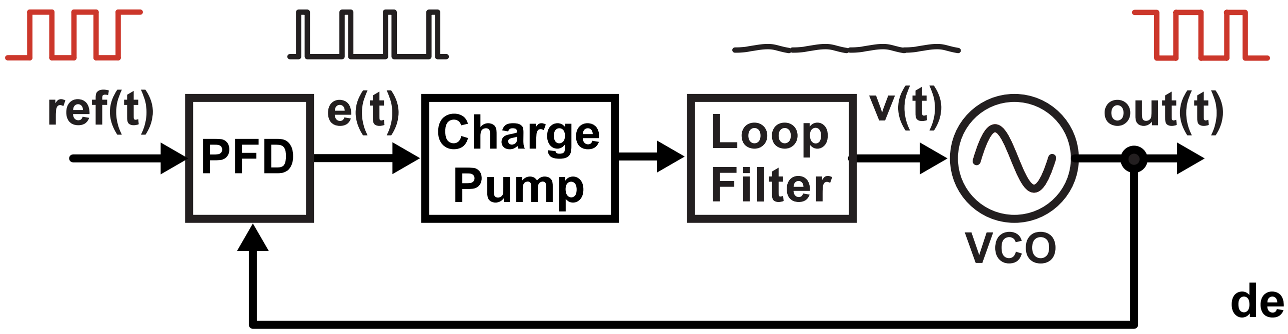
MIT OPENCOURSEWARE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.776
High Speed Communication Circuits
Lecture 21

Overview of Phase-Locked Loops and Integer-N Frequency Synthesizers

Michael Perrott
Massachusetts Institute of Technology
April 28, 2005

What Is A Phase-Locked Loop?

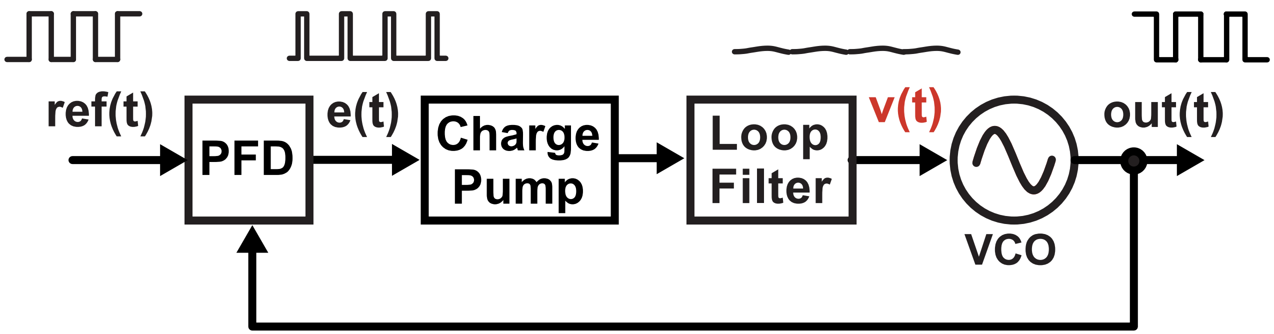


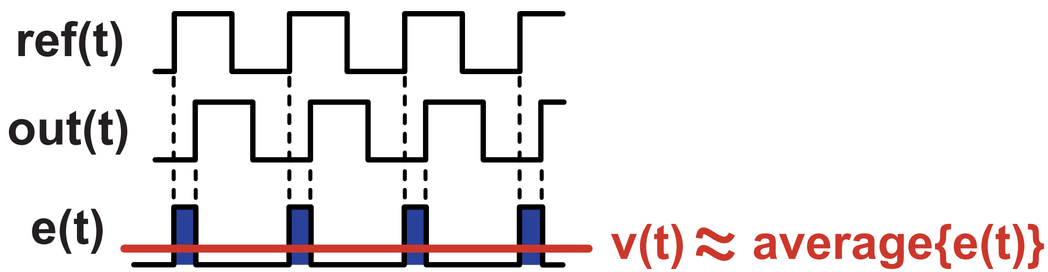
de Bellescize
Onde Electr, Vol. 11
1932

- 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

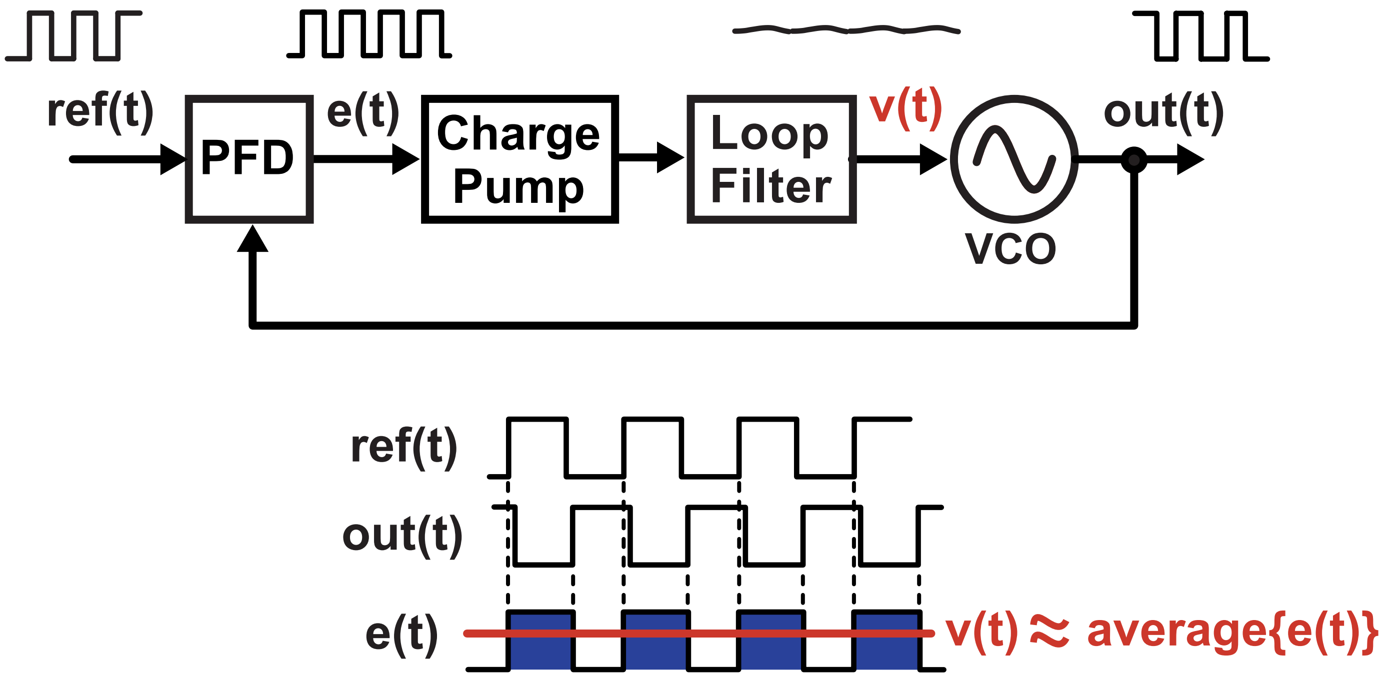
Method of Phase Detection





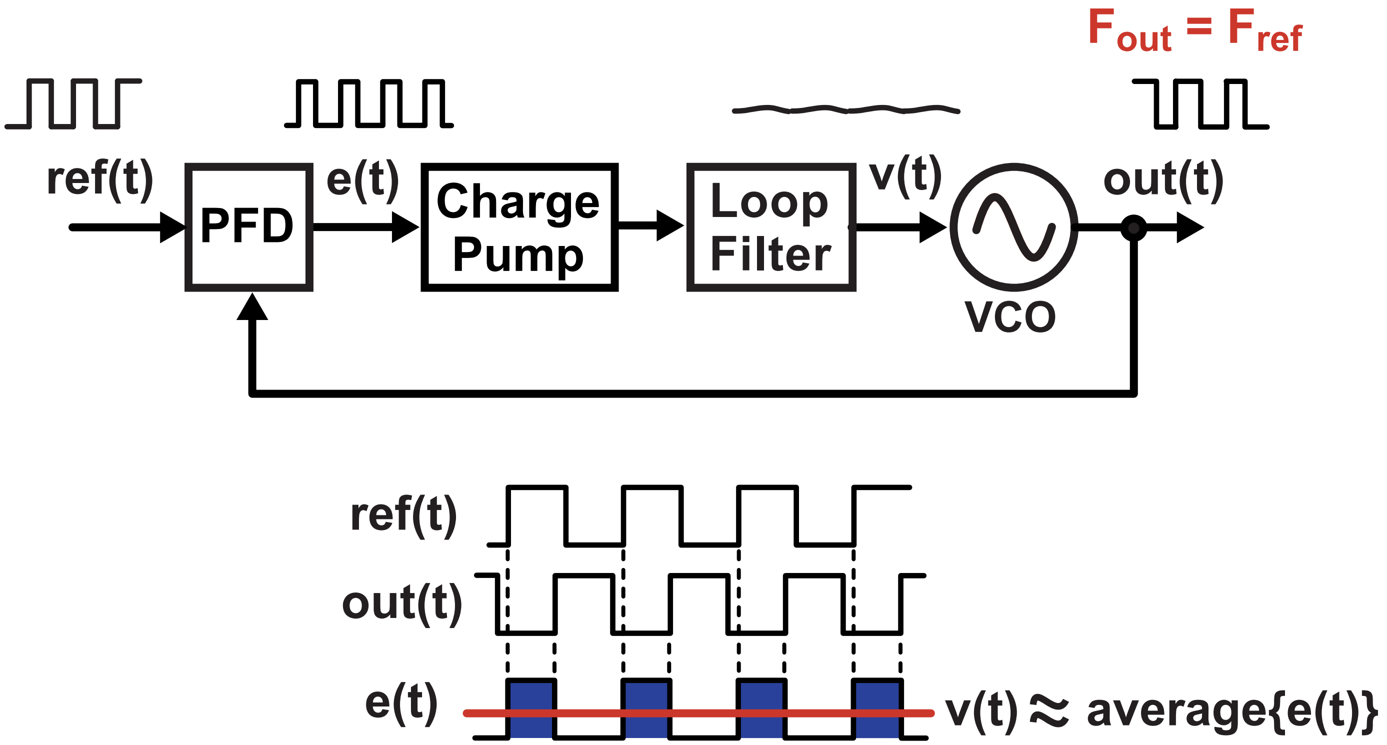
- **PFD output consists of pulses whose width is proportional to the phase error**
  - **Phase is only observable at edges**
- **Smooth PFD output to produce input voltage to VCO**

Impact of Changes in Phase Error



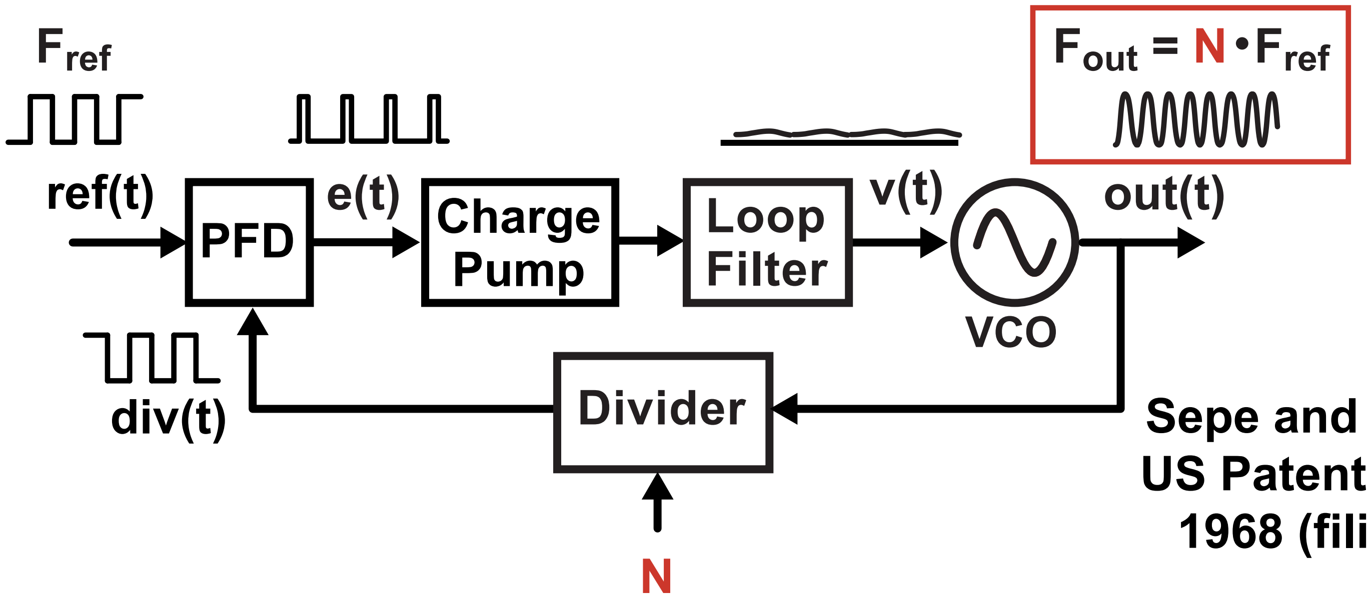
- Pulse width varies according to phase difference
- VCO input voltage changes accordingly
  - Adjusts VCO frequency and phase

Phase Lock Implies Frequency Lock



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

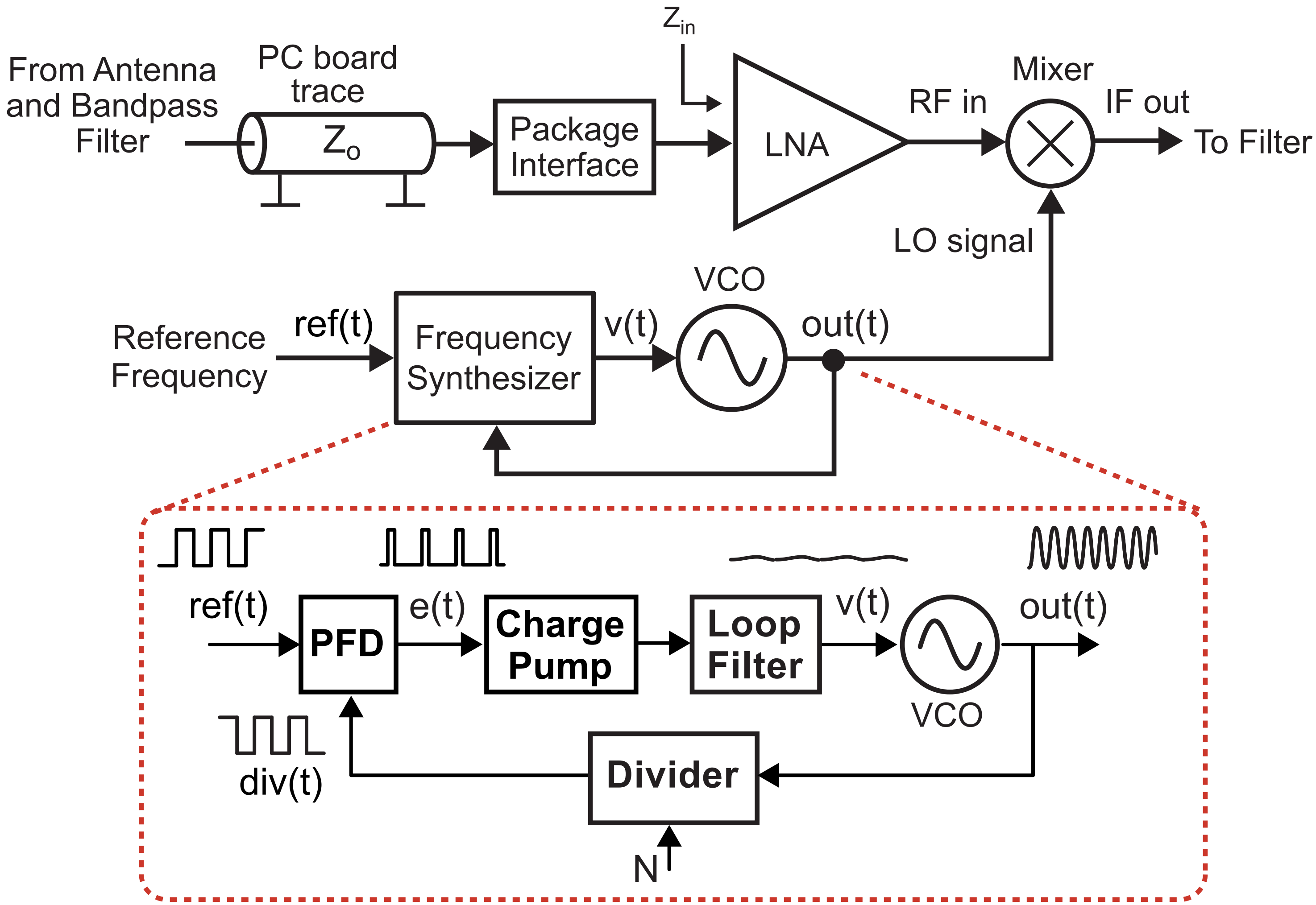
Integer-N Frequency Synthesizer



Sepe and Johnston
US Patent 3,551,826
1968 (filing date)

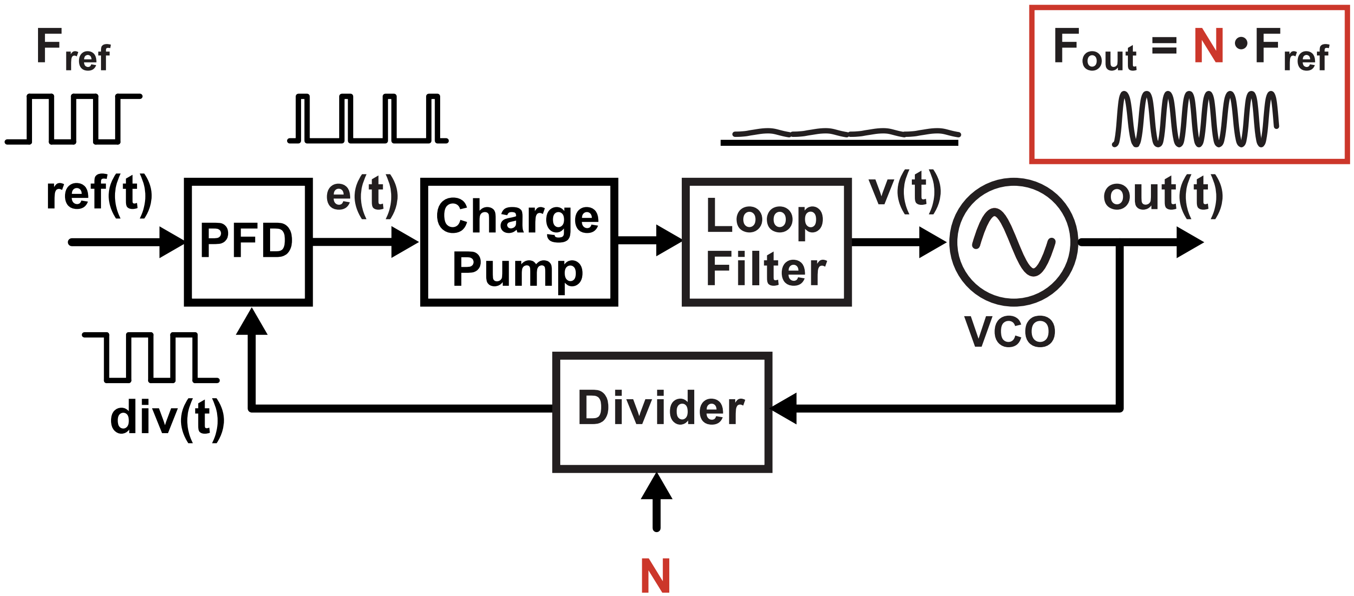
- Leverages frequency divider to create "indirect" frequency multiplication
  - Allows digital adjustment of output frequency in increments of the reference frequency

Integer-N Frequency Synthesizers in Wireless Systems



- **Design Issues: settling time, frequency resolution, noise, power**

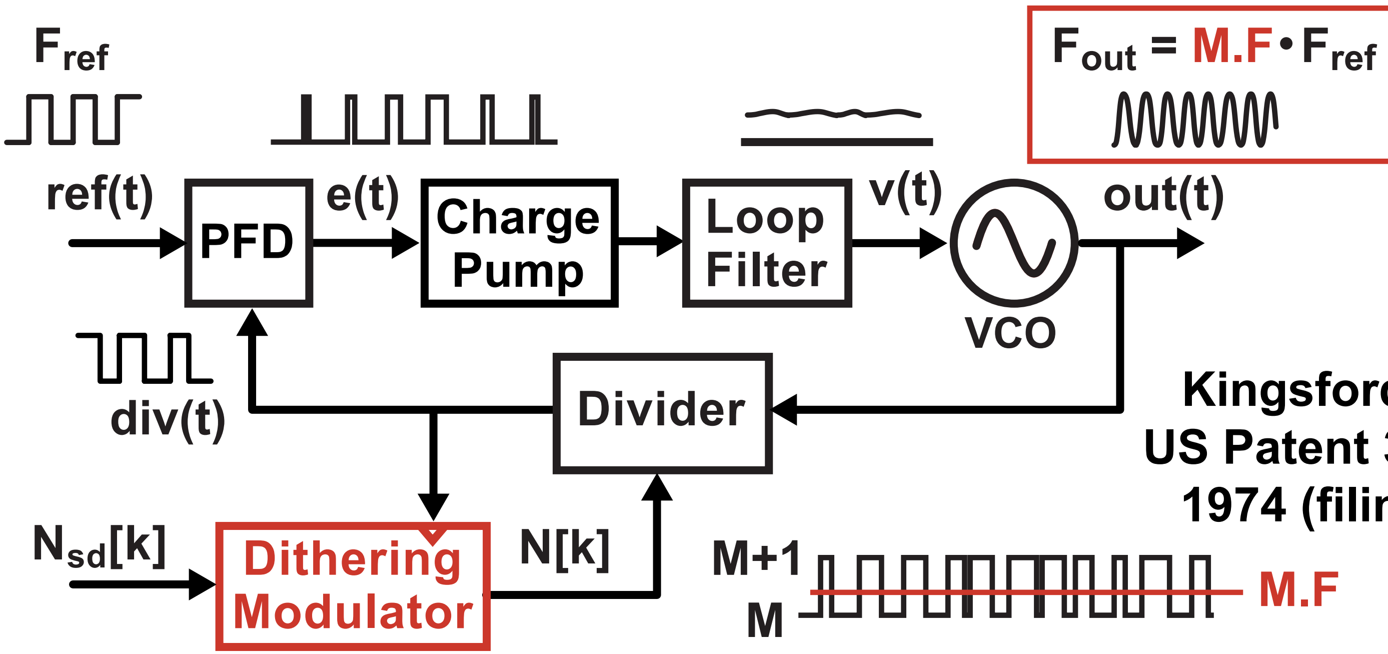
A Key Limitation of Integer-N Synthesizers



