QUALITY MEASUREMENT OF A COLORED GAUSSIAN NOISE GENERATOR HARDWARE IMPLEMENTATION BASED ON STATISTICAL PROPERTIES

Delphine DERRIEN
LESTER
Université de Bretagne Sud
56321 Lorient Cedex, France
delphine.derrien@univ-ubs.fr

Emmanuel BOUTILLON
LESTER
Université de Bretagne Sud
56321 Lorient Cedex, France
emmanuel.boutillon@univ-ubs.fr

Abstract

A Colored Gaussian Noise Generator (CGNG) adapted to hardware implementation in FPGA circuit is developed for the mobile communication channel emulation. The CGNG is done with a classical ARMA filter excited by a White Gaussian Noise. The MA filter and the 2nd order AR filter are studied in this work. The statistical properties, i.e. probability density function and power spectral density, of the fixed precision hardware design are mathematically defined. The CGNG can be used for the hardware emulation of the Rayleigh Fading channel. An example of CGNG with the Clarke filter that emulates the mobile communication channel is given.

Key words: channel, Gaussian, Rayleigh, FPGA.

1 INTRODUCTION

The design of a digital system for a communication application (error control coding, demodulation) is a very complex task requiring often trade-off between the complexity and the performances. In the ideal case, the formal expression of the Bit Error Rate (BER) can generally be expressed [7] and used to predict the performance of the system. But, in practice, the non-linearity of the system (fixed precision implementation) and/or the choice of a sub-optimal algorithm lead to a formal expression of the BER, which is too complex to derive. In that case, BER is evaluated using a Monte-Carlo simulation. The real system is emulated with an exact software model of the transmission system (transmitter, channel and receiver) and its statistical behavior is estimated by a software emulation of the transmission of thousands of bits. Monte-Carlo simulations are easy to set-up but they are time consuming. For example, 10^9 calculation iterations are needed to get an accurate (3.3%) estimation of a BER around 10^{-6}.

To overcome this problem, some authors propose to replace software emulation by hardware emulation (using a FPGA circuit) in order to speed-up the simulation by a few orders of magnitude [1,2,3,4,6]. Compared to a software compilation, this method is less flexible since each modification of the system requires the synthesis of the design from a Register Transfer Level (RTL) model and the place&route operations on the FPGA. But, once this is done, the simulation can run at a very high speed and an accurate BER evaluation can be obtained.

In previous work [3], a high quality White Gaussian Noise Generator (WGNG) is described. This WGNG allows the emulation, in a FPGA circuit, of the Additive White Gaussian Noise channel (AWGN). The characteristics of this WGNG are high accuracy, high speed and low cost. In fact, the samples are coded between –8 and +8 with a step of 1/64. The WGNG has a "(4σ, 1%) normal-like probability density function (p.d.f.)", i.e. the absolute value of the relative error $\xi_X(x)$ defined as:

$$\xi_X(x) = \frac{X(x) - N(0,1)(x)}{N(0,1)(x)}, \qquad (1)$$

between the p.d.f. of X and the normal distribution $N(0,1)$ –with mean 0 and standard deviation σ =1– is less than 1% for all $|x|<4\sigma$. For implementation with the FLEX10K100EQC240-1 of Altera, an output rate of 25 MHz with a complexity of 437 LCELL is reported [3] (information on this WGNG are freely available on a dedicated website [5]).

In this paper, the work made on the WGNG is extended to the Colored Gaussian Noise Generator. The objective of the CGNG described in the paper is to emulate the Rayleigh Fading Channel (RFC) as the module of independent phase and quadrature CGNG [7]. This point is developed in [7] and is not developed further in the paper.

The same approach as the one used for the WGNG is used again for the "high quality" CGNG, i.e. a base-band approach, a FPGA implementation and, moreover, an exact characterization of the performances of the CGNG in terms of probability density function (p.d.f) and power spectral density (p.s.d.).

The paper is organized in four sections. Section 2 analyses the evolution of the p.d.f. in the case of a Motion Average (MA) Clarke filter and in the case of a 2nd order Auto Regressive (AR) Clarke filter. Section 3 studies the evolution of the theoretical and exact p.s.d. according to the 2nd order AR filter coefficients and to the dynamic of the input gaussian distribution using the Clarke filter.

Finally, some conclusions and perspectives are given in section 4.

