A METHODOLOGY FOR EVALUATING THE PRECISION OF FIXED-POINT SYSTEMS

Daniel Menard †

† LASTI - University of Rennes I
6 Rue de Kerampont - 22300 Lannion, France
Daniel.Menard@enssat.fr

Olivier Sentieys †‡

‡ IRISA/INRIA - Campus de Beaulieu
35042 Rennes cedex, France
Olivier.Sentieys@enssat.fr

ABSTRACT

The minimization of cost, power consumption and time-to-market of DSP applications requires the development of methodologies for the automatic implementation of floating-point algorithms in fixed-point architectures. In this paper, a new methodology for evaluating the quality of an implementation through the automatic determination of the Signal to Quantization Noise Ratio (SQNR) is presented. The modelization of the system at the quantization noise level and the expression of the output noise power is detailed for linear systems. Then, the different phases of the methodology are explained and the ability of our approach for computing the SQNR efficiently is shown through examples.

1. INTRODUCTION

The efficient implementation of digital signal processing (DSP) algorithms in embedded systems requires the use of fixed-point arithmetic in order to satisfy the cost and power consumption constraints of these applications. The manual transformation of floating-point data into fixed-point data is a time-consuming and error prone task. Moreover, the reduction of the time-to-market of the applications needs to use high level development tools which allow the automation of some tasks. The manual conversion to the fixed-point level hinders the reduction of the development time [1]. Thus, methodologies for the automatic transformation of floating-point data into fixed-point data have been proposed [2, 3].

The efficient implementation of algorithms in hardware or software architectures requires to evaluate the precision of the implementation. The most common used criteria for evaluating the precision is the Signal to Quantization Noise Ratio (SQNR) which is the ratio between the signal power and the quantization noise power. Most of the available methodologies for computing the SQNR are based on a bit true simulation of the fixed-point algorithm [2, 3, 4, 5]. Thus, `C++` classes for emulating the fixed-point mechanisms have been developed as in *SystemC* [5]. This technique suffers from a major drawback which is the time required for the simulations [4]. They are made on floating-point machines and the extra-code used for emulating the fixed-point mechanisms of the operations increases the simulation time between one and two orders of magnitude compared to a traditional simulation with floating-point data types [3]. Moreover, for obtaining an accurate estimation of the noise statistic parameters, a great number of samples must be taken for the simulation. These long simulation times become a severe limitation when these methods are used in the process of data word-length optimization where multiple simulations are needed for exploring the design-space of the different data word-lengths [2]. Different techniques [2, 3, 4] have been investigated for reducing this simulation time.

An alternative to the simulation based method can be an analytical approach which determines the expression of the noise power at the output of a system according to the statistical parameters of the different noise sources. For this approach, two advantages can be underlined. Firstly, this method gives an analytic expression of the SQNR and thus provides more information about the noise behaviour in the system than a simulation based method which only gives the numerical value of the SQNR. Secondly, the requisite execution time for evaluating the noise power is definitely lower, especially for the process of data word-length optimization in hardware design. Indeed, the determination of the SQNR expression is done only once.

Analytical expressions of the SQNR have been formulated for some particular DSP applications as in [6]. In [7], an analytical approach based on the propagation in the system Signal Flow Graph (SFG) of the noise statistical parameters is proposed. This method suffers from two major drawbacks. The noise models are not realistic and the method is limited to non-recursive structures.

In this paper a new method for the automatic SQNR evaluation based on an analytical approach is proposed. It uses a realistic noise model which takes into account the different quantification laws (rounding and truncation). Moreover, this method allows to compute the SQNR in non-recursive structures and in linear recursive structures. The scope of this paper is limited to linear systems. In this case, our approach is based on the automatic computation of the transfer function of the system from its SFG representation. After a presentation of the different noise models in section 2, the theoretical concepts of the method are detailed in section 3 and the expression of the output noise power is defined. Then, the techniques used for implementing this method are briefly explained in section 4. Finally, the ability of our approach for computing the SQNR efficiently is shown through some examples in section 5.

2. NOISE MODELS

The use of fixed-point arithmetic introduces an unavoidable quantization error when a signal is quantified. In this section, the different available quantization noise models and the modelization of the propagation of these noises through the operators are presented.