- 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

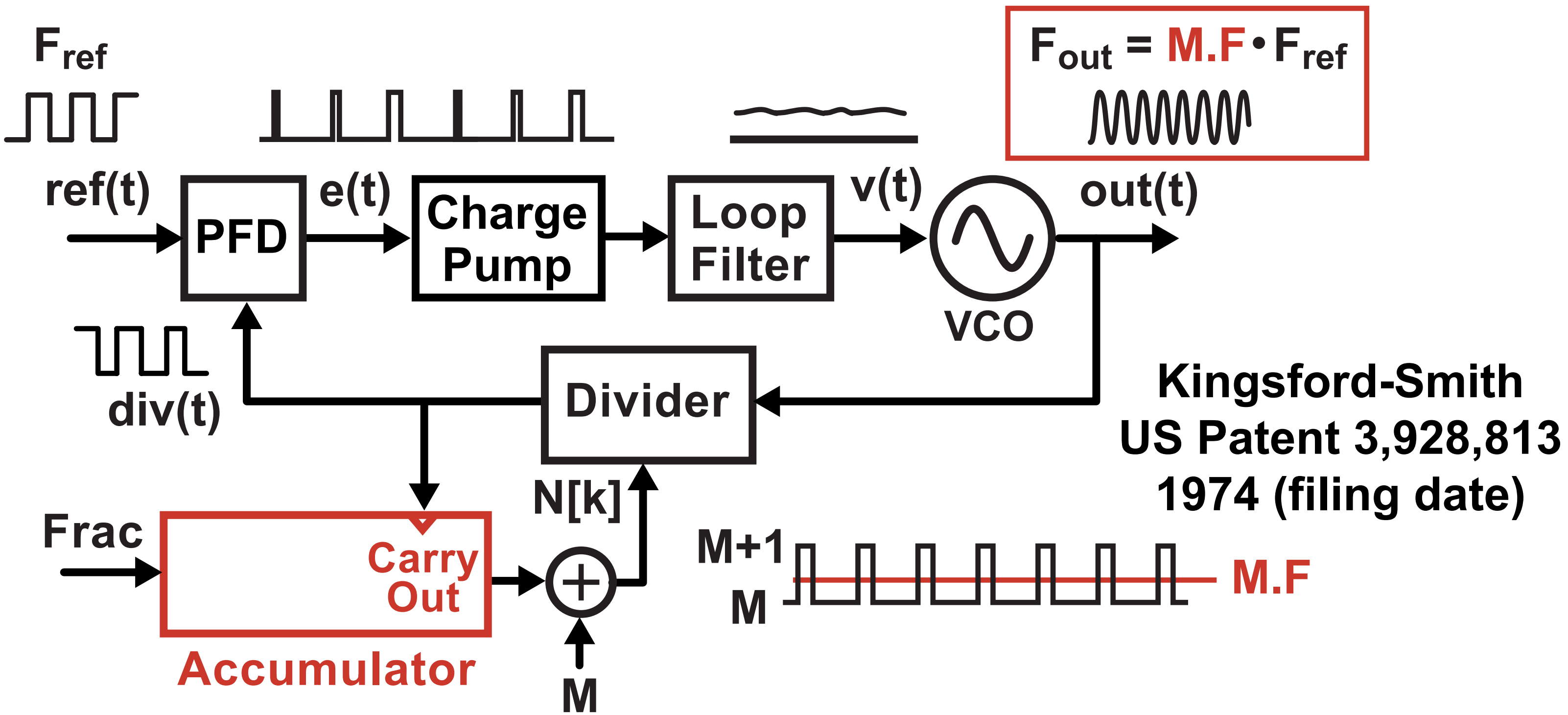


Kingsford-Smith
US Patent 3,928,813
1974 (filing date)

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

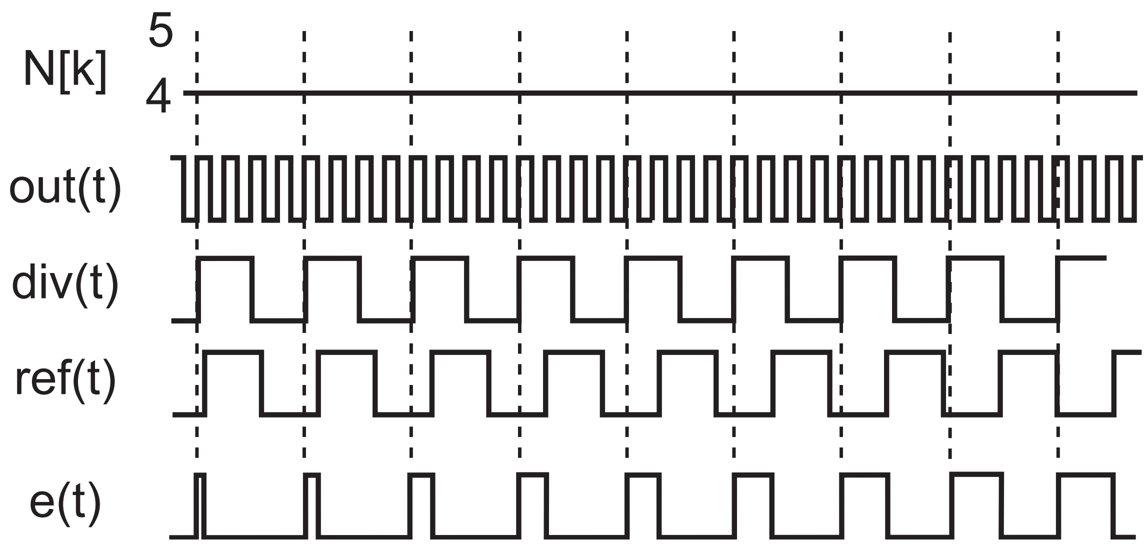
Very high frequency resolution

Classical Fractional-N Synthesizer Architecture



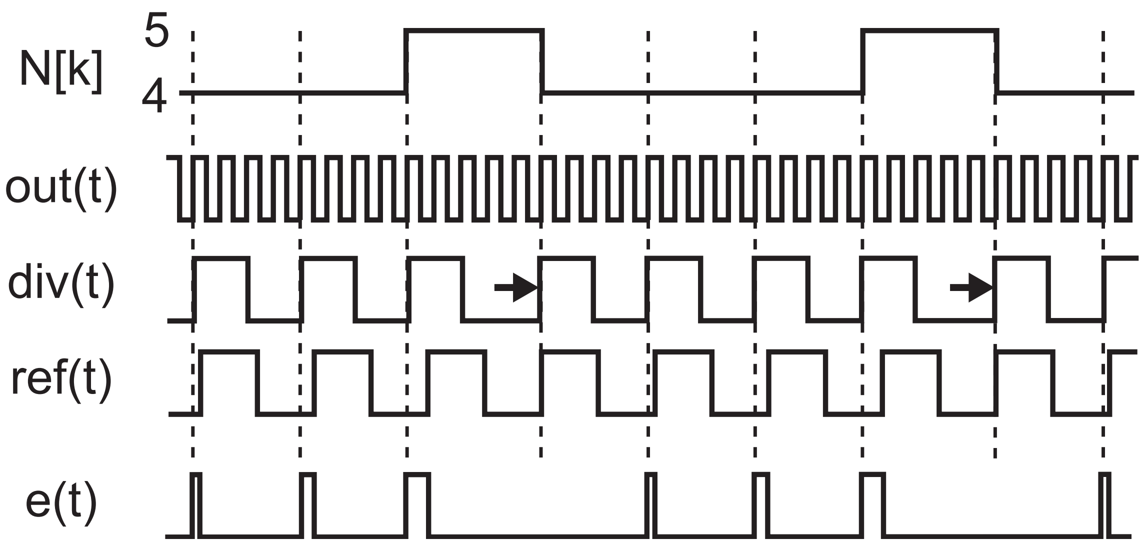
- Use an accumulator to perform dithering operation
  - Fractional input value fed into accumulator
  - Carry out bit of accumulator fed into divider

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



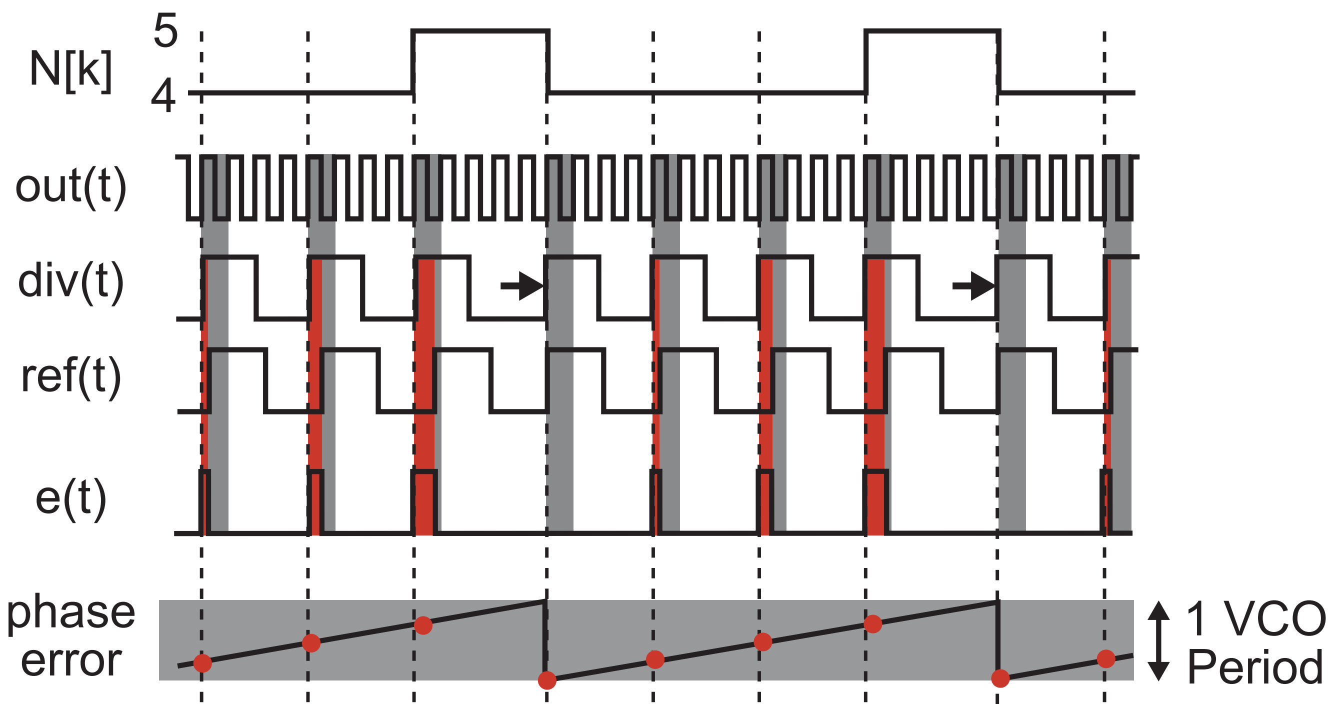
- **Constant divide value of N = 4 leads to frequency error**
  - **Error pulse widths increase as phase error accumulates**

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



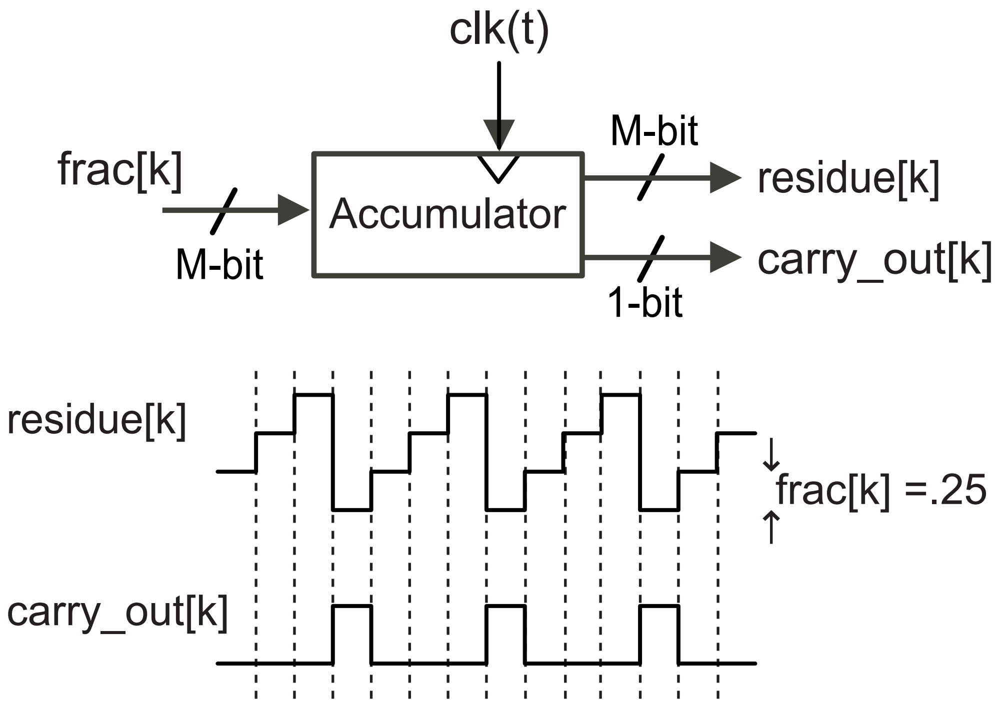
- **Dithering allows average divide value of N = 4.25**
  - **Reset phase error by periodically "swallowing" a VCO cycle**
    - Achieved by dividing by 5 every 4 reference cycles

Key Observations for Classical Fractional-N Dithering



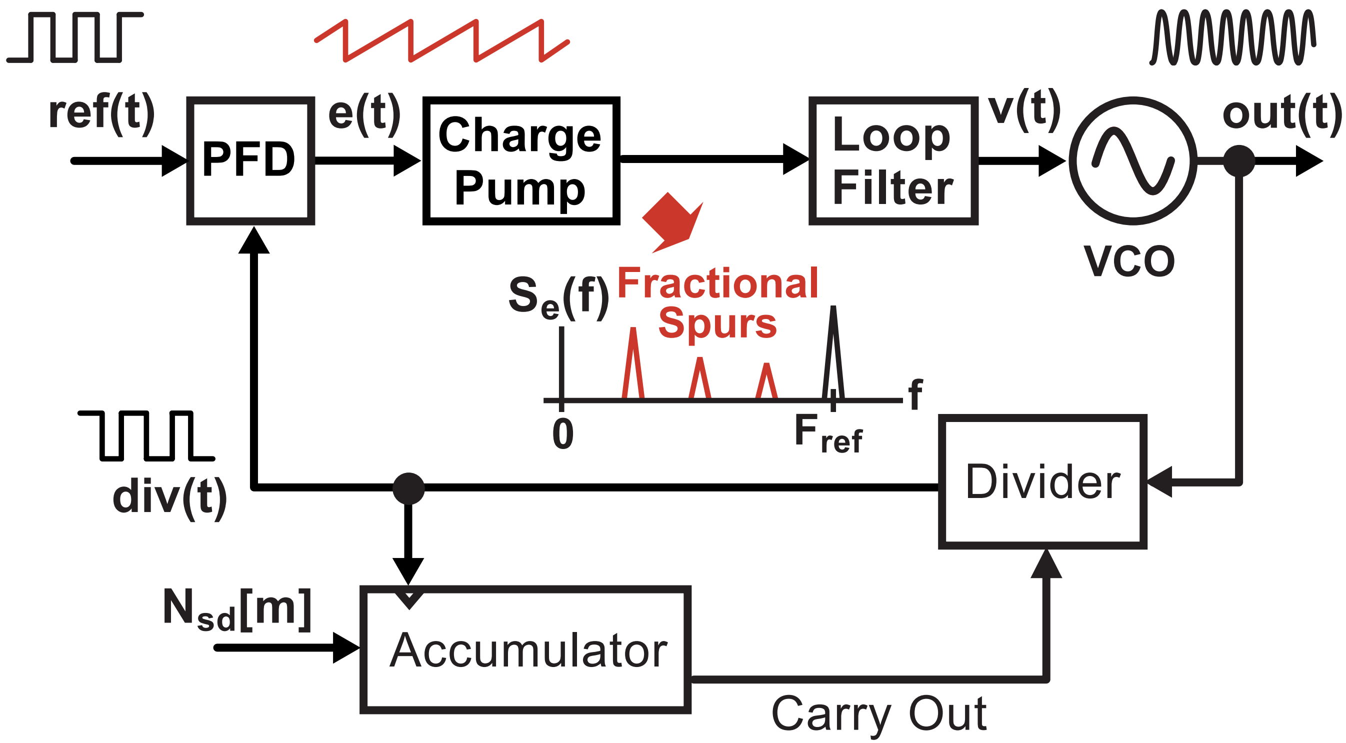
- The instantaneous phase error always remains less than one VCO cycle
- We can directly relate the phase error to the residue of the accumulator that is providing the dithering

Accumulator Operation



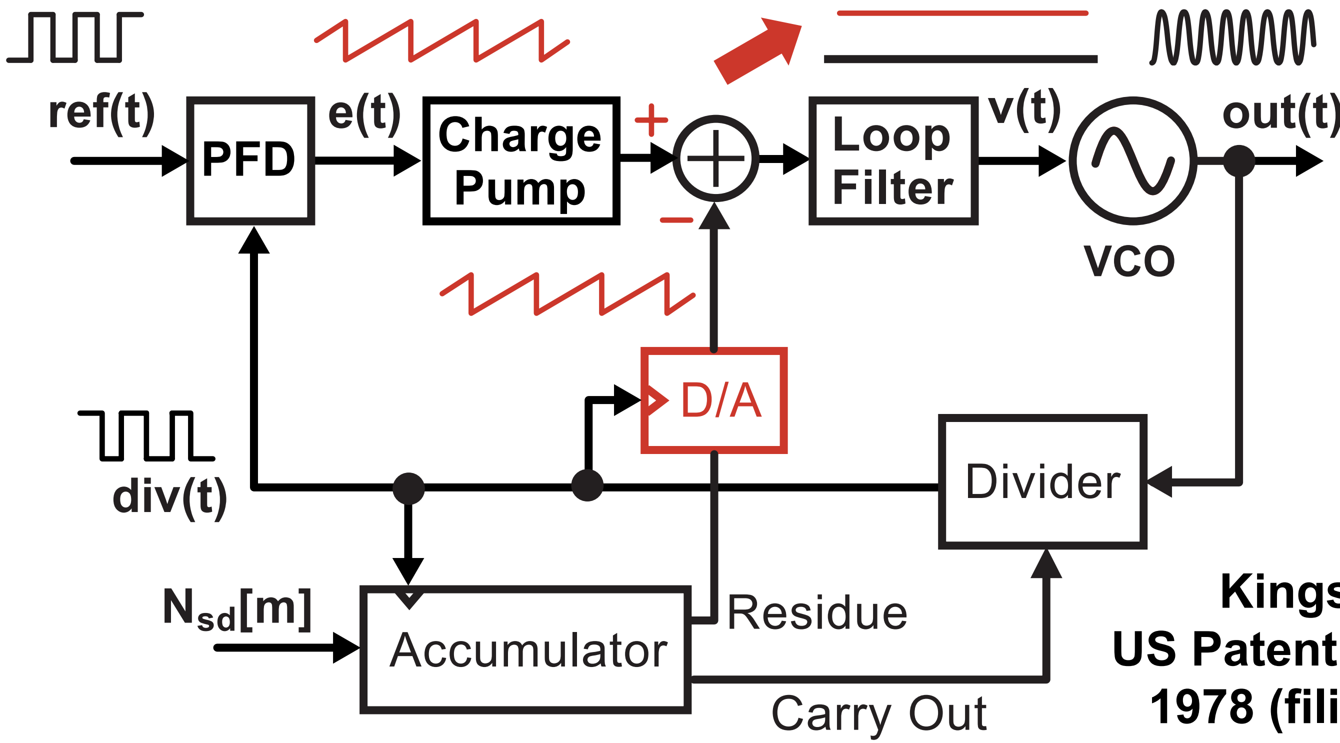
- Carry out bit is asserted when accumulator residue reaches or surpasses its full scale value
- Accumulator residue corresponds to instantaneous phase error
  - Increments by the fractional value input into the accumulator

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

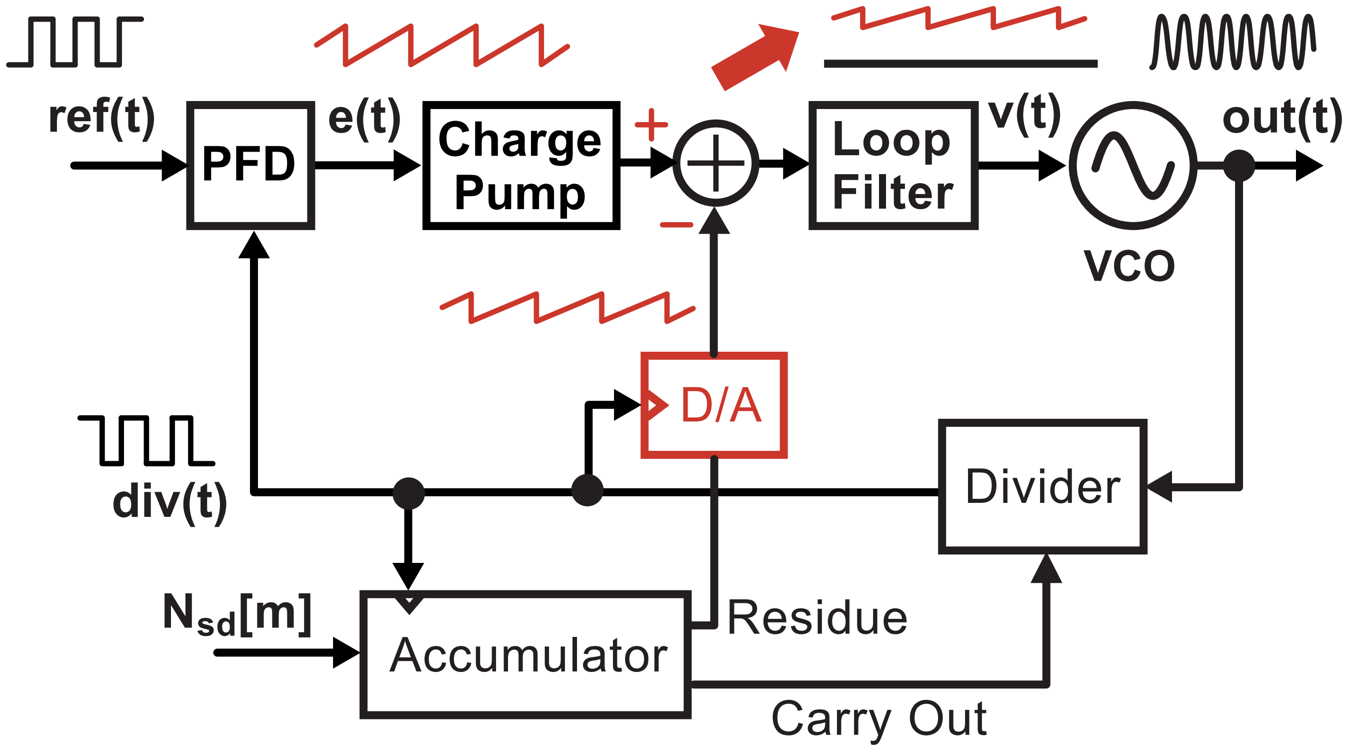
The Phase Interpolation Technique



Kingsbury
US Patent 4,179,670
1978 (filing date)