2 COMPUTATION OF THE CGNG P.D.F.

A fixed-point format is used for filter processing to speed up hardware simulation. Hence, rounding factor, saturation and/or overflow modifies the exact p.d.f. of the system. Moreover, the input gaussian distribution generated by the WGNG is not ideal [2]. The aim of this section is to define the tools needed for the computation of the exact p.d.f. of the Colored Noise generated, in order to compare it with the real normal law p.d.f. with (1).

To do so, all the elementary arithmetic operators are specified both in terms of arithmetic, but also in terms of operator which modifies the p.d.f. of a random variable (r.v.). An example of application of this method is given in section 2.2 for a MA filter and in section 2.3 for an AR filter.

2.1 The p.d.f. evolution in arithmetic operators

Addition operator.
Let $Z = X + Y$ be the sum of the two r.v. X and Y. The distribution P_Z of the r.v. Z is given by:

$$Z = X + Y \rightarrow P_Z(Z) = P_X \oplus P_Z = \sum_{u \in Z} P_X(u).P_Y(z-u). \quad (2)$$

Multiplication operator.
Consider a scalar a coded on N_a bits and a r.v. X coded on N bits (taking its value between $[-2^{N-1}+1,\ 2^{N-1}-1]$), the r.v. Z resulting of the multiplication of X and a will be represented with $N+N_a-1$ bits. For reducing this dilatation effect, a rescaling is performed after the multiplication, in order to suppress the N_a-1 least significant bits of the result. This rescaling preserves the distribution symmetry and can be associated with a saturation factor (not treated in the paper). This rescaling is given by:

$$z = a \times x = \left\lfloor \frac{a.x + 2^{N_a - 1}}{2^{N_a}} - s \right\rfloor, \quad (3)$$

where $\lfloor x \rfloor$ is the greatest integer lower than x and s the sign bit of $a \times x$ (s is equal to zero if $a \times x$ is positive, to one elsewhere). The p.d.f of the r.v. Z is given by:

$$P_Z = a \otimes P_X \rightarrow P_Z(z) = \sum_{\{x \in Z \,/\, a \times x = z\}} P_X(x). \quad (4)$$

2.2 Case of a 2nd order MA filter

To illustrate the method, let us consider the 2nd order MA filter shown in Figure 1. Let us assume that all the r.v. are coded on 3 bits (1 for the sign, 2 for the value). Note that the distributions resulting of the arithmetic operators are symmetrical, i.e. *f(x)=-f(-x)*.

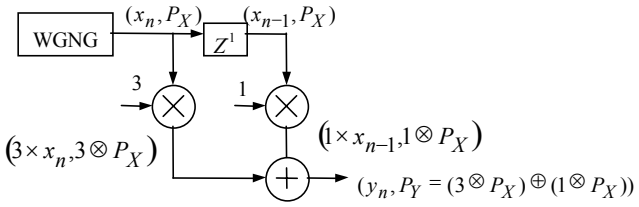

Figure 1: Example of MA filter

Table 1 gives the evolution of the r.v. distributions in the Figure 1 for the input distribution P_X.

Variable	Distribution	X=0	X=1 X=-1	X=2 X=-2	X=3 X=-3	X=4 X=-4
x_n	P_X	0.3	0.15	0.1	0.05	0.05
3* x_n	$3 \otimes P_X$	0.3	0.15	0.15	0.05	0
1* x_{n-1}	$1 \otimes P_X$	0.6	0.2	0	0	0
y_n	P_Y	0.24	0.18	0.13	0.06	0.01

Table 1: Distribution of the r.v. of the Figure 1

The computation of the exact p.d.f. in the output of a MA filter of order n is straightforward since all the input samples are uncorrelated.

2.3 Case of a 2nd order AR filter

The case of a 2nd ordre AR filter (see Figure 2) is more complex since the samples Y_n are correlated.

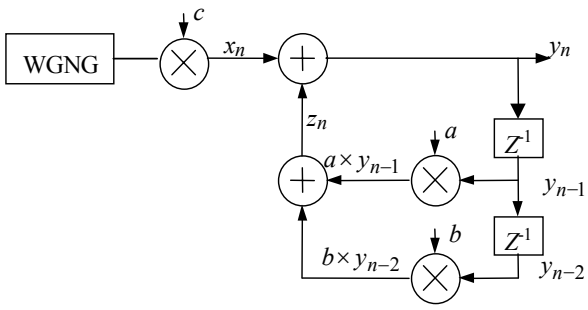

Figure 2: 2nd order AR filter structure

An original method to compute the exact p.d.f. of the r.v. Y_n in the output of the 2nd order AR filter is presented in this section. This method takes into account the correlation between Y_{n-1} and Y_{n-2} by the exact computation of the bi-dimensional r.v. $Y_{n-1,n-2}=(Y_{n-1},Y_{n-2})$.
To do so, we proceed by recursion. For n=0, we have the following initial conditions:

$$P(Y_{-1} = \alpha) = \delta_0(\alpha) \quad P(Y_{-2} = \beta) = \delta_0(\beta), \quad P(Y_{-1,-2} = (\alpha, \beta)) = \delta_0(\alpha) \cdot \delta_0(\beta), \quad (5)$$

with $\delta_0(x) = 1$ if x=0, 0 elsewhere.

