Design, Simulation, and Bandwidth Extension Methods for Fractional-N Frequency Synthesizers

ISSCC 2005 GIRAFE Forum on Clock and Frequency Generation

Michael Perrott

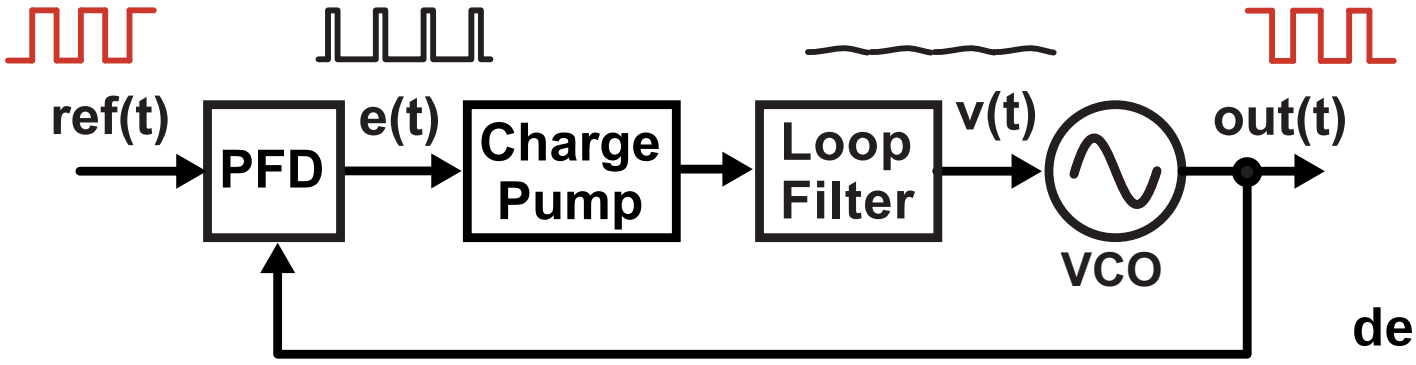
Massachusetts Institute of Technology

February 6, 2005

Outline

- **Fundamentals**
- **Proposed Bandwidth Extension Technique**
- **Design at the Transfer Function Level**
- **Simulation at the Behavioral Level**
- **Measured Results**

What Is A Phase-Locked Loop?

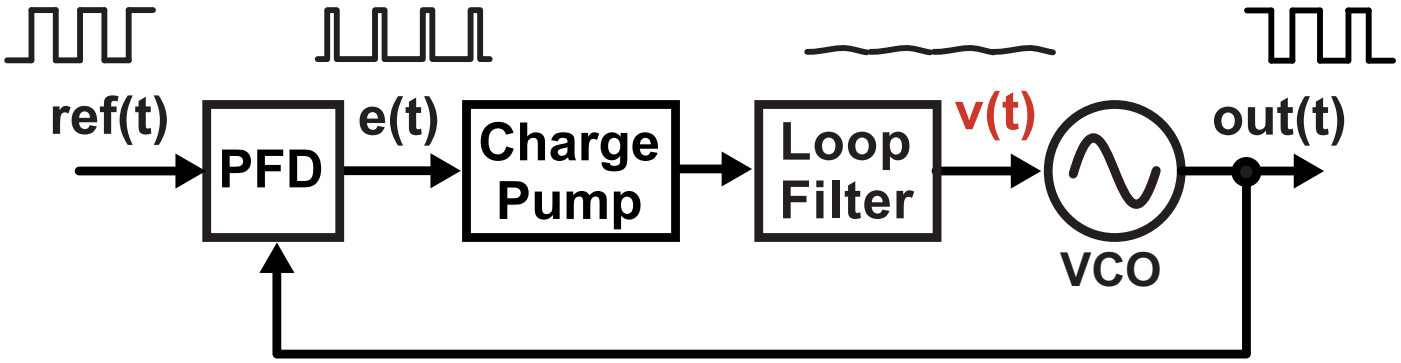


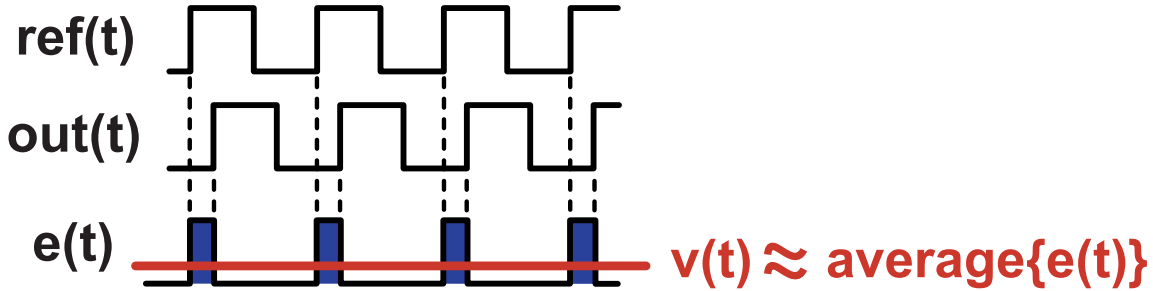
de Bellescize
Onde Electr, Vol. 11
1932

- VCO ⇨ produces variable frequency output
- Reference ⇨ provides input frequency/phase
- PFD ⇨ compares phase of ref and VCO output
- Charge pump ⇨ simplifies loop filter implementation
- Loop filter ⇨ smooths PFD signal

Objective: "Lock" VCO phase to reference phase

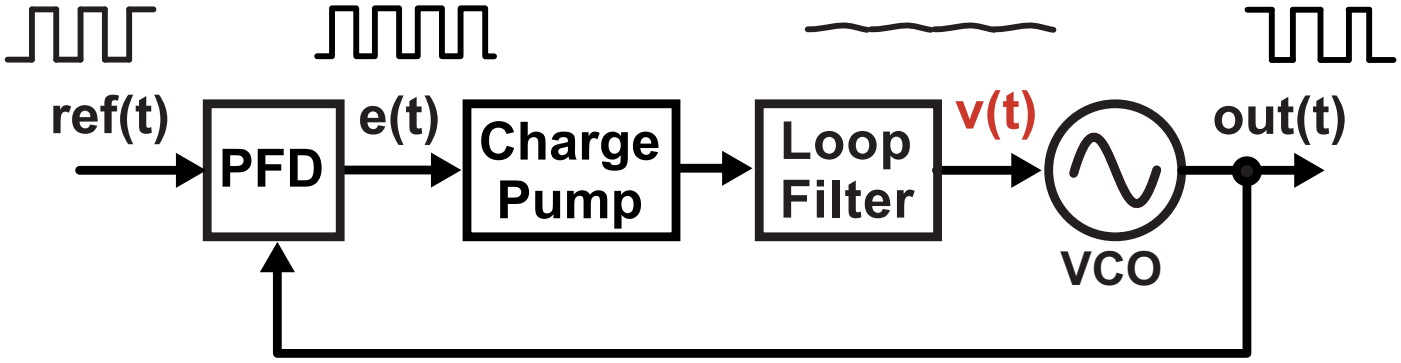
Method of Phase Detection

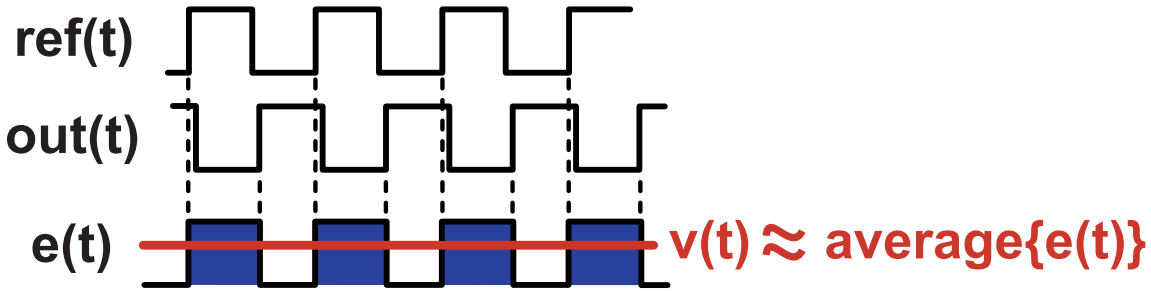




- PFD output consists of pulses whose width is proportional to the phase error
  - Phase is only observable at edges
- Smooth PFD output to produce input voltage to VCO

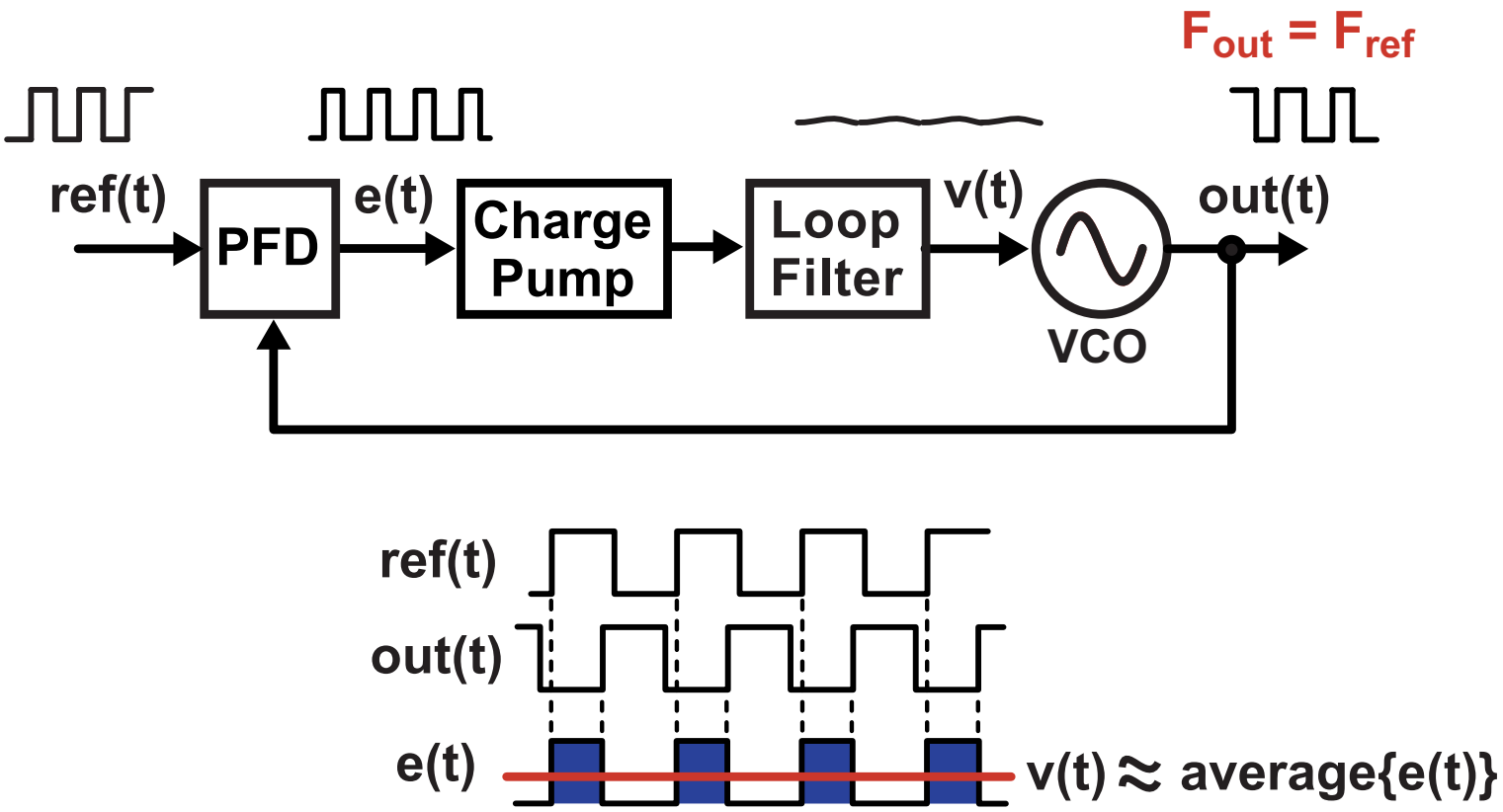
Impact of Changes in Phase Error





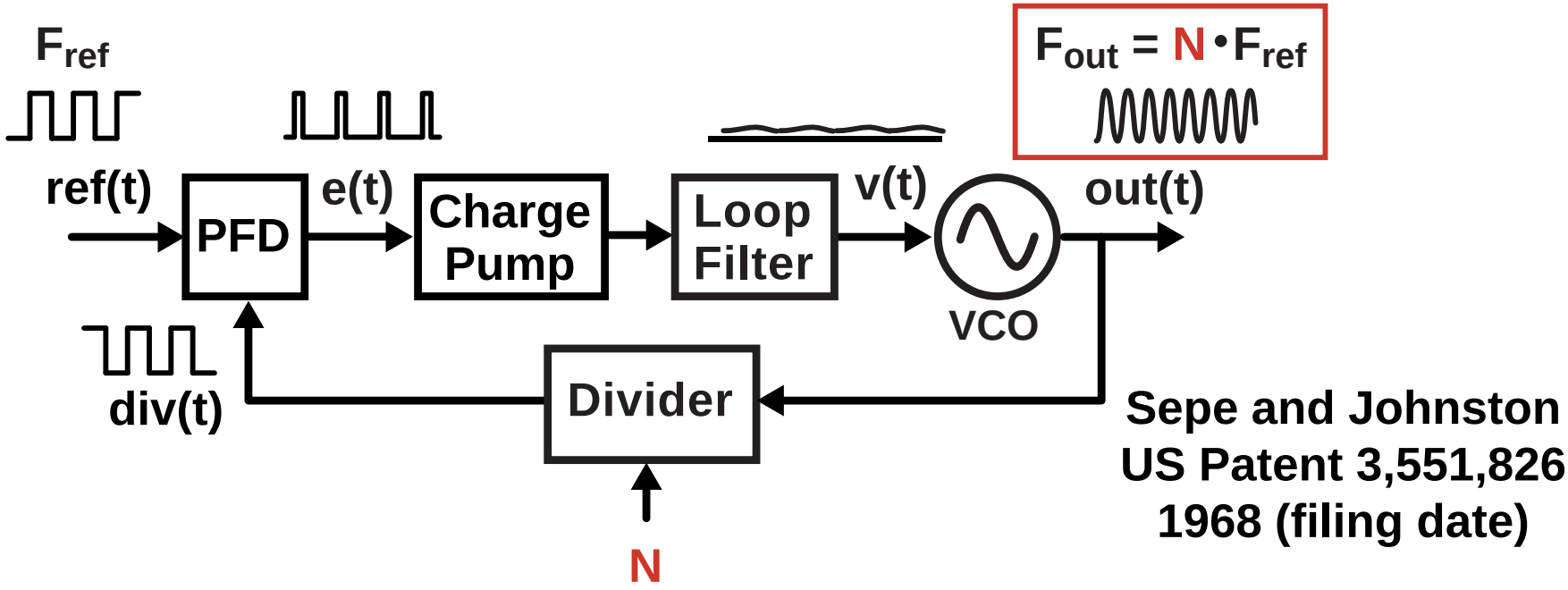
- Pulse width varies according to phase difference
- VCO input voltage changes accordingly
  - Adjusts VCO frequency and phase

Phase Lock Implies Frequency Lock



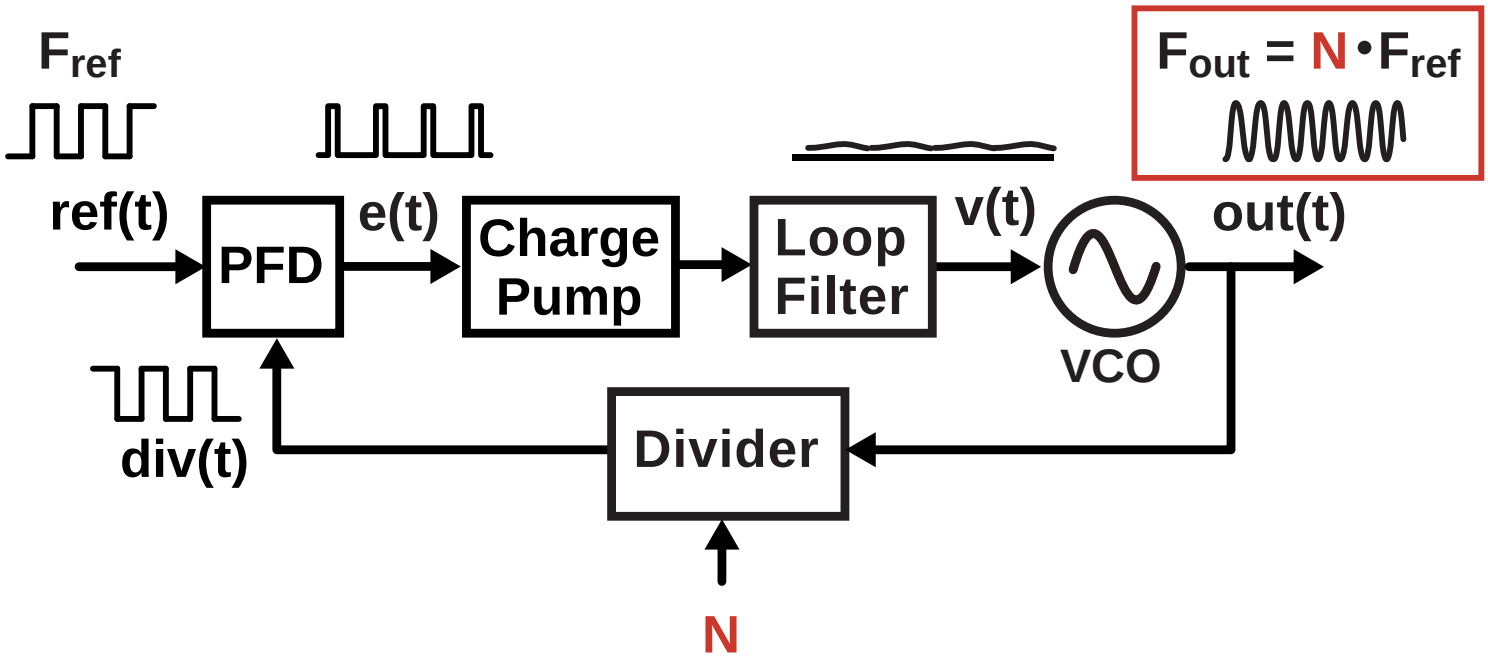
- Any error in frequency leads to a steady accumulation of phase error

Integer-N Frequency Synthesizer



- Leverages frequency divider to create “indirect” frequency multiplication
  - Allows digital adjustment of output frequency in increments of the reference frequency

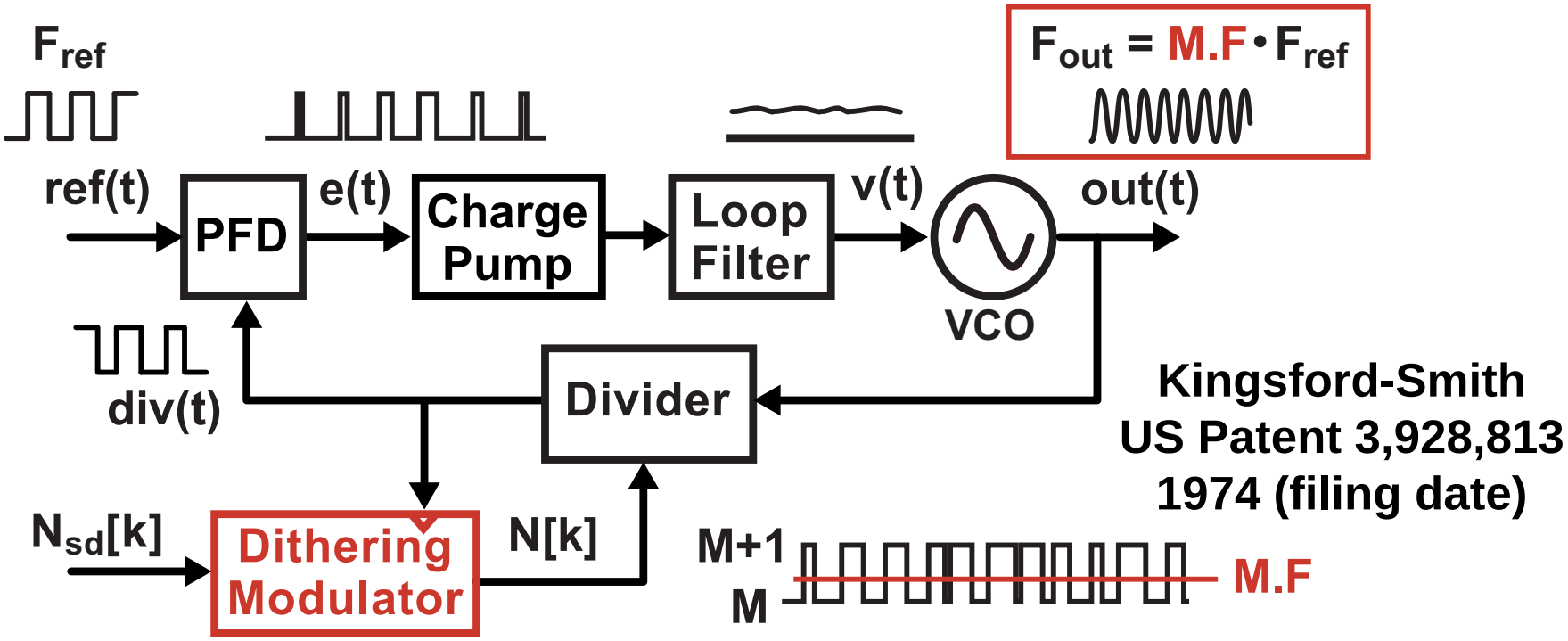
Issue



- **Key constraint: Divider value, N, must be integer**
  - **High frequency resolution requires low F_{ref}**
  - **High PLL bandwidth requires high F_{ref}**

Tradeoff: Frequency resolution vs PLL bandwidth

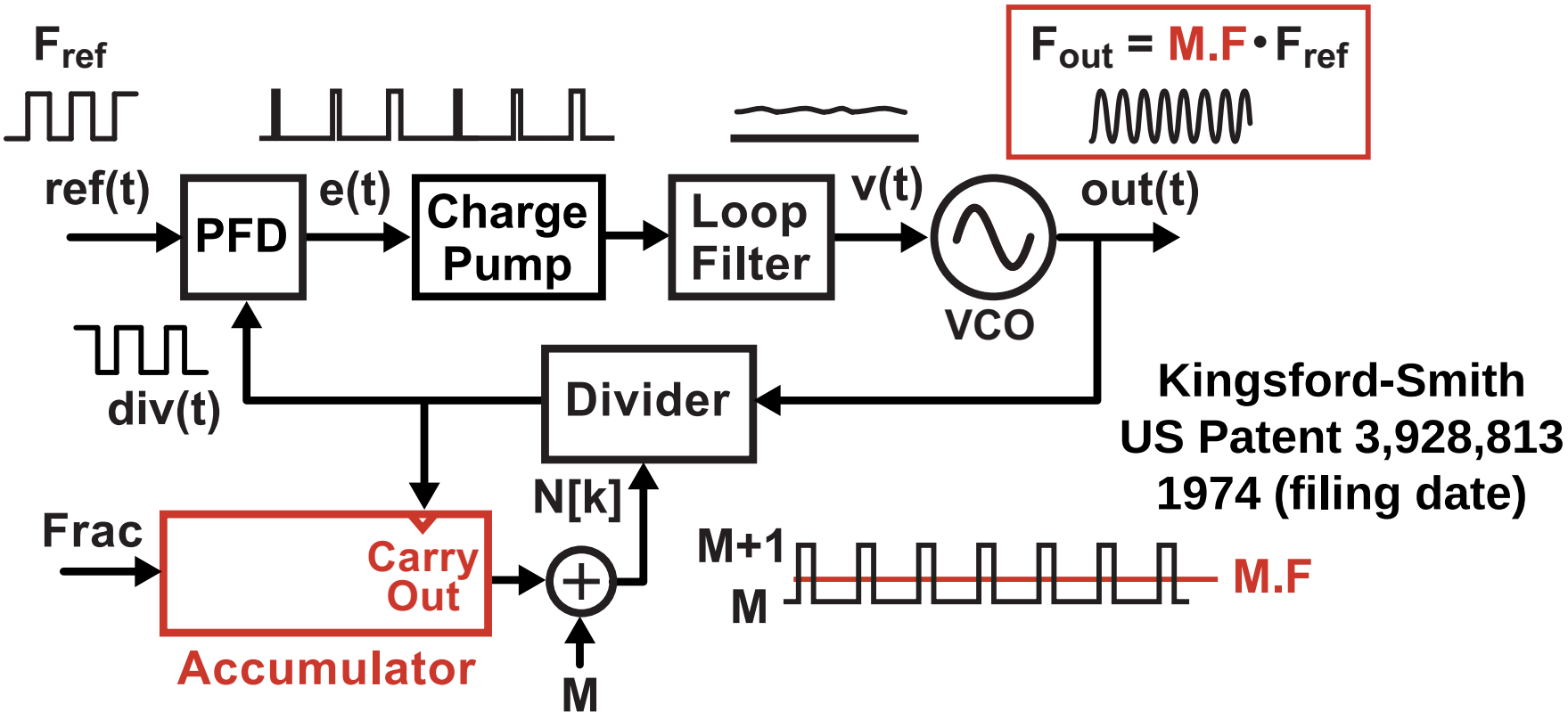
Fractional-N Frequency Synthesis



- Divide value is dithered between integer values
- Fractional divide values can be realized!

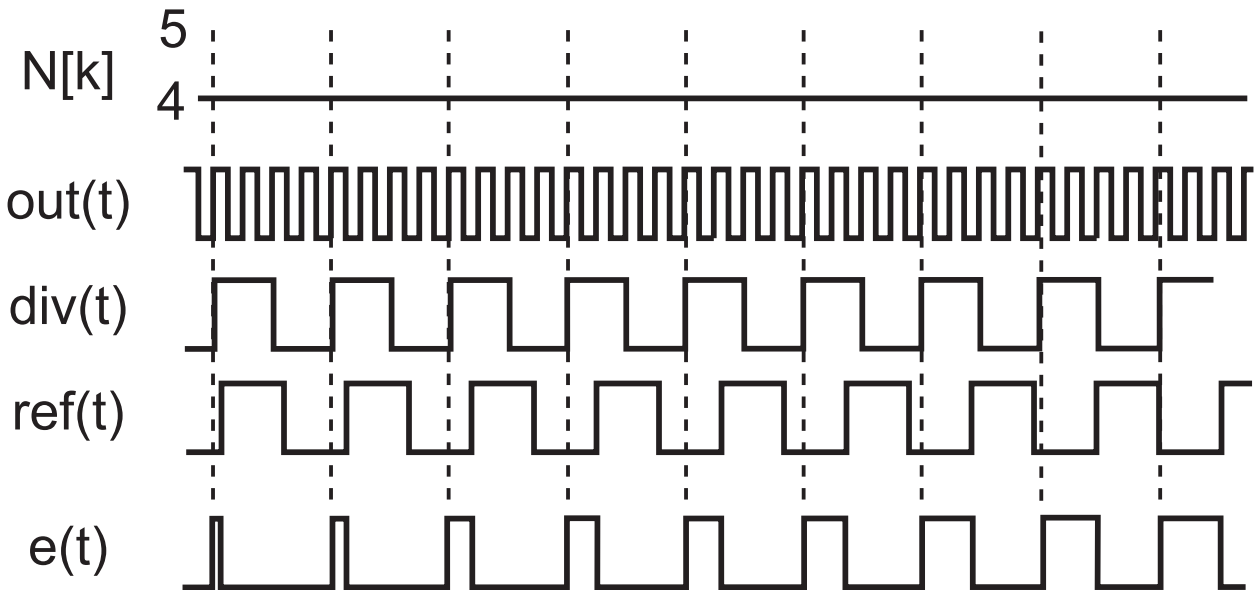
Very high frequency resolution

Classical Fractional-N Synthesizer Architecture



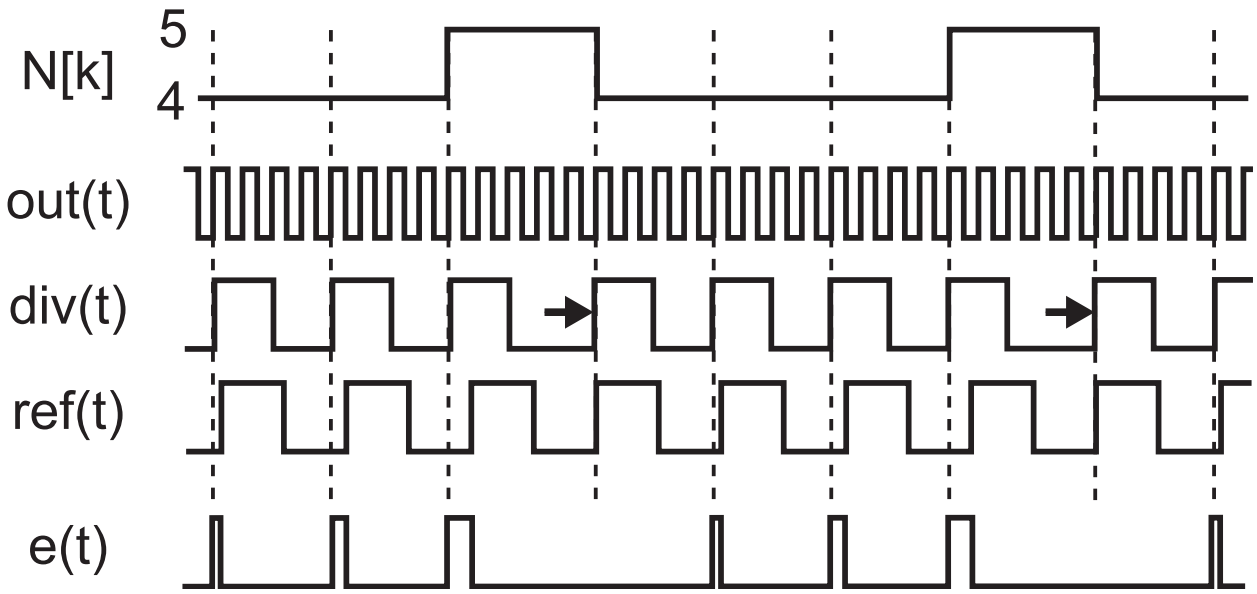
- Use an accumulator to perform dithering operation
  - Fractional input value fed into accumulator
  - Carry out bit of accumulator fed into divider

Integer-N Synthesizer Signals with $F_{out} = 4.25F_{ref}$



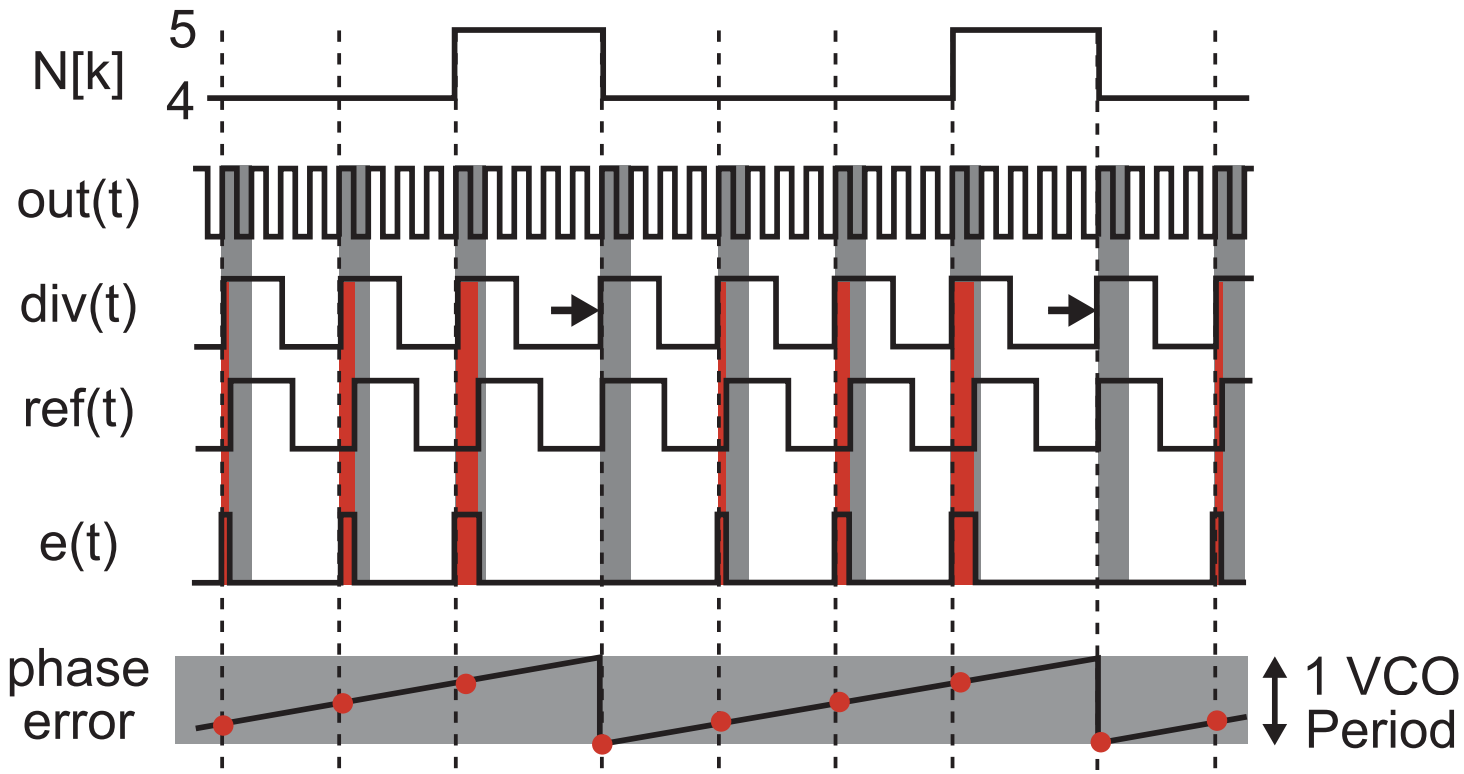
- **Constant divide value of N = 4 leads to frequency error**
  - **Error pulse widths increase as phase error accumulates**

Fractional-N Synthesizer Signals with $F_{out} = 4.25F_{ref}$



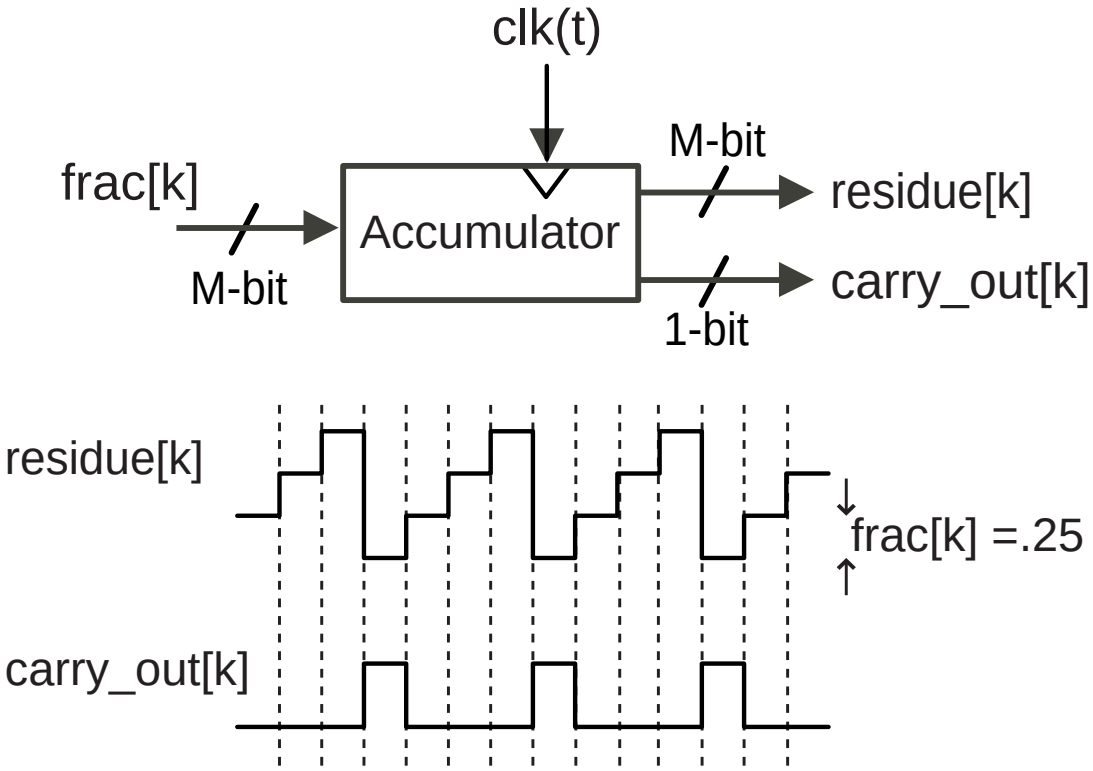
- **Dithering allows average divide value of N = 4.25**
  - **Reset phase error by periodically "swallowing" a VCO cycle**
    - Achieved by dividing by 5 every 4 reference cycles

