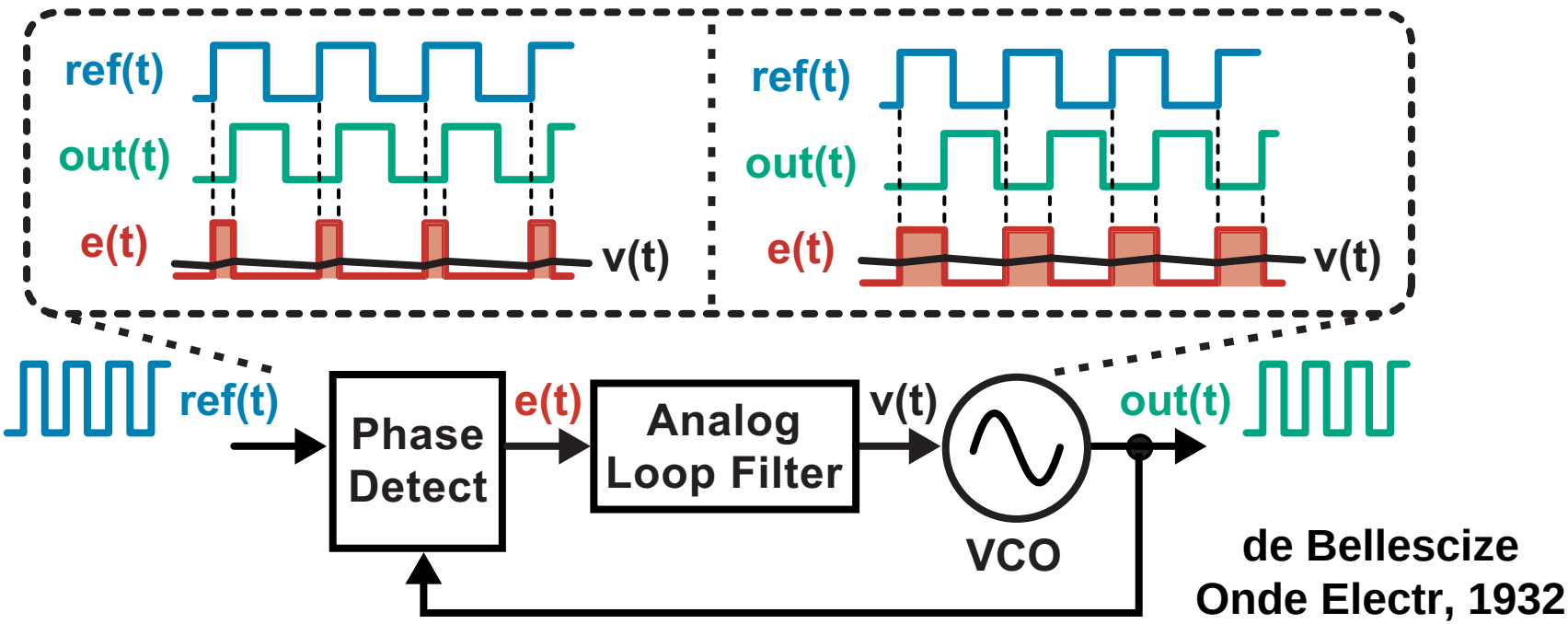
Short Course On Phase-Locked Loops
IEEE Circuit and System Society, San Diego, CA

Analog Frequency Synthesizers

Michael H. Perrott

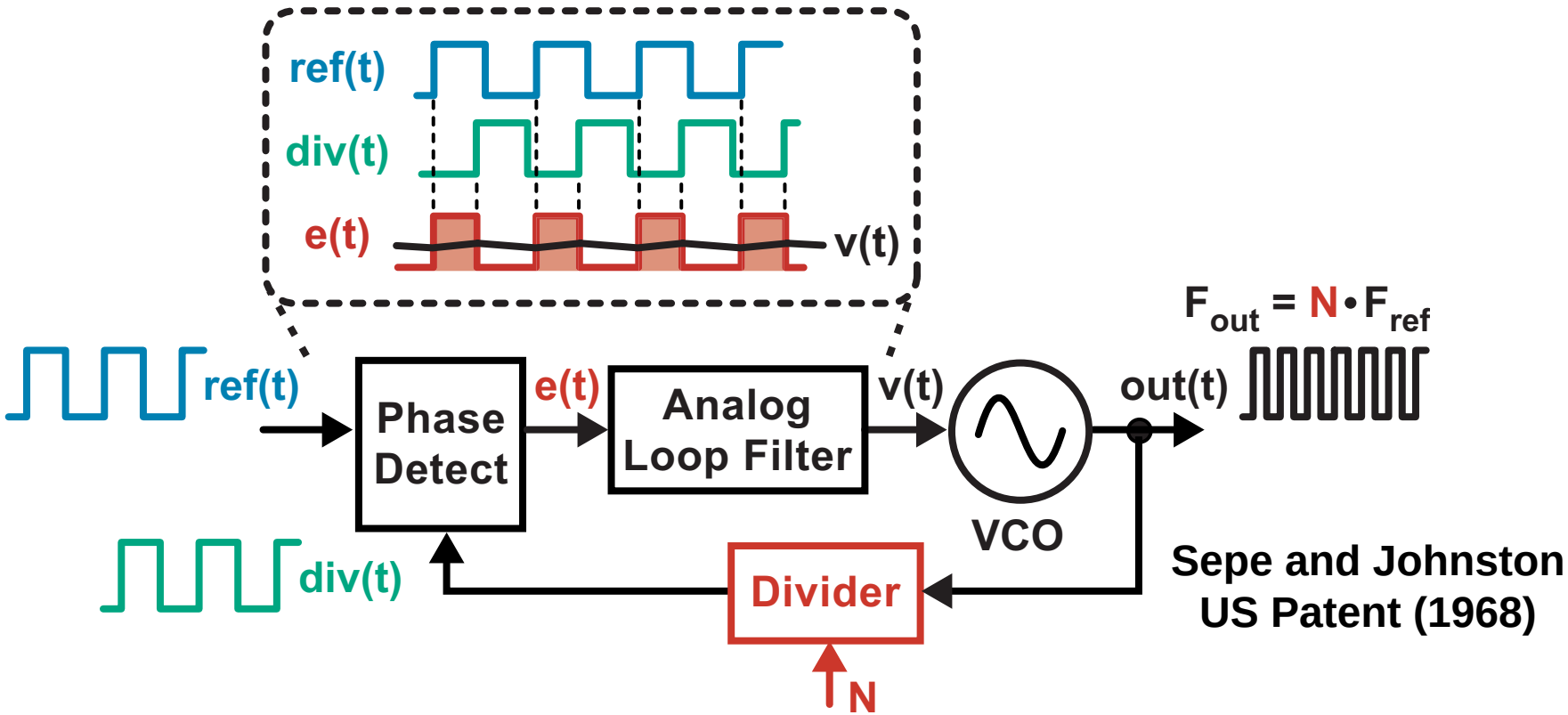
September 16, 2009

What is a Phase-Locked Loop (PLL)?



- VCO efficiently provides oscillating waveform with variable frequency
- PLL synchronizes VCO frequency to input reference frequency through feedback
  - Key block is phase detector
    - Realized as digital gates that create pulsed signals

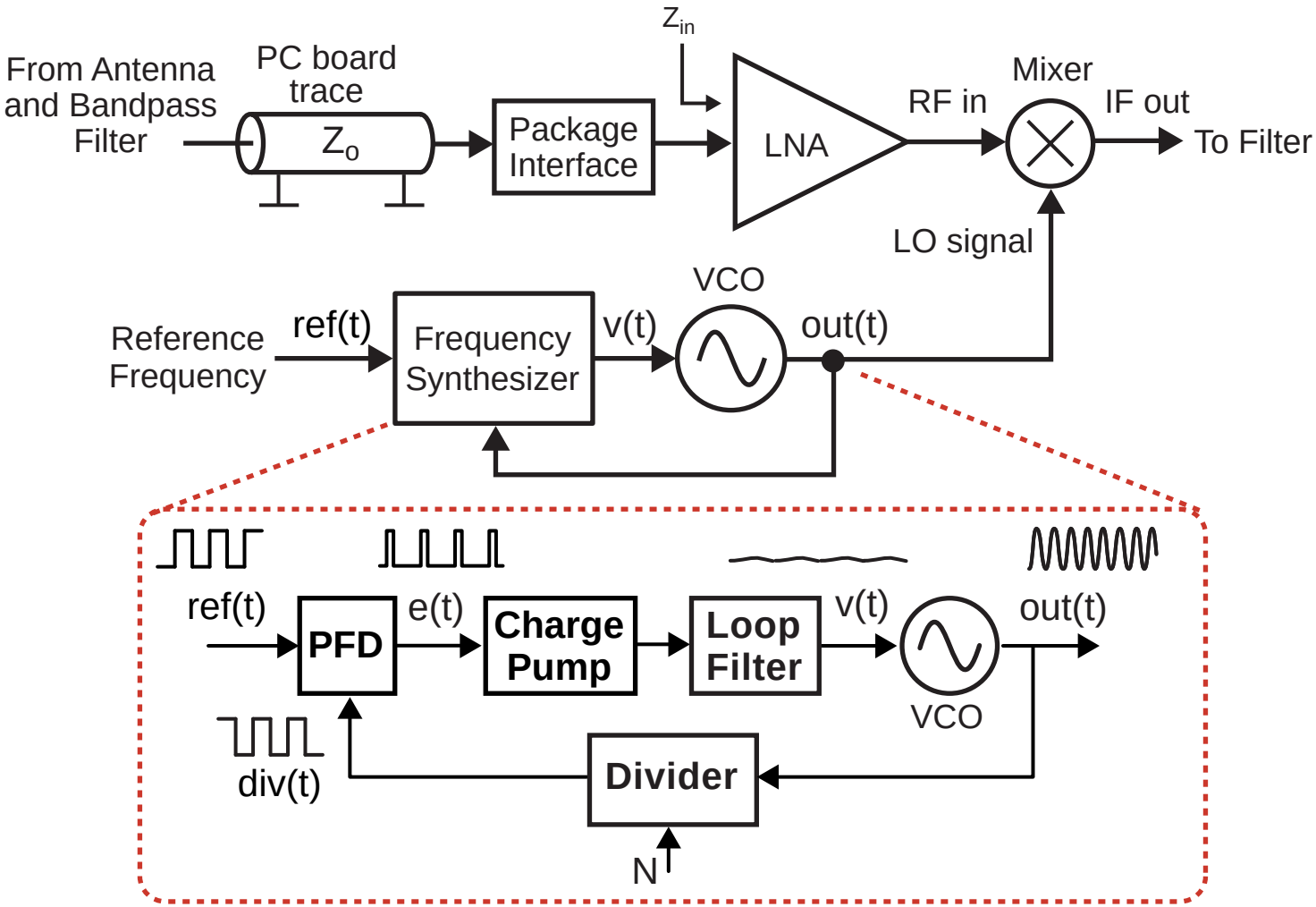
Integer-N Frequency Synthesizers



- Use digital counter structure to divide VCO frequency
  - Constraint: must divide by integer values
- Use PLL to synchronize reference and divider output

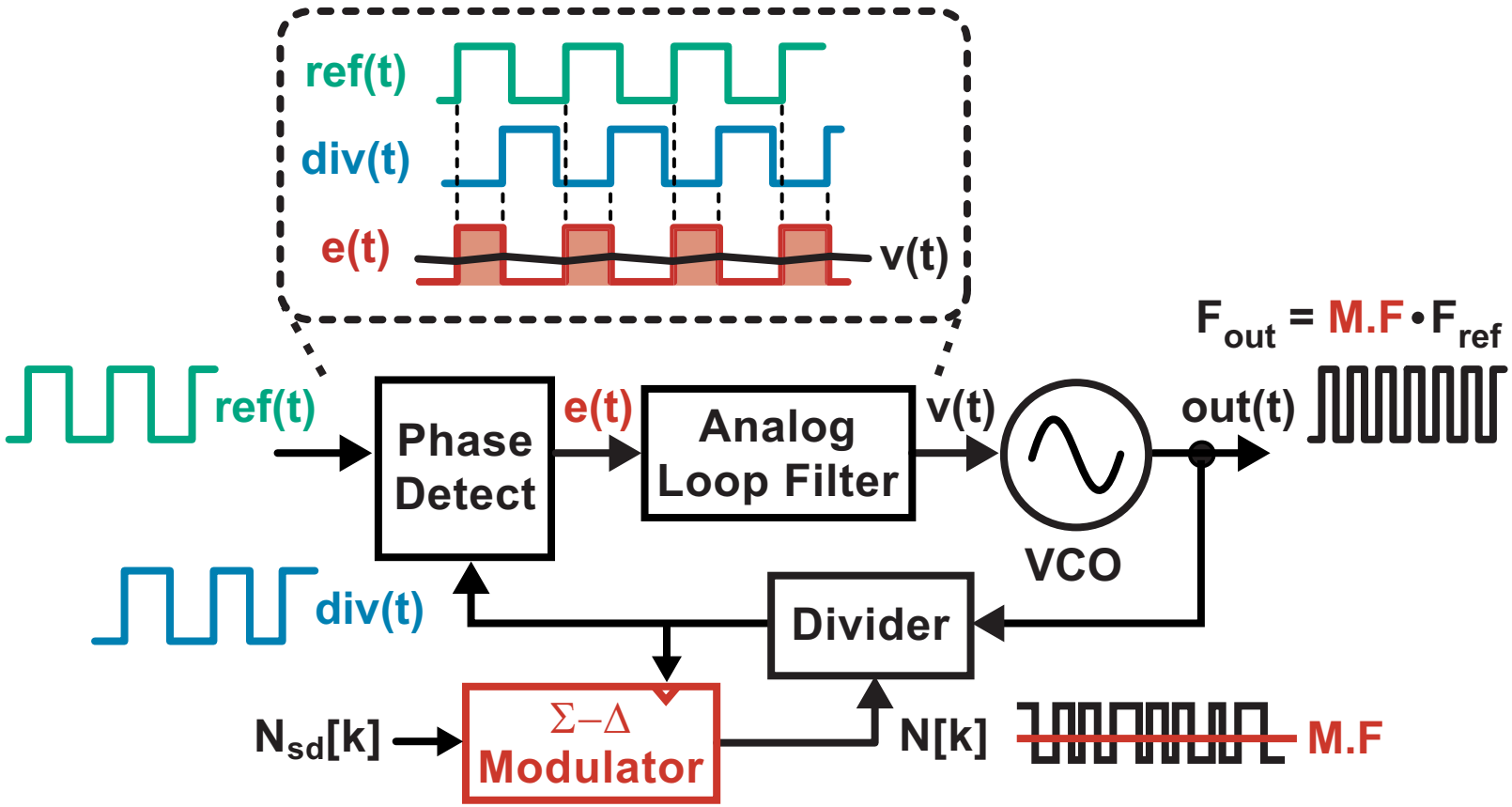
Output frequency is digitally controlled

Integer-N Frequency Synthesizers in Wireless Systems



- **Design Issues: low noise, fast settling time, low power**

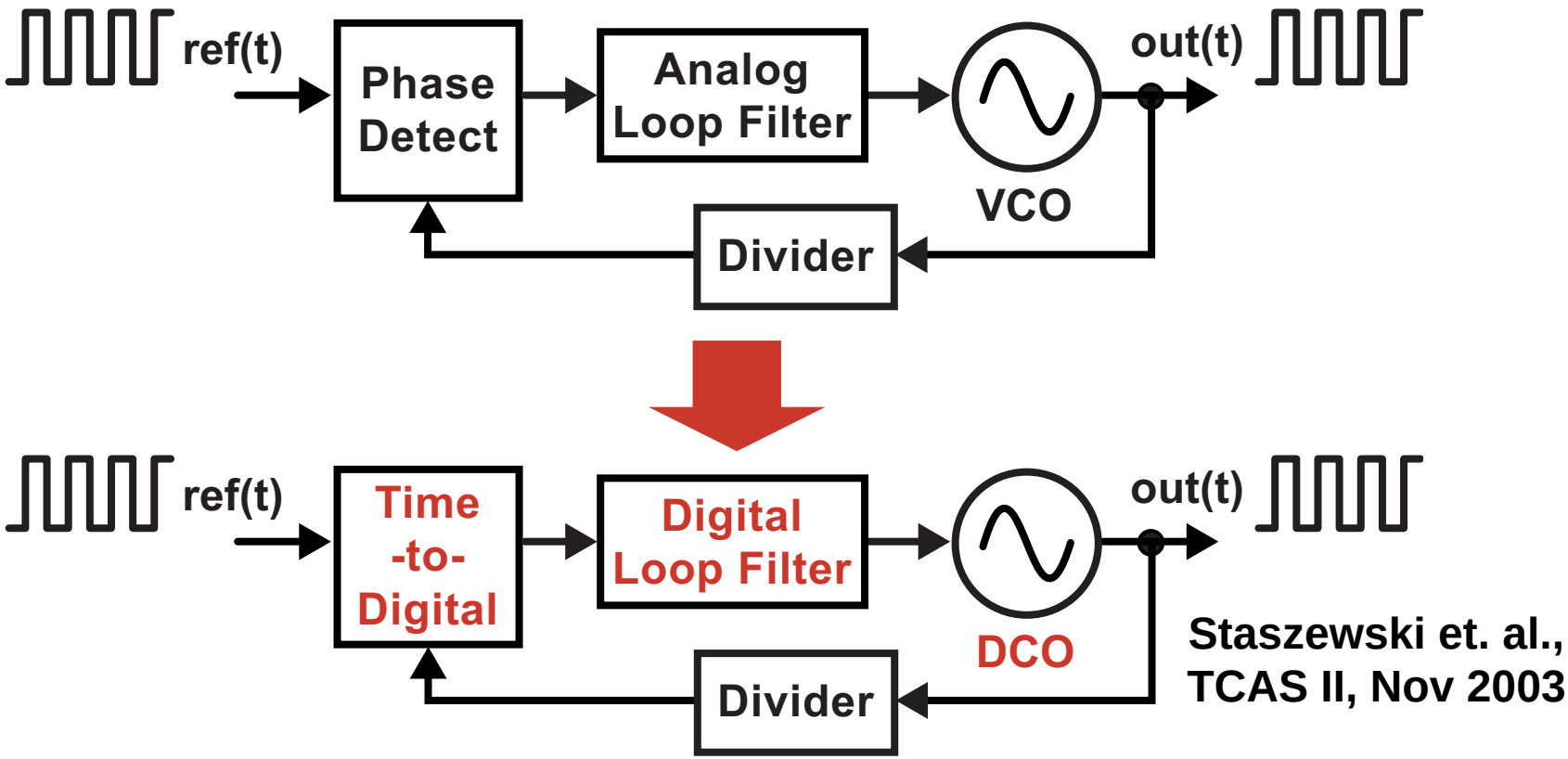
Fractional-N Frequency Synthesizers



- Dither divide value to achieve fractional divide values
  - PLL loop filter smooths the resulting variations

Very high frequency resolution is achieved

Going Digital …

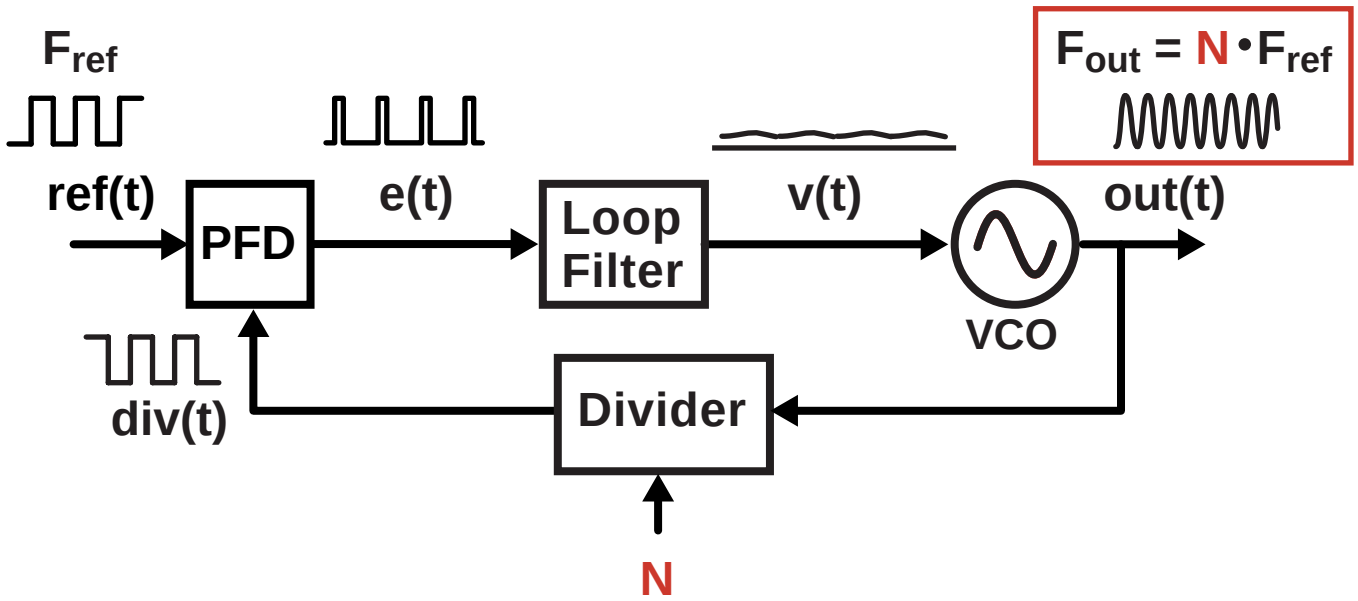


- Digital loop filter: compact area, insensitive to leakage
- Challenges:
  - Time-to-Digital Converter (TDC)
  - Digitally-Controlled Oscillator (DCO)

Outline of PLL Short Course

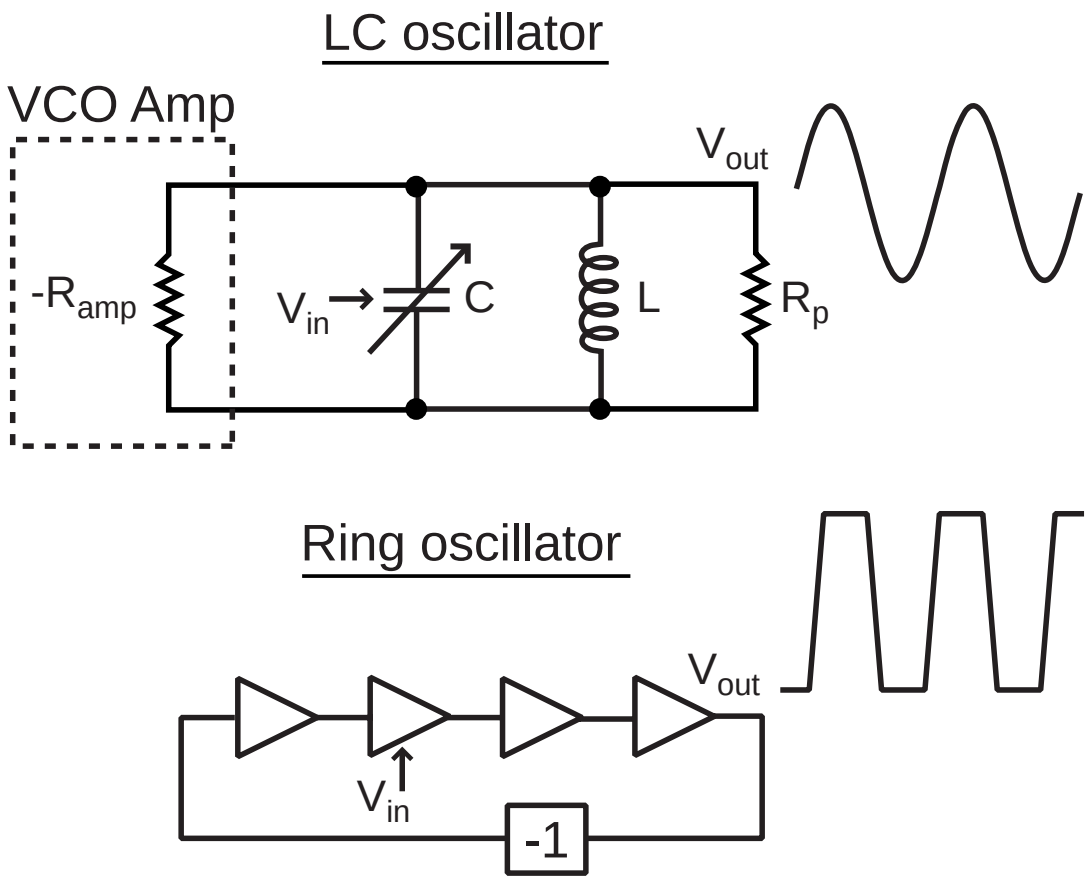
- **Analog frequency synthesizers**
  - **Integer-N synthesizers and PLL background**
  - **Fractional-N synthesizers**
- **Digital frequency synthesizers**
  - **Modeling and noise analysis**
  - **Time-to-digital conversion**

Outline of Integer-N Frequency Synthesizer Talk



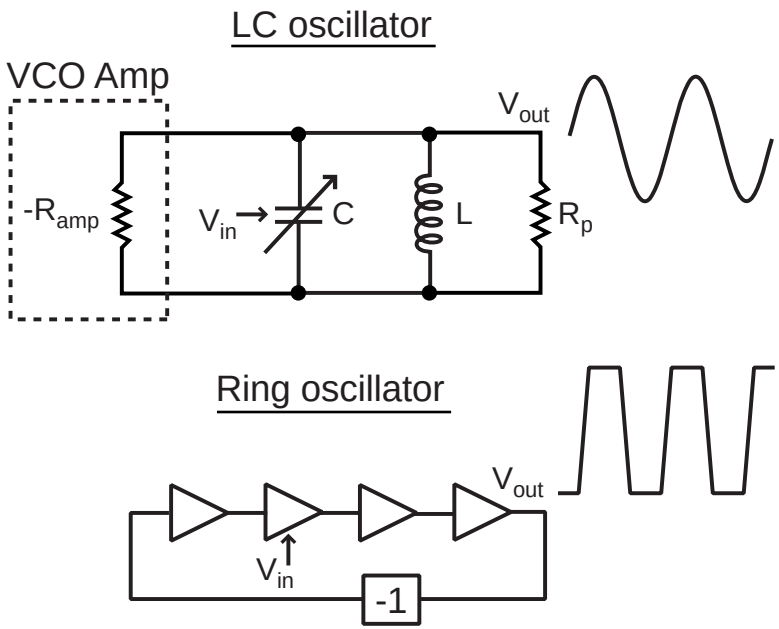
- Overview of PLL Blocks
- System Level Modeling
  - Transfer function analysis
  - Nonlinear behavior
  - Type I versus Type II systems
- Noise Analysis

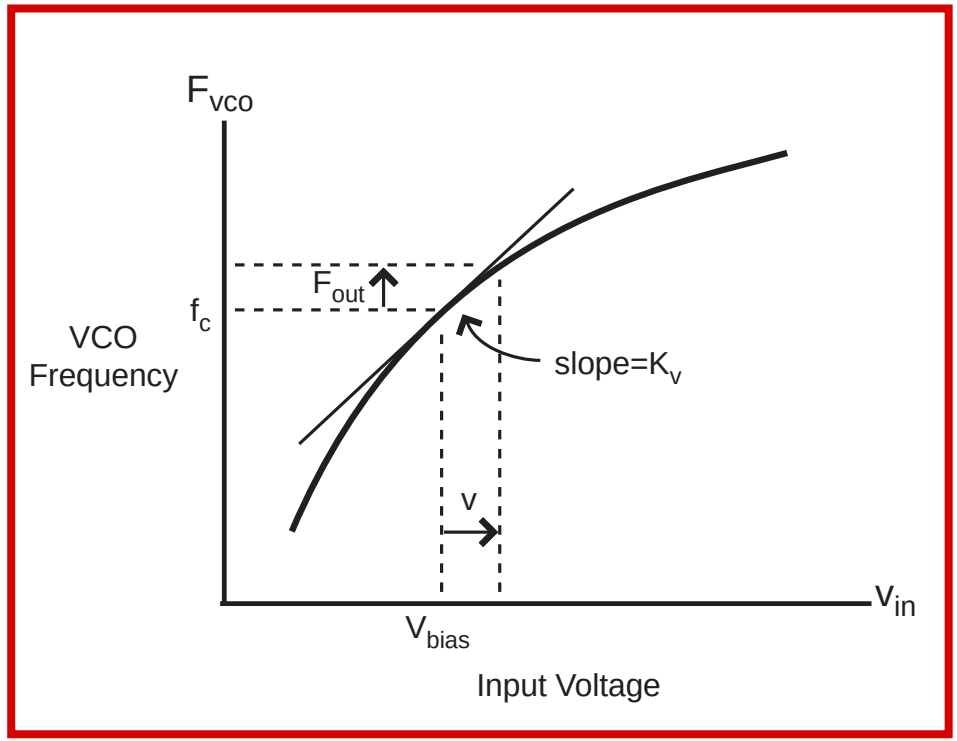
Popular VCO Structures



- **LC Oscillator: low phase noise, large area**
- **Ring Oscillator: easy to integrate, higher phase noise**

Model for Voltage to Frequency Mapping of VCO





$$F_{out}(t) = K_v v(t)$$

Model for Voltage to Phase Mapping of VCO

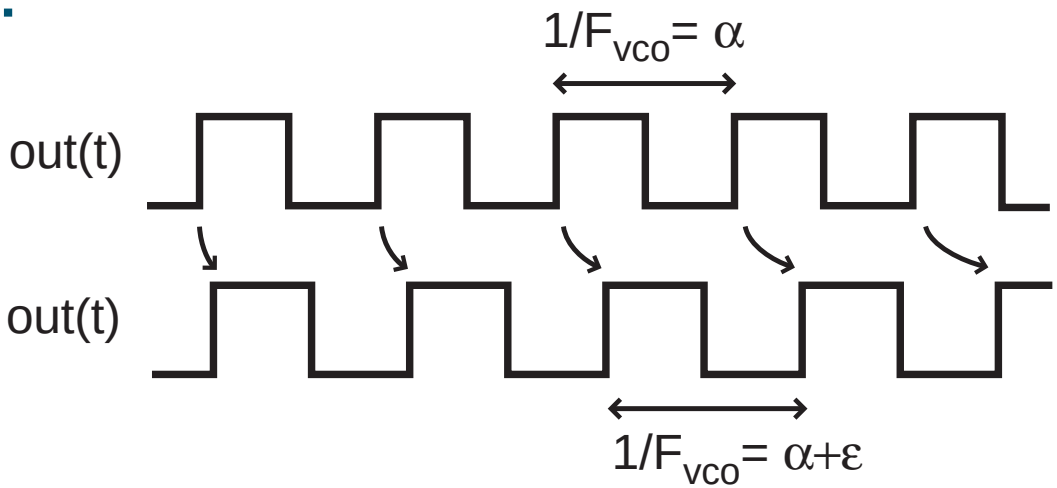
- **Time-domain frequency relationship (from previous slide)**

$$F_{out}(t) = K_v v(t)$$

- **Time-domain phase relationship**

$$\Phi_{out}(t) = \int_{-\infty}^{t} 2\pi F_{out}(\tau) d\tau = \int_{-\infty}^{t} 2\pi K_v v(\tau) d\tau$$

- **Intuition of integral relationship between frequency and phase:**

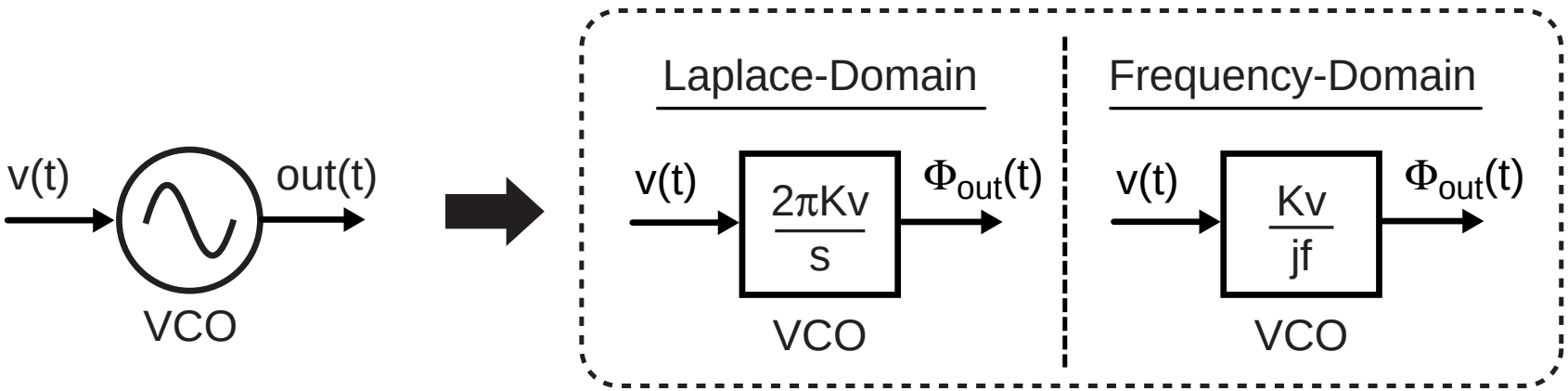


Frequency-Domain Model for VCO

- **Time-domain relationship (from previous slide)**

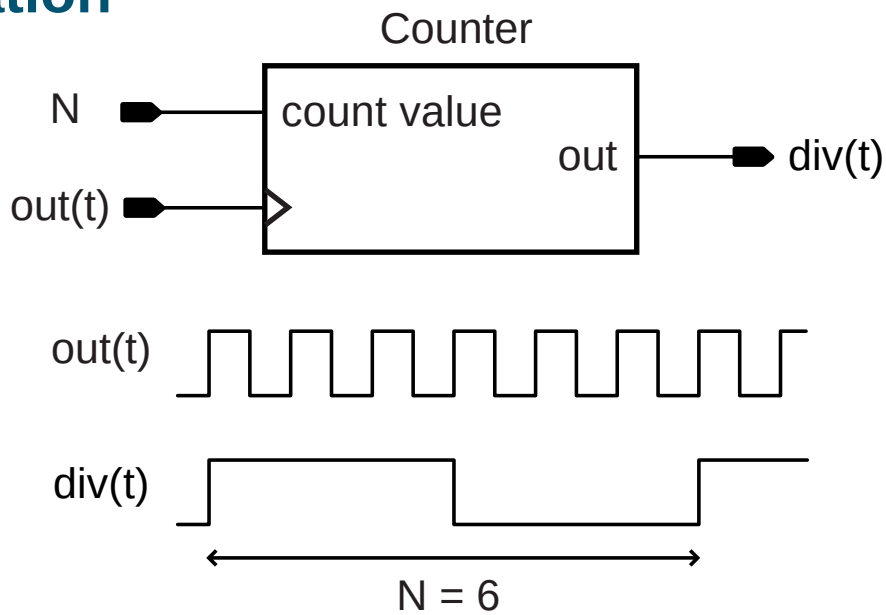
$$\Phi_{out}(t) = \int_{-\infty}^{t} 2\pi K_v v(\tau) d\tau$$

