

Génération de modèles de bruit en $1 / f^n$ pour la simulation des systèmes électroniques

$1 / f^n$ noise models generation for electronic systems simulation

N. Lewis, G. Monnerie, L. Lewis, J. Sabatier, P. Melchior

noelle.lewis@ixl.fr

IXL – Bordeaux University
LAPS – Bordeaux University



Plan

- Contexte et problématique
- Principe de modélisation
- Implémentation
- Applications
- Conclusion et améliorations

Outline

- Context and problem
- Modeling principle
- Implementation
- Applications
- Conclusion and improvements

Contexte



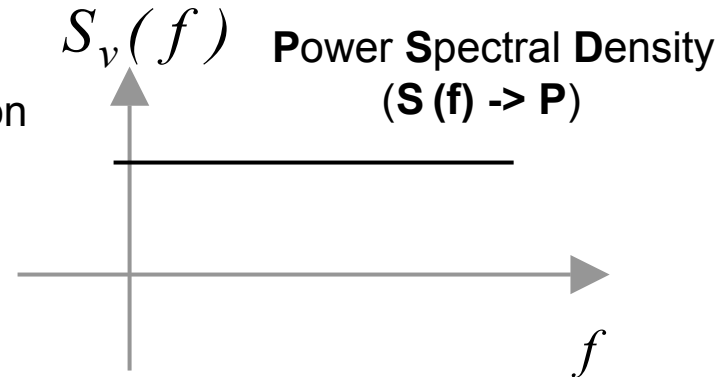
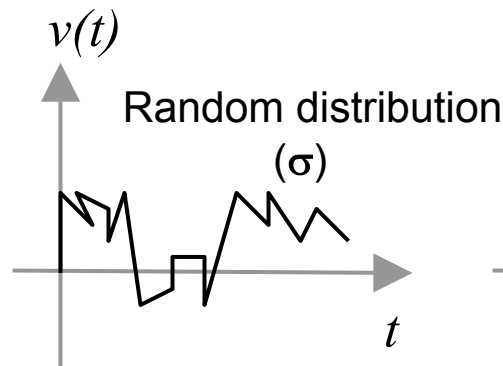
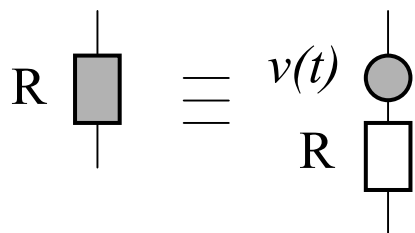
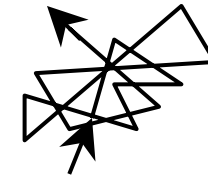
- Collaboration IXL / EADS – Telecom – St Quentin Yvelines
 - Private Mobile Radio
 - Point de départ : dégradation SNR mesure / simulation
- Thèse CIFRE Guillaume MONNERIE
 - « Etude et modélisation de sources de bruit dans des structures à temps discret »
- Objectif principal
 - Prévoir l'impact de différentes sources de bruit sur les performances globales du système
- Objectif intermédiaire
 - Se doter de modèles **simulables** représentant les différentes sources de bruit rencontrées

Problem

- HF communication systems :
 - are very sensitive to noise
 - External or internal
 - Amplitude noise, phase noise, jitter
 - often use discrete-time solutions
 - Ex : CAN stage – Mixed-Signal circuit
- Simulation of noise sensitivity in current CAD tools?
 - SPICE-like simulators
 - Noise parameters in transistor or passive elements models \Rightarrow transistor level description
 - Noise analysis is linear (AC domain)
 - RF simulators
 - Specific algorithms for noise simulation in non-linear circuits
 - Can't directly simulate jitter
- Requirements
 - transistor level \Leftrightarrow behavioral description of building blocks
 - Mixed-signal, discrete-time circuits are simulated in the time domain
 - VHDL-AMS language and compatible simulators
 - Development of multi-purpose noise models
 - Automatic generation of user-defined noise models