Then, assuming that at the rank n, $P_{Y_{n-1,n-2}}$ is known, $P_{Y_{n,n-1}}$ is computed using the AR filter structure. One can note that P_{Y_n} is obtained from $P_{Y_{n,n-1}}$ using (6):

$$P(Y_n = \gamma) = \sum_{\alpha=-\infty}^{\infty} P(Y_{n,n-1} = (\gamma, \alpha)) . \qquad (6)$$

Let us compute $P(Y_{n,n-1} = (\gamma, \alpha))$ for a given couple of values (γ, α). Since $Y_{n-1} = \alpha$ is fixed, we can deduce from $P_{Y_{n-1,n-2}}$ the law $P^{\alpha}_{Y_{n-2}}$ given by:

$$P^{\alpha}_{Y_{n-2}} = P(Y_{n-2} / Y_{n-1} = \alpha) . \qquad (7)$$

Let us estimate $P^{Y_n}_{\alpha} = P(Y_n / Y_{n-1} = \alpha)$.
In this case, we have the following distributions for the variables of Figure 2:

$$\begin{cases} a \times y_{n-1} & \rightarrow & \delta_{a \times \alpha} \\ b \times y_{n-2} & \rightarrow & b \otimes P^{\alpha}_{Y_{n-2}} \\ z_n & \rightarrow & P(Z_n | Y_{n-1} = \alpha) = \delta_{a \times \alpha} \oplus (b \otimes P^{\alpha}_{Y_{n-2}}) \\ y_n & \rightarrow & P^{Y_n}_{\alpha} = P_X \oplus P(Z_n | Y_{n-1} = \alpha) \end{cases} \qquad (8)$$

Finally, using the Bayes rules, $P(Y_{n,n-1} = (\gamma, \alpha))$ is obtained by:

$$\begin{aligned} &P(Y_{n,n-1} = (\gamma, \alpha)) \\ &= P(Y_n = \gamma / Y_{n-1} = \alpha) P(Y_{n-1} = \alpha). \end{aligned} \qquad (9)$$

Using this method for every value of γ and α, $P_{Y_{n,n-1}}$ can be computed. The iterative computation of P_{Y_n} is performed until stationary is reached, i.e.: $P_{Y_n} = P_{Y_{n+1}} = P_{Y_\infty}$. The distribution P_{Y_∞} thus obtained is then the exact p.d.f. of the r.v. Y_n in the output of the 2nd order AR filter. Figure 3 gives an example of such a convergence. The filter has a real transfer function defined by:

$$H(z) = \frac{c}{1 + a \times z^{-1} + b \times z^{-2}} . \qquad (10)$$

The coefficients of the resulting filter (see Figure 2) are defined by a=-420/256, b=210/256. An additional multiplication by c=64/256, using (4) is performed before entering the filter. These coefficients are coded with 8 bits after the dot. The results of the first 6 iterations of (8) and (9) are shown in Figure 3 where only the positive area of the distributions is shown (note that, on this figure, P_{Y_5} and P_{Y_6} are yet nearly equal). The initial input is a gaussian distribution obtained using the matlab-file available in [3] with a single accumulation. After 40 iterations, the relative variation of the distribution becomes negligible (below 10^{-12}), and thus, $P_{Y_{40}}$ is assumed to be equal to P_{Y_∞}.