- **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**

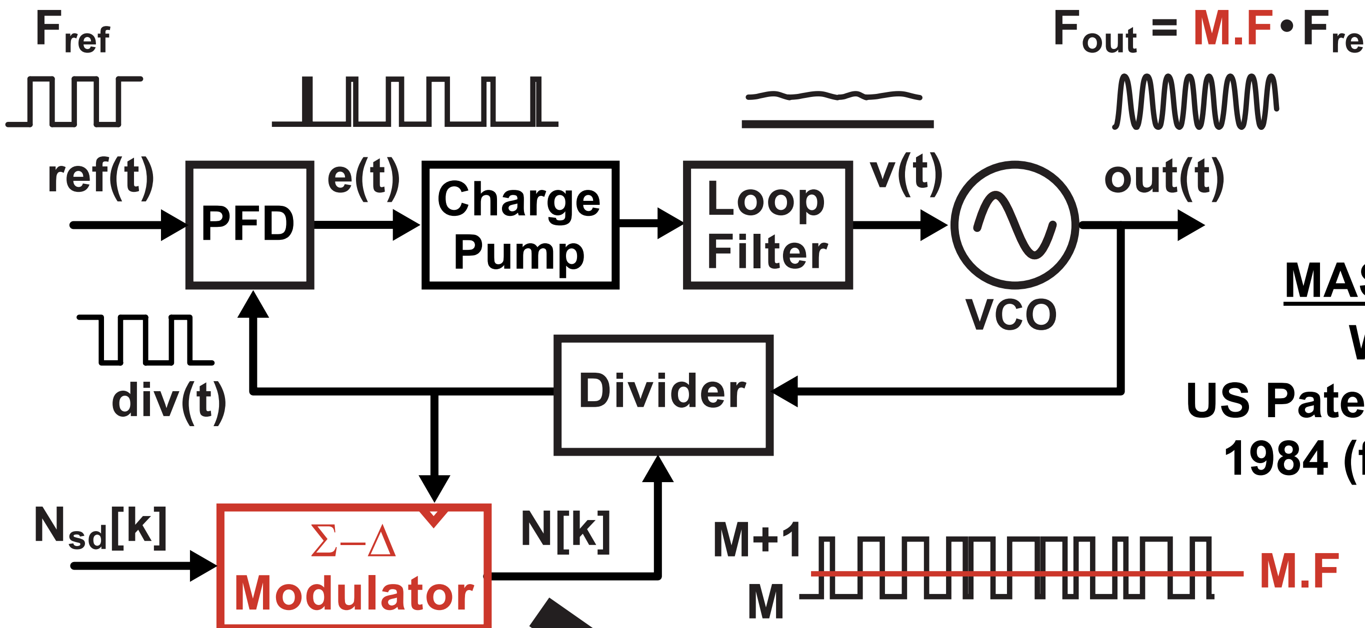
The Problem With Phase Interpolation

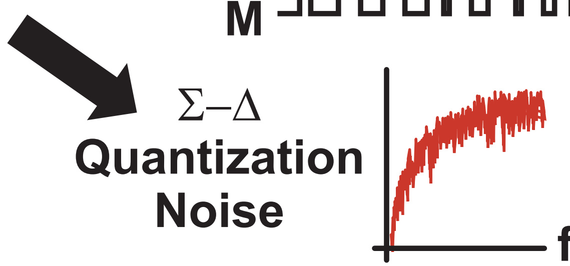


- **Gain matching between PFD error and scaled D/A output must be extremely precise**
  - **Any mismatch will lead to spurious tones at PLL output**

Matching issue prevented this technique from catching on

Σ–Δ Fractional-N Frequency Synthesis



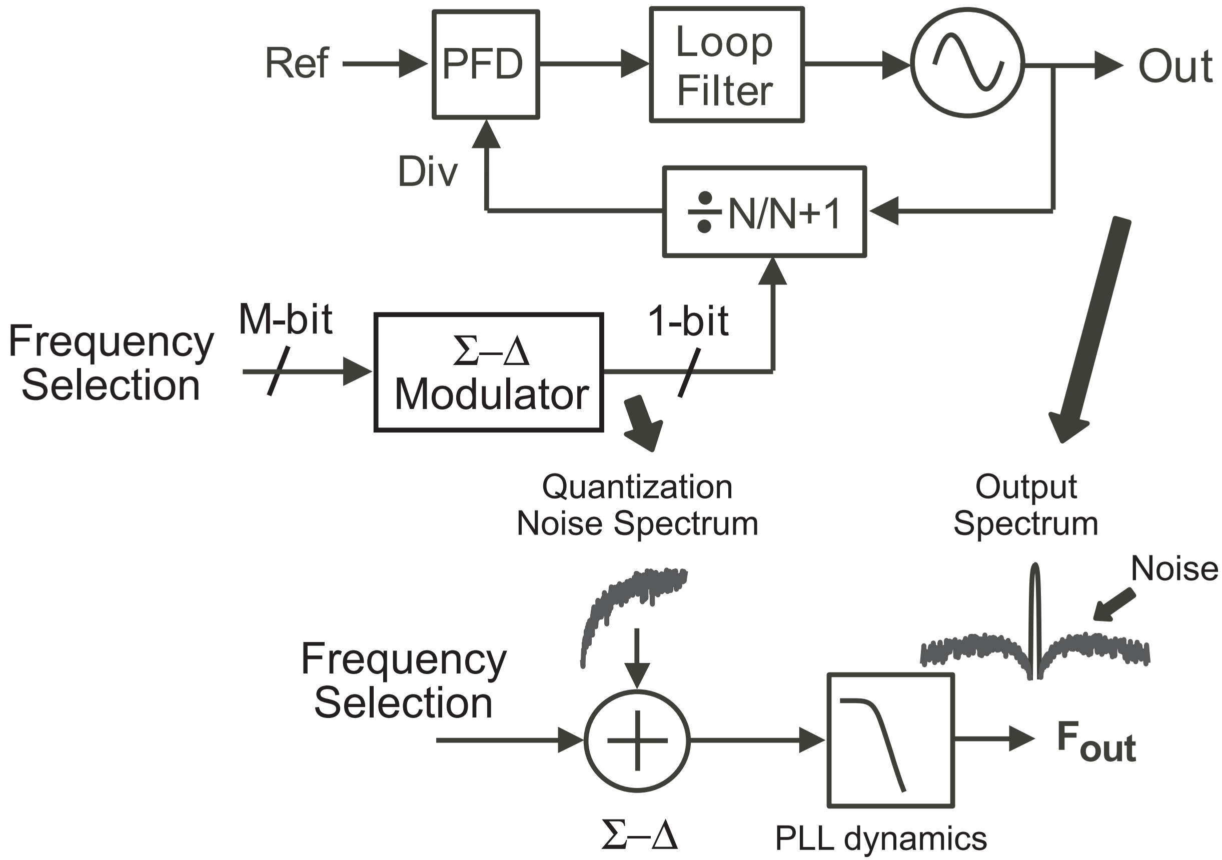


MASH Σ–Δ
Wells
US Patent 4,609,881
1984 (filing date)

General Σ–Δ
Riley
US Patent 4,965,531
1989 (filing date)
JSSC '93

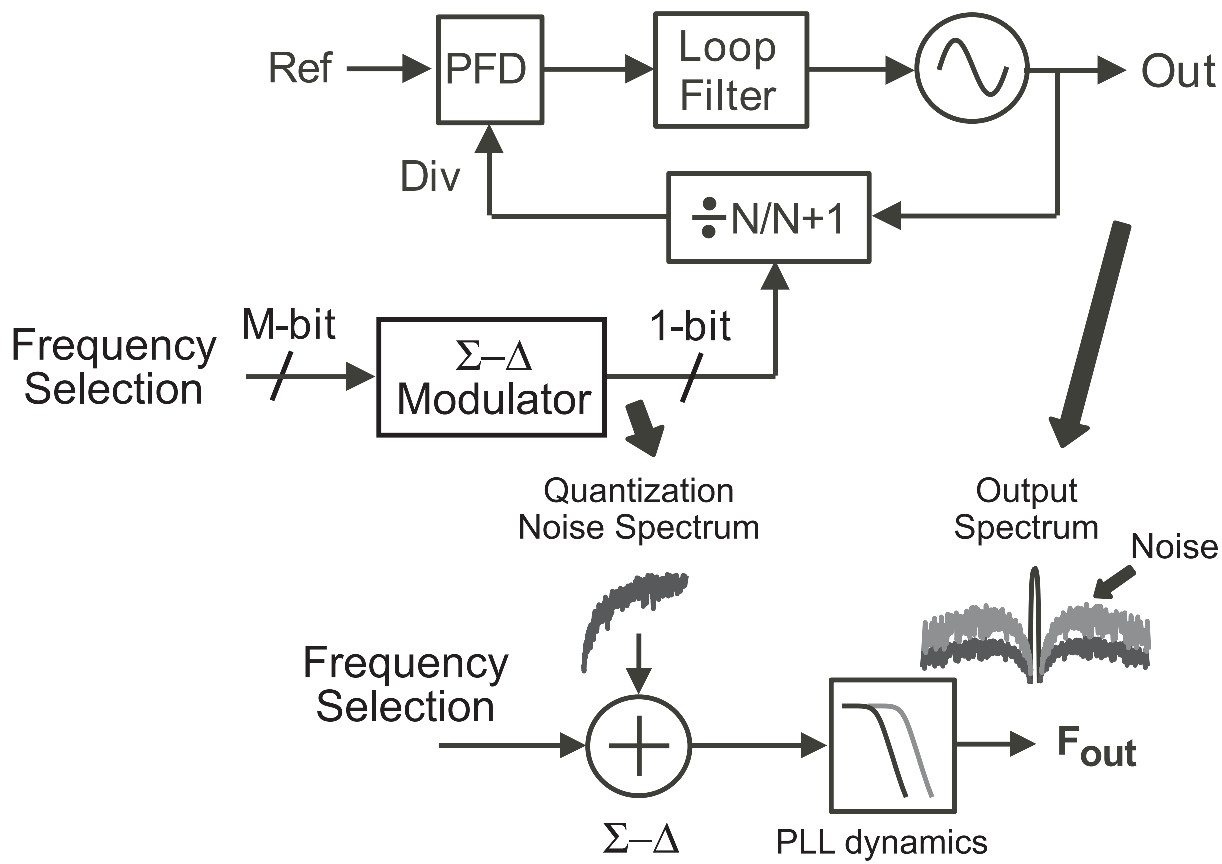
- Dither using a Σ–Δ modulator
  - Quantization noise is shaped to high frequencies
  - Spur content of the quantization noise can be reduced to negligible levels

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

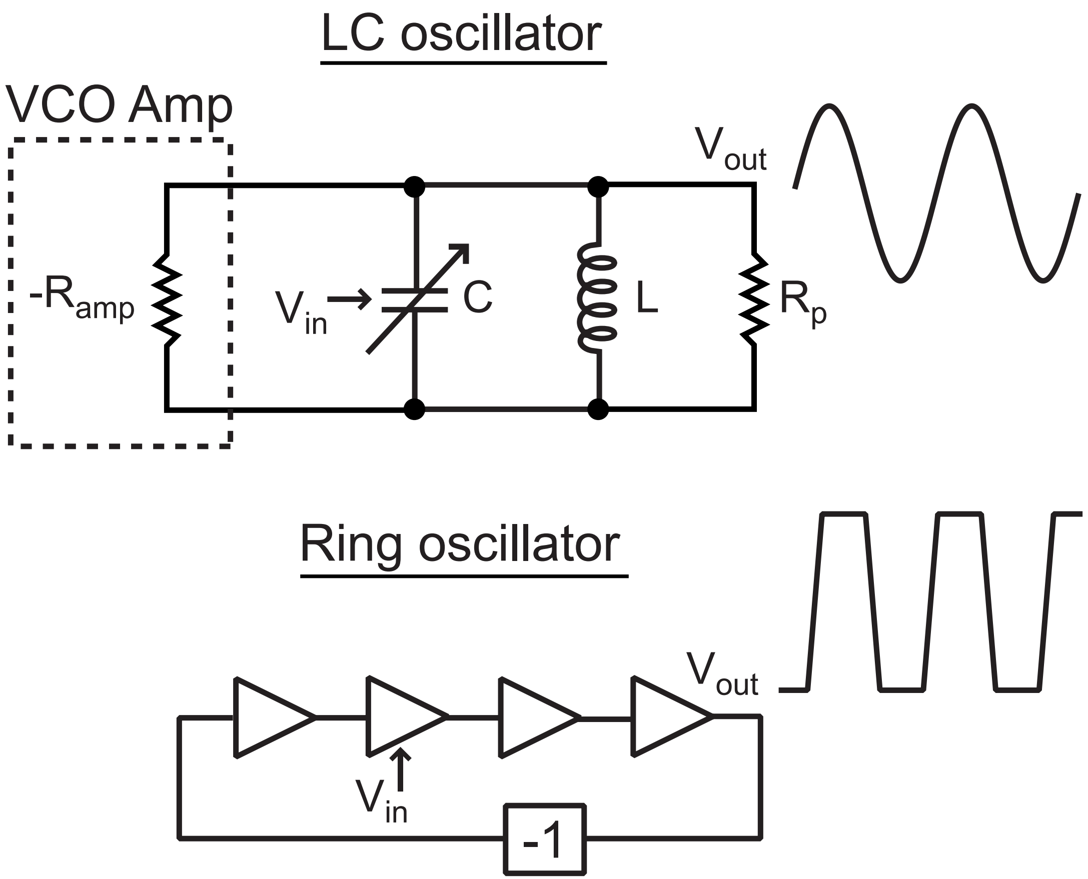
Outline of PLL Lectures

- **Integer-N Synthesizers**
  - **Basic blocks, modeling, and design**
  - **Frequency detection, PLL Type**
- **Noise in Integer-N and Fractional-N Synthesizers**
  - **Noise analysis of integer-N structure**
  - **Sigma-Delta modulators applied to fractional-N structures**
  - **Noise analysis of fractional-N structure**
- **Design of Fractional-N Frequency Synthesizers and Bandwidth Extension Techniques**
  - **PLL Design Assistant Software**
  - **Quantization noise reduction for improved bandwidth and noise**

PLL Building Blocks

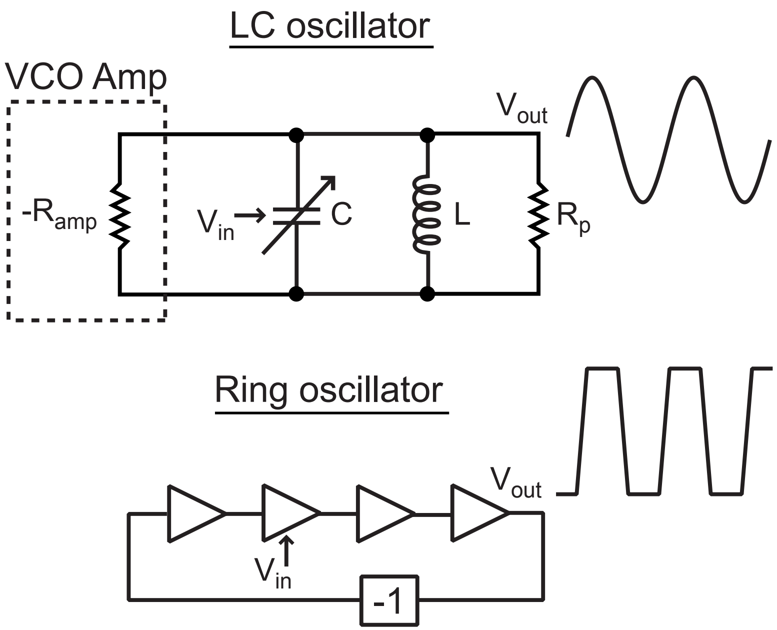
Voltage-Controlled Oscillators

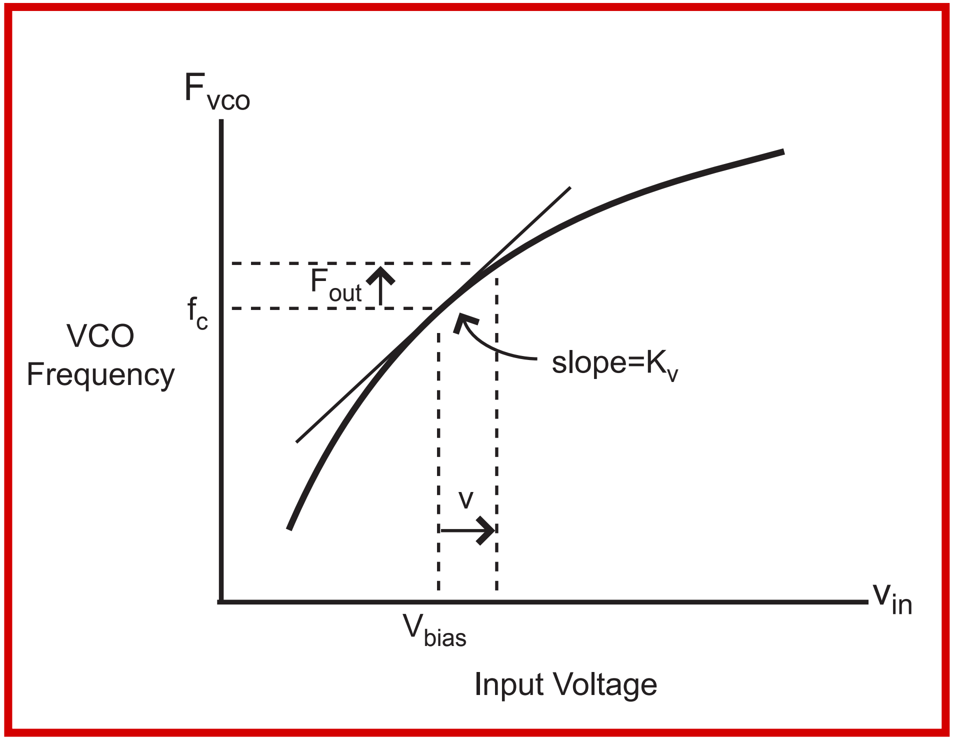
Popular VCO Structures



- **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

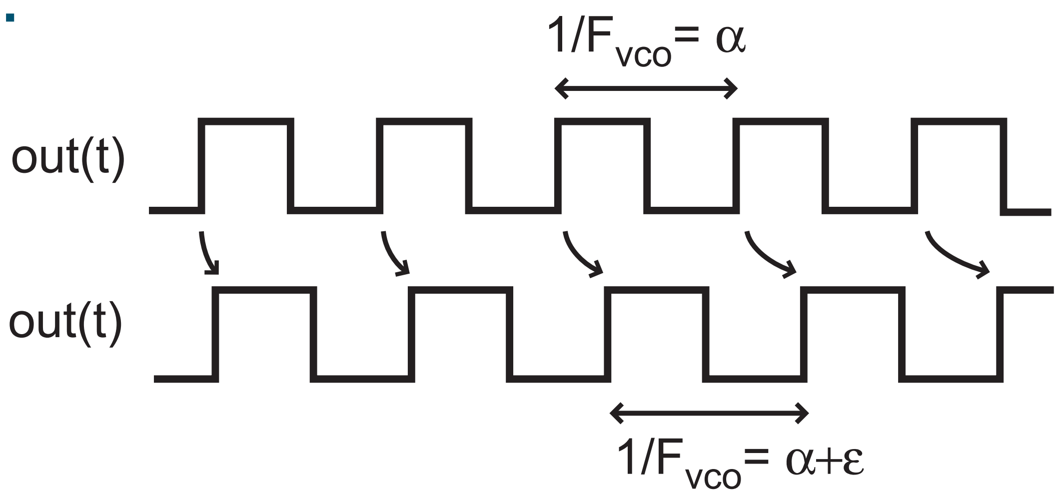
- **Time-domain frequency relationship (from previous slide)**

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

- **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:**

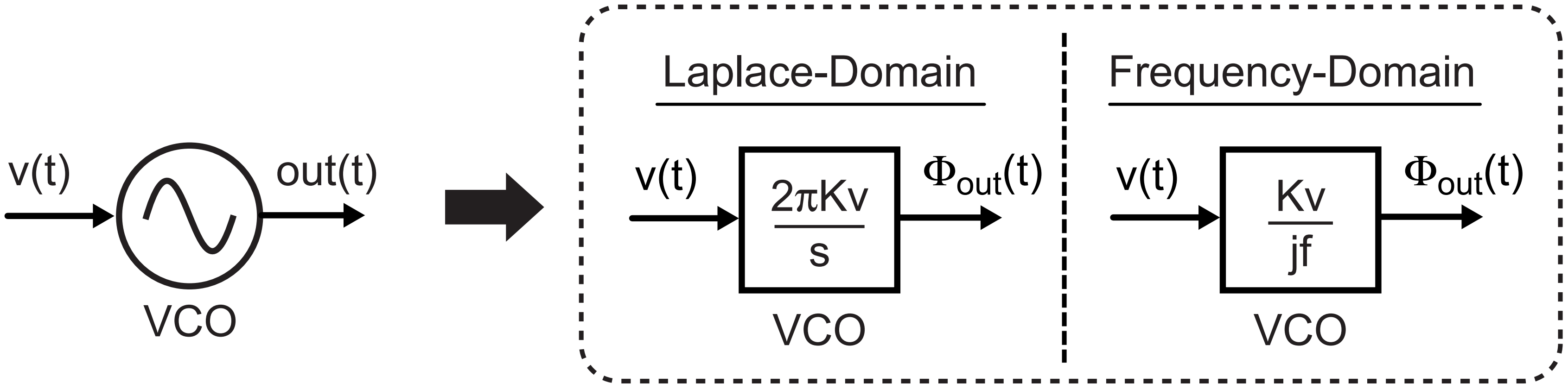


Frequency-Domain Model for VCO

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

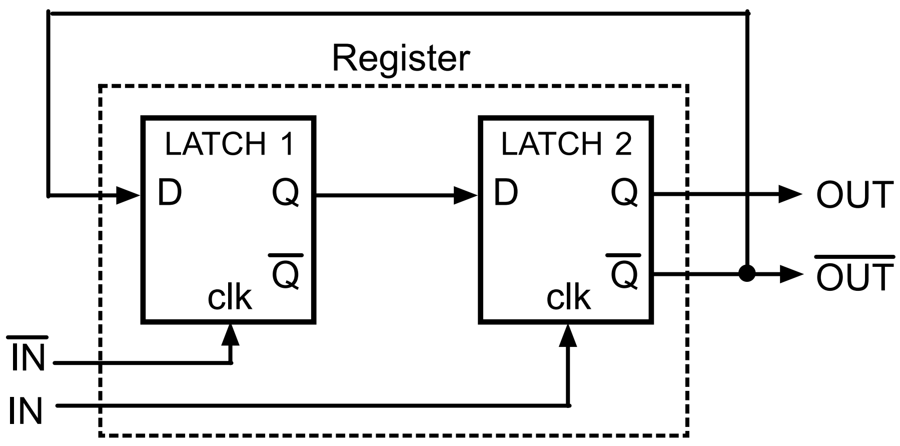
$$\Phi_{out}(t) = \int_{-\infty}^{t} 2\pi K_v v(\tau) d\tau$$