Key Observations for Classical Fractional-N Dithering



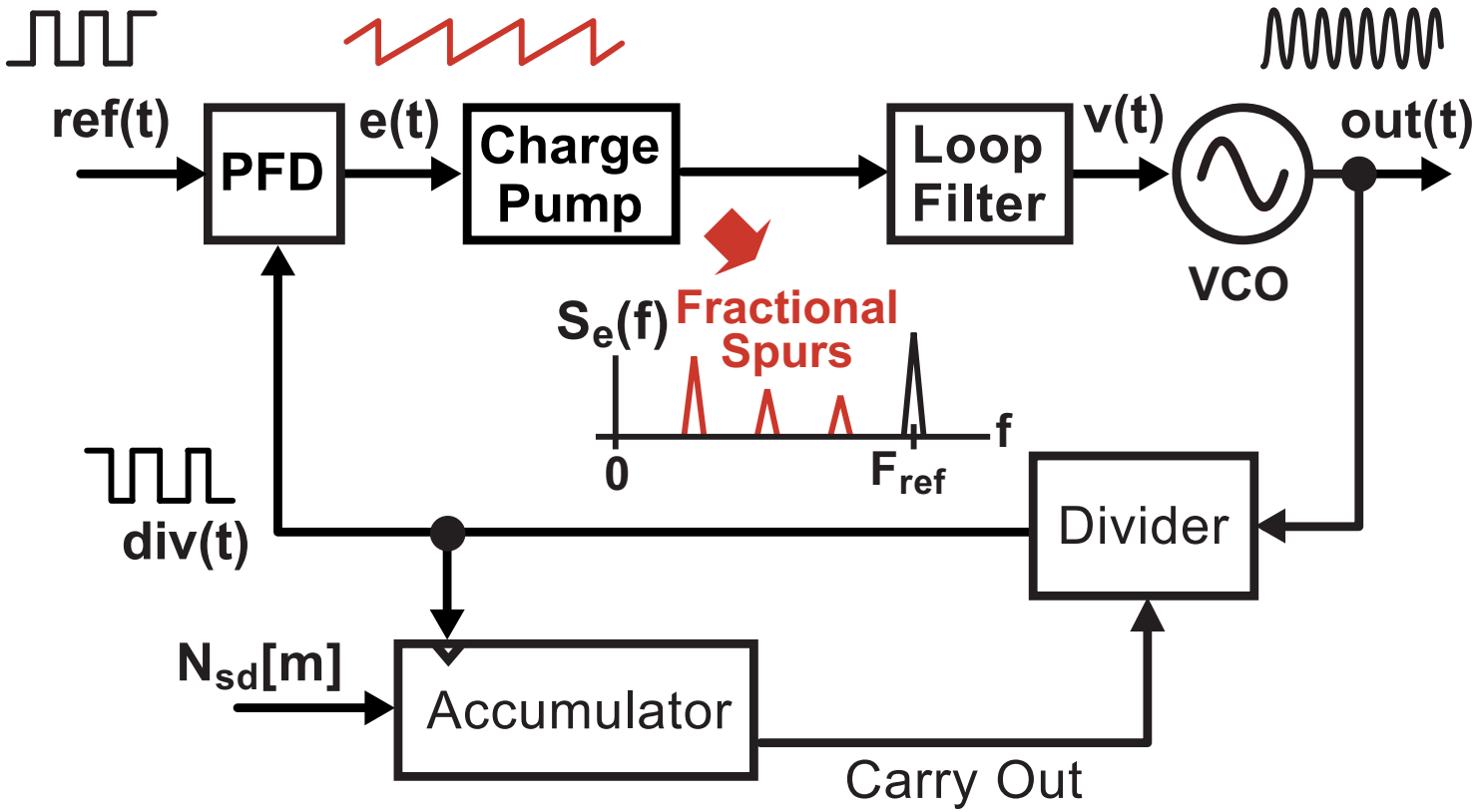
- **The instantaneous phase error always remains less than one VCO cycle**
- **We can directly relate the phase error to the residue of the accumulator that is providing the dithering**

Accumulator Operation



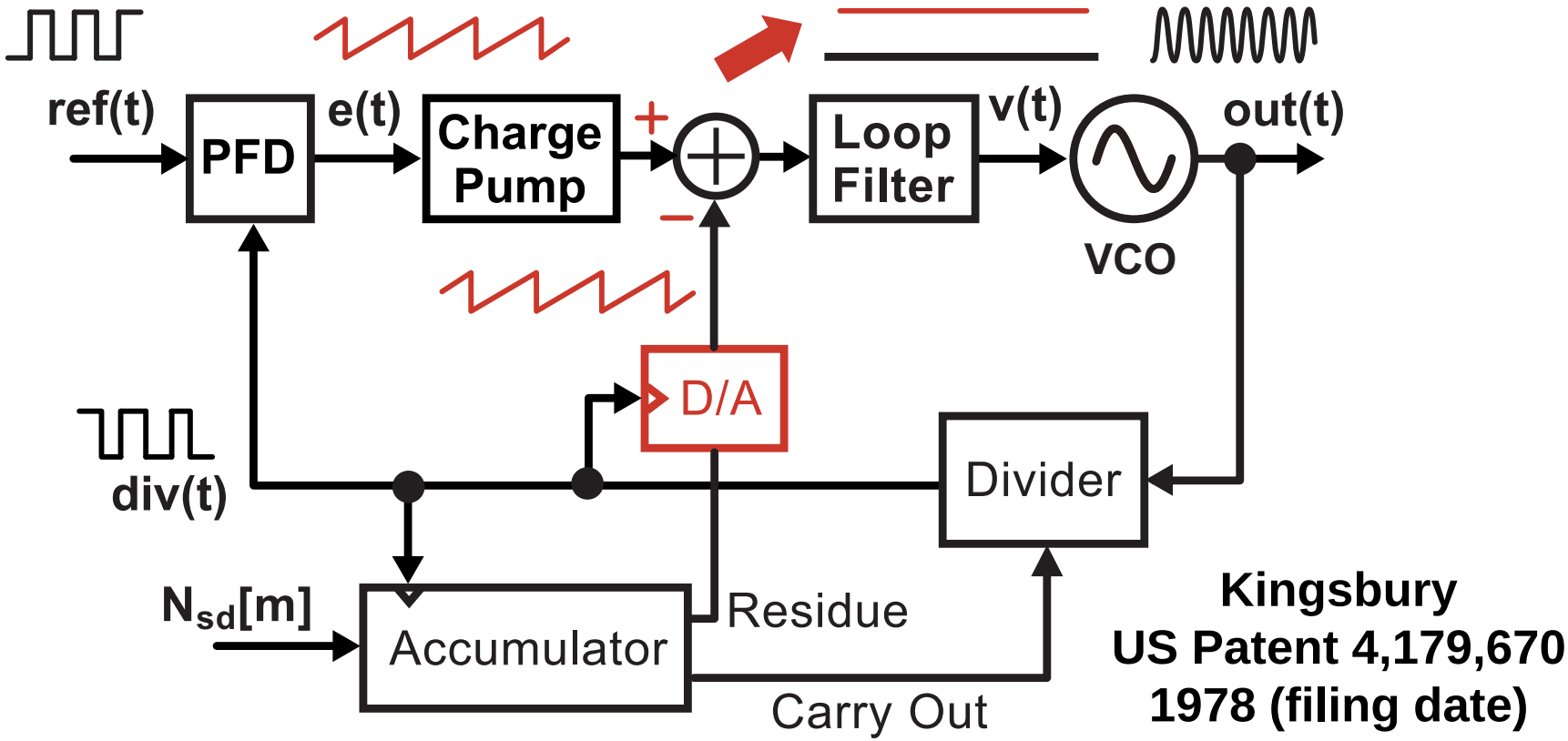
- Carry out bit is asserted when accumulator residue reaches or surpasses its full scale value
- Accumulator residue corresponds to instantaneous phase error
  - Increments by the fractional value input into the accumulator

The Issue of Spurious Tones



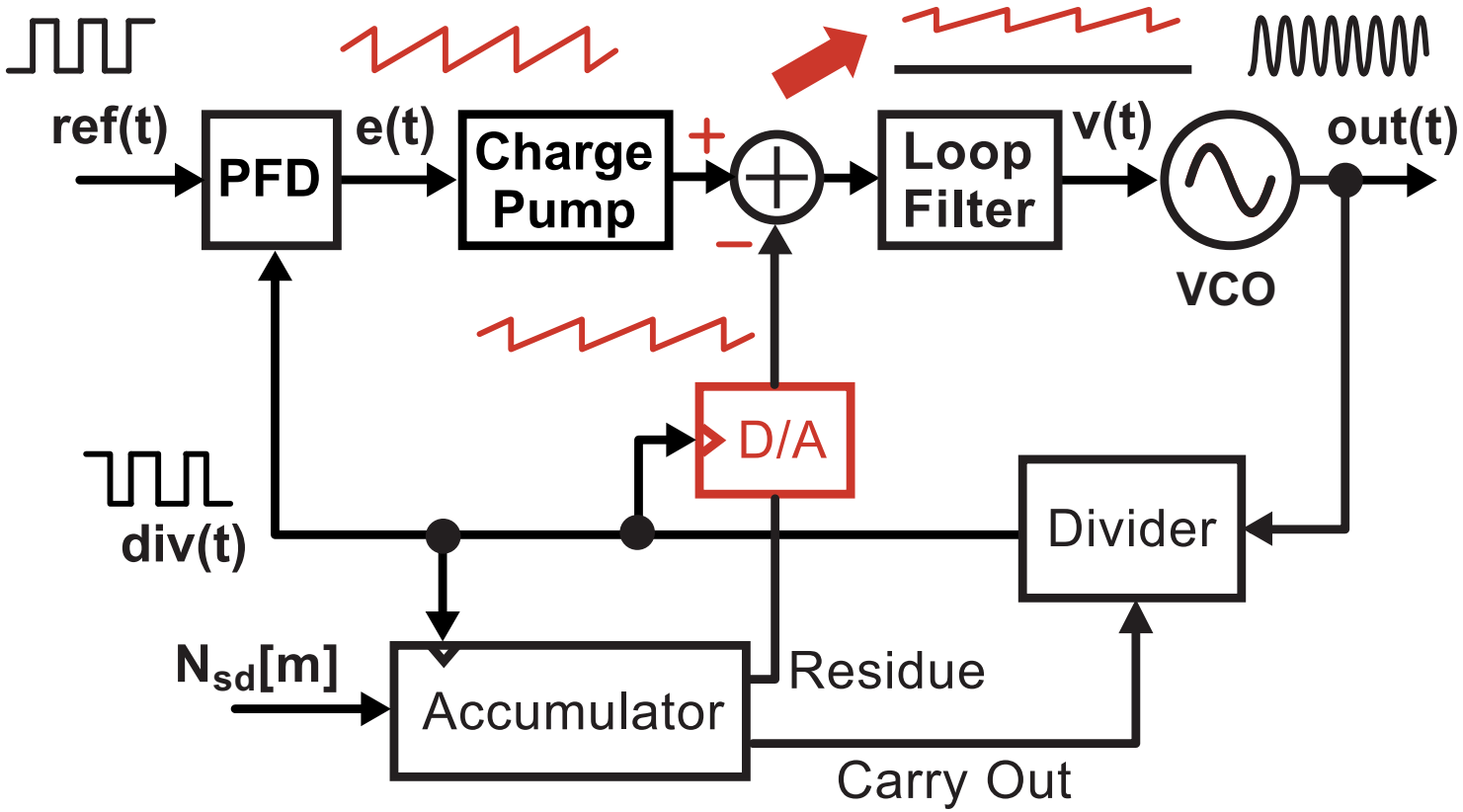
- PFD error waveform is periodic
  - Creates spurious tones in synthesizer output at lower frequencies than the reference
  - Ruins noise performance of the synthesizer

The Phase Interpolation Technique



- Leverage the fact that the phase error due to fractional technique is predicted by the instantaneous residue of the accumulator
  - Cancel out phase error based on accumulator residue

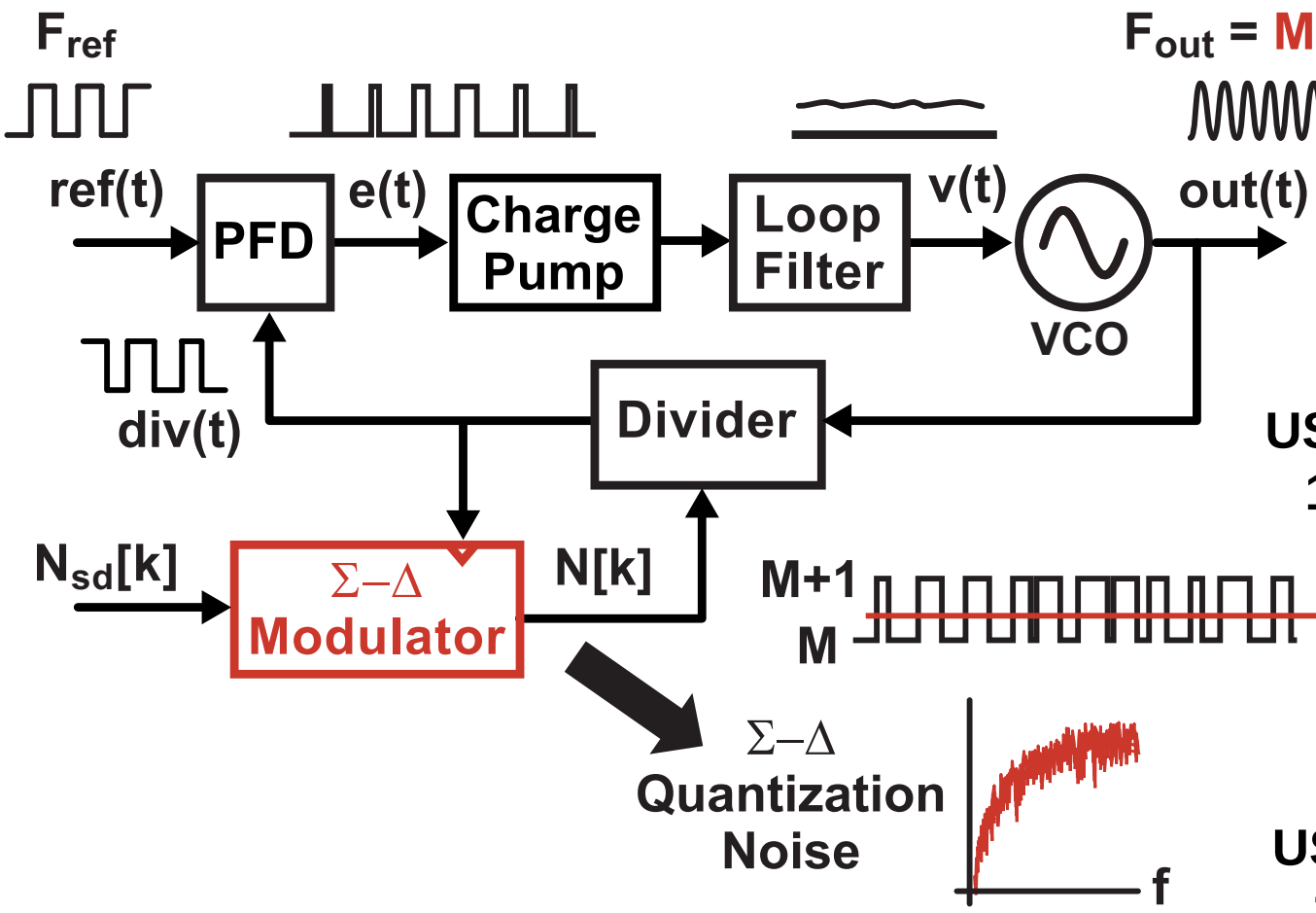
The Problem With Phase Interpolation



- **Gain matching between PFD error and scaled D/A output must be extremely precise**
  - **Any mismatch will lead to spurious tones at PLL output**

Matching issue prevented this technique from catching on

Σ–Δ Fractional-N Frequency Synthesis



MASH Σ–Δ

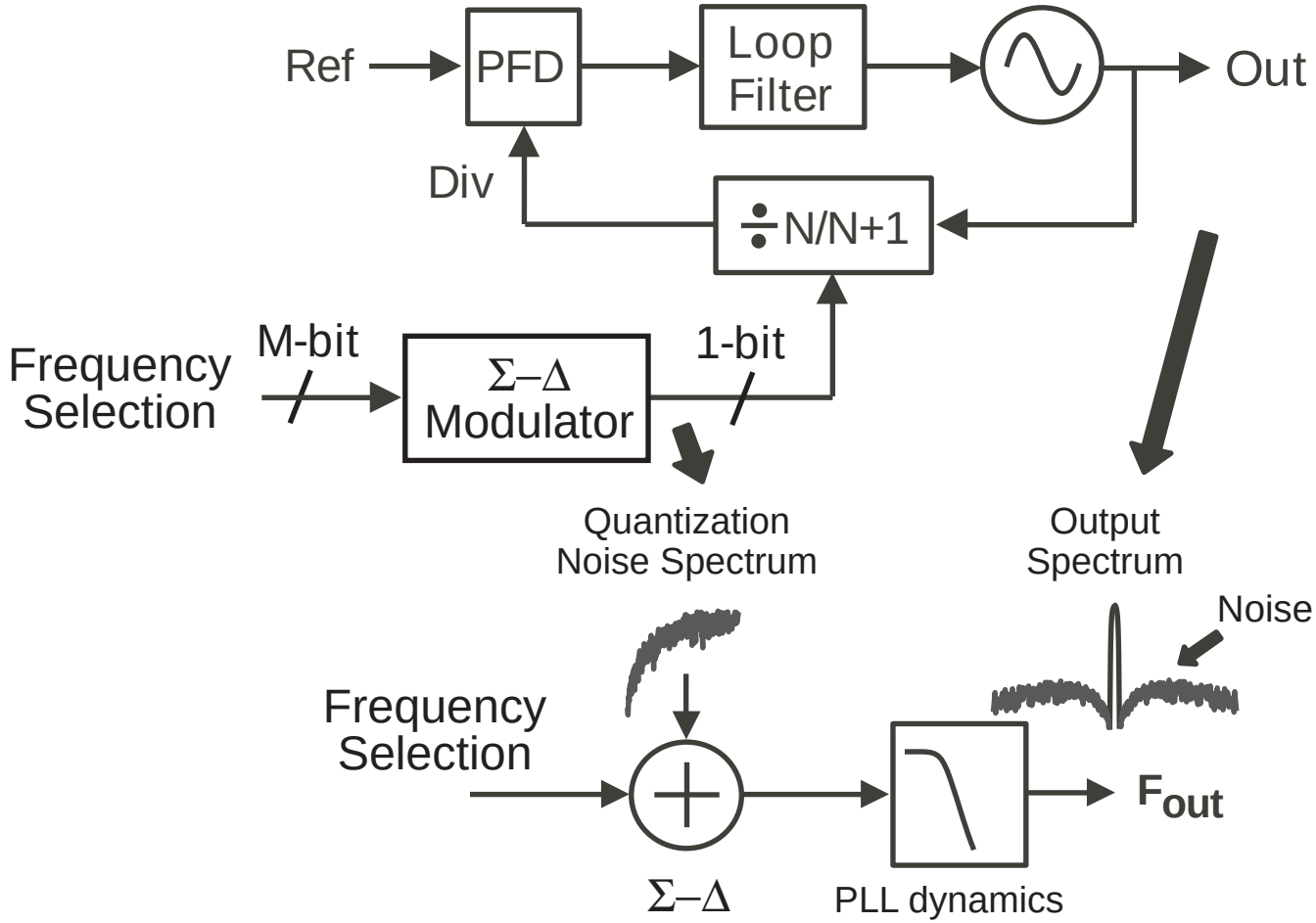
Wells
US Patent 4,609,881
1984 (filing date)

General Σ–Δ

Riley
US Patent 4,965,531
1989 (filing date)
JSSC '93

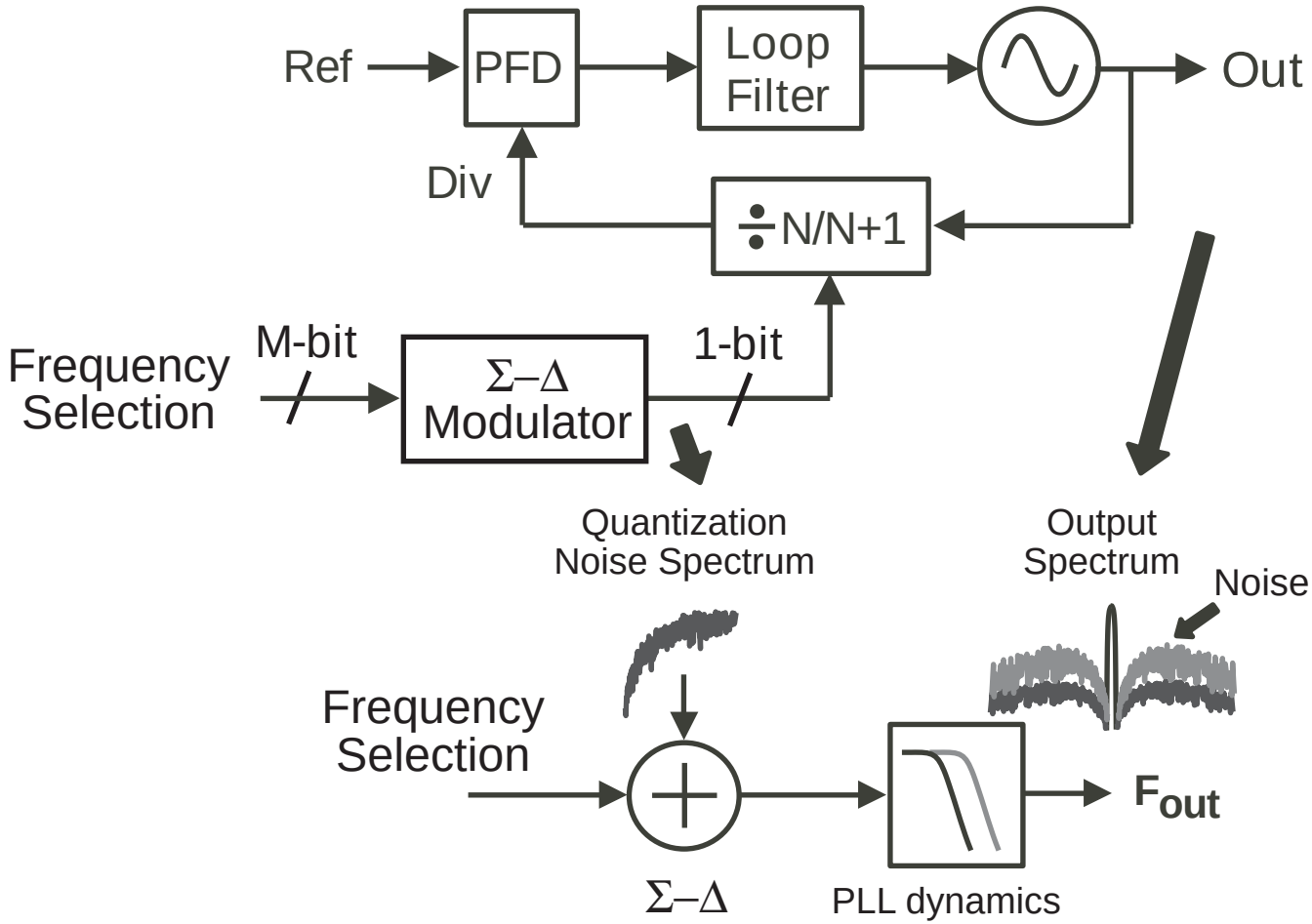
- **Dither using a Σ–Δ modulator**
  - **Quantization noise is shaped to high frequencies**
  - **Spur content of the quantization noise can be reduced to negligible levels**

Impact of Σ–Δ Quantization Noise on Synth. Output



- **Lowpass action of PLL dynamics suppresses the shaped Σ-Δ quantization noise**

Impact of Increasing the PLL Bandwidth



- **Higher PLL bandwidth leads to less quantization noise suppression**

Tradeoff: Noise performance vs PLL bandwidth

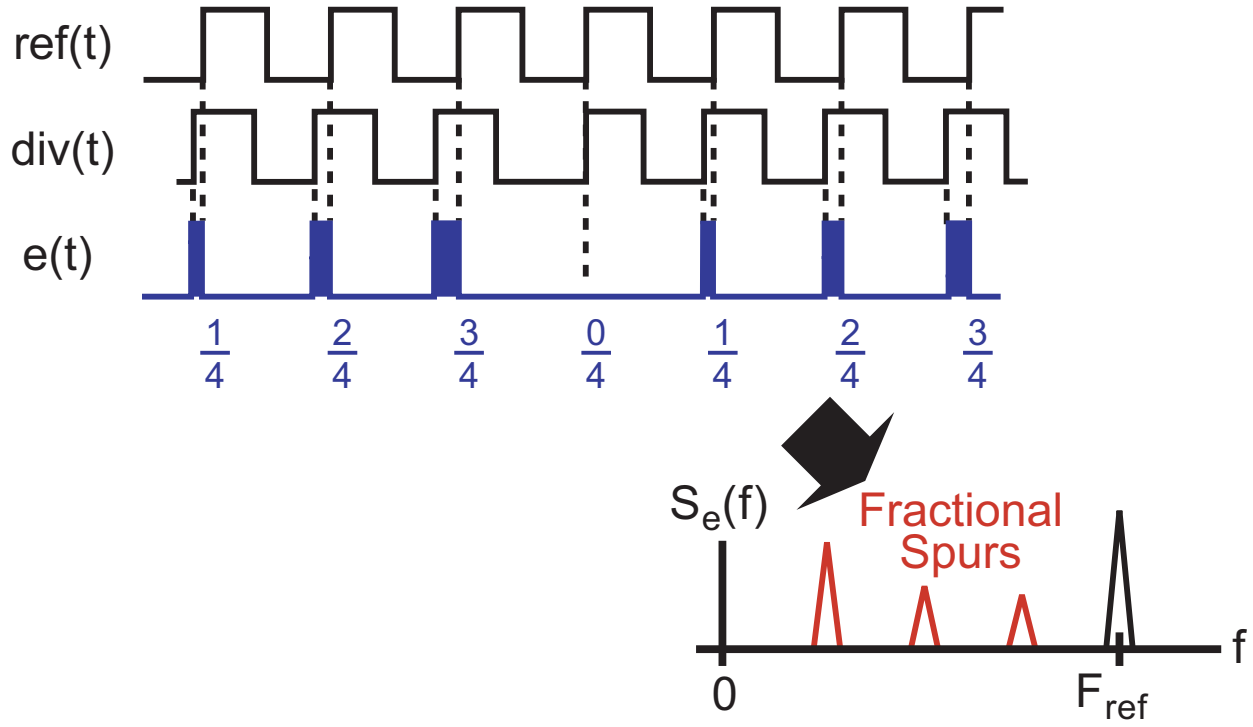
Can We Do Better?

Recent Approaches to Bandwidth Extension

- [1] C. Park, O. Kim, and B. Kim, "A 1.8GHz Self-Calibrated Phase-Locked Loop with Precise I/Q Matching," IEEE JSSC, May 2001.
- [2] K. Lee, et. al., "A Single Chip 2.4GHz Direct-Conversion CMOS Receiver for Wireless Local Loop Using Multiphase Reduced Frequency Conversion Technique ," IEEE JSSC, May 2001.
- [3] S. Pamarti, L. Jansson, and I. Galton, "A Wideband 2.4GHz Delta-Sigma Fractional-N PLL With 1Mb/s In-Loop Modulation", IEEE JSSC, Jan 2004
- [4] E. Temporiti, et. al., "A 700kHz Bandwidth Σ–Δ Fractional-N Frequency Synthesizer with Spurs Compensation and Linearization Techniques for WCDMA Applications", IEEE JSSC, Sept 2004

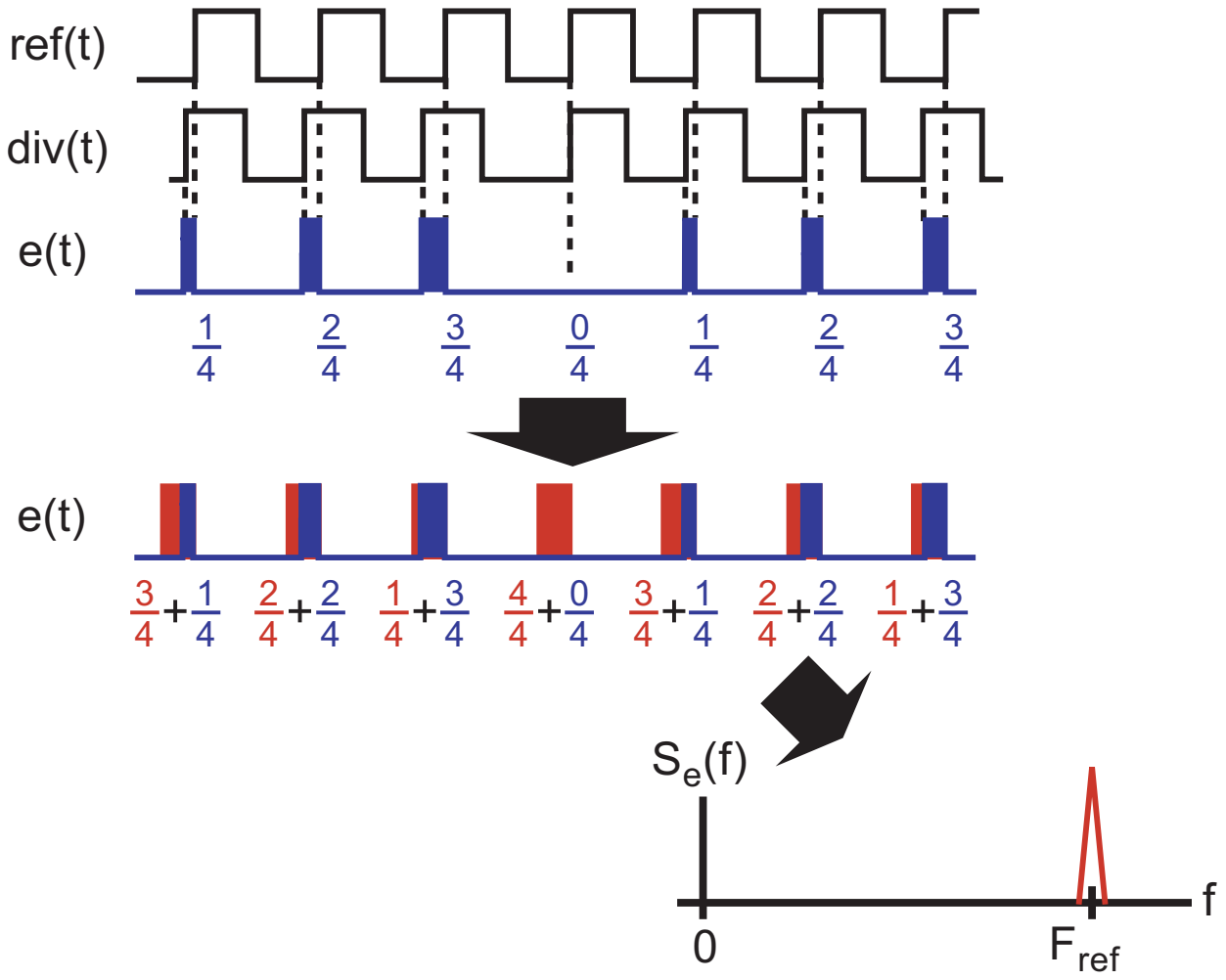
We will focus on our own approach in this talk

Examine Classical Fractional-N Signals



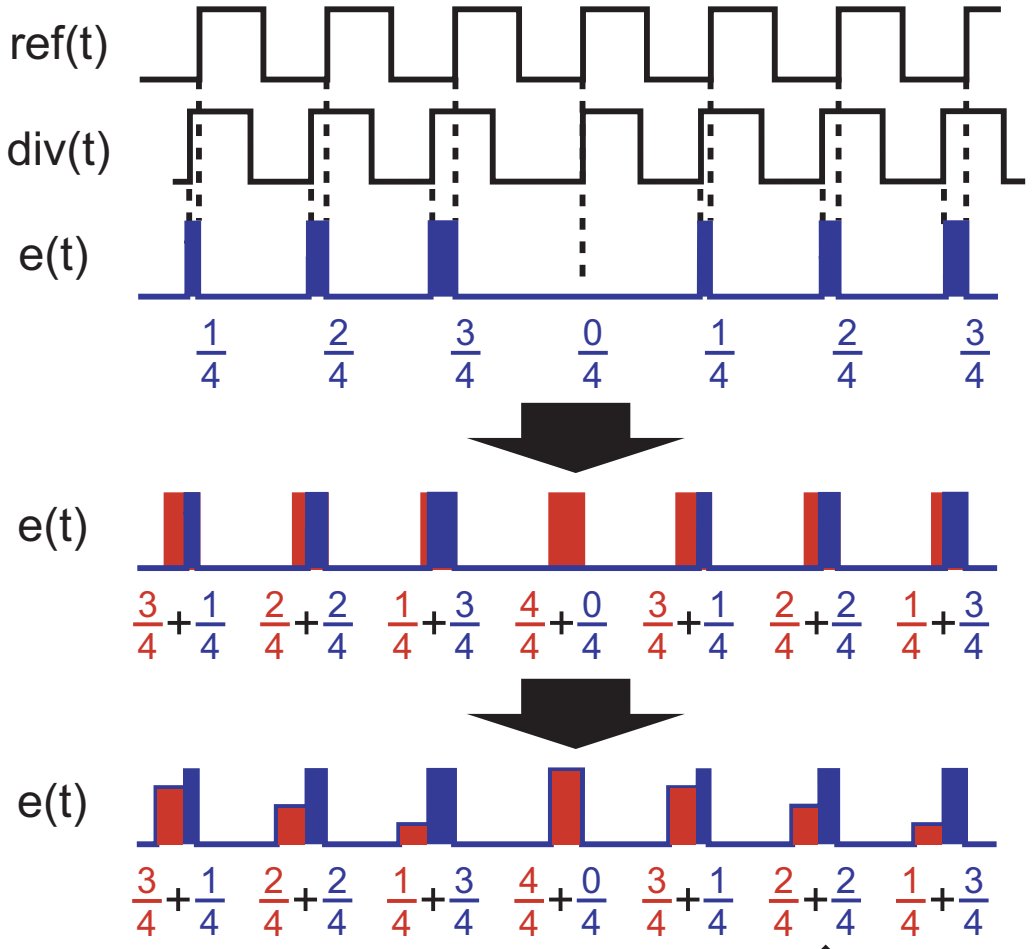
- **Goal: eliminate the fractional spurs**

Method 1: Vertical Compensation

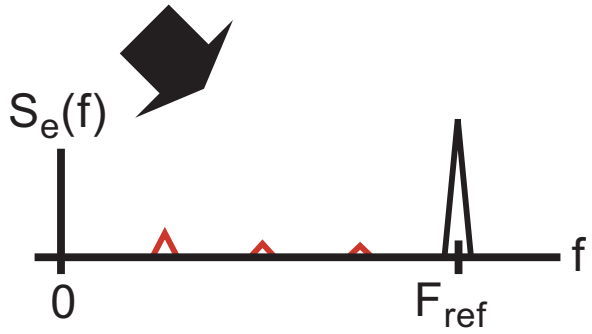


- "Fill in" pulses so that they are constant area
  - Fractional spurs are eliminated!