Modeling principle (1) : basic assumption

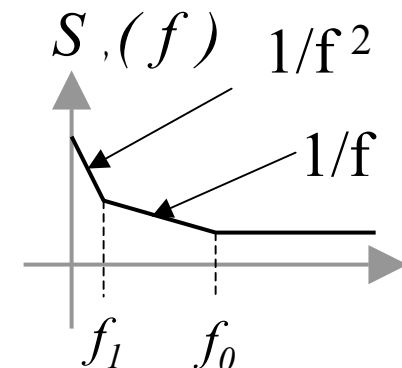
- Random motions at the microscopic level
- Macroscopic model



- Principally in electronics :
 - thermal, shot noise : constant PSD
 - flicker noise : 1/f PSD

- Electronic noises are a combination of :

$$S(f) = 1/f^n \quad n = 0, 1, 2, \dots$$

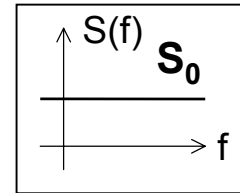


Modeling principle (2) :

$1/f^n$ noise

- $n = 0$: white noise

– Gaussian random generator \Rightarrow

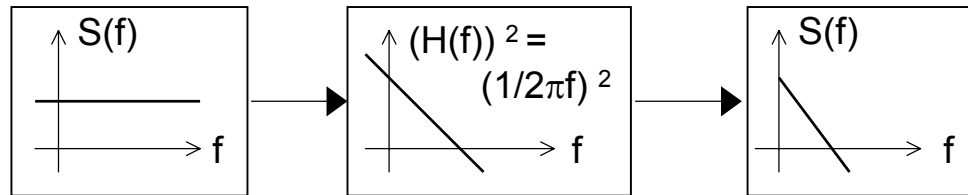


white noise source

- $n = 2, 4, 6 \dots, 2k$

– Gaussian random generator + classical integration

white noise source *1st order integrator* *1/f² noise source*

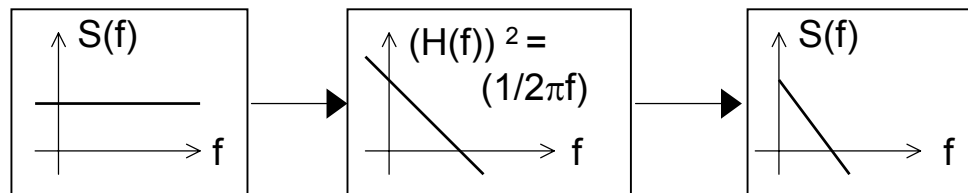


$$H(s) = \frac{1}{s}$$

- $n = 1, 3, 5, \dots, 2k+1$

– Gaussian random generator + fractional integration

white noise source *1/2 order integrator* *1/f noise source*

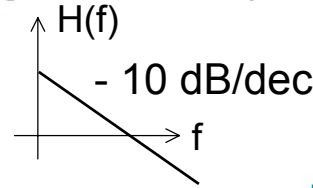


$$H(s) = \frac{1}{\sqrt{s}}$$

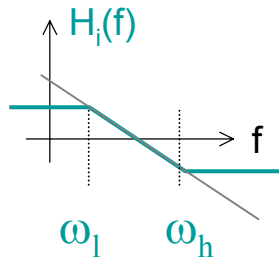
Modeling principle (3) :

1/2 order integrator syntesis

- Objective : $H(f) = 1/\sqrt{2\pi f}$
- Practically on a given frequency range $[f_{\min}, f_{\max}]$

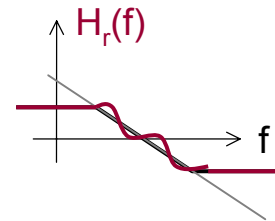


➔
$$H_i(s) = H_0 \frac{\left(1 + \frac{s}{\omega_l}\right)^{1/2}}{\left(1 + \frac{s}{\omega_h}\right)^{1/2}}$$



$$\begin{cases} \omega_h = 10 \times 2\pi f_{\max} \\ \omega_l = 2\pi f_{\min} / 10 \end{cases}$$

