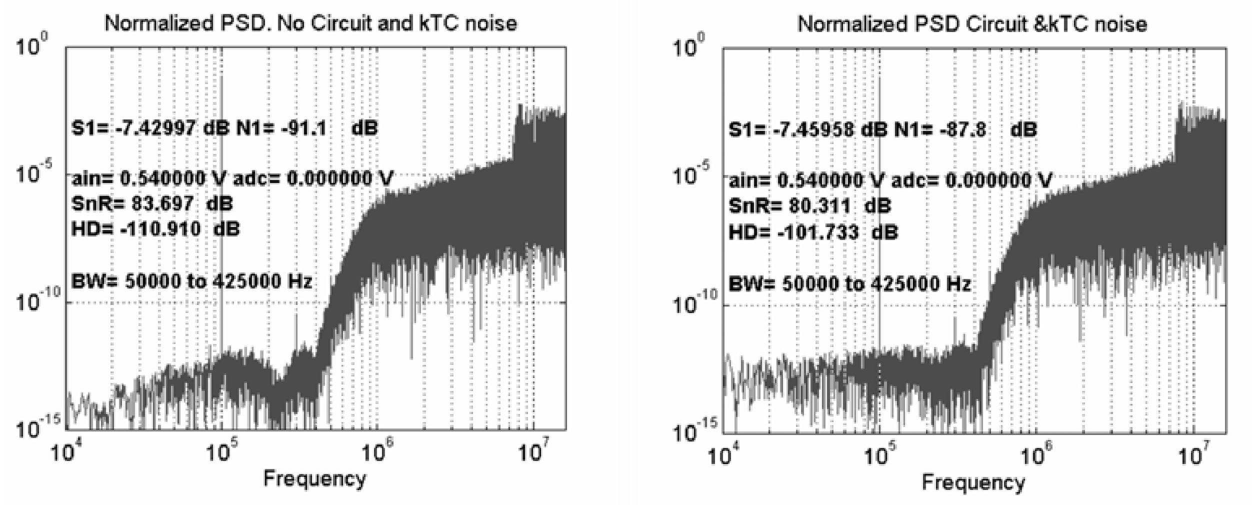


Fig. 28. Spectrum of the BS of the fifth order modulator, with non-ideal effects included. Left spectrum: No noise circuit included, Right spectrum: All noise sources and all other non-ideal effects included

Fig. 28 shows two simulation results. Both are calculated from the bit-stream of our fifth order example modulator. All non-ideal effects are included with parameters given in (23).

$$\begin{aligned}
&A_{0,i} = 1000; \quad i = 1...5 \\
&BW_i = 5 \cdot f_s \, [Hz]; \quad i = 1...5 \\
&\mathbf{SR} = [70, 38, 14, 8, 8] \left[\frac{V}{\mu s}\right] \\
&\mathbf{V}_{ndop} = [15, 40, 80, 80, 80] \left[\frac{nV}{\sqrt{Hz}}\right] \\
&Temp = 150\ ^{\circ}C
\end{aligned} \tag{23}$$

The kT/C noise source effects have been excluded from the simulation for the left part of Fig. 28, while on the right side of Fig. 28, all noise sources have been included with optimized unit capacitor sizes. The evidence of circuit noise is clearly seen on the right part of the figure. The switched capacitors and integrator contributions have been optimized so that their influences are acceptable and power consumption is optimized.
The last simulation result shown in Fig. 29 presents a Monte Carlo analysis of our fifth order modulator, with all non-ideal effects included. The absolute values of the capacitances and the capacitor ratios were randomly changed around their nominal (mean) value, with standard deviations defined on the upper part of the scatter plot. This information is compliant with technology information. The capacitor ratios were spread according to the matching information from the process technology. The properties of the opamps and the comparator were kept constant, with values slightly different than the values defined in (23). It would be possible to include also the spread of this circuit parameter values obtained from the circuit simulation results. The x-axis of Fig. 29 represents rms "distance" of each experiment from its nominal value, while the y-axis shows the SnR in dB. Each circle represents one result.