- **Corresponding frequency-domain model**



Divider

- **Implementation**



- **Time-domain model**
  - **Frequency:**

$$F_{div}(t) = \frac{1}{N} F_{out}(t).$$

  - **Phase:**

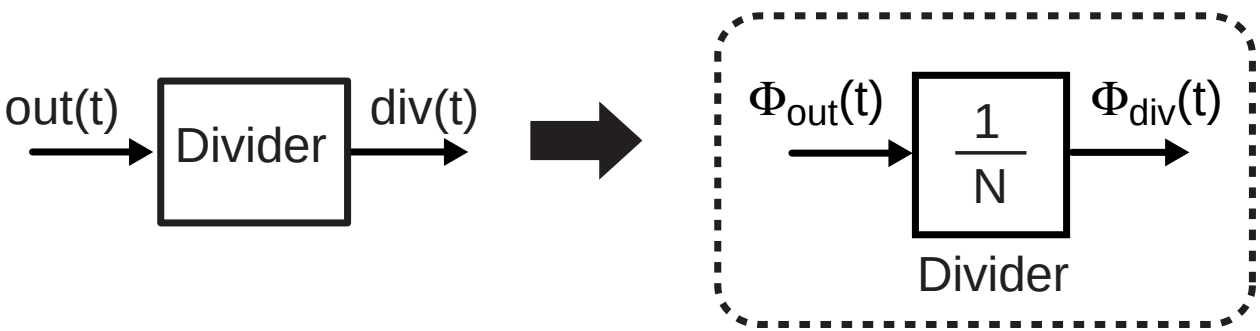
$$\Phi_{div}(t) = \int_{-\infty}^{t} 2\pi \frac{1}{N} F_{out}(\tau) d\tau = \frac{1}{N} \Phi_{out}(t)$$

Frequency-Domain Model of Divider

- **Time-domain relationship between VCO phase and divider output phase (from previous slide)**

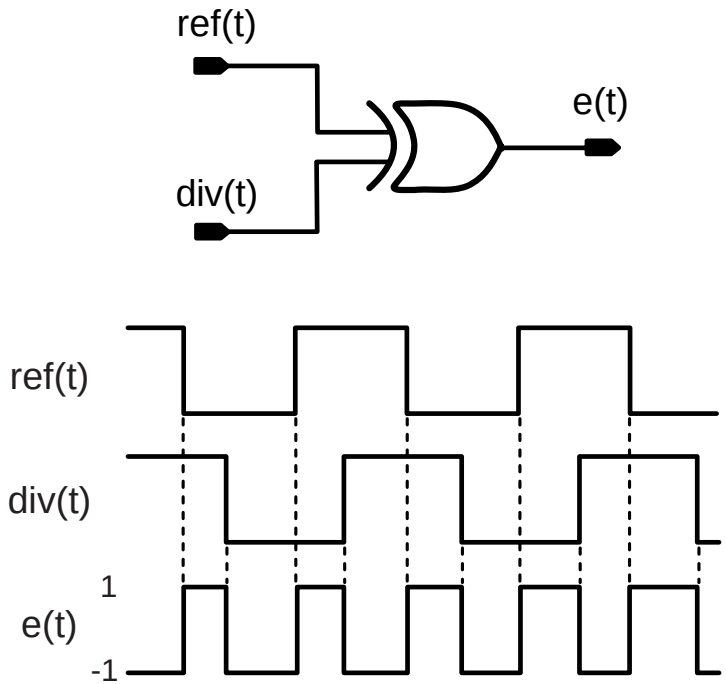
$$\Phi_{div}(t) = \frac{1}{N}\Phi_{out}(t)$$

- **Corresponding frequency-domain model (same as Laplace-domain)**

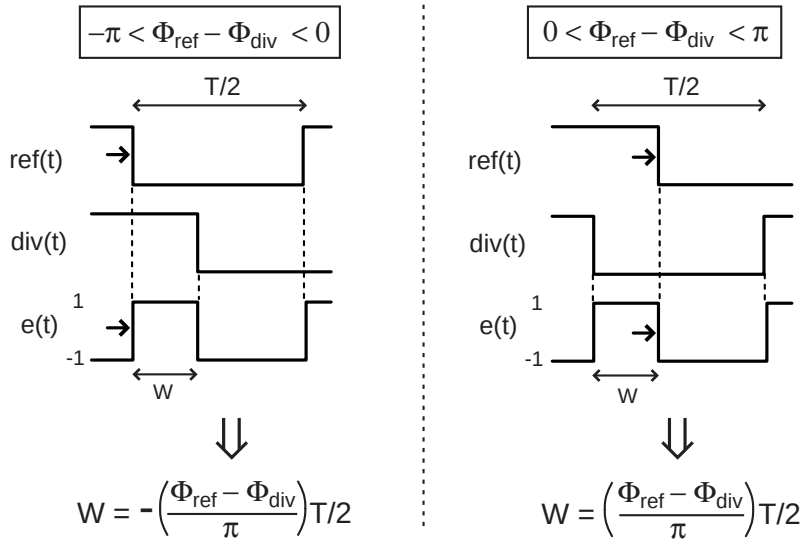


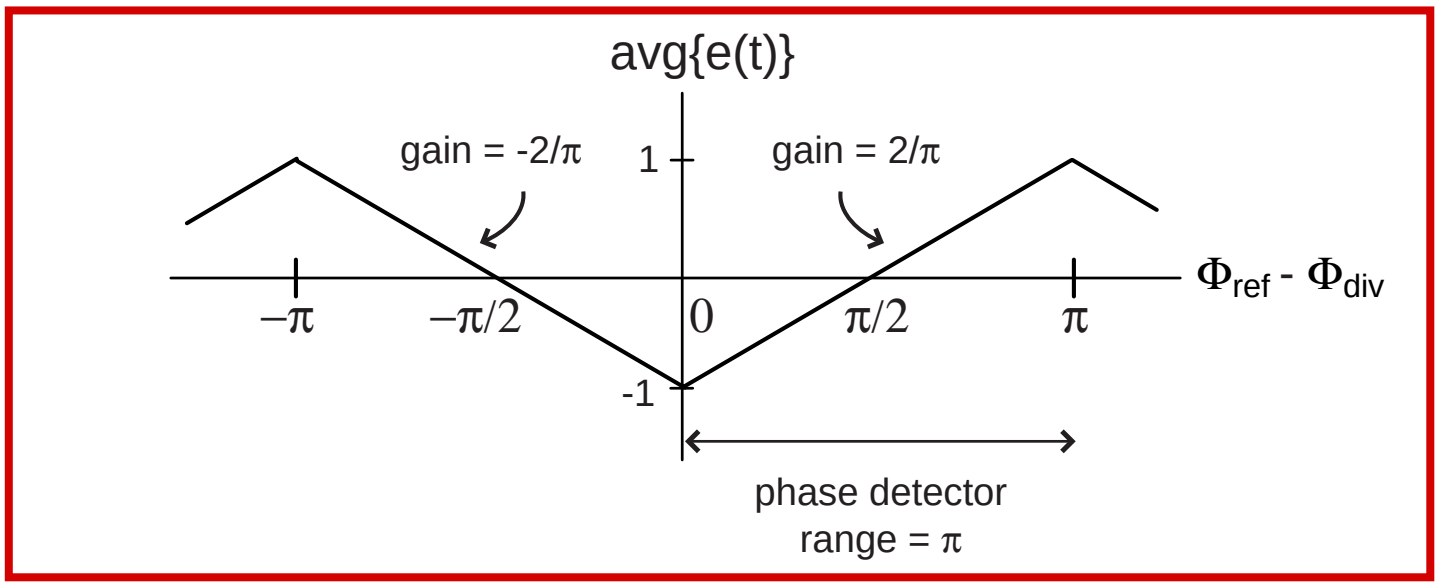
Phase Detector (PD)

- XOR structure
  - Average value of error pulses corresponds to phase error
  - Loop filter extracts the average value and feeds to VCO



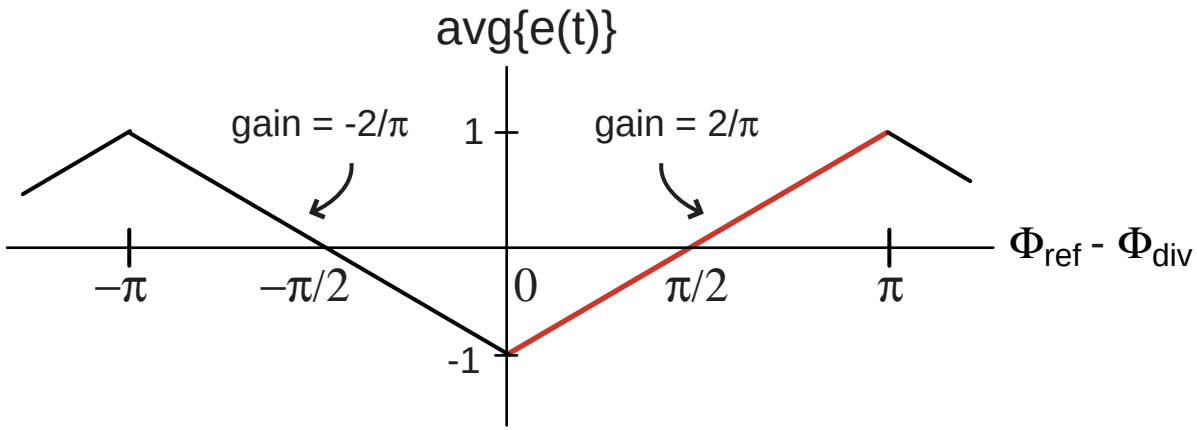
XOR Phase Detector Characteristic



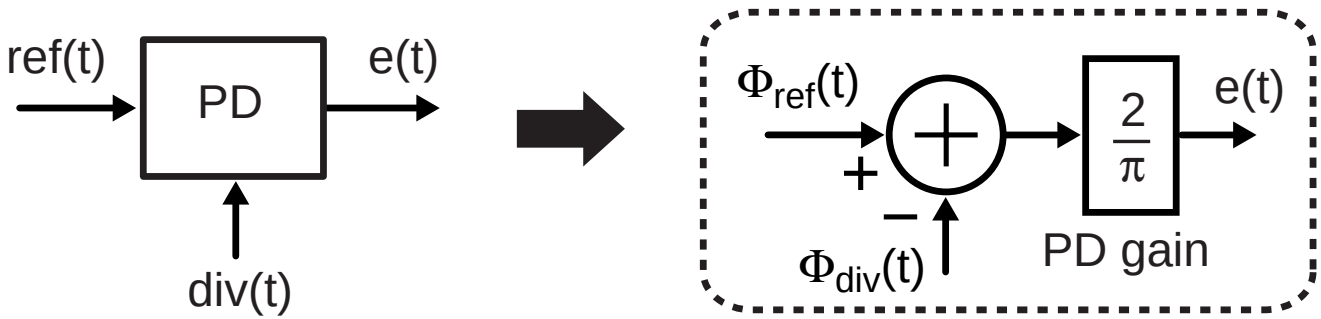


Frequency-Domain Model of XOR Phase Detector

- **Assume phase difference confined within 0 to π radians**
  - **Phase detector characteristic looks like a constant gain element**

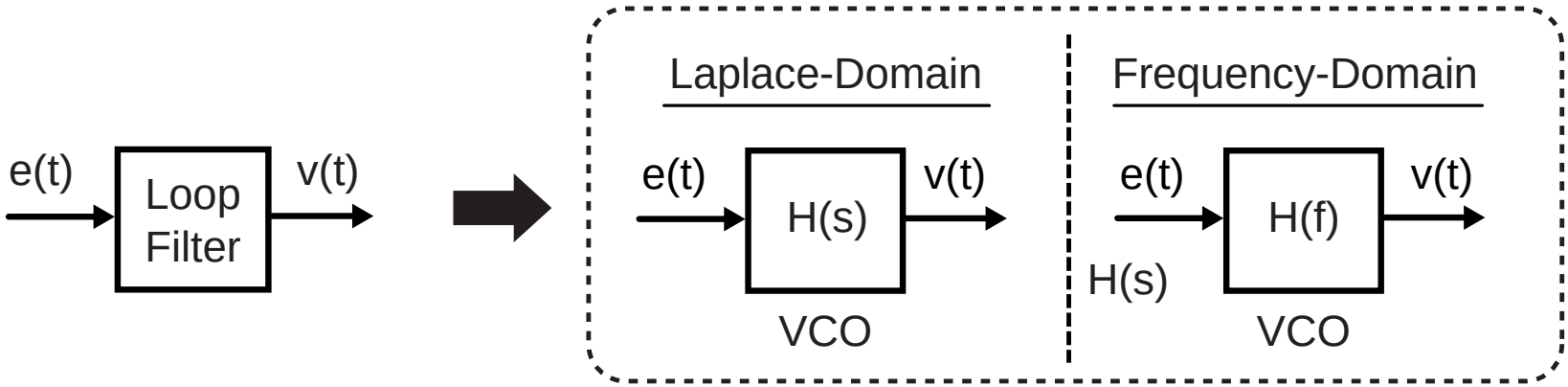


- **Corresponding frequency-domain model**

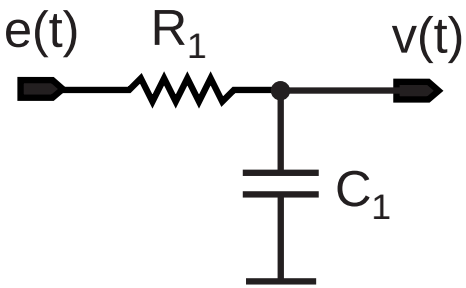


Loop Filter

- **Consists of a lowpass filter to extract average of phase detector error pulses**
- **Frequency-domain model**



- **First order example**

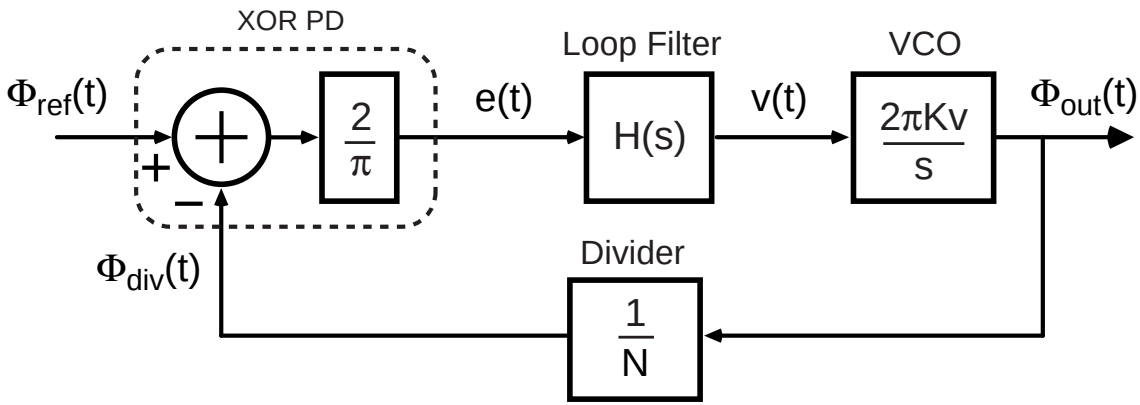


$$\Rightarrow \; H(s) = \frac{1}{1 + sR_1C_1}$$

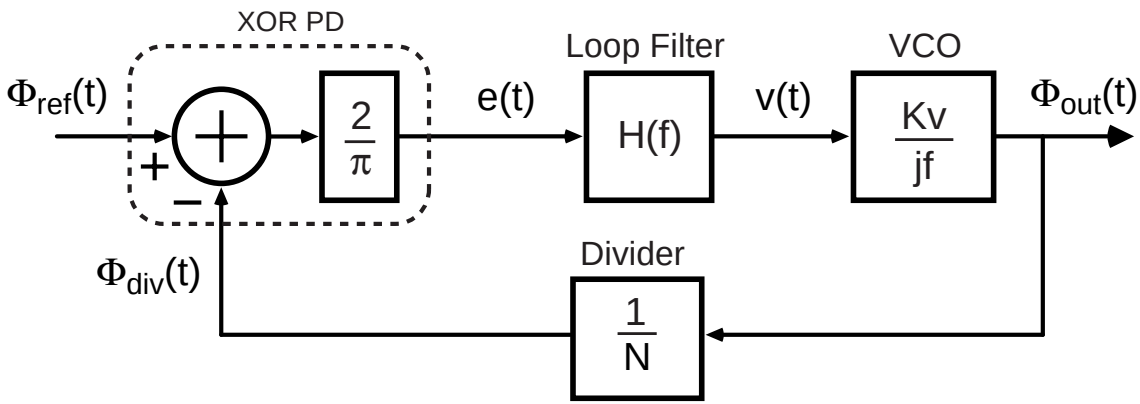
Overall Linearized PLL Frequency-Domain Model

- **Combine models of individual components**

Laplace-Domain Model

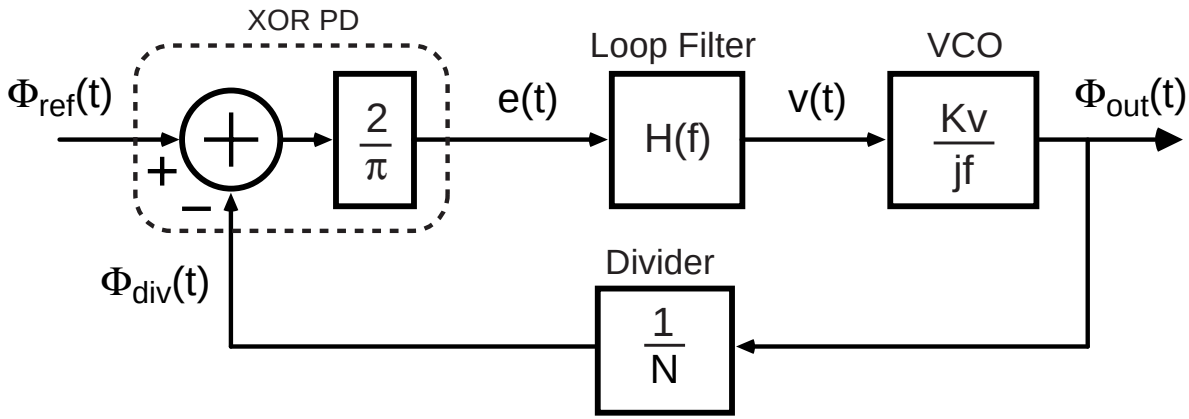


Frequency-Domain Model



Open Loop versus Closed Loop Response

- **Frequency-domain model**



- **Define A(f) as *open loop* response**

$$A(f) = \frac{2}{\pi} H(f) \left(\frac{K_v}{jf} \right) \frac{1}{N}$$

- **Define G(f) as a parameterizing *closed loop* function**
  - **More details later in this lecture**

$$G(f) = \frac{A(f)}{1 + A(f)}$$

Classical PLL Transfer Function Design Approach

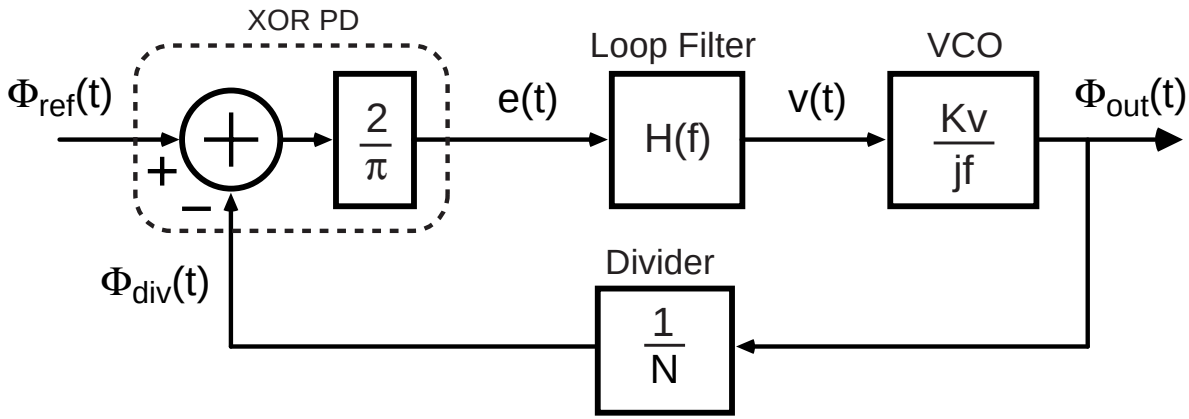
1. **Choose an appropriate topology for H(f)**
   - **Usually chosen from a small set of possibilities**

2. **Choose pole/zero values for H(f) as appropriate for the required filtering of the phase detector output**
   - **Constraint: set pole/zero locations higher than desired PLL bandwidth to allow stable dynamics to be possible**

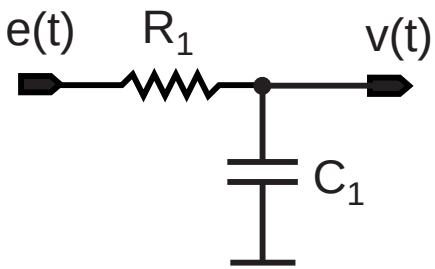
3. **Adjust the open-loop gain to achieve the required bandwidth while maintaining stability**
   - **Plot gain and phase bode plots of A(f)**
   - **Use phase (or gain) margin criterion to infer stability**

Example: First Order Loop Filter

- **Overall PLL block diagram**

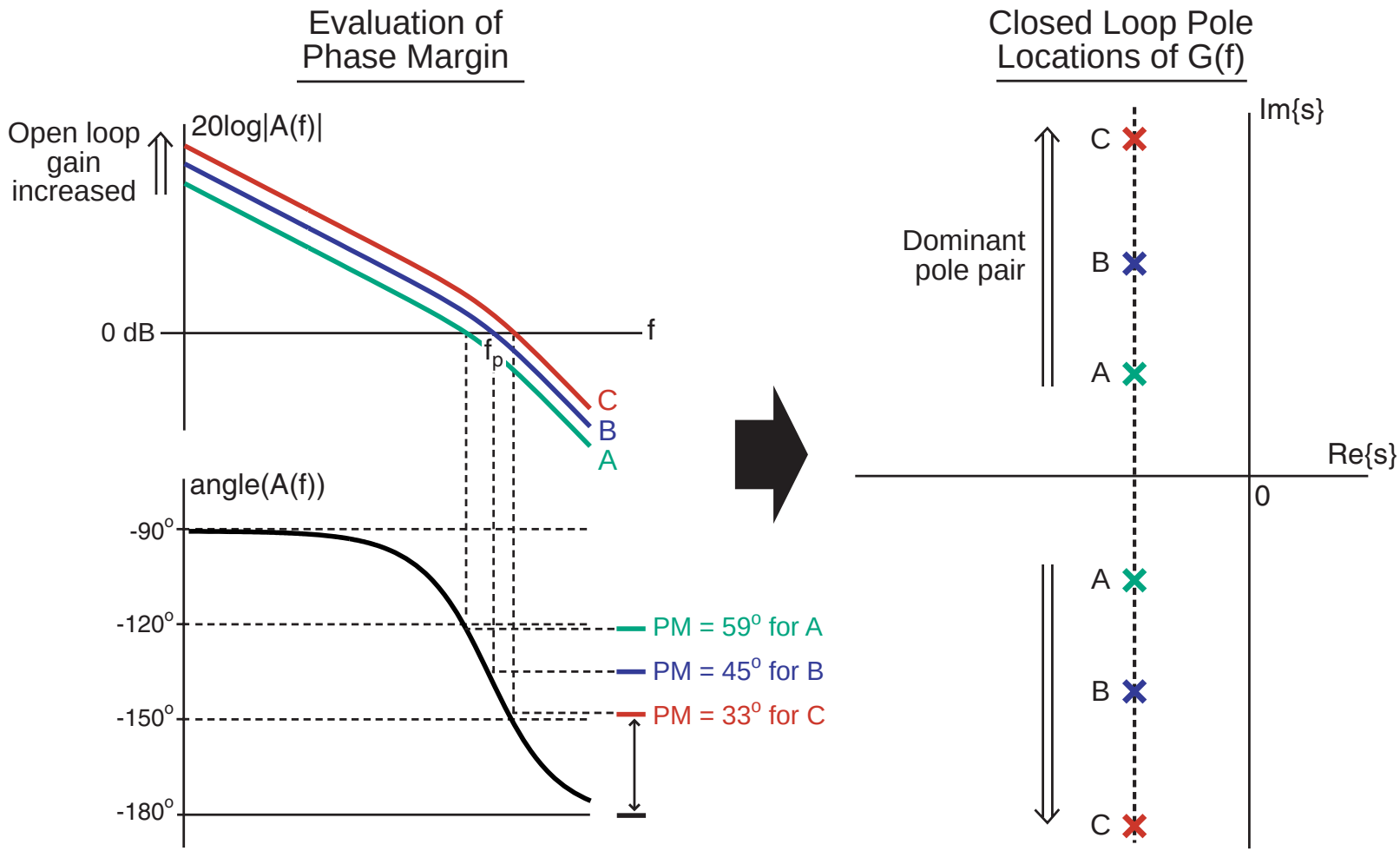


- **Loop filter**



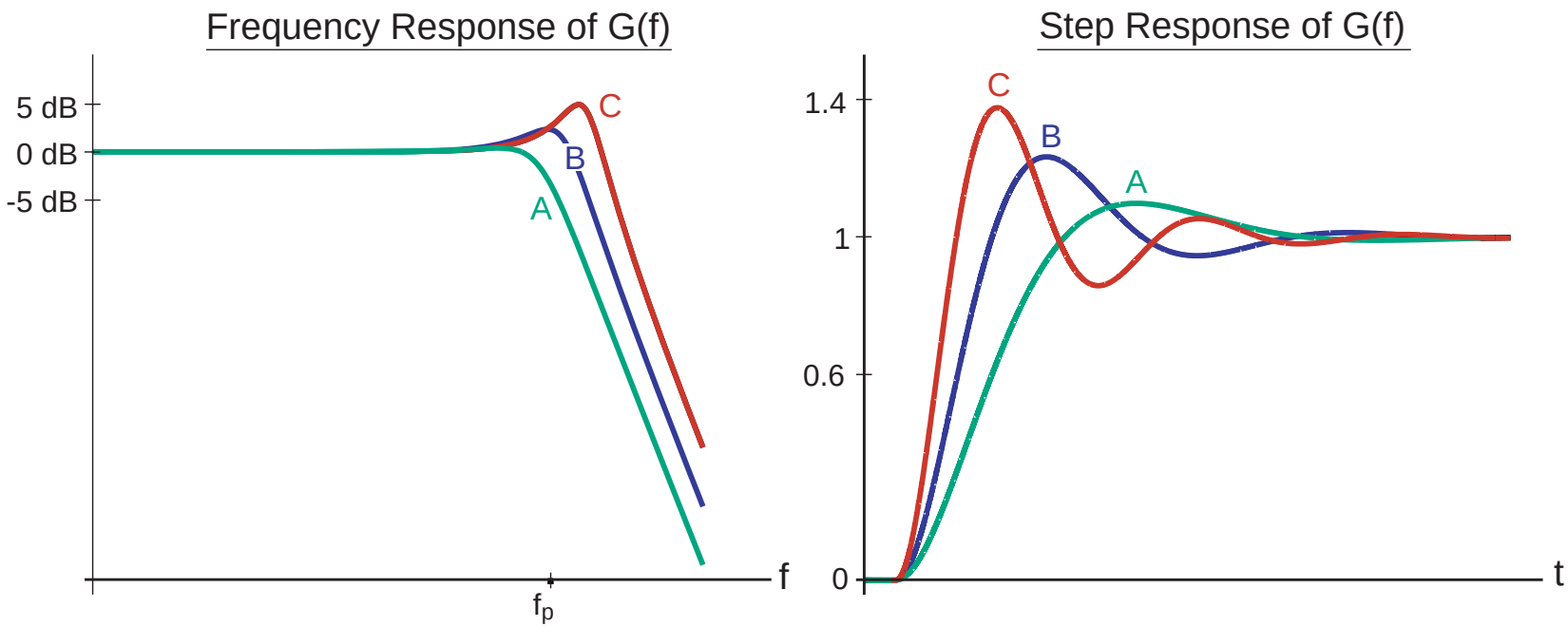
$$\Rightarrow \; H(f) = \frac{1}{1 + jf/f_p}$$

Closed Loop Poles Versus Open Loop Gain



- **Higher open loop gain leads to an increase in Q of closed loop poles**

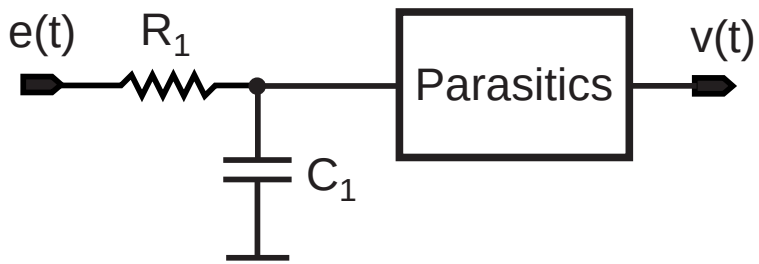
Corresponding Closed Loop Response



- **Increase in open loop gain leads to**
  - **Peaking in closed loop frequency response**
  - **Ringing in closed loop step response**

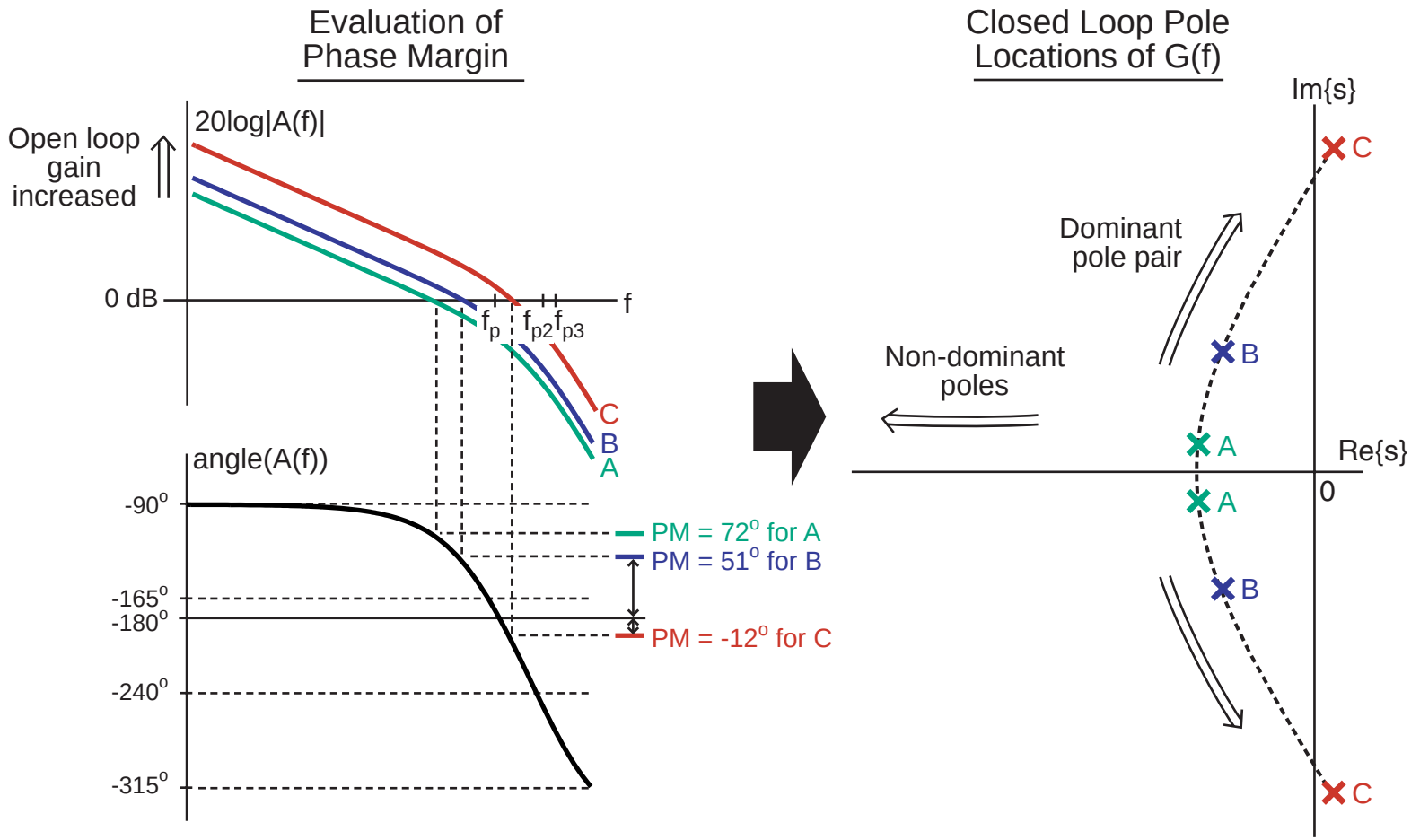
The Impact of Parasitic Poles

- Loop filter and VCO may have additional parasitic poles and zeros due to their circuit implementation
- We can model such parasitics by including them in the loop filter transfer function
- Example: add two parasitic poles to first order filter

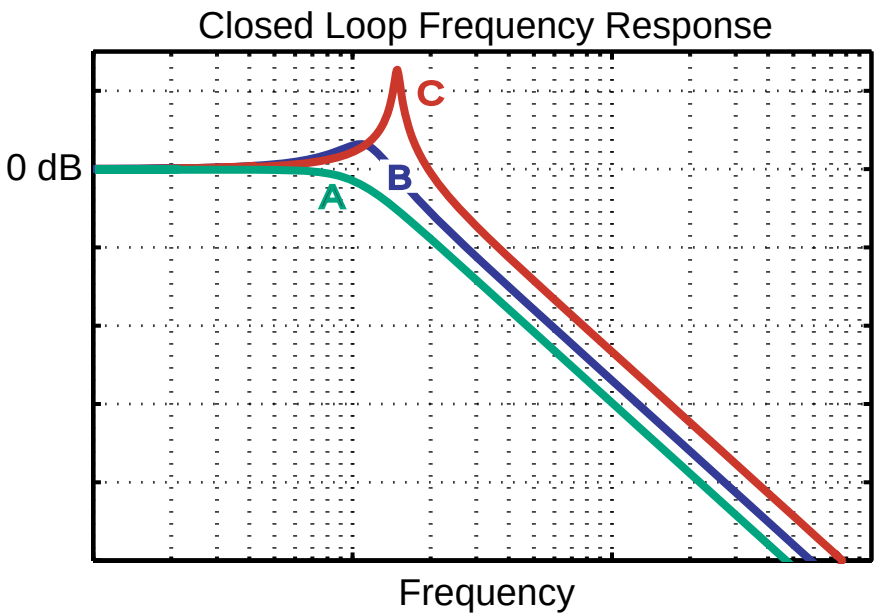


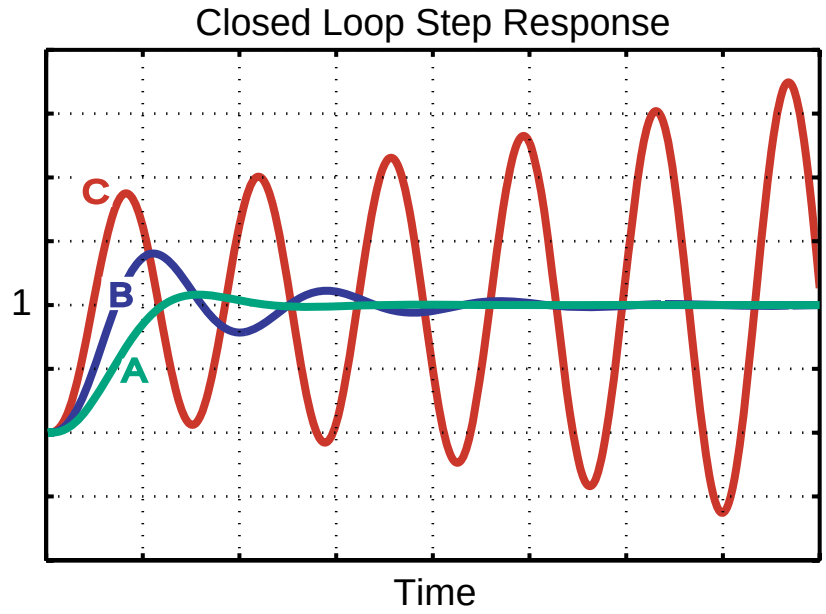
$$\Rightarrow \; H(f) = \left(\frac{1}{1+jf/f_1}\right)\left(\frac{1}{1+jf/f_2}\right)\left(\frac{1}{1+jf/f_3}\right)$$