➔
$$H_r(s) = H_0 \frac{\prod_{i=1}^N \left(1 + \frac{s}{\omega'_i}\right)}{\prod_{i=1}^N \left(1 + \frac{s}{\omega_i}\right)}$$



$$\begin{cases} \omega'_1 = \sqrt{\gamma} \omega_h \\ \omega'_{i+1} = \gamma^2 \omega'_i \end{cases} \quad \text{with} \quad \gamma = \left(\frac{\omega_l}{\omega_h}\right)^{\frac{1}{2N}}$$

$$\begin{cases} \omega_1 = \gamma \omega'_1 \\ \omega_{i+1} = \gamma^2 \omega_i \end{cases}$$

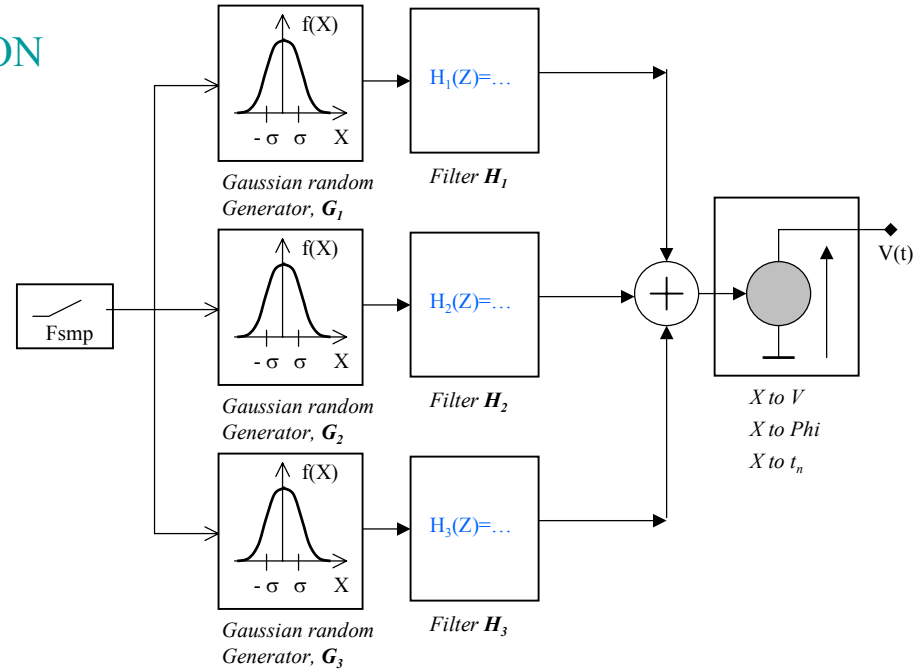
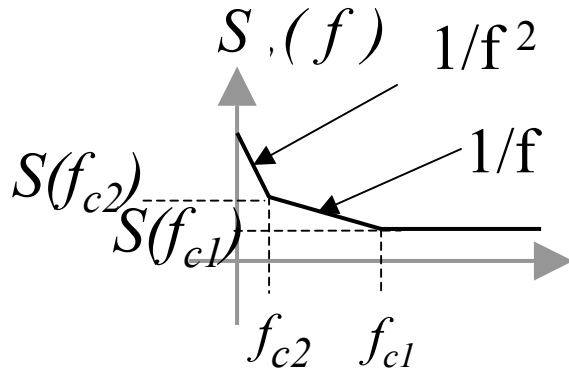
Implementation (1)