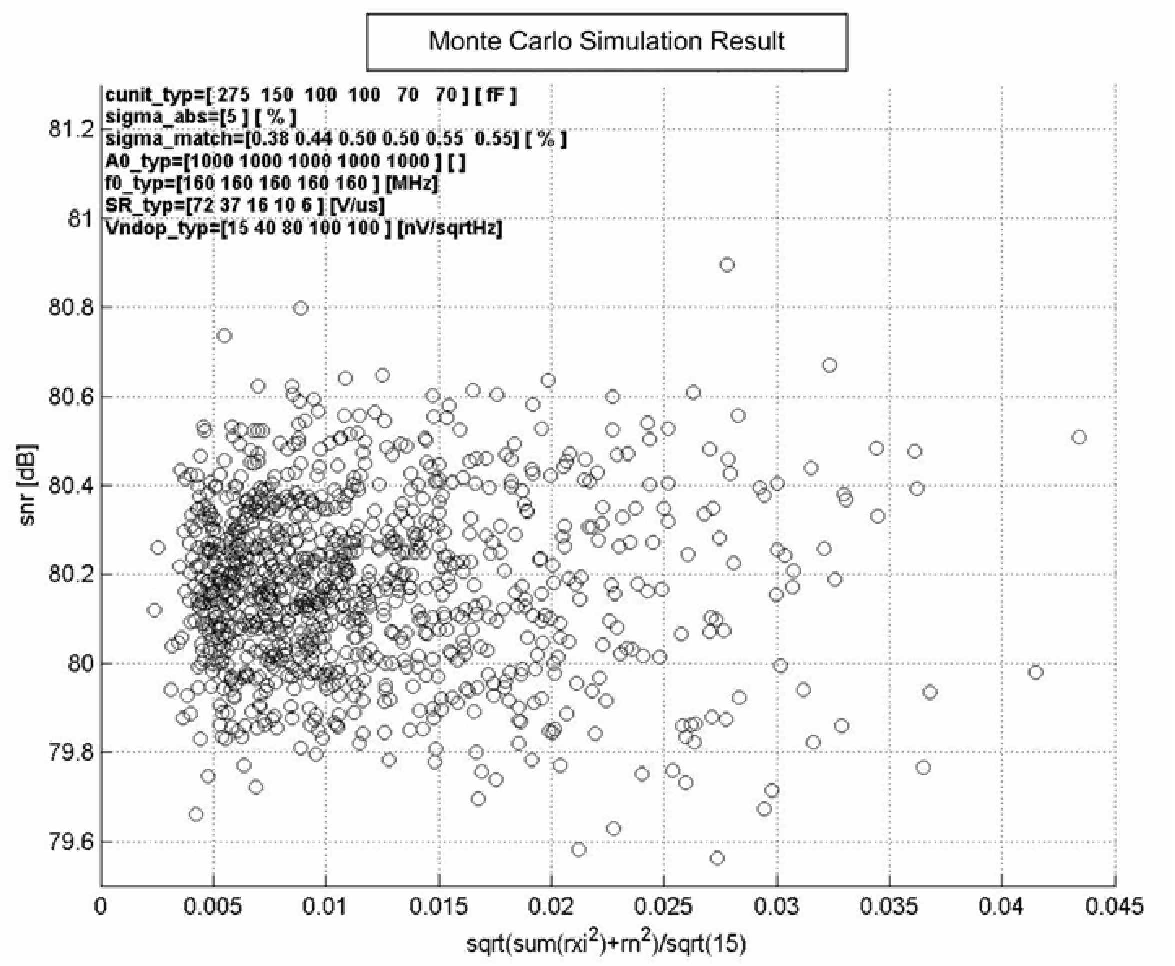


Fig. 29. Monte Carlo simulation result of the 5th order modulator. Each circle represent result (SnR) of one simulation at the conditions defined in upper left portion of the picture. The capacitors are spread by $|\sigma| = 5\%$. The x-axis is the rms distance of each capacitor set of the experiment from the nominal rms value