Method 2: Horizontal Compensation

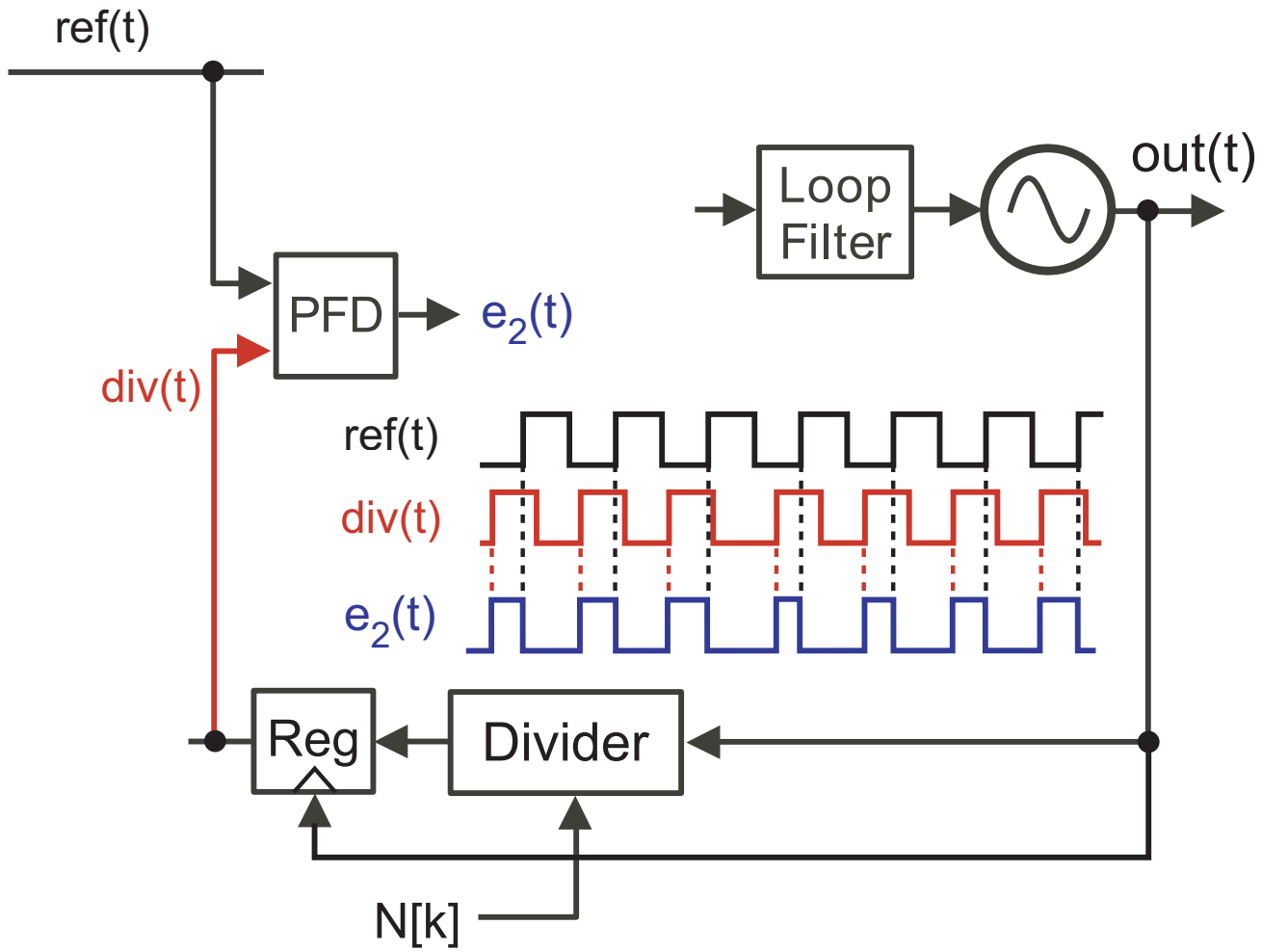


- **Use constant width pulses of varying height to achieve constant area pulses**
  - **Largely eliminates fractional spurs**

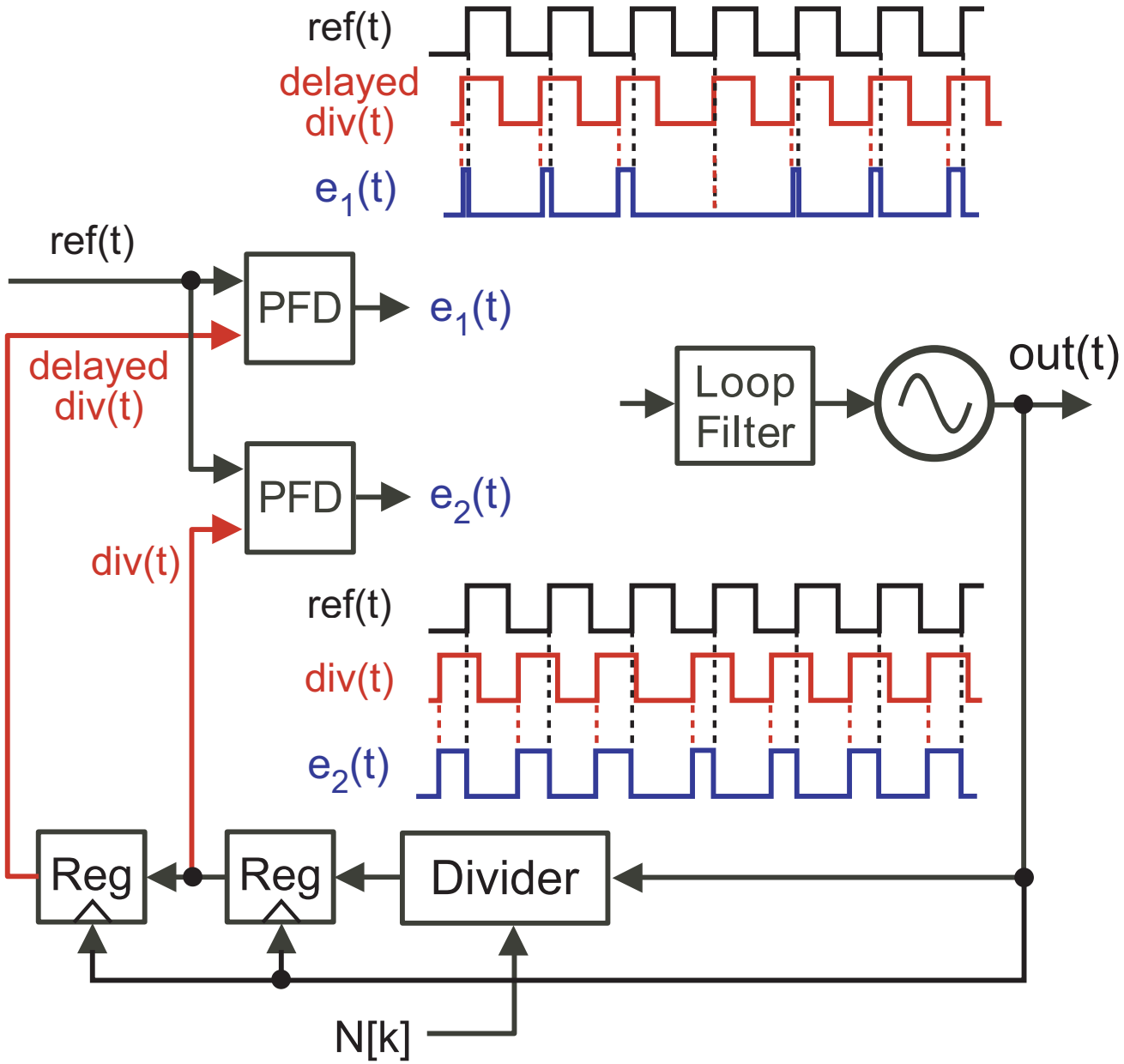


Implementation of Horizontal Cancellation

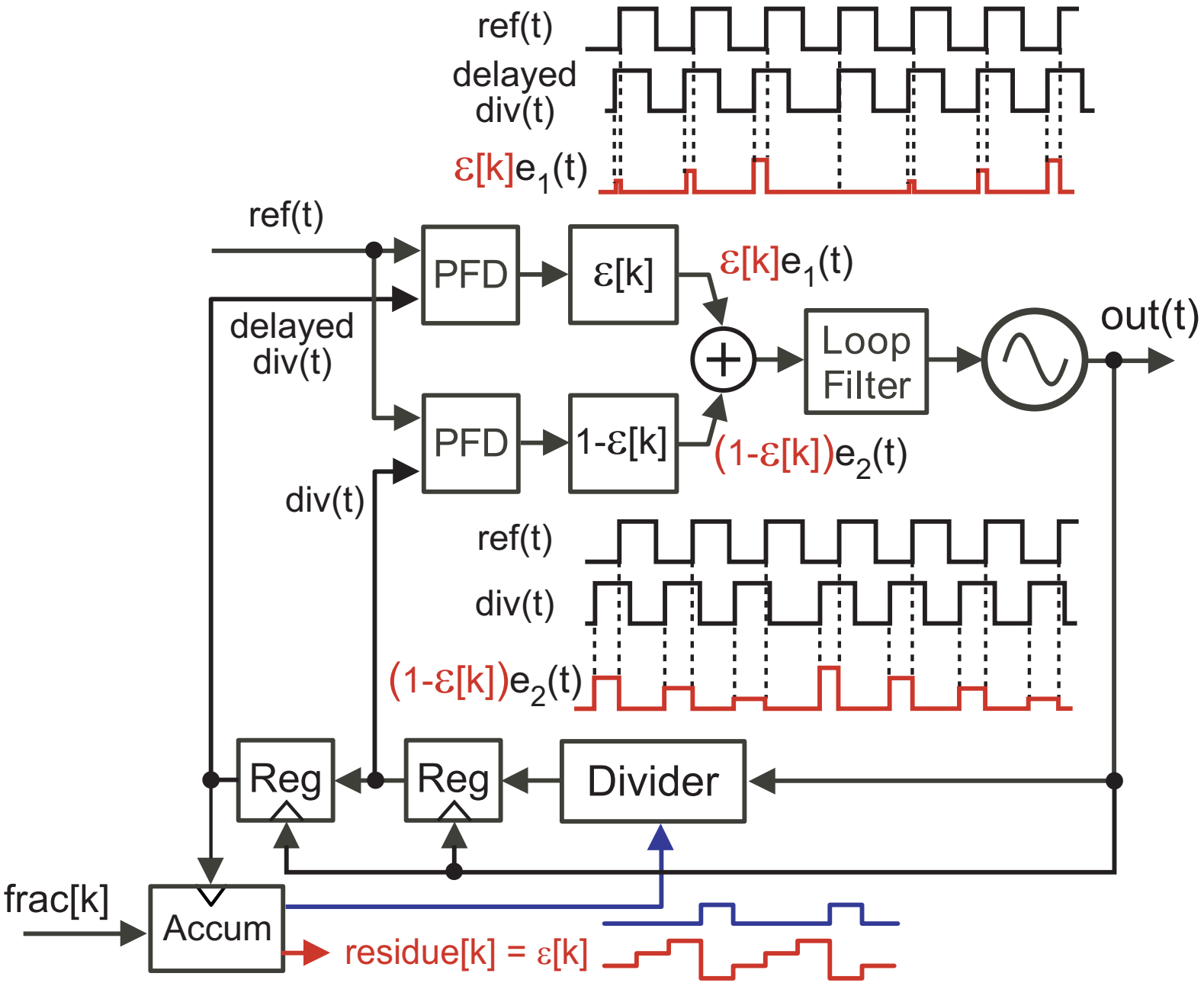
- **We begin with the basic fractional-N structure**



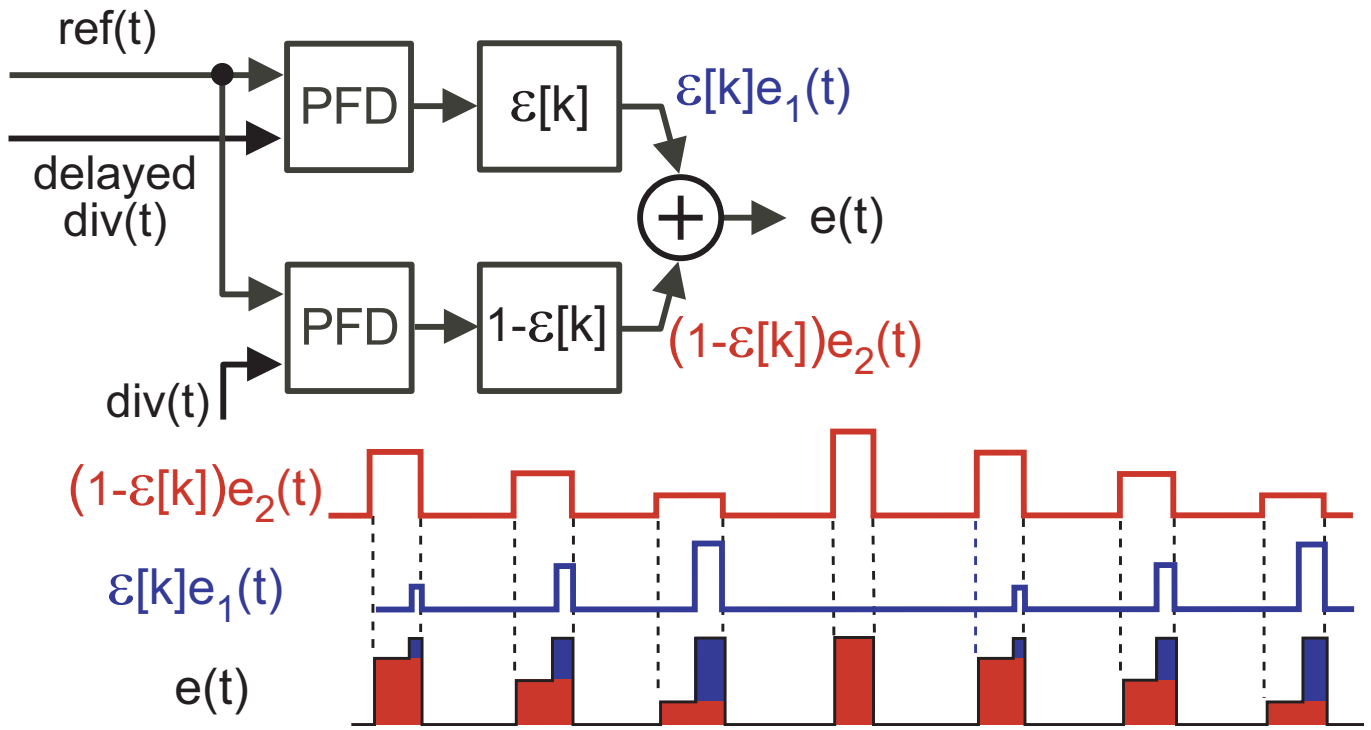
Add a Second PFD with Delayed Divider Signal



Scale Error Pulses According to Accumulator Residue



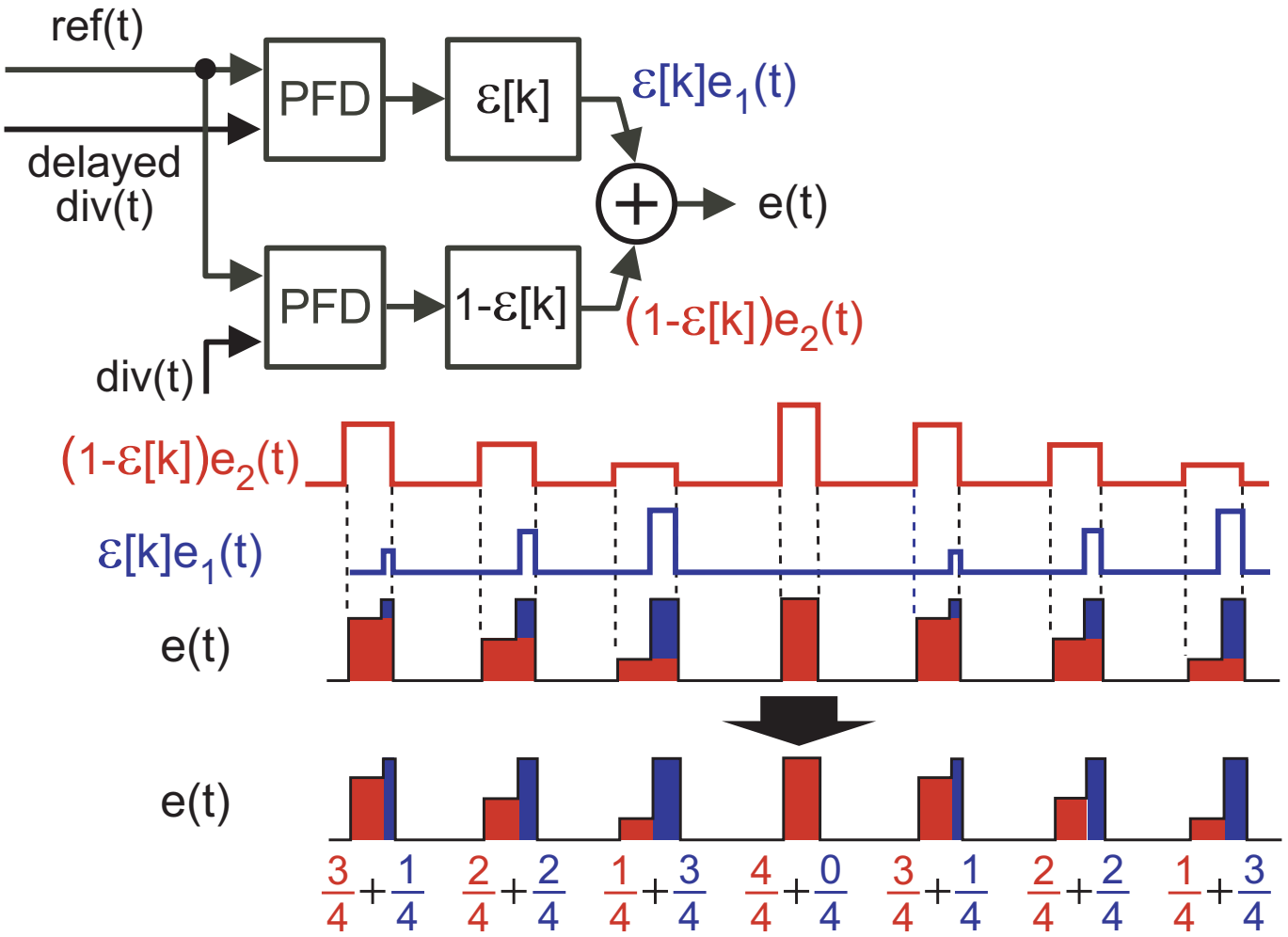
A Closer Look at Adding the Scaled Error Pulses



- **Goal – keep area constant for each pulse**
  - **It's easier to see this from a slightly different viewpoint**

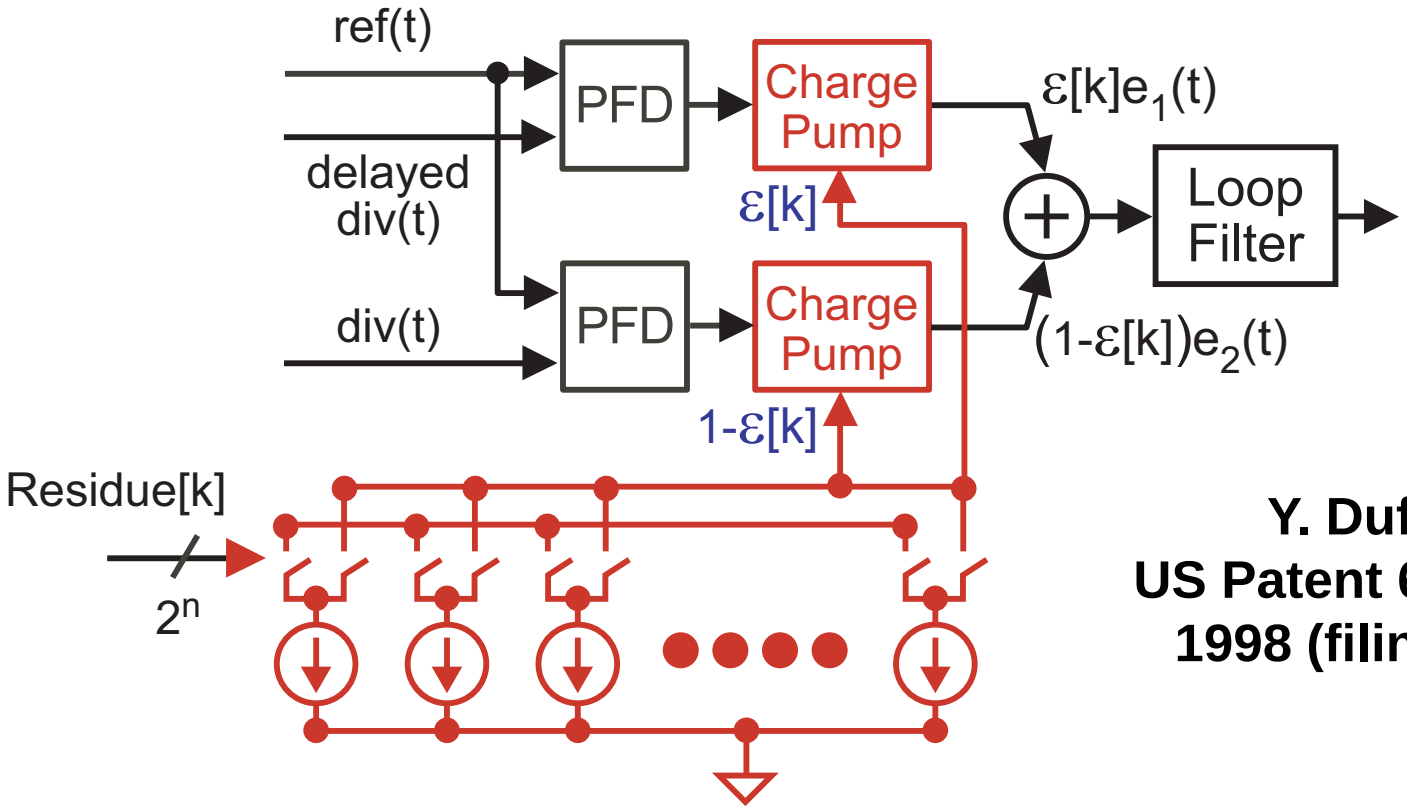
Alternate Viewpoint

- **The sum of scaled pulses can now be viewed as horizontal cancellation**



Implementation of Pulse Scaling Operation

- **Direct output of a differential current DAC into two charge pumps**



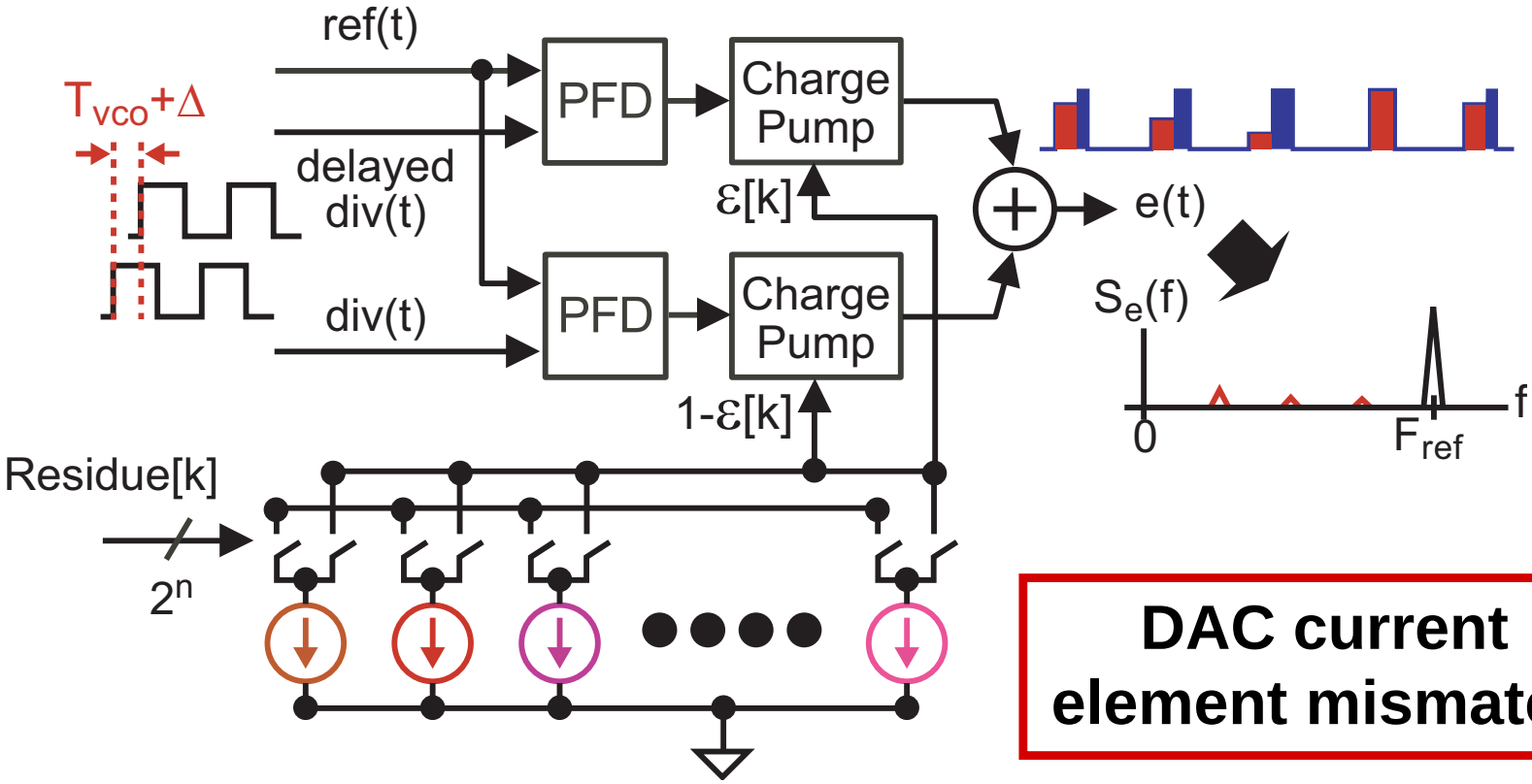
Y. Dufour
US Patent 6,130,561
1998 (filing date)

- **Issue: practical non-idealities kill performance**

Primary Non-idealities of Concern

Delay mismatch

Incomplete Fractional Spur Suppression

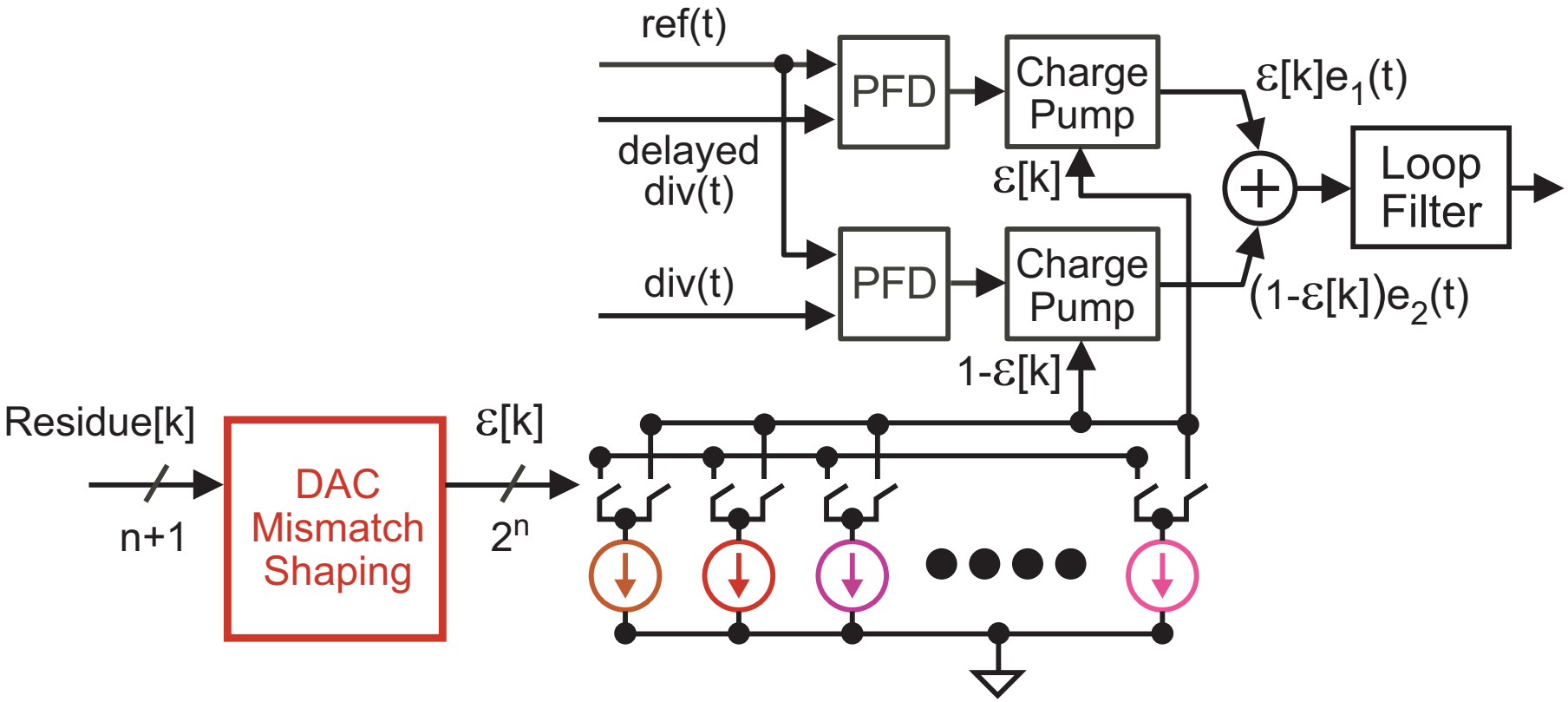


DAC current element mismatch

Proposed approach: dramatically reduce impact of these non-idealities using mixed-signal processing techniques

Eliminate Impact of DAC Current Element Mismatch

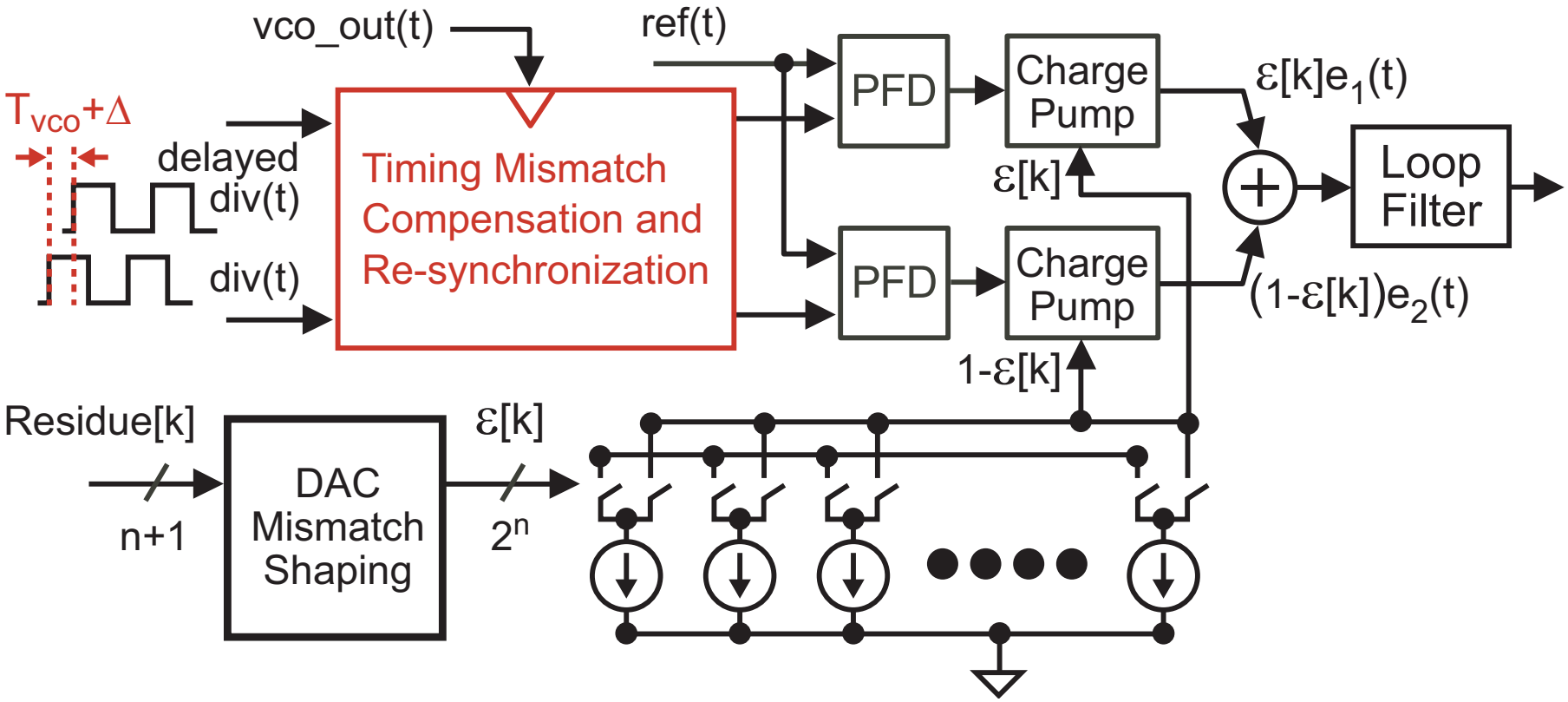
- **Apply standard DAC noise shaping techniques to shape mismatch noise to high frequencies**
  - **See Baird and Fiez, TCAS II, Dec 1995**



- **Allows up to 5% mismatch between unit elements without degrading our desired performance targets**

Eliminate Impact of Timing Mismatch

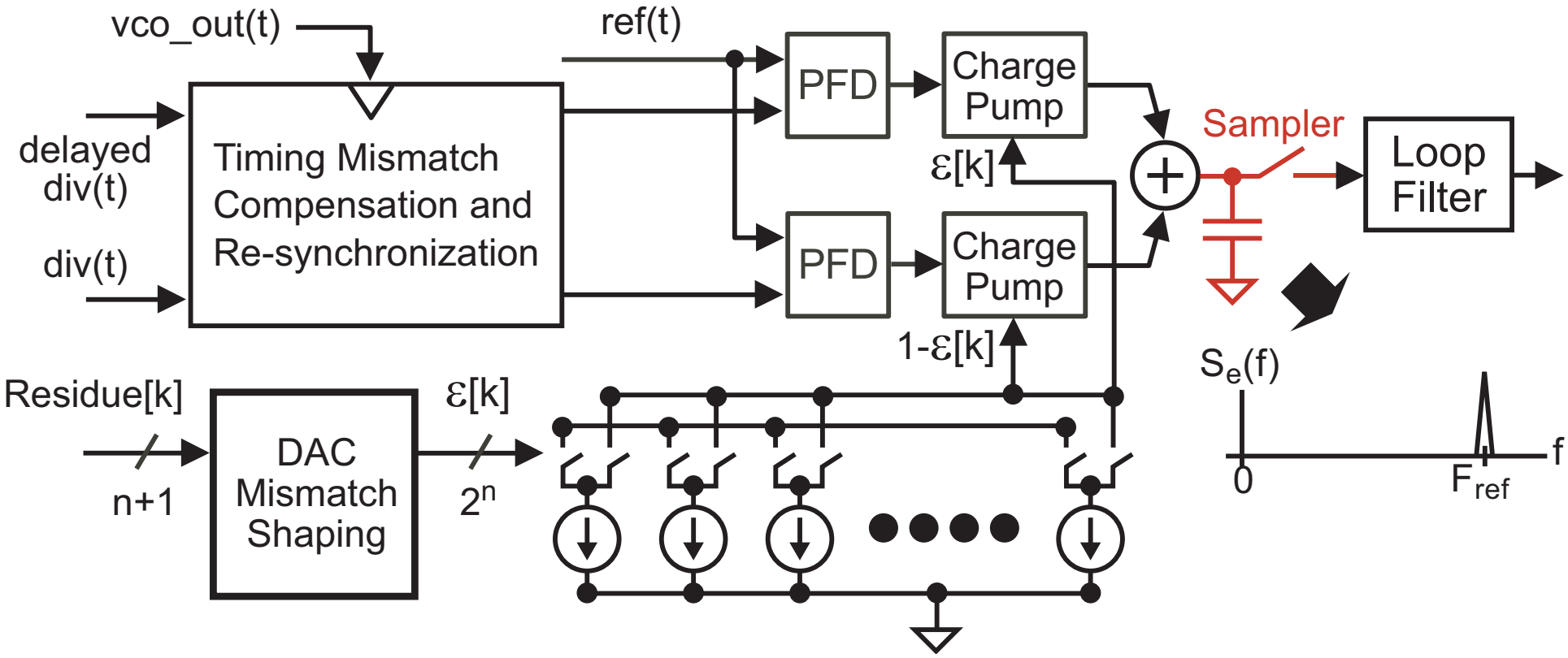
- **Swap paths between divider outputs in a pseudo-random fashion**
  - **Need to also swap ε[k] and 1-ε[k] sequence**