- SCILAB : scientific free software package
- Step 1 : INPUTS DEFINITION
 - Asymptotical Spectrum description ($[f_{c1}, S(f_{c1})], [f_{c2}, S(f_{c2})], \dots$)
 - f_{smp} : sampling frequency of the current application
 - $f_{\text{min}}, f_{\text{max}}$ frequency range for $\frac{1}{2}$ order integrator synthesis
 - N : number of recursive cells for $\frac{1}{2}$ order integrator synthesis
 - Type of noise : amplitude noise, phase noise voltage, clock jitter ?
- Step 2 : $\frac{1}{2}$ ORDER INTEGRATOR SYNTHESIS
 - $H_r(s)$: Calculation of $\gamma, \omega_i, \omega_i'$
 - Transformation in a sum of 1 order elements
 - Discretization : Tustin approximation

Implementation (2)

- Step 3 : MODEL CALIBRATION

- For each $1/f^n$ portion :



- Dimensionless random numbers => physical signal

Phase noise

$$x(t) = A(1 + \Delta A(t)) \sin(2\pi f_0 t + \Delta\phi(t))$$

Jitter

$$t_n = \frac{n}{f_{\text{smp}}} + \Delta t_{\text{FM}}(t) = \frac{n}{f_{\text{smp}}} + \frac{\Delta\phi(t)}{2\pi f_{\text{smp}}}$$

- Step 4 : MODEL CODE GENERATION (VHDL-AMS)

Applications (1) :

NOISY SINUSOIDAL GENERATOR

- User inputs :

$A = 14,14 \text{ V}$; $f_0 = 500 \text{ kHz}$; $fc3 = 50 \text{ kHz}$; $S_{\Phi}(fc3) = -183 \text{ dB/Hz}$;
($L_{\Phi}(fc3) = S_{\Phi}(fc3) + 3\text{dB} = -180 \text{ dBc/Hz}$) ; $fsmp = 2 \text{ MHz}$; $N = 7$

- Transient simulation with ADVanceMS

- total number of samples $N = 2^{20}$

- Post-processing : spectral analysis

- Periodogram : $L = 32$ sub-windows ; $M = 2^{15}$ samples for each

- Blackman 7 window

- Measurement results :

- $PSDeffective = PSDmeas - 10 \cdot \log(ENBW)$

Applications (2) :

