

VHDL as a Mixed-Signal Simulator for RF System Simulations

IEEE RFIC Symposium 2004

Workshop "Mixed-Signal Design Methodology"

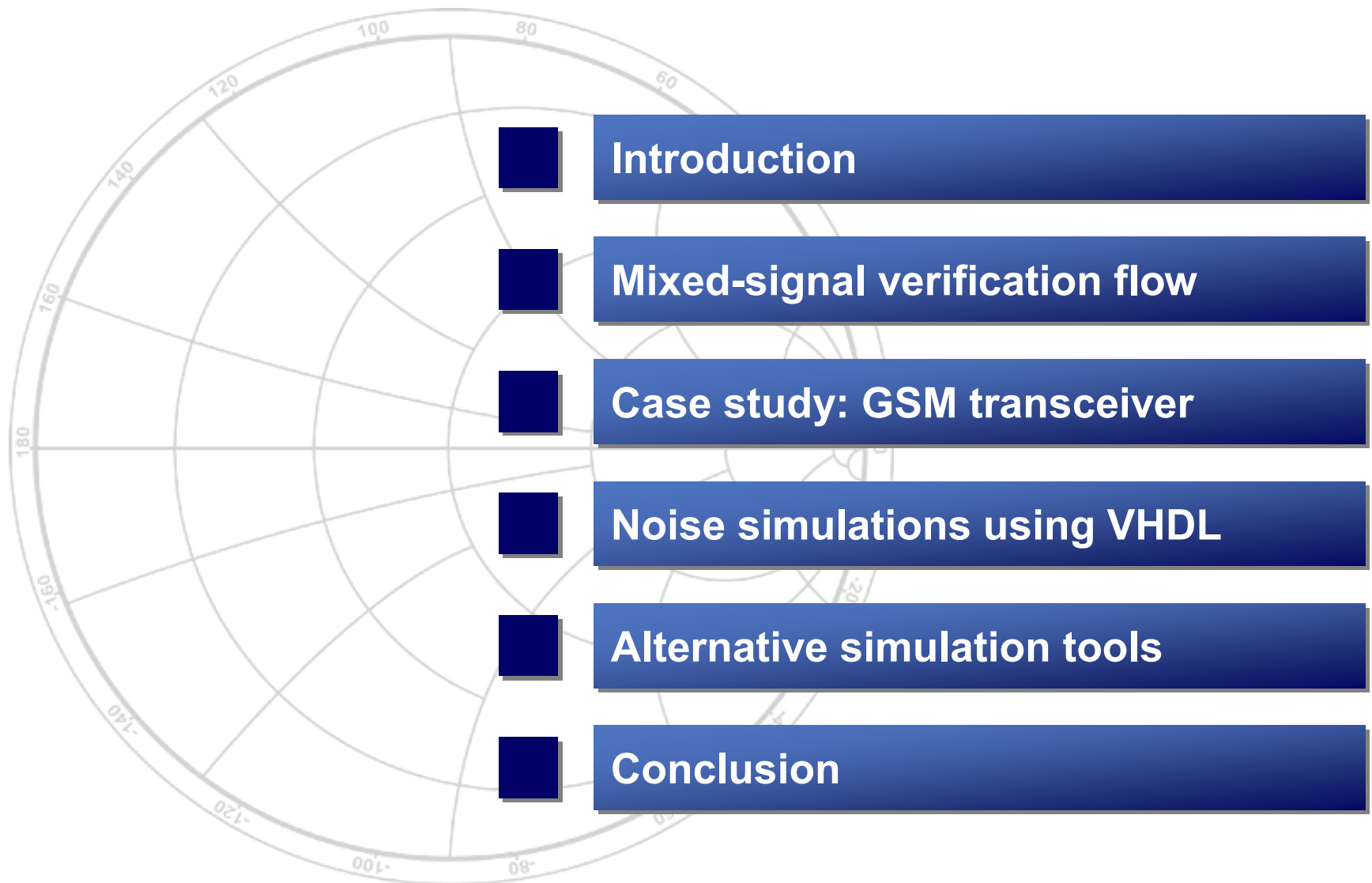
Christian Munker
Infineon Technologies AG



Never stop thinking.

Agenda

stop thinking
Never



Christian Munker

June 6, 2004

Page 2 of 31

Mixed-Signal Verification Flow (1)

- Moore's Law: „The number of possible devices on a chip doubles every 18 ... 24 months“
- Competitive pressure enforces increasing degree of system integration and algorithm complexity
- But: there is no fully hierarchical design flow for MS-Chips like in the digital design world
- Gap between concept and block level



Use digital representation for analog signals to apply tools from the digital world to analog / mixed-signal blocks!

Mixed-Signal Verification Flow (2)

Level	Example	Tools
Concept / Algorithm	Direct Conversion	Matlab, CoCentric, C++
System	GSM Transceiver	
Block / RTL	VCO (Model) DSP (Model, RTL)	VHDL / Verilog / Mixed-Signal Simulators
Gatelevel	Synthesized Netlist	
Transistor-Netlist	VCO (detailed)	SPICE
Layout		Layout-Tools
Device	RF-Transistor	Field Simulator



stop thinking
Never

Analog- vs. Digital Simulator

Analog Simulator

- Solves differential equation systems
- Solves implicit / evaluates explicit functions
- Structure driven
- Continuous-valued and time-continuous
- Time-steps defined by required precision
- Lots of analyses
- Analog post-processing

Digital Simulator

- Elaborates event queues
- Evaluates only explicit functions
- Driven by data flow / algorithms
- Discrete-time and discrete / continuous-valued
- Time-step defined by events
- Only transient simulations
- Digital post-processing

Standard VHDL for Analog Simulations?

Features

- Continuous-valued using real numbers
- Libraries for mathematical functions (even complex)

Best suited for

- Systems with clearly defined functional blocks
- Systems without (or only slow) feedback loops
- Mainly discrete-time systems