6. Decimator

Each Σ-Δ A/D converter is composed of a modulator and a decimation filter. This pair forms a simple mixed-signal system (circuit). Many different decimation filter architectures are possible (Schrier & Temes, 2005). In this chapter, we will not go into the details of a decimation filter design; rather, we will present the bit-true model of one example filter using simple Simulink element library and blocks. The model used is very simple. It leads to the efficient design of decimation filters.
The simplest decimation filter architecture is the so-called sinc filter defined in z-domain transfer function (24), where *ord* is the order of the filter and R_1 is the decimation factor. To implement that, a delay, registers, and adder Simulink blocks are needed; the structure is regular which leads to efficient VLSI design.

$$H(z)=\left[\frac{1-z^{-R_1}}{1-z^{-1}}\right]^{ord} \tag{24}$$

Usually, the order of the decimation filter must be bigger than the order of the modulator to suppress the out of band quantization noise sufficiently. For our example, a sixth order Sinc decimation filter is used. The block diagram of the filter is presented in Fig. 30. A complete filter is built with fixed word-length of WL=32 bits using so called wrap-around two's complement arithmetic. The digital integrator stages are running with f_s while differentiators are running with f_s/R_1 (R_1=32). The word length needed is calculated in such a way that no additional quantization noise is added into the process and that the wrap-around two's complement arithmetic works correctly. At the output of the filter, the word-length is reduced to 24 bits using block Gain6 with correct parameter settings.

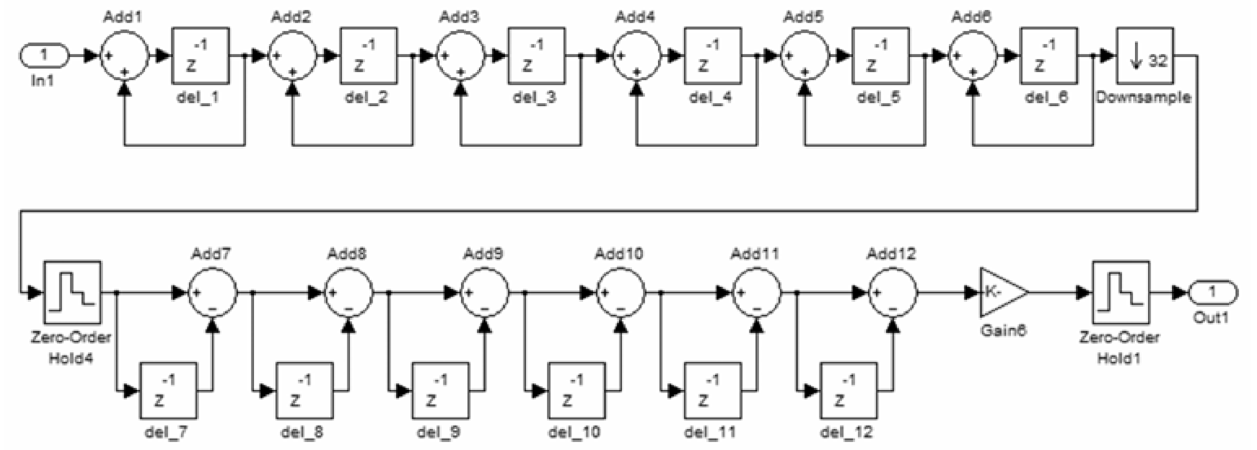
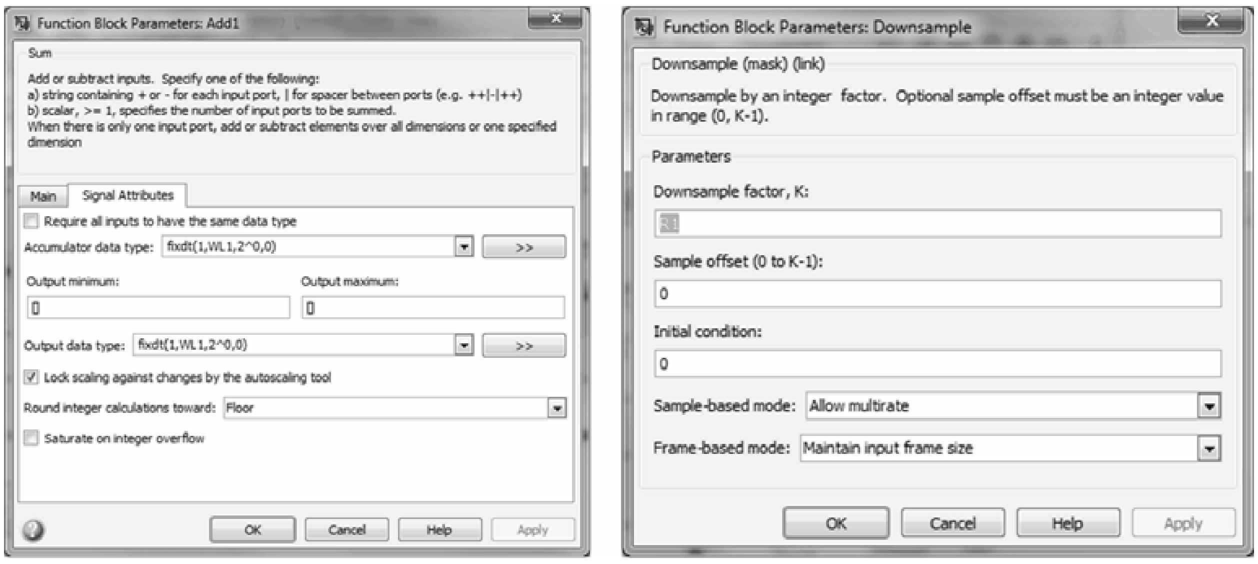