- **Allows up to 5 ps mismatch without degrading our desired performance targets**

Improve Horizontal Cancellation Performance

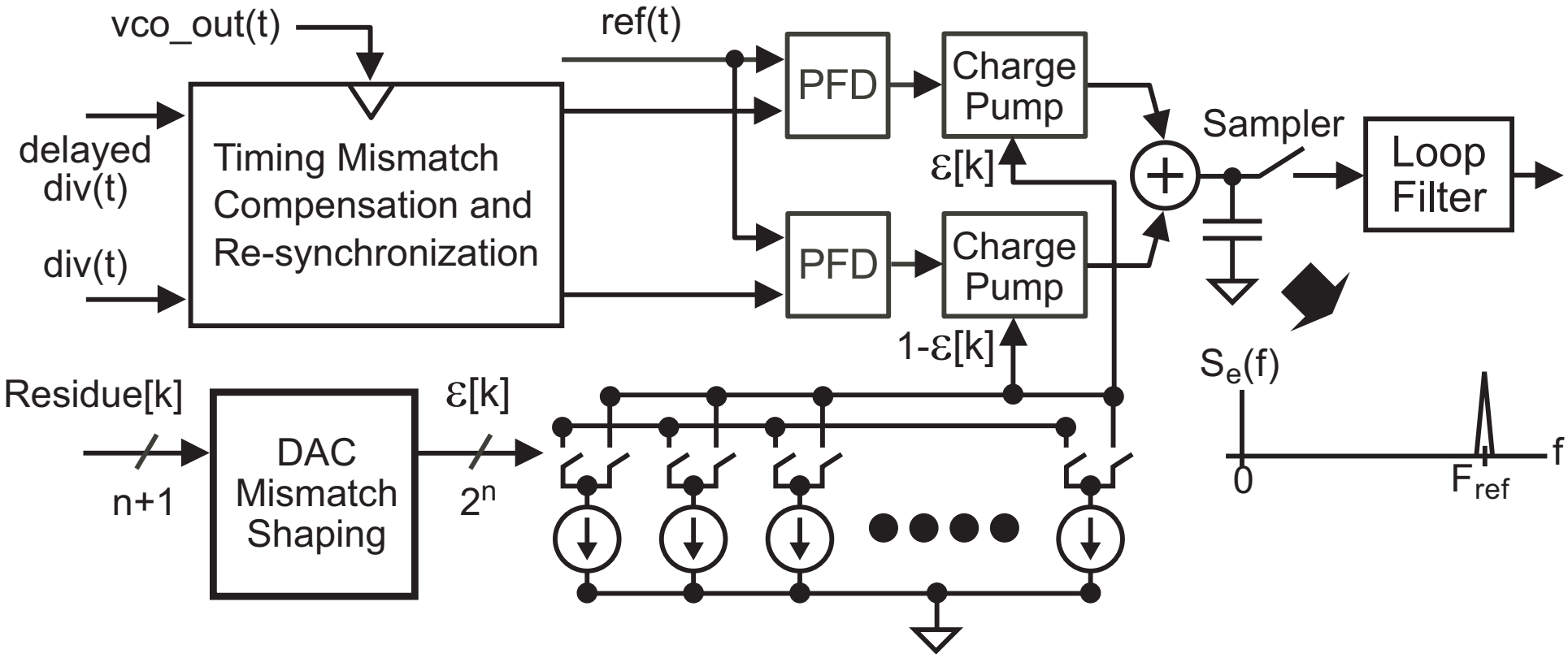
- **Sampling circuit accumulates error pulses before passing their information to the loop filter**
  - **A common analog trick used for decades**



- **Eliminates issue of having non-square error pulse shapes**

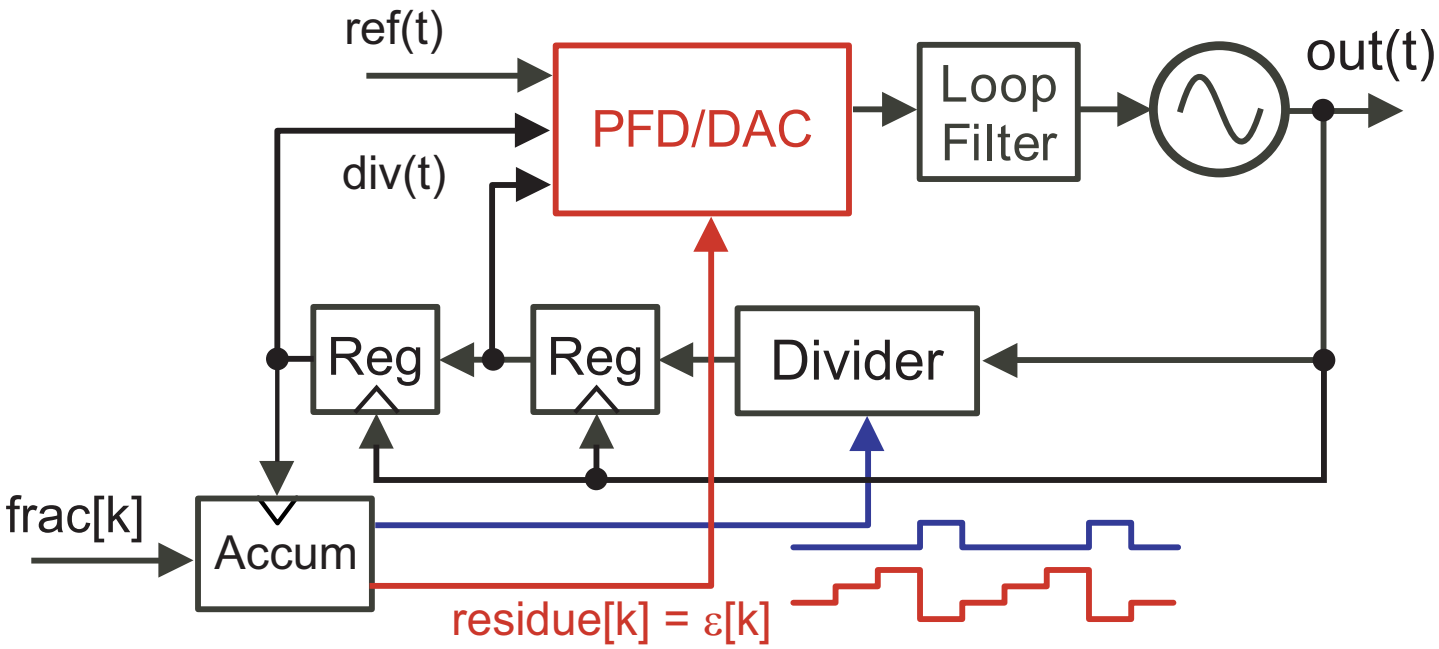
For More Details on This Approach

- **Theory and simulations presented in TCAS II paper**
  - **Meninger and Perrott, TCASII, Nov 2003**



Design and Simulation

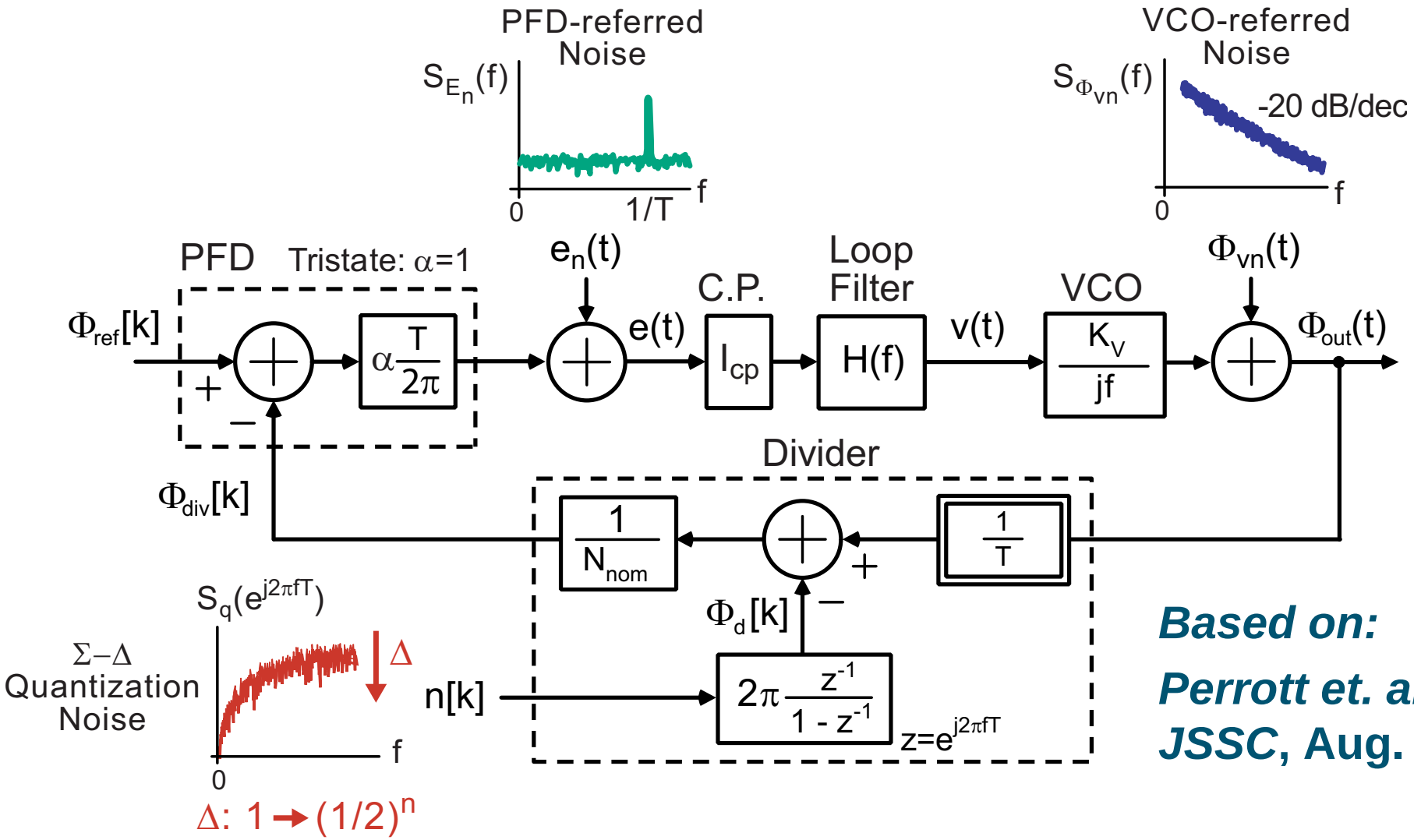
Design and Simulation of 'PFD/DAC' Synthesizer



- **Step 1: Derive analytical model**
- **Step 2: Design at system level**
- **Step 3: Simulate at system level (CppSim)**
- **Step 4: Simulate at transistor level (SPICE)**

Iterate between all of these steps in practice

Analytical Model of 'PFD/DAC' Fractional-N PLL

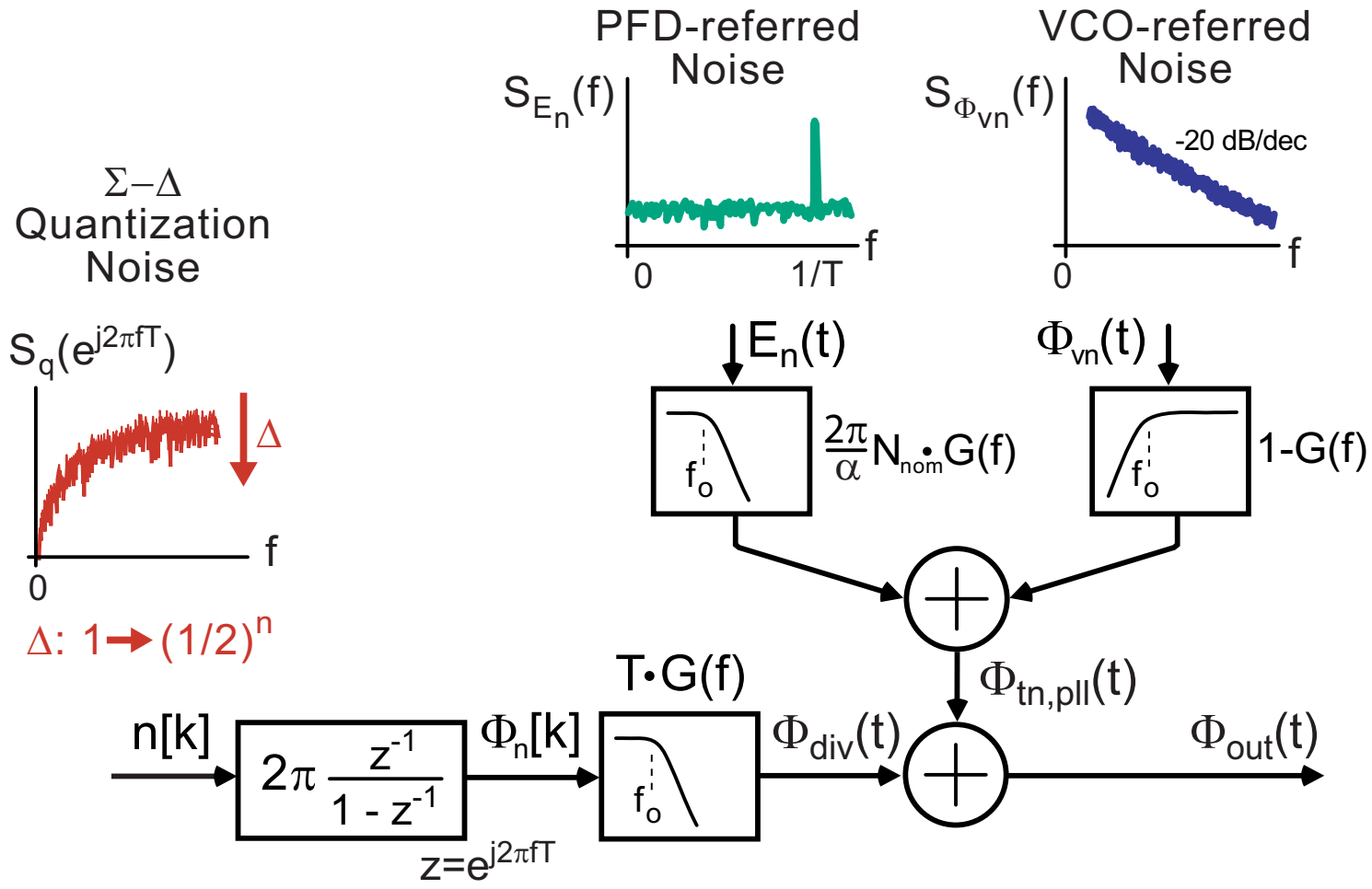


Based on:

Perrott et. al., JSSC, Aug. 2002

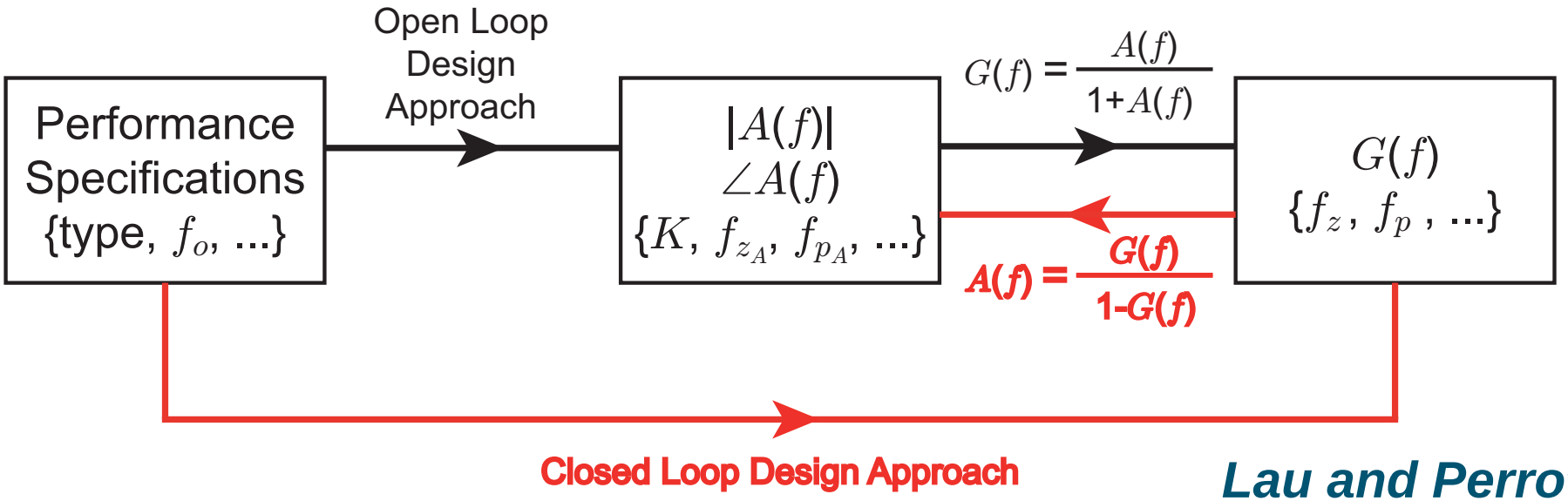
n-bit PFD/DAC → Δ is reduced from 1 to $(1/2)^n$

Parameterized PLL Model



$$G(f) = \frac{A(f)}{1 + A(f)} \quad \text{where} \quad A(f) = \frac{\alpha I_{cp} H(f) K_V}{N_{nom} 2\pi j f}$$

Closed Loop Design Approach

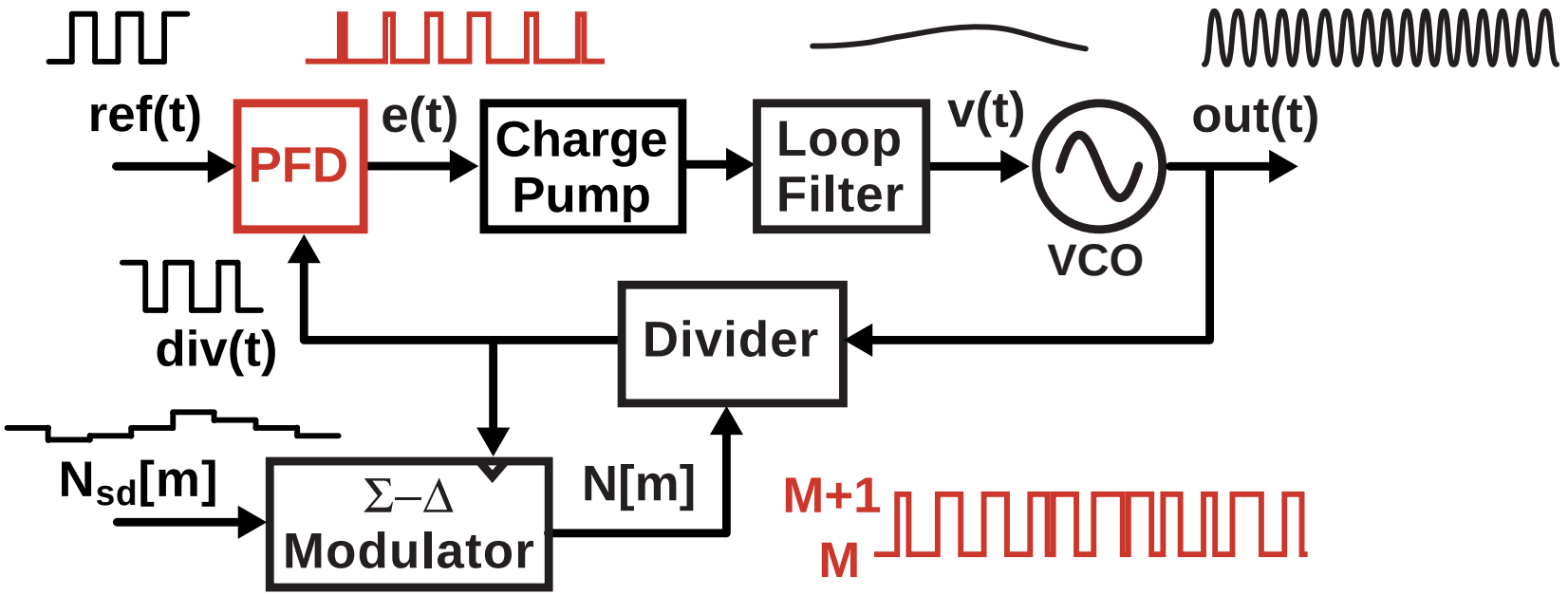


***Lau and Perrott, DAC*, June 2003**

- **Classical approach**
  - **Indirectly design $G(f)$ using bode plots of $A(f)$**
- **Proposed approach**
  - **Directly design $G(f)$**
  - **Solve for $A(f)$ that will achieve desired $G(f)$**

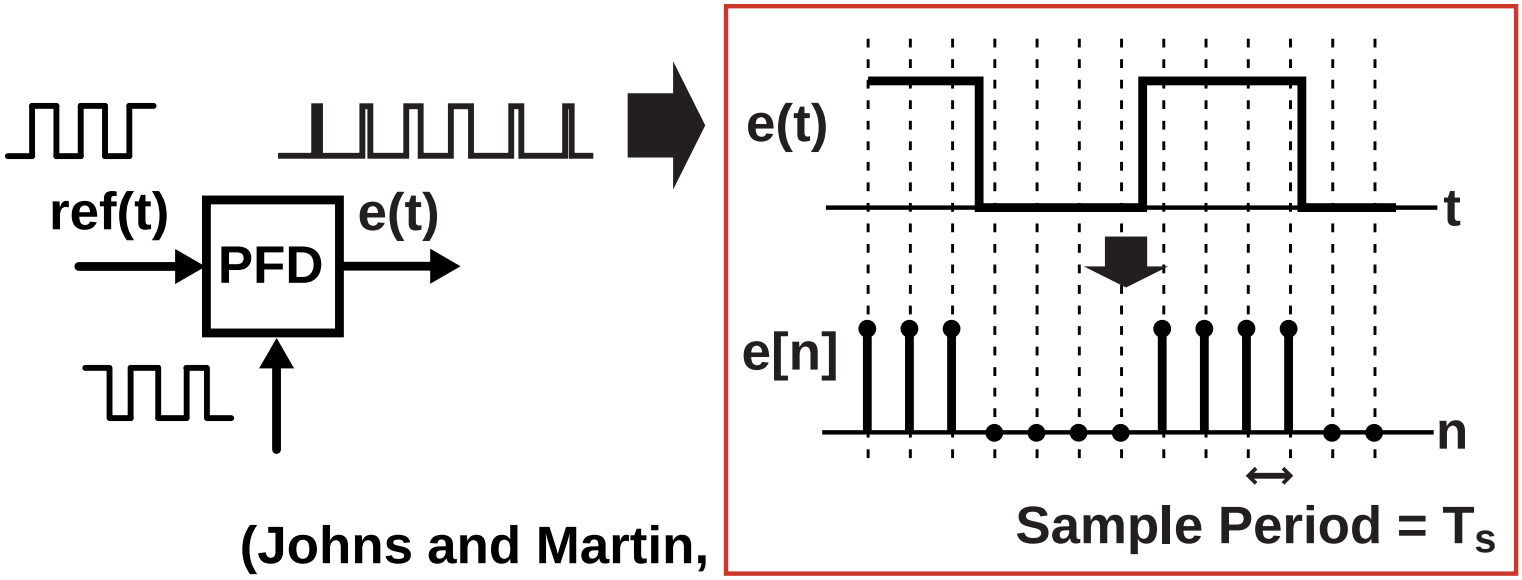
Implemented in *PLL Design Assistant* Software

PLL Simulation Issues (Behavioral Level)



- Slow simulation time
  - High output frequency ⇨ High sample rate
  - Long time constants ⇨ Long time span for transients
- Inaccurate results
  - PFD output information conveyed in pulses

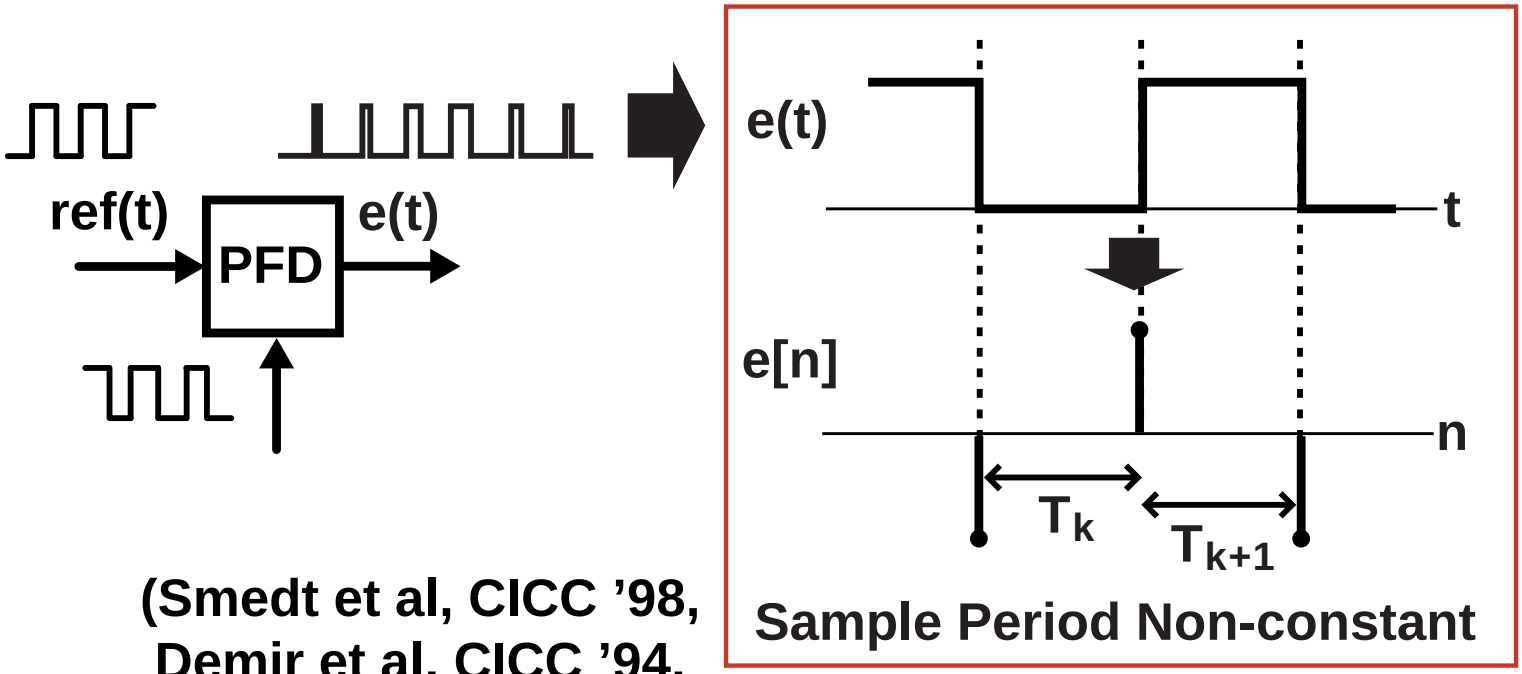
Example: Classical Constant-Time Step Method



(Johns and Martin, Analog Integrated Circuit Design)

- Directly sample the PFD output according to the simulation sample period
  - Simple, fast, readily implemented in Matlab, Verilog, C++
- Issue – quantization noise is introduced
  - This noise overwhelms the PLL noise sources we are trying to simulate

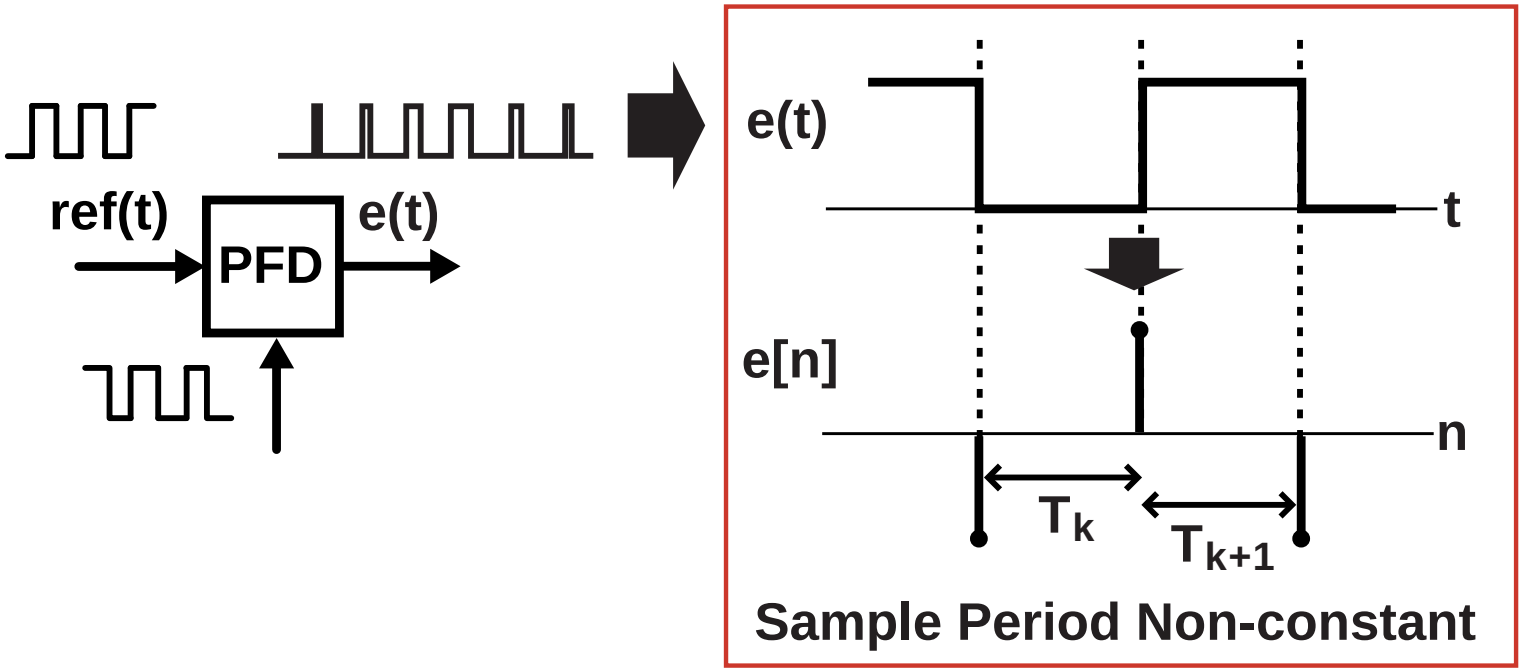
Alternative: Event Driven Simulation



**(Smedt et al, CICC '98,
Demir et al, CICC '94,
Hinz et al, Circuits and Systems '00)**

- **Set simulation time samples at PFD edges**
  - **Sample rate can be lowered to edge rate!**

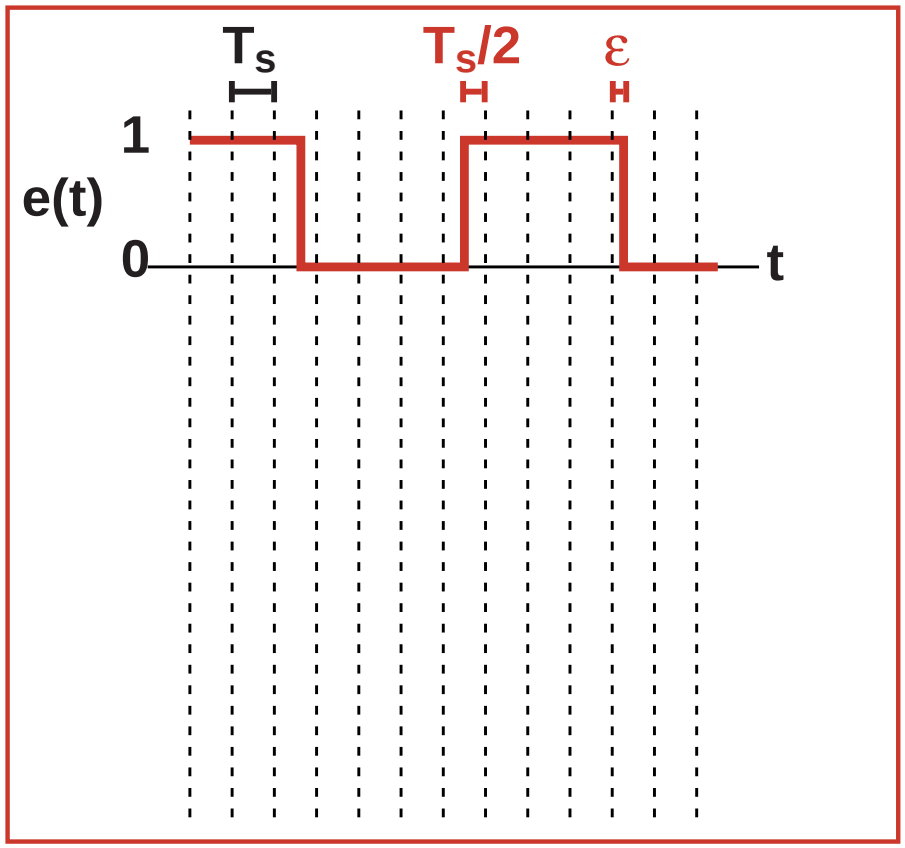
Issue: Non-Constant Time Step Brings Complications

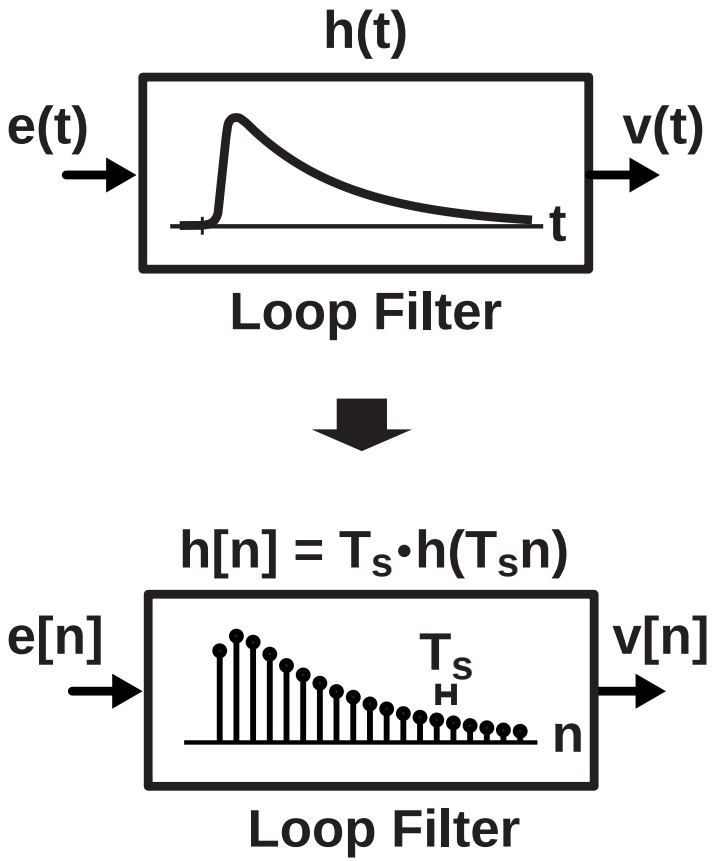


- **Filters and noise sources must account for varying time step in their code implementations**
- **Spectra derived from mixing and other operations can display false simulation artifacts**
- **Setting of time step becomes progressively complicated if multiple PLL's simulated at once**

Is there a better way?