2.1. Quantization noise models

A common used model for the continuous-amplitude signal quantization, has been proposed by Widrow [8]. The quantization of a signal x is modeled by the sum of this signal and a random variable b. This additive noise b is a stationary and uniformly distributed

white noise that is uncorrelated with the signal x and the other quantization noises. This model has been extended for modeling the computation noise in a system resulting from the elimination of some bits during a format conversion (cast operation). In [9], the authors have demonstrated that the model presented above can be used if the dynamic range of the signal is sufficiently greater than the quantum step size and if the bandwidth of the input is large enough. Moreover, the expressions of the first and second-order moments of the noise have been refined. In [10], the number of bits eliminated during a cast operation has been taken into account for expressing the first and second-order moments of the quantization noise.

2.2. Propagation noise models

The propagation noise model of an operator defines the expression of the output noise from the input noises and signals. An operator with two inputs X and Y and one output Z is under consideration. Each input and output is made up of a signal s and a quantization noise b. The expressions of the output noise b_z of an adder and a multiplier are

$$\begin{array}{ll} b_z = b_x + b_y & (\text{adder}) \\ b_z = b_x s_y + b_y s_x + b_x b_y & (\text{multiplier}) \end{array} \quad (1)$$

For the multiplier, the term $b_x b_y$ represents the product of two noises which is much smaller than the two other terms, then it can be neglected in the following. As well, for the multiplication of X by a constant C, the expression of the output noise is $b_x C' + \Delta_C s_x$ where C' is the value of the quantified constant and Δ_C a bias corresponding to the difference between C' and C.

3. THEORETICAL APPROACH

3.1. Output noise expression

A linear time-invariant system made up of N_e inputs $x_j(n)$ and one output $y(n)$ is considered. For multi-outputs systems our method is repeated for each output. Let $H_j(z)$ be the partial transfer function between the output $Y(z)$ and each input $X_j(z)$, and $h_j(n)$ be the impulse response associated with $H_j(z)$. The expression of the output $y(n)$ is equal to

$$y(n) = \sum_{j=0}^{N_e-1} h_j(n) * x_j(n) \quad (2)$$

The fixed-point version of this system is detailed thereafter. Let $\widehat{x_j}(n)$ be the j^{th} quantified system input and $\widehat{H_j}(z)$ be the transfer function between the output $Y(z)$ and the input $X_j(z)$ with quantified coefficients. The use of fixed-point arithmetic gives rise to three kinds of source of error which are due to the quantization of the system inputs, the quantization of the coefficients and the error generated when some bits are eliminated during a cast operation.

Let b'_{ej} be the quantization noise associated with each quantified system input $\widehat{x_j}(n)$ such as defined in equation 3. Let b_{ej} be the output noise due to the propagation in the system of the input noise b'_{ej}

$$\widehat{x_j}(n) = x_j(n) + b'_{ej}(n) \quad (3)$$

Let $\Delta H_j(z)$ be the transfer function corresponding to the difference between the quantified transfer function $\widehat{H_j}(z)$ and the real transfer function $H_j(z)$

$$\Delta H_j(z) = \widehat{H_j}(z) - H_j(z) \quad (4)$$

The elimination of some bits during a cast operation in the system leads to the generation of a quantization noise b'_{gi}. Let b_{gi} be the output noise due to the propagation in the system of the generated noise b'_{gi}. The transfer function between the system output and the noise source b'_{gi} is $H_{gi}(z)$. Let b_g be the sum of the N_g noise sources b_{gi} generated in the system

$$b_g(n) = \sum_{i=0}^{N_g-1} h_{gi}(n) * b'_{gi}(n) \quad (5)$$

The expression of the fixed-point version of the system output $\hat{y}(n)$ is equal to

$$\hat{y}(n) = \sum_{j=0}^{N_e-1} \widehat{h_j}(n) * \widehat{x_j}(n) + b_g(n) \quad (6)$$

By introducing the equations 3, 4 and 5 in equation 6, the expression of $\hat{y}(n)$ becomes

$$\hat{y}(n) = \sum_{j=0}^{N_e-1} \left(h_j(n) + \Delta h_j(n)\right) * \left(x_j(n) + b'_{ej}(n)\right) + b_g(n) \quad (7)$$