Fig. 30. 6th order sinc decimation filter structure

The number of bits used in integrator stages can be fixed by the appropriate number of bits of the input signal and appropriate settings of the adders. The input signal to the decimation filter is obtained from the bit-stream of the modulator; in our example, the modulator has a one bit internal quantizer with a data type double. For the fixed point arithmetic used in the decimation filter, the BS signal is first converted to an integer by using block Convert (SI) on Fig. 33. Fixed point arithmetic inside a decimation filter is defined by parameters inside the adders, as presented on the left part of Fig. 31, where accumulator data type is set to fixed point with WL1=32 bits. This number is calculated in Matlab m-file using formula $WL_1 = ceil\left(ord_1 * \log 2(R_1)\right) + WL_{in}$; the output data type of the adder block is also set to WL1. The adder must not saturate on integer overflow to accommodate wrap-around arithmetic.


Function Block Parameters: Add1

Sum

Add or subtract inputs. Specify one of the following:
a) string containing + or - for each input port, | for spacer between ports (e.g. ++|-|++)
b) scalar, >= 1, specifies the number of input ports to be summed.
When there is only one input port, add or subtract elements over all dimensions or one specified dimension

Main | Signal Attributes

- [ ] Require all inputs to have the same data type
- Accumulator data type: fixdt(1,WL1,2^0,0)
- Output minimum: []
- Output maximum: []
- Output data type: fixdt(1,WL1,2^0,0)
- [✓] Lock scaling against changes by the autoscaling tool
- Round integer calculations toward: Floor
- [ ] Saturate on integer overflow