Closed Loop Poles Versus Open Loop Gain



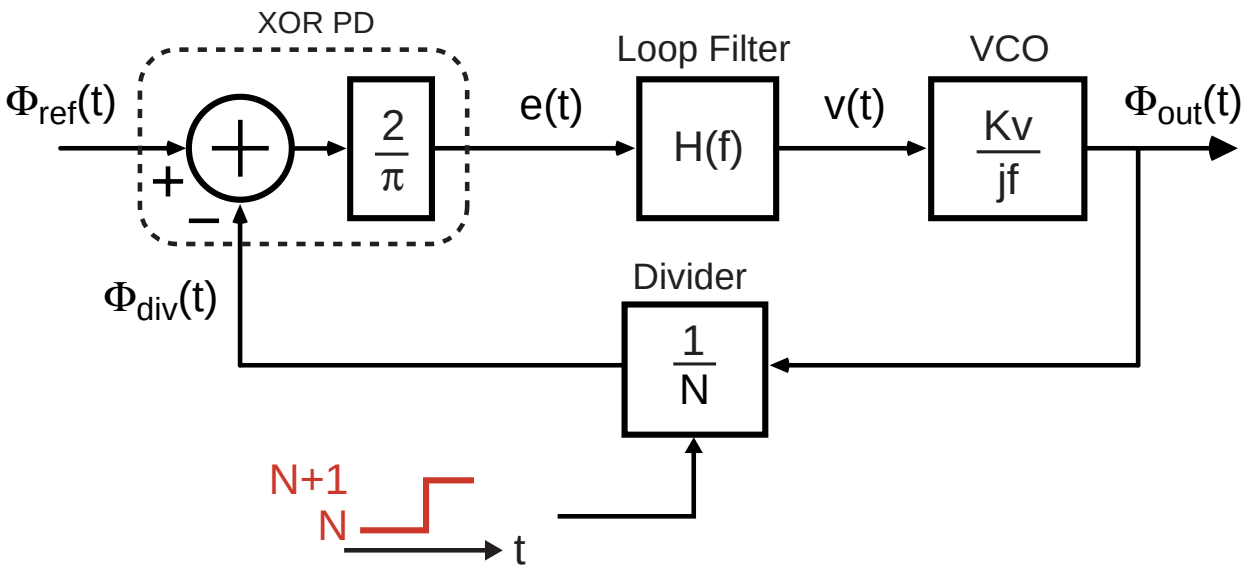
Corresponding Closed Loop Response





- **Increase in open loop gain now eventually leads to instability**
  - **Large peaking in closed loop frequency response**
  - **Increasing amplitude in closed loop step response**

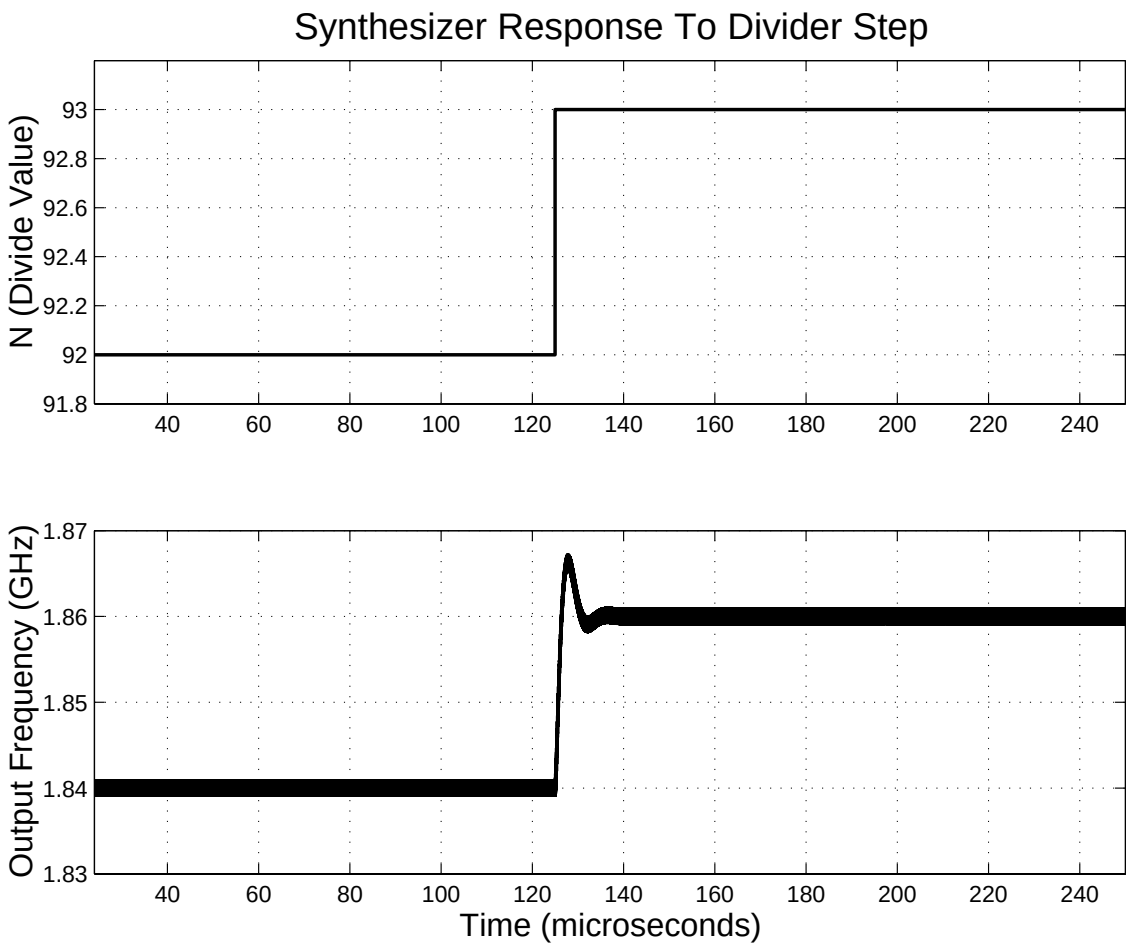
Response of PLL to Divide Value Changes



- **Change in output frequency achieved by changing the divide value**
- **Classical approach provides no direct model of impact of divide value variations**
  - **Treat divide value variation as a perturbation to a linear system**
    - PLL responds according to its closed loop response

Response of an Actual PLL to Divide Value Change

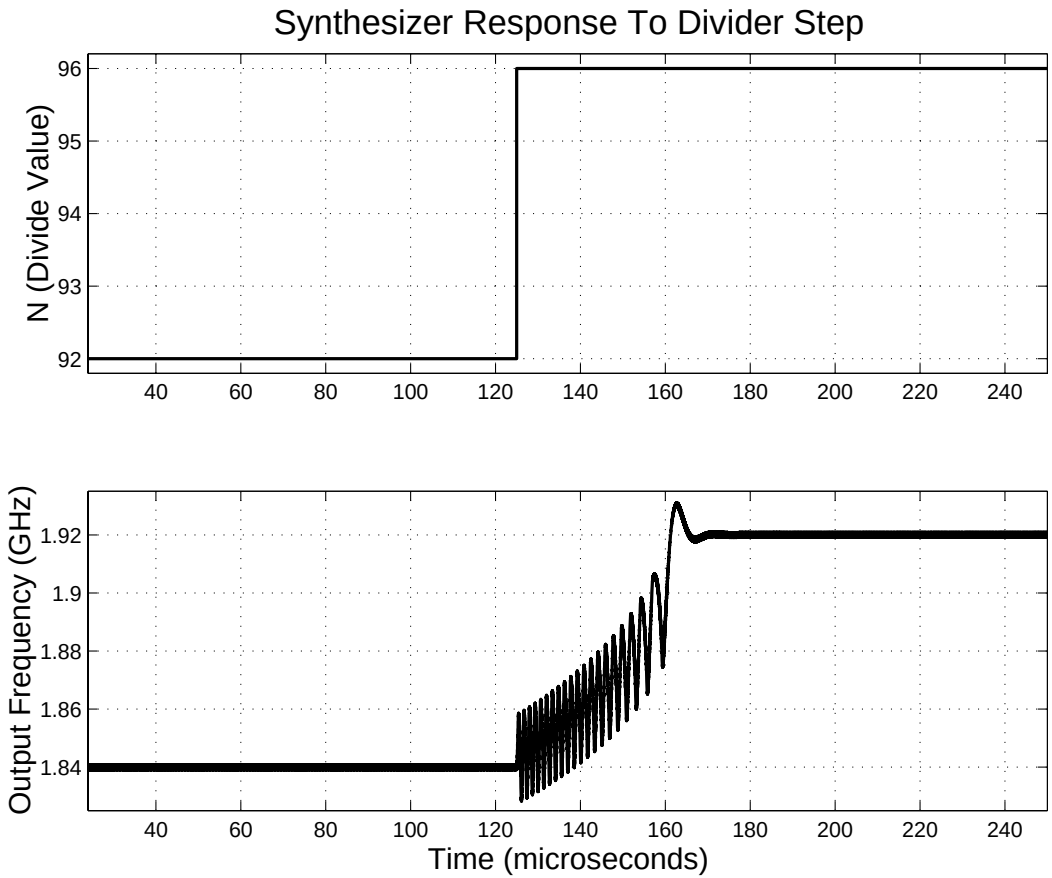
- **Example: Change divide value by one**



- **PLL responds according to closed loop response!**

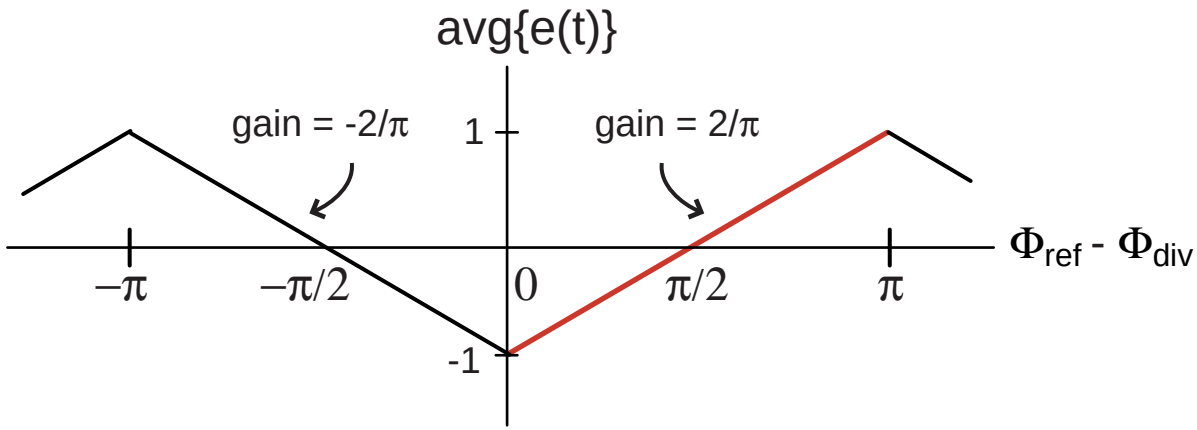
What Happens with Large Divide Value Variations?

- PLL temporarily loses frequency lock (cycle slipping occurs)



  - Why does this happen?

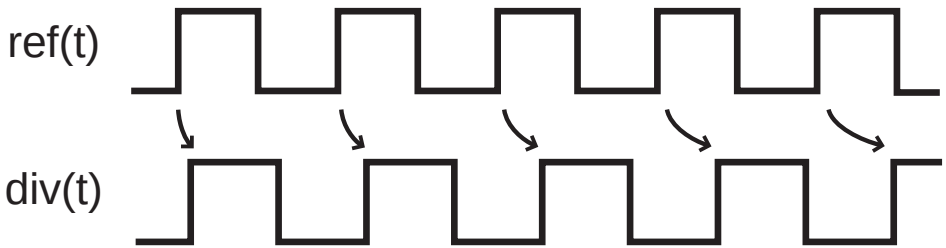
Recall Phase Detector Characteristic



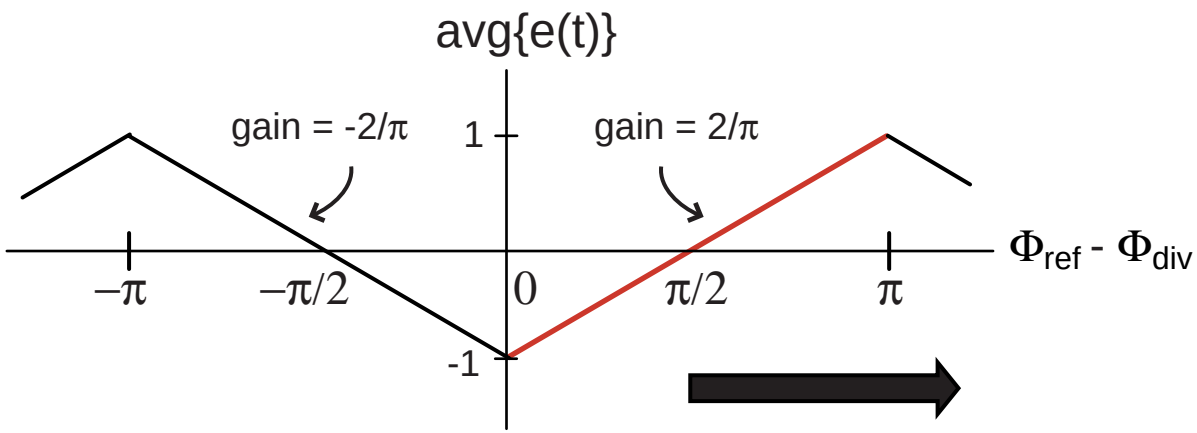
- To simplify modeling, we assumed that we always operated in a confined phase range (0 to π)
  - Led to a simple PD model
- Large perturbations knock us out of that confined phase range
  - PD behavior varies depending on the phase range it happens to be in

Cycle Slipping

- Consider the case where there is a frequency offset between divider output and reference
  - We know that phase difference will accumulate

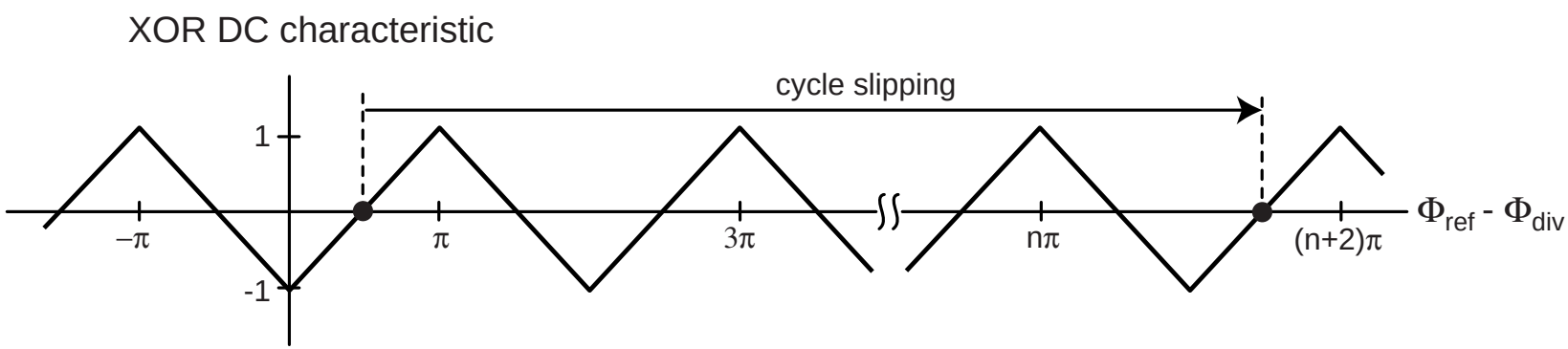


- Resulting ramp in phase causes PD characteristic to be swept across its different regions (cycle slipping)



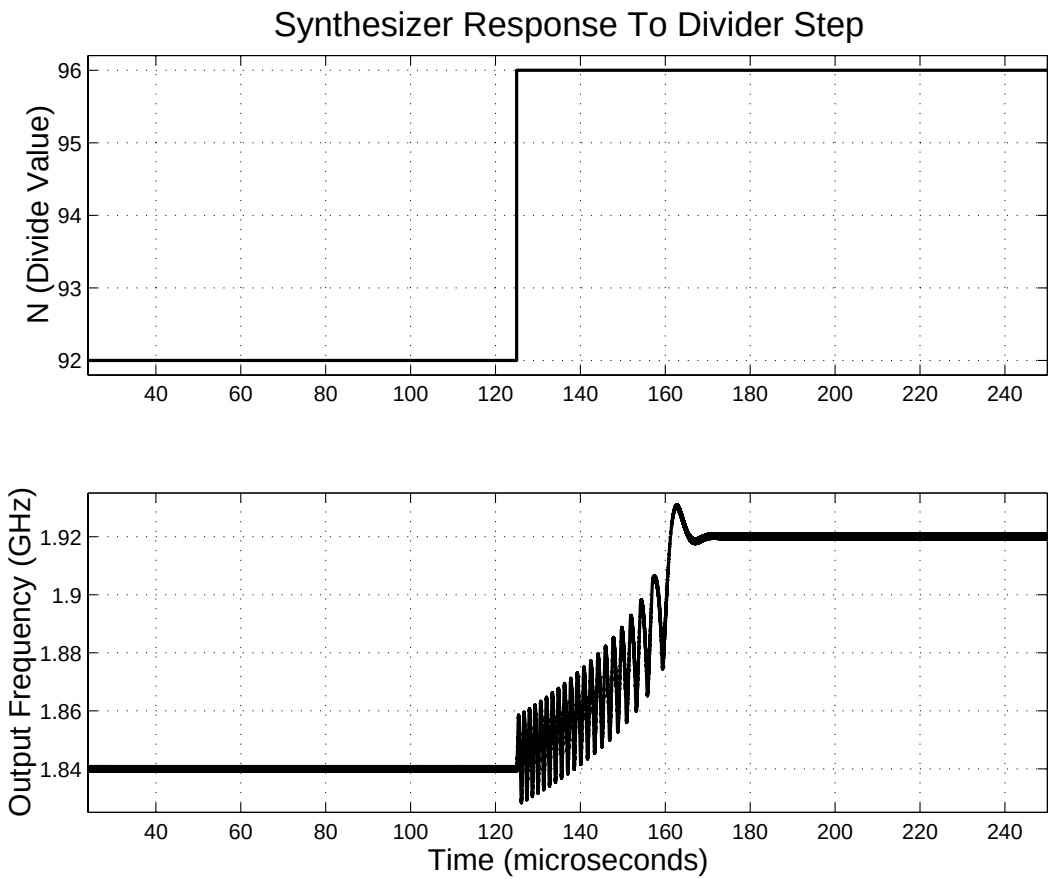
Impact of Cycle Slipping

- **Loop filter averages out phase detector output**
- **Severe cycle slipping causes phase detector to alternate between regions very quickly**
  - **Average value of XOR characteristic can be close to zero**
  - **PLL frequency oscillates according to cycle slipping**
  - **In severe cases, PLL will not re-lock**
    - PLL has finite frequency lock-in range!



Back to PLL Response Shown Previously

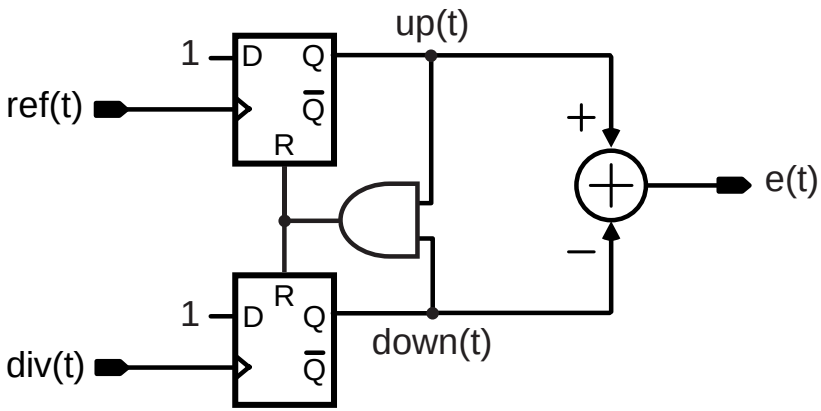
- PLL output frequency indeed oscillates
  - Eventually locks when frequency difference is small enough

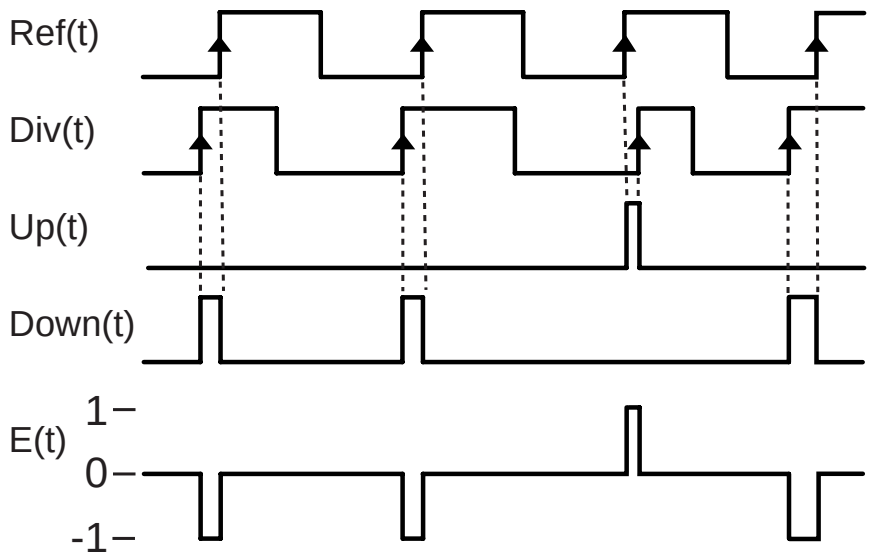


  - How do we extend the frequency lock-in range?

Phase Frequency Detectors (PFD)

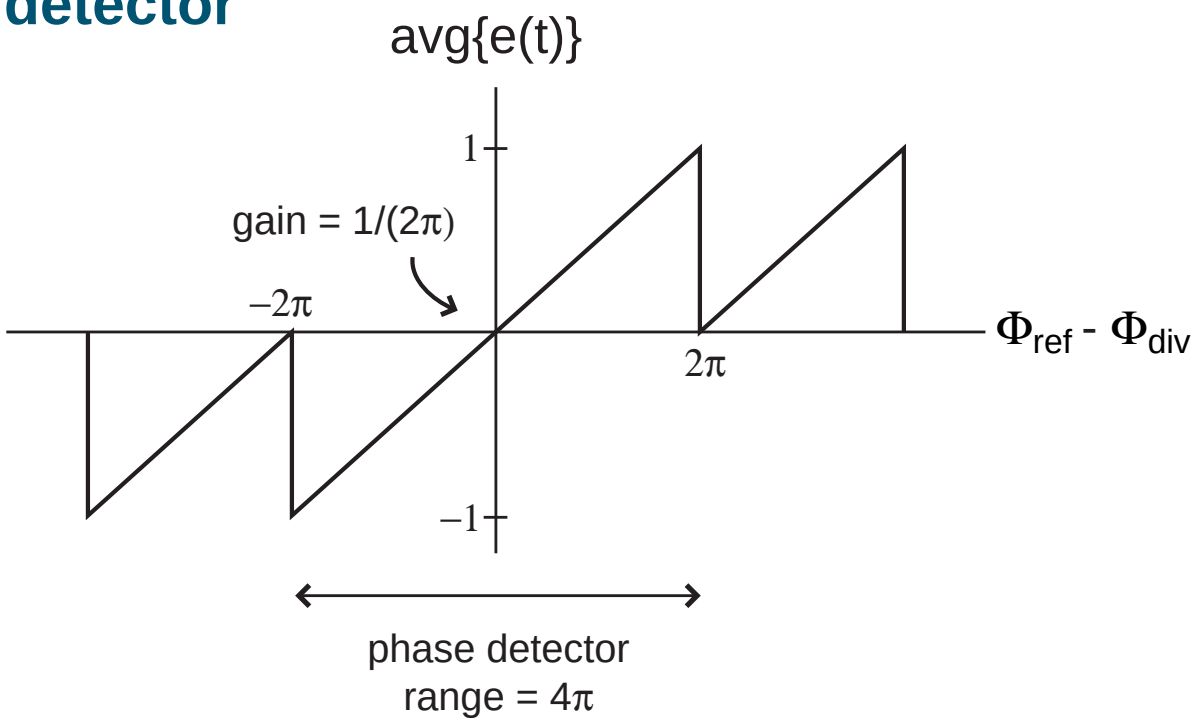
- **Example: Tristate PFD**





Tristate PFD Characteristic

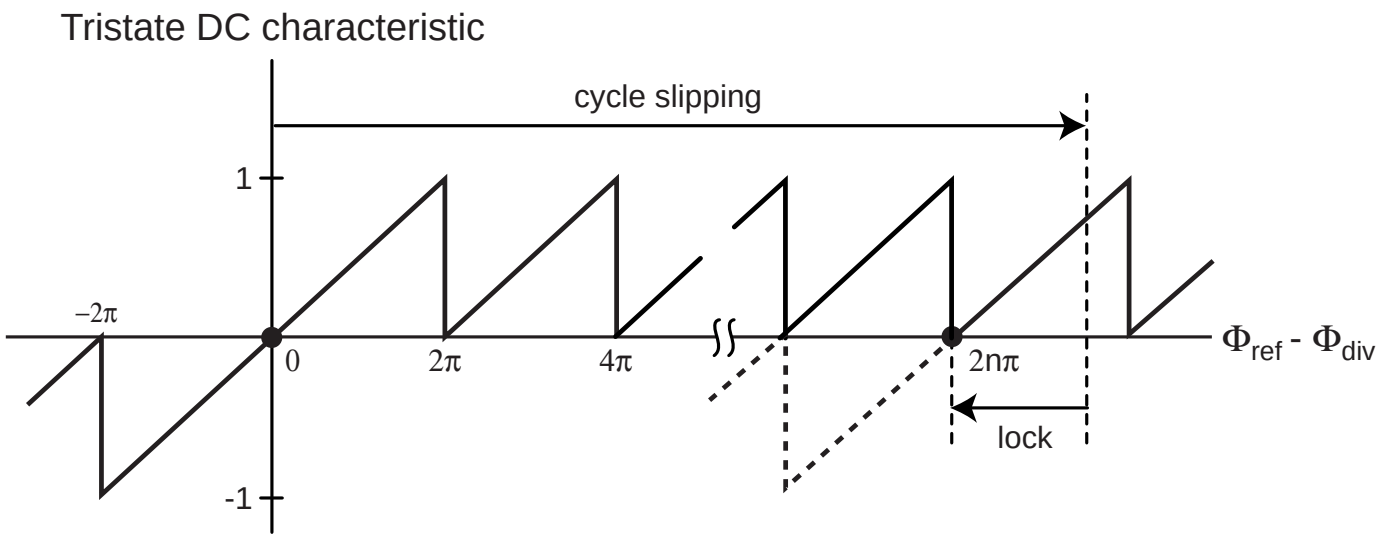
- **Calculate using similar approach as used for XOR phase detector**



- **Note that phase error characteristic is asymmetric about zero phase**
  - **Key attribute for enabling frequency detection**

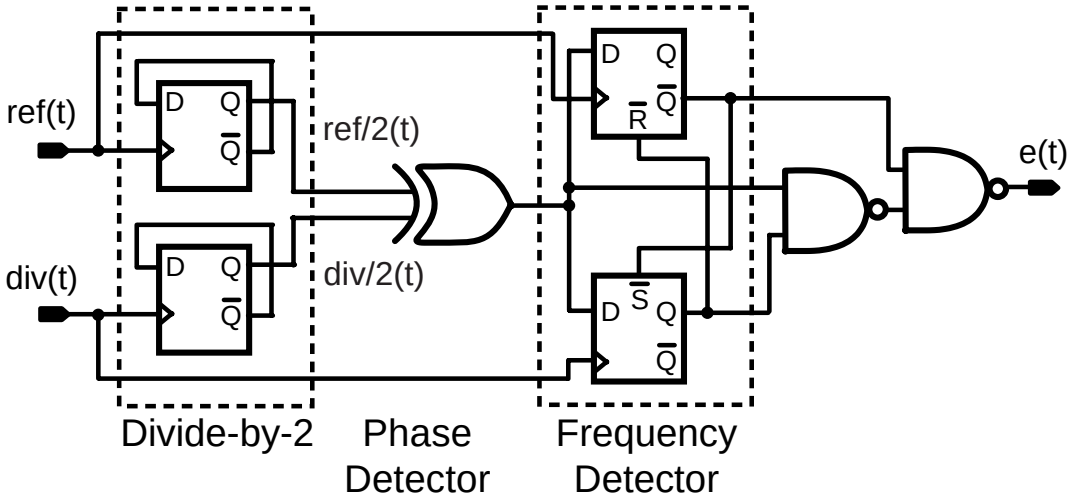
PFD Enables PLL to Always Regain Frequency Lock

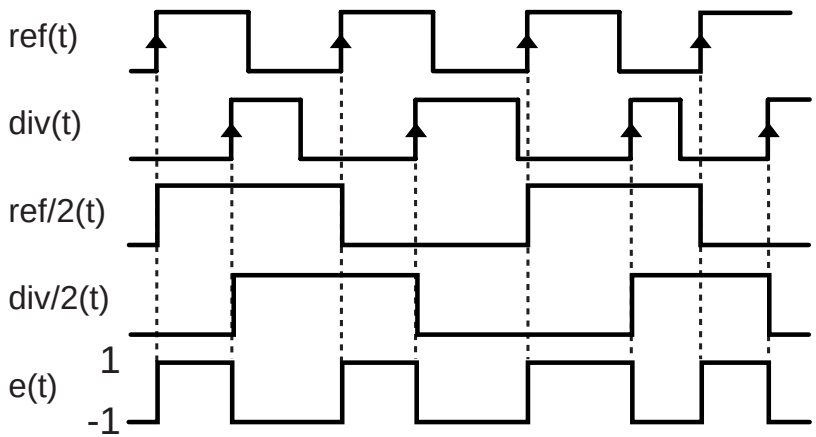
- **Asymmetric phase error characteristic allows positive frequency differences to be distinguished from negative frequency differences**
  - **Average value is now positive or negative according to sign of frequency offset**
  - **PLL will always relock**



Another PFD Structure

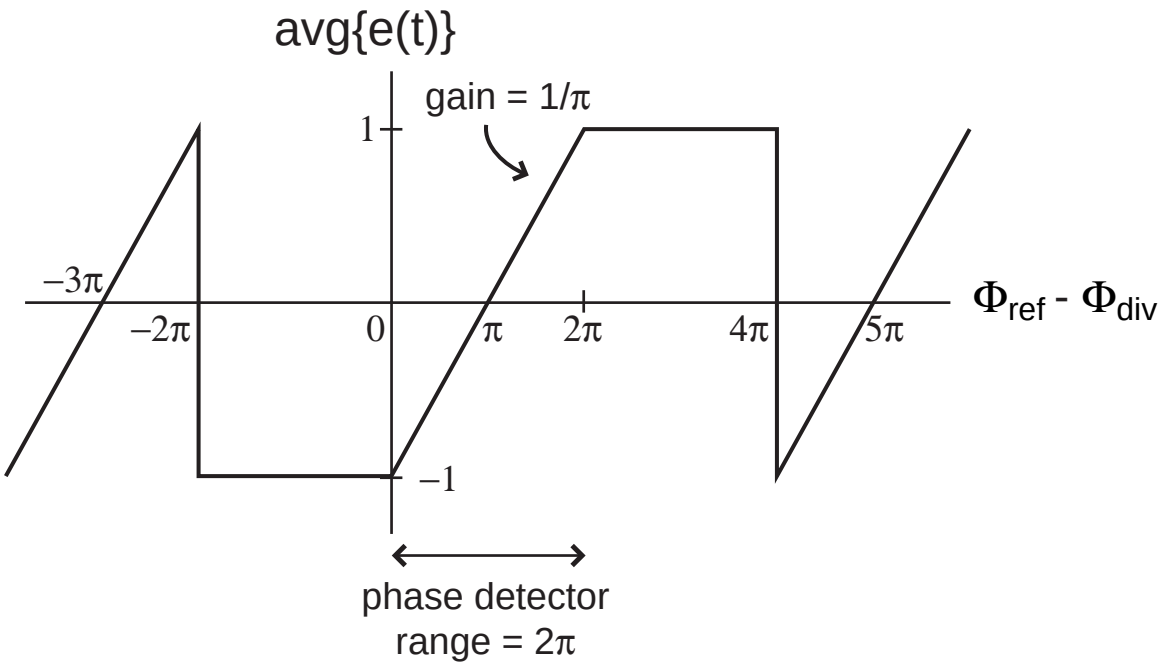
- **XOR-based PFD**





XOR-based PFD Characteristic

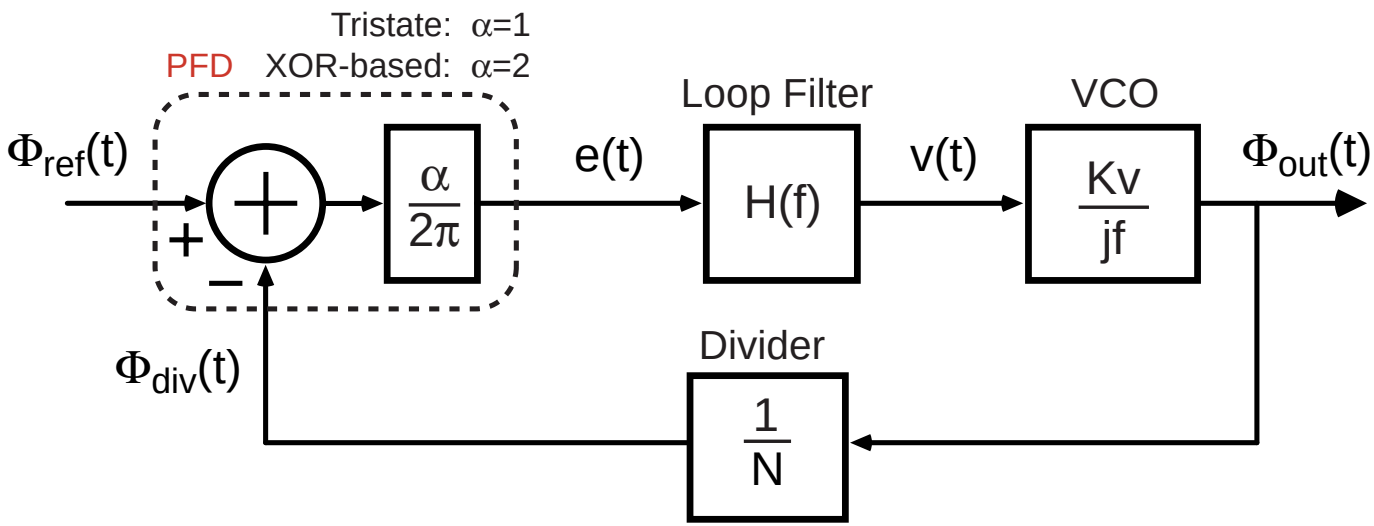
- **Calculate using similar approach as used for XOR phase detector**



- **Phase error characteristic asymmetric about zero phase**
  - **Average value of phase error is positive or negative during cycle slipping depending on sign of frequency error**