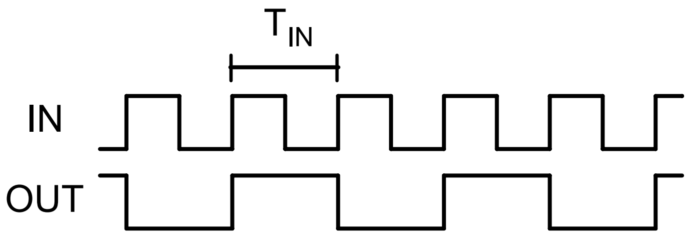
- **Corresponding frequency-domain model**



Frequency Dividers

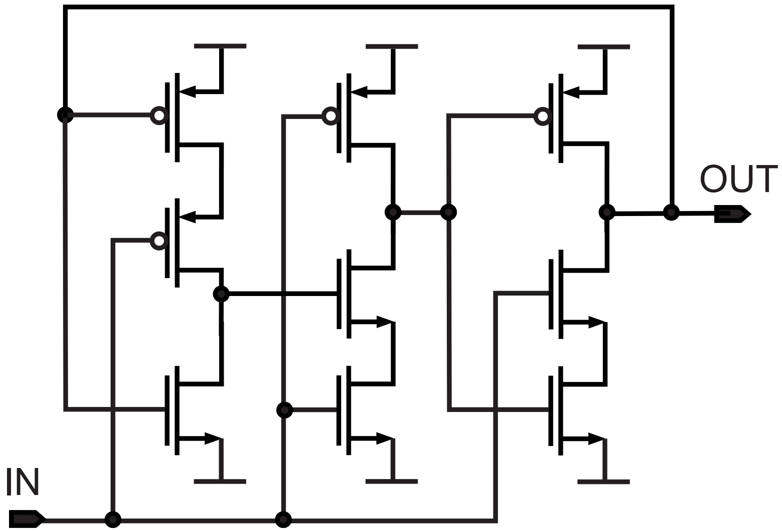
Divide-by-2 Circuit (Johnson Counter)

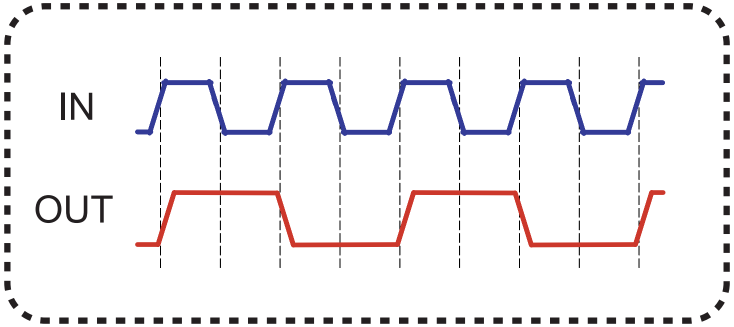




- **Achieves frequency division by clocking two latches (i.e., a register) in negative feedback**
- **Latches may be implemented in various ways according to speed/power requirements**

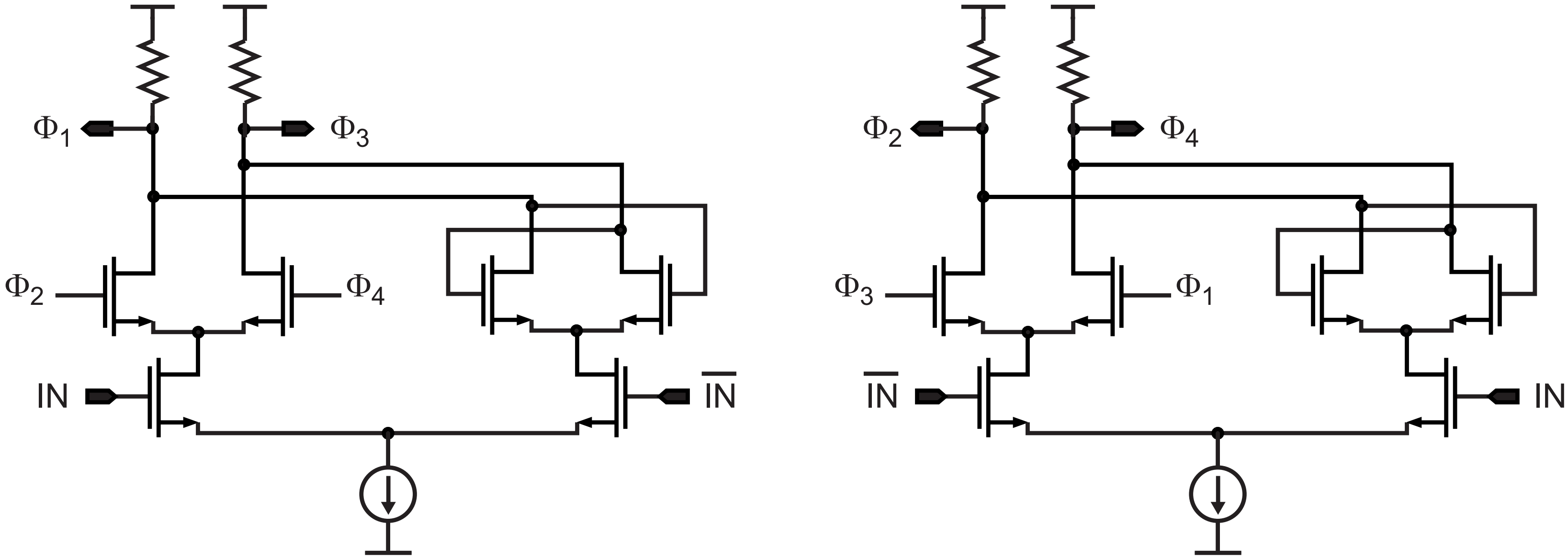
Divide-by-2 Using a TSPC register





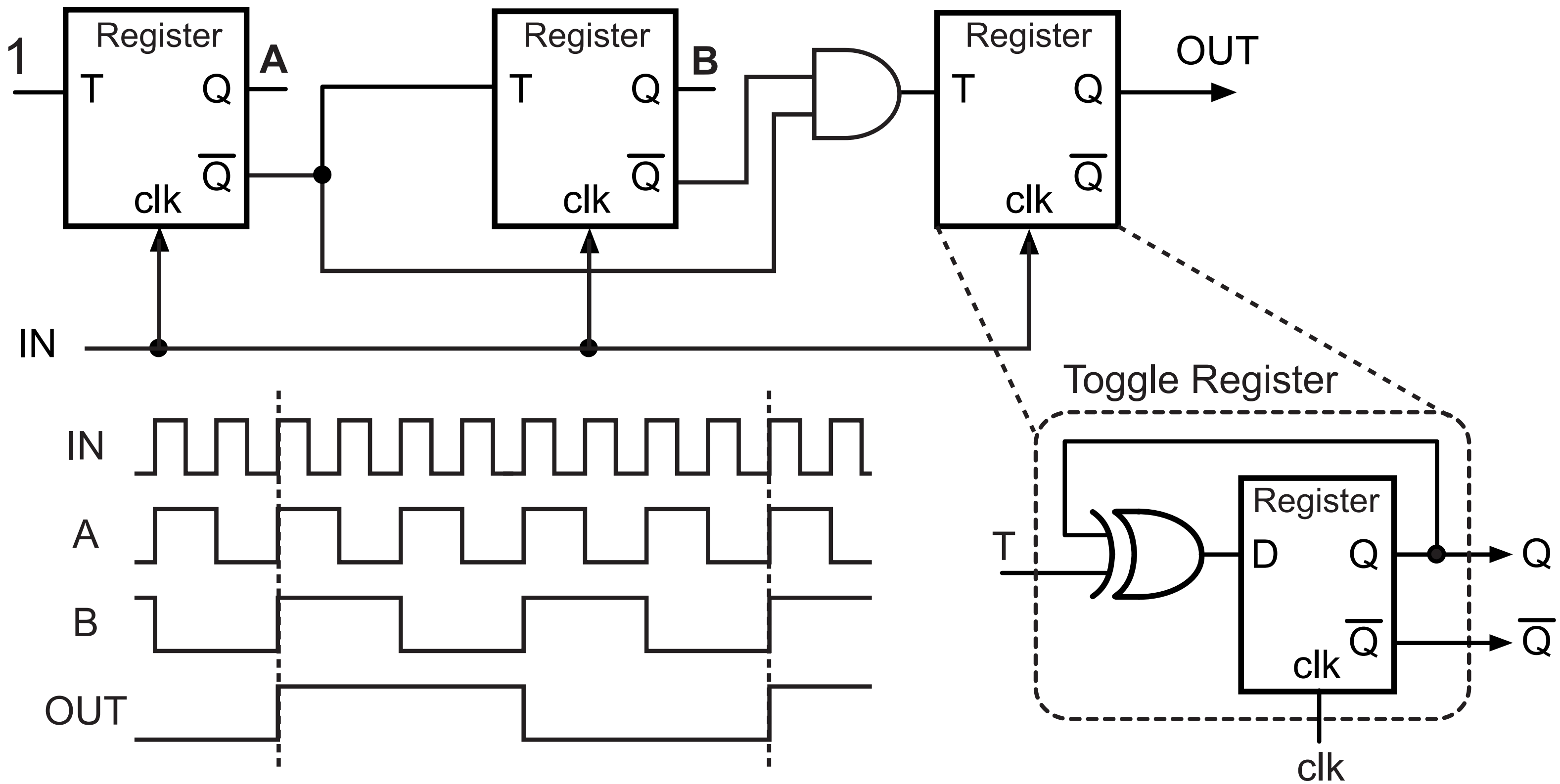
- **Advantages**
  - **Reasonably fast, compact size**
  - **No static power dissipation, differential clock not required**
- **Disadvantages**
  - **Slowed down by stacked PMOS, signals goes through three gates per cycle**
  - **Requires full swing input clock signal**

Divide-by-2 Using SCL (also called CML) Latches



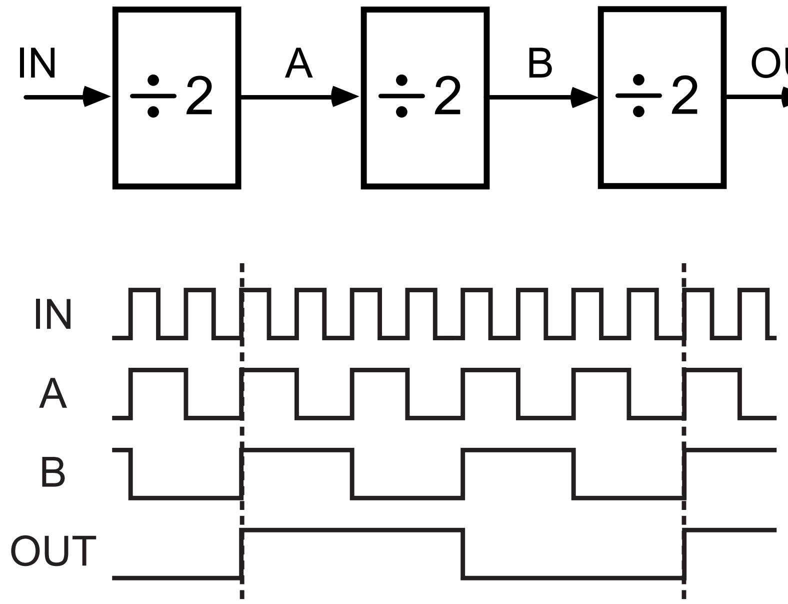
- **Advantage**
  - **Very fast due to small swing and absence of PMOS devices**
    - Additional speedup can be obtained by using inductors
- **Disadvantages**
  - **High power, large area relative to TSPC**
  - **Differential signals required**
  - **Biasing sources required**

Creating Higher Divide Values (Synchronous Approach)



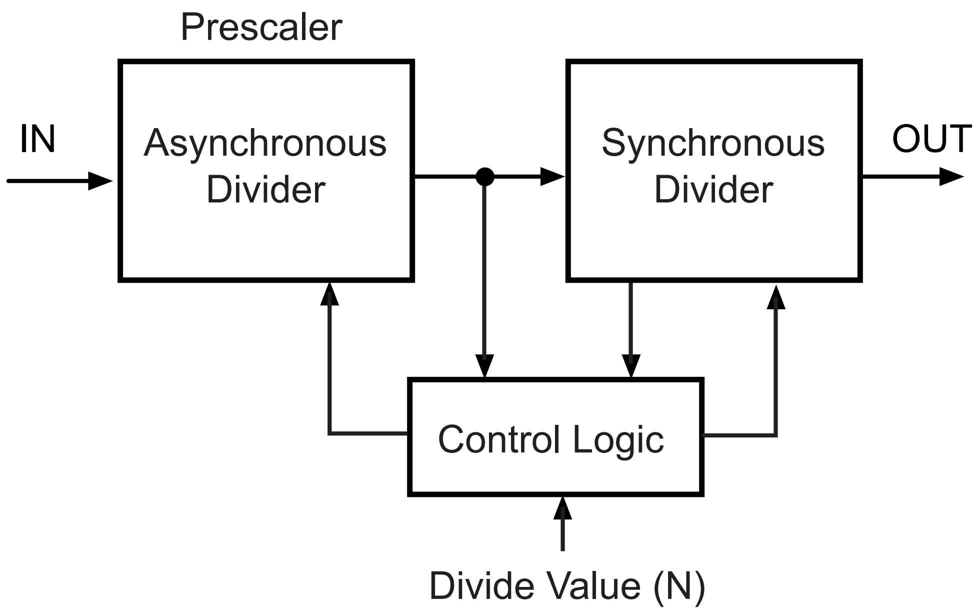
- Cascades toggle registers and logic to perform division
  - Advantage: low jitter
  - Problems: high power (all registers run at high frequency), high loading on clock (IN signal drives *all* registers)

Creating Higher Divide Values (Asynchronous Approach)



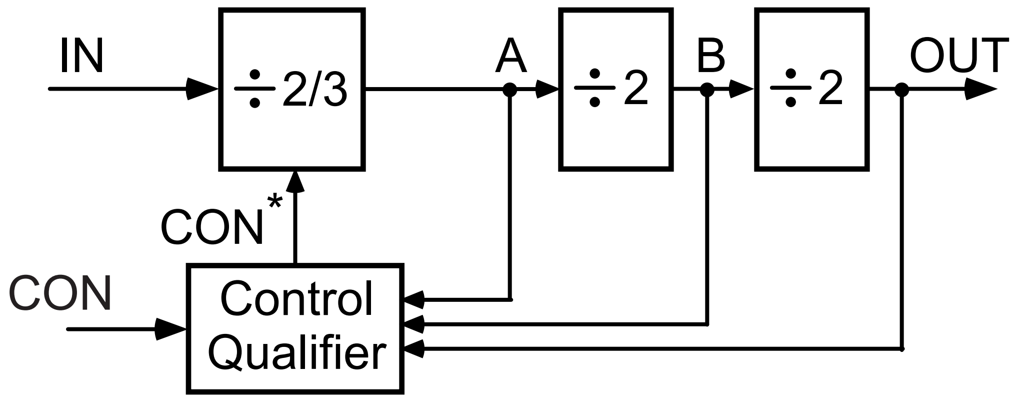
- Higher division achieved by simply cascading divide-by-2 stages
- Advantages over synchronous approach
  - Lower power: each stage runs at a lower frequency, allowing power to be correspondingly reduced
  - Less loading of input: IN signal only drives first stage
- Disadvantage: jitter is larger

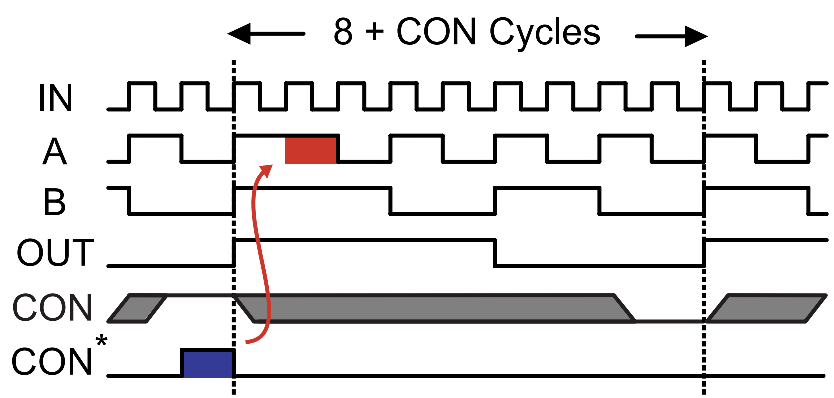
Variable Frequency Division



- **Classical design partitions variable divider into two sections**
  - **Asynchronous section (called a prescaler) is fast**
    - Often supports a limited range of divide values
  - **Synchronous section has no jitter accumulation and a wide range of divide values**
  - **Control logic coordinates sections to produce a wide range of divide values**

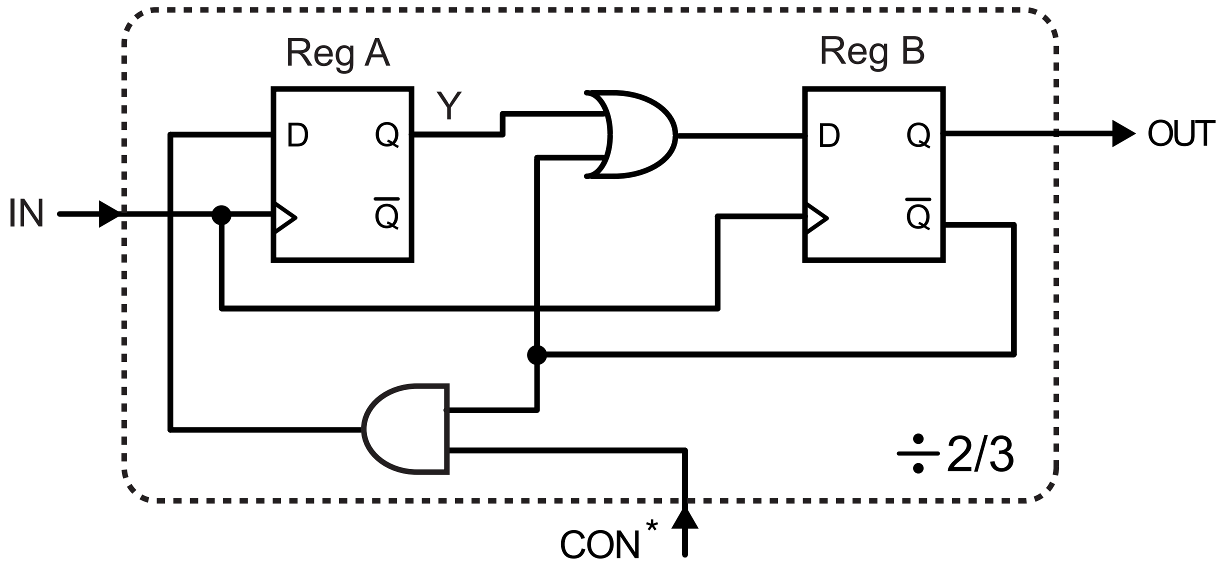
Dual Modulus Prescalers





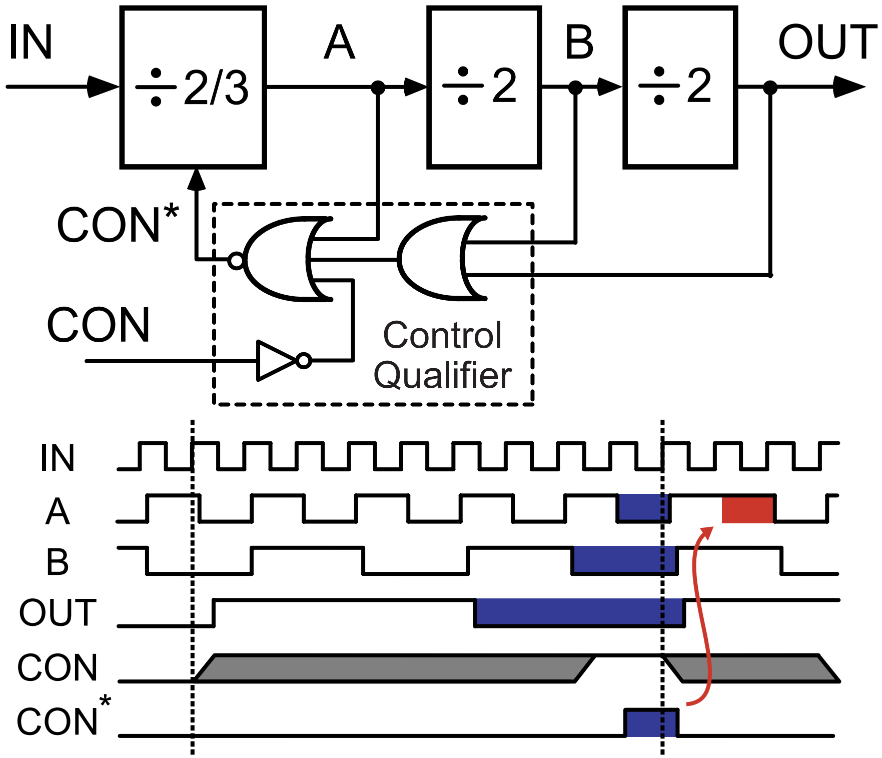
- **Dual modulus design supports two divide values**
  - **In this case, divide-by-8 or 9 according to CON signal**
- **One cycle resolution achieved with front-end "2/3" divider**

Divide-by-2/3 Design (Classical Approach)