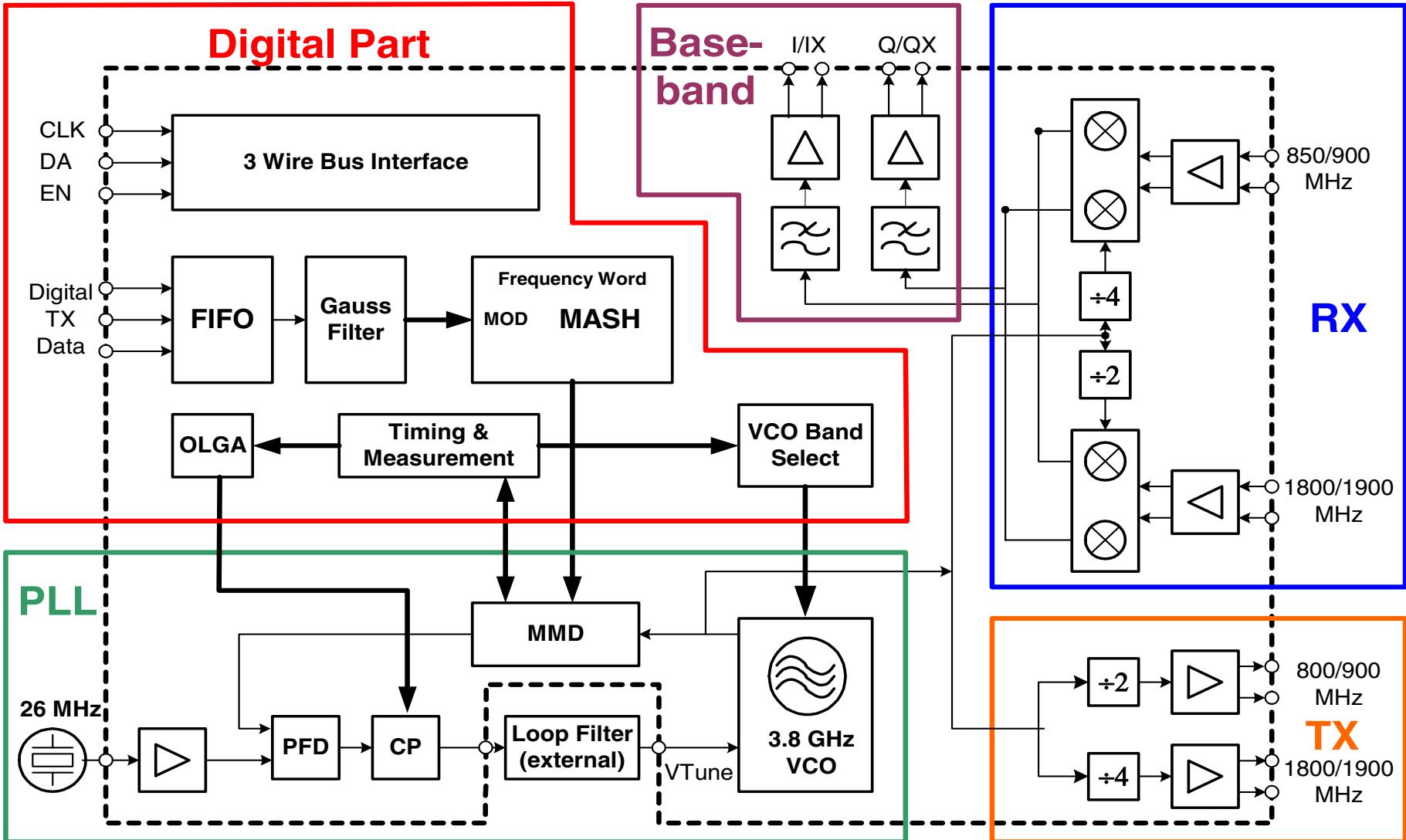
Examples

- Hard disk R/W - channel
- PLLs
- Digital calibrations

Numeric Limits of Standard VHDL

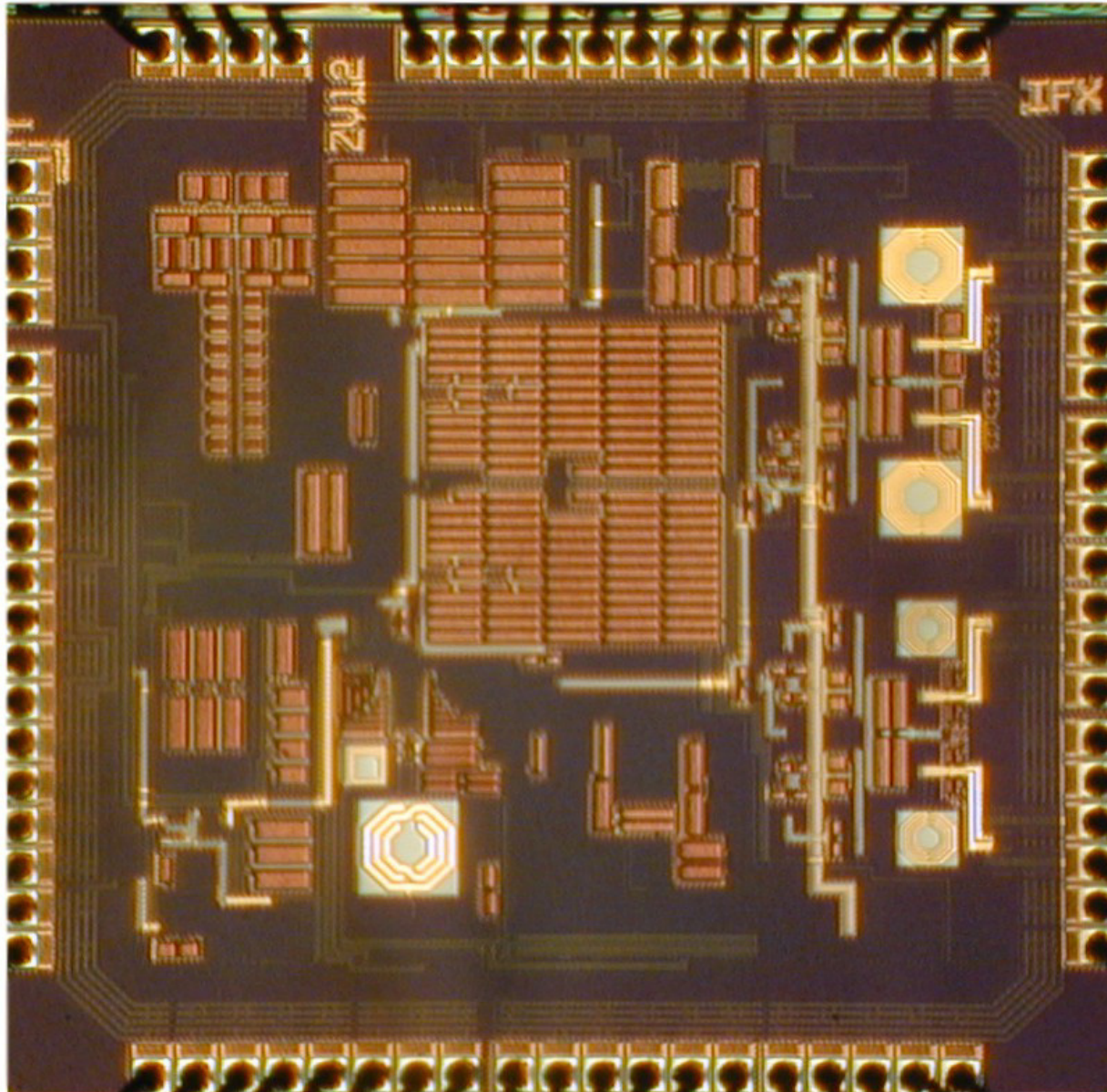
- Type **integer**: usually 32 bits wide
⇒ 0 ... $2.1 \cdot 10^9$
- Type **time** represents time / events
Min. time step: 1 fs, 64 bits range
⇒ 1 fs ... 2^{63} fs \approx 2 ½ hours
- Type **real** for „analog“ signals:
ANSI / IEEE Std., i.e. 11 bits exponent, 53 bits mantissa
⇒ $\pm 2.2 \cdot 10^{-308}$... $1.8 \cdot 10^{308}$ with $\varepsilon = 2.2 \cdot 10^{-16}$