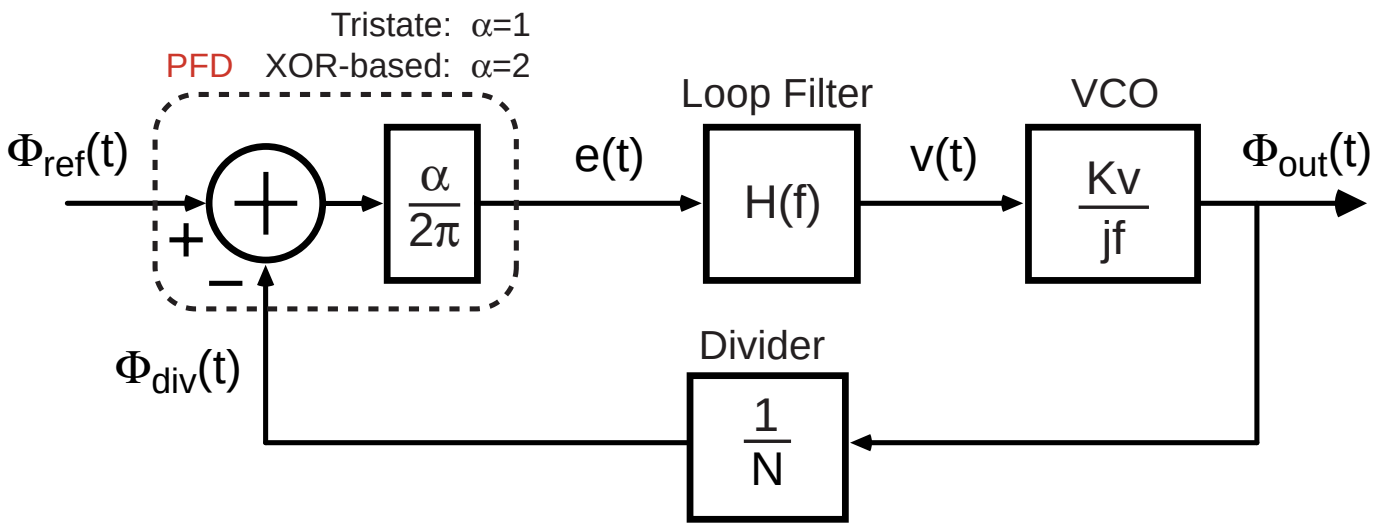
Linearized PLL Model With PFD Structures

- Assume that when PLL in lock, phase variations are within the linear range of PFD
  - Simulate impact of cycle slipping if desired (do not include its effect in model)
- Same frequency-domain PLL model as before, but PFD gain depends on topology used



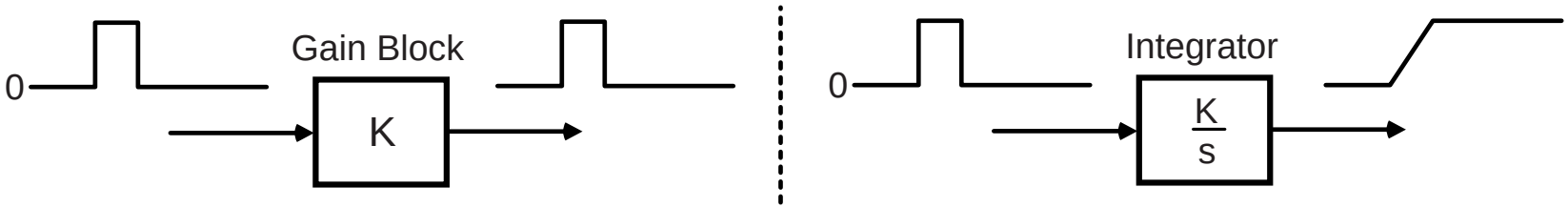
Type I versus Type II PLL Implementations

- **Type I: one integrator in PLL open loop transfer function**
  - **VCO adds on integrator**
  - **Loop filter, H(f), has no integrators**
- **Type II: two integrators in PLL open loop transfer function**
  - **Loop filter, H(f), has one integrator**

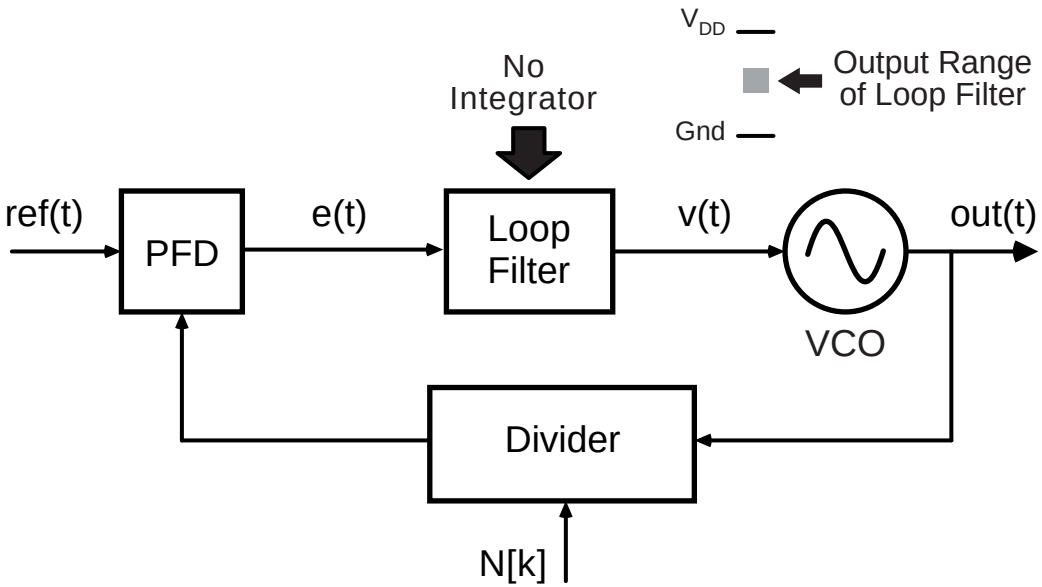


VCO Input Range Issue for Type I PLL Implementations

- DC output range of gain block versus integrator

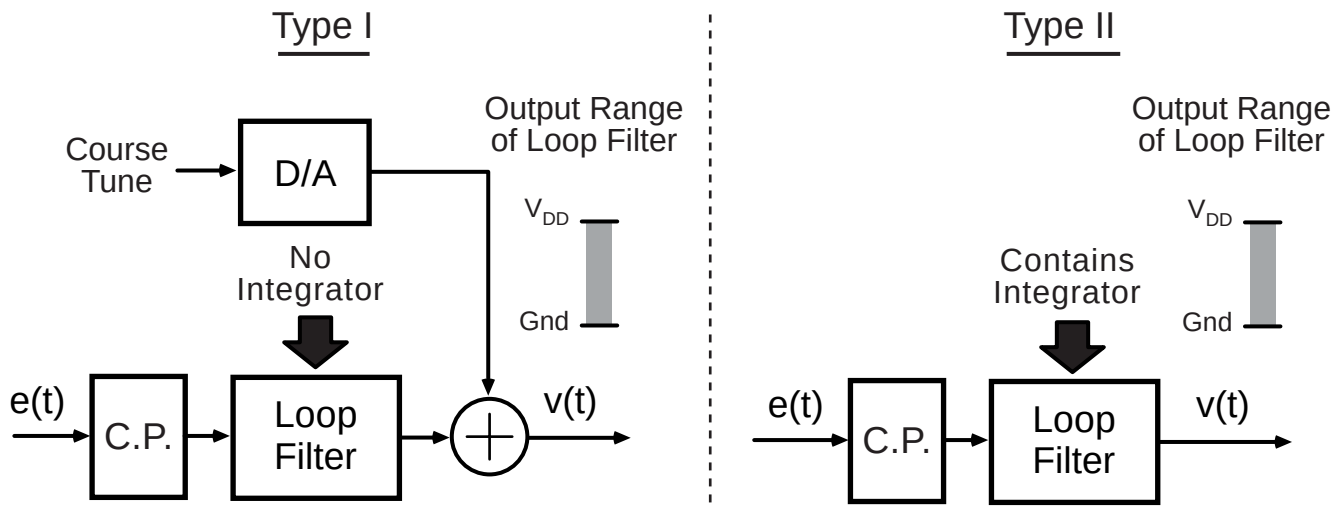


- Issue: DC gain of loop filter often small and PFD output range is limited
  - Loop filter output fails to cover full input range of VCO



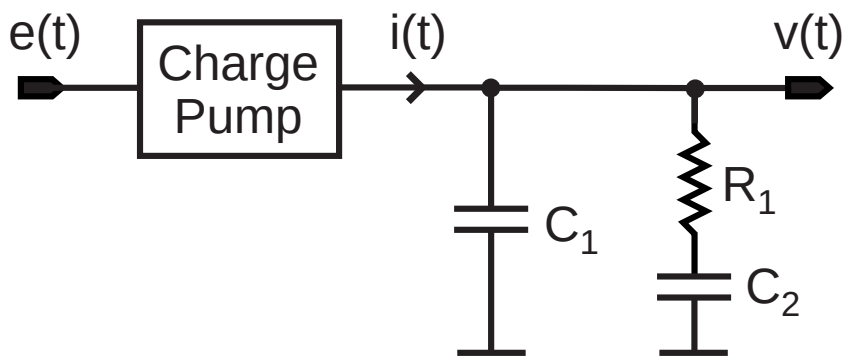
Options for Achieving Full Range Span of VCO

- **Type I**
  - **Add a D/A converter to provide coarse tuning**
    - Adds power and complexity
    - Steady-state phase error inconsistently set
- **Type II**
  - **Integrator automatically provides DC level shifting**
    - Low power and simple implementation
    - Steady-state phase error always set to zero



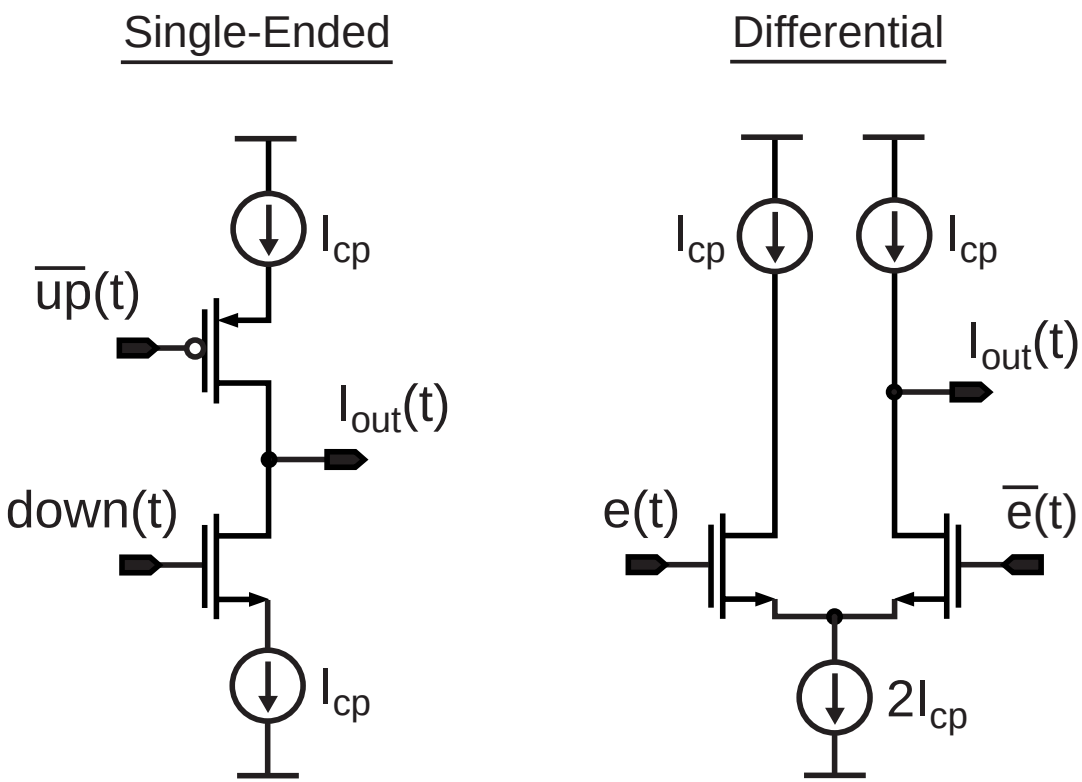
A Common Loop Filter for Type II PLL Implementation

- Use a charge pump to create the integrator
  - Current onto a capacitor forms integrator
  - Add extra pole/zero using resistor and capacitor
- Gain of loop filter can be adjusted according to the value of the charge pump current
- Example: lead/lag network



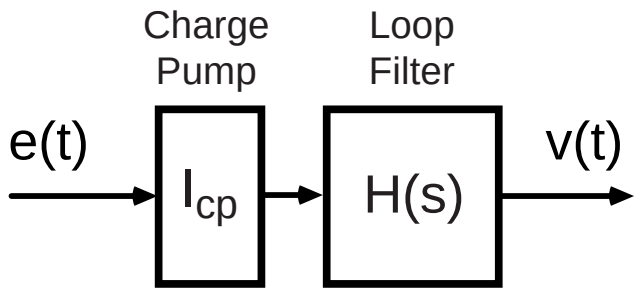
Charge Pump Implementations

- **Switch currents in and out:**



Modeling of Loop Filter/Charge Pump

- Charge pump is gain element
- Loop filter forms transfer function



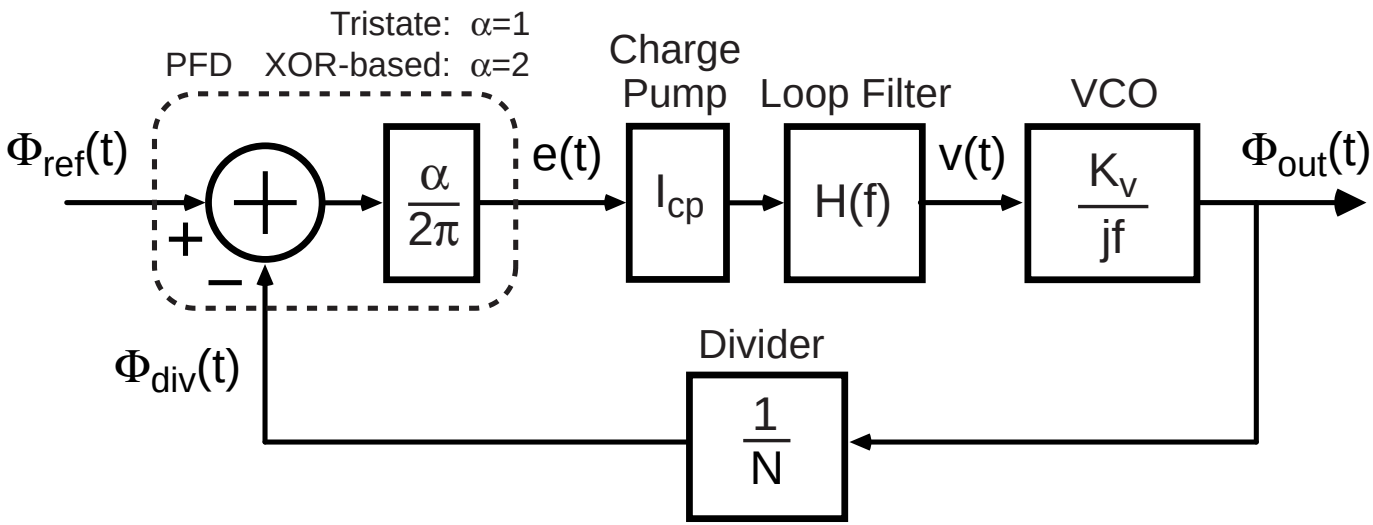
- Example: lead/lag network from previous slide

$$H(f) = \left(\frac{1}{sC_{sum}}\right)\frac{1+jf/f_z}{1+jf/f_p}$$

$$C_{sum} = C_1 + C_2, \quad f_z = \frac{1}{2\pi R_1 C_2}, \quad f_p = \frac{C_1 + C_2}{2\pi R_1 C_1 C_2}$$

PLL Design with Lead/Lag Filter

- **Overall PLL block diagram**

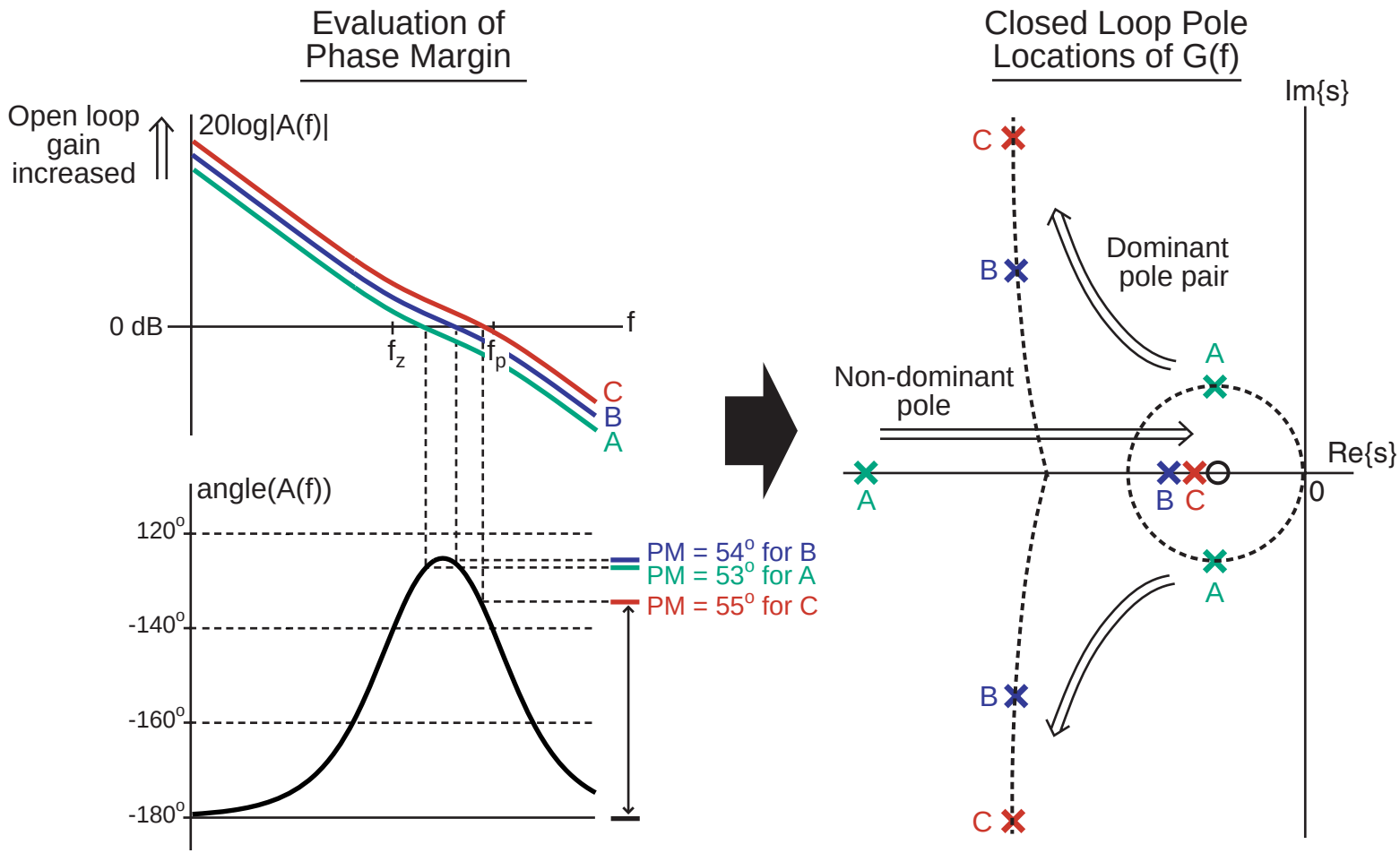


- **Loop filter**

$$H(f) = \left(\frac{1}{sC_{sum}}\right)\frac{1 + jf/f_z}{1 + jf/f_p}$$

- **Set open loop gain to achieve adequate phase margin**
  - **Set f_z lower than and f_p higher than desired PLL bandwidth**

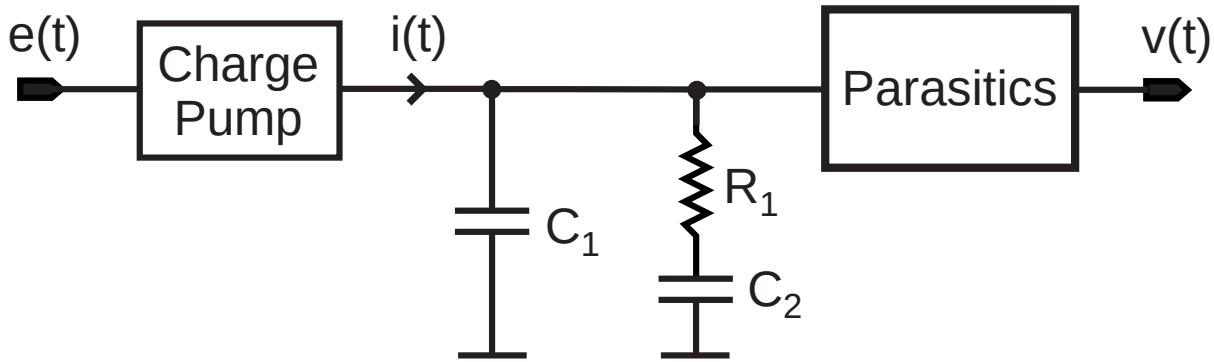
Closed Loop Poles Versus Open Loop Gain



- **Open loop gain cannot be too low or too high if reasonable phase margin is desired**

Impact of Parasitics When Lead/Lag Filter Used

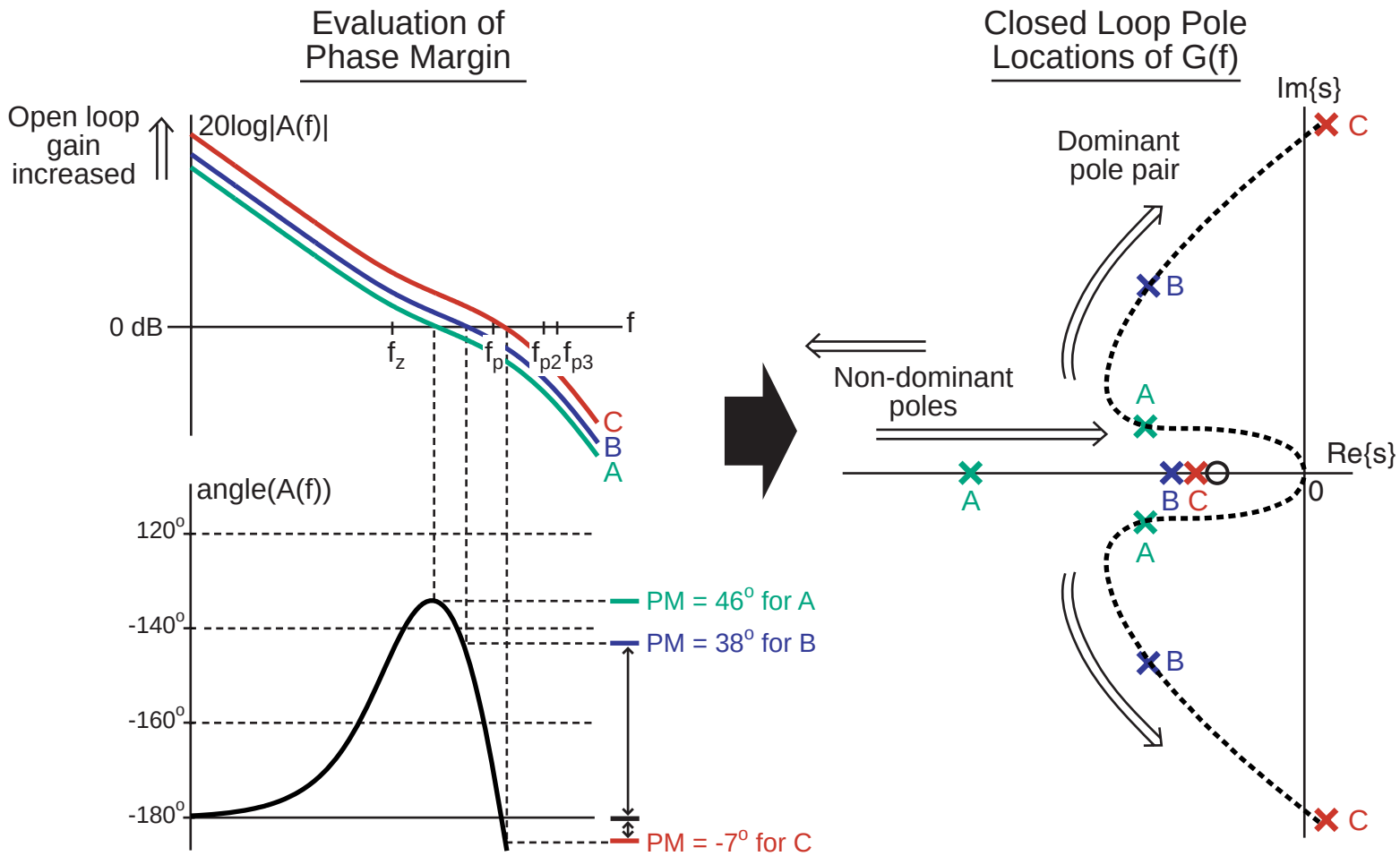
- **We can again model impact of parasitics by including them in loop filter transfer function**



- **Example: include two parasitic poles with the lead/lag transfer function**

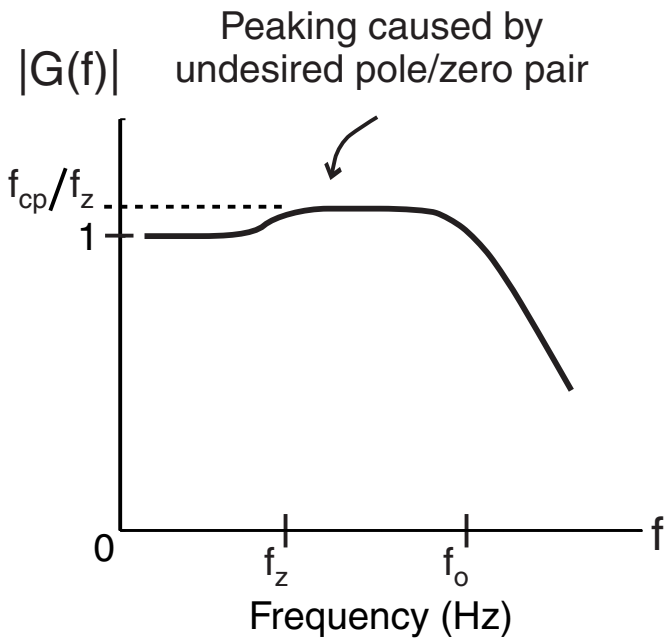
$$H(f) = \left(\frac{1}{sC_{sum}}\right)\frac{1+jf/f_z}{1+jf/f_p}\left(\frac{1}{1+jf/f_{p2}}\right)\left(\frac{1}{1+jf/f_{p3}}\right)$$

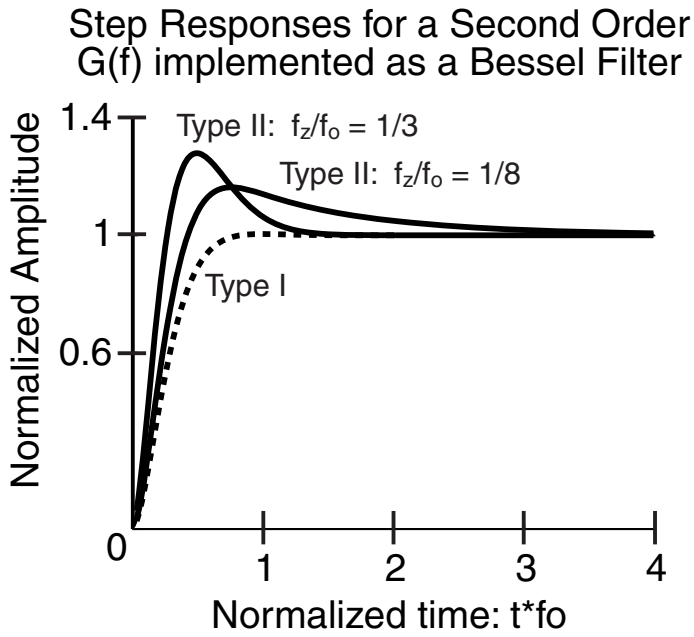
Closed Loop Poles Versus Open Loop Gain



- **Closed loop response becomes unstable if open loop gain is too high**

Negative Issues For Type II PLL Implementations



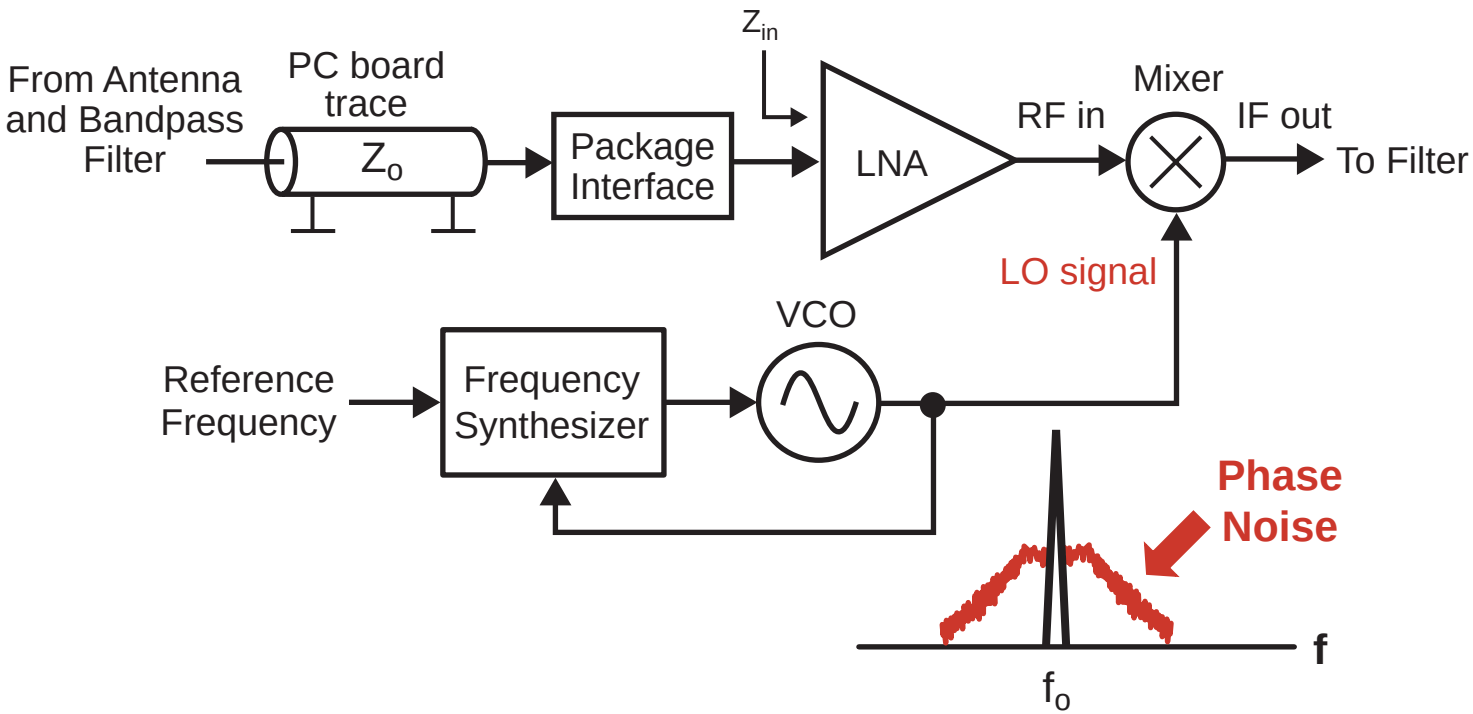


- **Parasitic pole/zero pair causes**
  - **Peaking in the closed loop frequency response**
  - **Extended settling time due to parasitic "tail" response**
    - Bad for wireless systems demanding fast settling time

Summary of Integer-N Dynamic Modeling

- **Linearized models can be derived for each PLL block**
  - **Resulting transfer function model of PLL is accurate for small perturbations in PLL**
  - **Linear PLL model breaks down for large perturbations on PLL, such as a large step change in frequency**
    - Cycle slipping is key nonlinear effect
- **Key issues for designing PLL are**
  - **Achieve stable operation with desired bandwidth**
  - **Allow full range of VCO with a simple implementation**
    - Type II PLL is very popular to achieve this

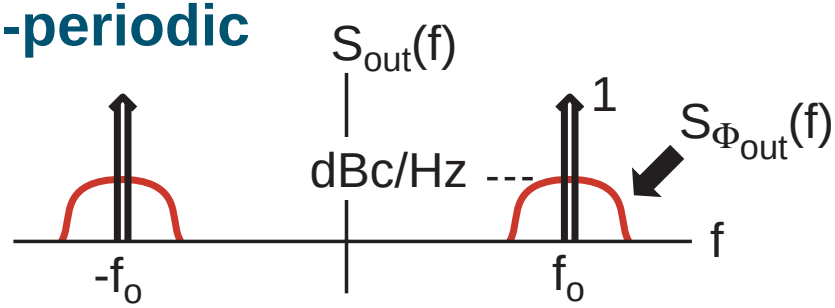
Frequency Synthesizer Noise in Wireless Systems



- **Synthesizer noise has a negative impact on system**
  - **Receiver – lower sensitivity, poorer blocking performance**
  - **Transmitter – increased spectral emissions (output spectrum must meet a mask requirement)**
- **Noise is characterized in frequency domain**

