DESIGN AND PERFORMANCE ANALYSIS OF A HIGH SPEED AWGN COMMUNICATION CHANNEL EMULATOR

Prof. Emmanuel Boutillon
LESTER
South Brittany University

emmanuel.boutillon@univ-ubs.fr
1995: ENST Paris, UofT (Glenn Gulak)

2000-2001: ENST Paris (France), SUP’COM Tunis (Tunisia), LESTER, Lorient (France)
Outline

I Introduction

II Previous White Gaussian Noise Generator

III Proposed WGNG

IV Hardware architecture design

V Conclusion
Motivations

Design of a communication system...

...find the best complexity-performance trade-off
Motivation

Performance:
- BER
- Jammer rejection
- time of synchronization…
- …

Complexity:
- area, power dissipation
- time to market
- …

A very complex problem...
Monte-carlo simulation

* Formal expression of the BER: refer to Proakis

* In practice, estimation of the BER using Monte-carlo simulation

1) Software model of emitter, channel, receiver
2) Emulation of the transmission of N bits
3) Estimation of the BER as Nb_errors/N

VERY FLEXIBLE

but...

TIME CONSUMMING: BER of $10^{-6}$ (+-3%) requires $10^9$ bits.
Software simulation

Three methods to reduce the simulation time:

a) code optimization

b) powerful computing

c) parallel computing
(One Mbps for a turbo-decoder with a cluster of 16 PCs)

also use hardware emulation
Current methodology

Software

Algorithm
C programs
Compilation
Validation/optimization with long simulations
Fix specifications

Hardware

VHDL programs
Synthesis, place and route operations
Validation
Final prototype
Proposed methodology

**Software**
- Algorithm
- C programs
- Compilation
- Validation/optimization
- Fix algorithm + Set of non-specified parameters

**Hardware**
- Generic VHDL programs, IP
- Synthesis, place and route operations (on FPGA)
- Hardware simulation/validation
- Final prototype
Channel emulation

Type of communication channel:
- AWGN
- Rice
- Rayleigh
...

All those channels can be derived from Gaussian Noise (with ARMA filter, non-linear operators).

=> Need a White Gaussian Noise Generator (WGNG)
Specifications of the WGNG

Sample between -8 and 8

Output rate > 10MHz

A periodicity > $2^{60}$

$b=2$ to 10 bit after the dot

A Flat spectrum

a (4σ, 1%) normal-like p.d.f

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

$$|\xi_X(x)| < 1\% \text{ for } |x| < 4$$

+ LOW COST FPGA IMPLEMENTATION
Outline

I Introduction

II Previous White Gaussian Noise Generator

III Proposed WGNG

IV Hardware architecture design

V Conclusion
Previous WGNG

0) Using thermal noise of a resistor (non deterministic)

1) Case of low ADC precision

2) Central limit theorem

3) Box-Muller method
Case of low ADC precision (1)

The probabilities $P(x=i, y_j)$ are known for a given SNR

Example: $P(b=+1, y_3) = 0.3$  $P(b=+1, y_2) = 0.5$
$P(b=+1, y_1) = 0.15$  $P(b=+1, y_3) = 0.05$
Case of low ADC precision (2)

Repartition function: \[ S_k = \sum_{j=0}^{k} P(x = i, y = j) \]

Segment \([0,1]\)

<table>
<thead>
<tr>
<th>Segment</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(S_0 = 0.05)</td>
</tr>
<tr>
<td>0.2</td>
<td>(S_1 = 0.2)</td>
</tr>
<tr>
<td>0.8</td>
<td>(S_2 = 0.8)</td>
</tr>
<tr>
<td>1</td>
<td>(S_3 = 1)</td>
</tr>
</tbody>
</table>

Precision depends on \(q\), Complexity in \(O(qN)\)
Central limit theorem

$X$ is a real r.v. of mean $m_x$ and standard deviation $\sigma_x$,

$X_N$ defined as:

$$X_N = \frac{1}{\sigma_x \sqrt{N}} \sum_{i=0}^{N-1} (x_i - m_x)$$

tends towards $N(0,1)$, when $N$ tends towards infinity.

Let $U(q,N) = \text{sum of } N \ U^q$, (Uniform distribution over \{0,…,2^q-1\})
P.d.f. $U(q=8, N=2)$
P.d.f. of $U(q=8,N=4)$
P.d.f $U(q=8, N=8)$
Epsilon function

The convergence is very slow...
Box-Muller method

Method used in software program:

If $x_1$ and $x_2$ are two uniform r.v. over $[0,1]$, then:

$$f(x_1) = \sqrt{-\ln(x_1)}$$
$$g(x_2) = \sqrt{2} \cos(2\pi x_2)$$
$$n = f(x_1) g(x_2)$$

give a sample $n$ of the normal distribution

Efficient with a floating CPU unit, not with an FPGA
Outline

I Introduction

II Previous White Gaussian Noise Generator

III Proposed WGNG

IV Hardware architecture design

V Conclusion
Proposed method

Quantized version of Box-Muller method adapted to hardware implementation
  => rough distribution