Case Study: Quad Band GSM Transceiver



Christian Mürker

Chip Photo GSM Transceiver



Area:
5 mm²

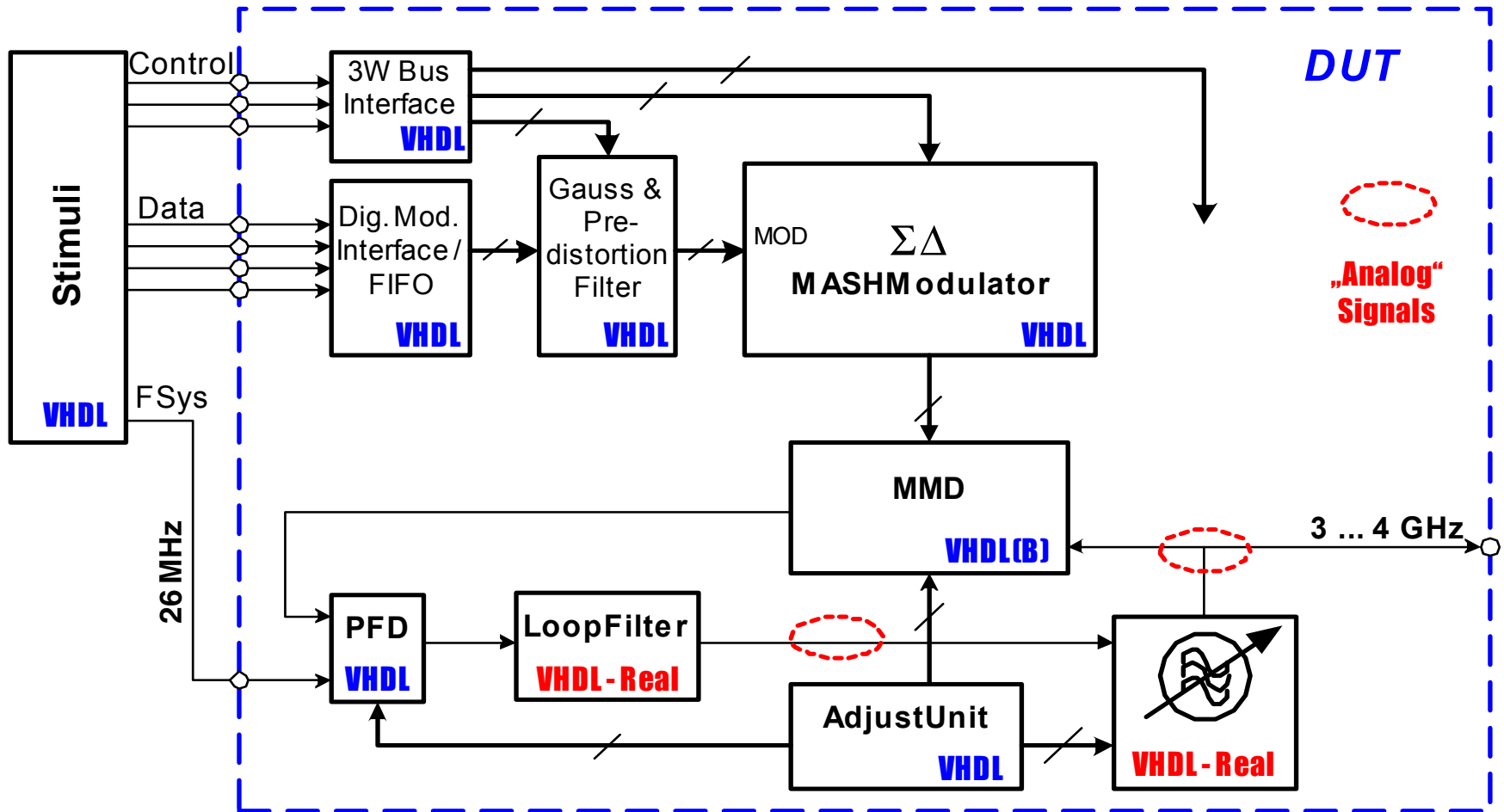
Technology:
0.13 μ m CMOS

RX:
250 mW

TX:
210 mW

Package:
48 Pin VQFN

Simulation Setup for Verification of TX Path



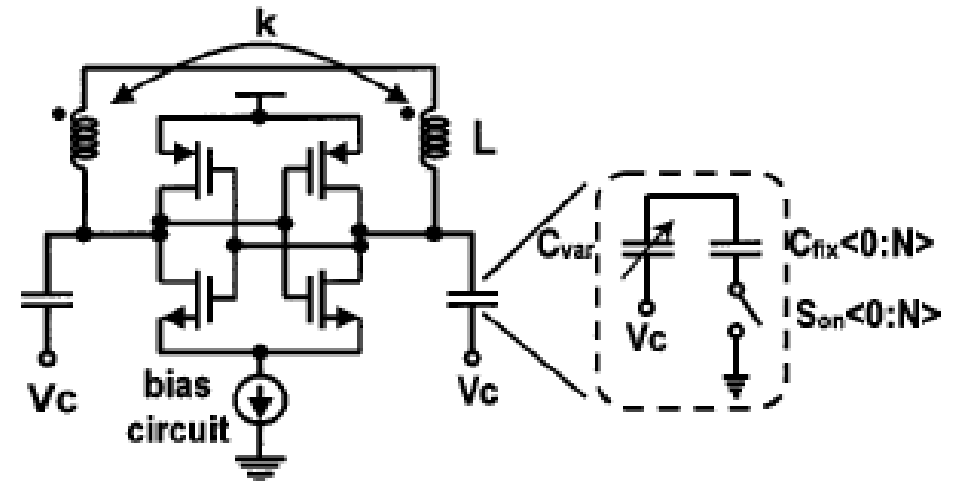
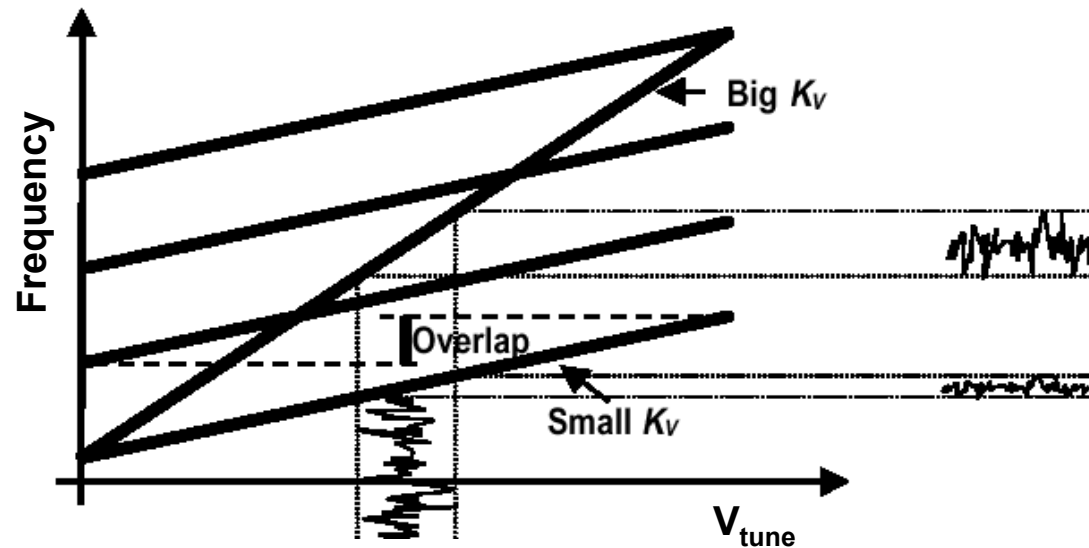
Christian Mürker

Verification problems: Digital calibration, MASH-modulator, closed loop operation, digital modulation, ...

Digitally Tunable VCO

Multiple Bands

- Large tuning range with small $K_V \Rightarrow$ insensitive to noise
 - Overlap: no gaps between bands, has to be larger than modulation
 - K_V should be constant over all bands
- \Rightarrow two calibrations needed: band selection and steepness



Example: VCO Behavioral Modeling (1)

- **Derive behavioral model from circuit principle**
 - ⇒ Calculate VCO period from circuit parameters (L, C)
 - ⇒ Calculate timing events from VCO period
- **VCOs are continuous-valued and time-continuous systems, but ...**
- **Regard only phase / frequency**
 - ⇒ approximate sinusoidal waveform by digital signal
 - ⇒ discrete-valued representation speeds up simulation
- **Update frequency only once after each period**
 - ⇒ recalculate VCO period from tuning voltage only once per period (discrete-time approximation)
 - ⇒ works only for slowly changing control voltages!

Example: VCO Behavioral Modeling (2)

```
VCOLoop : loop

    vco_out  <= transport '1' after delay_t,
                '0' after period_t/2;

    wait for period_t;

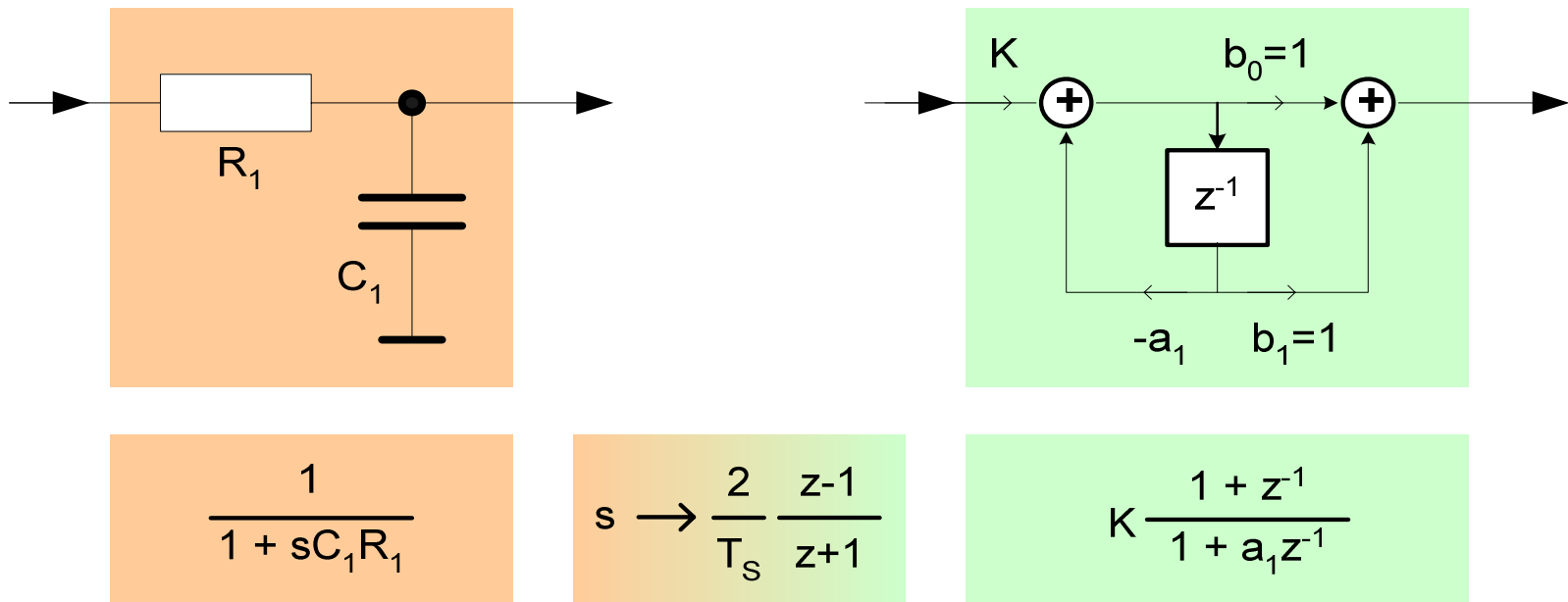
    C_tot    := (vt_max - vt) * C_var + C_fix;
    period   <= 2.0 * math_pi * sqrt(L_0 *
                C_tot) * 1.0e3;

    period_t <= period * fs;  -- period in fs
    delta_f  <= (1.0e6/period - F_TARGET * 1.0e9);

end loop VCOLoop;
```