Phase Noise Versus Spurious Noise

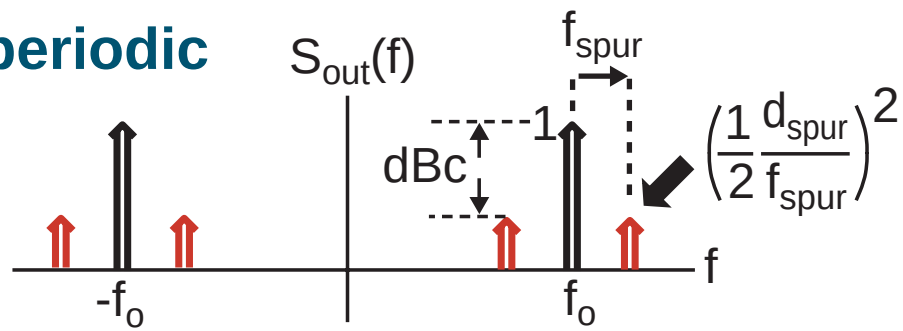
- **Phase noise is non-periodic**



  - **Described as a spectral density relative to carrier power**

$$L(f) = 10\log(S_{\Phi_{out}}(f))\ \text{dBc/Hz}$$

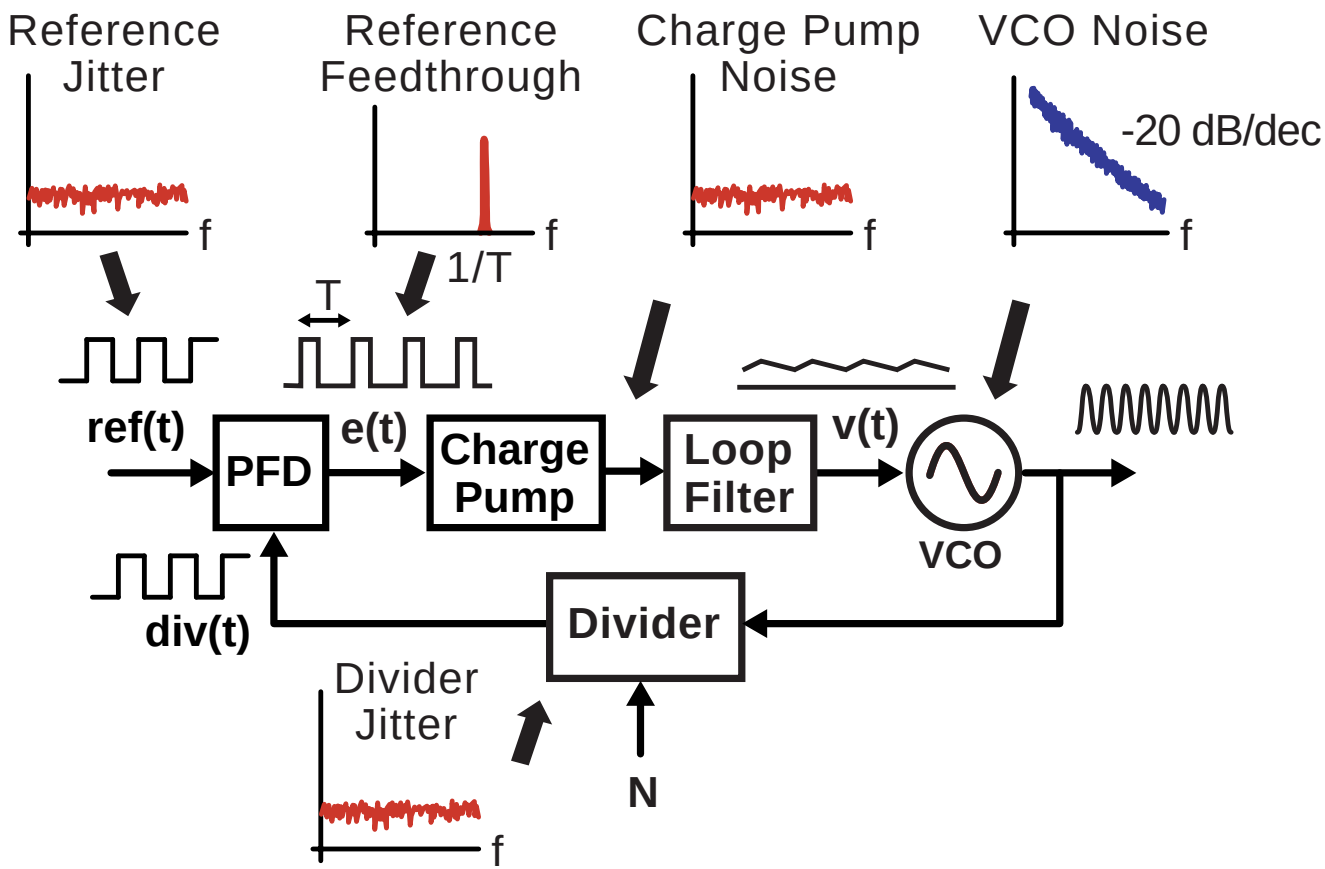
- **Spurious noise is periodic**



  - **Described as tone power relative to carrier power**

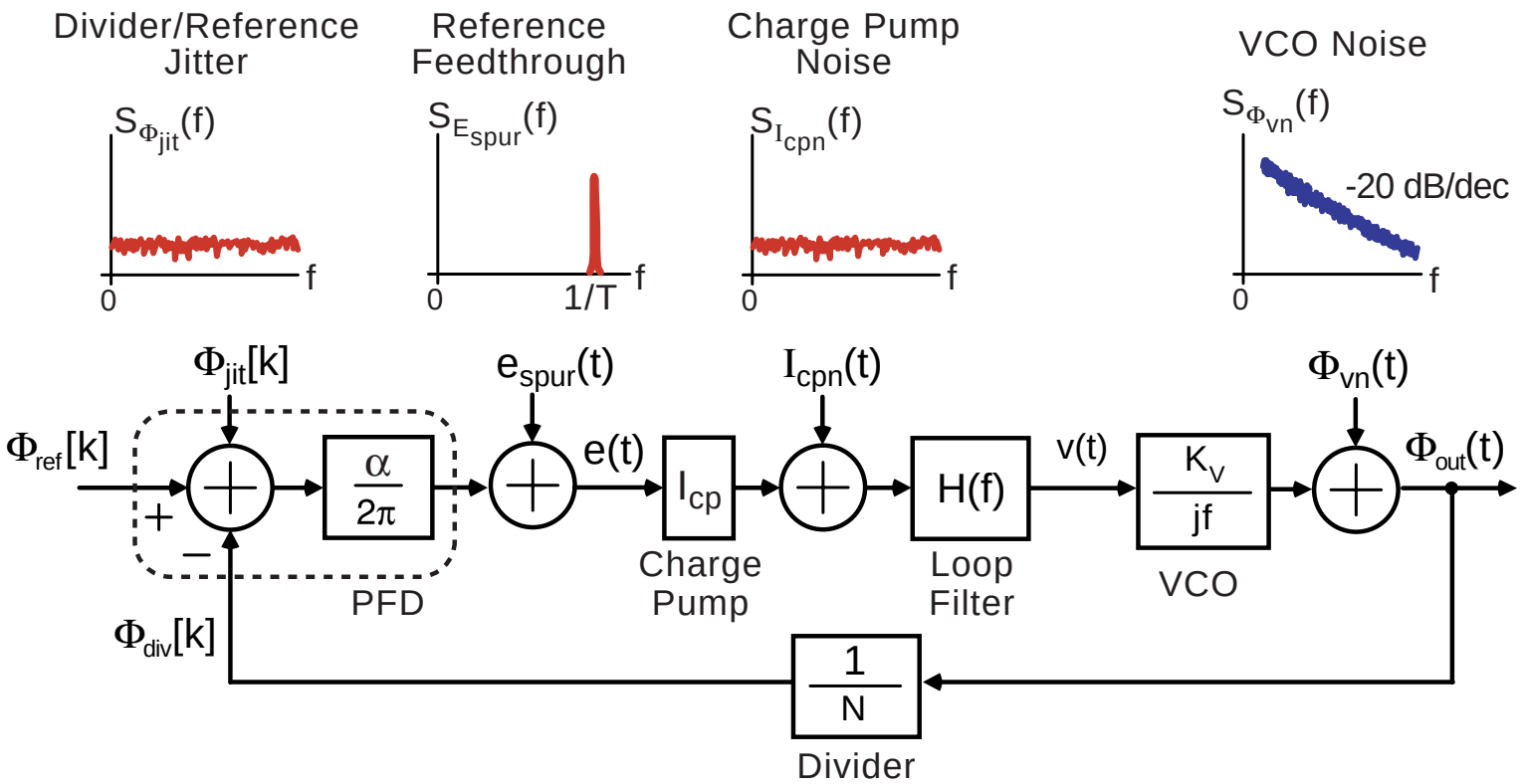
$$20\log\left(\frac{d_{spur}}{2f_{spur}}\right)\ \text{dBc}$$

Sources of Noise in Frequency Synthesizers



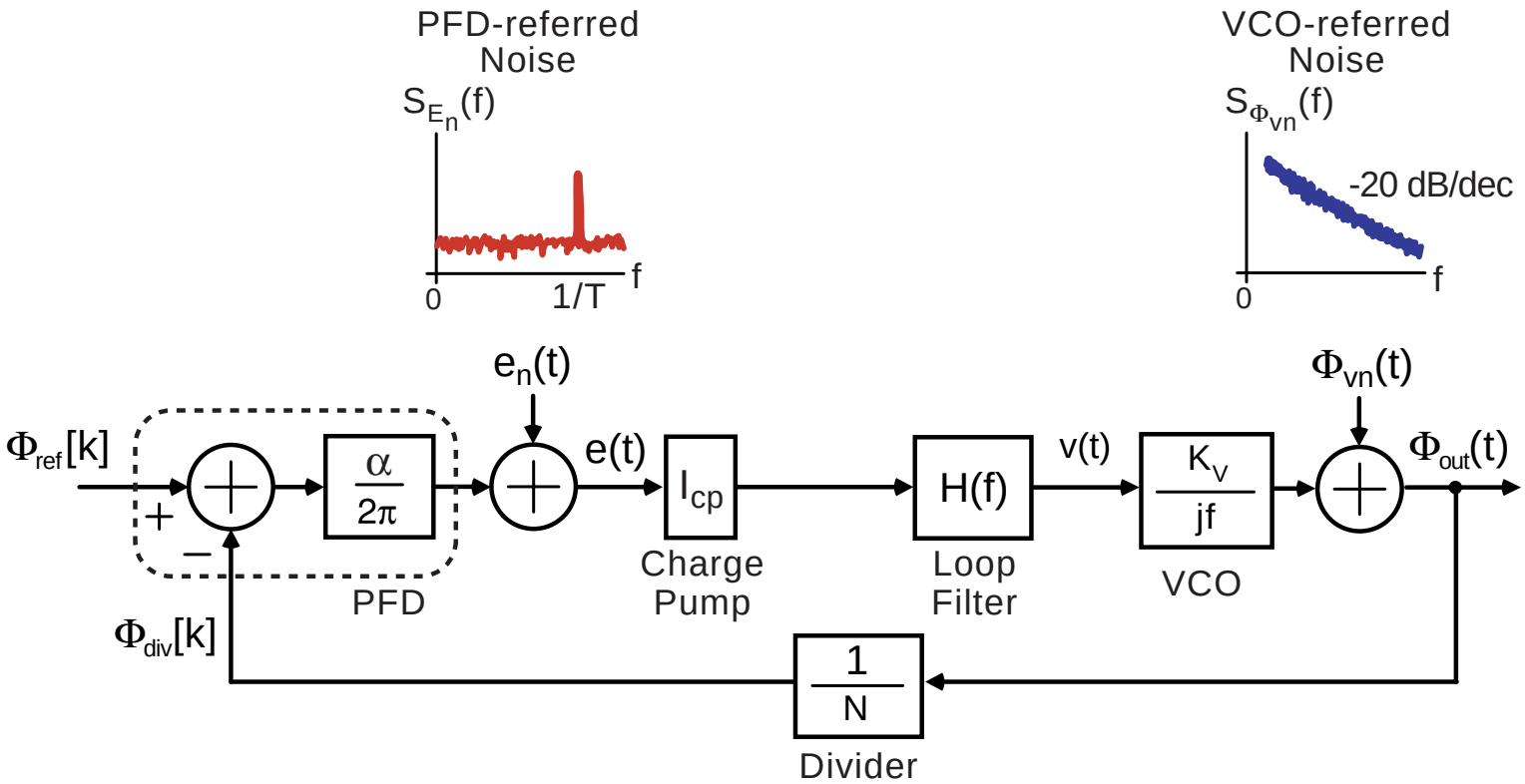
- **Extrinsic noise sources to VCO**
  - **Reference/divider jitter and reference feedthrough**
  - **Charge pump noise**

Modeling the Impact of Noise on Output Phase of PLL



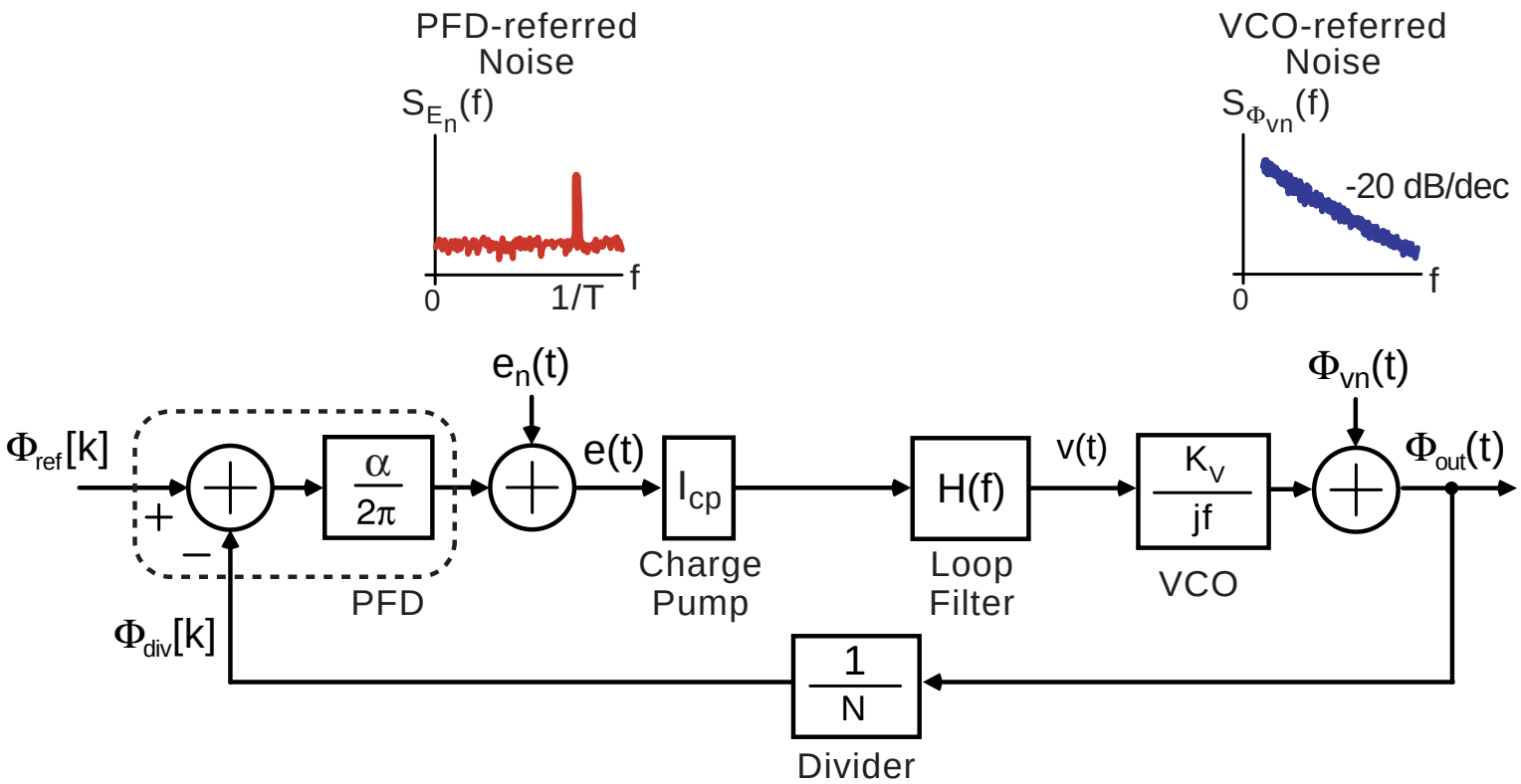
- **Determine impact on output phase by deriving transfer function from each noise source to PLL output phase**
  - **There are a lot of transfer functions to keep track of!**

Simplified Noise Model



- **Refer all PLL noise sources (other than the VCO) to the PFD output**
  - **PFD-referred noise corresponds to the sum of these noise sources referred to the PFD output**

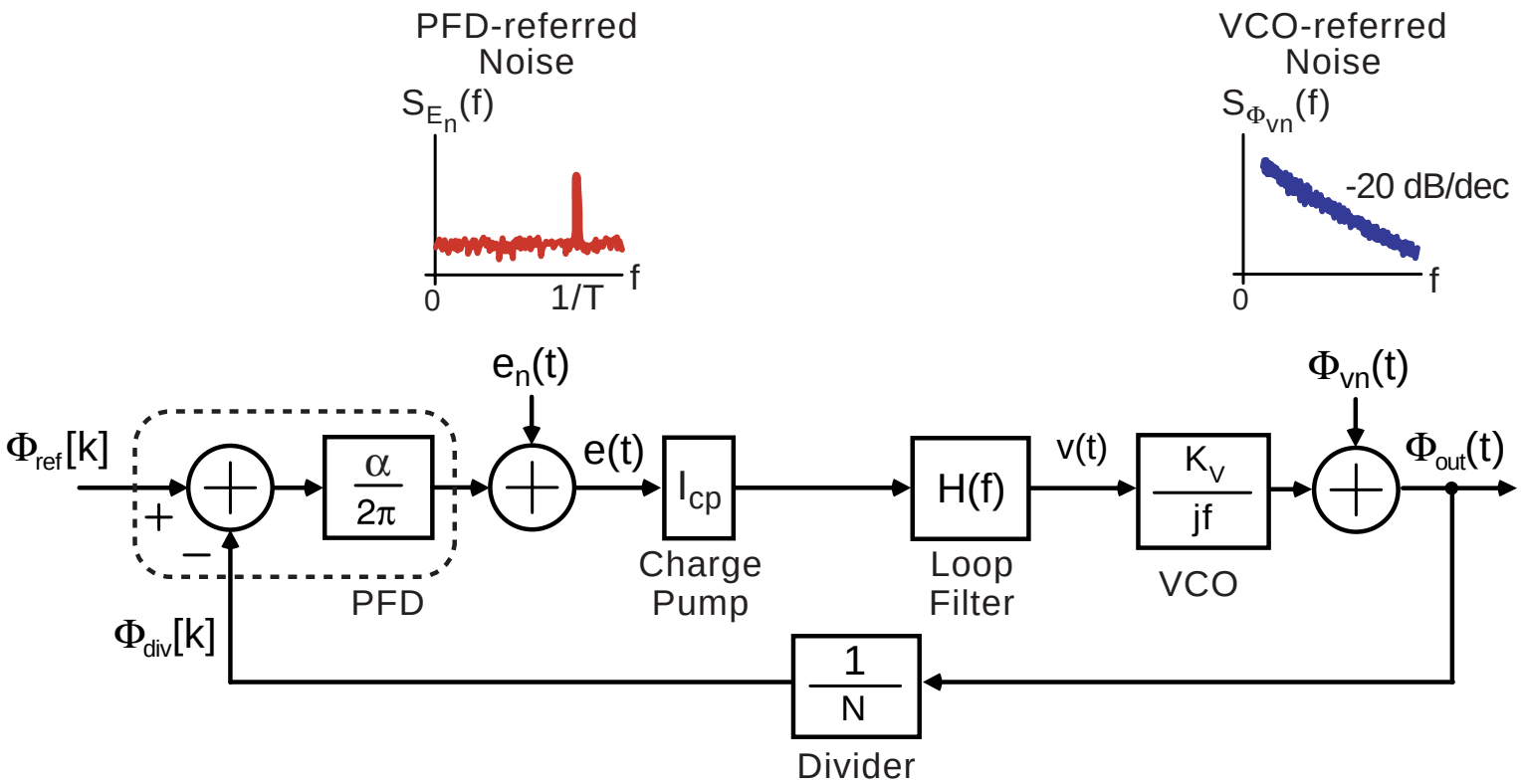
Impact of PFD-referred Noise on Synthesizer Output



- **Transfer function derived using Black's formula**

$$\frac{\Phi_{out}}{e_n} = \frac{I_{cp}H(f)K_v/(jf)}{1+\alpha/(2\pi)I_{cp}H(f)K_v/(jf)(1/N)}$$

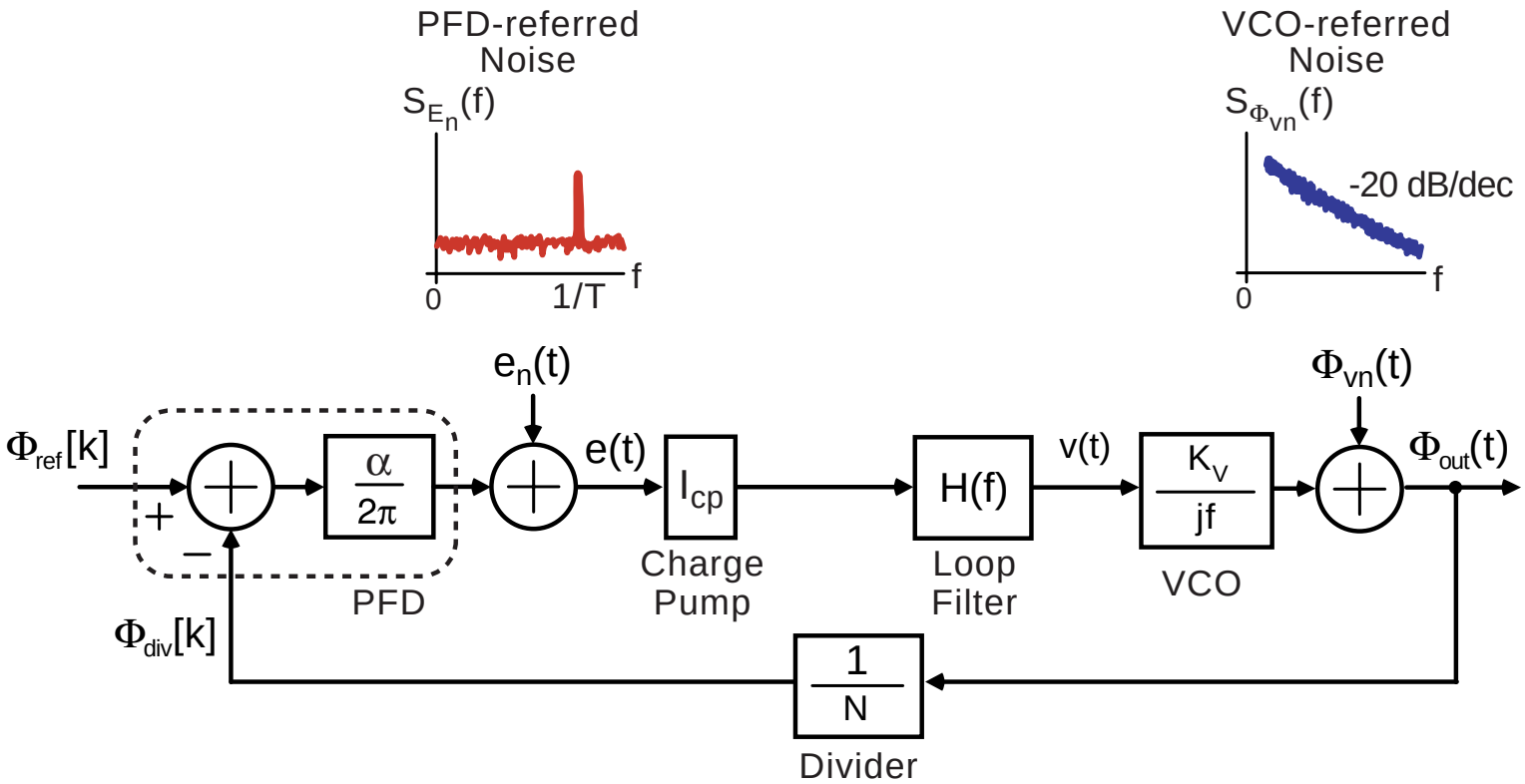
Impact of VCO-referred Noise on Synthesizer Output



- **Transfer function again derived from Black's formula**

$$\frac{\Phi_{out}}{\Phi_{vn}} = \frac{1}{1 + \alpha/(2\pi) I_{cp} H(f) K_v/(jf)(1/N)}$$

A Simpler Parameterization for PLL Transfer Functions



- **Define G(f) as**

$$G(f) = \frac{A(f)}{1 + A(f)}$$

Always has a gain of one at DC

  - **A(f) is the open loop transfer function of the PLL**

$$A(f) = \alpha/(2\pi) I_{cp} H(f) K_v/(jf)(1/N)$$

Parameterize Noise Transfer Functions in Terms of G(f)

- **PFD-referred noise**

$$\frac{\Phi_{out}}{e_n} = \frac{I_{cp}H(f)K_v/(jf)}{1+\alpha/(2\pi)I_{cp}H(f)K_v/(jf)(1/N)}$$

$$= \frac{2\pi}{\alpha}N\frac{\alpha/(2\pi)I_{cp}H(f)K_v/(jf)(1/N)}{1+\alpha/(2\pi)I_{cp}H(f)K_v/(jf)(1/N)}$$

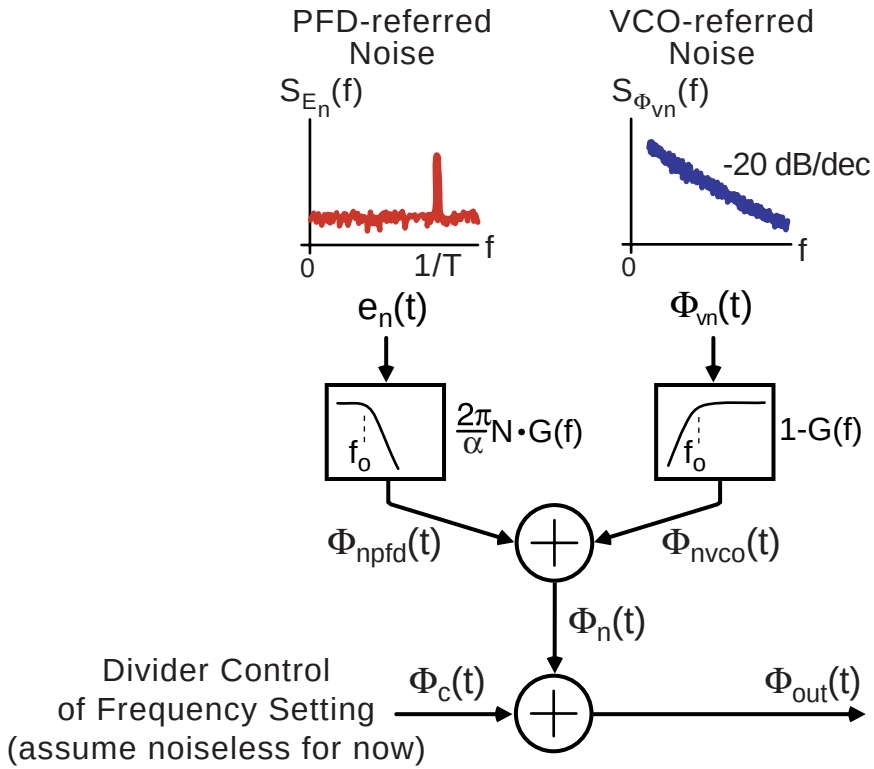
$$= \frac{2\pi}{\alpha}N\frac{A(f)}{1+A(f)} = \boxed{\frac{2\pi}{\alpha}NG(f)}$$

- **VCO-referred noise**

$$\frac{\Phi_{out}}{\Phi_{vn}} = \frac{1}{1+\alpha/(2\pi)I_{cp}H(f)K_v/(jf)(1/N)}$$

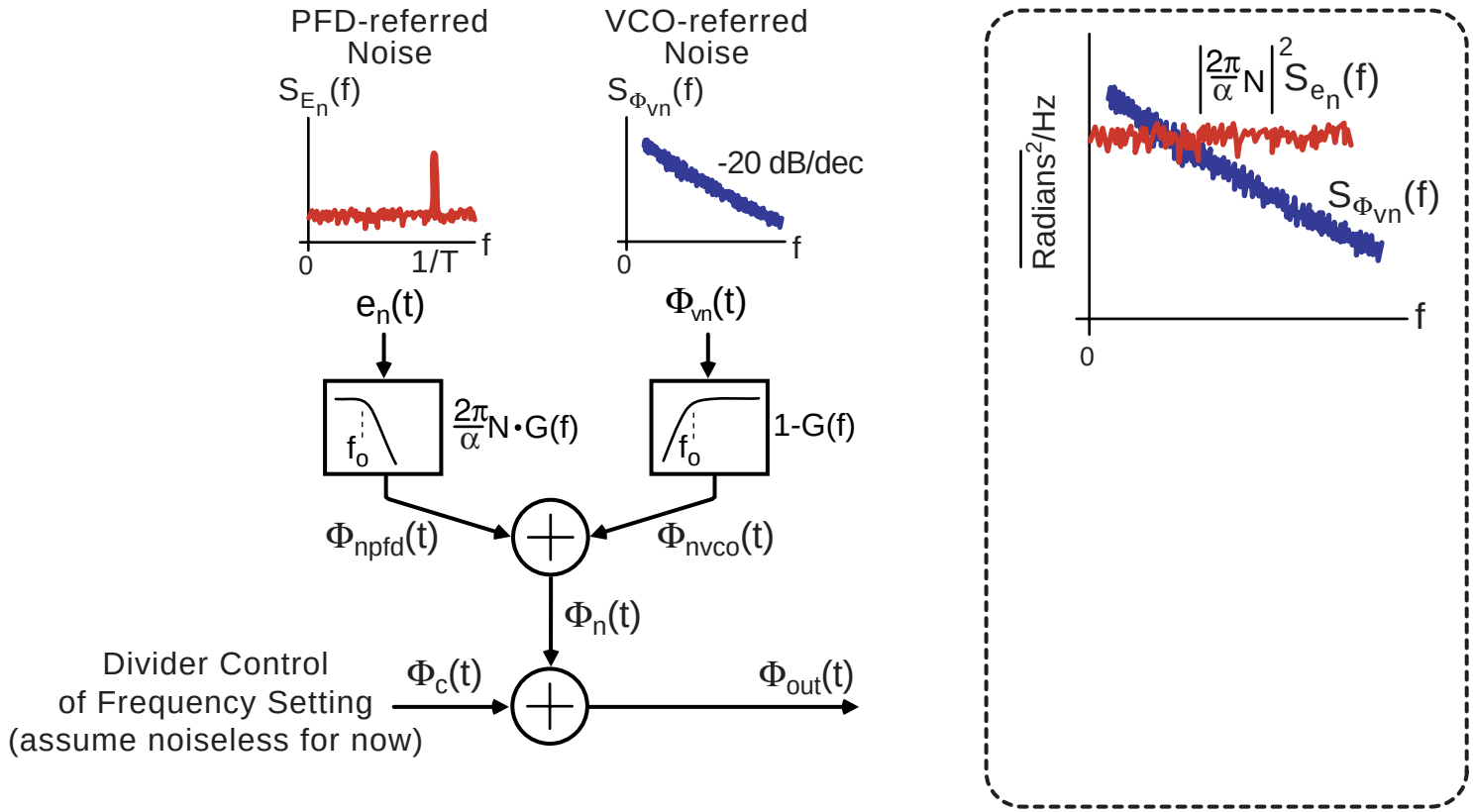
$$= \frac{1}{1+A(f)} = 1-\frac{A(f)}{1+A(f)} = \boxed{1-G(f)}$$

Parameterized PLL Noise Model



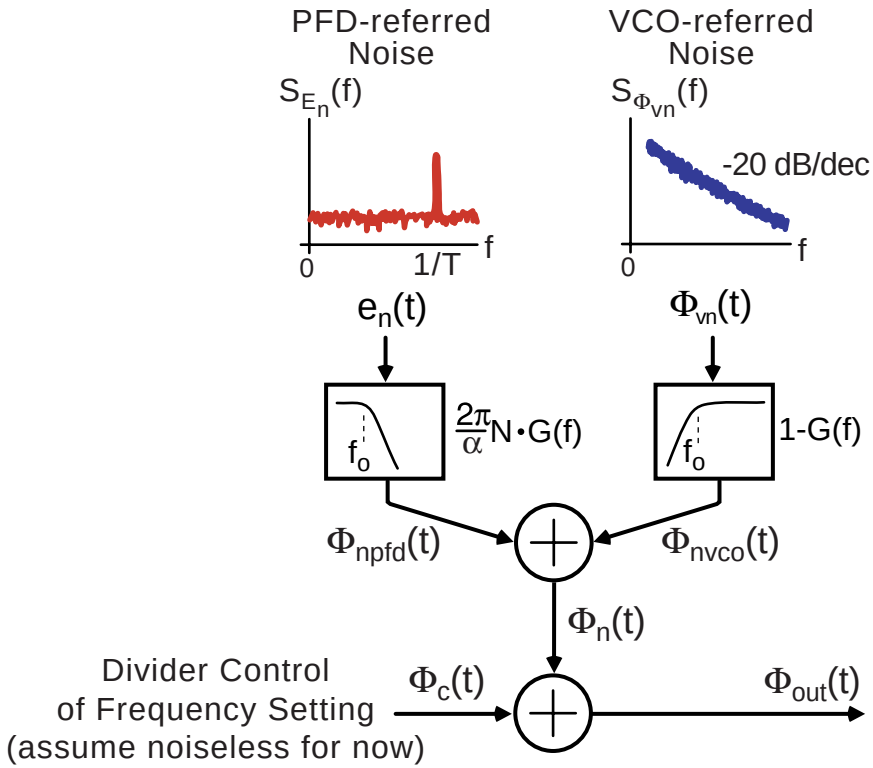
- **PFD-referred noise is lowpass filtered**
- **VCO-referred noise is highpass filtered**
- **Both filters have the same transition frequency values**
  - **Defined as f_o**

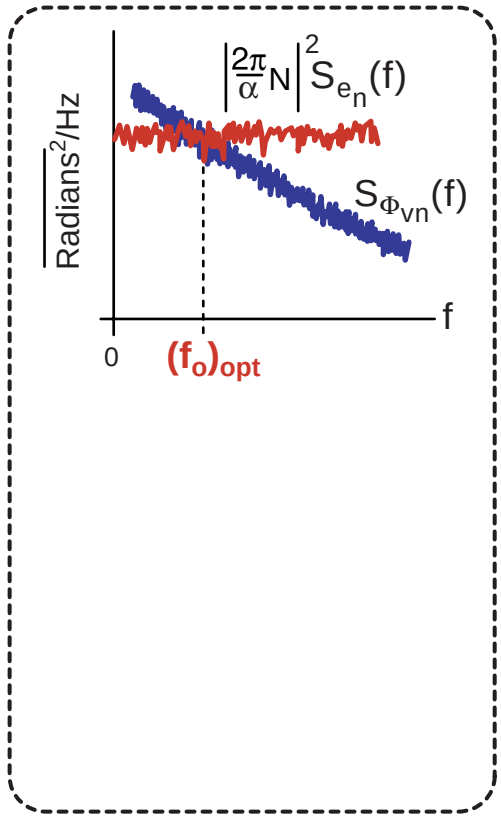
Impact of PLL Parameters on Noise Scaling



- **PFD-referred noise is scaled by square of divide value and inverse of PFD gain**
  - **High divide values lead to large multiplication of this noise**
- **VCO-referred noise is not scaled (only filtered)**

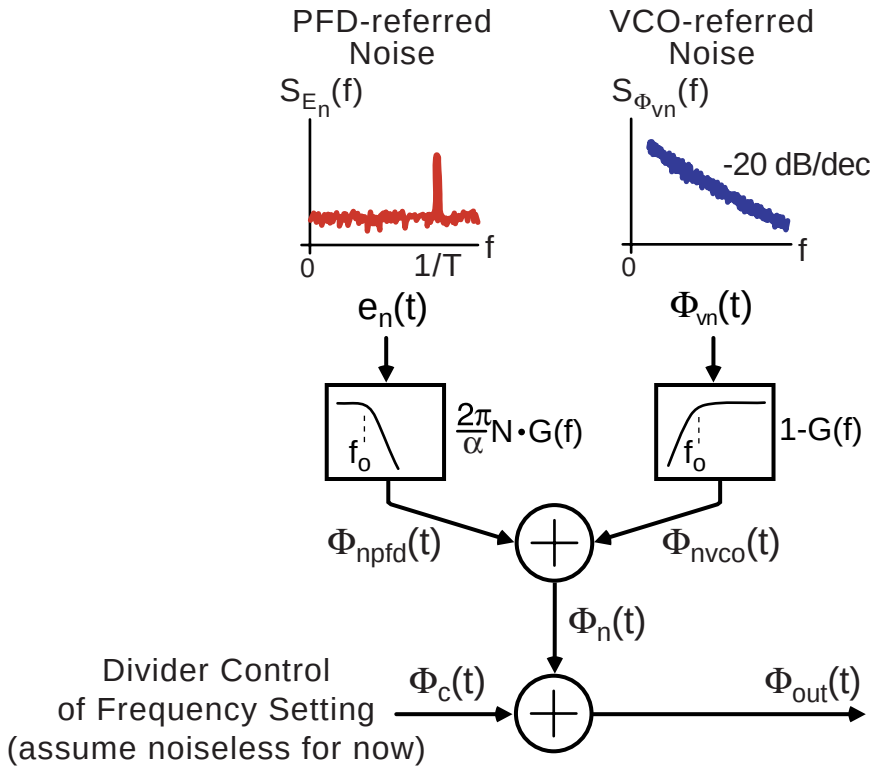
Optimal Bandwidth Setting for Minimum Noise