Function Block Parameters: Downsample

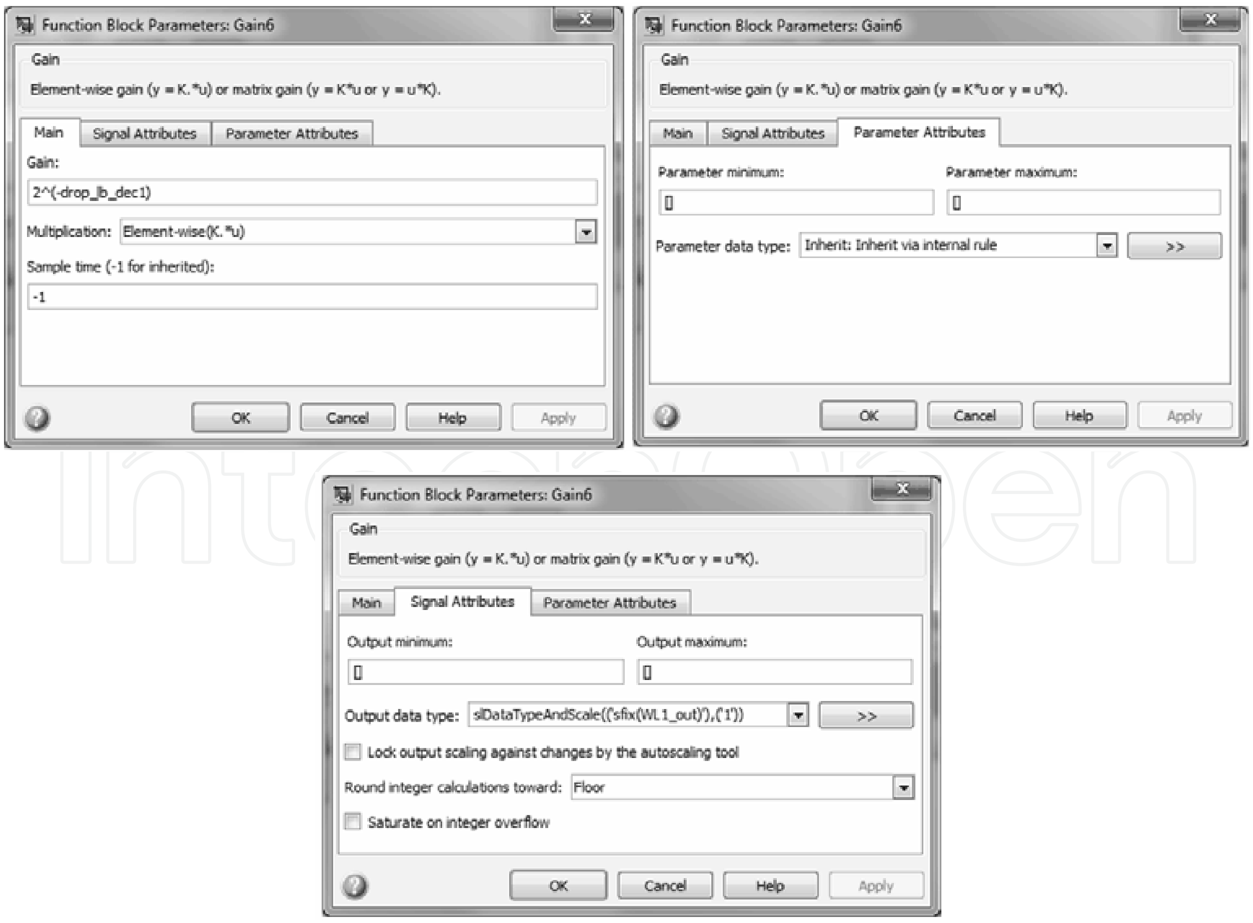
Downsample (mask) (link)

Downsample by an integer factor. Optional sample offset must be an integer value in range (0, K-1).

Parameters

- Downsample factor, K: [illegible]
- Sample offset (0 to K-1): 0
- Initial condition: 0
- Sample-based mode: Allow multirate
- Frame-based mode: Maintain input frame size

Fig. 31. Adder and Down-sampler parameters. Left: Adder. Right: Down-sampler


Function Block Parameters: Gain6

Gain

Element-wise gain (y = K.*u) or matrix gain (y = K*u or y = u*K).

Main | Signal Attributes | Parameter Attributes

- Gain: 2^(-drop_lb_dec1)
- Multiplication: Element-wise(K.*u)
- Sample time (-1 for inherited): -1

Function Block Parameters: Gain6 (Parameter Attributes)

- Parameter minimum: []
- Parameter maximum: []
- Parameter data type: Inherit: Inherit via internal rule

Function Block Parameters: Gain6 (Signal Attributes)

- Output minimum: []
- Output maximum: []
- Output data type: slDataTypeAndScale(('sfix(WL1_out)'),('1'))
- [ ] Lock output scaling against changes by the autoscaling tool
- Round integer calculations toward: Floor
- [ ] Saturate on integer overflow