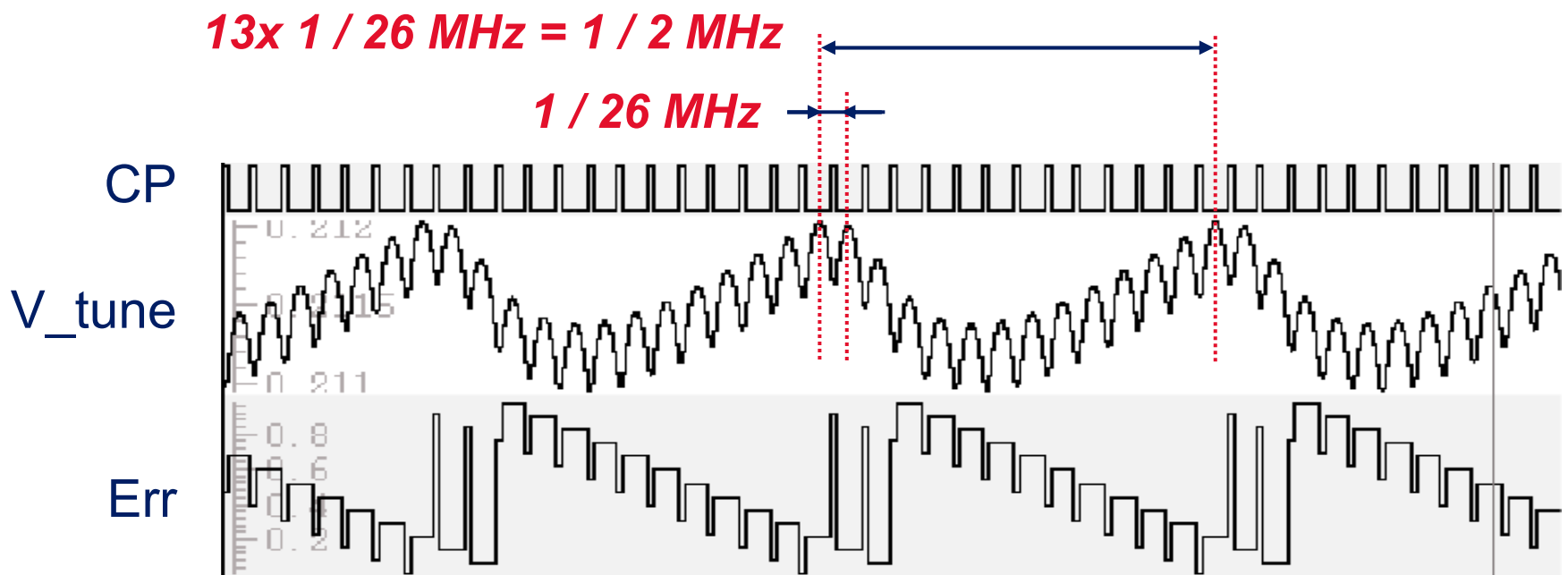
Example: Modeling of Loop Filters

- Problem: filter is time- and value-continuous
- Solution: Transform s-plane to z-plane



- Sampling rate $f_s \Rightarrow$ Average Timing Error $T_S/2$

Simulation Errors Due to Discrete-Time Loop Filter Model

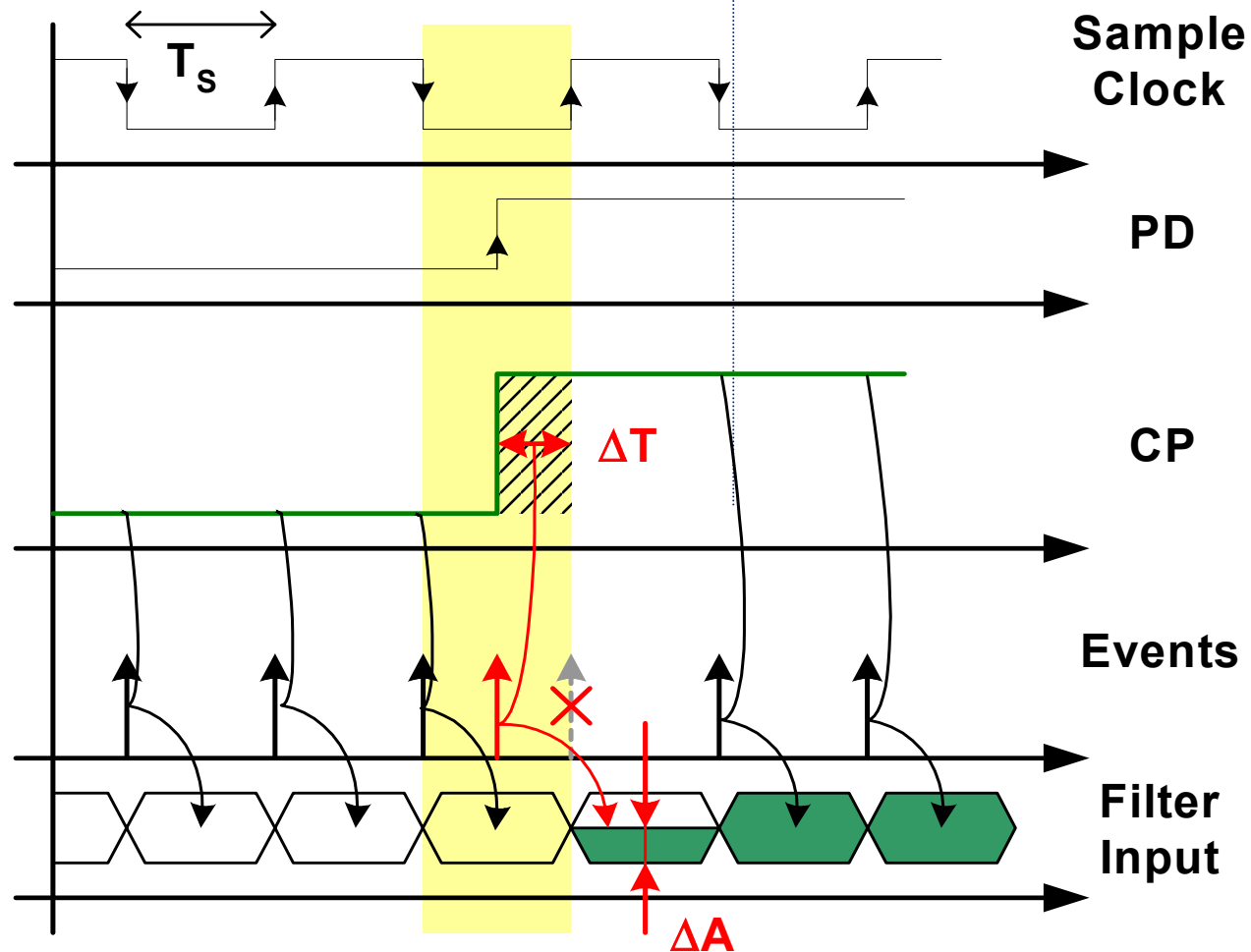


$f_{\text{ref}} = 26 \text{ MHz}, f_s = 1 \text{ GHz} \Rightarrow$ “beat frequency” at 2 MHz