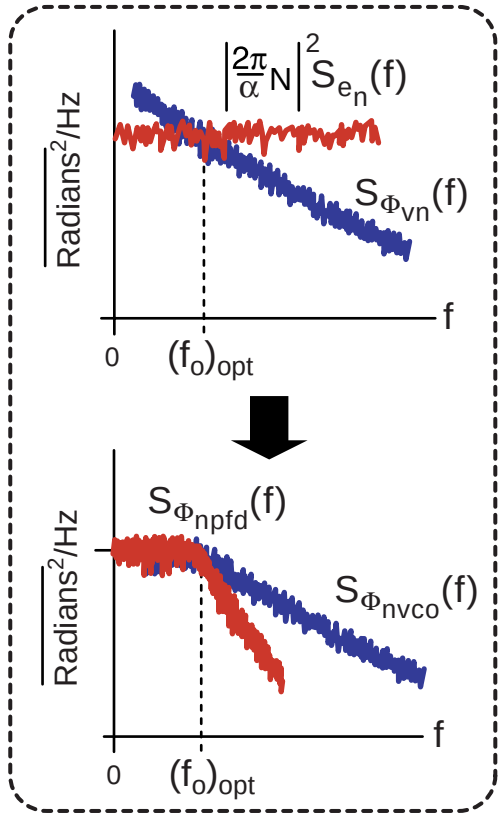




- **Optimal bandwidth is where scaled noise sources meet**
  - **Higher bandwidth will pass more PFD-referred noise**
  - **Lower bandwidth will pass more VCO-referred noise**

Resulting Output Noise with Optimal Bandwidth





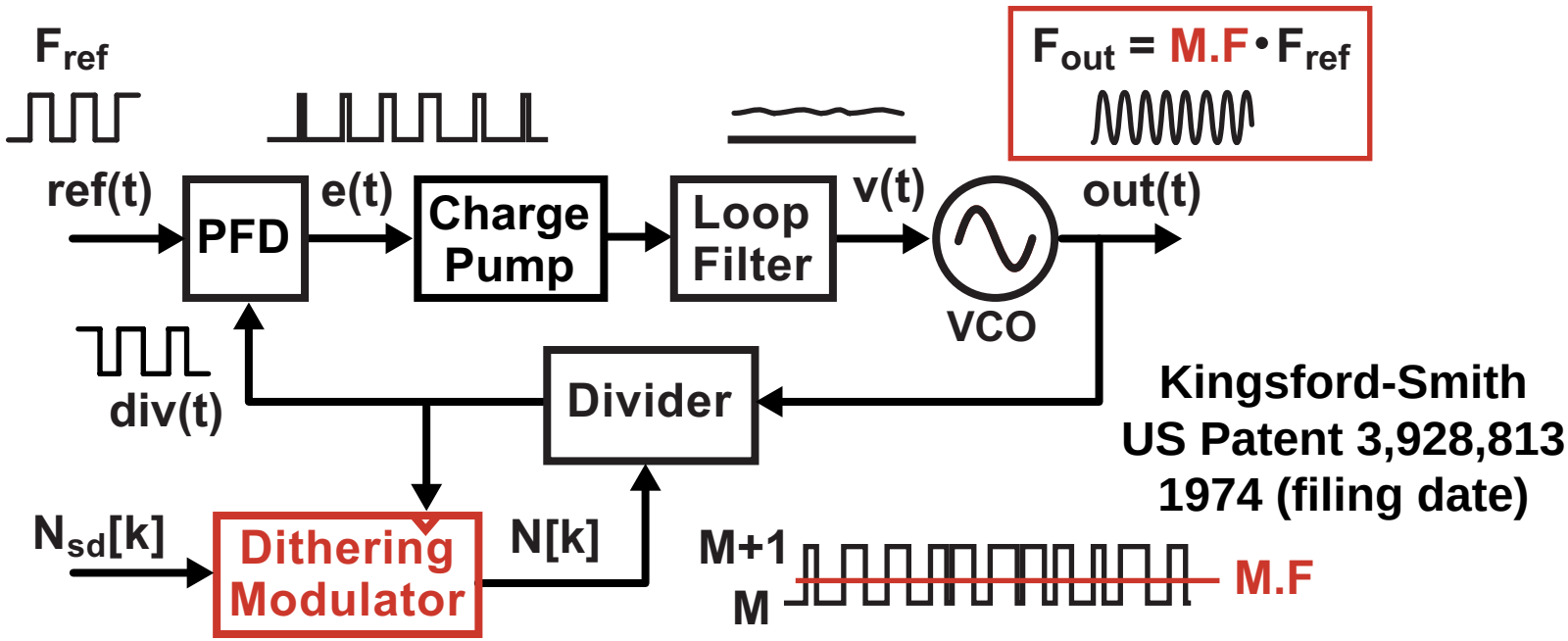
- **PFD-referred noise dominates at low frequencies**
  - **Corresponds to close-in phase noise of synthesizer**
- **VCO-referred noise dominates at high frequencies**
  - **Corresponds to far-away phase noise of synthesizer**

Summary of Noise Analysis of Integer-N Synthesizers

- **Key PLL noise sources are**
  - **VCO noise**
  - **PFD-referred noise**
    - Charge pump noise, reference noise, etc.

- **Setting of PLL bandwidth has strong impact on noise**
  - **High PLL bandwidth suppresses VCO noise**
  - **Low PLL bandwidth suppresses PFD-referred noise**

Fractional-N Frequency Synthesis



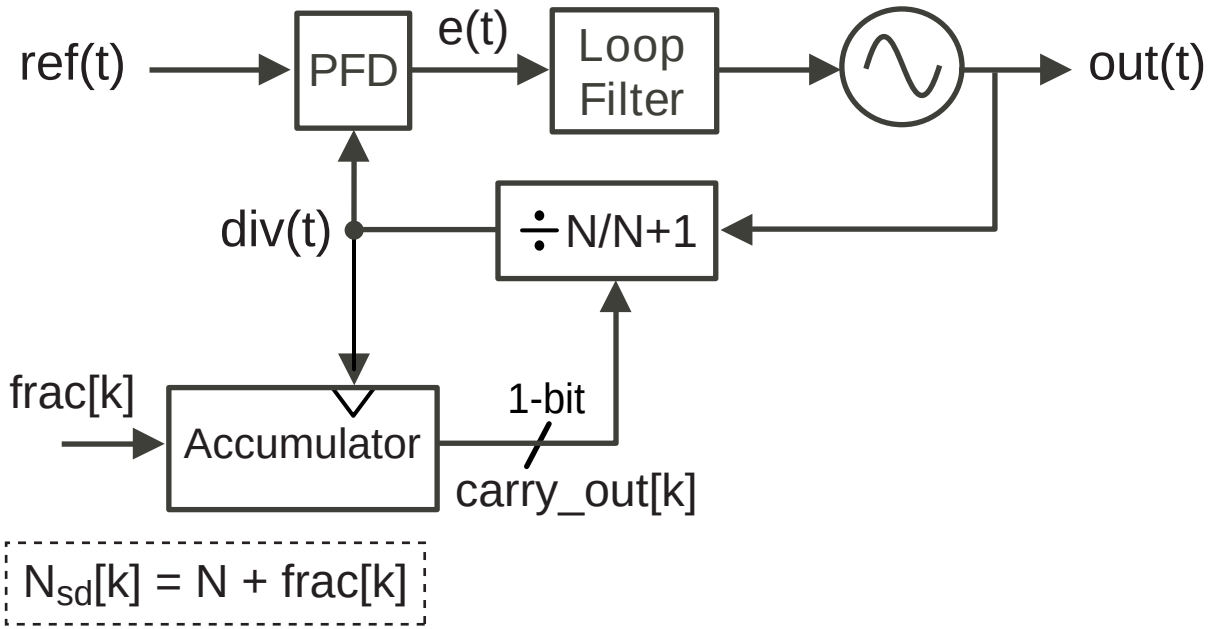
- Divide value is dithered between integer values
- Fractional divide values can be realized!

Very high frequency resolution

Outline of Fractional-N Synthesizers

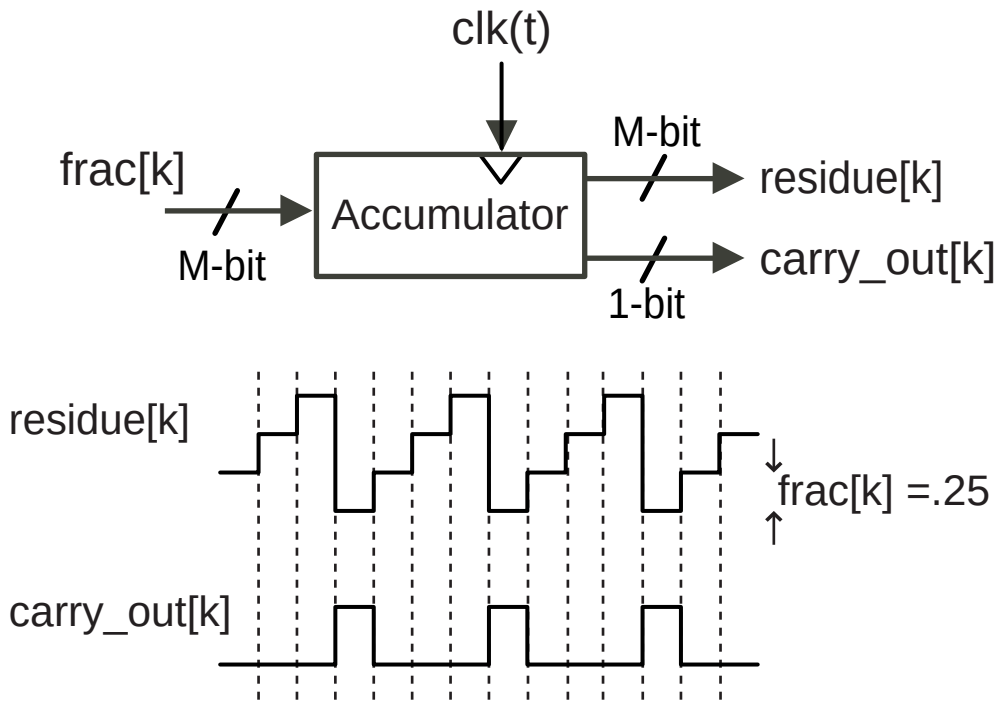
- **Traditional Approach**
- **Sigma-Delta Concepts**
- **Synthesizer Noise Analysis**

Classical Fractional-N Synthesizer Architecture



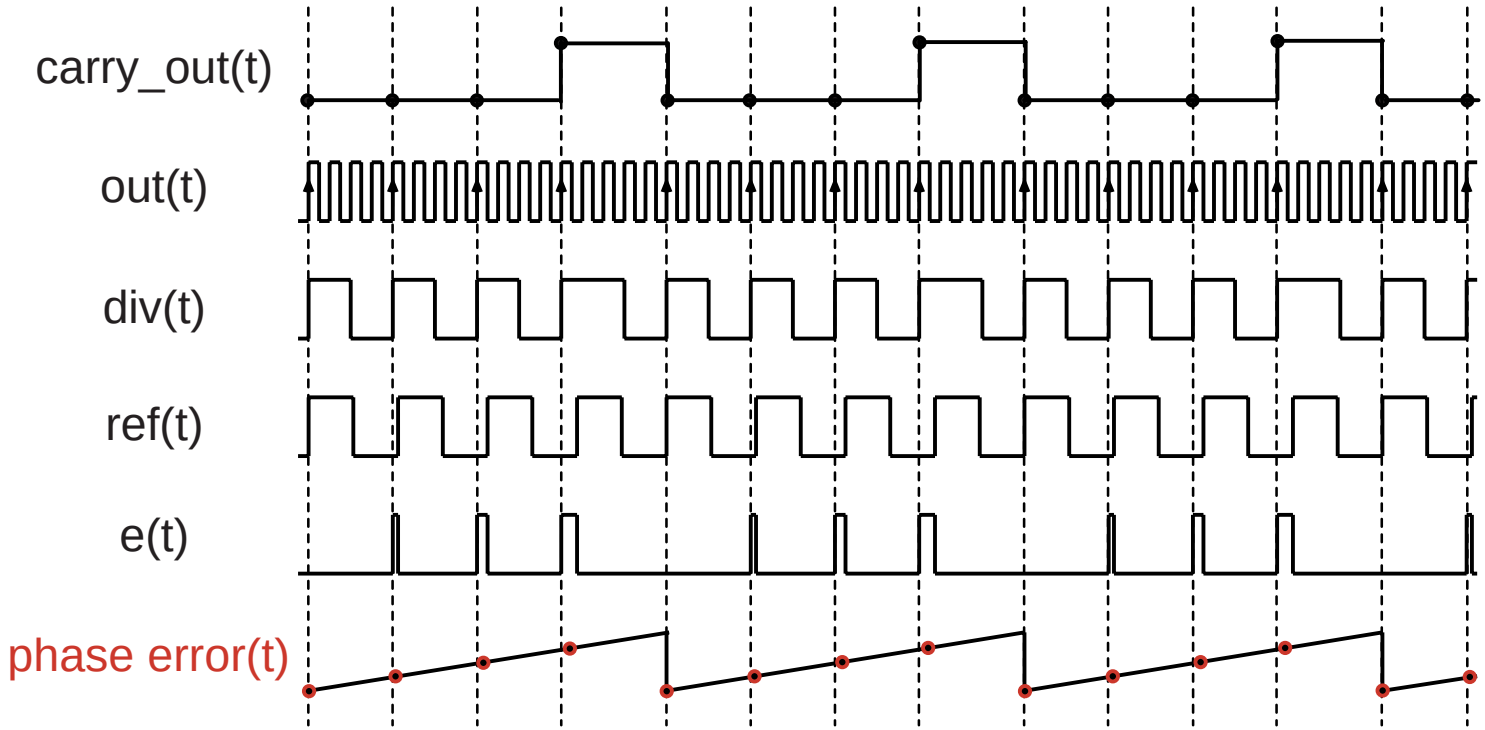
- **Use an accumulator to perform dithering operation**
  - **Fractional input value fed into accumulator**
  - **Carry out bit of accumulator fed into divider**

Accumulator Operation



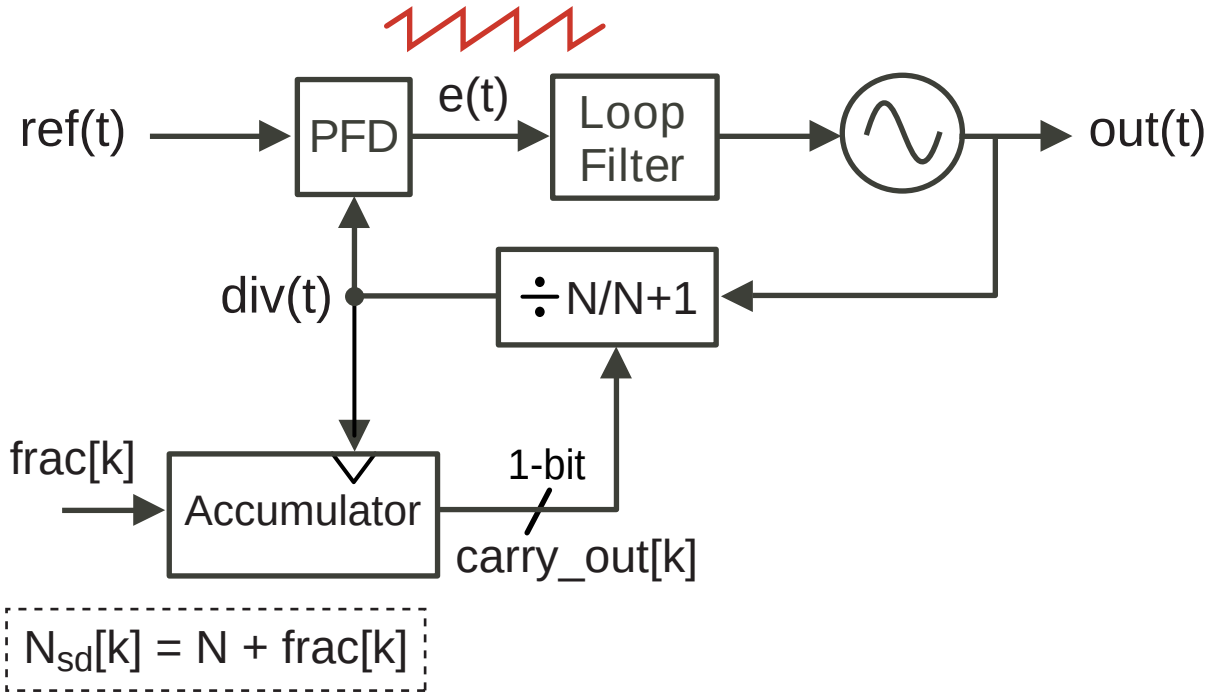
- **Carry out bit is asserted when accumulator residue reaches or surpasses its full scale value**
  - **Accumulator residue increments by input fractional value each clock cycle**

Fractional-N Synthesizer Signals with N = 4.25



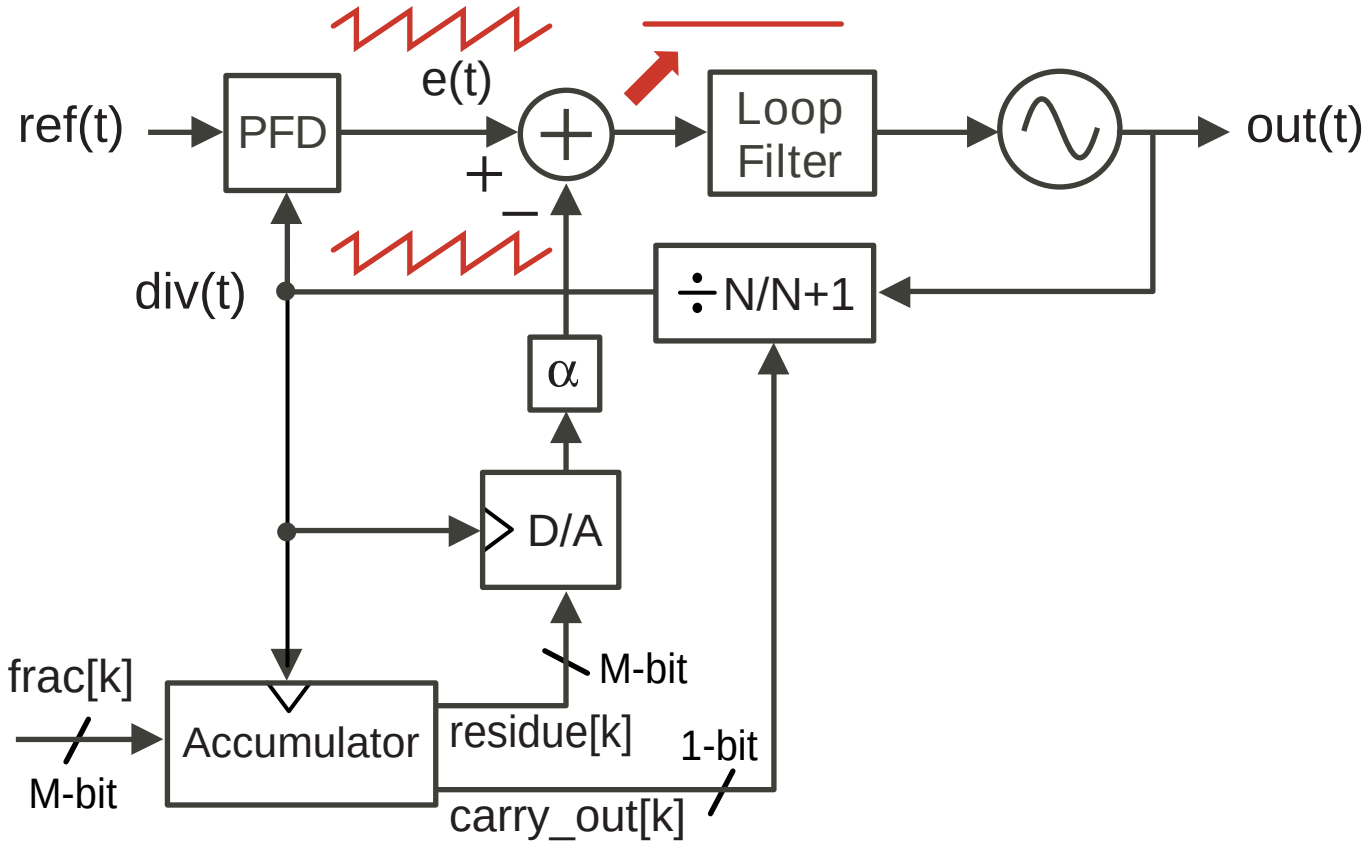
- **Divide value set at N = 4 most of the time**
  - **Resulting frequency offset causes phase error to accumulate**
  - **Reset phase error by "swallowing" a VCO cycle**
    - Achieved by dividing by 5 every 4 reference cycles

The Issue of Spurious Tones



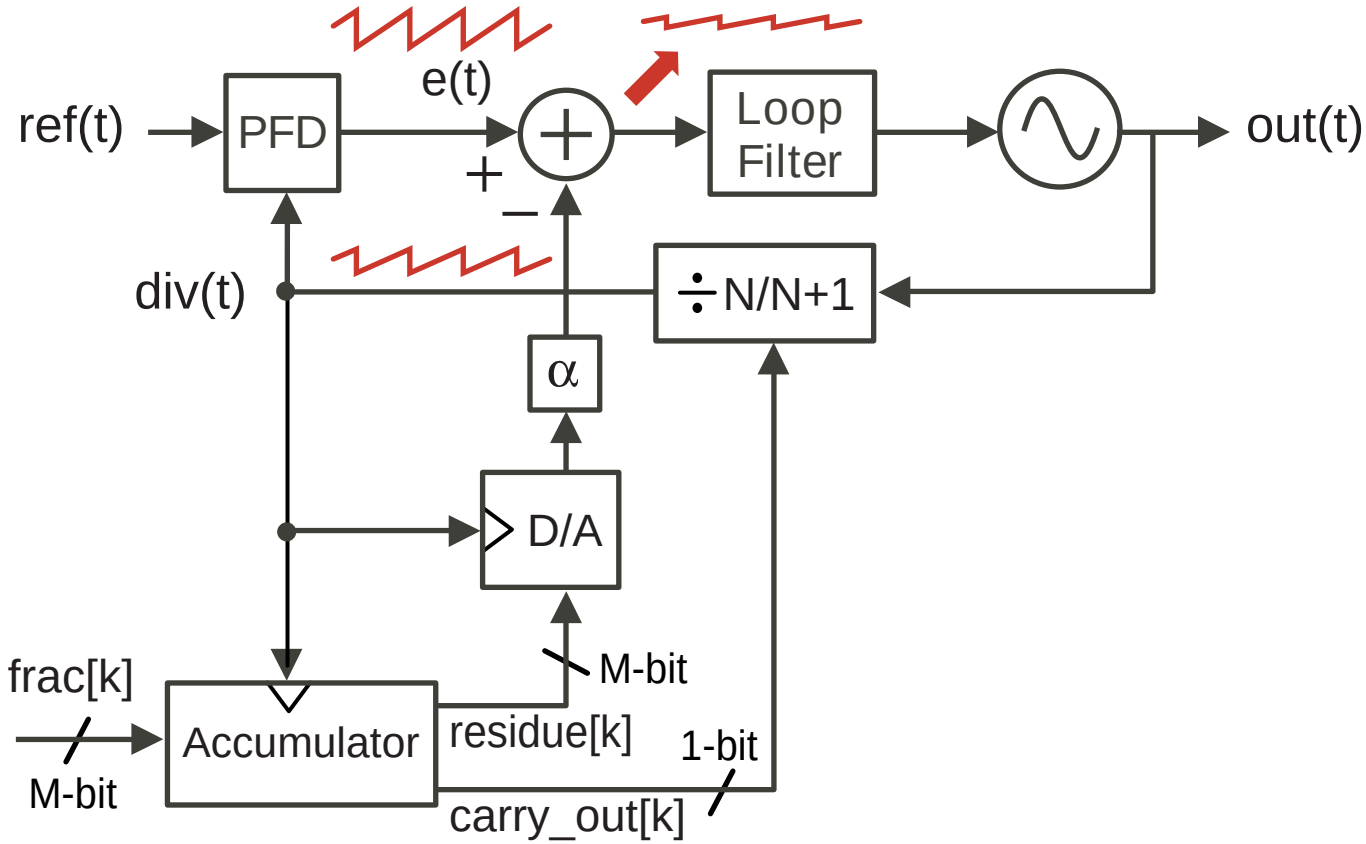
- **PFD error is periodic**
  - **Note that actual PFD waveform is series of pulses – the sawtooth waveform represents pulse width values over time**
- **Periodic error signal creates spurious tones in synthesizer output**
  - **Ruins noise performance of synthesizer**

The Phase Interpolation Technique



- **Phase error due to fractional technique is predicted by the instantaneous residue of the accumulator**
  - **Cancel out phase error based on accumulator residue**

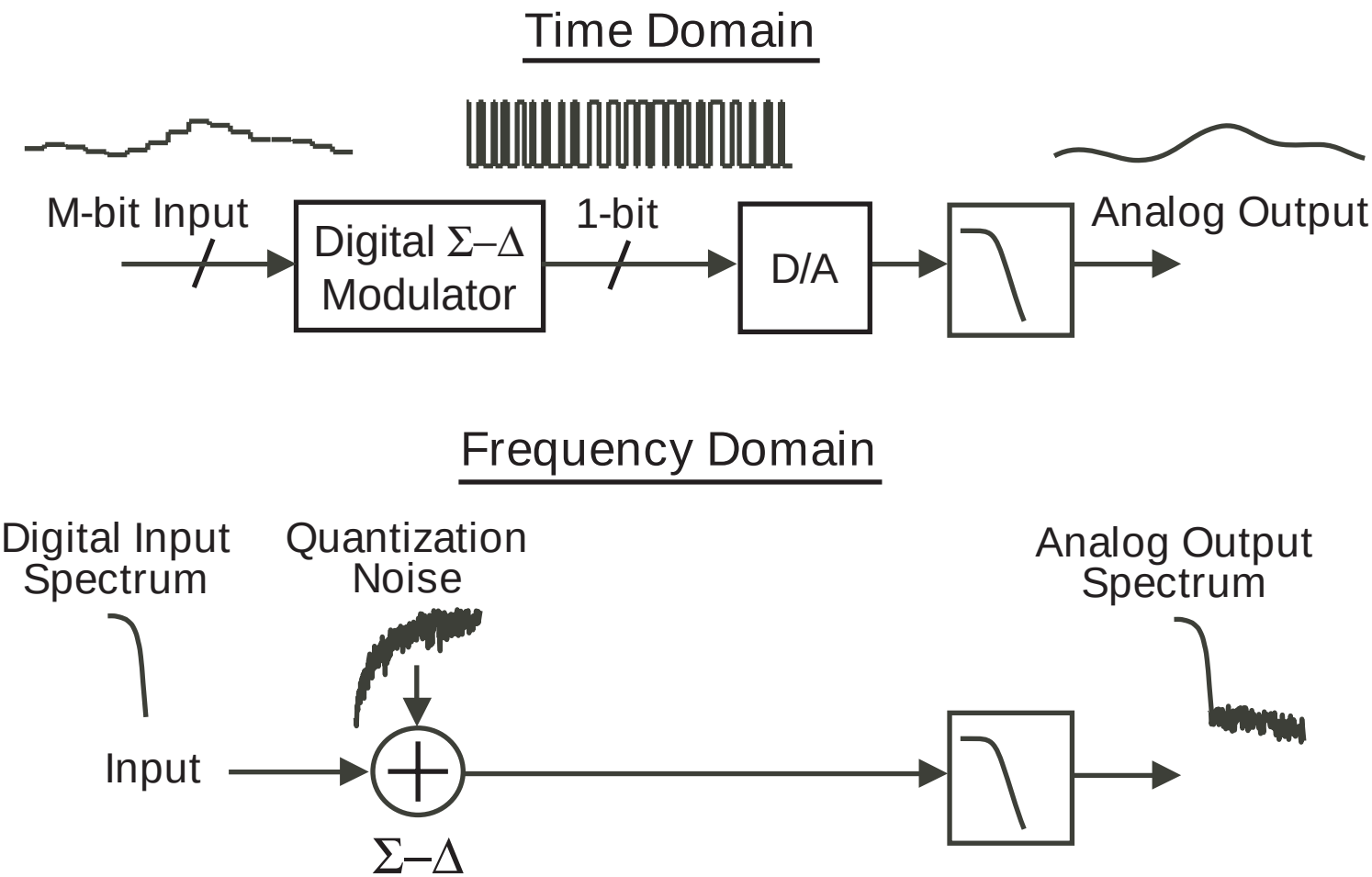
The Problem With Phase Interpolation



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

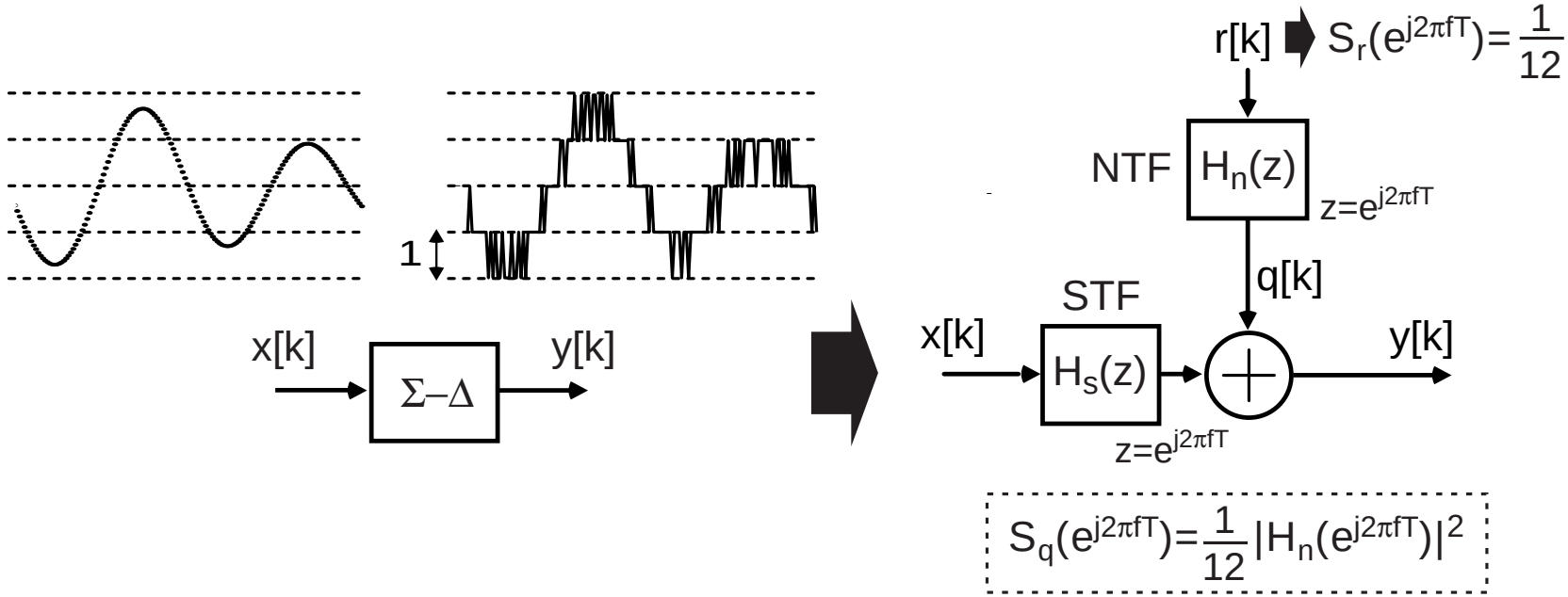
Is There a Better Way?