Fig. 32. Settings for block Gain6 that determines output word-length

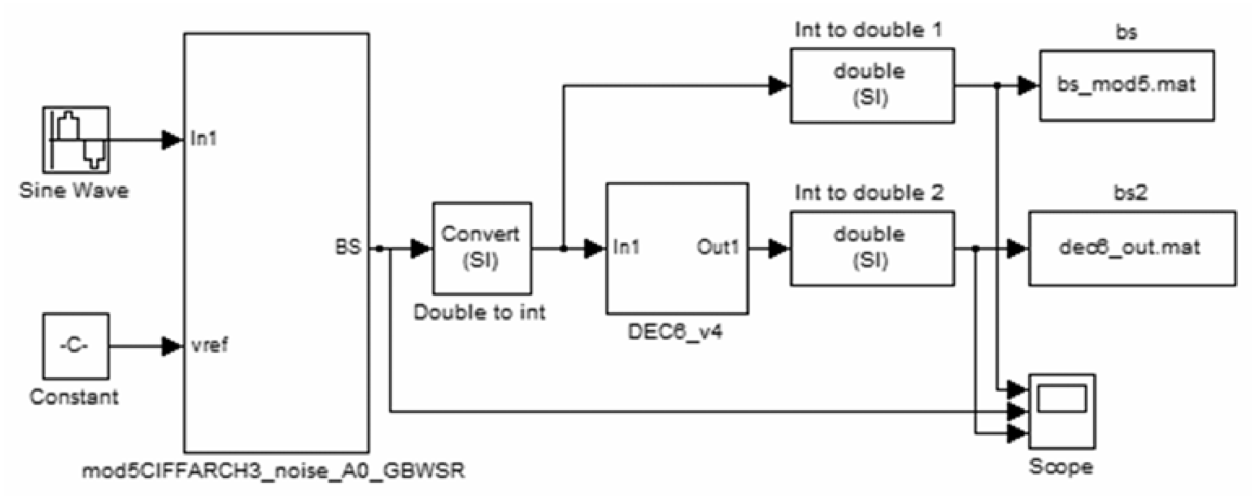


Fig. 33. Top-level Simulink block diagram for modulator-decimator simulation

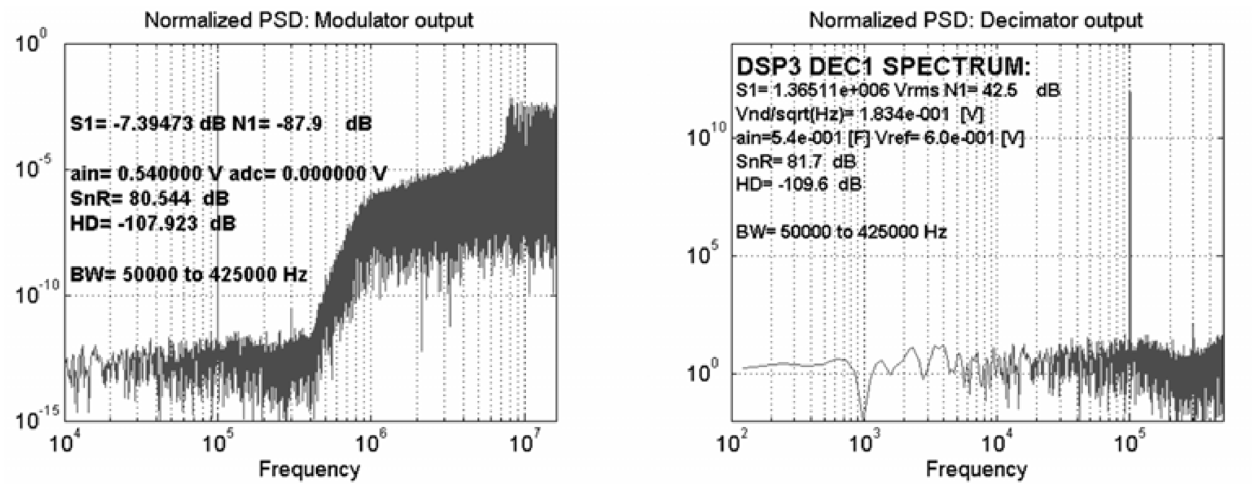


Fig. 34. Left spectrum: Spectrum of the modulator's BS including non-ideal effects. Spectrum on the right: The decimation filter output (the decimator is driven by the BS with spectrum on the left side

At the end of the sixth integrator stage, every R_1th sample is taken, which is implemented by a Down-sampler. The parameter settings are presented on the right side of Fig. 31. The whole decimator is running with the same number of bits WL1 because the structure in this case is more regular. Nevertheless, it is possible to use a smaller number of bits at the beginning of the chain and increase the word-length at the end of the structure, taking into consideration only the quantization noise penalty.