As explained in section 2.2, the term $\Delta h_j(n) * b'_{ej}$ involves the multiplications of *noise* terms and thus, it can be neglected in regard of the other terms. Then, the error b_y corresponding to the difference between the fixed-point and the floating-point version of the system output is equal to

$$b_y = \hat{y}(n) - y(n) = b_q(n) + b_h(n) \quad (8)$$

with

$$b_q(n) = b_g(n) + b_e(n) \quad (9)$$

$$b_e(n) = \sum_{j=0}^{N_e-1} b_{ej}(n) = \sum_{j=0}^{N_e-1} h_j(n) * b'_{ej}(n) \quad (10)$$

and

$$b_h(n) = \sum_{j=0}^{N_e-1} b_{hj}(n) = \sum_{j=0}^{N_e-1} \Delta h_j(n) * x_j(n) \quad (11)$$

A representation of the noise model of the system is given in figure 1. This noise model is a generalization of the one proposed in [6].

3.2. Statistical parameters of b_q

The expression of the statistical parameters of b_{ej} and b_{gi} are identical. Indeed, these noises represent the output of a linear subsystem excited by a white noise (b'_{ej} or b'_{gi}) as defined in section 2.1. For simplicity, let b_j be the output of this subsystem, b'_j the input and $H_j(z)$ its transfer function

$$b_j(n) = b'_j(n) * h_j(n) \quad (12)$$

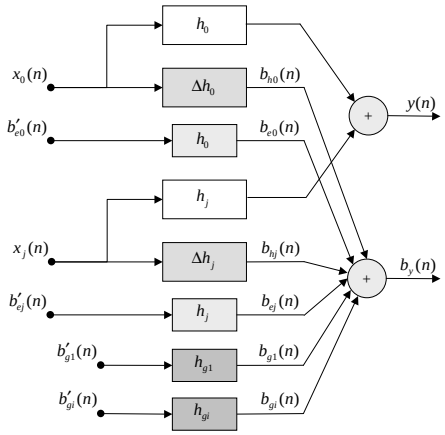

Fig. 1. Representation of the linear system with the noise sources

The noise b'_j is a white noise with a mean $\mu_{b'_j}$ and a variance $\sigma^2_{b'_j}$. These statistical parameters are computed from the models presented in [8] and [10]. From the theory of linear systems, the second-order moment of the output noise $b_j(n)$ is equal to

$$E(b_j^2) = \left(\mu_{b'_j} H_j(e^{j0})\right)^2 + \frac{\sigma^2_{b'_j}}{2\pi} \int_{-\pi}^{\pi} |H_j(e^{j\omega})|^2 d\omega \qquad (13)$$

The first term of this expression represents the mean of b_j and the second term its variance.

From equations 9 and 5, the noise b_q is the sum of $N_e + N_g - 2$ random variables b_j. From the noise proprieties presented in section 2.1, each noise source b'_j is not correlated with any signal or noise source. Thus, the output b_j of the linear system h_j excited by the noise b'_j will not be correlated with any signal or noise source. The expression of the first and second-order moments of b_q are equal to

$$\mu_{b_q} = E(b_q) = \sum_{j=0}^{N_e-1} \mu_{b_{ej}} + \sum_{i=0}^{N_g-1} \mu_{b_{gi}} \qquad (14)$$

$$E(b_q^2) = \sum_{j=0}^{N_e-1} \sigma^2_{b_{ej}} + \sum_{i=0}^{N_g-1} \sigma^2_{b_{gi}} + \mu^2_{b_q} \qquad (15)$$

3.3. Statistical parameters of b_h

The expression of the output error b_{hj} due to the quantization of the coefficients of $H_j(z)$ is defined through equation 11. Each system input x_j is considered to be a stationary and ergodic random variable. The second-order moment of b_{hj} is obtained from the spectral density $\phi_{x_j x_j}$ of the input system x_j

$$E(b_{hj}^2) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \phi_{x_j x_j}(e^{j\omega}) \left|\Delta H_j(e^{j\omega})\right|^2 d\omega \qquad (16)$$

By following the same approach, the cross-correlation between two noises b_{h_j} and b_{h_k} can be obtained from the mutual spectral density $\phi_{x_j x_k}$ between the system inputs x_j and x_k

$$E(b_{h_j} b_{h_k}) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \phi_{x_j x_k}(e^{j\omega}) \Delta H_j(e^{j\omega}) \Delta H_k(e^{-j\omega}) d\omega \qquad (17)$$