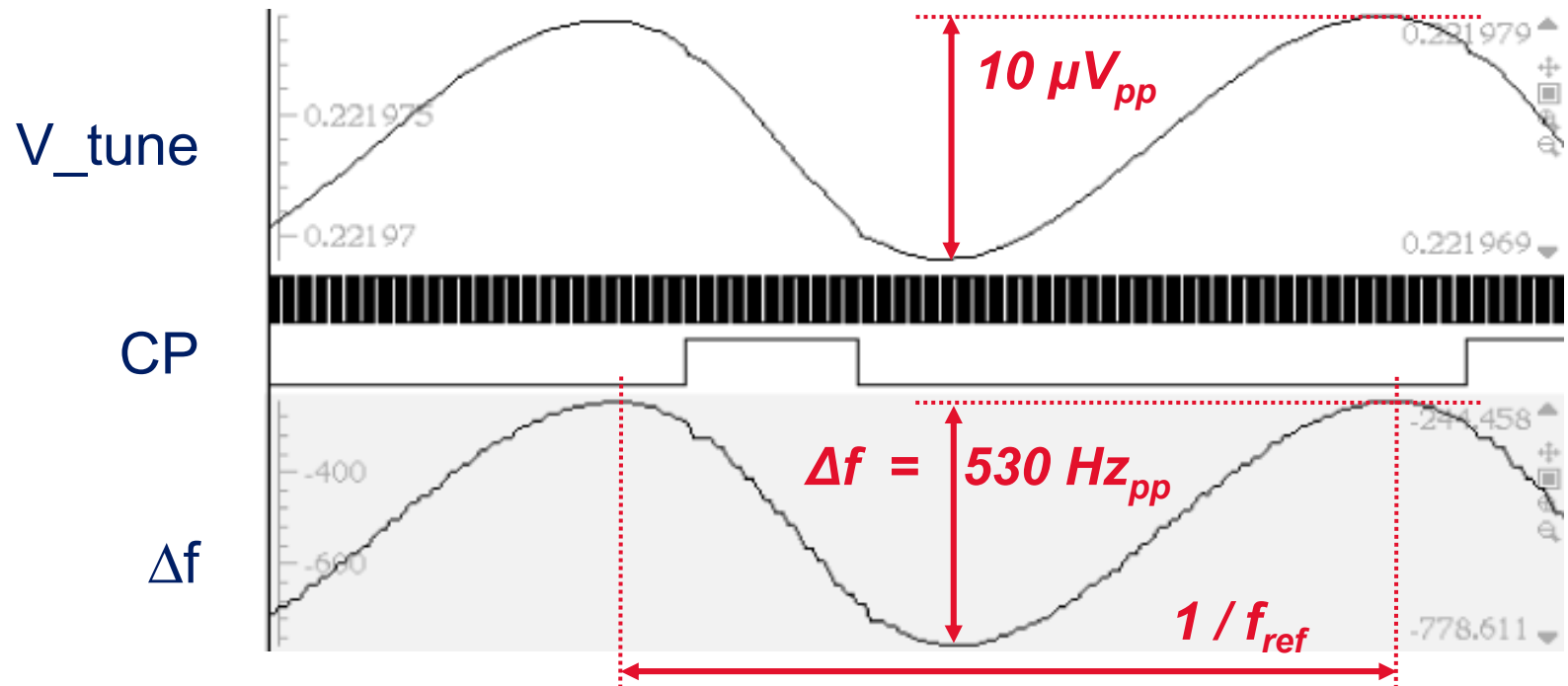
Timing errors due to sampled loop filter model lead to simulation artifacts!

Fractional Sampling in the Loop Filter Model

- Model timing error ΔT as amplitude variation ΔA to improve accuracy in spite of fixed sampling interval T_s :



Simulation: Post Loop Filter Ripple



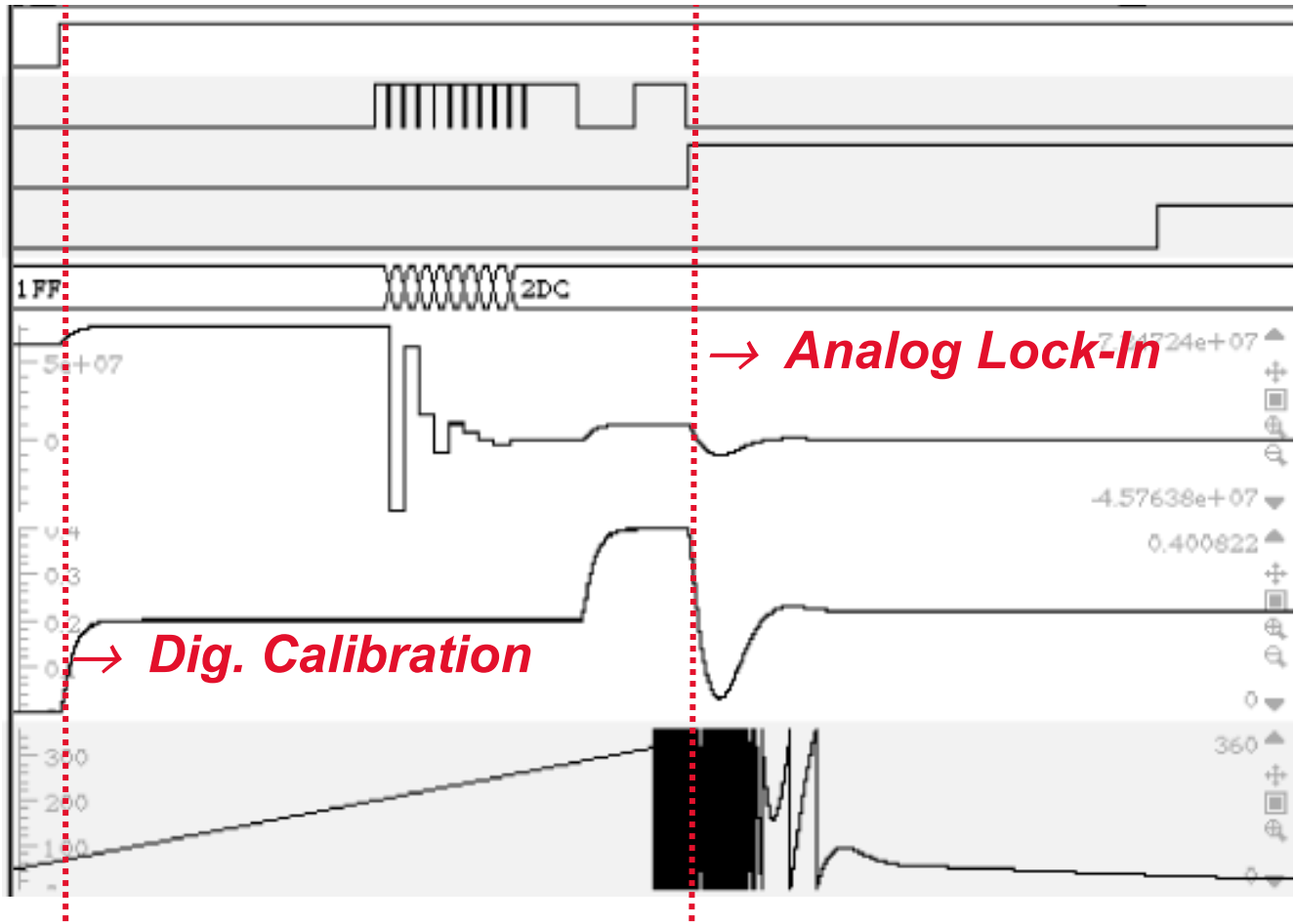
Post loop filter ripple on tune voltage causes periodic frequency error which can be seen as spurious sidebands:

$$spur = \frac{1}{2} \frac{\Delta f}{f_{ref}} = \frac{265 \text{ Hz}}{2 \cdot 26 \text{ MHz}} = 5.1 \cdot 10^{-6} \equiv -106 \text{ dBc}$$