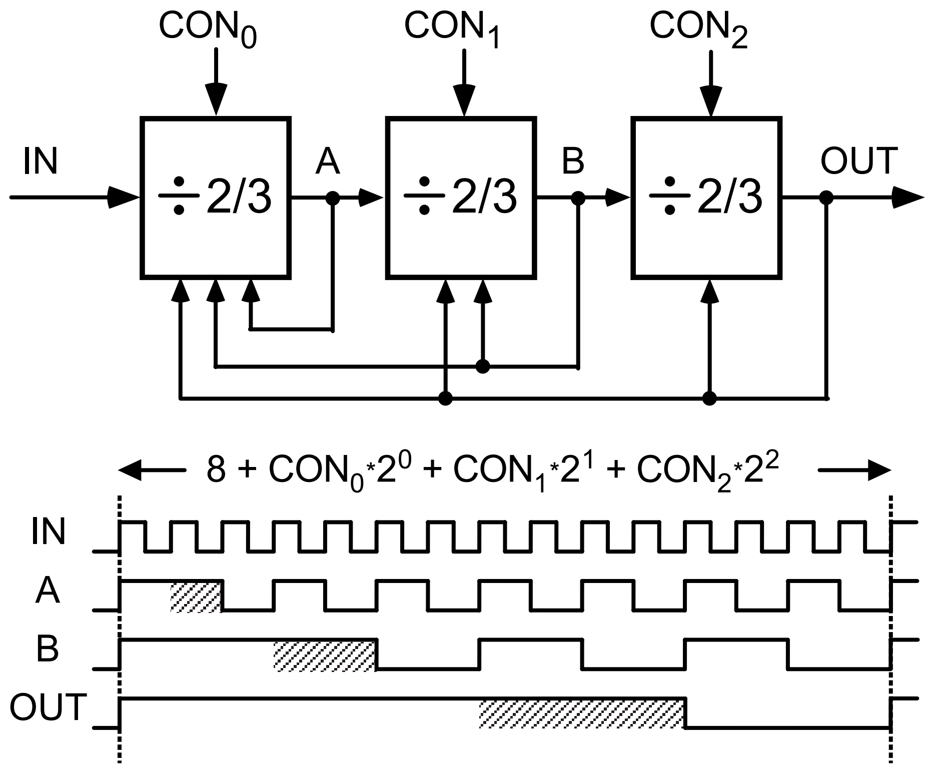
- **Normal mode of operation: CON* = 0 ⇒ Y = 0**
  - **Register B acts as divide-by-2 circuit**
- **Divide-by-3 operation: CON* = 1 ⇒ Y = 1**
  - **Reg B remains high for an extra cycle**
    - Causes Y to be set back to 0 ⇒ Reg B toggles again
    - CON* must be set back to 0 before Reg B toggles to prevent extra pulses from being swallowed

Control Qualifier Design (Classical Approach)



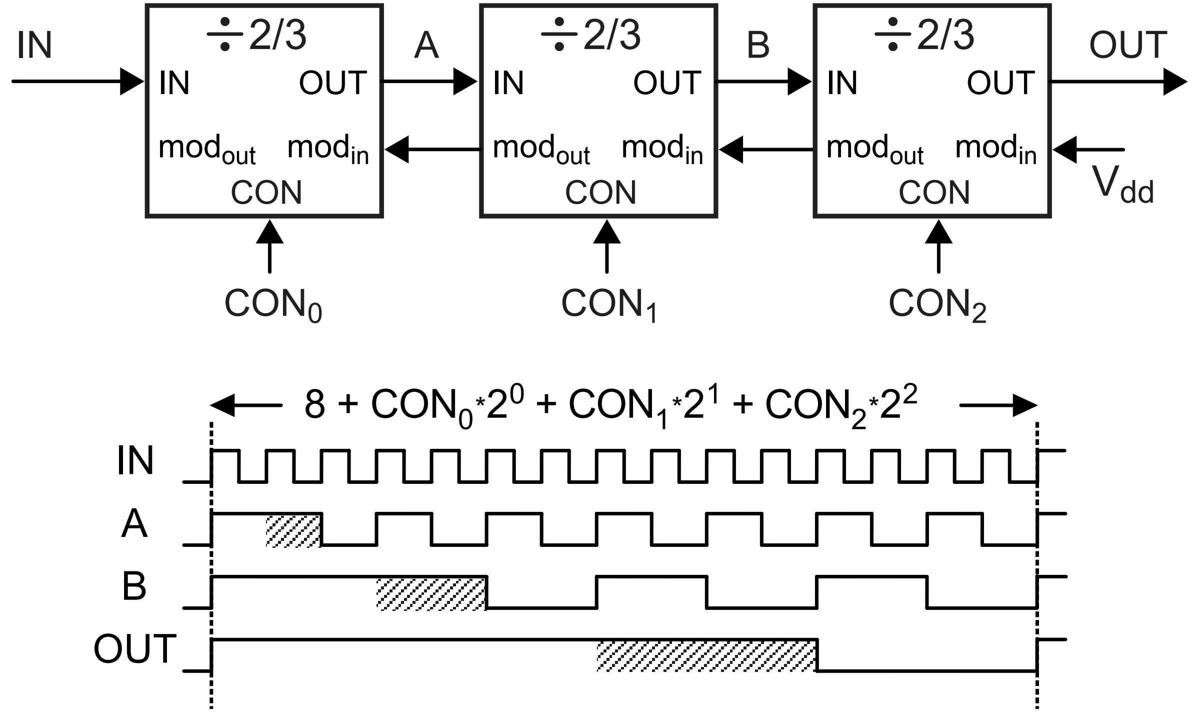
- **Must align CON signal to first "2/3" divider stage**
  - **CON signal is based on logic clocked by divider output**
    - There will be skew between "2/3" divider timing and CON
- **Classical approach cleverly utilizes outputs from each section to "gate" the CON signal to "2/3" divider**

Multi-Modulus Prescalers



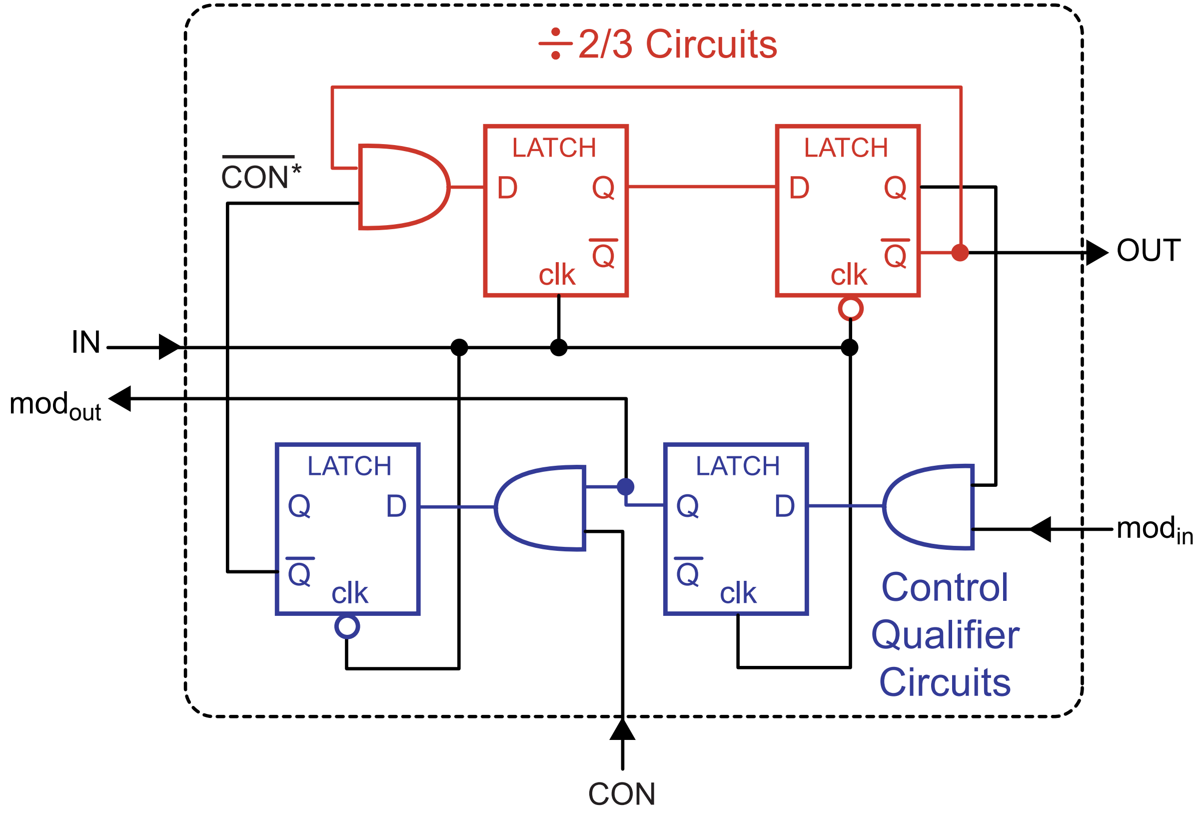
- **Cascaded 2/3 sections achieves a range of 2^n to 2^{n+1}-1**
  - **Above example is 8/ ⋯ /15 divider**
- **Asynchronous design allows high speed and low power operation to be achieved**
  - **Only negative is jitter accumulation**

A More Modular Design



- **Perform control qualification by synchronizing within each stage before passing to previous one**
  - **Compare to previous slide in which all outputs required for qualification of first 2/3 stage**
- **See Vaucher et. al., "A Family of Low-Power Truly Modular Programmable Dividers …", JSSC, July 2000**

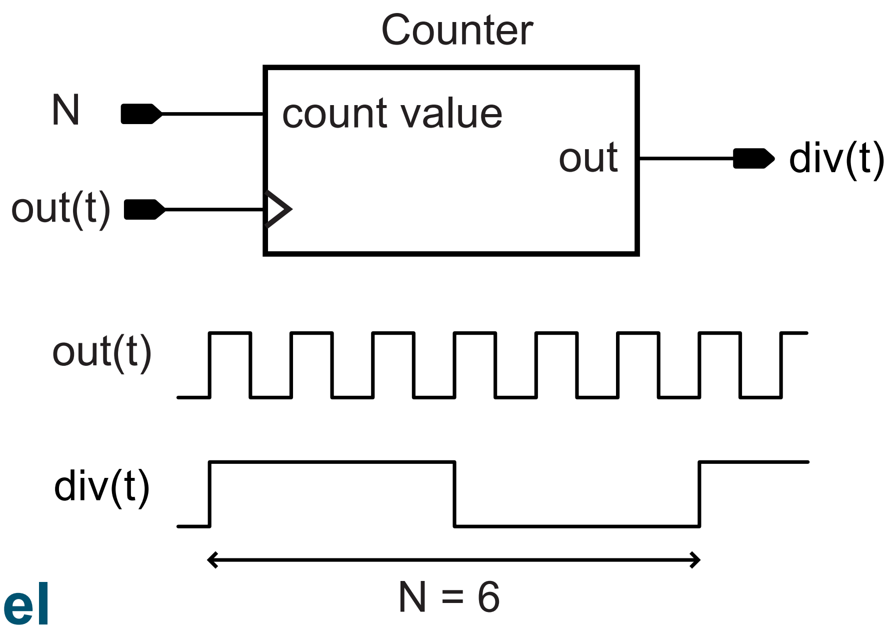
Implementation of 2/3 Sections in Modular Approach



- **Approach has similar complexity to classical design**
  - **Consists of two registers with accompanying logic gates**
- **Cleverly utilizes "gating" register to pass synchronized control qualifying signal to the previous stage**

Divider Modeling

- **Conceptual implementation**



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

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

  - **Phase:**

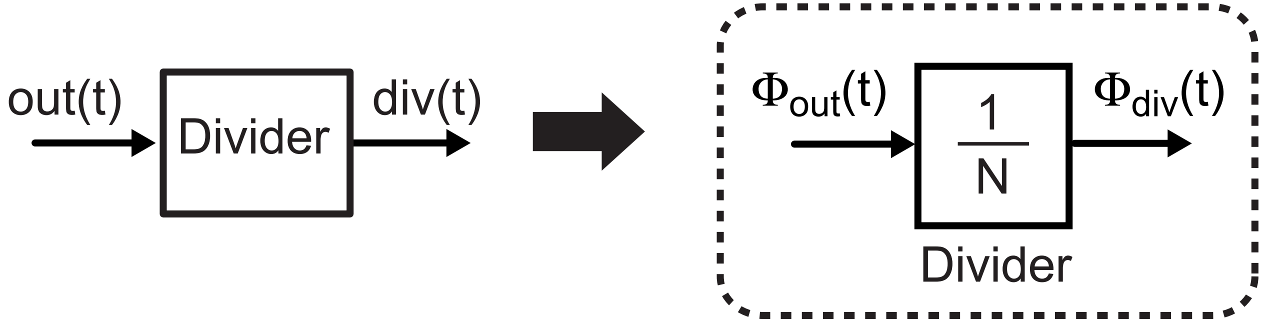
$$\Phi_{div}(t) = \int_{-\infty}^{t} 2\pi \frac{1}{N} F_{out}(\tau) d\tau = \frac{1}{N} \Phi_{out}(t)$$

Frequency-Domain Model of Divider

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

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

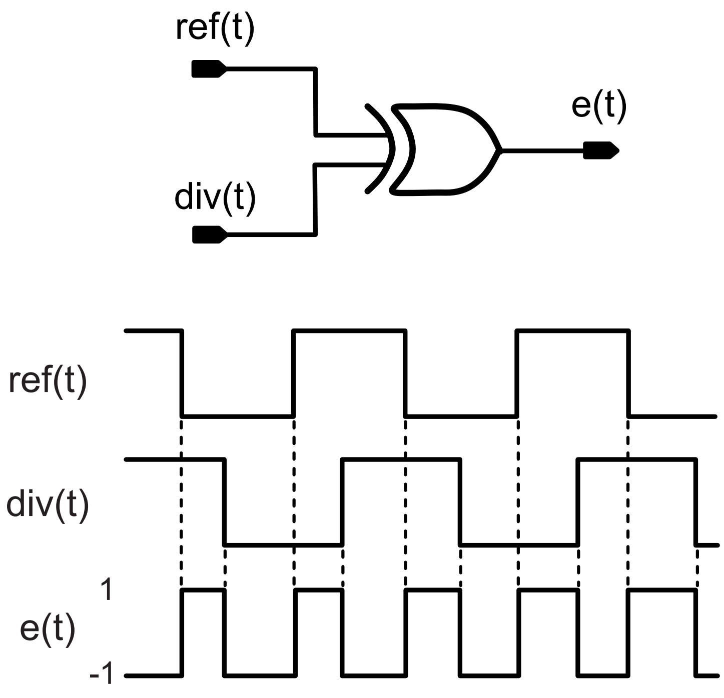
- **Corresponding frequency-domain model (same as Laplace-domain)**



Phase Detection

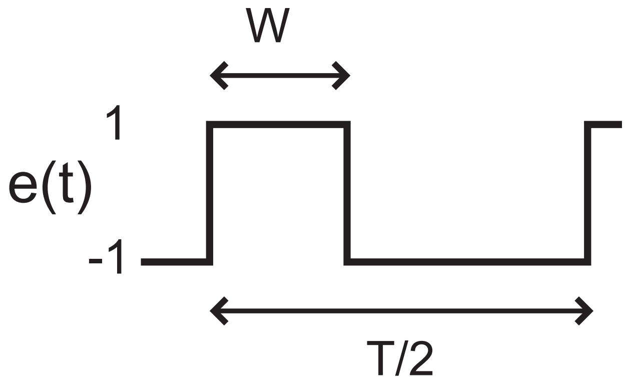
Phase Detector (PD)

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



Modeling of XOR Phase Detector

- Average value of pulses is extracted by loop filter
  - Look at detector output over one cycle:



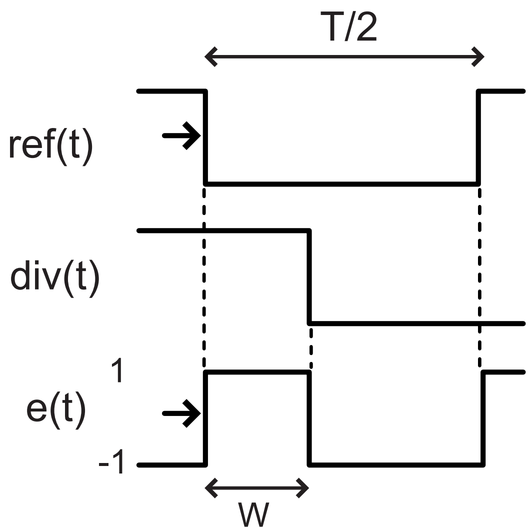
- Equation:

$$avg\{e(t)\} = -1 + 2\frac{W}{T/2}$$

Relate Pulse Width to Phase Error

- **Two cases:**

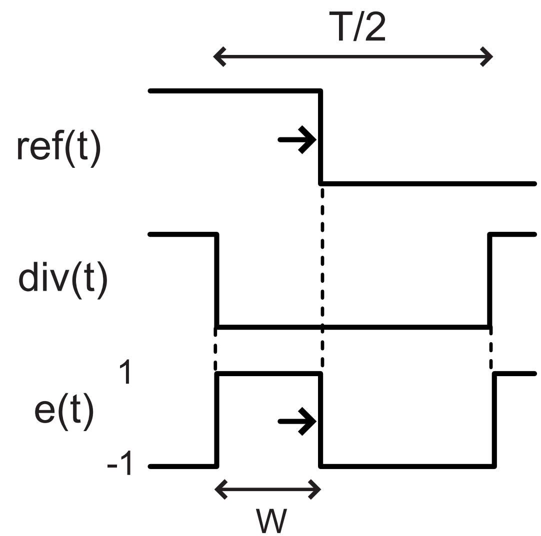
$$-\pi < \Phi_{ref} - \Phi_{div} < 0$$



$$\Downarrow$$

$$W = -\left(\frac{\Phi_{ref} - \Phi_{div}}{\pi}\right) T/2$$

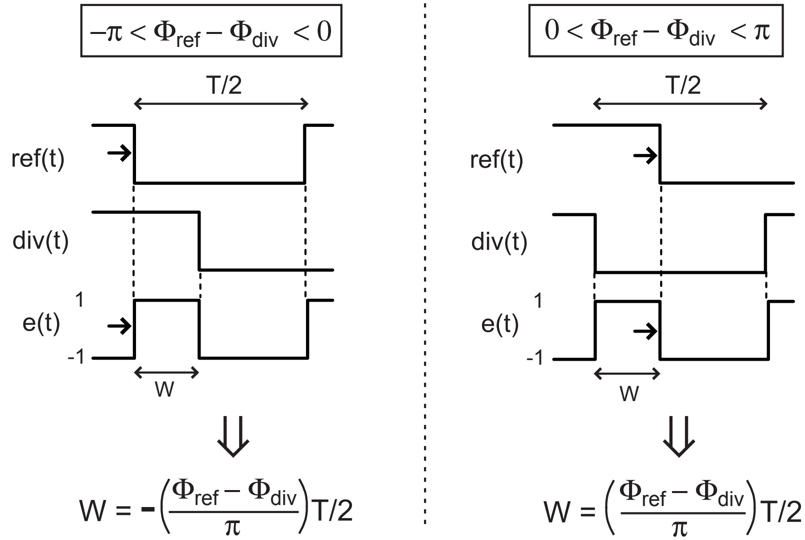
$$0 < \Phi_{ref} - \Phi_{div} < \pi$$

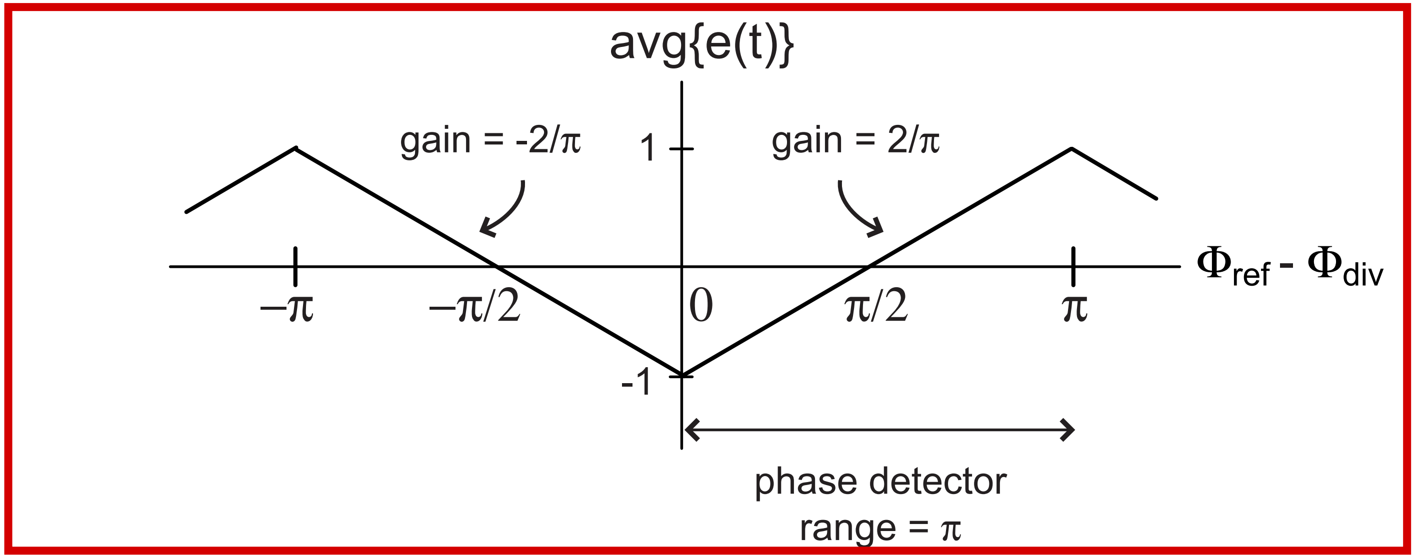


$$\Downarrow$$

$$W = \left(\frac{\Phi_{ref} - \Phi_{div}}{\pi}\right) T/2$$

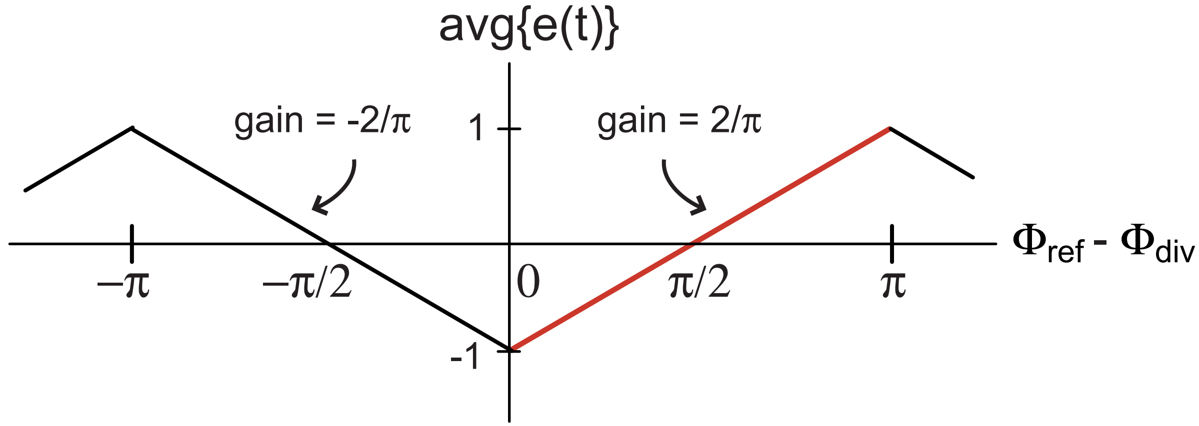
Overall XOR Phase Detector Characteristic