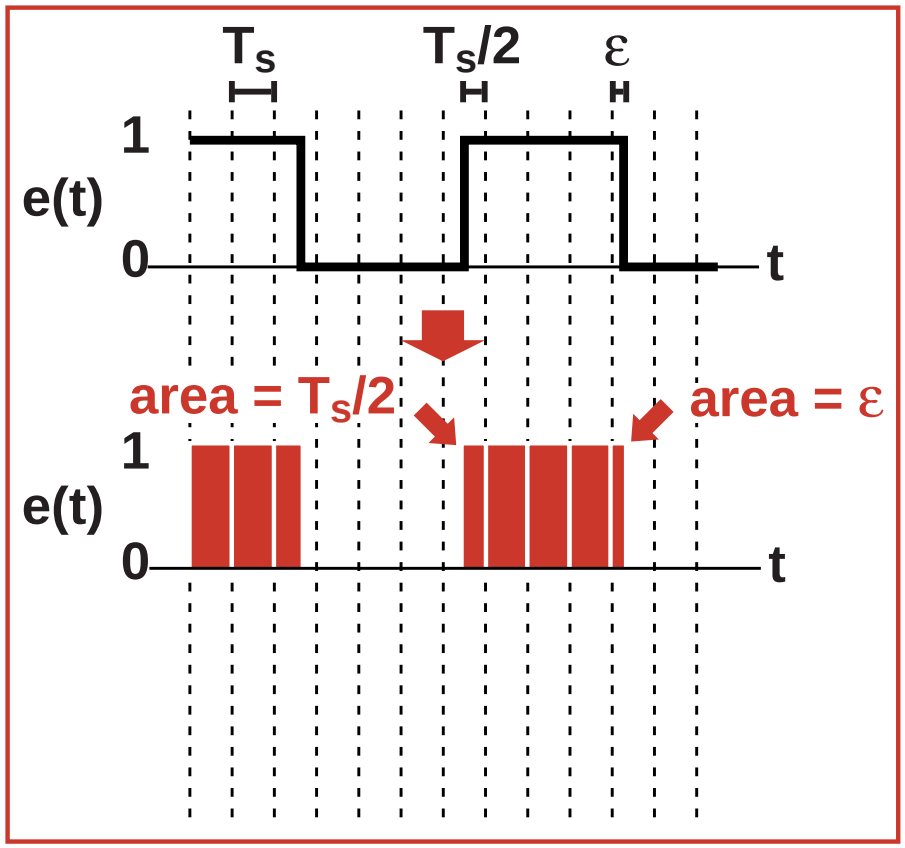
Problem: Quantization Noise at PFD Output

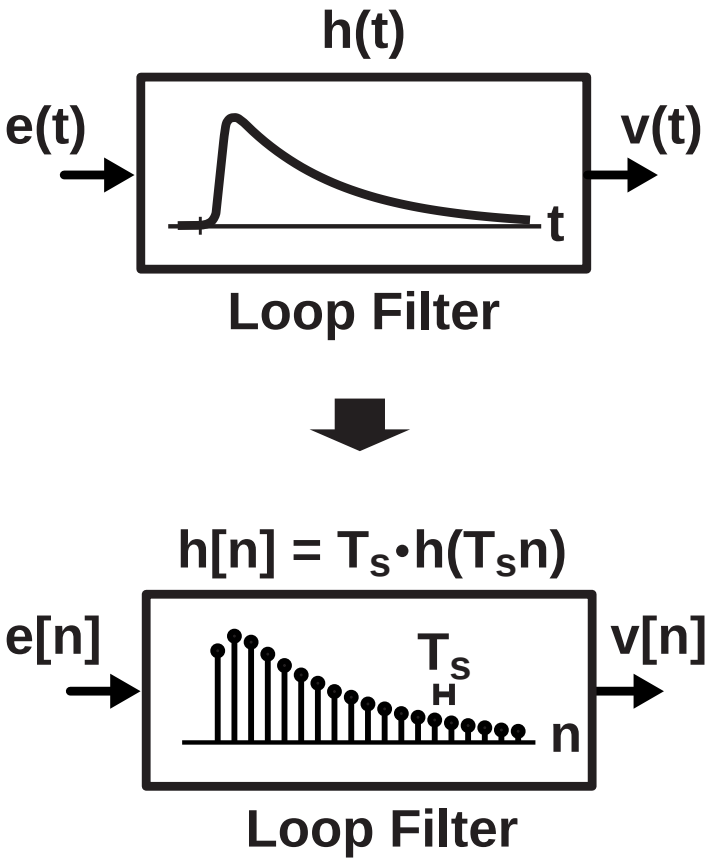




- **Edge locations of PFD output are quantized**
  - **Resolution set by time step: T_s**
- **Reduction of T_s leads to long simulation times**

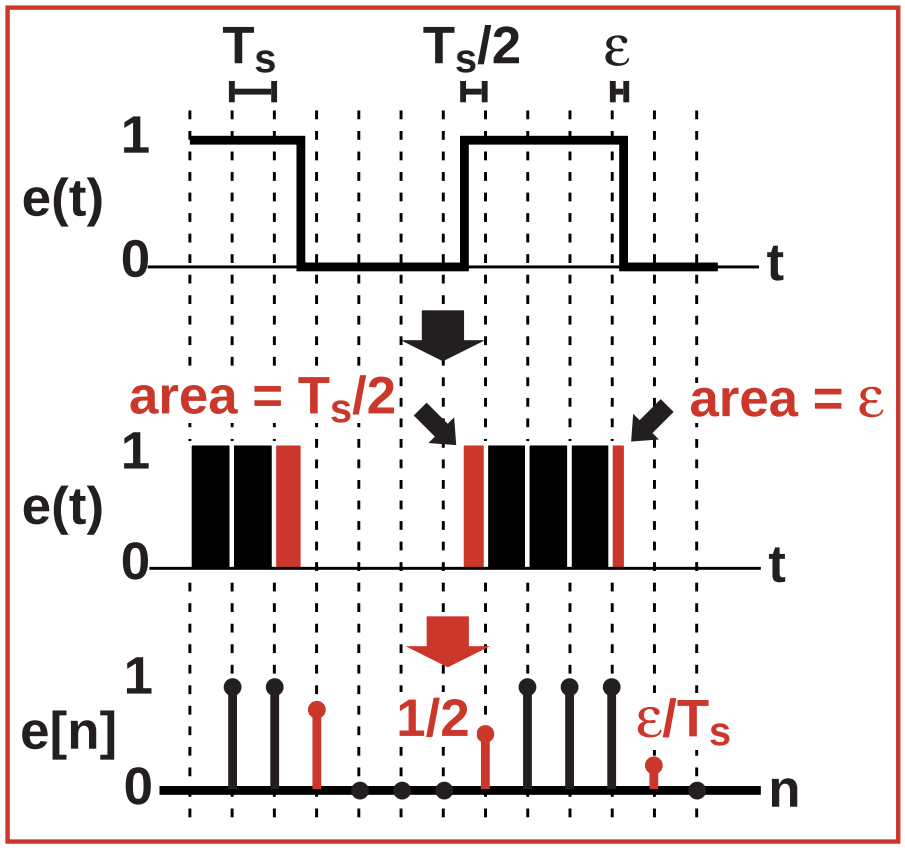
Proposed Approach: View as Series of Pulses

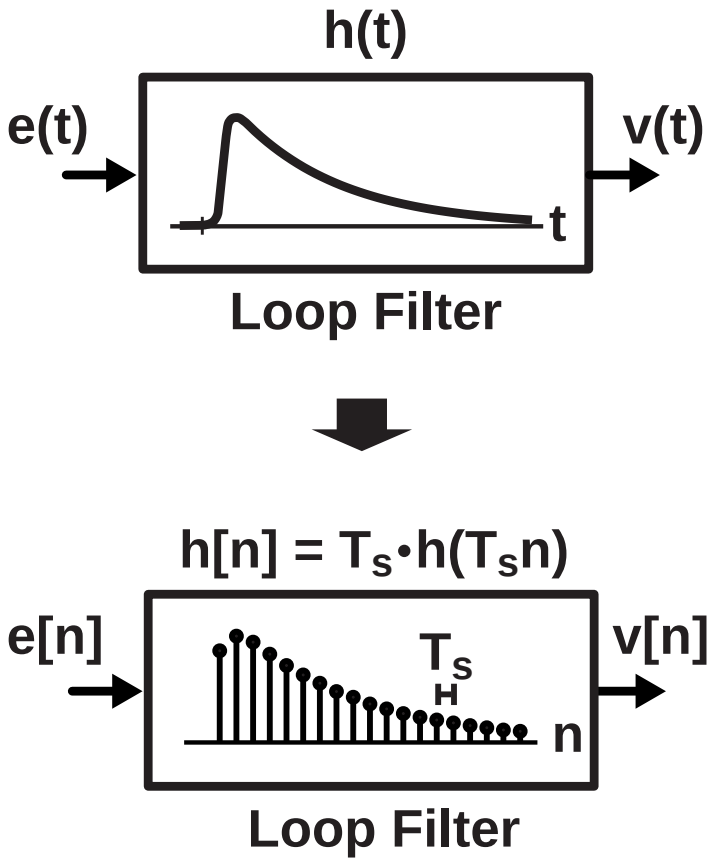




- Area of each pulse set by edge locations
- Key observations:
  - Pulses look like impulses to loop filter
  - Impulses are parameterized by their area and time offset

Proposed Method





Perrott, DAC, **June 2002**

- **Set e[n] samples according to pulse areas**
  - **Leads to very accurate results and fast computation**
- **Implemented in the CppSim Behavioral Simulator**

Application:
A 1 MHz Bandwidth Fractional-N Frequency Synthesizer Implementation

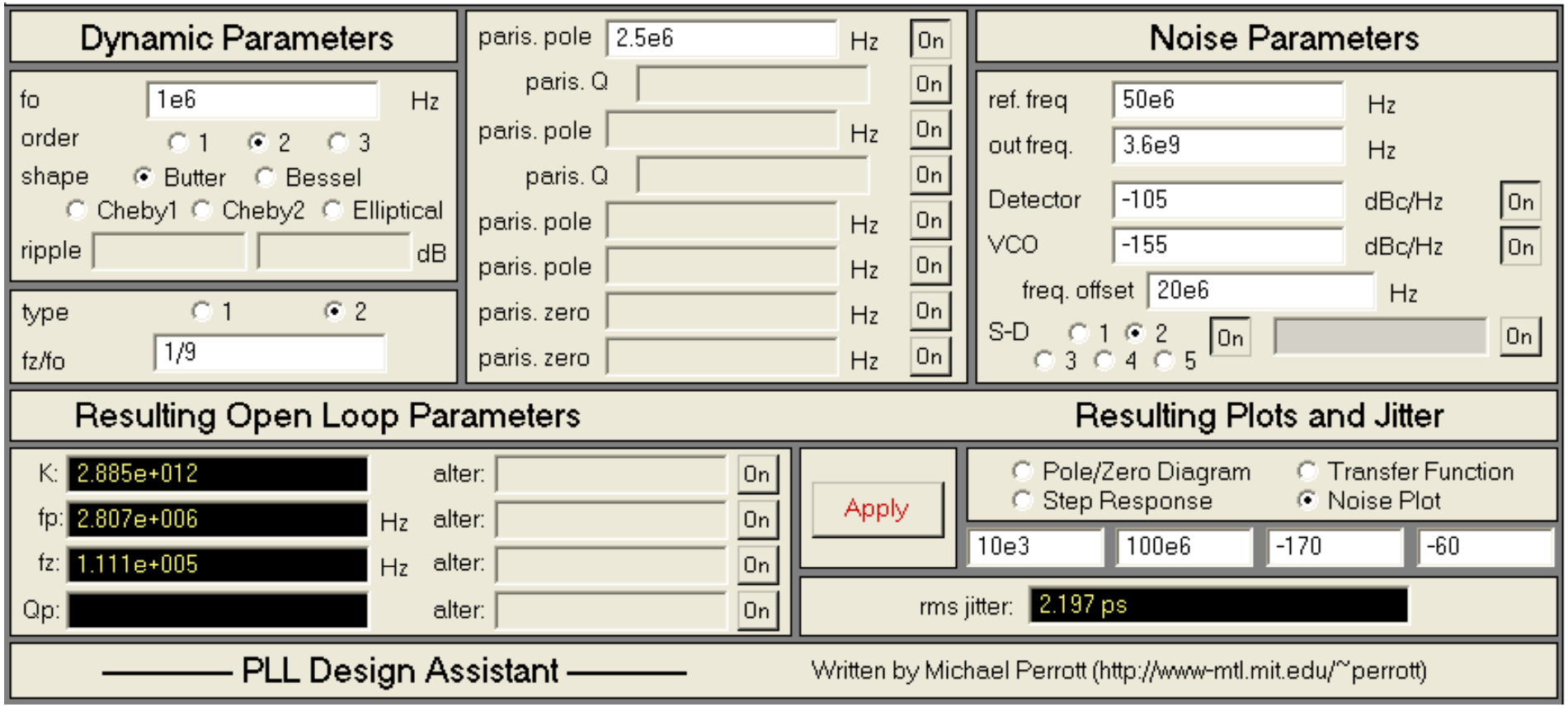
Design Goals

- **Output frequency: 3.6 GHz**
  - **Allows dual-band output (1.8 GHz and 900 MHz)**
- **Reference frequency: 50 MHz**
  - **Allows low cost crystal reference**
- **Bandwidth: 1 MHz**
  - **Allows fast settling time and ~1 Mbit/s modulation rate**
- **Noise: < -150 dBc/Hz at 20 MHz offset (3.6 GHz carrier)**
  - **Phase noise at the 20 MHz frequency offset is very challenging for GSM and DCS transmitters**
    - GSM: -162 dBc/Hz at 20 MHz offset (900 MHz carrier)
    - DCS: -151 dBc/Hz at 20 MHz offset (1.8 GHz carrier)

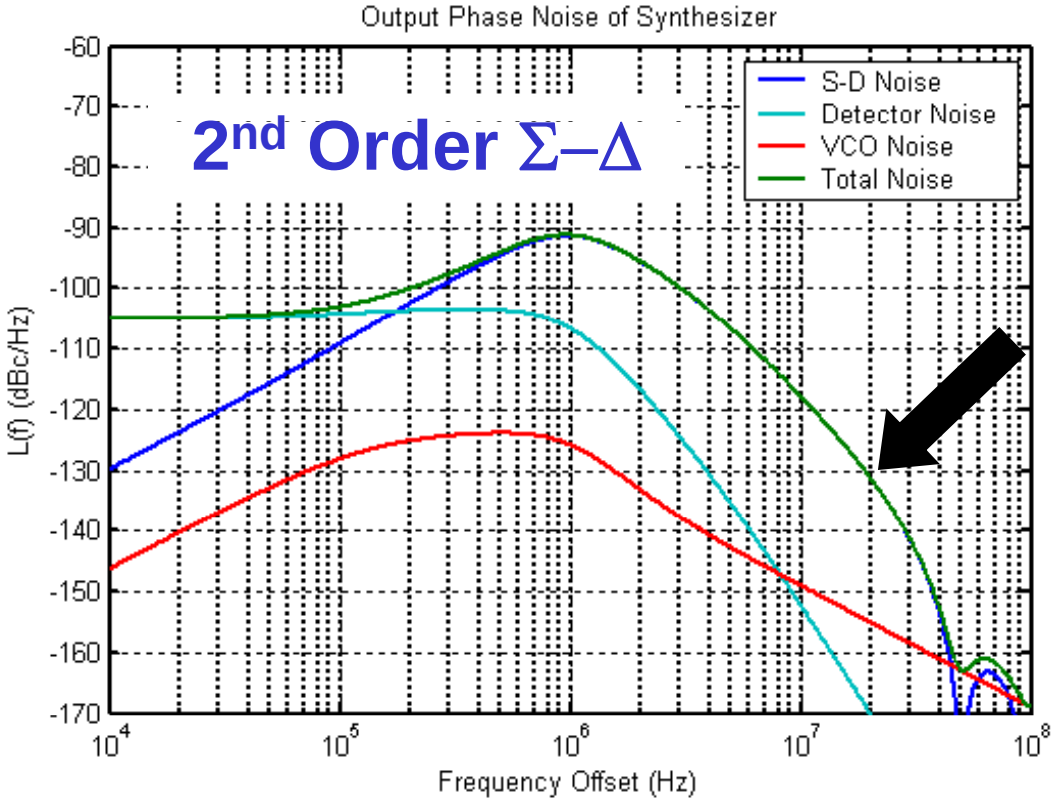
Simultaneous achievement of the above bandwidth and noise targets is very challenging

Evaluate Noise Performance with 1 MHz PLL BW

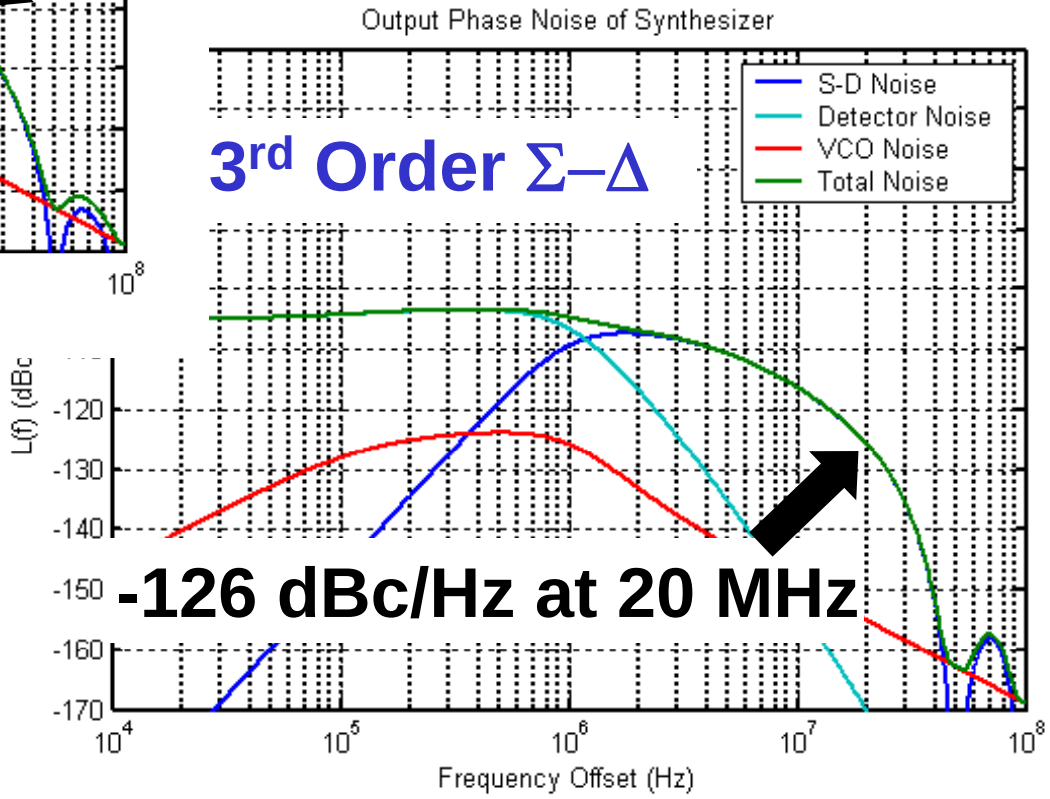
- **G(f) parameters**
  - **1 MHz BW, Type II, 2nd order rolloff, extra pole at 2.5 MHz**
- **Required PLL noise parameters (with a few dB of margin)**
  - **Output-referred charge pump noise: -105 dBc/Hz**
  - **VCO noise: -155 dBc/Hz at 20 MHz offset (3.6 GHz carrier)**



Calculated Phase Noise for Classical Fractional-N



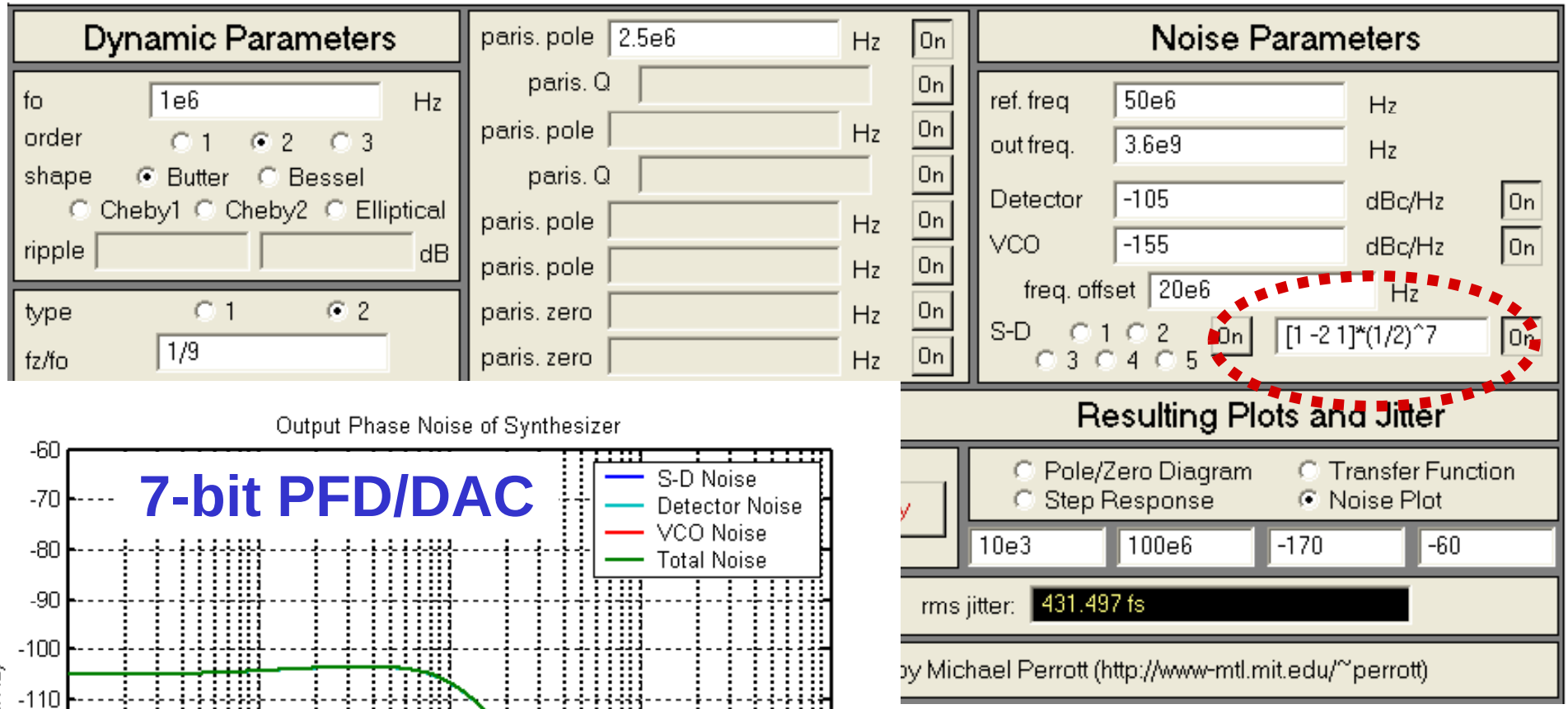
-132 dBc/Hz at 20 MHz

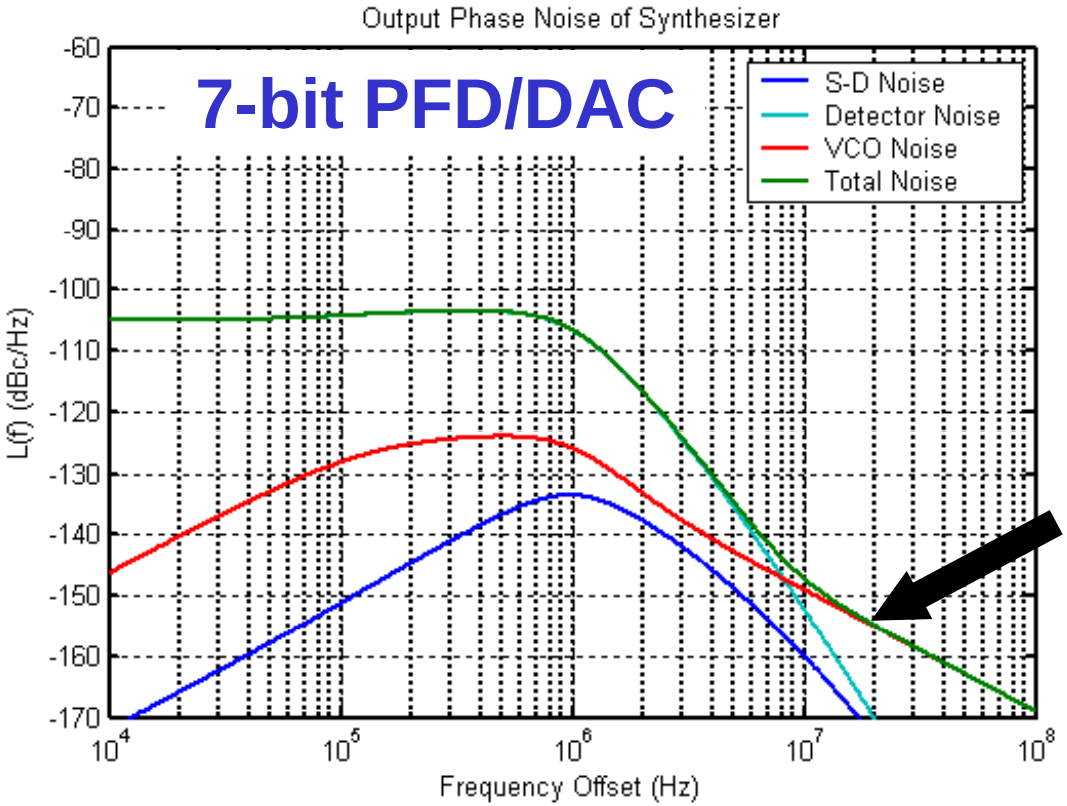


-126 dBc/Hz at 20 MHz

These do NOT meet our target of -150 dBc/Hz at 20 MHz (3.6 GHz carrier freq.)

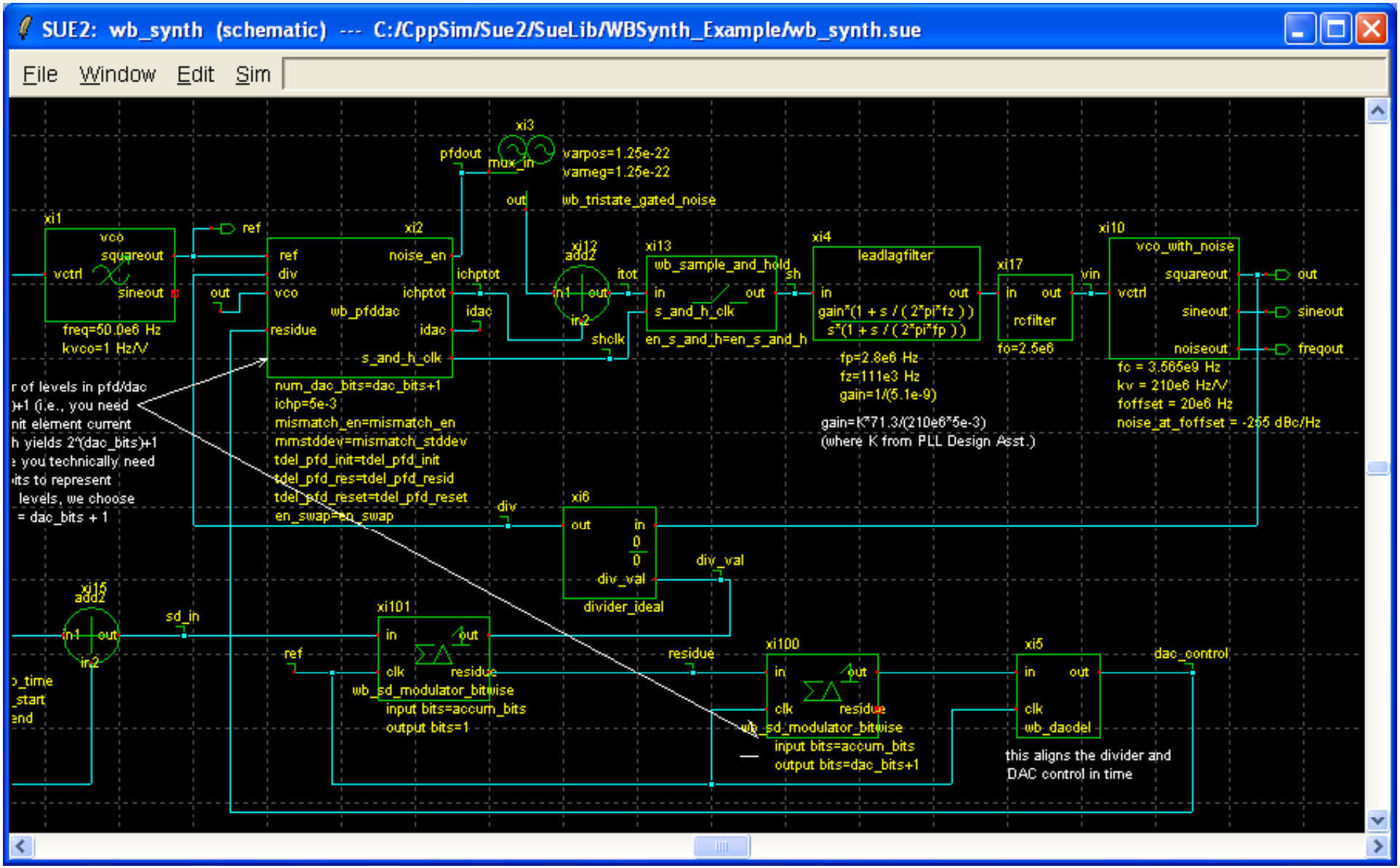
Calculated Phase Noise for 7-bit PFD/DAC Synth





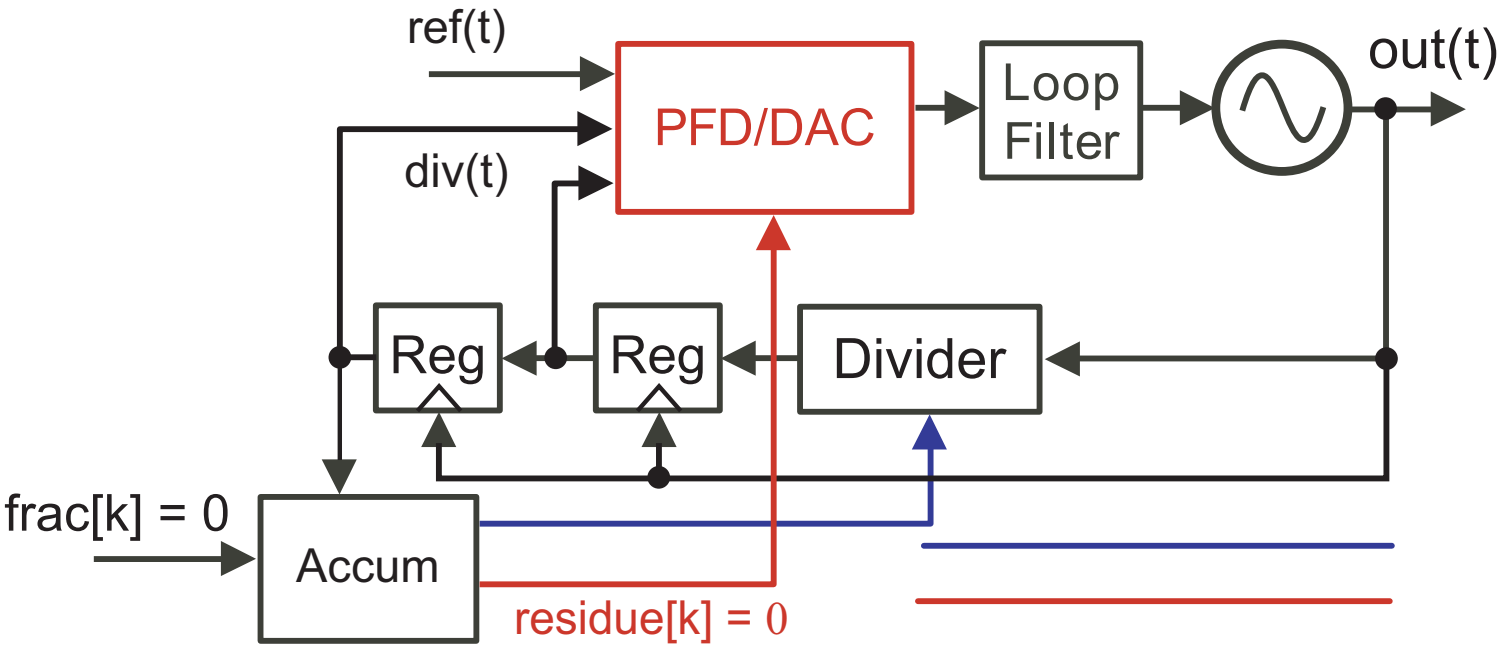
-155 dBc/Hz at 20 MHz !