A Better Dithering Method: Sigma-Delta Modulation



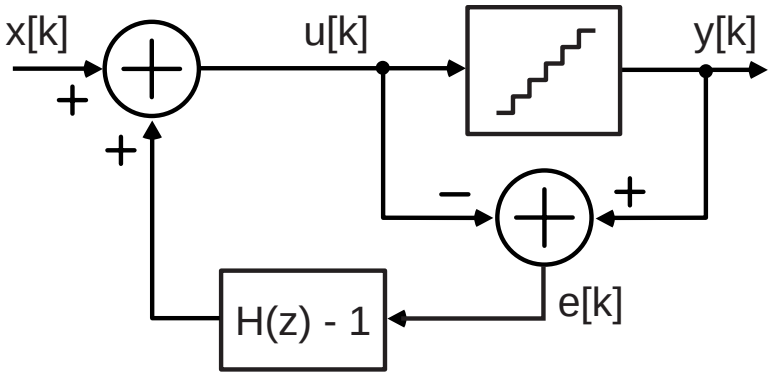
- **Sigma-Delta dithers in a manner such that resulting quantization noise is "shaped" to high frequencies**

Linearized Model of Sigma-Delta Modulator



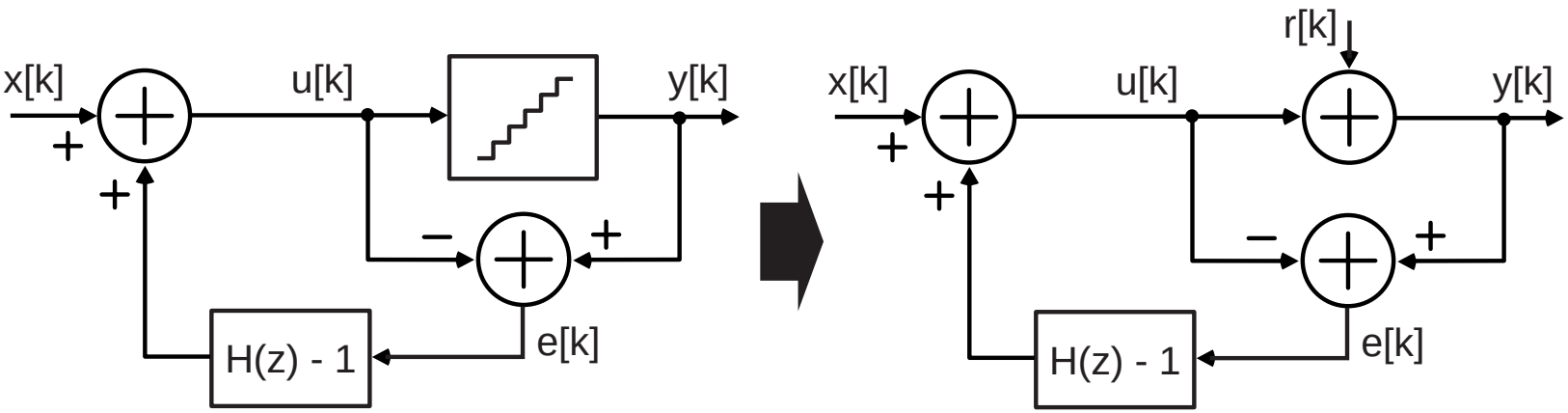
- **Composed of two transfer functions relating input and noise to output**
  - **Signal transfer function (STF)**
    - Filters input (generally undesirable)
  - **Noise transfer function (NTF)**
    - Filters (i.e., shapes) noise that is assumed to be white

Example: Cutler Sigma-Delta Topology



- **Output is quantized in a multi-level fashion**
- **Error signal, e[k], represents the quantization error**
- **Filtered version of quantization error is fed back to input**
  - **H(z) is typically a highpass filter whose first tap value is 1**
    - i.e., $H(z) = 1 + a_1 z^{-1} + a_2 z^{-2} \cdots$
  - **H(z) – 1 therefore has a first tap value of 0**
    - Feedback needs to have delay to be realizable

Linearized Model of Cutler Topology

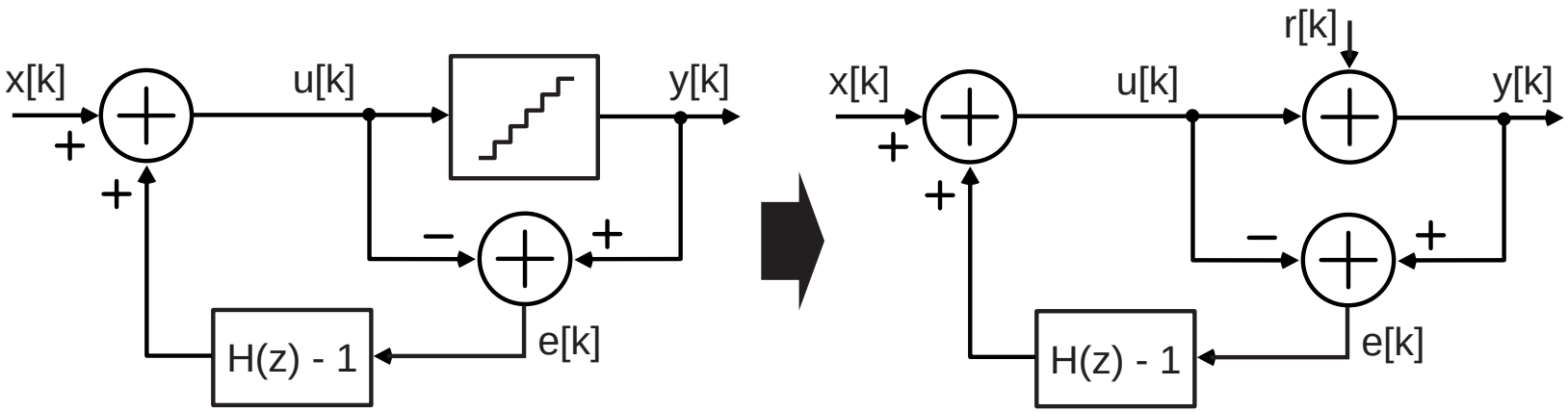


- **Represent quantizer block as a summing junction in which r[k] represents quantization error**
  - **Note:**

$$e[k] = y[k] - u[k] = (u[k] + r[k]) - u[k] = r[k]$$

- **It is assumed that r[k] has statistics similar to white noise**
  - **This is a key assumption for modeling – often not true!**

Calculation of Signal and Noise Transfer Functions



- **Calculate using Z-transform of signals in linearized model**

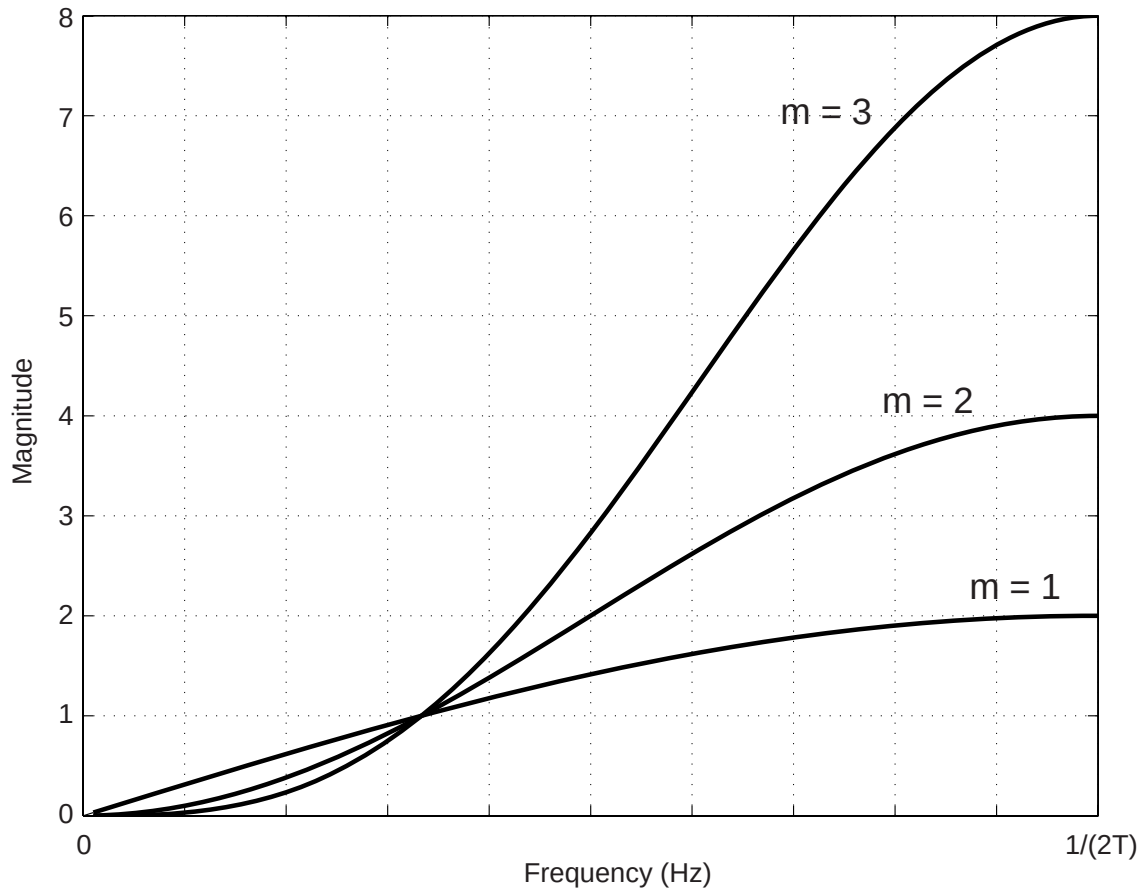
$$\begin{aligned} Y(z) &= U(z) + R(z) \\ &= X(z) + (H(z) - 1)E(z) + R(z) \\ &= X(z) + (H(z) - 1)R(z) + R(z) \\ &= X(z) + H(z)R(z) \end{aligned}$$

  - **NTF: $H_n(z) = H(z)$**
  - **STF: $H_s(z) = 1$**

A Common Choice for H(z)

$$H(z) = (1 - z^{-1})^m$$

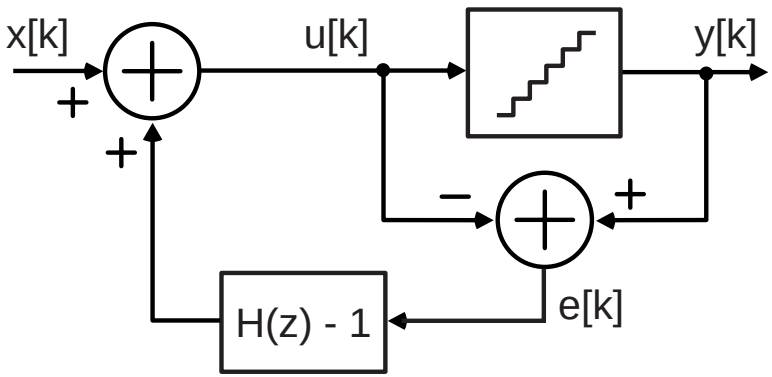
$$\Rightarrow \; |H(e^{j2\pi fT})| = |(1 - e^{-j2\pi fT})^m|$$



Example: First Order Sigma-Delta Modulator

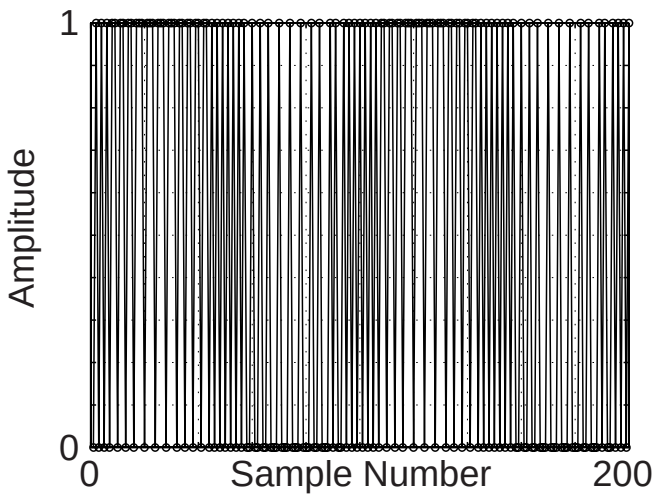
- **Choose NTF to be**

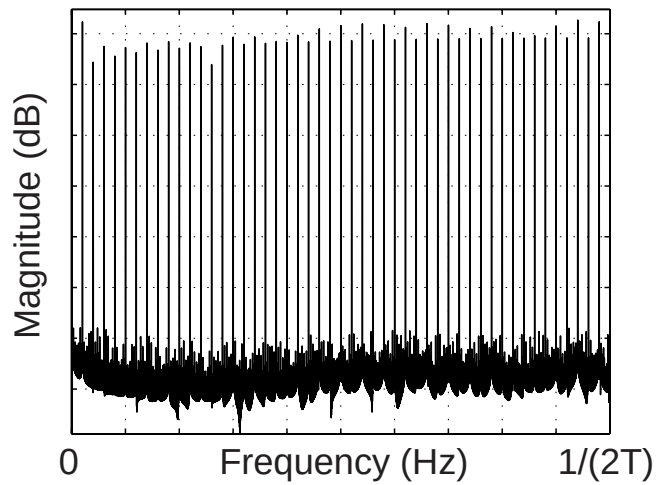
$$H_n(z) = H(z) = 1 - z^{-1}$$



- **Plot of output in time and frequency domains with input of**

$$x[k] = 0.5 + 0.25 \sin\left(\frac{2\pi}{100}k\right)$$

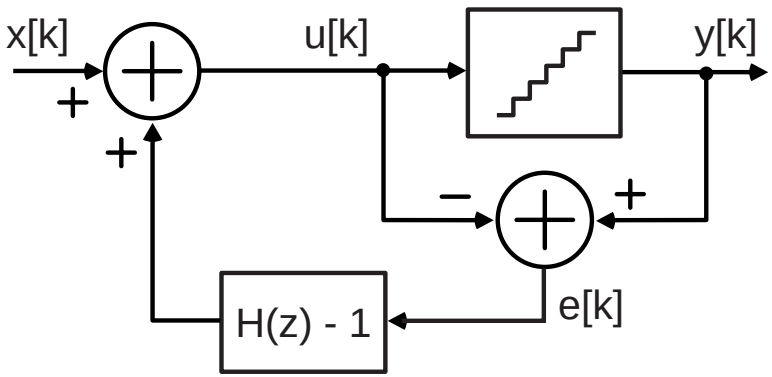




Example: Second Order Sigma-Delta Modulator

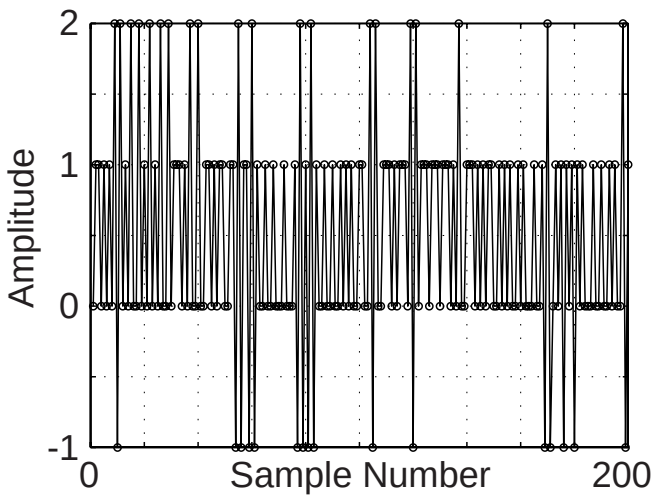
- **Choose NTF to be**

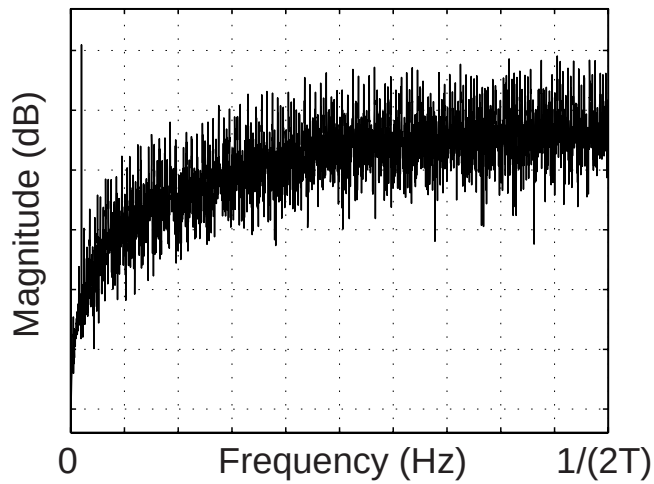
$$H_n(z) = H(z) = (1 - z^{-1})^2$$



- **Plot of output in time and frequency domains with input of**

$$x[k] = 0.5 + 0.25 \sin\left(\frac{2\pi}{100}k\right)$$

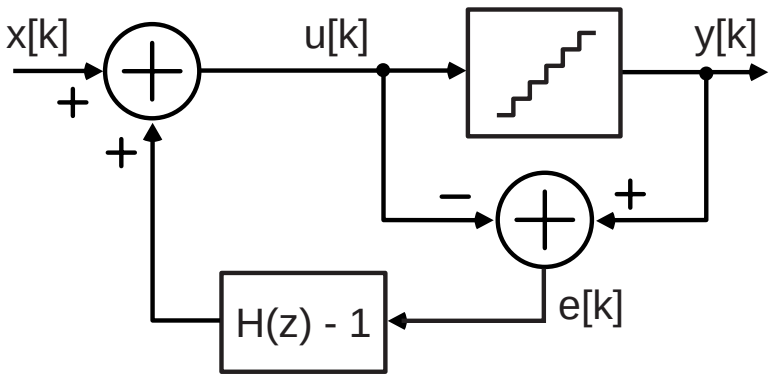




Example: Third Order Sigma-Delta Modulator

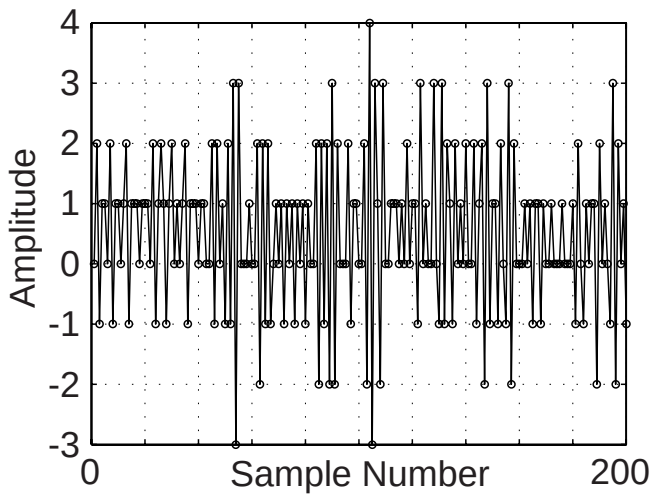
- **Choose NTF to be**

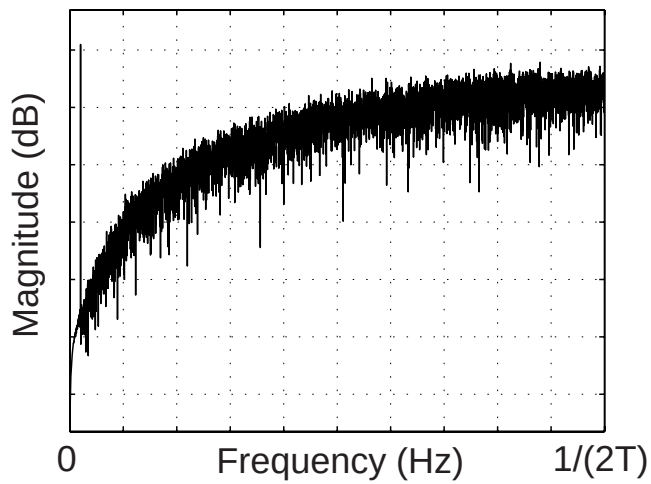
$$H_n(z) = H(z) = (1 - z^{-1})^3$$



- **Plot of output in time and frequency domains with input of**

$$x[k] = 0.5 + 0.25 \sin\left(\frac{2\pi}{100}k\right)$$



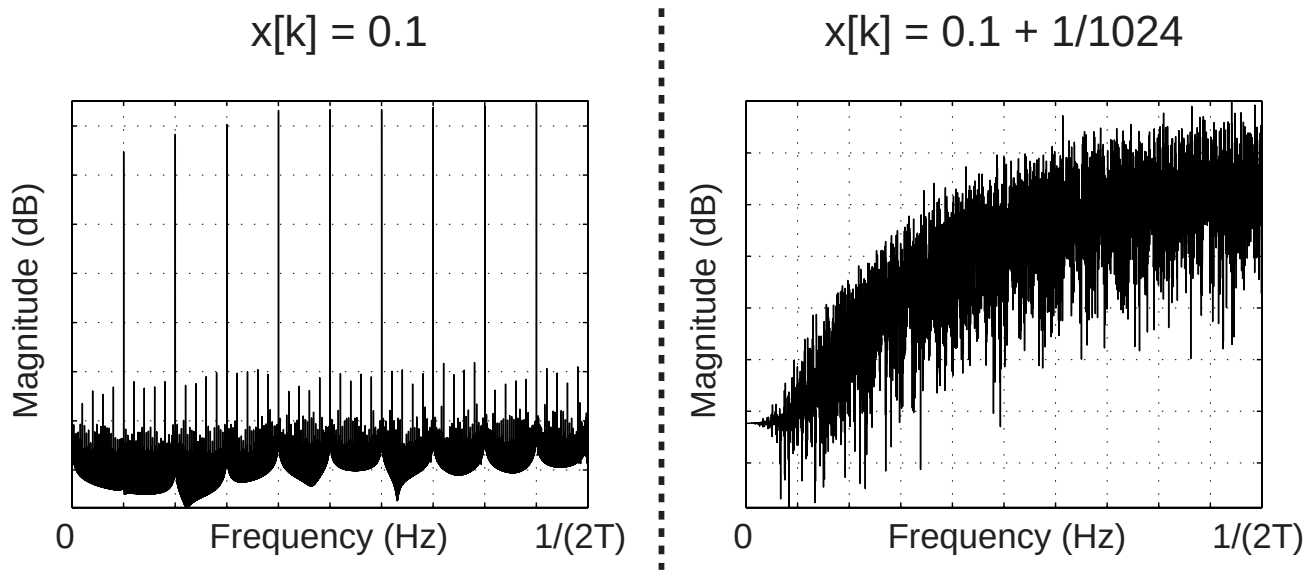


Observations

- **Low order Sigma-Delta modulators do not appear to produce "shaped" noise very well**
  - **Reason: low order feedback does not properly "scramble" relationship between input and quantization noise**
    - Quantization noise, r[k], fails to be white
- **Higher order Sigma-Delta modulators provide much better noise shaping with fewer spurs**
  - **Reason: higher order feedback filter provides a much more complex interaction between input and quantization noise**

Warning: Higher Order Modulators May Still Have Tones

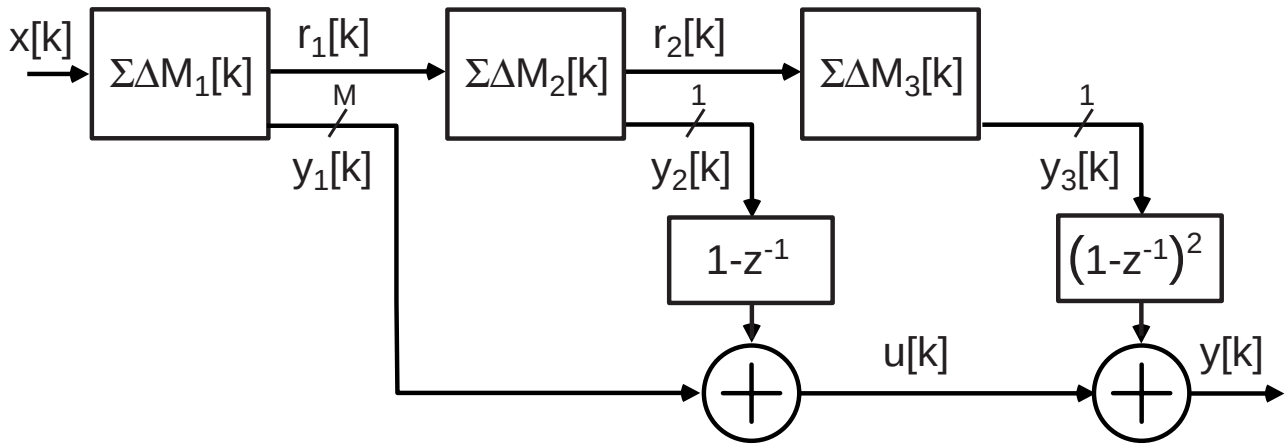
- Quantization noise, r[k], is best whitened when a "sufficiently exciting" input is applied to the modulator
  - Varying input and high order helps to "scramble" interaction between input and quantization noise
- Worst input for tone generation are DC signals that are rational with a low valued denominator
  - Examples (third order modulator with no dithering):



Fractional Spurs Can Be Theoretically Eliminated

- **See:**
  - **M. Kozak, I. Kale, “Rigorous Analysis of Delta-Sigma Modulators for Fractional-N PLL Frequency Synthesis”, *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 51, no. 6, pp. 1148-1162, June 2004.**
  - **S. Pamarti, I. Galton, "LSB Dithering in MASH Delta–Sigma D/A Converters", *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 4, pp. 779 – 790, April 2007.**

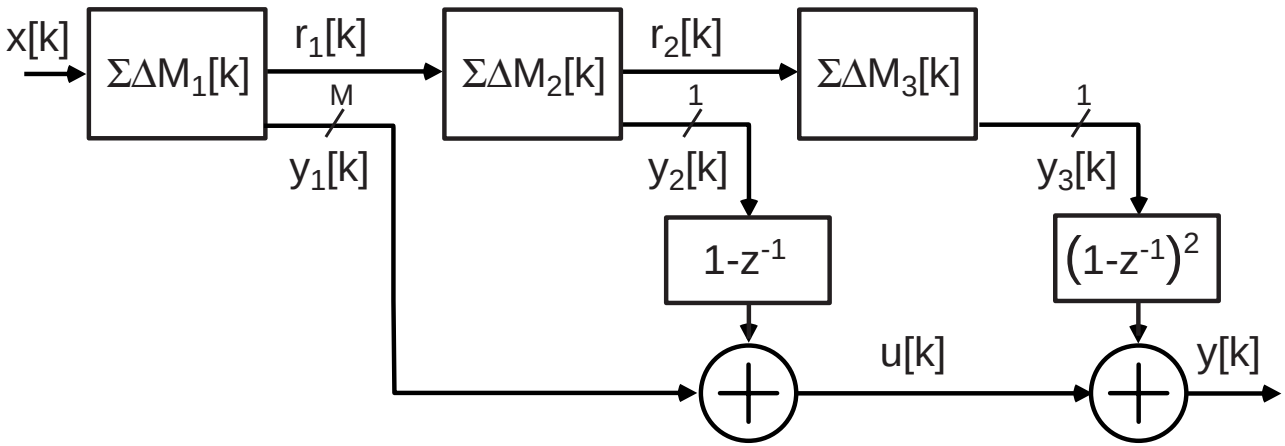
MASH topology



- **Cascade first order sections**
- **Combine their outputs after they have passed through digital differentiators**

- **Advantage over single loop approach**
  - **Allows pipelining to be applied to implementation**
    - High speed or low power applications benefit

Calculation of STF and NTF for MASH topology (Step 1)



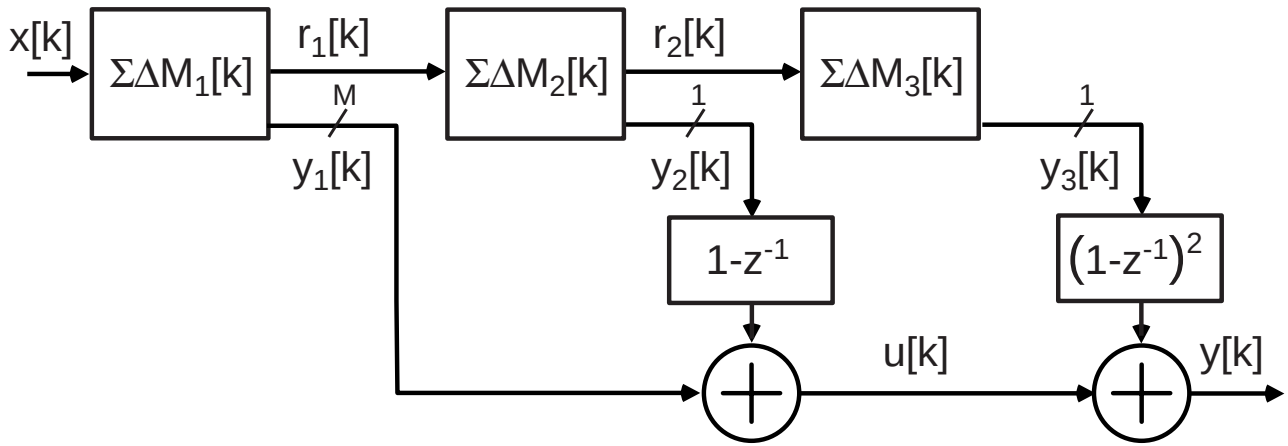
- **Individual output signals of each first order modulator**

$$
\begin{aligned}
y_1(z) &= x(z) - (1-z^{-1})r_1(z) \\
y_2(z) &= r_1(z) - (1-z^{-1})r_2(z) \\
y_3(z) &= r_2(z) - (1-z^{-1})r_3(z)
\end{aligned}
$$

- **Addition of filtered outputs**

$$
\begin{aligned}
& y_1(z) \\
+ \quad & (1-z^{-1})y_2(z) \\
+ \quad & (1-z^{-1})^2 y_2(z) \\
\hline
= \quad & x(z) - (1-z^{-1})^3 r_3(z)
\end{aligned}
$$

Calculation of STF and NTF for MASH topology (Step 1)

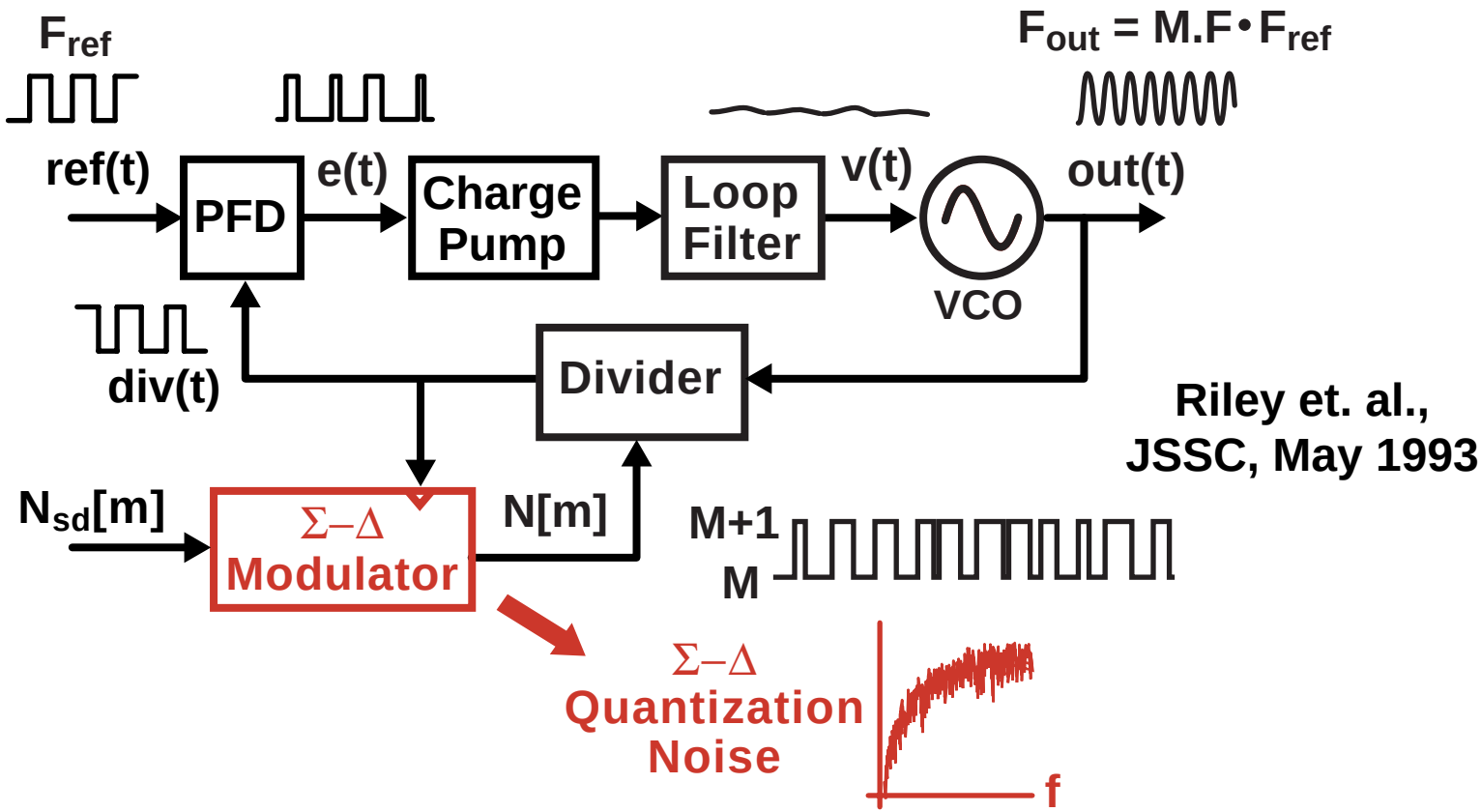


- **Overall modulator behavior**

$$y(z) = x(z) - (1 - z^{-1})^3 r_3(z)$$

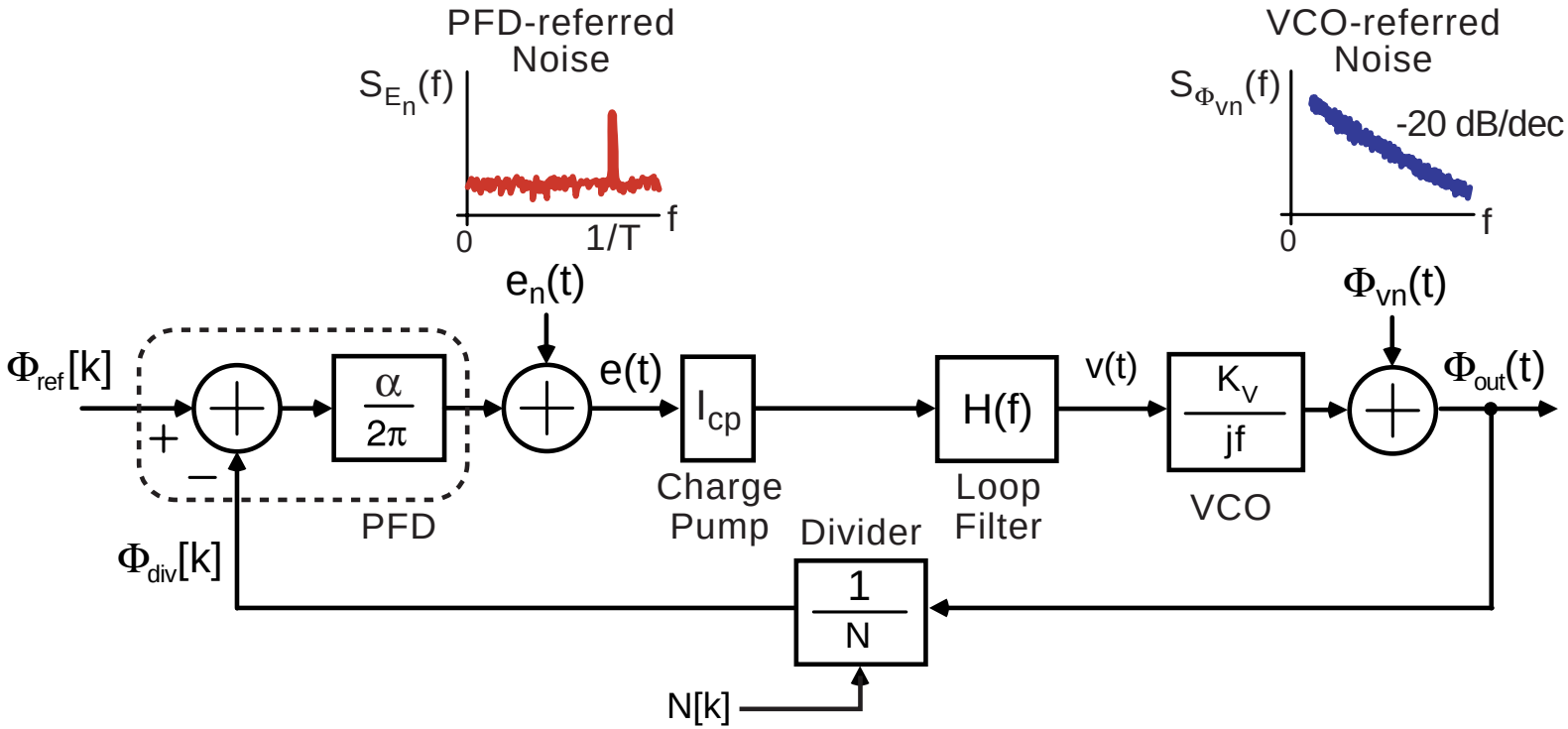
  - **STF: $H_s(z) = 1$**
  - **NTF: $H_n(z) = (1 - z^{-1})^3$**