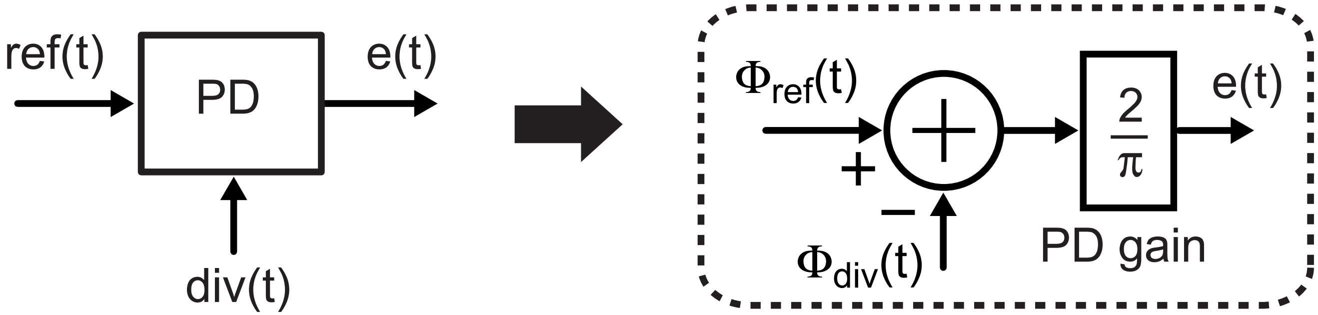


Frequency-Domain Model of XOR Phase Detector

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



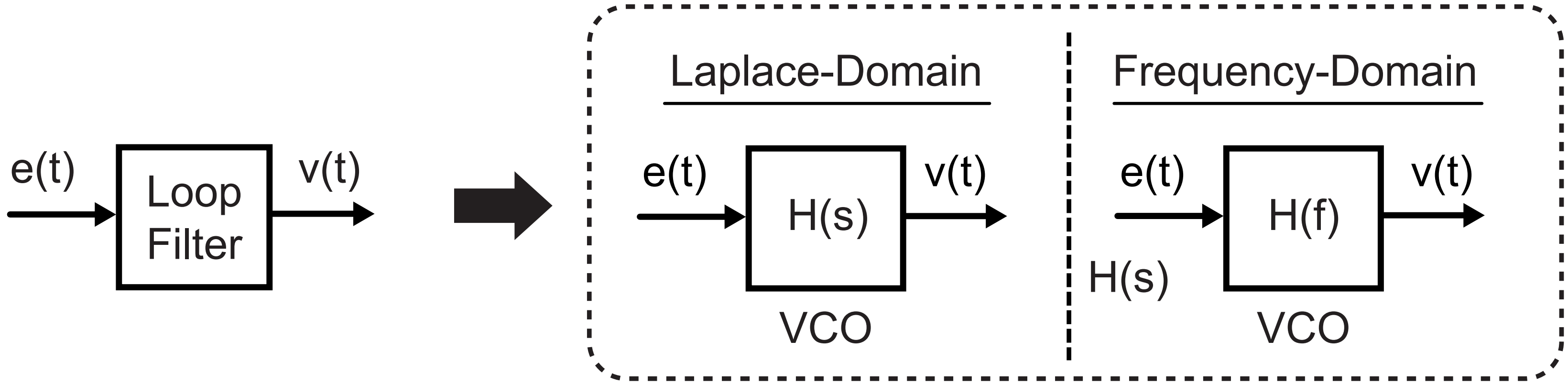
- **Corresponding frequency-domain model**



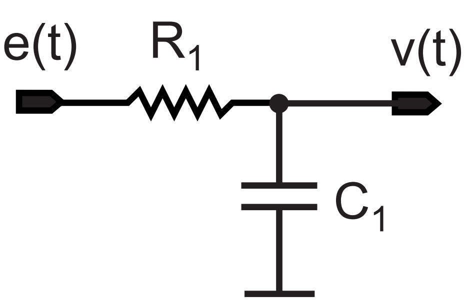
Loop Filter

Loop Filter

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



- **First order example**



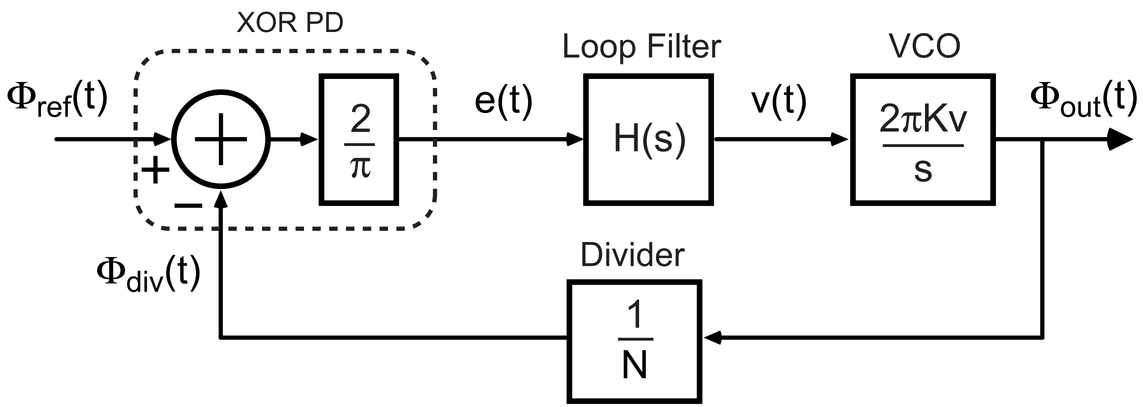
$$\Rightarrow \; H(s) = \frac{1}{1 + sR_1C_1}$$

Integer-N Frequency Synthesizers

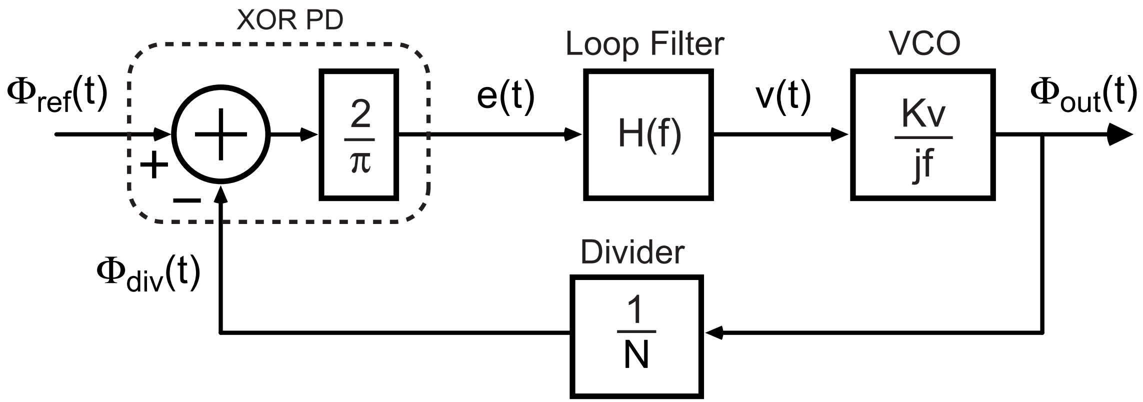
Overall Linearized PLL Frequency-Domain Model

- **Combine models of individual components**

Laplace-Domain Model

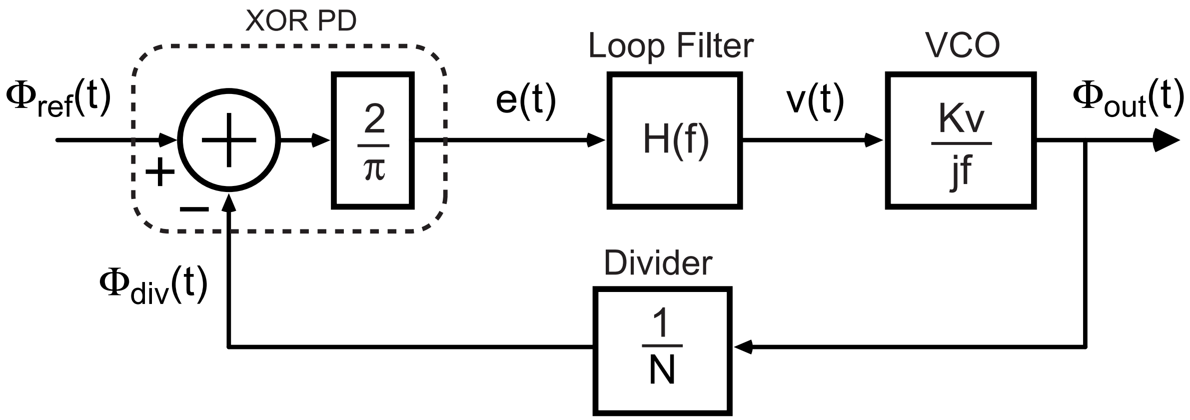


Frequency-Domain Model



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 function (related to closed loop response)**

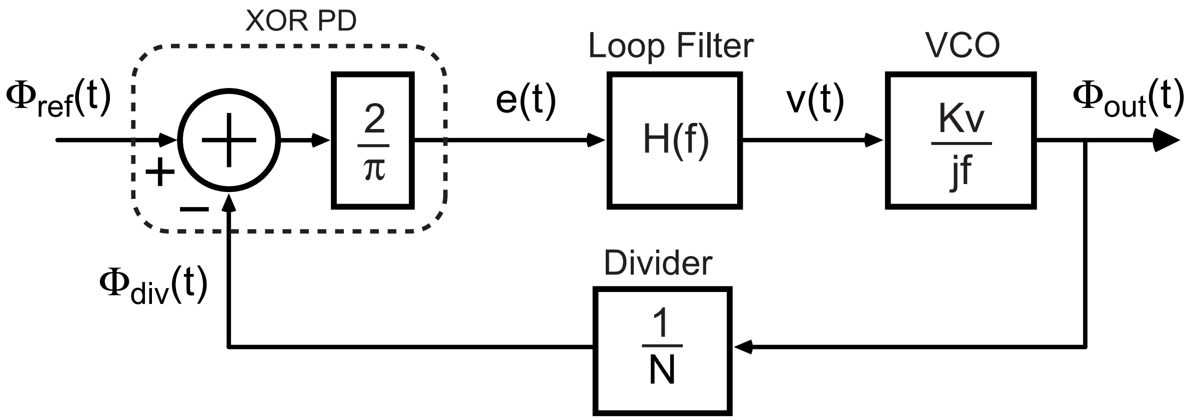
$$G(f) = \frac{A(f)}{1 + A(f)}$$

Classical PLL Transfer Function Design Approach

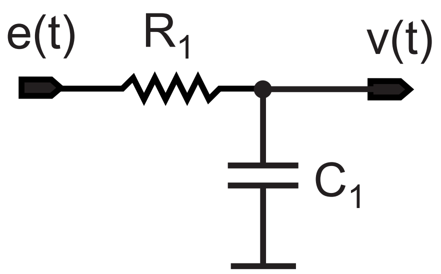
1. Choose an appropriate topology for $H(f)$
   - Usually chosen from a small set of possibilities
2. Choose pole/zero values for $H(f)$ as appropriate for the required filtering of the phase detector output
   - Constraint: set pole/zero locations higher than desired PLL bandwidth to allow stable dynamics 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**

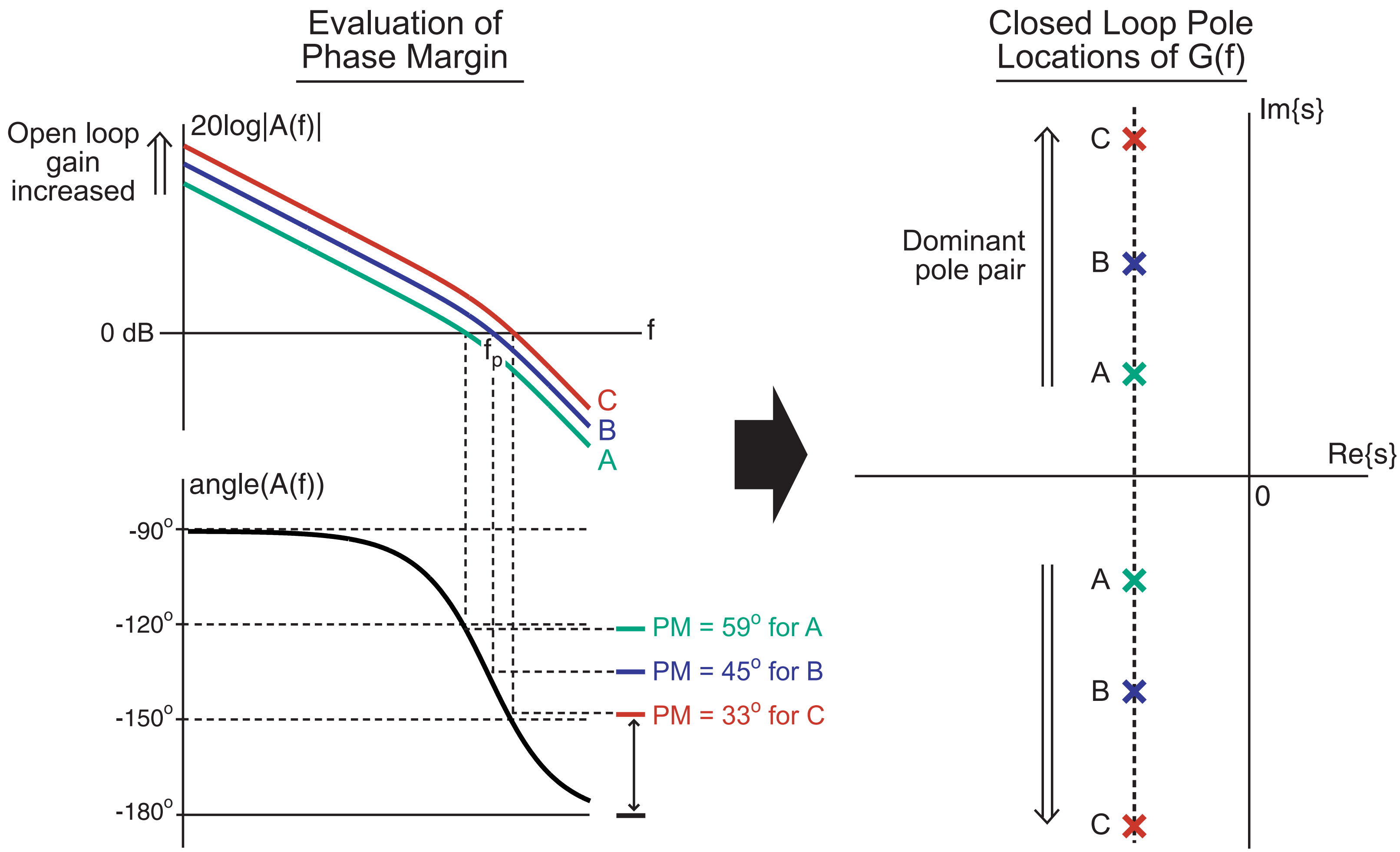


- **Loop filter**



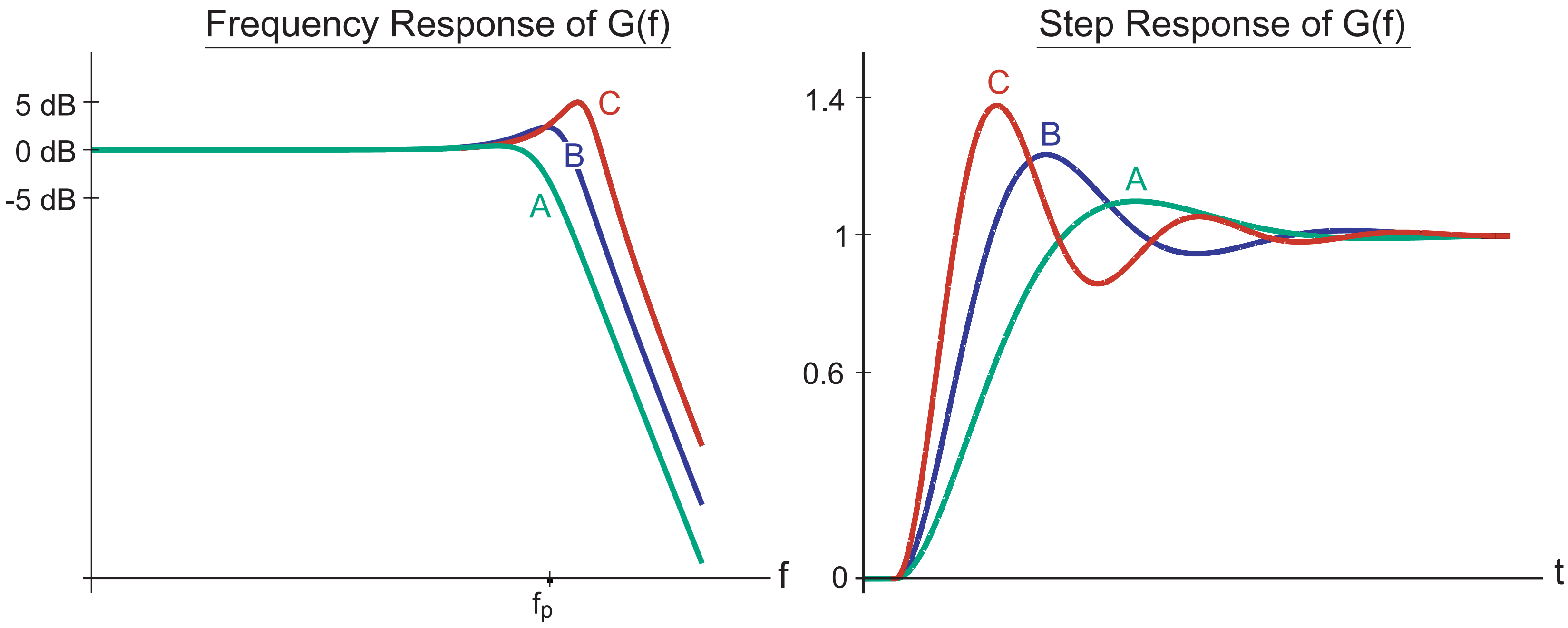
$$\Rightarrow \quad H(f) = \frac{1}{1 + jf/f_p}$$

Closed Loop Poles Versus Open Loop Gain



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

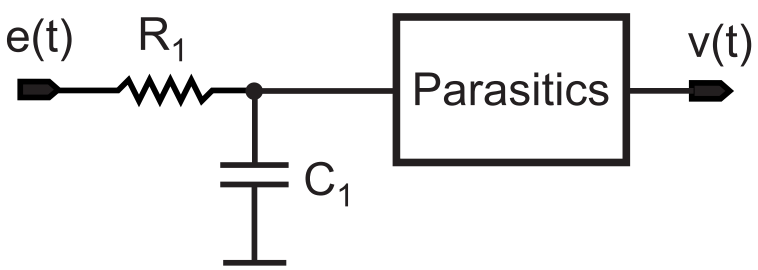
Corresponding Closed Loop Response



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

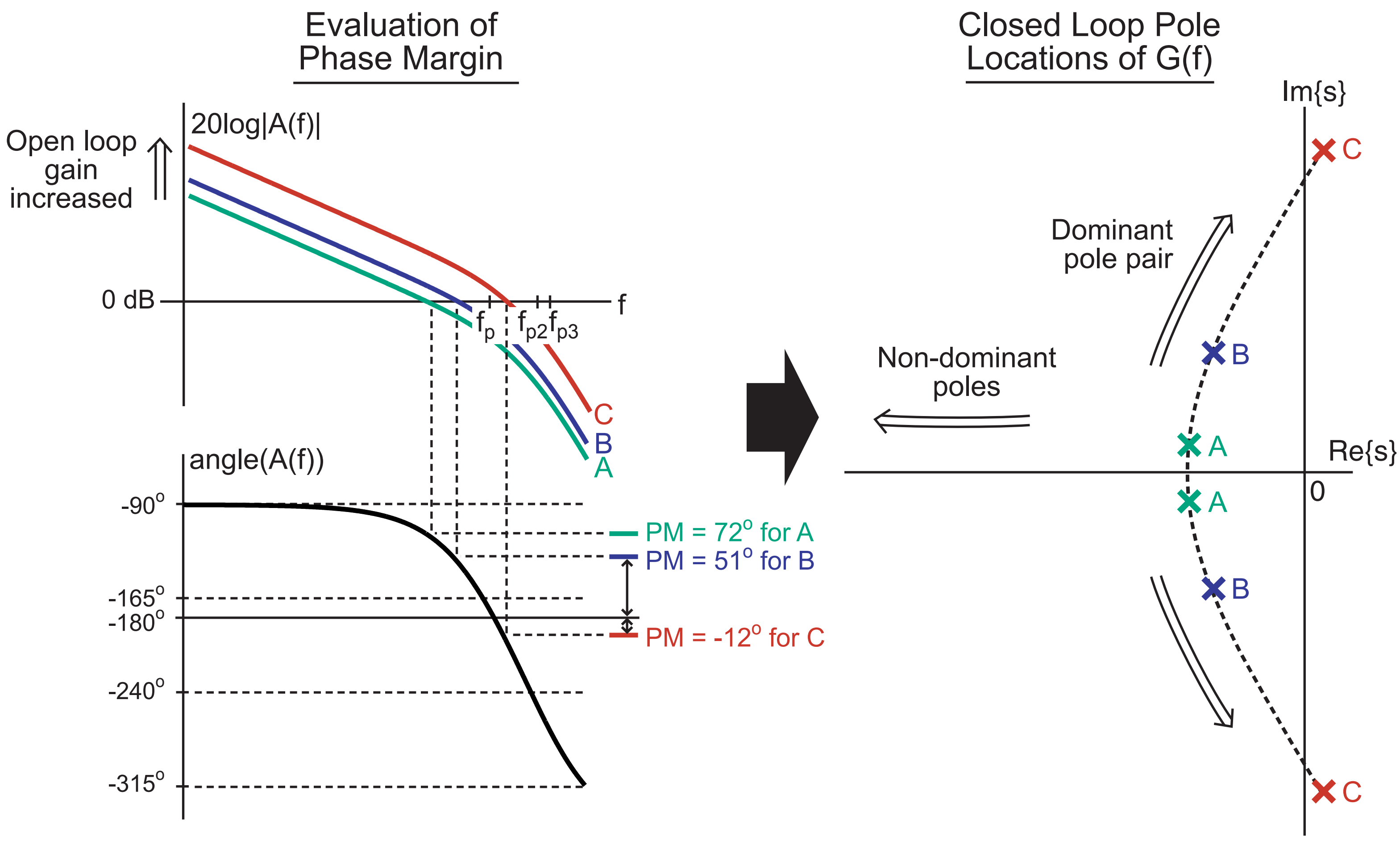
The Impact of Parasitic Poles

- **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**

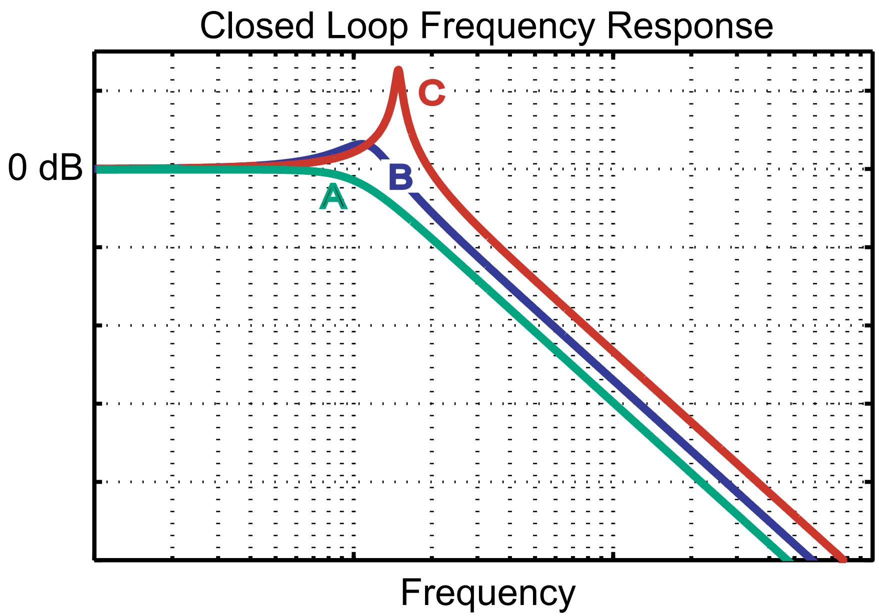


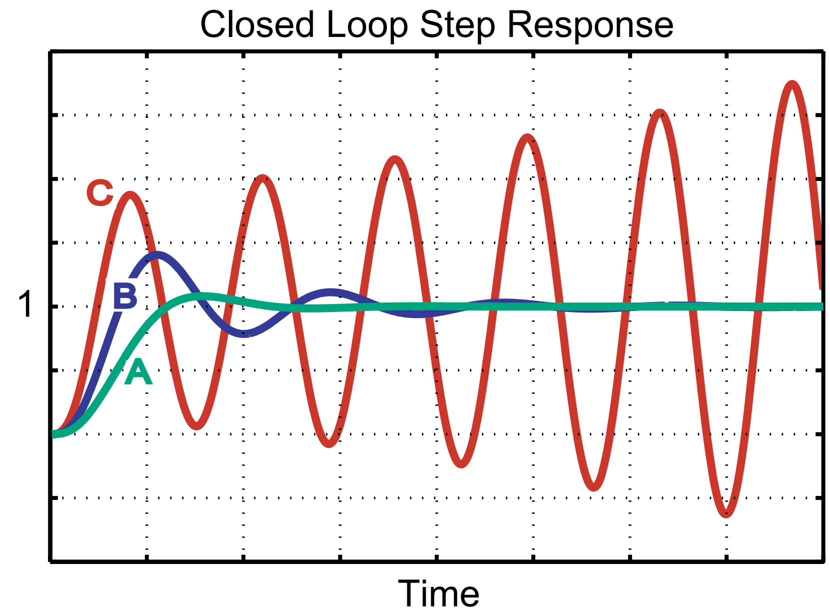
$$\Rightarrow \quad H(f) = \left(\frac{1}{1+jf/f_1}\right)\left(\frac{1}{1+jf/f_2}\right)\left(\frac{1}{1+jf/f_3}\right)$$