Smooth the distribution using central limit theorem

Desire an accurate complexity model and an exact distribution
Quantization of $f(x_1)$

Plot of function $f(x_1)$

$f(x_1) = \sqrt{-\ln(x_1)}$

Need a fine quantization around 0

$f^{-1}(1) = 0.36$

$f^{-1}(2) = 1.8 \times 10^{-2}$

$f^{-1}(3) = 1.2 \times 10^{-4}$

$f^{-1}(4) = 1.1 \times 10^{-7}$
Non uniform quantization (1)

Let $s_1, s_2, \ldots, s_K$ be $K$ independent r.v. of $q$ bits (distribution $U^q$)

If $s_1 > 0$, use ROM $f_1$, else if $s_2 > 0$, use ROM $f_2$ ... and so on...

Result: the probability to draw segment $s$ of rank $r$ is $2^{-rq}$
Pre-compute values of the ROMs

The quantized value associated with the ROM $r$ at the address $s$ is:

$$f(x_1) = \sqrt{-\ln(x_1)}$$

$$f_r(s) = \left\lfloor 2^m \sqrt{\ln((s + \delta)\Delta^r)} \right\rfloor \times 2^{-m}$$

$\delta$ relative position of $x_1$ in segment $[s\Delta^r, (s+1) \Delta^r[$

Remark: Probability to draw $f_r(s)$ is $P(f_r(s)) = 2^{-rq}$
Example of quantization of $f(x_1)$

$K=3$, $q=2$, $m=2$, $\delta=1/4$

$K=3$, $q=2$, $m=2$, $\delta=1/4$

$K=3$, $q=2$, $m=2$, $\delta=1/4$
Quantization of $g(x_2)$

Let us define $s'$, a $q'$ bit random variable

$\Delta' = 2^{-q'}$ is the quantization step of segment $[0,1/4]$

ROM $g(s')$ is quantized as:

$$g(s') = \left\lfloor 2^{m'} \sqrt{2} \cos\left(\frac{\pi \Delta'(s'+\delta')}{2}\right) \right\rfloor \times 2^{-m'}$$

$\delta'$ relative position of the point in segment $[s'\Delta', (s'+1) \Delta']$

The problem of sign is analyzed later
Example of quantization of $g(x_1)$

$q' = 3, \quad m' = 3, \quad \delta' = 1/2$

Probability to draw a given point is

$P(s') = 2^{-q'}$
Half Box-Muller r.v.

For a given triple \((s,r,s')\), \(n^+\) (Half Box Muller) is computed as:

\[
n^+ = \left\lfloor \frac{f_r(s) \times g(s')}{2^{m+m-b}} \right\rfloor \times 2^{-b} \quad (*)
\]

Let \(S_n\) be the subset of \(\{0, \ldots, 2^{q-1}\} \times \{1, \ldots, K\} \times \{0, \ldots, 2^{q'-1}\}\) of all triples \((s,r,s')\) that give \(n^+\) using (*)

\[
P(f_r(s), g(s')) = 2^{-(rq+q')}
\]

\[
P(HBM = n^+) = \sum_{(s,r,s') \in S_n} P(f_r(s), g(s'))
\]

The exact probability density function of HBM can be computed
Construction of HBM

\[
\text{scaling} = \text{pow2}(m_f + m_g - b);
\]

\[
\begin{align*}
\text{for } s &= 1:\text{pow2}(q_f) - 1 \\
& \quad \text{for } r = 1:K \\
& \quad \quad \text{for } u = 1:\text{pow2}(q_g) \\
& \quad \quad \quad n = \text{floor}\left(\frac{(\text{rom}_f(s,r) \times \text{rom}_g(u))}{\text{scaling}}\right); \\
& \quad \quad \text{HBM}(n+1) = \text{HBM}(n+1) + \text{pow2}\left(- (r \times q_f + q_g)\right); \\
& \quad \text{end;}
\end{align*}
\]

Exhaustive exploration

Probability of the triplet \(s,r,u\)
From a binary r.v. \textit{sign}, Box-Muller p.d.f. is obtained

\[ n = (1-2\text{sign}) \ n^+\text{-sign} \]

The exact p.d.f. of \textit{BM} can also be computed
Example of distribution

Parameters:
- \( b = 6 \) bits after dot
- \( K = 5 \) \( f_r \) ROMs
- \( q = 4 \) (16 words ROM for \( f_r \))
- \( q' = 8 \) (256 words ROM for \( g \))
- \( m = 7 \) (3+m=10 bit-word for \( f_r \))
- \( m' = 6 \) (1+m’=7 bit-word for \( g \))
- \( \delta = 0.36 \)
- \( \delta' = 0.5 \)

Complexity:
- 5 ROMs 16x10 for \( f_r \)
- 1 ROMs 256x7 for \( g \)
- \( 5 \times 4 + 8 + 1 = 29 \) binary r.v.
- 10 bits x 7 bits multiplier
Large variations around $N(0,1)$ due to quantization effects
Epsilon function

Need to smooth the variation with central limit theorem
Use of central limit theorem

Generation of $BM_2$
as the sum of
two independent
draws of $BM_1$

Distribution of $BM_2$ can be computed ($BM_2=BM_1 \otimes BM_1$)
Use of central limit theorem

$BM_2$ is much better than $BM_1$, but still not $(4\sigma, 1\%)$ normal-like p.d.f ...