NOISY SINUSOIDAL GENERATOR - results



demo

Scilab Choose

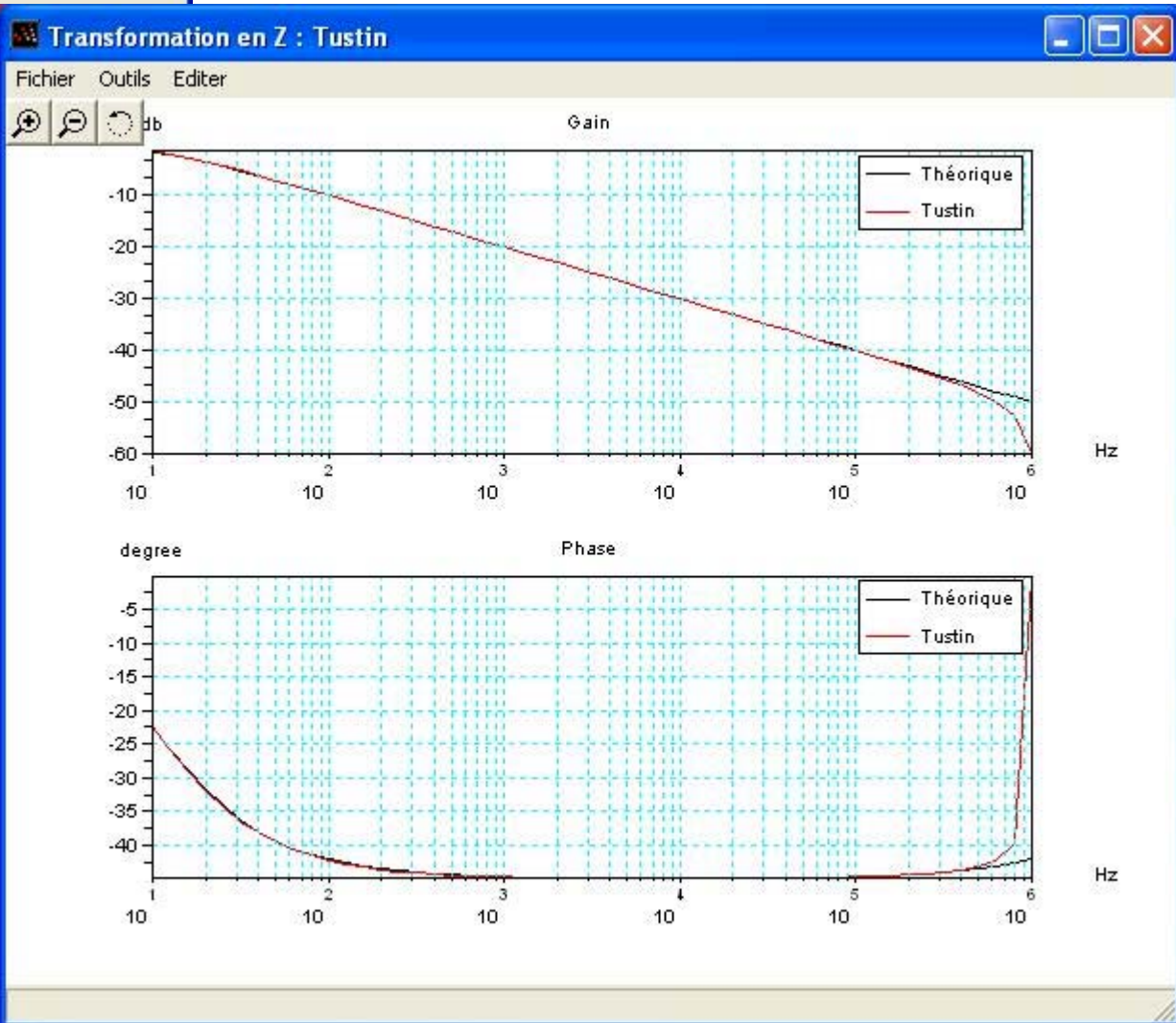
Scilab md... Transformation en Z : Tustin

Fichier Outils Editer

db

Synthès
Synthès
Synthès
Synthès
Synthès
Charger

OK



scilab-3.1.1 (0)