Simulation of PFD/DAC Synthesizer using CppSim



- **Phase noise plots to follow: 40e6 time steps, < 15 min**

Intrinsic PLL Noise Performance



- **Set fractional portion of divide value to zero**
  - **Leads to residue variation of zero**
    - No quantization noise!
- **Need to calculate and simulate impact of detector and VCO noise**

In essence, operate as integer-N synthesizer

Calculate Intrinsic PLL Noise Sources

- **Estimate detector noise (dominated by charge pump)**
  - **From SPICE Simulation, $S_{Icp}(f)$ = Duty*1.25e-22 A^2/Hz**
  - **Output-referred PLL noise due to above noise:**

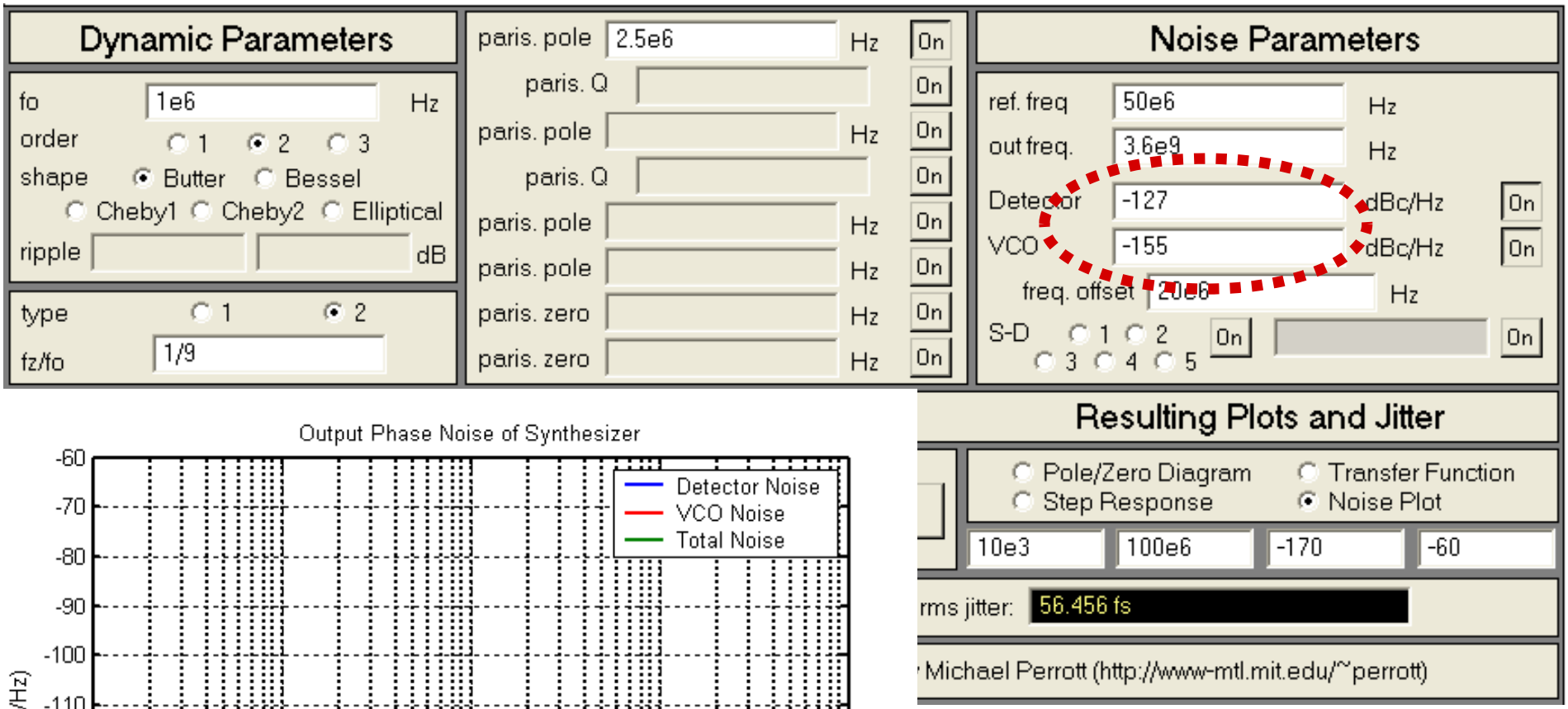
$$S_{\Phi_{out}}(f) = \left(\frac{1}{I_{cp}}\right)^2 \left(\frac{2\pi}{\alpha} N_{nom}\right)^2 |G(f)|^2 \, S_{Icp}(f)$$

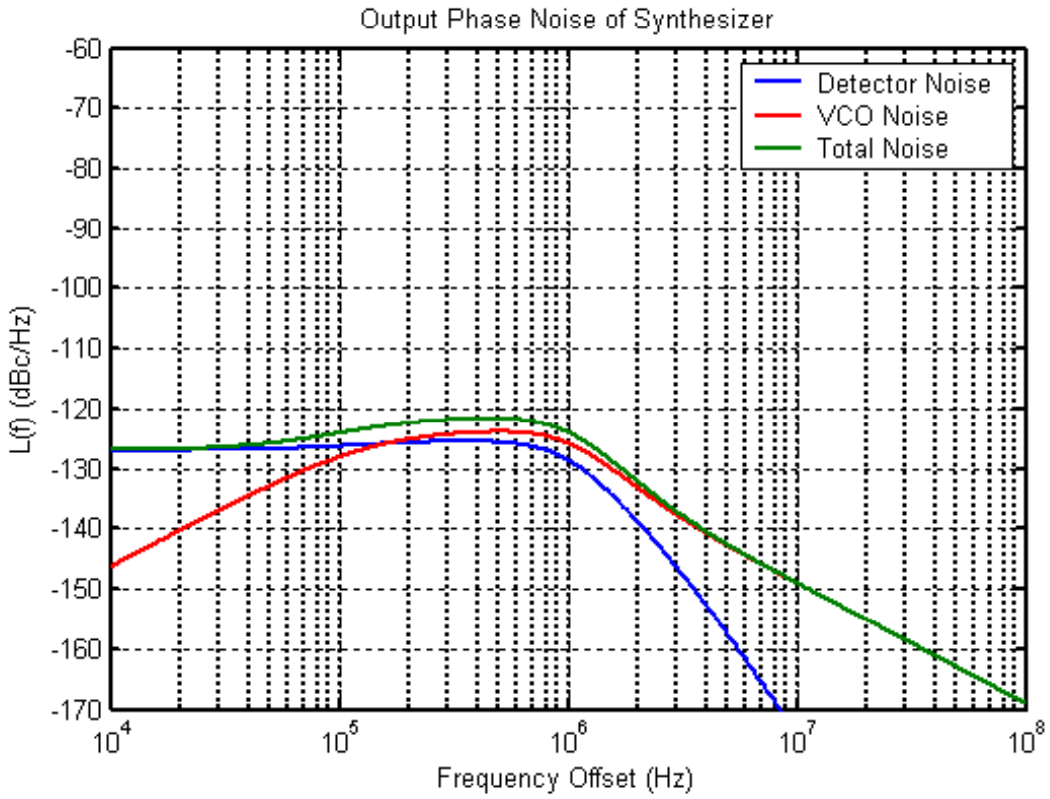
$$= \left(\frac{1}{5 \cdot 10^{-3}}\right)^2 \left(\frac{2\pi}{1} 71.3\right)^2 |G(f)|^2 \, 0.2 \cdot 1.25 \cdot 10^{-22}$$

$$\Longrightarrow 10 \log(S_{\Phi_{out}}(f)) = -127 \, |G(f)|^2 \, \text{dBc/Hz}$$

- **Estimate VCO noise**
  - **For off-chip VCO, examine data sheet:**
    - In this case, $S_{\Phi vco}$ = -155 dBc/Hz at 20 MHz offset
  - **For on-chip VCO, use Spectre RF or other CAD tool**

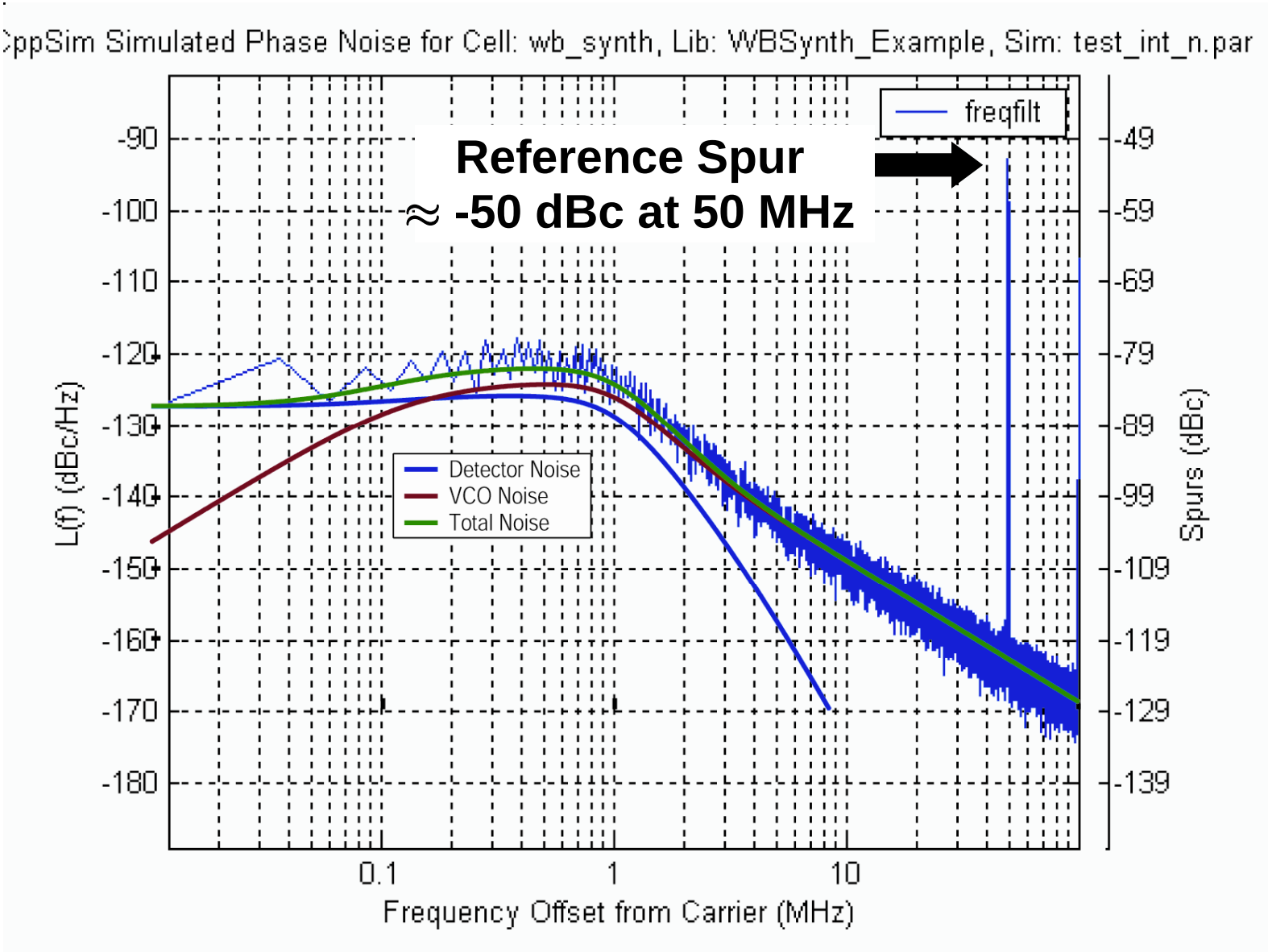
Calculate PLL Noise Due to Intrinsic Noise Sources





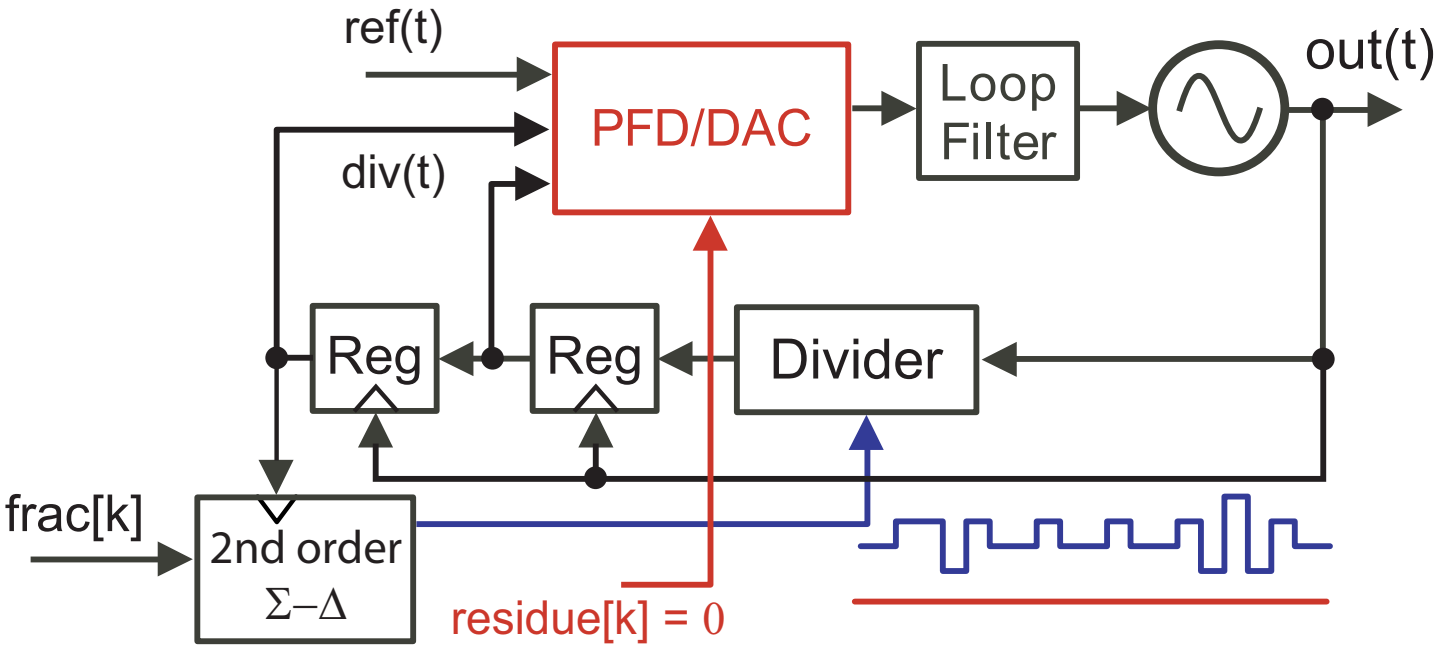
- **We will see that we will need to include:**
  - **Excess noise**
  - **1/f noise**

Simulated Phase Noise due to Intrinsic Noise Sources



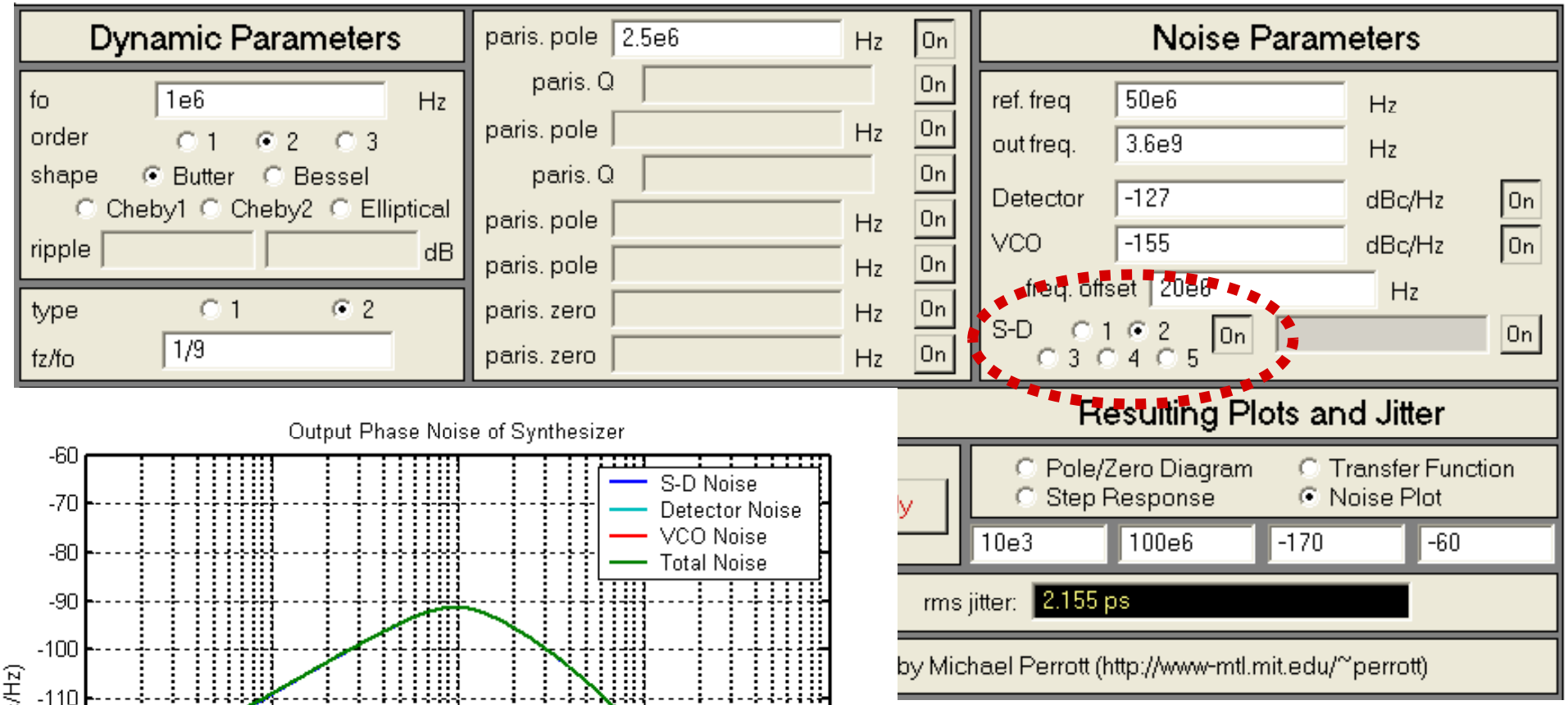
- **PLL Design Assistant accurately models simulated noise!**

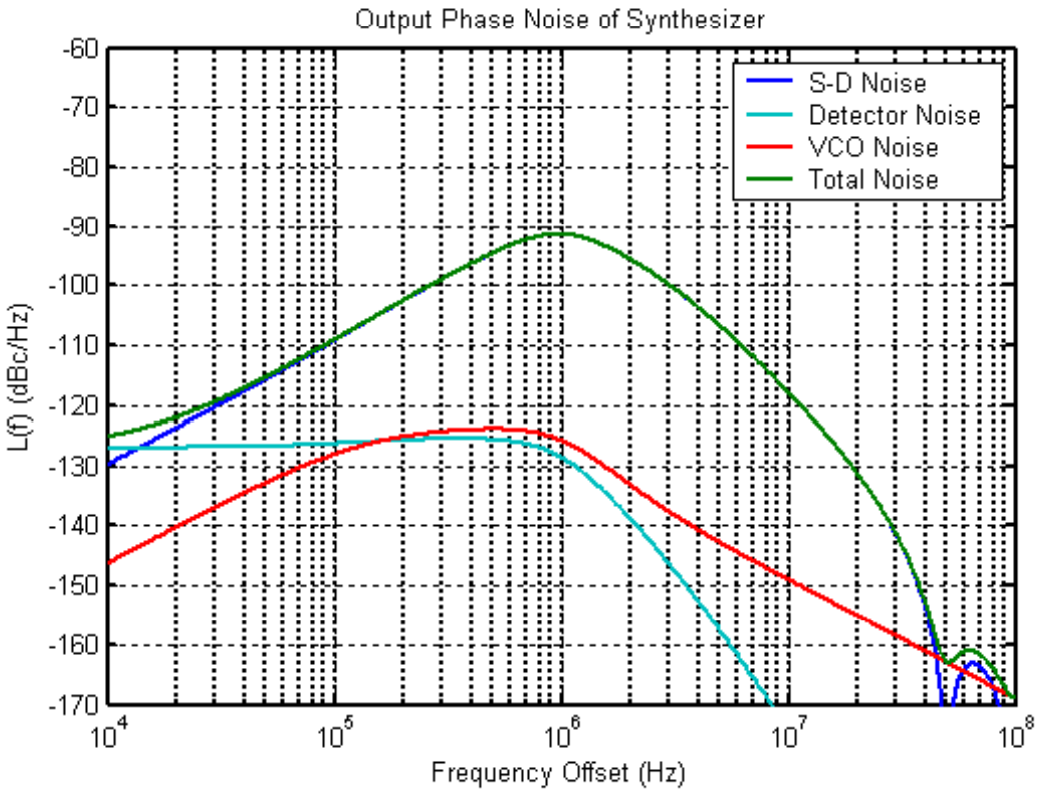
2nd Order Σ–Δ Fractional-N Performance



- Replace accumulator with second order Σ–Δ modulator
- Set residue into PFD/DAC equal to zero

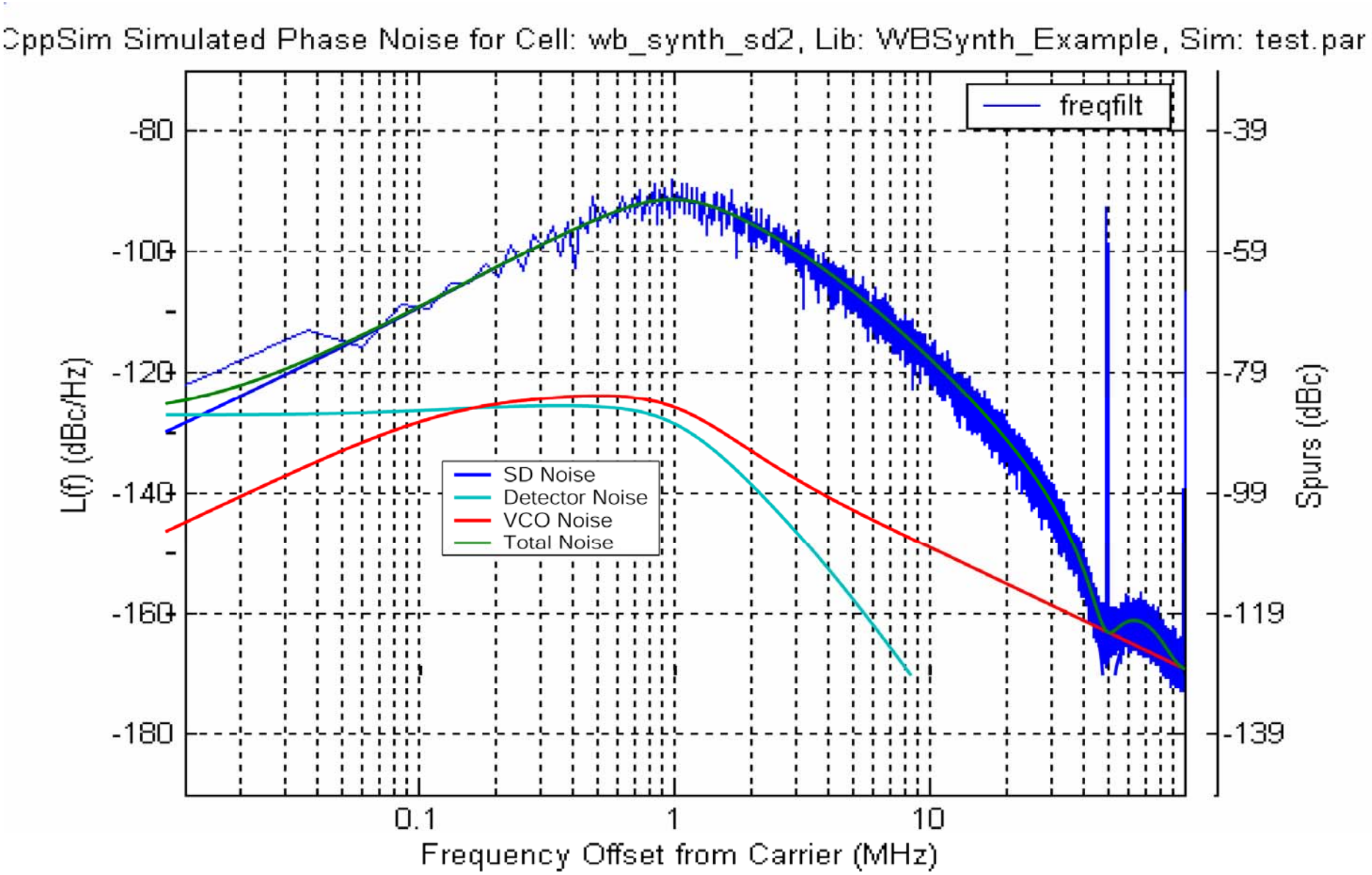
Calculate PLL Noise for 2nd Order Σ–Δ Synthesizer





- 2nd order Σ–Δ
  - Click on 2nd order S-D quantization noise in tool

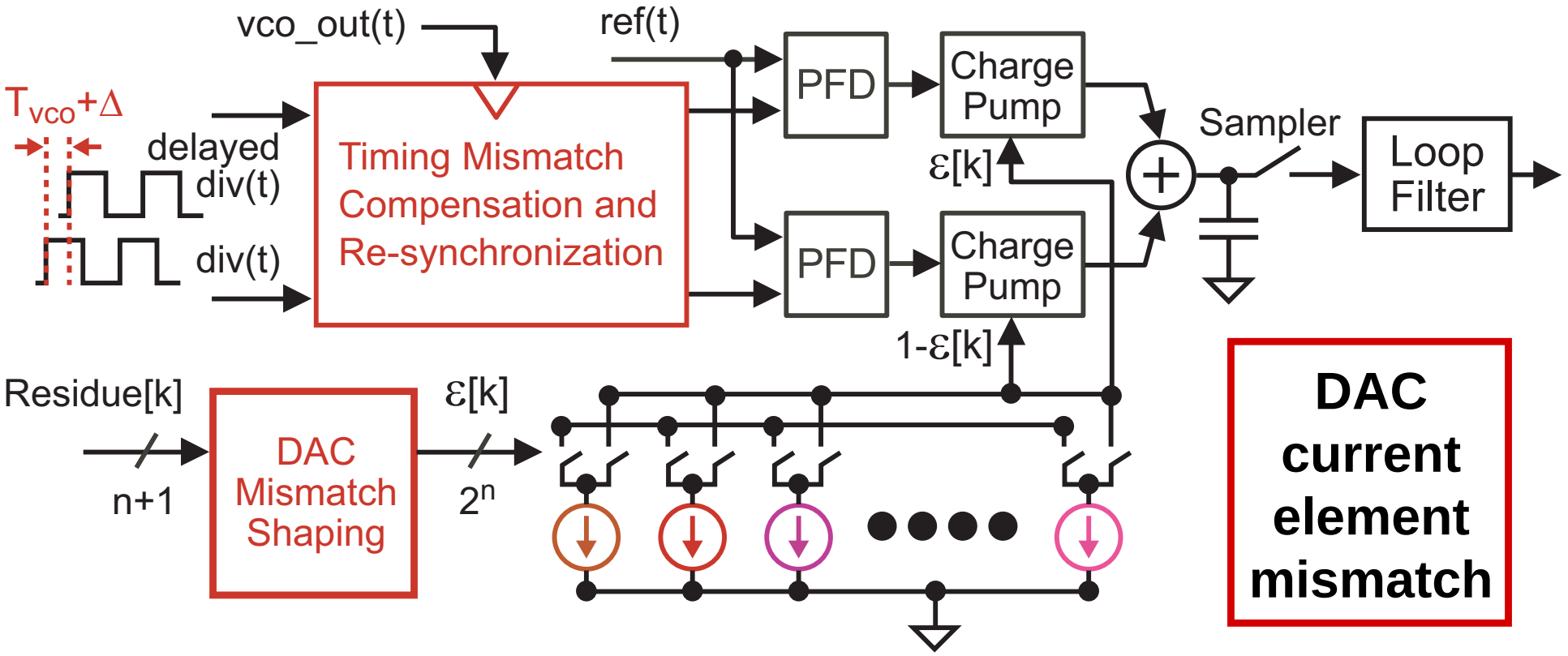
Simulated Phase Noise of 2nd Order Σ–Δ Synthesizer



- **PLL Design Assistant accurately models simulated noise!**

7-bit PFD/DAC Synthesizer Performance

Delay mismatch



DAC current element mismatch

Application of proposed noise scrambling/shaping techniques leads to broadband noise from delay and DAC current mismatch

Impact on PLL Noise due to Non-idealities of PFD/DAC

- **Impact of DAC mismatch**
  - **Lowers achievable quantization noise suppression**
  - **Negligible in this case**
- **Impact of delay mismatch**
  - **Model as white reference noise uniformly distributed from 0 to Δt**
    - Calculation for Δt = 5 ps:

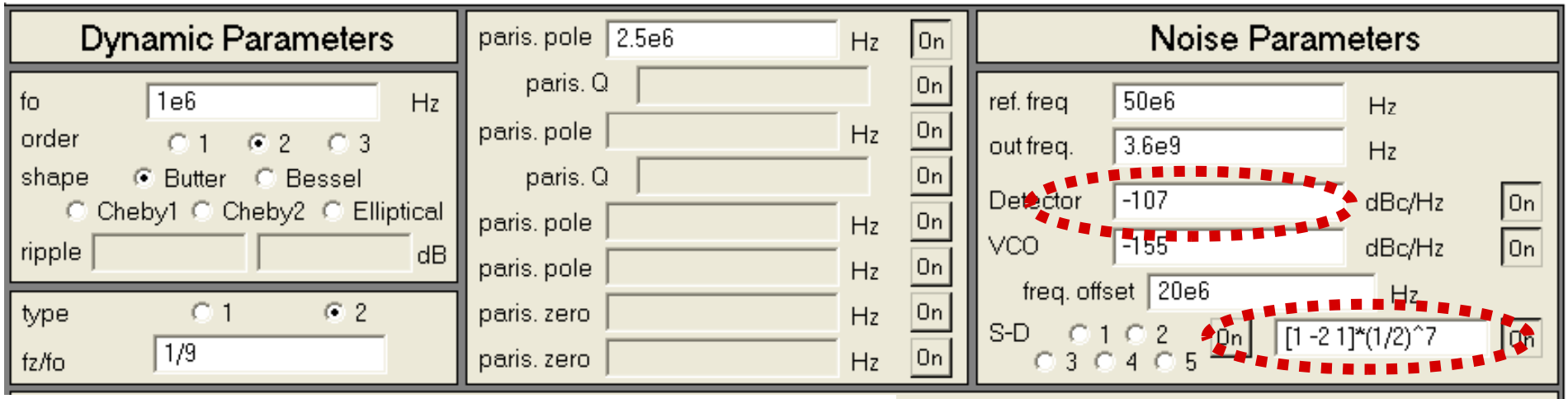
$$S_{\Phi_{out}}(f) = \frac{1}{T}\left|TN_{nom}G(f)\right|^2 S_{\Phi_{jit}}(e^{j2\pi fT})$$

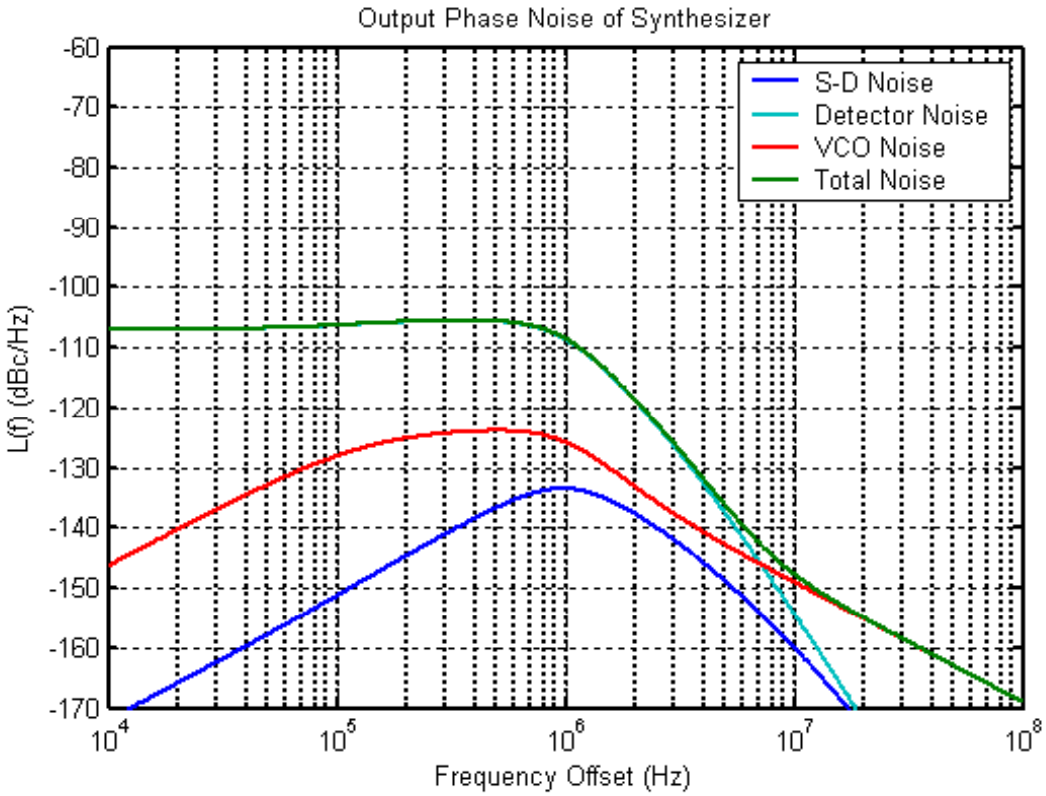
$$= \frac{1}{T}\left|TN_{nom}G(f)\right|^2 \left|\frac{2\pi}{T}\right|^2 \frac{(\Delta t)^2}{12}$$

$$= (50\ \text{MHz})(71.3)^2(2\pi)^2\frac{(5\text{ps})^2}{12}$$

$$\Longrightarrow 10\log(S_{\Phi_{out}}(f)) = -107\left|G(f)\right|^2 \text{dBc/Hz}$$

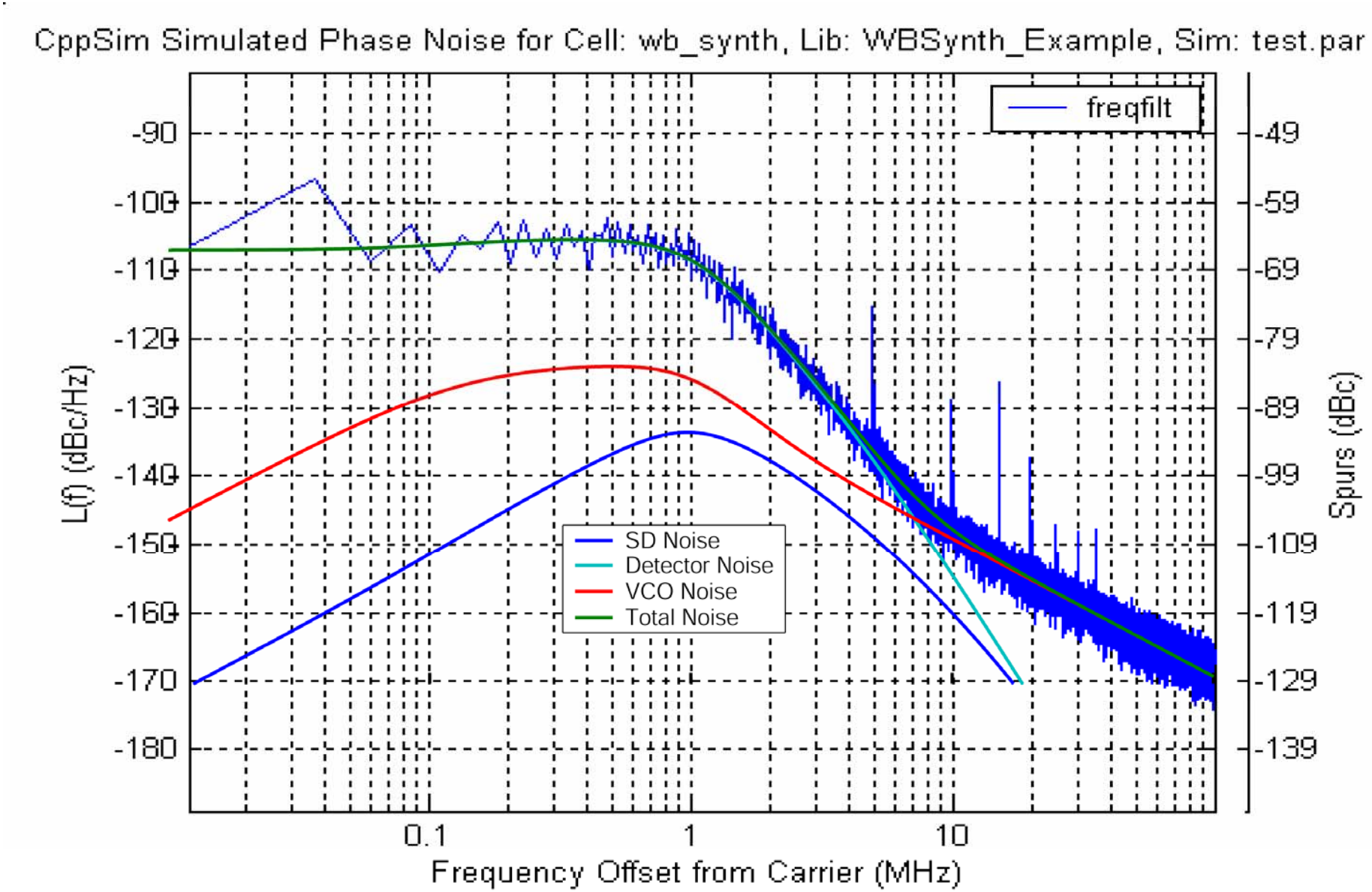
Calculate PLL Noise for 7-bit PFD/DAC Synthesizer





- **PFD/DAC**
  - **Adjust S-D Quant. Noise**
- **Delay mismatch**
  - **Adjust Detector noise**

Simulated PLL Phase Noise of 7-bit PFD/DAC



- **PLL Design Assistant accurately models simulated noise!**

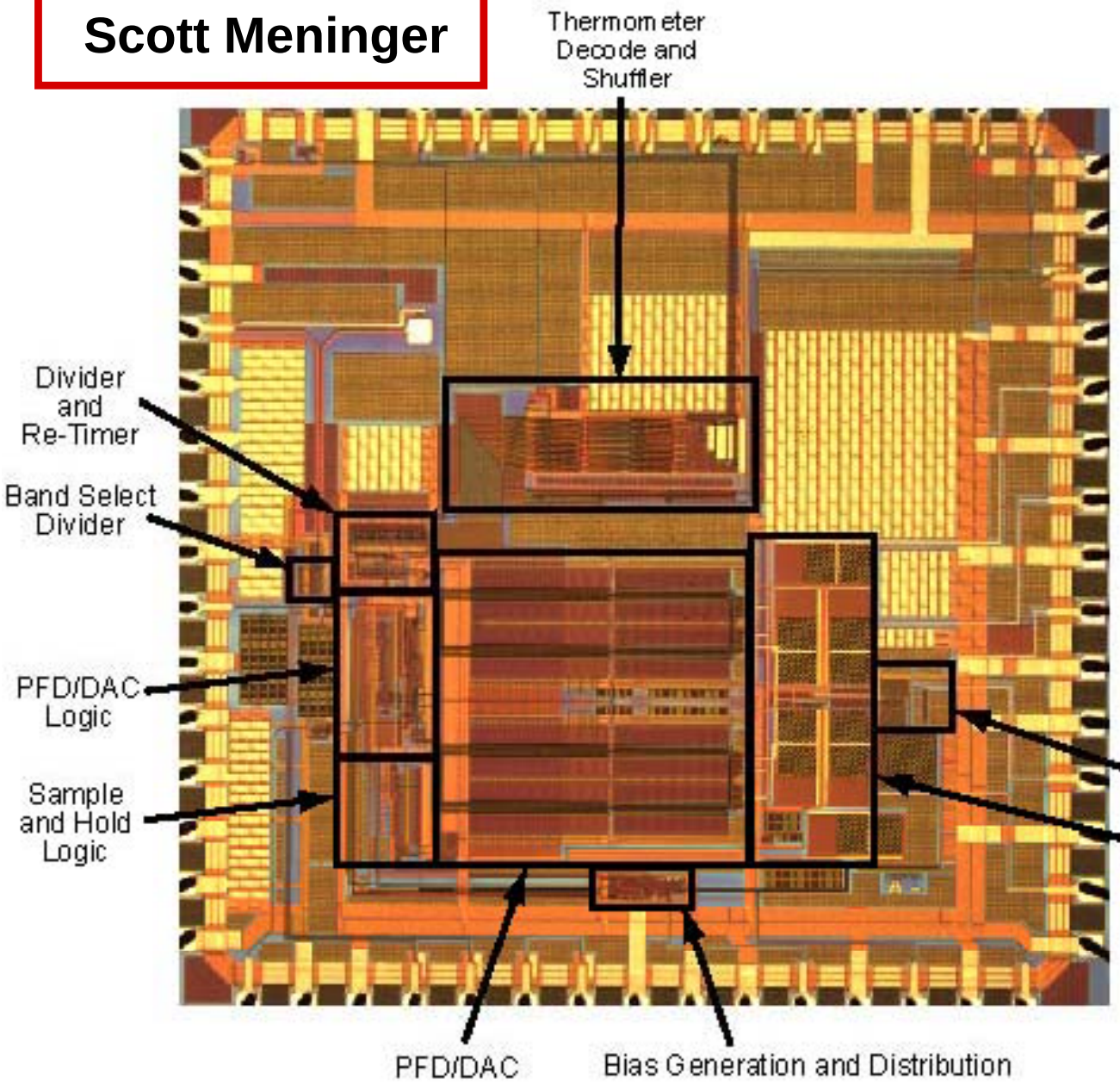
Summary of Design/Simulation Results

- **The PLL Design Assistant can be used to model the impact of**
  - **Intrinsic PLL noise sources**
  - **Quantization noise due to Σ–Δ dithering of divide value**
  - **Suppression of quantization noise by n-bit PFD/DAC**
  - **Impact of delay and current mismatch on PLL phase noise**
- **CppSim simulations confirm the accuracy of the above analysis**

How do PLL Design Assistant calculations compare to measured results?