Closed Loop Poles Versus Open Loop Gain



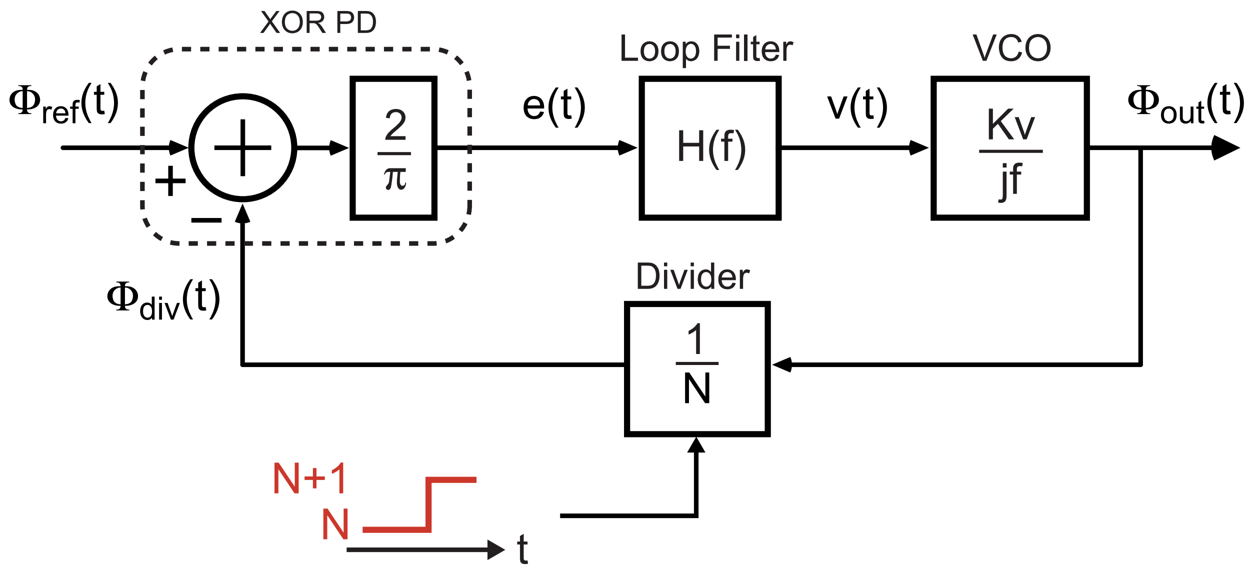
Corresponding Closed Loop Response





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

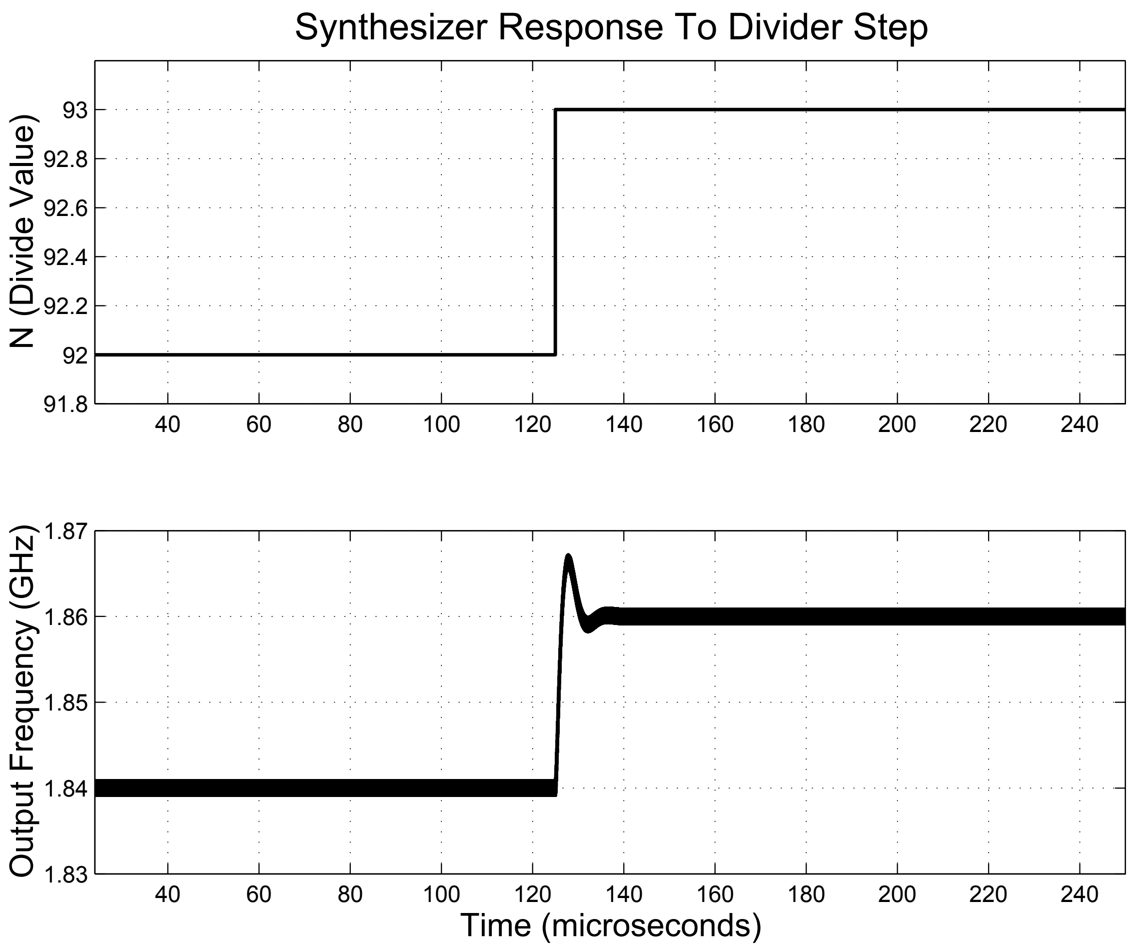
Response of PLL to Divide Value Changes



- **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

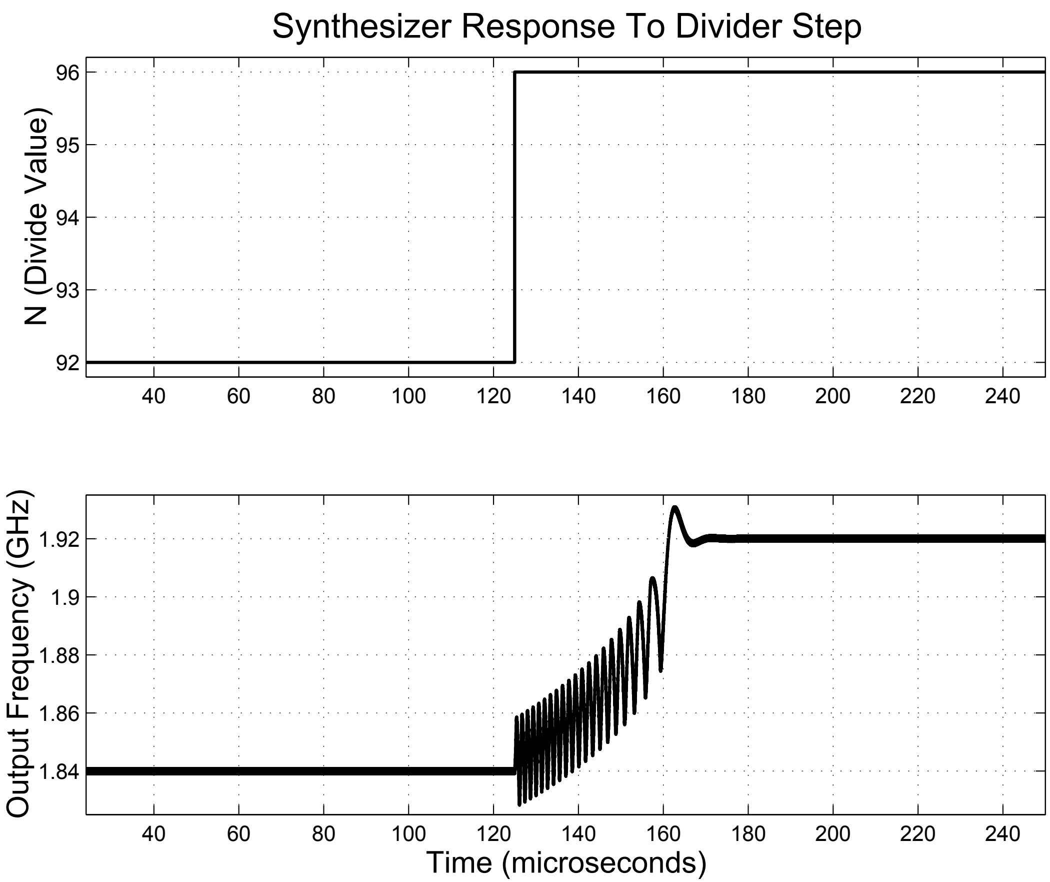
- **Example: Change divide value by one**



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

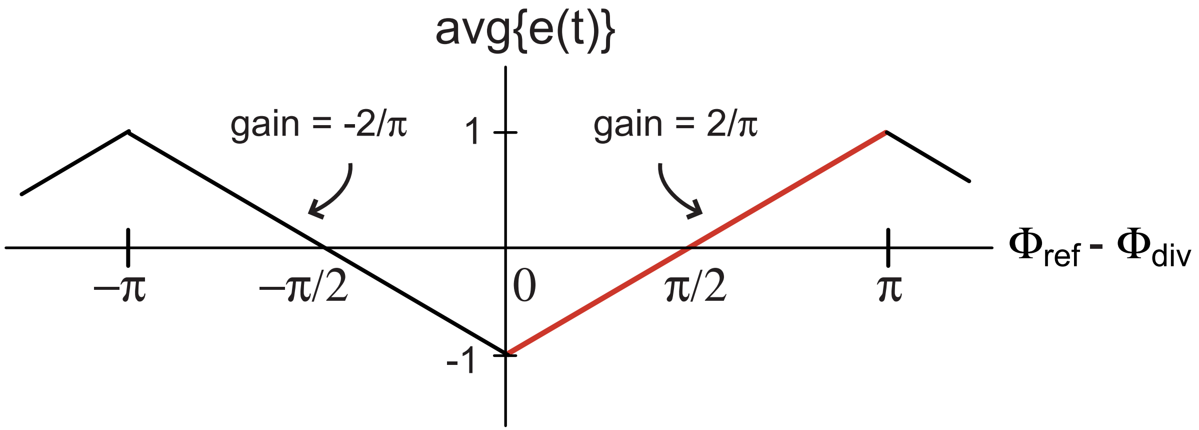
What Happens with Large Divide Value Variations?

- **PLL temporarily loses frequency lock (cycle slipping occurs)**



  - **Why does this happen?**

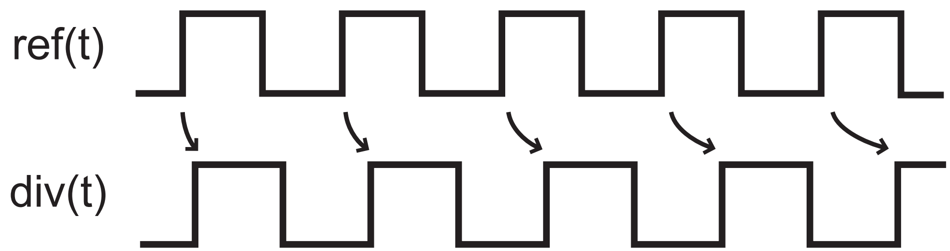
Recall Phase Detector Characteristic



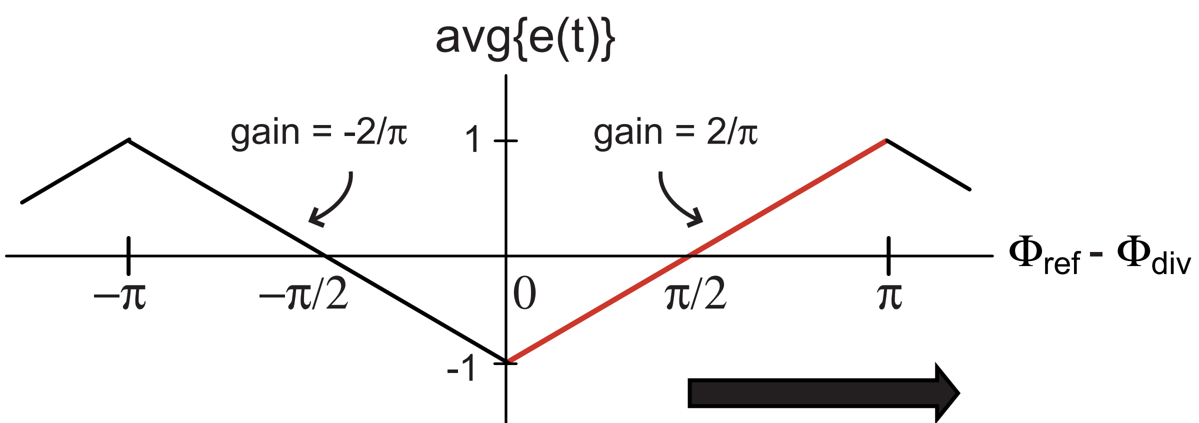
- **To simplify modeling, we assumed that we always operated in a confined phase range (0 to π)**
  - **Led to a simple PD model**
- **Large perturbations knock us out of that confined phase range**
  - **PD behavior varies depending on the phase range it happens to be in**

Cycle Slipping

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

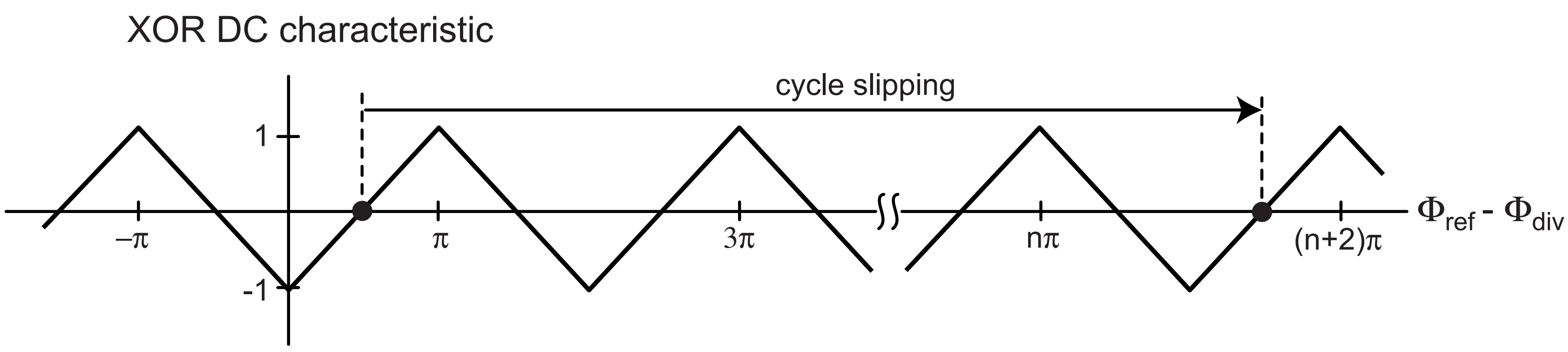


- **Resulting ramp in phase causes PD characteristic to be swept across its different regions (cycle slipping)**



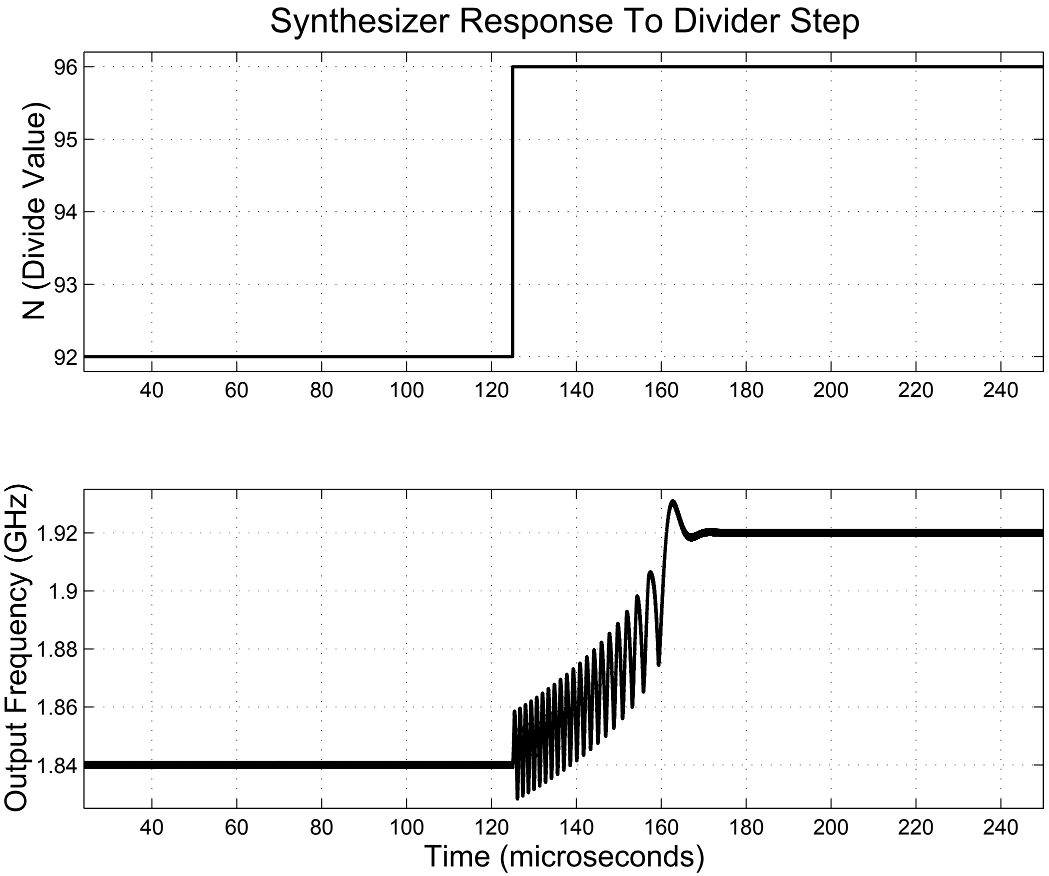
Impact of Cycle Slipping

- **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

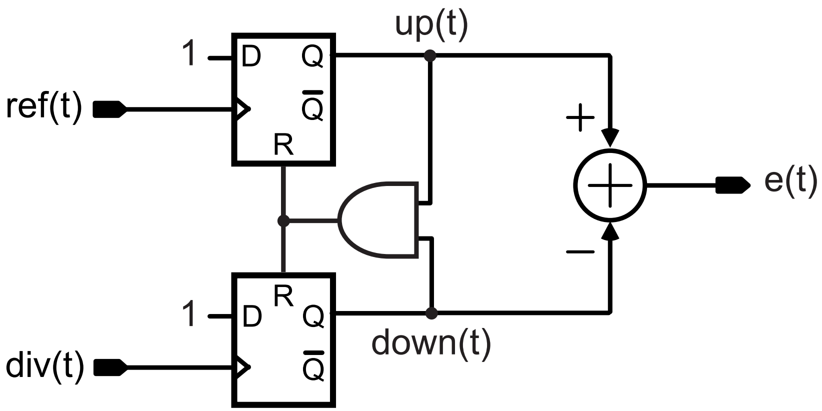
- **PLL output frequency indeed oscillates**
  - **Eventually locks when frequency difference is small enough**

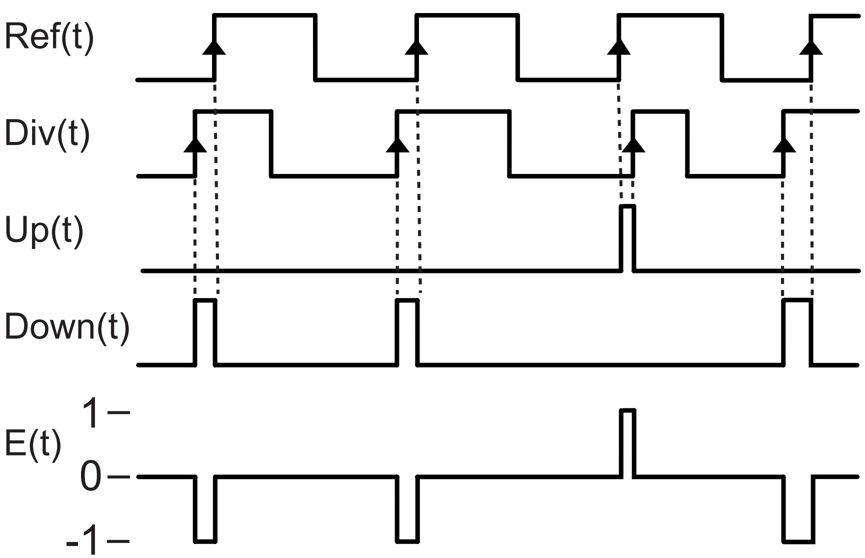


  - **How do we extend the frequency lock-in range?**

Phase Frequency Detectors (PFD)