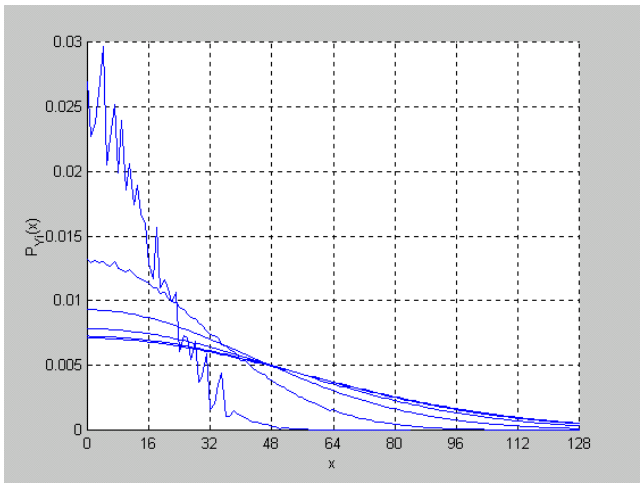

Figure 3: Distribution P_{Y_i}, i=1,...,6
(from up to down)

The distribution thus obtained has a "(4σ, 1%) normal-like p.d.f.", as shown in Figure 4.
If the dynamic of the input distribution becomes to low, then, the relative energy of truncation errors compared to input signal becomes more important and the output is no longer "(4σ, 1%) normal-like". For example, the p.d.f. obtained with c = 4/256 is given in Figure 4.

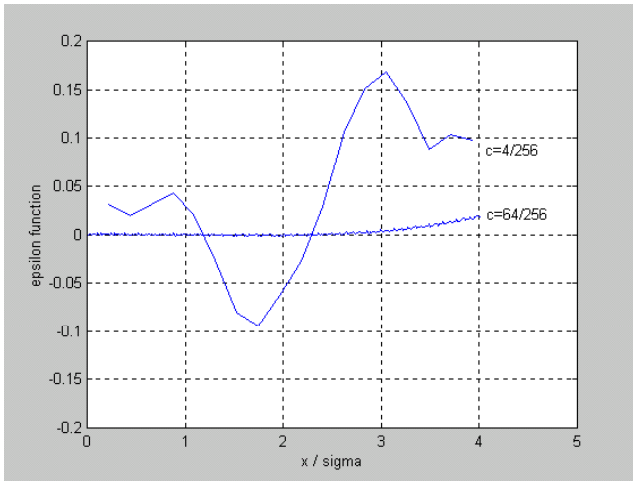

Figure 4: Epsilon function according to the factor c

This study can be generalized with different input distribution and with AR filter that takes into account saturation or overflow. The main point is that the proposed method allows the designed to guaranty a given level of quality to the CGNG p.d.f..

3 COMPUTATION OF THE CGNG P.S.D.

Finally, the exact p.s.d. issued of the CRNG is computed as the autocorrelation function $R_Y(\tau)$ Fourier Transform:

$$
\begin{aligned}
R_Y(\tau) &= E[Y_{n+\tau} Y_n] \\
&= \sum_{\chi=-\infty}^{\infty} \sum_{\xi=-\infty}^{\infty} \xi \cdot \chi \cdot P(Y_{n+\tau,n} = (\xi, \chi)) \qquad (11) \\
&= \sum_{\chi=-\infty}^{\infty} \chi \sum_{\xi=-\infty}^{\infty} \xi \cdot P(Y_{n+\tau} = \xi | Y_n = \chi) \cdot P(Y_n = \chi) \\
&= \sum_{\chi=-\infty}^{\infty} \chi \cdot E[Y_{n+\tau} | Y_n = \chi] \cdot P(Y_n = \chi),
\end{aligned}
$$

where n is the stationary rank obtained in section 2.3.

To compute $R_Y(\tau)$, we proceed by recursion:
For each value of χ we suppose $Y_n = \chi$, $P(Y_{n+m,n} = (\xi, \chi))$ is then computed recursively for m=1…τ using the same method than (6), (7) and (8) starting from the initial conditions:

$$P(Y_n = \chi) = 1 \text{ and } P_{Y_{n+1}}(\xi) = P(Y_{n+1,n} = (\xi, \chi)). \qquad (12)$$

Note that the theoretical p.s.d. is computing using (10) with the coefficients a=-420/256, b=210/256 and c. These coefficients correspond to the Doppler of a radiomobile communication [7] with the following conditions: f_c=800 MHz , v=70 KM/s and f_e=10 KHz, where f_c is the frequency of the carrier, v the speed of the mobile and f_e the sampling frequency.

Figure 5 and Figure 6 show the theoretical and the exact p.s.d. generated with the 2nd order filter applied to the WGN and with the respectively filter coefficients:
a=-420/256, b=210/256, c=4/256 and a=-420/256, b=210/256, c=64/256.
We can observe that the error between the theoretical and the exact p.s.d. depends to the factor c in the filter input (see Figure 2) and decreases when this factor increases. Moreover, from the Figure 6, we can note that the exact p.s.d. is very similar to the theoretical p.s.d. when c=64/256.