Sigma-Delta Frequency Synthesizers



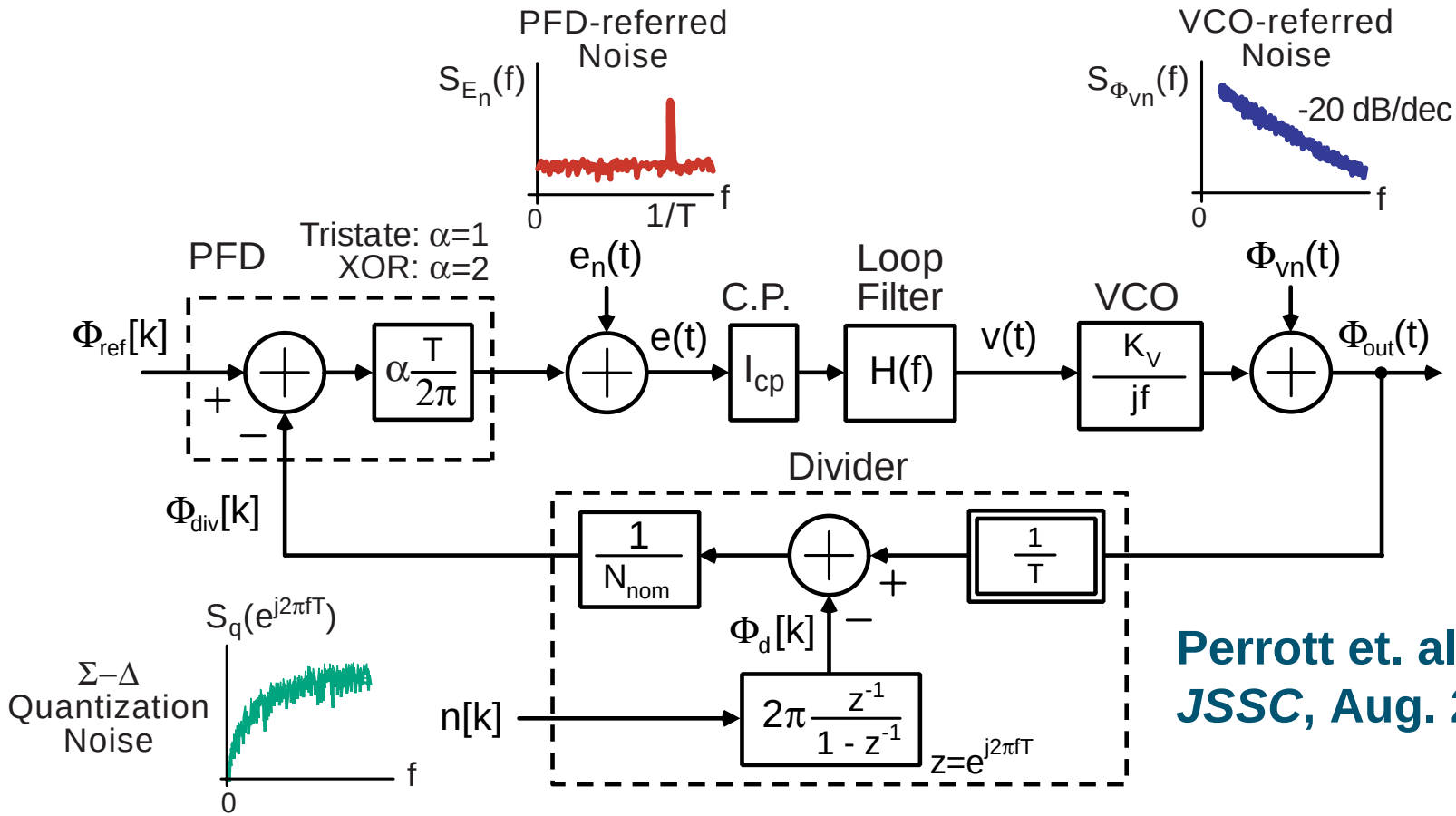
- Use Sigma-Delta modulator rather than accumulator to perform dithering operation
  - Achieves much better spurious performance than classical fractional-N approach

The Need for A Better PLL Model



- **Classical PLL model**
  - **Predicts impact of PFD and VCO referred noise sources**
  - **Does not allow straightforward modeling of impact due to divide value variations**
    - This is a problem when using fractional-N approach

Fractional-N PLL Model

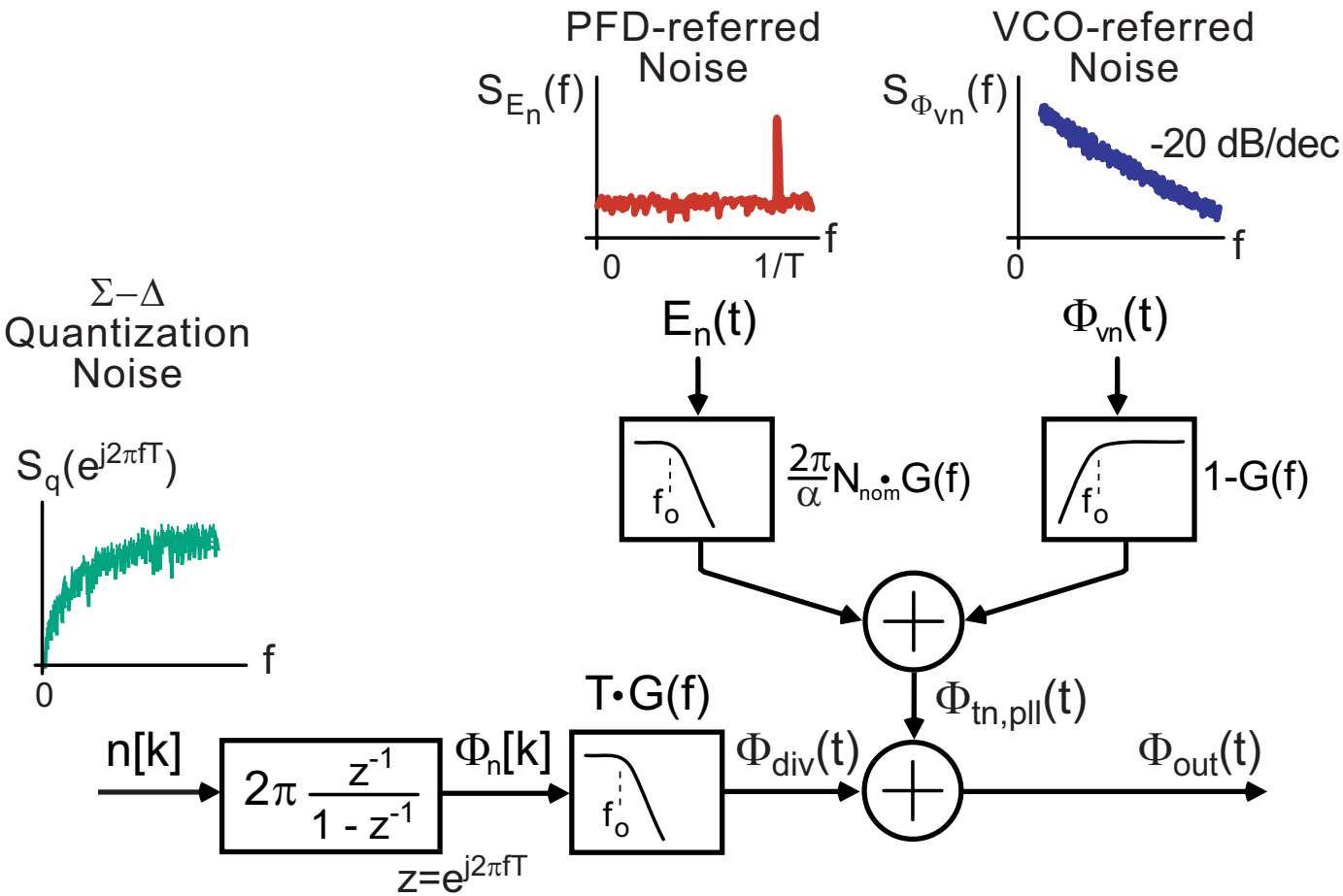


Perrott et. al.
***JSSC*, Aug. 2002**

- **Closed loop dynamics parameterized by**

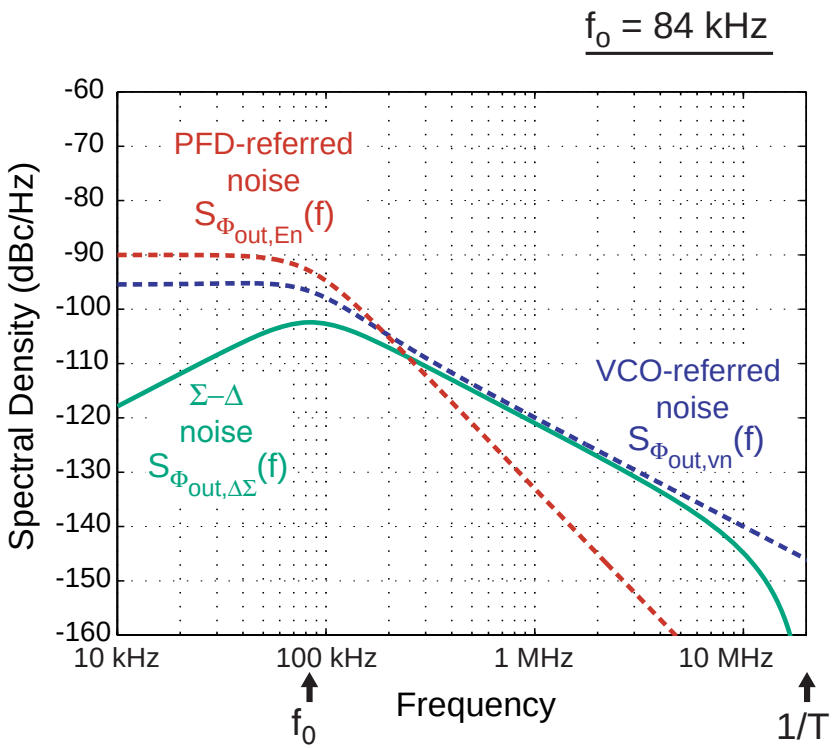
$$G(f) = \frac{A(f)}{1 + A(f)} \quad \text{where} \quad A(f) = \frac{\alpha I_{cp} H(f) K_V}{N_{nom} 2\pi j f}$$

Parameterized PLL Noise Model



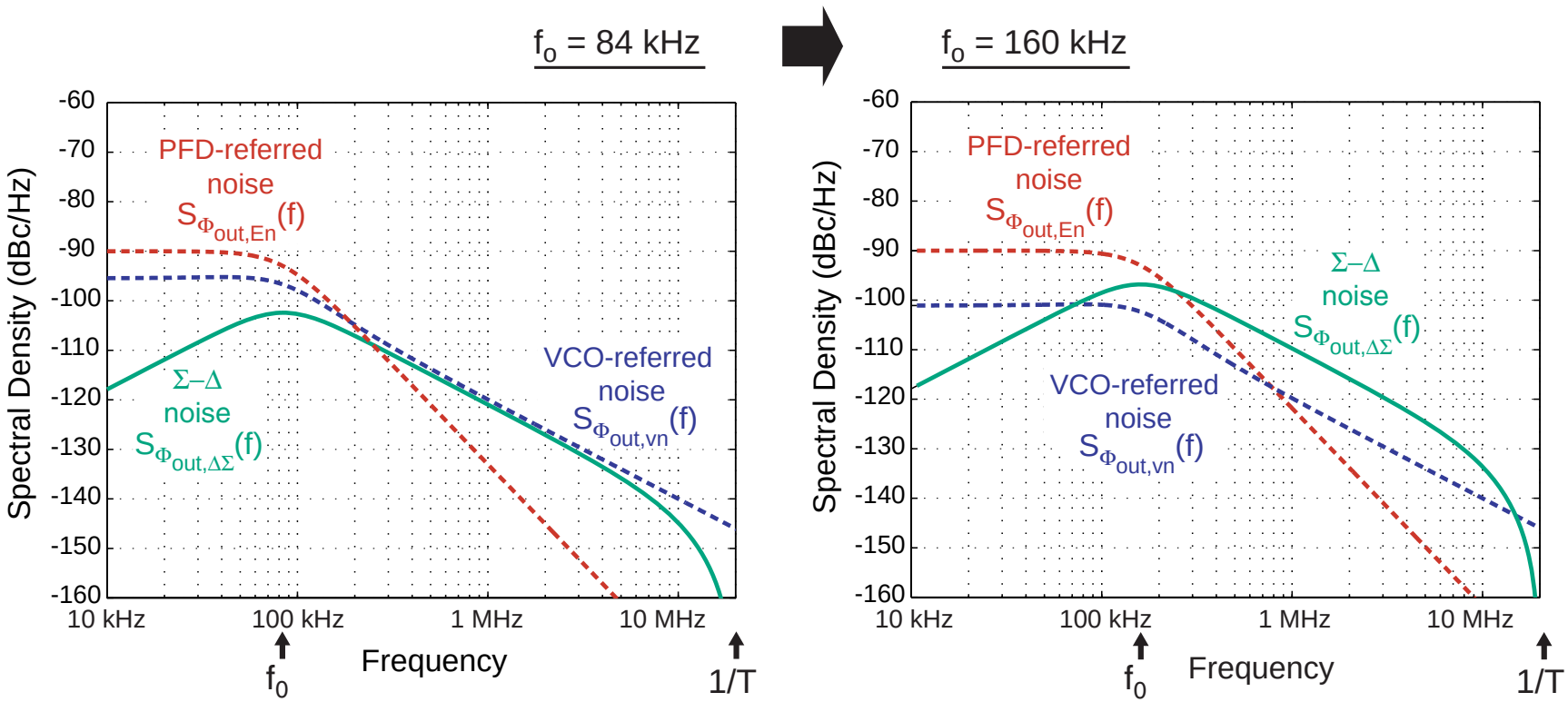
- **Design revolves around choice of Σ–Δ and G(f)**
  - **We will focus on G(f) design here**

A Well Designed Sigma-Delta Synthesizer



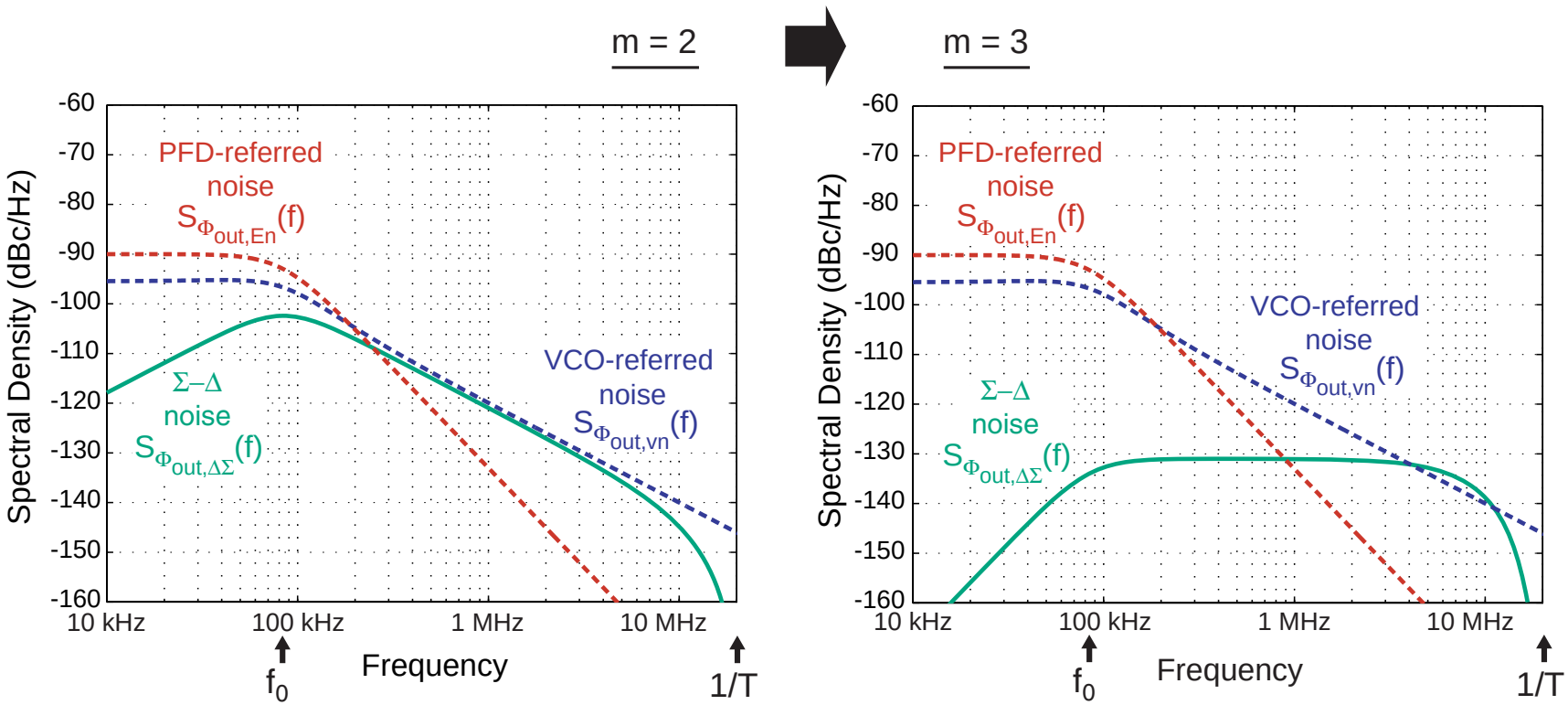
- **Order of G(f) is set to equal to the Sigma-Delta order**
  - **Sigma-Delta noise falls at -20 dB/dec above G(f) bandwidth**
- **Bandwidth of G(f) is set low enough such that synthesizer noise is dominated by intrinsic PFD and VCO noise**

Impact of Increased PLL Bandwidth



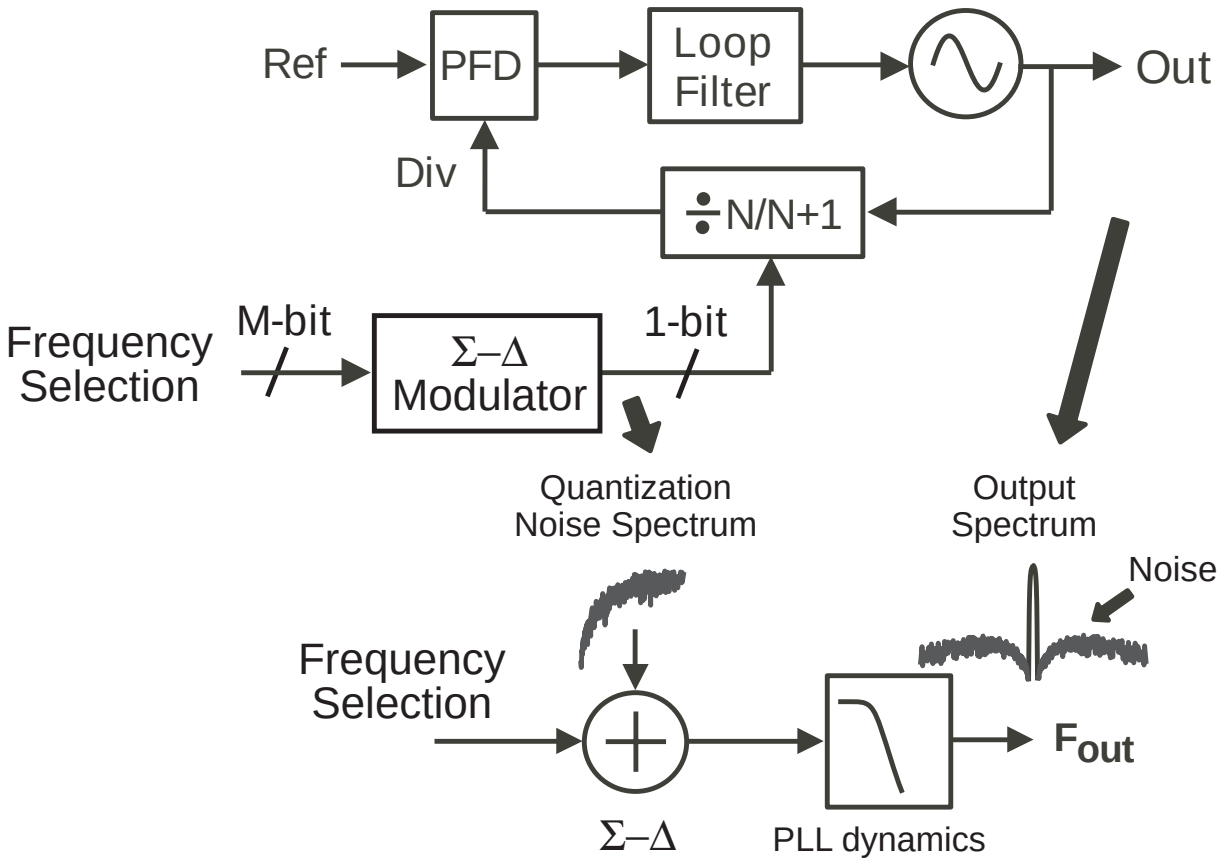
- Allows more PFD noise to pass through
- Allows more Sigma-Delta noise to pass through
- Increases suppression of VCO noise

Impact of Increased Sigma-Delta Order



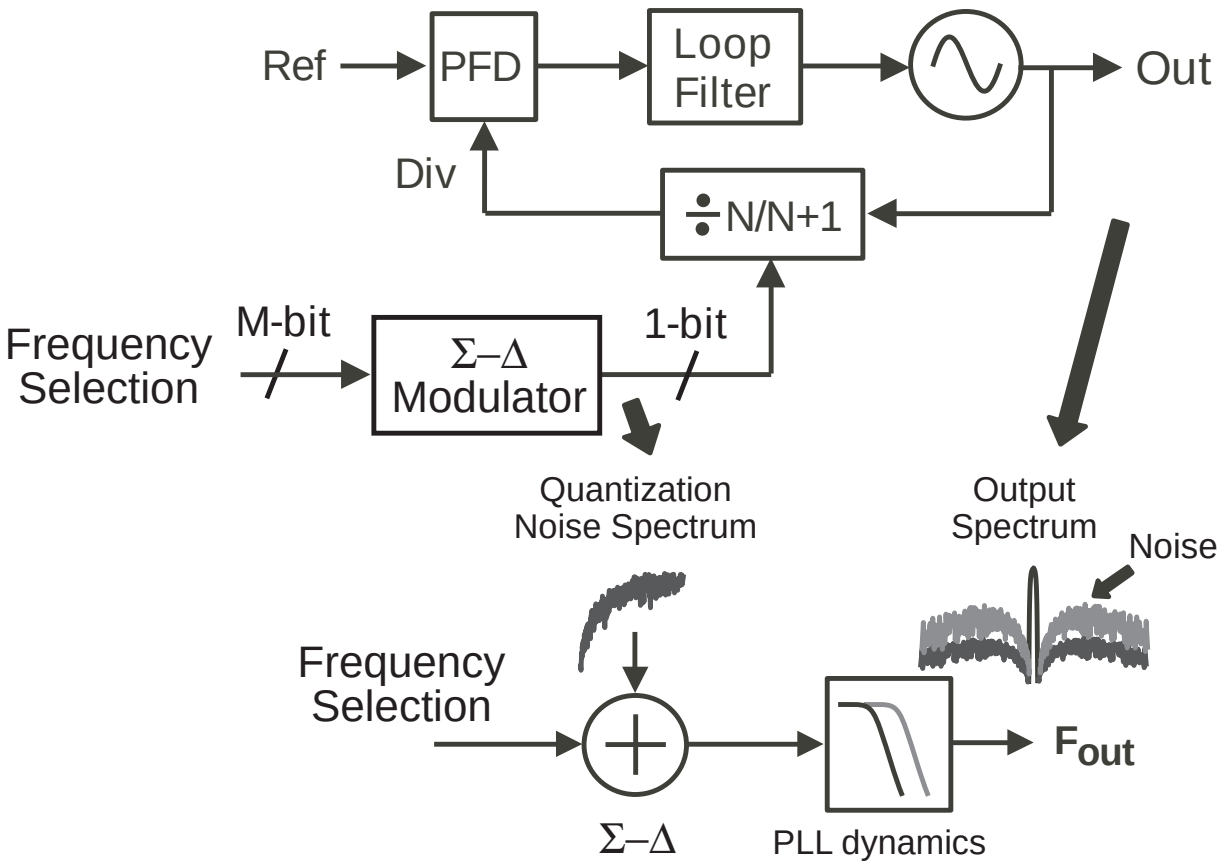
- **PFD and VCO noise unaffected**
- **Sigma-Delta noise no longer attenuated by G(f) such that a -20 dB/dec slope is achieved above its bandwidth**

Impact of Σ–Δ Quantization Noise on Synth. Output



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

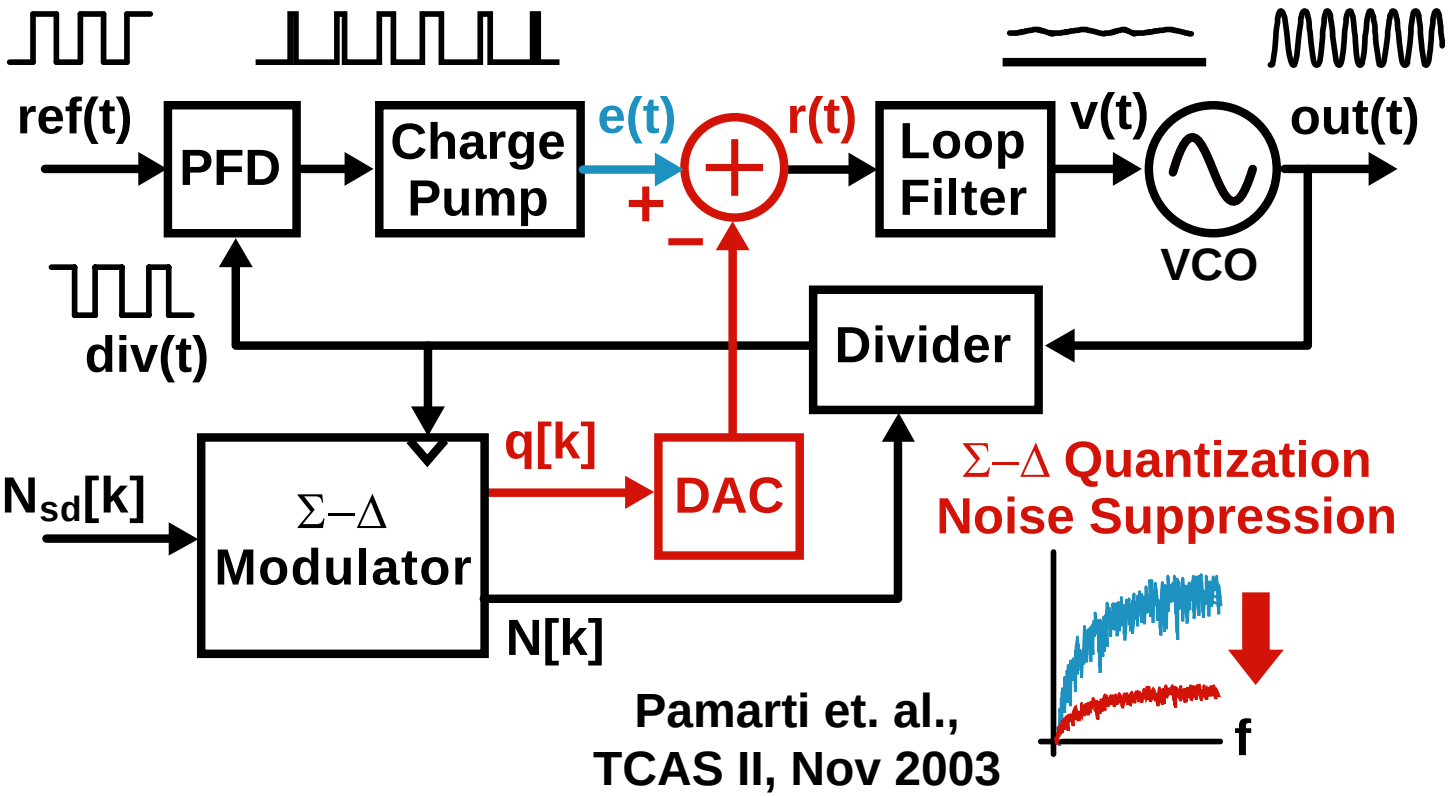
Impact of Increasing the PLL Bandwidth



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

Tradeoff: Noise performance vs PLL bandwidth

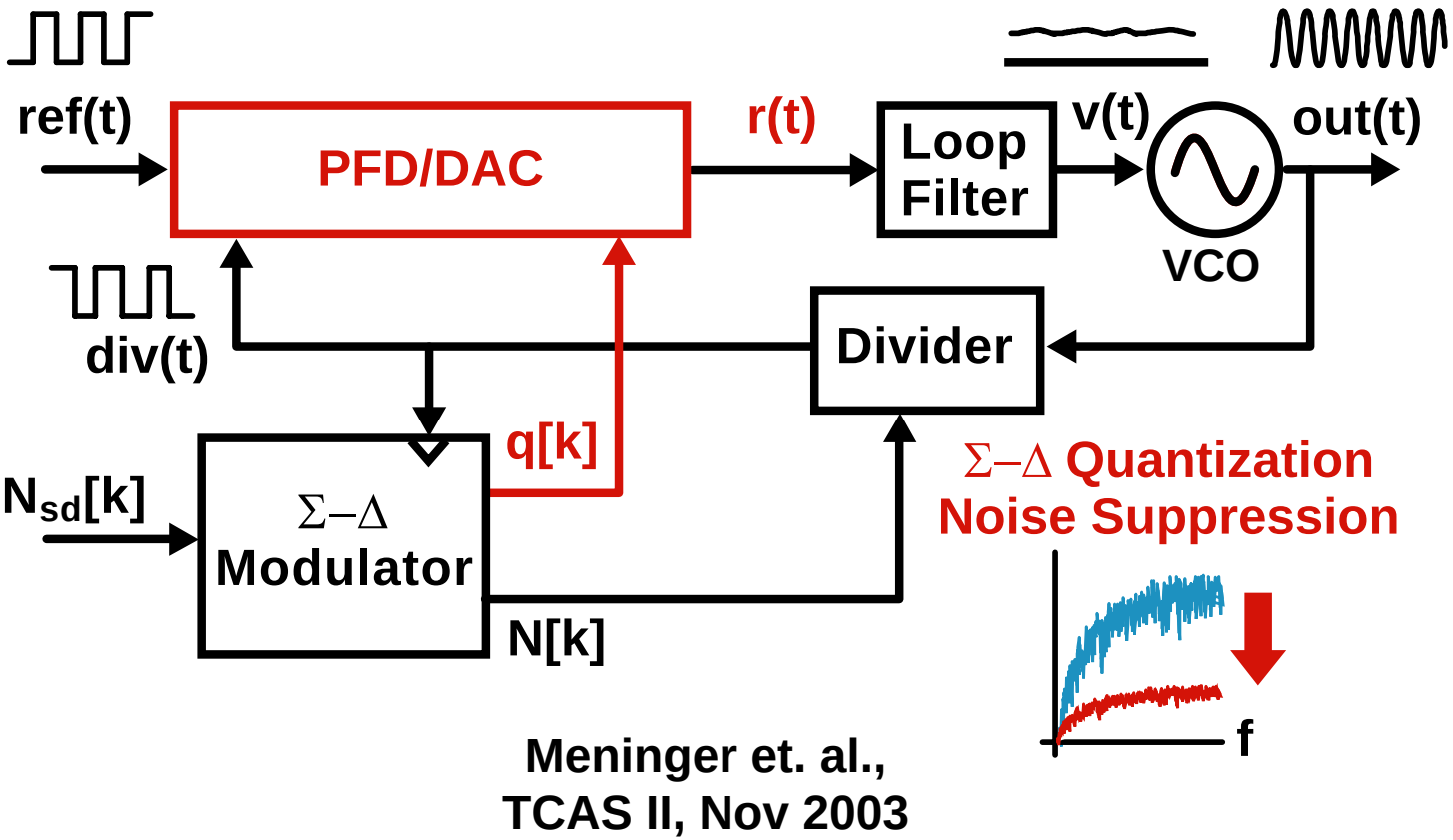
A Cancellation Method for Reducing Quantization Noise



Pamarti et. al.,
TCAS II, Nov 2003

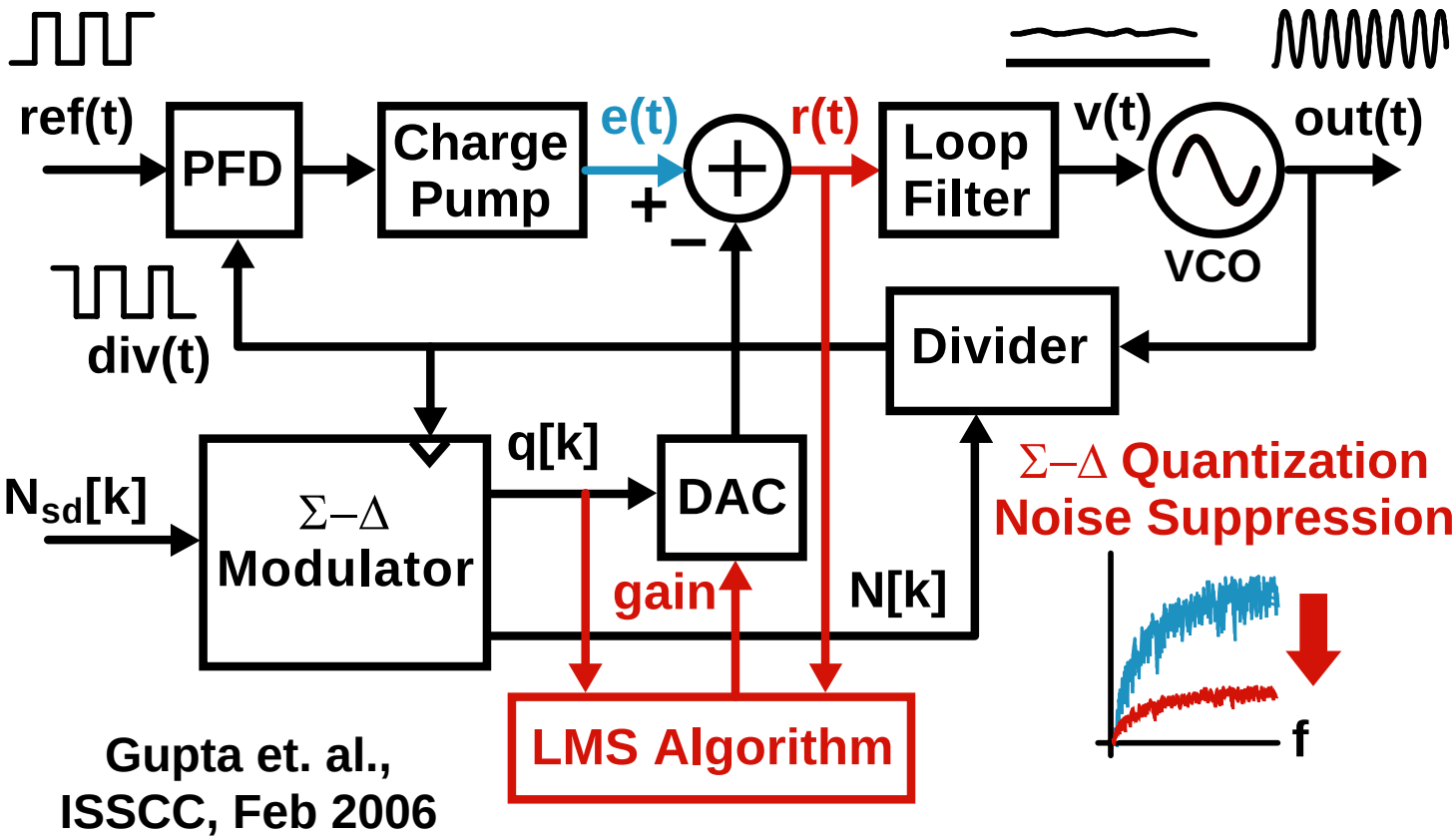
- Key idea: quantization noise can be predicted within the digital Σ–Δ modulator structure
- Issue: cancellation is limited by analog matching
  - Achieves < 20 dB cancellation in practice

Improved Cancellation Through Inherent Matching



- Combined PFD/DAC structure achieves inherent matching between error and cancellation signal
  - > 29 dB quantization noise cancellation achieved

Improved Cancellation Through Continuous Calibration



- Gain of DAC is adjusted in an adaptive manner using LMS algorithm
  - > 30 dB noise cancellation achieved

Summary of Fractional-N Frequency Synthesizers

- **Fractional-N synthesizers allow very high resolution to be achieved with relatively high reference frequencies**
  - **Cost is introduction of quantization noise due to dithering of divider**
- **Classical fractional-N synthesizers used an accumulator for dithering**
  - **Quantization noise cancellation was attempted**
- **Sigma-Delta fractional-N synthesizers improve quantization noise by utilizing noise shaping techniques**
  - **Key tradeoff: PLL bandwidth versus phase noise**
  - **Quantization noise cancellation has made a comeback**