A 1 MHz BW Fractional-N Frequency Synthesizer IC

Scott Meninger

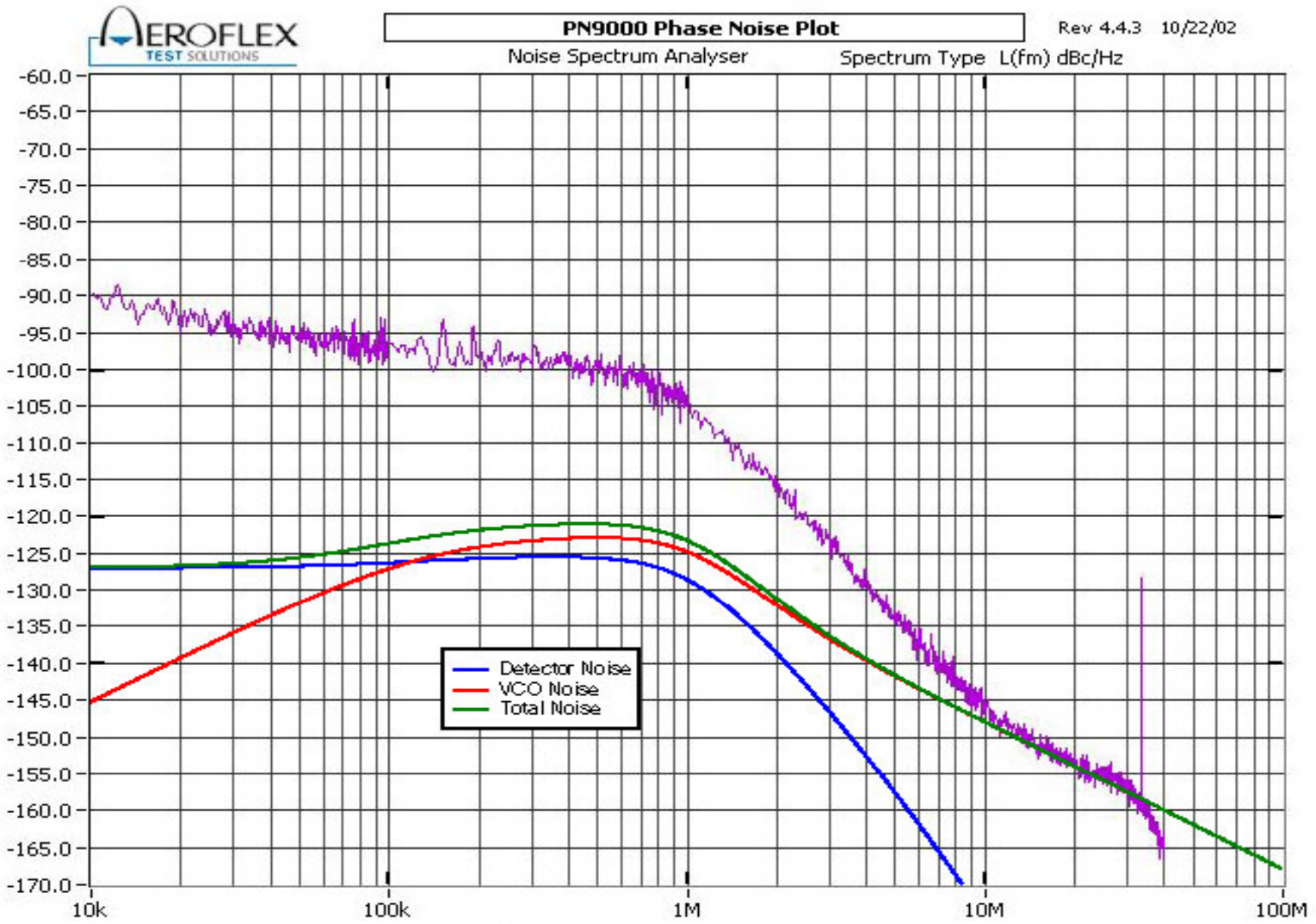


- **Implements proposed 7-bit PFD/DAC structure**
  - **0.18u CMOS**
  - **Circuit details to be published in the future**

Funded by MARCO C2S2

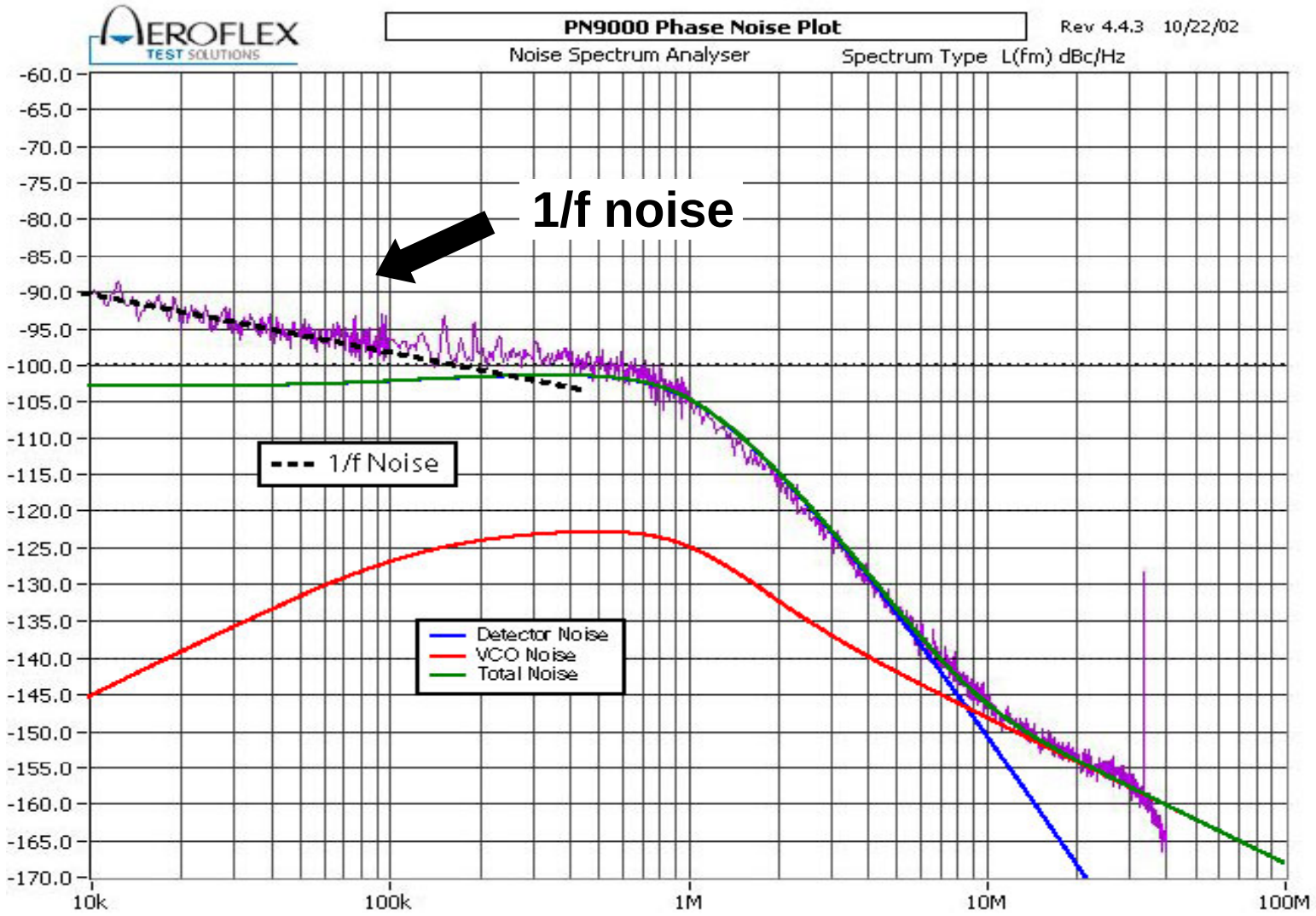
Fabricated by National Semiconductor

Measured Phase Noise of Intrinsic Noise Sources



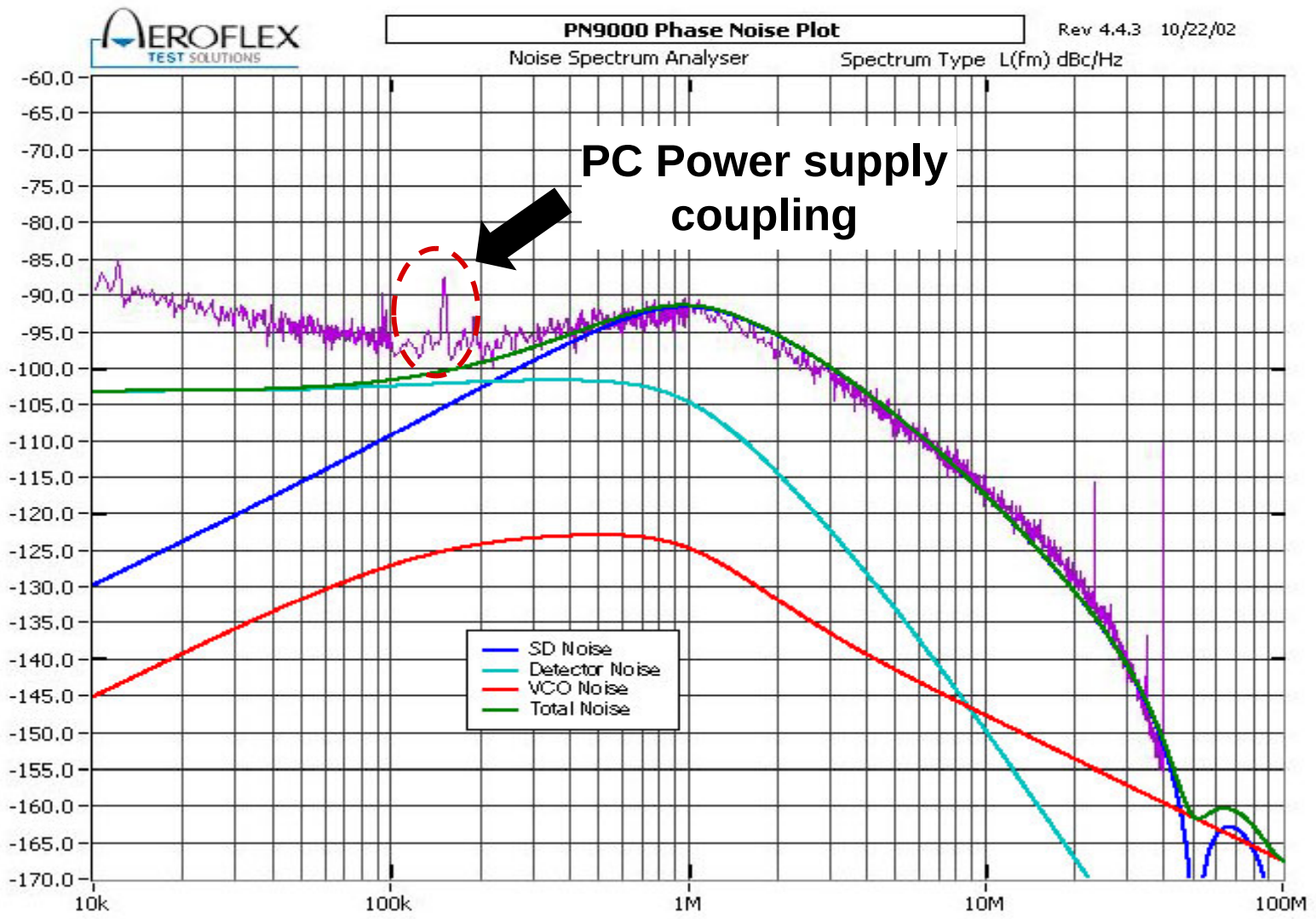
- IC configured as integer-N synth for above measurement
  - The detector noise calculation is way off!

Adjustment of Calculations to Fit Measured Result



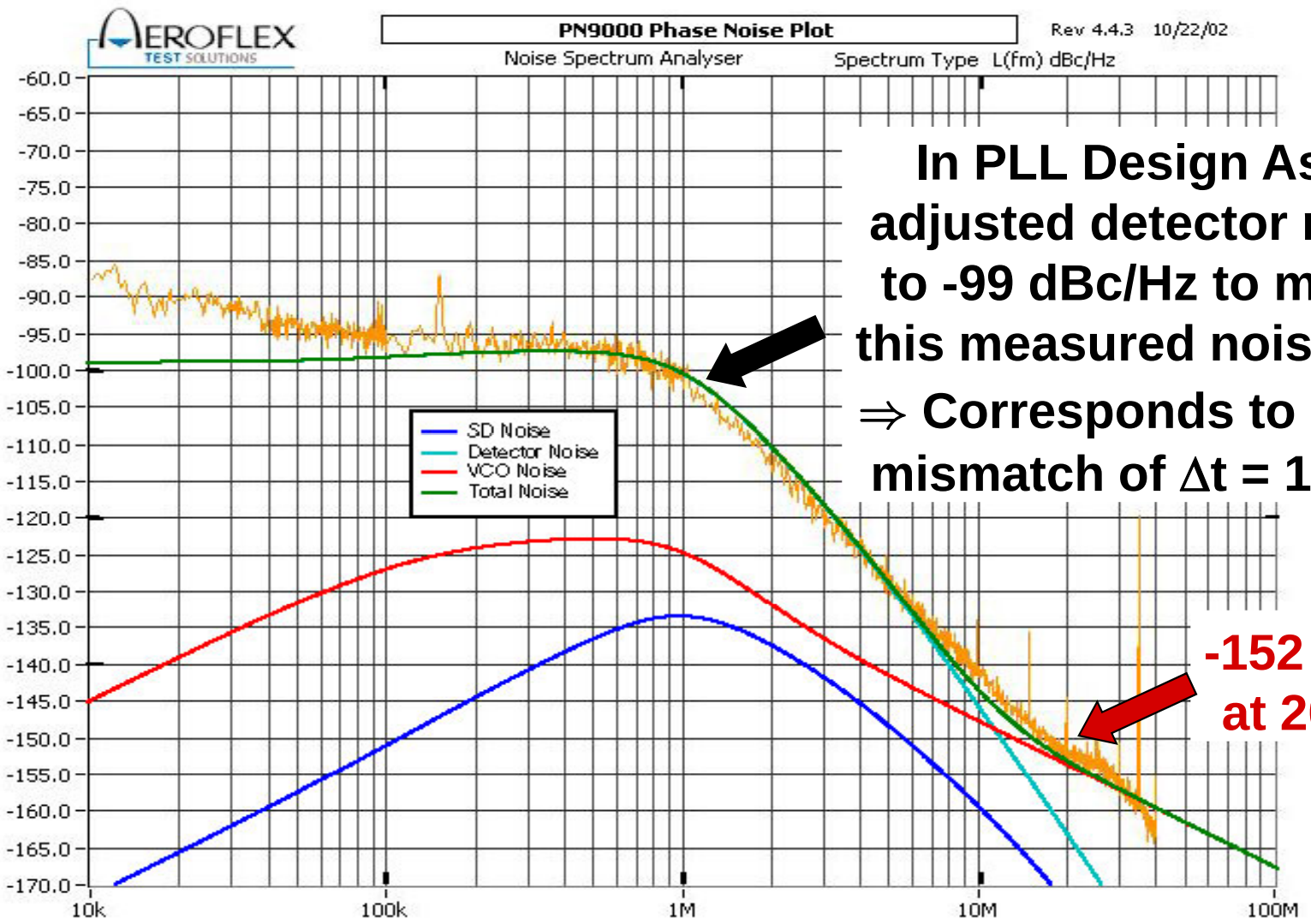
- Calculated noise above assumes:
  - Detector noise is -103 dBc/Hz at low freq

Measured Phase Noise of 2nd order Σ–Δ Synthesizer



- Calculations match measured results reasonably well
  - Note: Sampler is active for this measurement

Measured Phase Noise of 7-bit PFD/DAC Synthesizer



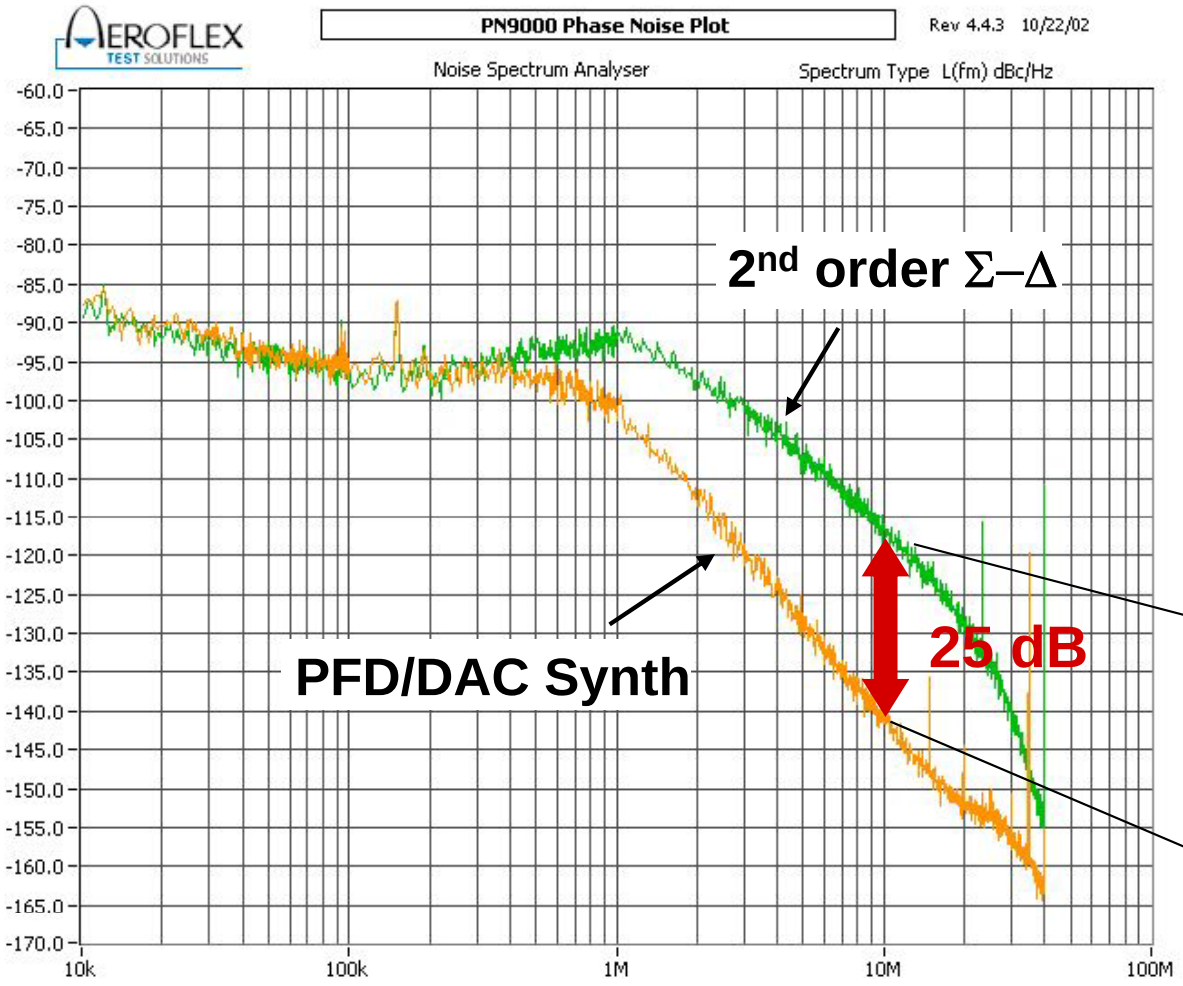
In PLL Design Asst: adjusted detector noise to -99 dBc/Hz to match this measured noise plot

⇒ Corresponds to delay mismatch of Δt = 12 ps

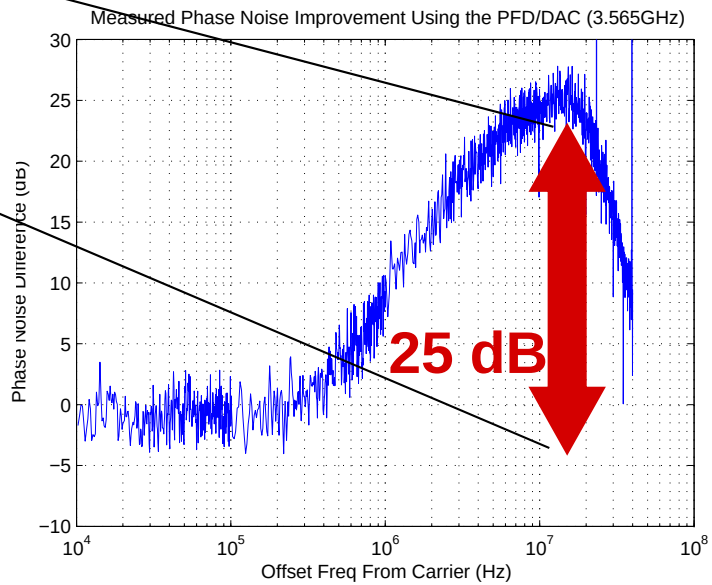
-152 dBc/Hz at 20 MHz!

- **Calculations match measured results reasonably well**

Comparison of Measured 2nd Order Σ–Δ to PFD/DAC

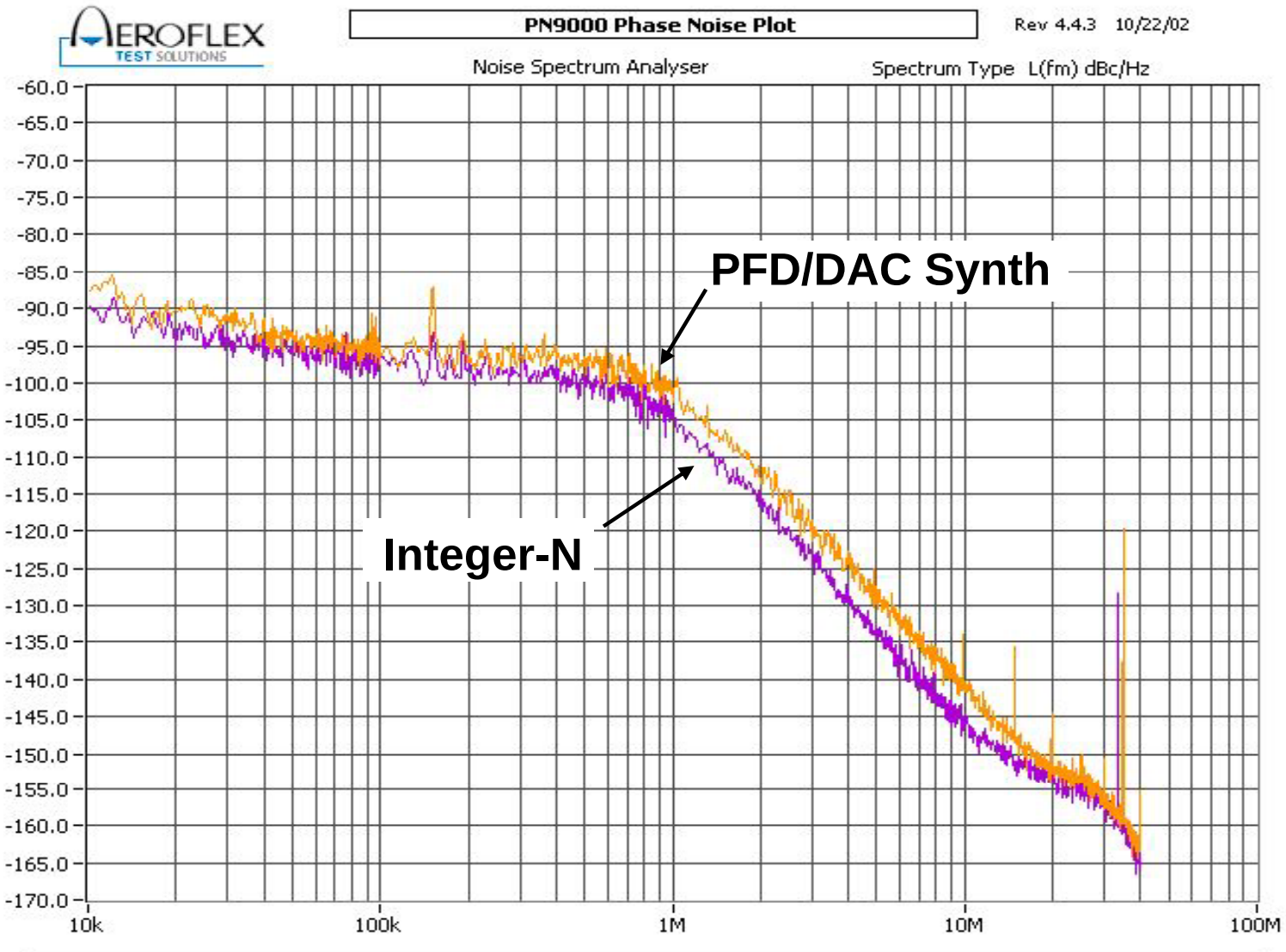


Suppression Plot



- **Measured 7-bit PFD/DAC quantization noise suppression is *better* than 25 dB!**

Comparison of Measured Integer-N to PFD/DAC

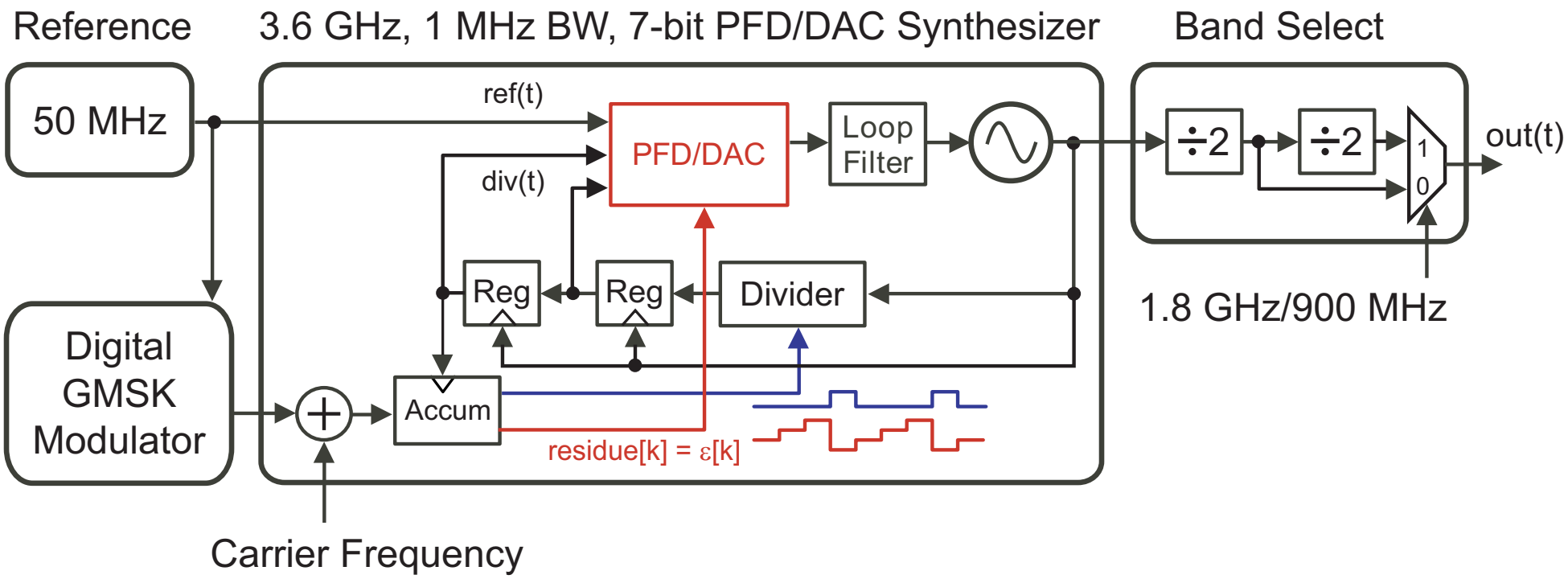


- 7-bit PFD/DAC approach is within 5 dB of intrinsic noise performance of PLL

Summary of Calculation/Measured Results Comparison

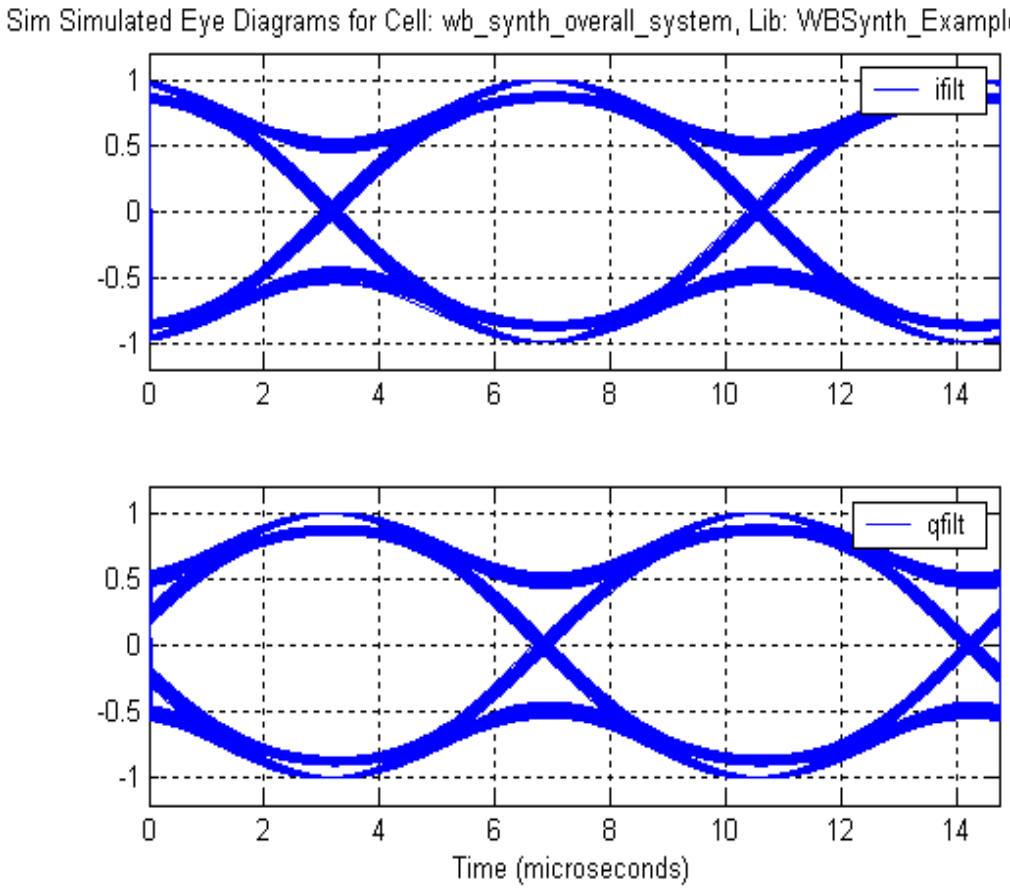
- **Comparison of PLL Design Assistant results to measured data allow back extraction of key parameters:**
  - **Intrinsic noise**
    - Detector and VCO noise
  - **PFD/DAC nonidealities**
    - Delay mismatch value
- **Future work: better low frequency noise accuracy**