Fichier Edition Préférences Contrôle Ed

scil

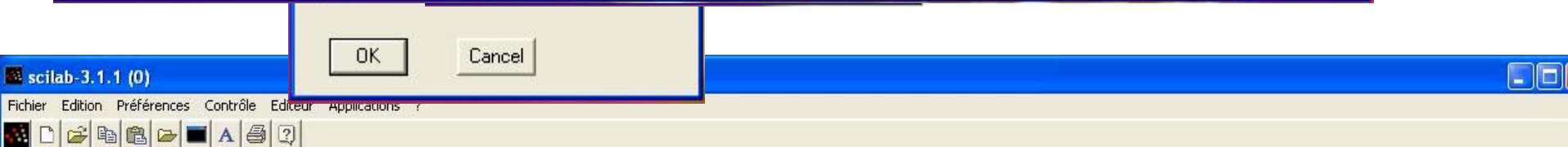
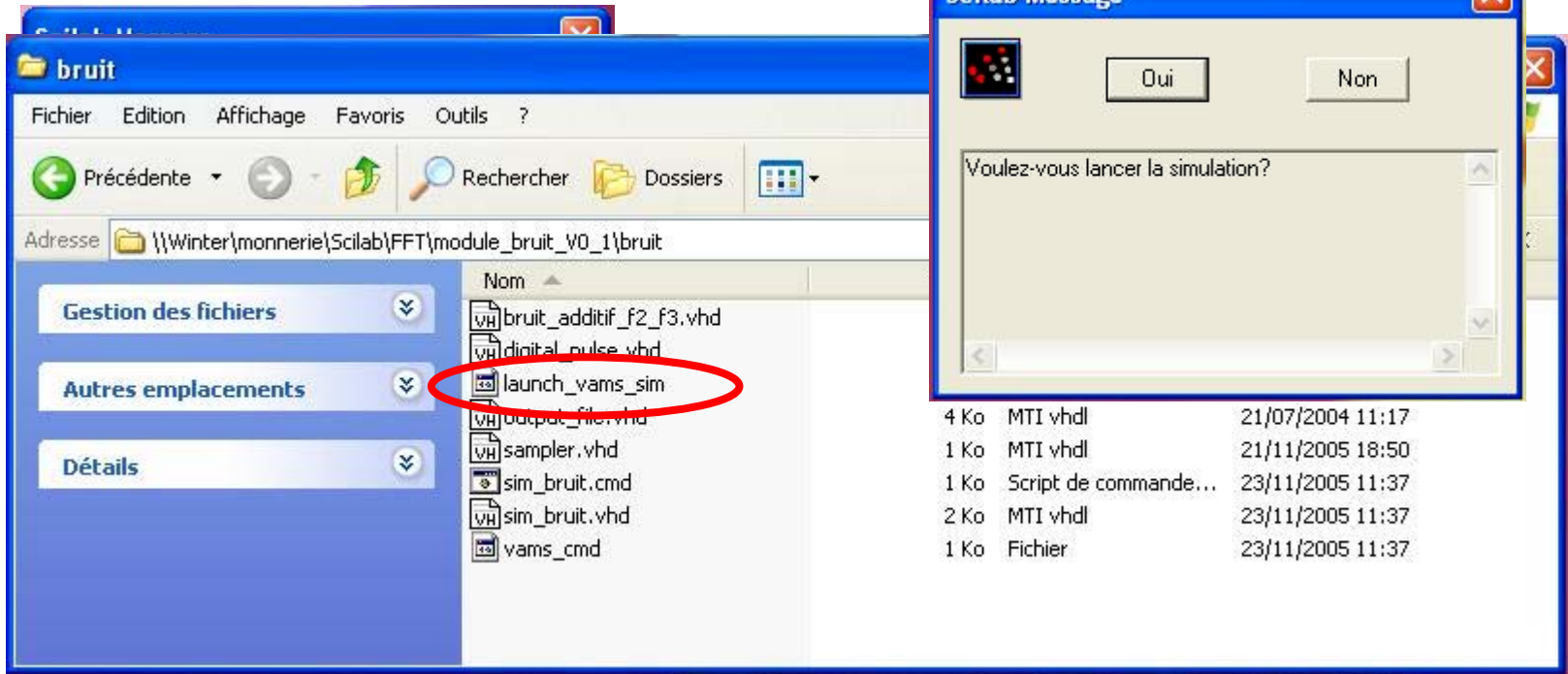
Copyright (

Consortium Scil

```
-->exec('\\\\Winter\\monnerie\\Scilab\\FF\\module_bruit_vo_1\\main.sce ');disp( exec done );
```