From equation 11, the noise b_h is the sum of $N_e - 1$ random variables. Thus, the expression of the first and second-order moments are equal to

$$\mu_{b_h} = E(b_h) = \sum_{j=0}^{N_e-1} \mu_{x_j} \Delta H_j(e^{j0}) \qquad (18)$$

$$E(b_h^2) = \sum_{j=0}^{N_e-1} E(b_{h_j}^2) + \sum_{j=0}^{N_e-1} \sum_{\substack{k=0 \\ k\neq j}}^{N_e-1} E(b_{h_j} b_{h_k}) \qquad (19)$$

The different terms of this expression are computed from equations 16 and 17 and require the knowledge of the spectral density $\phi_{x_j x_j}$ of each input system and the mutual spectral density $\phi_{x_j x_k}$ between all the system inputs. All the statistical parameters associated with the inputs x_j are specified by the user.

3.4. Output noise power

The output noise power P_{b_y} corresponds to the second-order moment of b_y. From equation 8 and the quantization noise proprieties, the expression of the second-order moment of b_y is

$$P_{b_y} = E(b_y^2) = E(b_q^2) + E(b_h^2) + 2\mu_{b_q}\mu_{b_h} \qquad (20)$$

The different terms of this expression are computed from the equations 14, 15, 18 and 19.

4. SQNR COMPUTATION METHODOLOGY

We have developed a new methodology to compute the SQNR at the output of an application by using an analytical method based on the theoretical approach presented in section 3. This analytical method uses as input, an application representation based on a Signal Flow Graph (SFG) where all the fixed-point formats and parameters (quantification mode ...) are defined. In order to be independent of the specification language of the application, the tool is split into two parts, a front-end and a back-end. The front-end transforms the original application representation in a unique intermediate representation called G_s corresponding to the application SFG which specifies the behaviour of the algorithm at the fixed-point level. At present, the front-end of a high level synthesis tool is used for generating the application SFG G_s. The back-end determines the SQNR according to the analytical approach. It consists of several successive transformations (T_1 to T_3) of the SFG, described in the following sections.

4.1. Back-end description

The goal of the transformation T_1 is to represent the application at the quantization noise level through the graph G_{sn}. The aim of the first stage of T_1 is to detect and to include in the graph the 3 types of noise source defined in the section 3.1. Then, each operator is replaced by its noise propagation model as defined in section 2.2.

The goal of the transformation T_2 is to determine the transfer functions of the system. These transfer functions are computed

with the $\mathcal{Z}$ transform from the linear functions which define the system. They are built by traversing the graph from the inputs to the output. But this technique is unusable if cycles are present in the graph as in our case when recursive structures are considered. Consequently, the use of this technique requires first of all, to transform this graph in several directed acyclic graphs (DAG). The different stages of the transformation T_2 are presented below.

The goal of the transformation T_{21} is to transform the graph G_{sn} into several DAG G_k if G_{sn} contains circuits. After a fast circuit detection algorithm all the circuits are enumerated. Then, each circuit is properly dismantled at the last common points of the circuit and the paths between any point of this circuit and the system output node.

The aim of the transformation T_{22} is to build the graph G_{eq} from the different DAG G_k. The graph G_{eq} is a weighted and directed graph which specifies the system with a set of linear functions. The linear function associated with each DAG is obtained by a depth-first traversal of the DAG with a post-order recursive algorithm.

The goal of the transformation T_{23} is to compute the weighted and directed graph G_{Hi} which specifies the application with a set of intermediate transfer functions. After a set of variable substitutions, the transfer function of a subsystem is computed from the $\mathcal{Z}$ transform of the linear function associated with this subsystem.

The aim of the transformation T_{24} is to build the weighted tree A_H which specifies the algorithm with a set of global transfer functions between the output and each input of the system. A_H represents the modelization of the system given at the figure 1.

The transformation T_3 computes the SQNR from the output signal power and the output quantization noise power. This one is computed from the approach detailed in section 3 after the evaluation of the frequency responses of the different subsystems from their transfer functions. The output signal power is specified by the user or is computed from the transfer functions of the system and the parameters of the input signals according to the same method.