A Highly Digital Implementation of a GMSK Transmitter

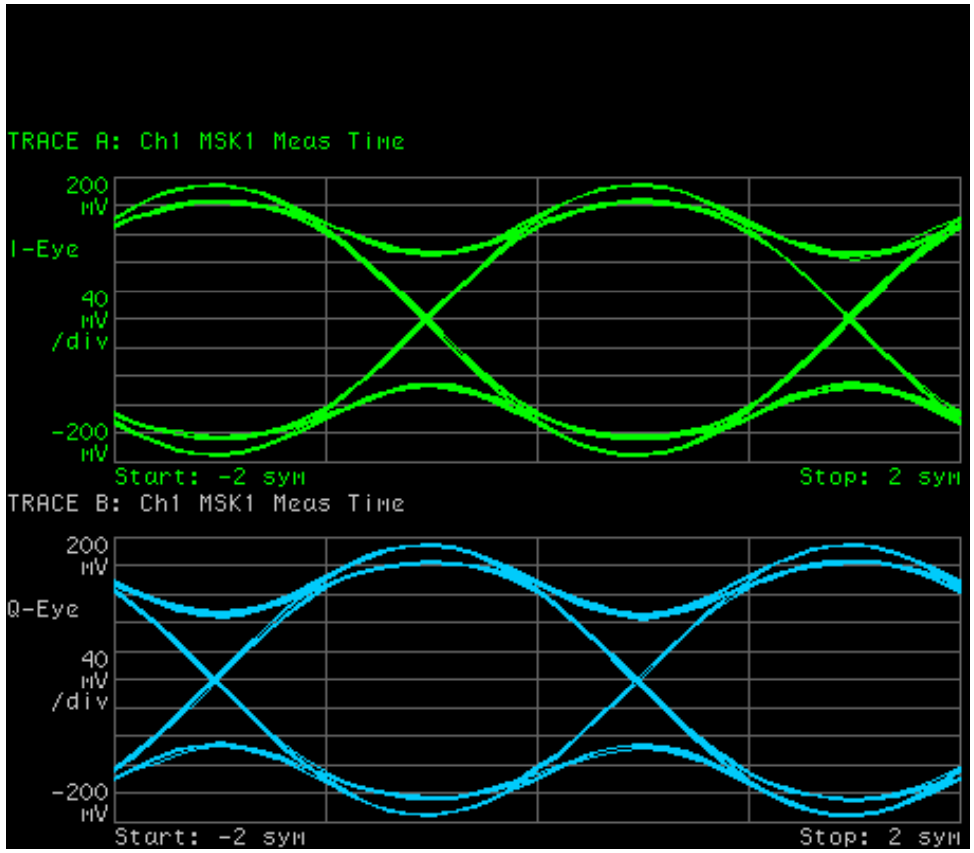


- **A fractional-N frequency synthesizer provides highly accurate phase/frequency modulation capability**
  - **Multiple carrier frequencies easily achieved with digital frequency division**
  - **N-bit PFD/DAC extends achievable data rate for a given noise performance (at higher frequency offsets)**

GMSK Eye Diagrams at 271 kbit/s (~900 MHz Carrier)



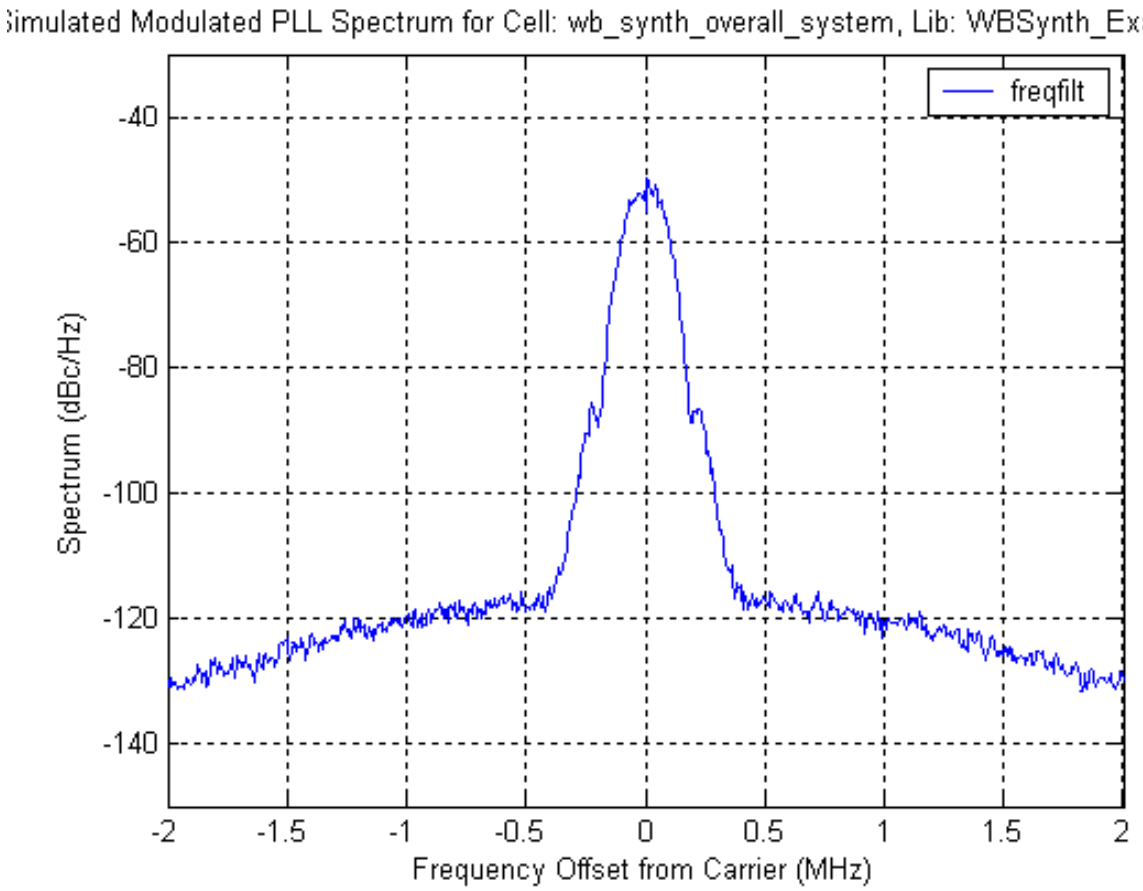
- **Measured**
  - **HP 89441 Vec. Analyzer**



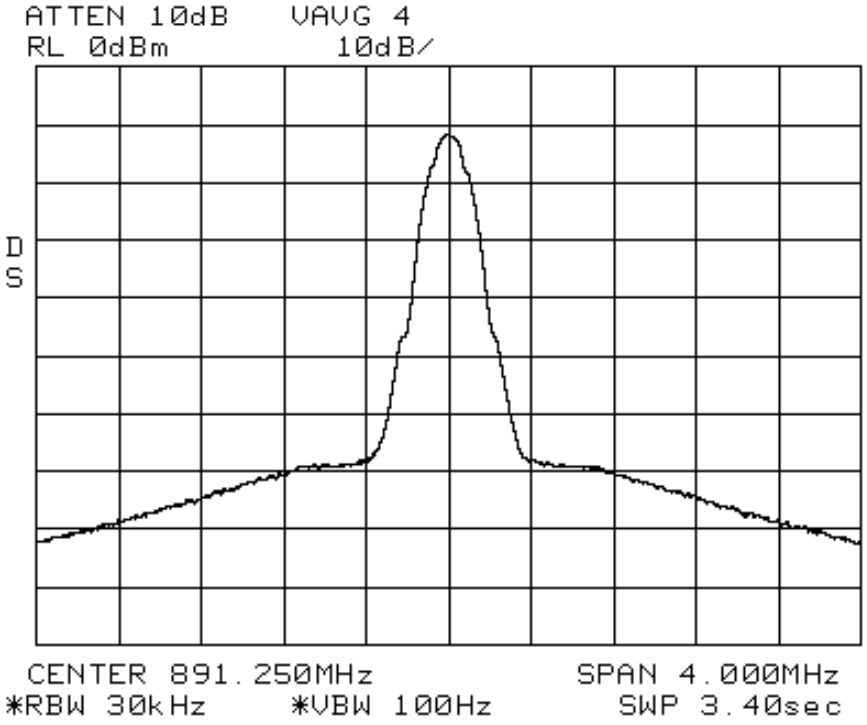
- **CppSim simulation**
  - **100e6 points: < 44 min**

Close agreement between simulated and measured results!

GMSK Spectra Plots at 271 kbit/s (~900 MHz Carrier)

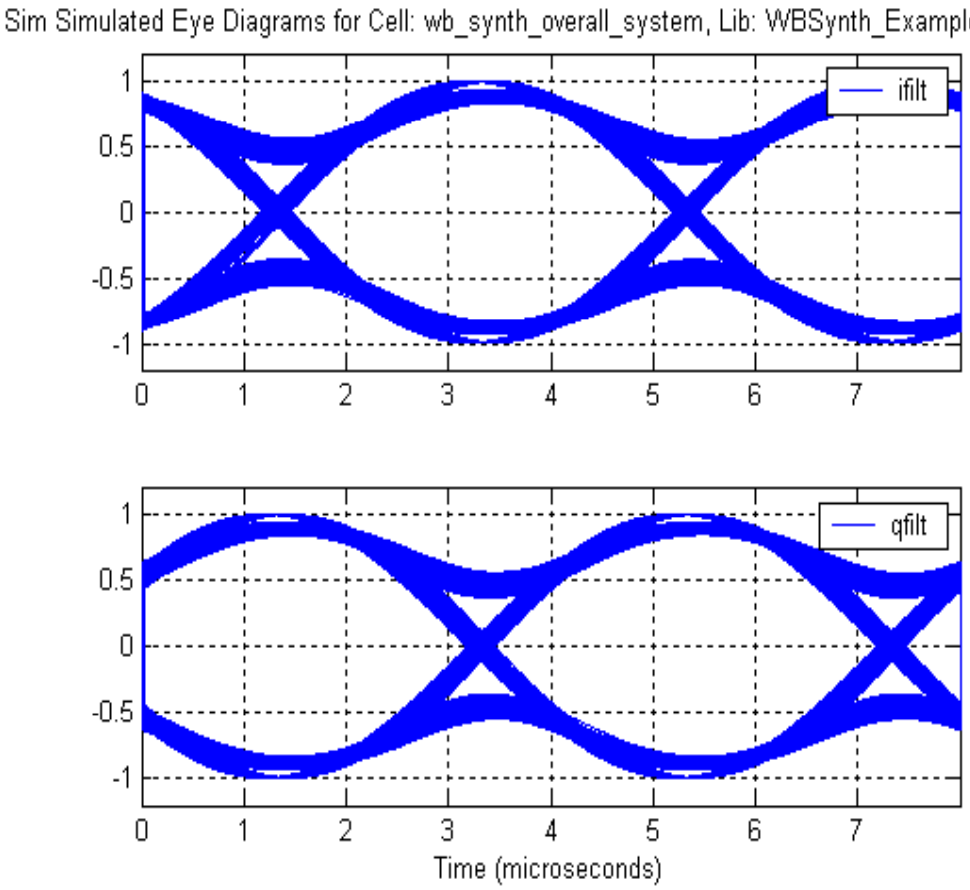


- **Measured**
  - **HP 8563E Spectrum Analyzer**

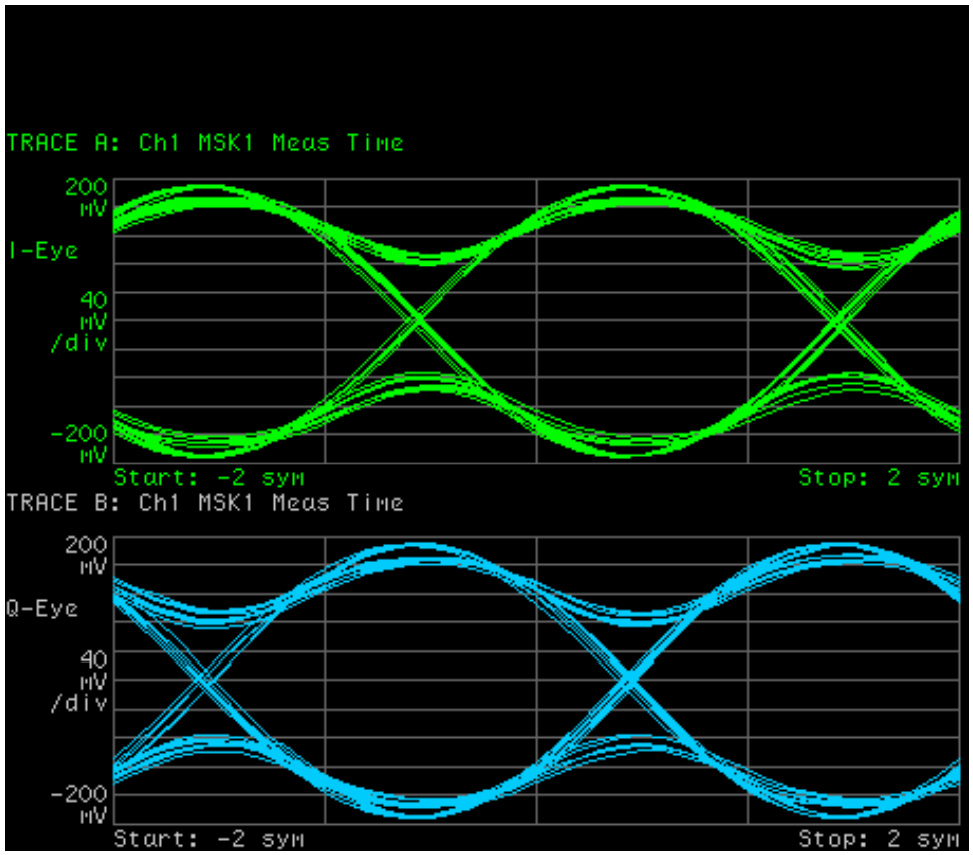


- **CppSim simulation**
  - **100e6 points: < 44 min**

GMSK Eye Diagrams at 500 kbit/s (~900 MHz Carrier)

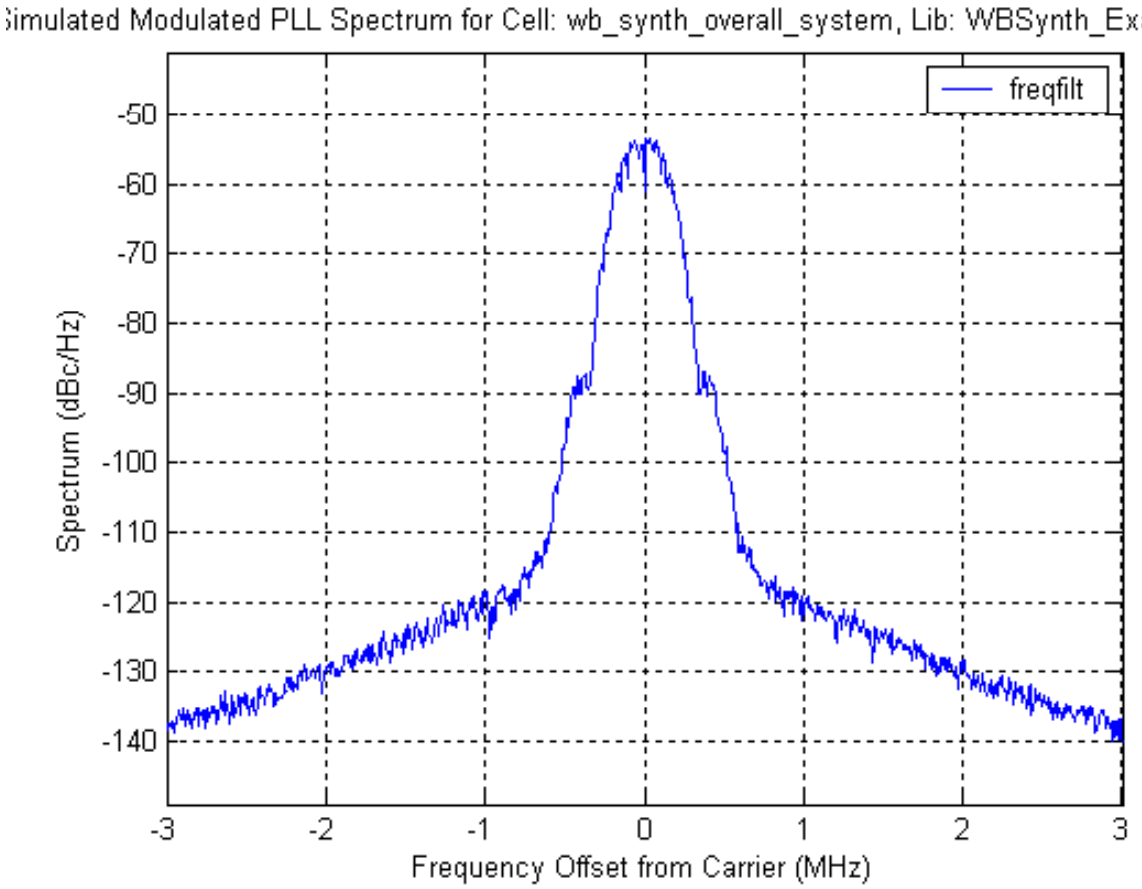


- **Measured**
  - **HP 89441 Vec. Analyzer**

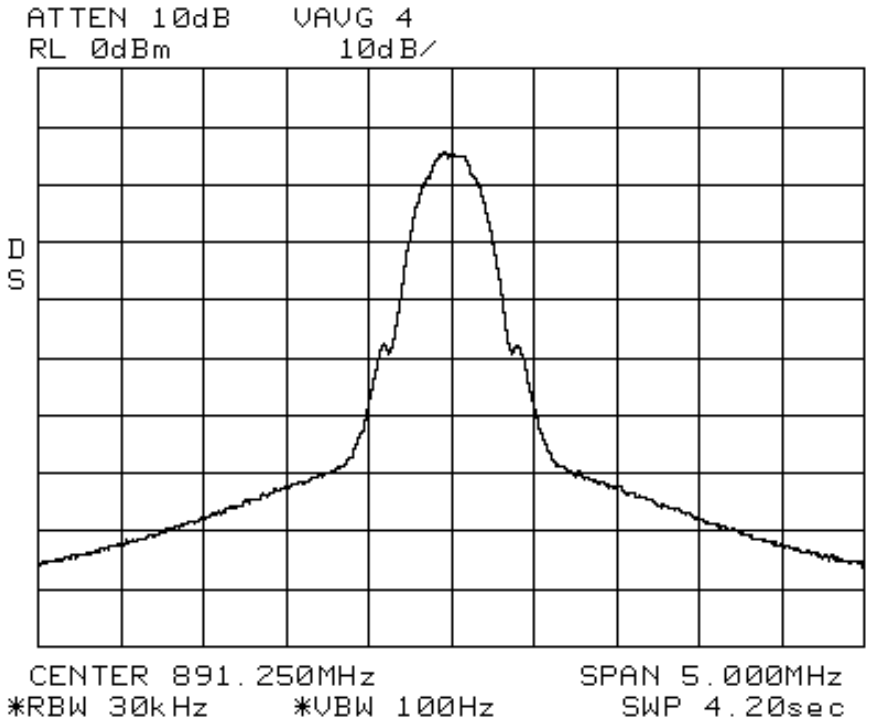


- **CppSim simulation**
  - **100e6 points: < 44 min**

GMSK Spectra Plots at 500 kbit/s (~900 MHz Carrier)

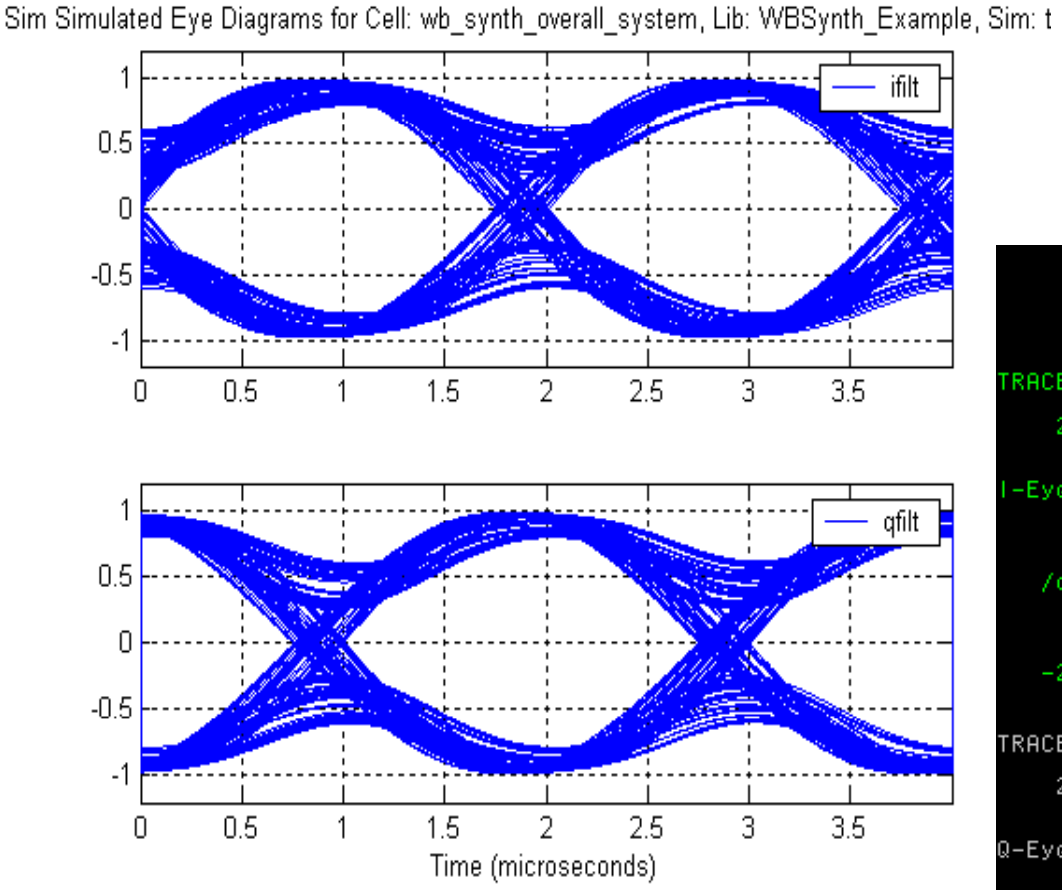


- **Measured**
  - **HP 8563E Spectrum Analyzer**

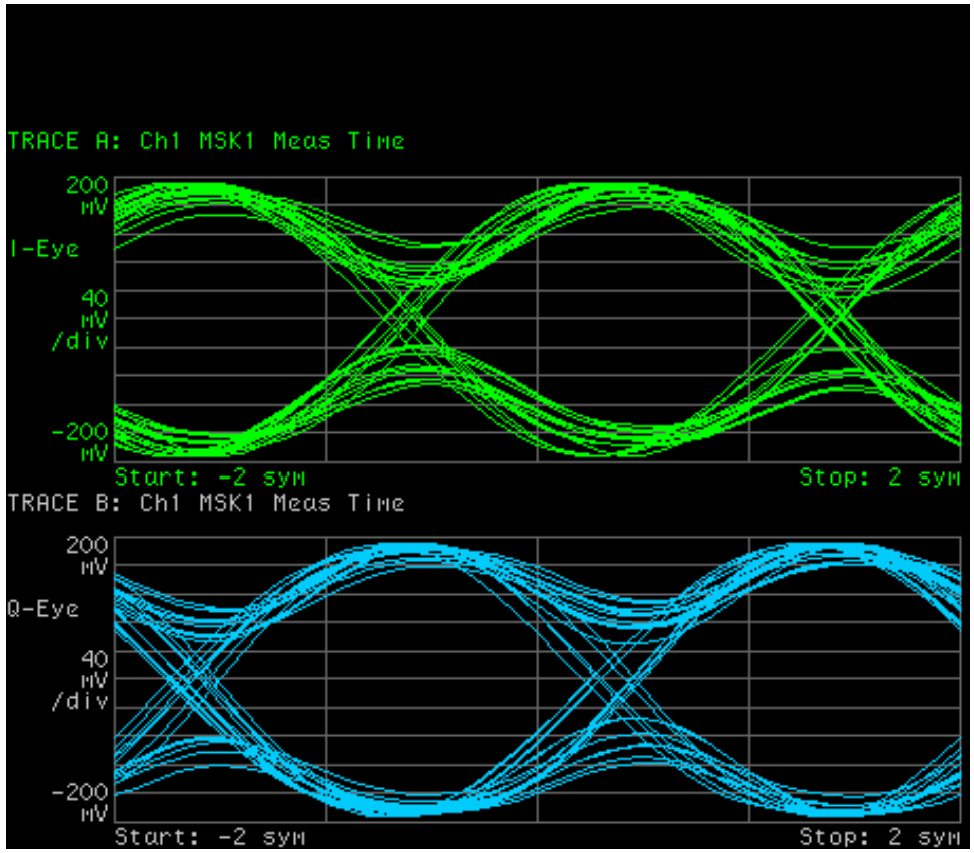


- **CppSim simulation**
  - **100e6 points: < 44 min**

GMSK Eye Diagrams at 1 Mbit/s (~900 MHz Carrier)

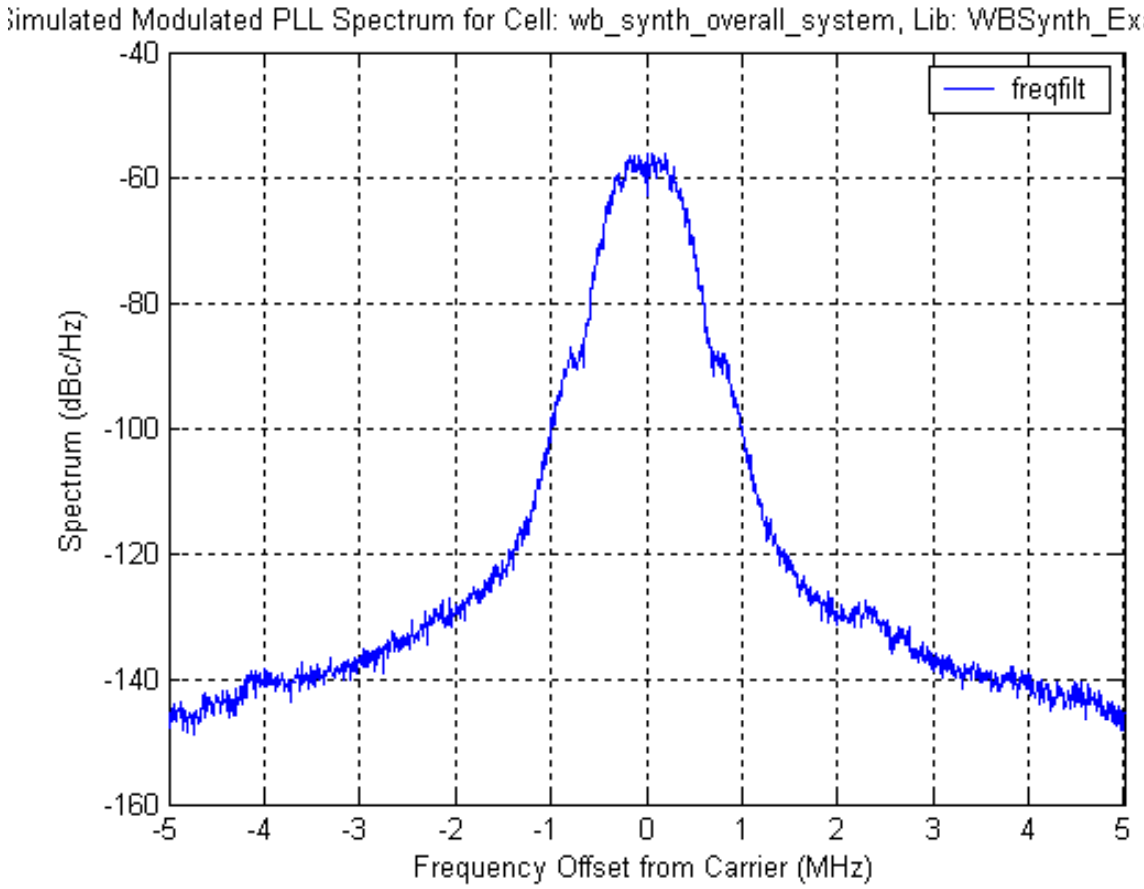


- **Measured**
  - **HP 89441 Vec. Analyzer**

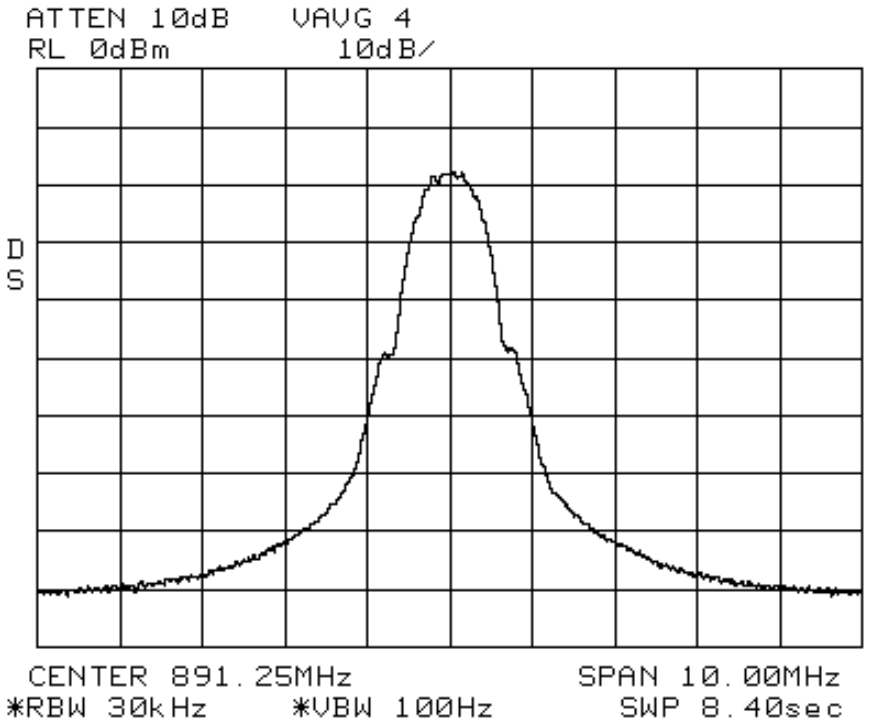


- **CppSim simulation**
  - **100e6 points: < 44 min**

GMSK Spectra Plots at 1 Mbit/s (~900 MHz Carrier)

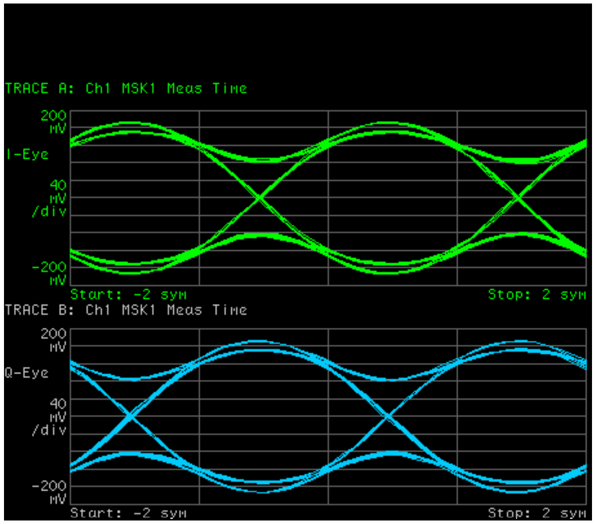


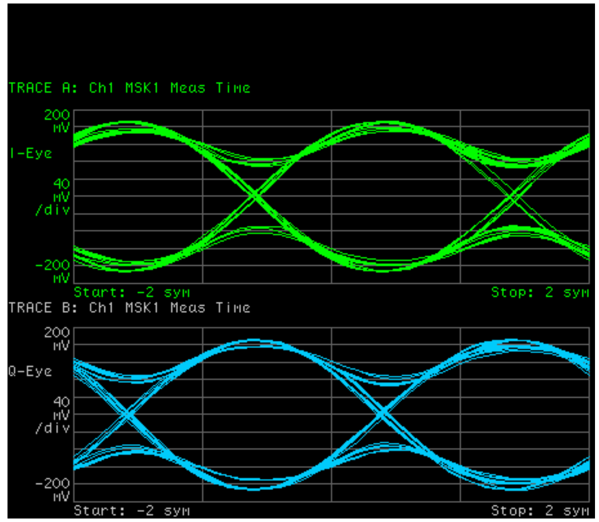
- **Measured**
  - **HP 8563E Spectrum Analyzer**

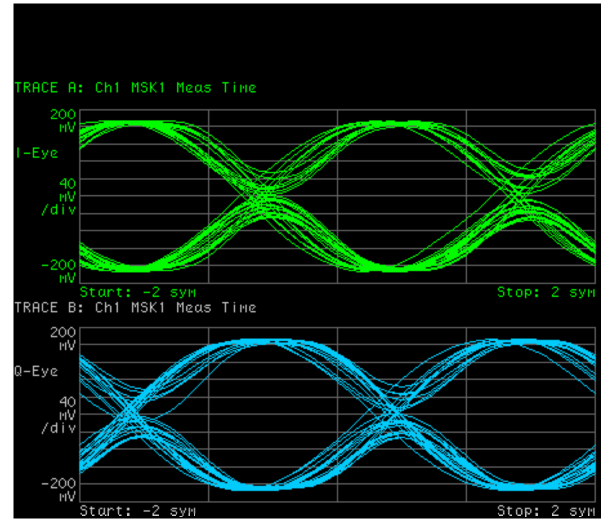


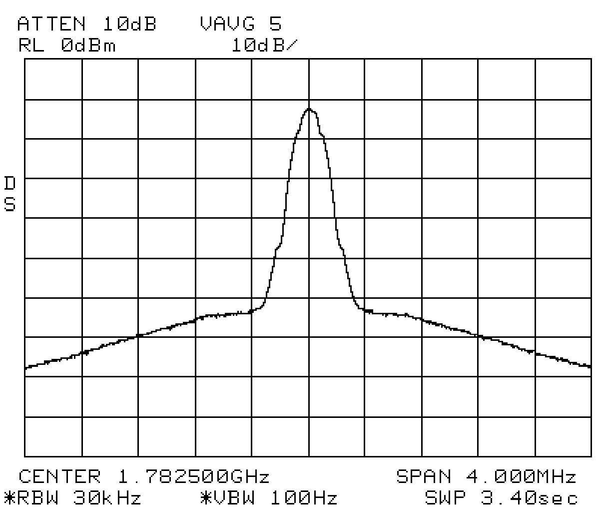
- **CppSim simulation**
  - **100e6 points: < 44 min**

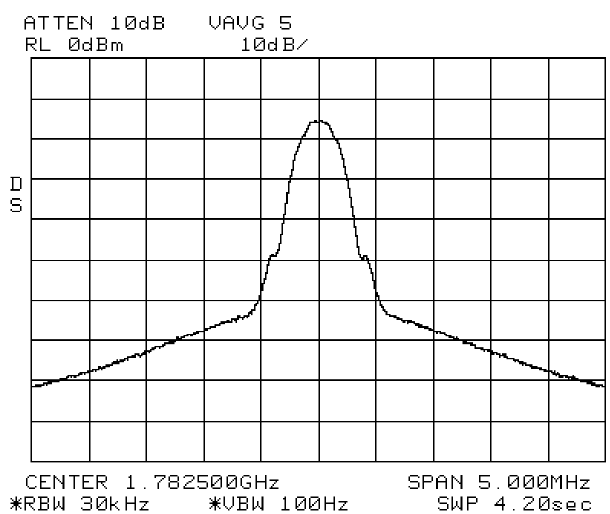
GMSK Measured Data at ~1.8 GHz Carrier Frequency

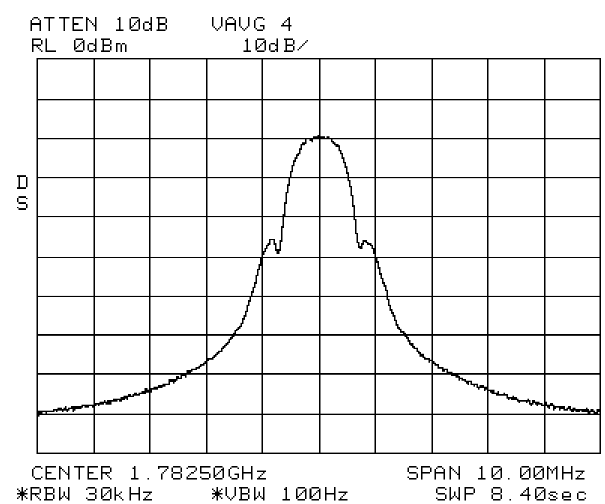












271 kbit/s | **500 kbit/s** | **1 Mbit/s**

Conclusions

- **Fractional-N frequency synthesizers are about to undergo dramatic improvement in achieving high PLL bandwidth with excellent noise performance**
  - **The PFD/DAC approach presented here is only one of many possibilities to achieve this goal**
- **Design and simulation methodologies are starting to emerge**
  - **Analytical modeling of noise can be quite accurate**
    - The PLL Design Assistant can be useful in this area
  - **Behavioral simulation can be used to verify analytical models**
    - CppSim offers a convenient and fast framework for this

Research into High Bandwidth PLL Architectures is at an Exciting Crossroads