For our example decimator, the lower eight bits are removed by block Gain6 (from Fig. 30) at the output by setting appropriate parameters as shown in Fig. 32.

Fig. 33 shows a top-level Simulink block diagram that includes a modulator (mod5CIFFARCH3_noise_A0_GBWSR) with all previously defined non-ideal effects, and a model of a decimation filter (DEC6_v4) using previously defined settings for fixed-point arithmetic.

Fig. 34 shows the results of a Simulink simulations in two spectrums. The left plot shows the spectrum of the bit-stream of the 5th order modulator model that includes all non-ideal circuit effects defined before with nominal capacitor ratios, while the spectrum on the right side corresponds to the bit-true decimation filter output.

7. Conclusion

This chapter deals with modelling and simulation of a mixed-signal module (circuit) using high level Matlab/Simulink model on a high hierarchical level. More specifically, it presents an example design and the modelling details of a fifth order Σ-Δ A/D converter, which is built of a fifth order modulator and sixth order decimation filter. The modulator is modelled by using data types "double" and models of many non-ideal circuit effects, while the decimator is modelled by using different fixed-point data types. The most important non-ideal effects are included into the model of the "analogue part"; the decimation filter uses a bit-true, fixed-point arithmetic and hence we can say that it uses a bit-true model. Using the proposed modelling technique, one can speed up the design and simulations of complex mixed-signal circuits and systems considerably. For example, in the design shown in Fig. 34, less than one minute simulation time is needed to obtain 2^{18} samples and to get the presented final spectrums.

8. Acknowledgment

This work was partly supported by the NAMASTE Centre of Excellence.

9. References

Boche, H. &. Monich, U. J. (2010). Behavior of the Quantization Operator for Band-Limited, Nonoversampled Signals. *IEEE Trans. on Information Theory* , vol. 56, No. 5, pp. 2433-2440.

De la Rosa, J. M. (2011). Sigma_Delta Modulators: Tutorial Overview, Desing Guide, and State-of-Art Survey. *IEEE Trans. on Circuits and systems - I: Regular papers* ,vol. 58, no. 1, pp. 1-21.

Gregorian, R. T. & Temes, G.C. (1986). *Analog MOS Integrated Circuits for signal processing,* John Wiley & Sons, New York

Maloberti, F. (2007). *Data COnverters.* New York: Springer.

Moris, K. (2004). Destination DSP; Methodologies for signal processing success. *FPGA and Programmable Logic Journal (www.fpgajournal.com/articles/20041130_dsp.htm)* .

Perelroyzen, E. (2007). *Digital Integrated Circuits: Design-for-Test Using Simulink and Stateflow.* N.Y.: CRC Press.

S.Mann, D. (1999). Limit Cycle Behaviour in the Double Loop Band-pass Sigma-Delta A/D Converter. *IEEE Trans CAS.II, Analogue and digital Signal Processing* , vol .46, no. 8, pp. 1086 – 1089.

Schreier, R. (2009). *The Delta-Sigma Toolbox Version 7.3.* (http://www.mathworks.com/matlabcentral/fileexchange/delsig), Mathworks.

Schrier, R. T. & Temes, G.C. (2005). *Understanding Delta-Sigma Data Converters,* John Wiley & Sons, New York.

Strle, D. &. Kempe, V. (2007). MEMS-based inertial systems. *Inf. MIDEM*, pp. 199-209.

Strle, D. (2006). Limit Cycles in High Order Sigma Delta Modulators. *Inf. MIDEM,* vol. 36, No.1, pp. 11-18.

Synopsis. (2004). *Saber Users Guide*. Synopsis.

Zepernick, H. &. (2005). *Pseudo Random Signal Processing: Theory and applications,* John Wiley & Sons, New York.