5. RESULTS

The ability of our method for computing the SQNR has been successfully verified on several classical DSP applications such as FIR and IIR filters and the FFT algorithm. Indeed, the transfer functions obtained after the transformation T_2 are exact. The accuracy of our estimation has been analyzed by computing the relative error between the estimation of the output noise power obtained with a bit true simulation and with our method. Experiments have been achieved with different alternatives for the quantization mode, the scaling strategy and the data word-length. The results obtained with the two kinds of estimation are very closed. The relative error between these two estimations is included between 0.29% and 8.2% for different implementations of a second-order IIR filter and smaller than 1.5% for a 16 taps FIR filter. Two different reasons can explain the difference between these two estimations. First, the accuracy of the estimation based on simulation depends on the number of samples used. Secondly, for our method, a slight error can be present due to the assumptions made on the noise model. Thus, the accuracy of our methodology is sufficient enough to evaluate the precision of a fixed-point implementation.

The execution time of the different parts of the tool has been measured. The global SQNR computation times for a fourth order cascaded IIR filter (IIR 4) and 256 taps FIR filter (FIR 256) are smaller than 1 second. Most of the time is consumed by the transformation T_2 and especially by the circuit and path enumeration procedure. The execution time of the transformation T_2 is $0.65s$ for a IIR 4 and $0.86s$ for a FIR 256. These results show the efficiency of our approach on SFG with multiple cycles and on acyclic SFG with a great number of nodes. These results are smaller than those obtained with a simulation based method. Indeed, the global SQNR computation time for a second-order IIR filter simulated with the MATLAB's *Fixed-Point Toolbox* is around $34s$ for 100000 input samples.

6. CONCLUSION

A new methodology for computing the output SQNR of an application based on an analytical approach has been presented. More precisely, the modelization of the system at the quantization noise level and the expression of the output noise power have been detailed for linear systems. For non-recursive and non-linear systems a method based on the propagation of the statistical parameters of the noise through the SFG of the system is used.

This method provides a significant improvement compared to the simulation based methods for evaluating the precision of most of the DSP applications. It allows a more efficient design-space exploration for hardware (HW) and software (SW) implementation. In SW implementation, the execution time overhead due to the cast operations is minimized as long as the precision constraint is fulfilled. In HW implementation, the time required for minimizing the data word-length is considerably lower with our method. After the determination of the SQNR analytical expression, the word-lengths of the data are obtained by minimizing the size of the chip as long as the SQNR is greater than the desired SQNR. Moreover, supplementary constraints like the maximum deviation of the frequency response in the case of linear filters can be added.

7. REFERENCES

[1] T. Grötker, E. Multhaup, and O.Mauss, "Evaluation of HW/SW Tradeoffs Using Behavioral Synthesis," in *ICSPAT'96*, Oct. 1996.

[2] S. Kim, K. Kum, and W. Sung, "Fixed-Point Optimization Utility for C and C++ Based Digital Signal Processing Programs," *IEEE Transactions on Circuits and Systems II*, vol. 45, no. 11, Nov. 1998.

[3] H. Keding, M. Willems, M. Coors, and H. Meyr, "FRIDGE: A Fixed-Point Design And Simulation Environment," in *DATE'98*, 1998.

[4] L. De Coster, M. Ade, R. Lauwereins, and J.A. Peperstraete, "Code Generation for Compiled Bit-True Simulation of DSP Applications," in *Proceedings of ISSS'98*, Taiwan, Dec. 1998.

[5] Synopsys, *Converting ANSI-C into Fixed-Point using CoCentric Fixe-Point Designer*, Synopsys Inc., April 2000.

[6] B. Liu, "Effect of Finite Word Length on the Accuracy of Digital Filters - A Review," *IEEE Transaction on Circuit Theory*, vol. 18, no. 6, November 1971.

[7] J. Toureilles, C. Nouet, and E. Martin, "A Study on Discrete wavelet transform implementation for a high level synthesis tool," in *EUSIPCO'98*, Rhodes, Greece, Septembre 1998.

[8] B. Widrow, "Statistical Analysis of Amplitude Quantized Sampled-Data Systems," *Trans. AIEE, Part. II:Applications and Industry*, vol. 79, pp. 555–568, 1960.

[9] C. Barnes, B. N. Tran, and S. Leung, "On the Statistics of Fixed-Point Roundoff Error," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 3, 1985.

[10] G. Constantinides, P. Cheung, and W. Luk, "Truncation Noise in Fixed-Point SFGs," *IEE Electronics Letters*, vol. 35, no. 23, Nov. 1999.