Simulation: PLL Lock-In Process

Never stop thinking

Digital Signals

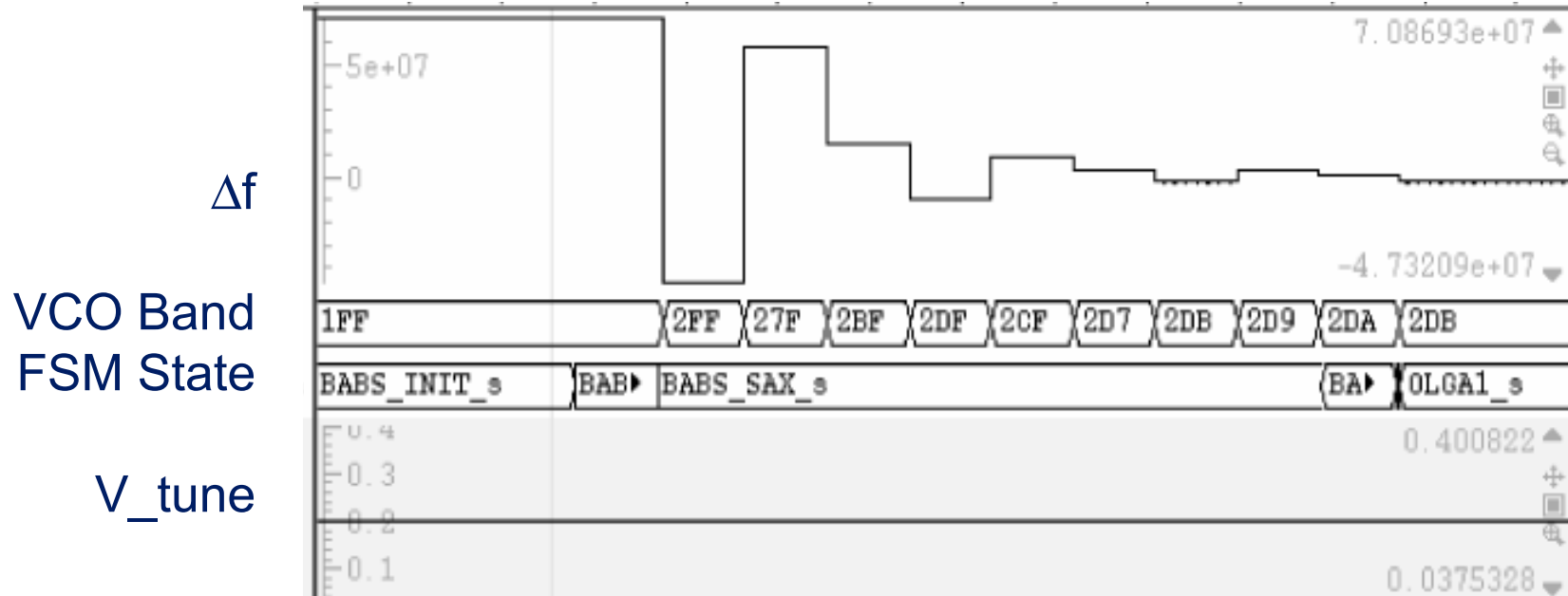


Christian Munker

Verification: Timing, Settling, Phase Error

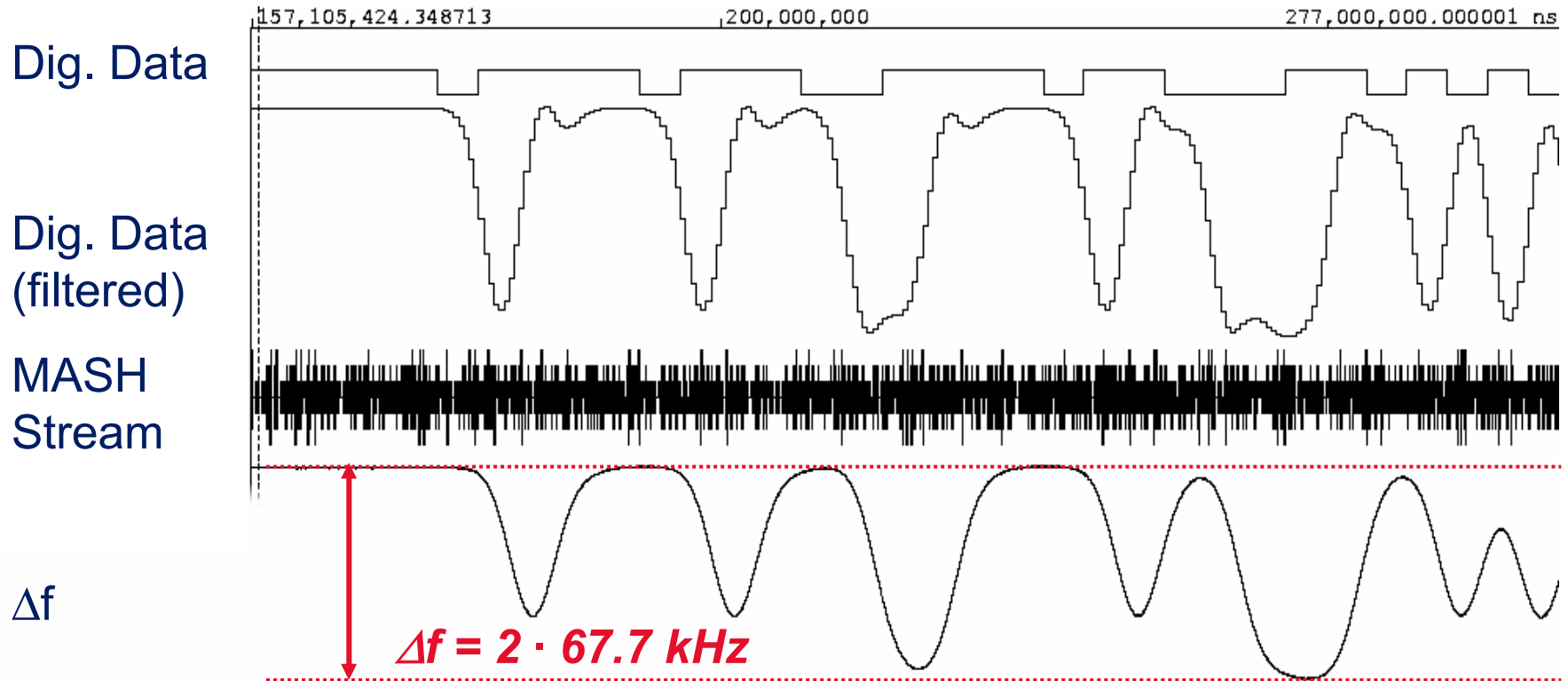
Simulation: Digital VCO Calibration

stop thinking
Never



Verification: digital calibration of VCO bands (algorithm, timing, residual error etc.)

Simulation: Digital Modulation



Verification: Gauss-Filter with pre-distortion, modulation depth at TX-output, spectrum of modulated carrier

Noise Simulations Using VHDL

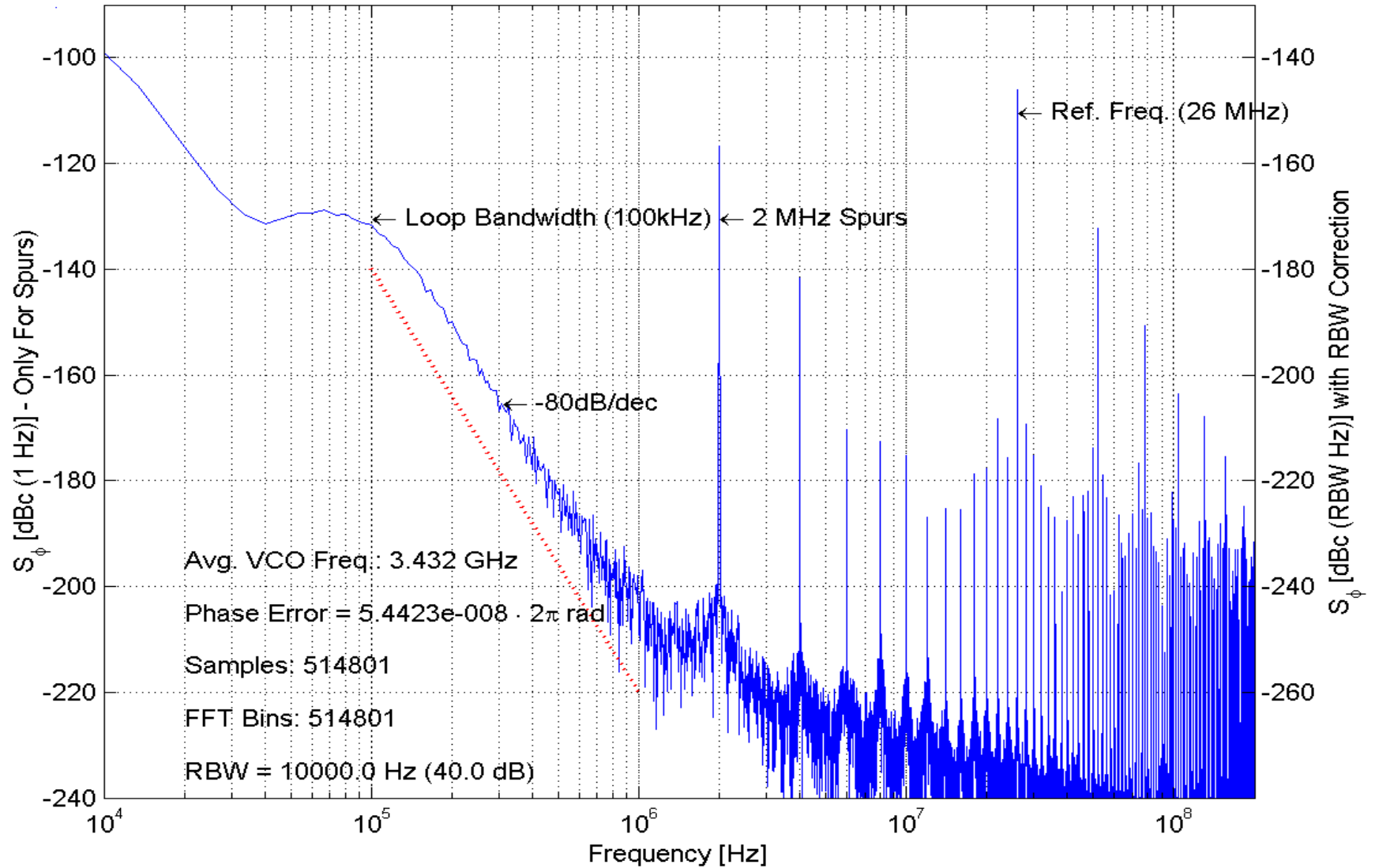
■ VHDL can only run transient simulations