Tracé des diagrammes de Bode réel et théorique
Saisie des paramètres du modèle

Synthèse de l'intégrateur d'ordre 1/2
Transformation en Z : Tustin



scilab-3.1.1

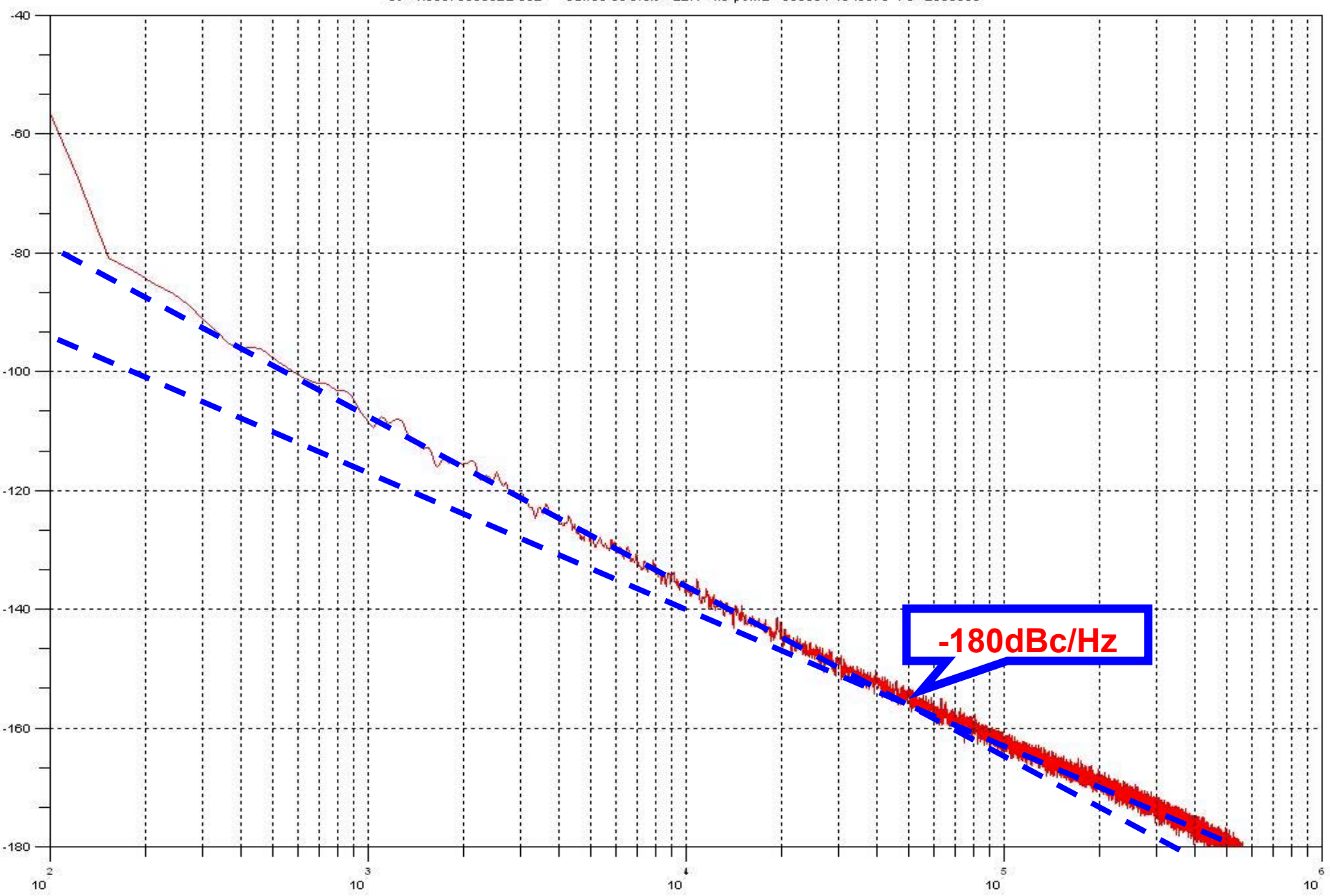
Copyright (c) 1989-2005
Consortium Scilab (INRIA, ENPC)

Ecriture des fichiers VHDL

Ecriture des fichiers de simulation



d= 1.0367080862E-002 bande de bruit= -22.1 nb points= 65536 / 1048576 Fe= 2000000



Conclusion - Améliorations

- $S(f)$: gabarit « quelconque »
- Saisie de la marge d'erreur acceptable
 - Calcul automatique de N
 - Calcul automatique du rapport K
$$\begin{cases} \omega_h = K \times 2\pi f_{\max} \\ \omega_l = 2\pi f_{\min} / K \end{cases}$$
- Implémentation du filtre - Optimisation
 - Décomposition de $H_r(f)$
 - Tustin?
 - Etc ...