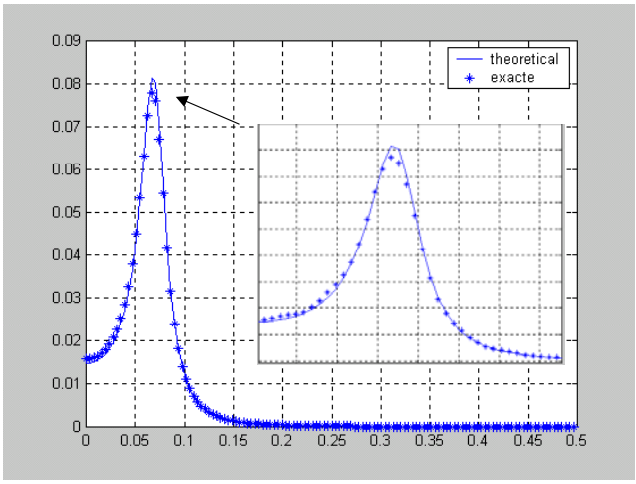

Figure 5: Theoretical and exact p.s.d with c=4/256

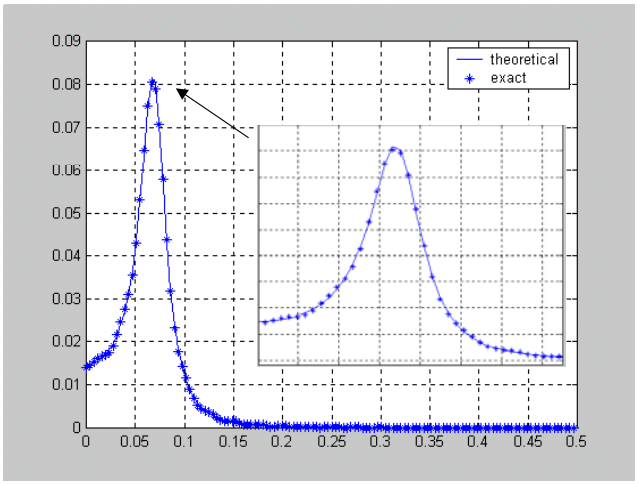

Figure 6: Theoretical and exact p.s.d. with c = 64/256

4 CONCLUSION

In this paper, tools to compute the statistical properties of a CGNG, i.e. the p.d.f. and the p.s.d., are described for the MA filter case and the 2nd order AR filter case (all the matlab files used to generate the result of this paper are freely available in [5]). Theoretically, the proposed method can be generalized to every MA filter and every 2nd order AR filter. To emulate mobile communication, we consider the 2nd order Clarke filter. According to the dynamic of the input gaussian distribution, the approximation of the distribution $N(0,1)$ has a "(4σ, 1%) normal-like" p.d.f. and a p.s.d. very similar to the theoretical p.s.d.. Hence, this design leads to very good performances and the proposed CGNG can be used for the hardware emulation of the Rayleigh Fading channel.

Our long-term perspective is to use the CRNG to emulate Multiple Input Multiple Output (MIMO) channel for new applications. The objective is to generate a reference model for various comparison or description solutions, for example in a normalization case.

5 REFERENCES

[1] R.J. Andraka, R.M. Phelps, "An FPGA based processor yields a real time high fidelity radar environment simulator", www.andraka.com/files/mapld.pdf

[2] J.R. Ball, "A real time fading simulator for mobile radio", The radio and Electronic Engineer, Vol 52, N°10, October 1982.

[3] E. Boutillon, J.L. Danger, A. Ghazel, "Design of high speed AWGN channel emulator", ICECS'2K Special Issue of the Analog, Kluwer Press, 2002.

[4] L. Feng, E. B. Nakamura, P. Joshi, J. Paik, T. Ha, G. T. Uehara, S. Lin, "An FPGA-Based Testbed for Time-Efficient Evaluation of Error-Correcting Codes in Additive Gaussian Noise", Department of Electrical Engineering University of Hawaii, www.mrc.unm.edu/symposiums/prev-symp/symp99/s8/feng

[5] "http :// lester.univ-ubs :8080/ ", → projet → WGNG

[6] M. Pätzold, R. Garcia and F. Laure, "Design of High-speed Simulation Models for Mobile Fading Channels by Using Look-Up Techniques", IEEE Transactions on Vehicular Technology, October 1999.

[7] J.G. Proakis, Digital communications, Mc GRAW-HILL International Editions, Electrical Engineering Series, 1998.