⇒ **phase noise can only be represented as jitter:**

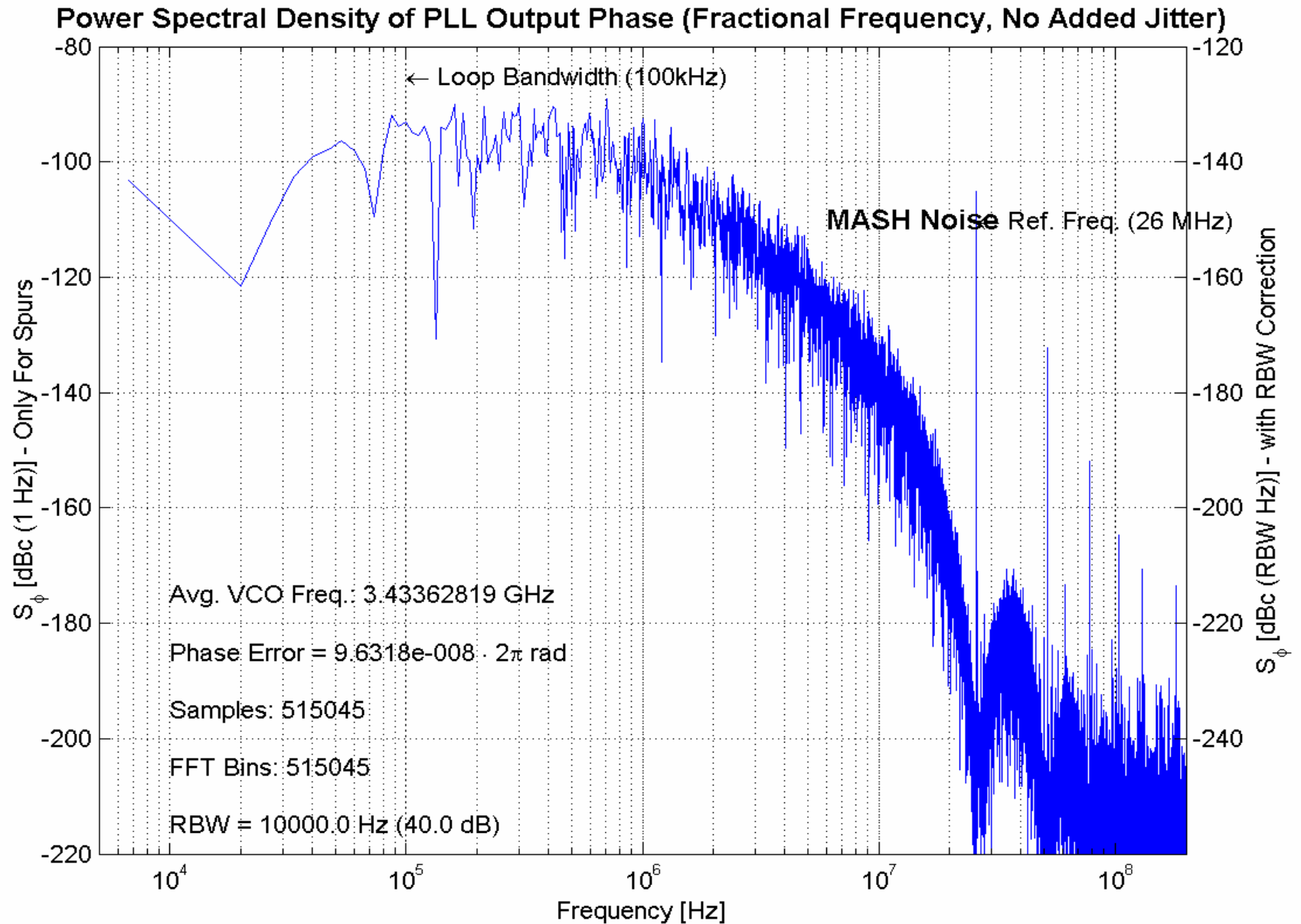
- generate a random process (uniform distribution)
- transform uniform distribution into Gaussian distribution
- transform phase noise values into jitter values
- add jitter processes to your simulation blocks
- use external post-processing (e.g. Matlab) for spectral and other analyses

Noise Simulation: Accuracy Limits

Power Spectral Density of PLL Output Phase (Integer Frequency, No Added Jitter)



Noise Simulation: MASH Noise



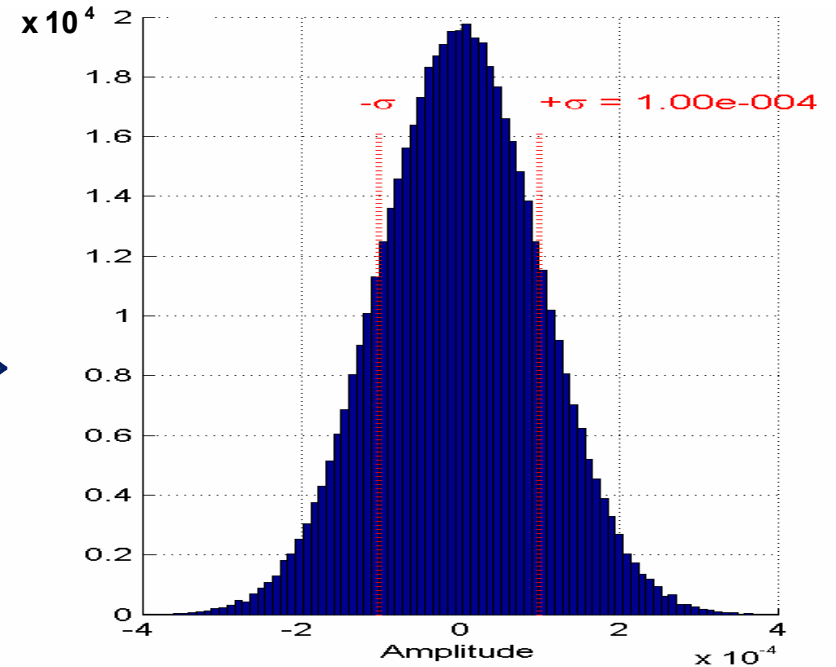
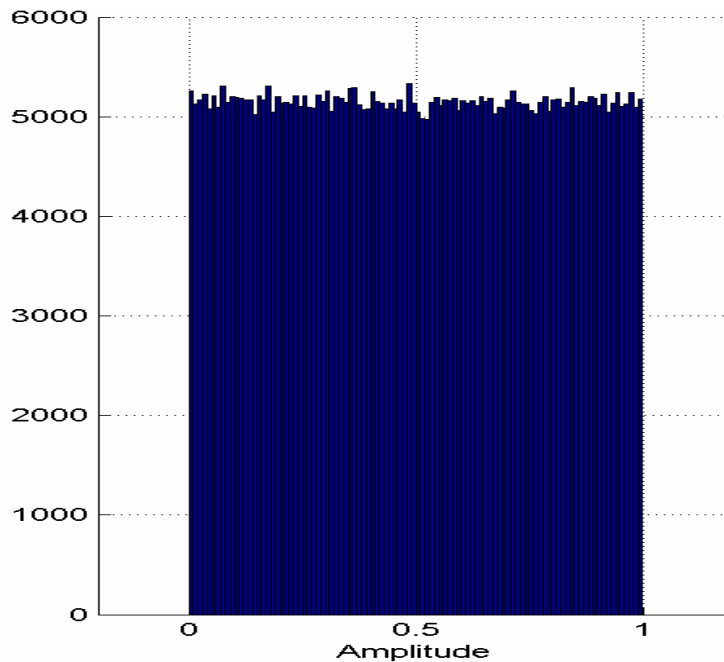
Christian Mürker