...thus, use central limit theorem again
The p.d.f. of BM4 is (4\sigma, 1\%) normal-like
Performance results

Maximum relative error $\xi_X(x)$ between the ideal gaussian distribution and $BM_A$

<table>
<thead>
<tr>
<th>$b$</th>
<th>$\delta=0.44$</th>
<th>$\delta=0.453$</th>
<th>$\delta=0.445$</th>
<th>$\delta=0.467$</th>
<th>$\delta=0.467$</th>
<th>$\delta=0.467$</th>
<th>$\delta=0.467$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.65</td>
<td>11.5</td>
<td>20.2</td>
<td>64.6</td>
<td>57.3</td>
<td>71.9</td>
<td>237</td>
</tr>
<tr>
<td>2</td>
<td>0.08</td>
<td>1.96</td>
<td>2.12</td>
<td>5.4</td>
<td>5.4</td>
<td>5.8</td>
<td>8.4</td>
</tr>
<tr>
<td>3</td>
<td>0.15</td>
<td>0.93</td>
<td>0.56</td>
<td>0.71</td>
<td>1.12</td>
<td>1.38</td>
<td>0.68</td>
</tr>
<tr>
<td>4</td>
<td>0.29</td>
<td>0.43</td>
<td>0.34</td>
<td>0.31</td>
<td>0.69</td>
<td>0.93</td>
<td>0.28</td>
</tr>
<tr>
<td>5</td>
<td>0.43</td>
<td>0.34</td>
<td>0.31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.69</td>
<td>0.93</td>
<td>0.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.12</td>
<td>1.38</td>
<td>0.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.69</td>
<td>0.93</td>
<td>0.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$b$ : number of bits after decimal point

$A$ : number of accumulations

$q=4$, $K=5$, $q'=8$

The quality can be controlled with MATLAB
Outline

I Introduction

II Previous White Gaussian Noise Generator

III Proposed WGNG

IV Hardware architecture design

V Conclusion
Global architecture

Architecture complexity = f(parameters)
Generation of binary variables

A LFSR of length \( l \) can generate a binary “random like” sequence of periodicity \( 2^l - 1 \)

\[
\begin{align*}
&\ \ \ \ D \ \ \ \ Q \ \ \ \ X \ \ \ \ D \ \ \ \ Q \ \ \ \ X^2 \ + \ D \ \ \ \ Q \ \ \ \ X^3 \ + \ D \ \ \ \ Q \ \ \ X^4 \ + \ D \ \ \ \ Q \ X^5
\end{align*}
\]

\( X^n \mod P[X] \)

The periodicity of all LFSRs should be relatively prime in order to maximize the periodicity of the WGNG

\[\Rightarrow \text{Choice of } l \text{ so that } 2^l - 1 \text{ is a prime number} \]
Optimization for FPGA

LCELL of the FPGA:

=> $q=4$, in order to use LCELL for ROMs $f_r$

=> Use LFSR performing $X^{4n} \mod P[X]$ instead of $X^n \mod P[X]$:
   - 4 bits generated per cycle instead of 1 bit
   - Same hardware complexity
Synthesis results

Parameters

\[
\begin{align*}
A &= 4 \\
b &= 6 \\
\text{LFSR length} &= 22, 21, 20, 17, 13, 7, 15 \ (G, \ Fr \ and \ sign)
\end{align*}
\]

<table>
<thead>
<tr>
<th>FPGA device</th>
<th>cells</th>
<th>memory block</th>
<th>clock rate</th>
<th>Output rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10K100AR C240-1</td>
<td>434</td>
<td>1</td>
<td>74MHz</td>
<td>18.5MHz</td>
</tr>
<tr>
<td>10K100EQ C240-1</td>
<td>437</td>
<td>0.5</td>
<td>98MHz</td>
<td>24.5MHz</td>
</tr>
</tbody>
</table>

Less than 10\% of FLEX10K100 resources
Experimental results

Theoretical distribution = measured distribution
Outline

I Introduction

II Previous White Gaussian Noise Generator

III Proposed WGNG

IV Hardware architecture design

V Conclusion
Conclusion

Parameterizable low complexity WGNG

Quality can be fixed

Undergoing work to extend WGNG to Rayleigh Noise generator