June 6, 2004

Page 23 of 31

Jitter Modeling in VHDL

Generate uniform distribution and transform into a Gaussian one:



$$G_{m,1} \equiv m_1 + \sigma_1 \sqrt{2 \ln \frac{1}{1 - U_{n,1}}} \cos 2\pi U_{n,2}$$

$$G_{m,2} \equiv m_2 + \sigma_2 \sqrt{2 \ln \frac{1}{1 - U_{n,1}}} \sin 2\pi U_{n,2}$$

[e.g. Proakis / Manolakis]

Transformation of Phase Noise into Jitter (1)

Jitter J_{UI} is the relative disturbance per period (unit interval, UI):

$$J_{UI}(t) = \frac{\Delta T(t)}{T_0} = \frac{\Delta \phi(t)}{2\pi}$$

$S_\phi(f)$ is the spectral power density of phase change:

$$\int_{f_1}^{f_2} S_\phi(f) df = \Delta \phi_{RMS}^2$$

$$\Rightarrow J_{UI,RMS} = \frac{\Delta \phi_{RMS}}{2\pi} = \frac{1}{2\pi} \sqrt{\int_{f_1}^{f_2} S_\phi(f) df}$$

Phase noise $L(f)$ is $S_\phi(f)$ folded around carrier f_0 :

$$L(f) \approx \frac{1}{2} S_\phi(f - f_0)$$

Transformation of Phase Noise into Jitter (2)

PM with white noise: $S_{\phi}(f) = S_{PM} = \text{const.}$

$$\Rightarrow J_{UI,RMS,PM} = \frac{1}{4\pi} \sqrt{S_{PM} f_0}$$

$$f_0 = 3.4 \text{ GHz}, S_{PM} = -150 \text{ dBc} \Rightarrow J_{UI,RMS,PM} = 1.4 \cdot 10^{-4}$$

FM with white noise $\Rightarrow S_{\phi}(f) = S_{FM}(1 \text{ Hz}) / f^2$

$$\Rightarrow J_{UI,RMS,FM} = \sqrt{\frac{S_{FM}(1 \text{ Hz})}{2f_0}}$$

$$f_0 = 3.4 \text{ GHz}, S_{FM}(1 \text{ Hz}) = 0 \text{ dBc} \Rightarrow J_{UI,RMS,FM} = 1.2 \cdot 10^{-5}$$

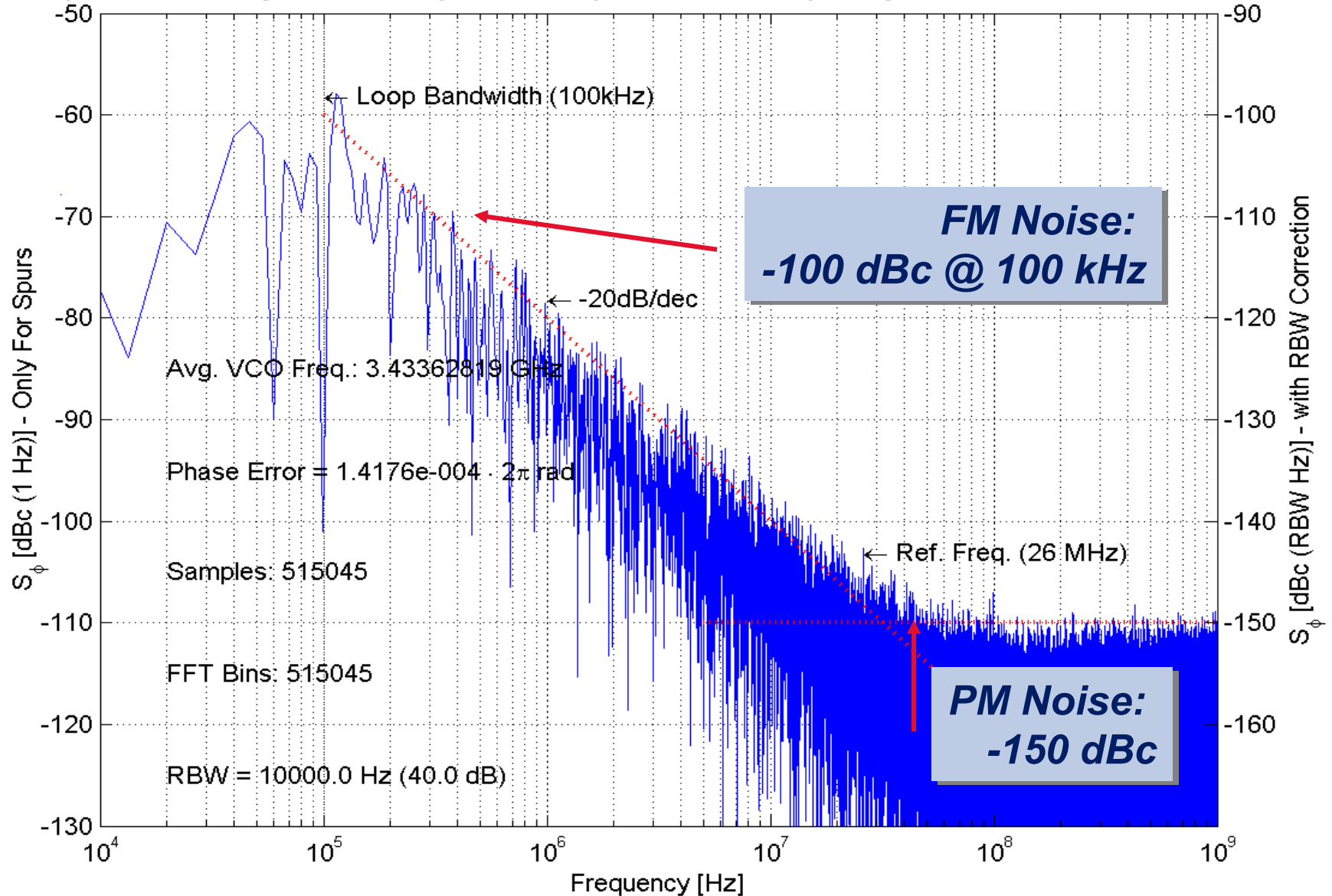
VCO Noise Modeling

- FM Jitter: modulation of period duration
- PM Jitter: modulation of transition time

```
period  <=  2.0 * math_pi * sqrt(L_0 * C_tot)
          *  (1.0 + fm_jitter)
          +  pm_jitter - pm_jitter_last;
```

Noise Simulation: PLL with PM and FM Jitter at VCO

Power Spectral Density of PLL Output Phase (Fractional Frequency, PM: $\sigma = 1.41 \cdot 10^{-4}$, FM: $\sigma = 10^{-4}$)



Christian Mürker

June 6, 2004

Page 28 of 31

Alternative Solutions for Mixed-Signal Verification

■ Matlab for modeling of analog blocks + VHDL

- ⇒ Fast transfer from concept engineering
- ⇒ No verification of connectivity, implementation of feedback loops is difficult

■ ADS System (Ptolemy) + VHDL + Matlab

- ⇒ Very powerful tool but steep learning curve

■ Mixed-signal simulator

- ⇒ Too resource hungry and too slow for complex systems

Conclusion: System Simulations Using VHDL

VHDL as a mixed-signal simulator allows

■ Modeling of analog blocks

- ⇒ Continuous-time systems / blocks can be included by transforming their s-plane behavior into the z-plane
- ⇒ “Fractional sampling” mitigates negative effects due to sampling

■ Mixed-signal system simulations

- ⇒ Modeling of analog blocks and stimuli in VHDL allows fast and accurate co-simulation with digital blocks (pin-true, gate level)
- ⇒ Bridges the gap between concept and circuit engineering

■ Simulation of system phase noise / jitter

- ⇒ Estimation of noise effects involving analog-digital interaction
- ⇒ Very efficient because no additional timing events are generated

Room for Improvement:

- Improved modeling of noise and jitter
- Modeling of periodic disturbances (spurious sidebands)
- Modeling of systems with feedback
- Improved post-processing and measurement functions