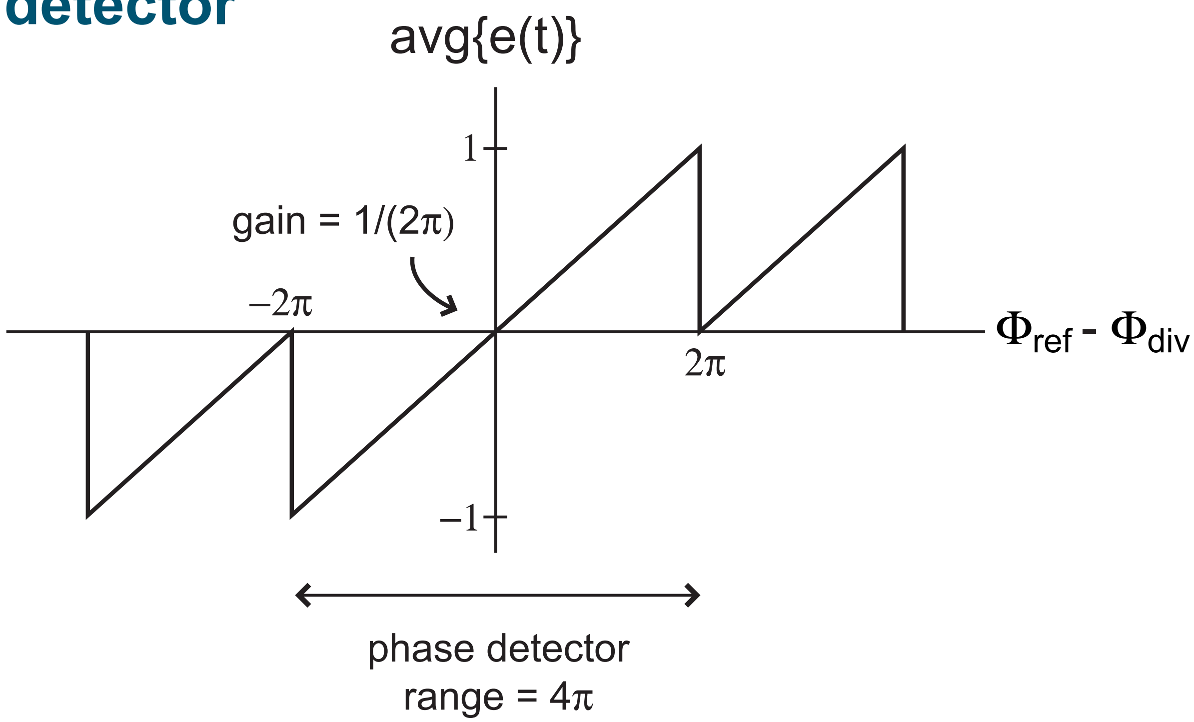
- **Example: Tristate PFD**





Tristate PFD Characteristic

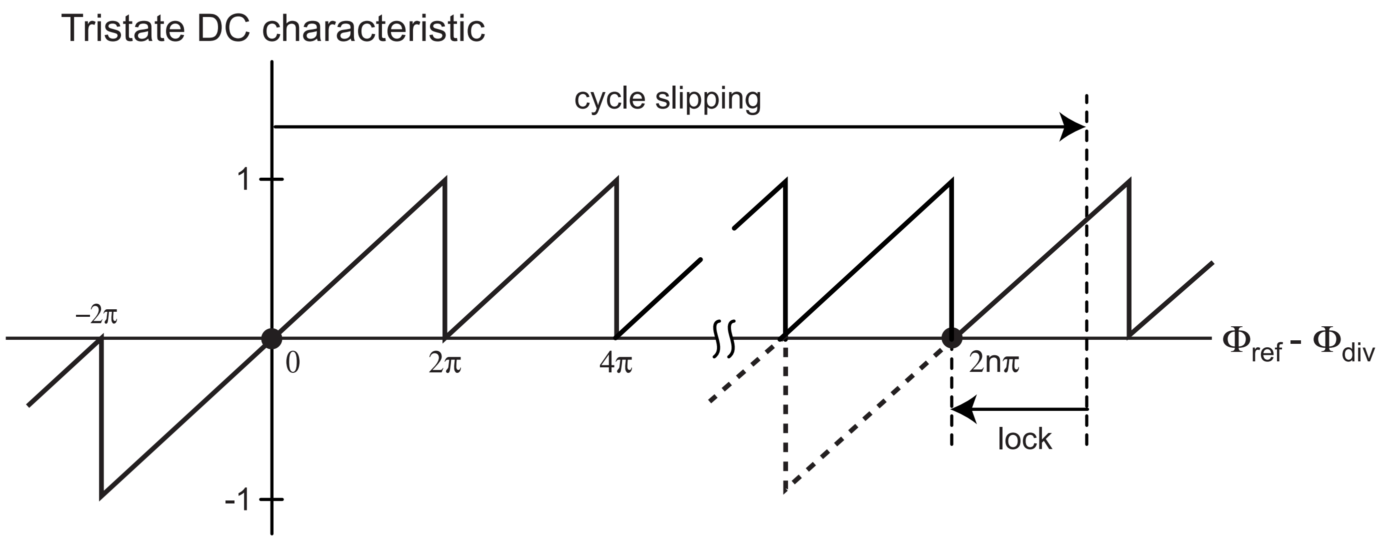
- **Calculate using similar approach as used for XOR phase detector**



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

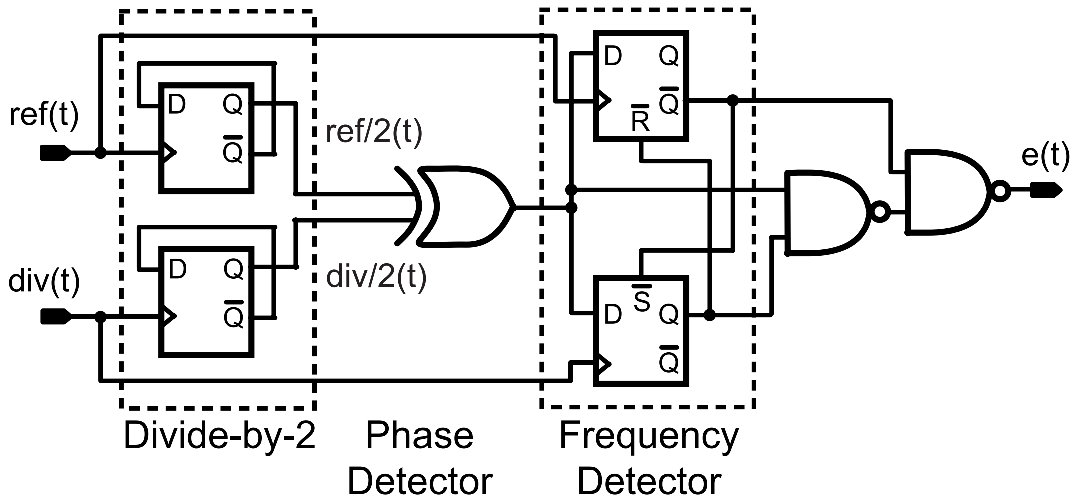
PFD Enables PLL to Always Regain Frequency Lock

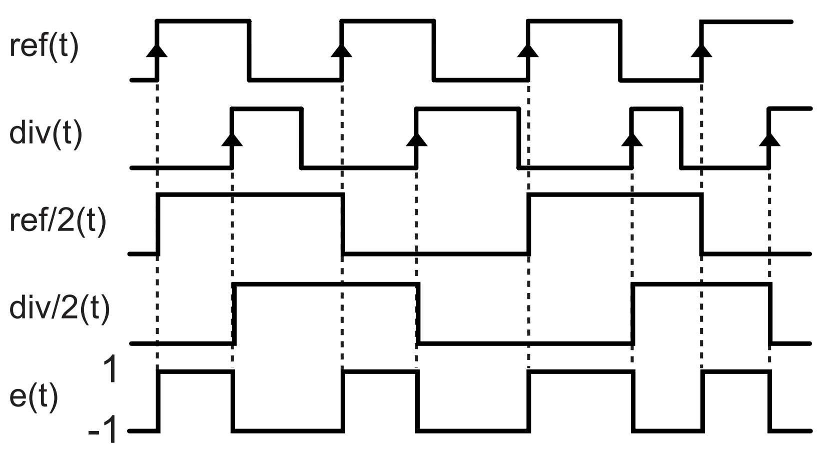
- **Asymmetric phase error characteristic allows positive frequency differences to be distinguished from negative frequency differences**
  - **Average value is now positive or negative according to sign of frequency offset**
  - **PLL will always relock**



Another PFD Structure

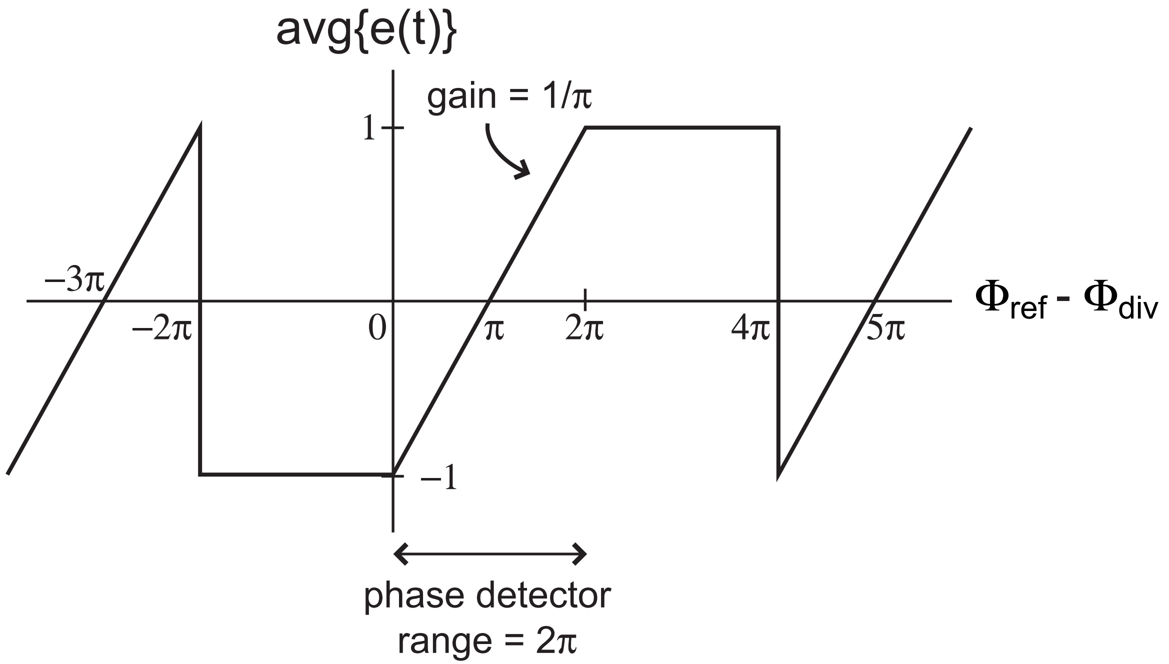
- **XOR-based PFD**





XOR-based PFD Characteristic

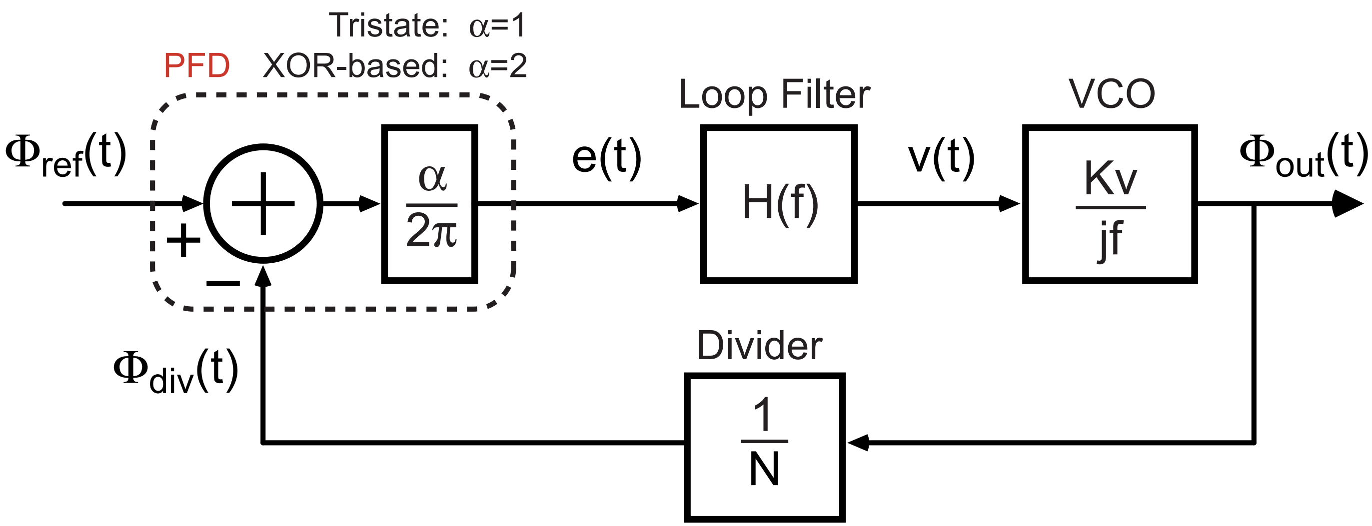
- **Calculate using similar approach as used for XOR phase detector**



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

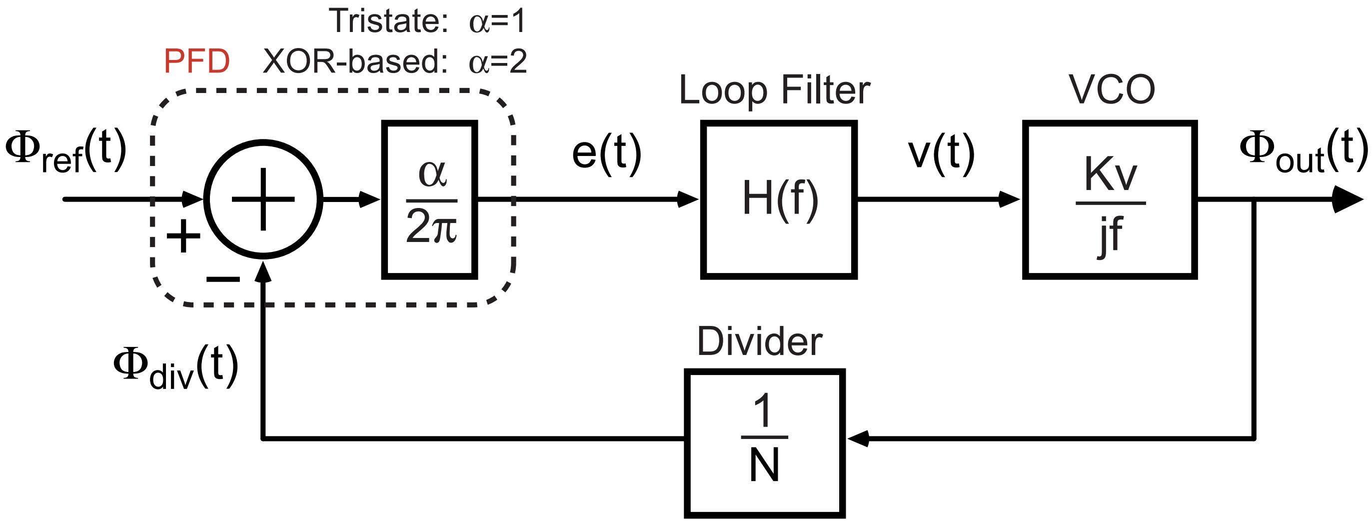
Linearized PLL Model With PFD Structures

- 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



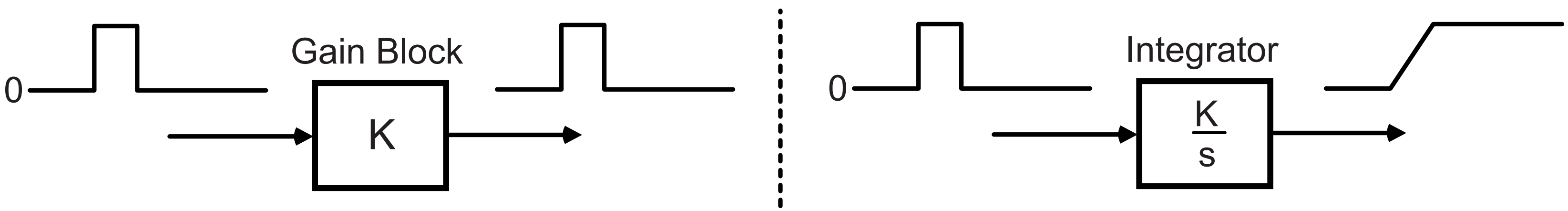
Type I versus Type II PLL Implementations

- **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**

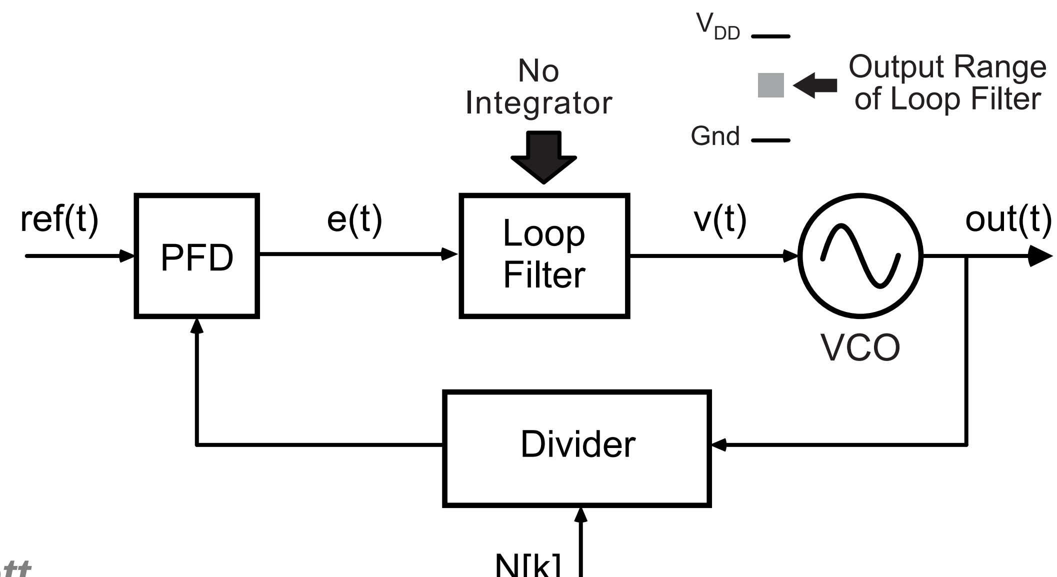


VCO Input Range Issue for Type I PLL Implementations

- **DC output range of gain block versus integrator**

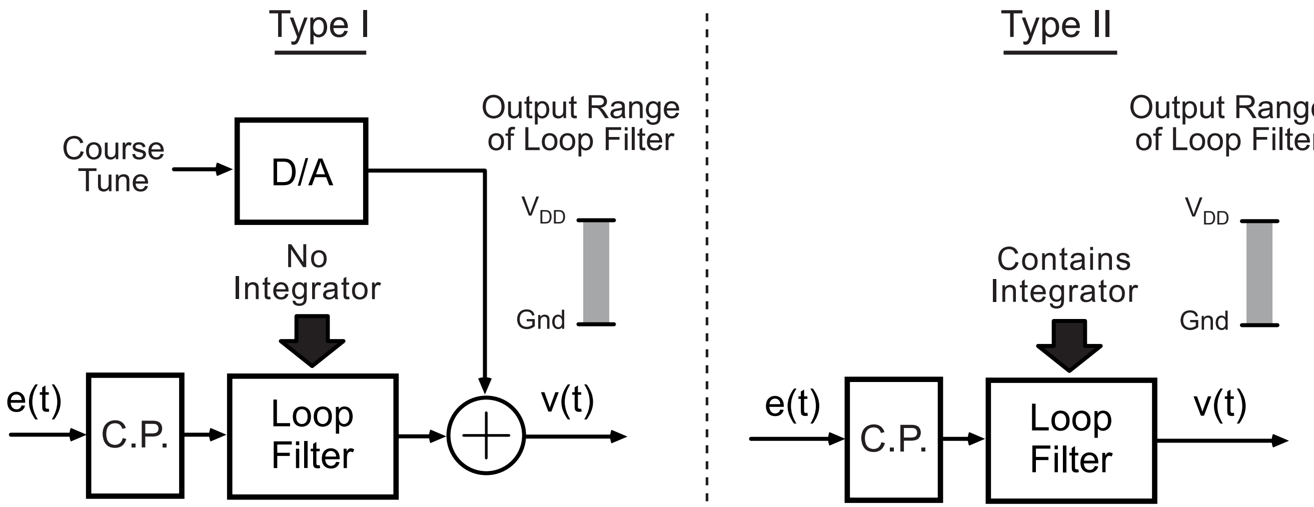


- **Issue: DC gain of loop filter often small and PFD output range is limited**
  - **Loop filter output fails to cover full input range of VCO**



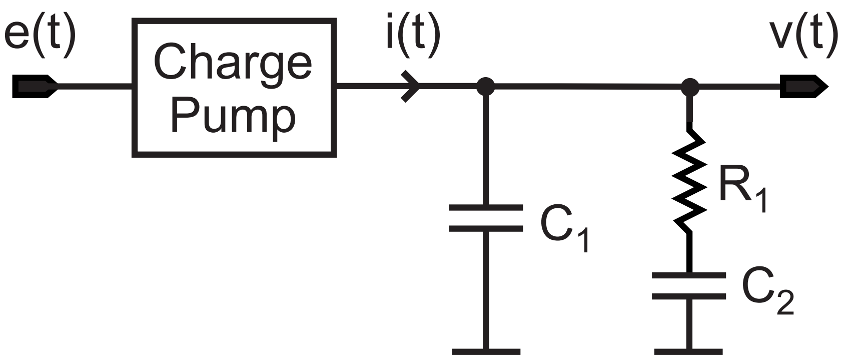
Options for Achieving Full Range Span of VCO

- **Type I**
  - **Add a D/A converter to provide coarse tuning**
    - Adds 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



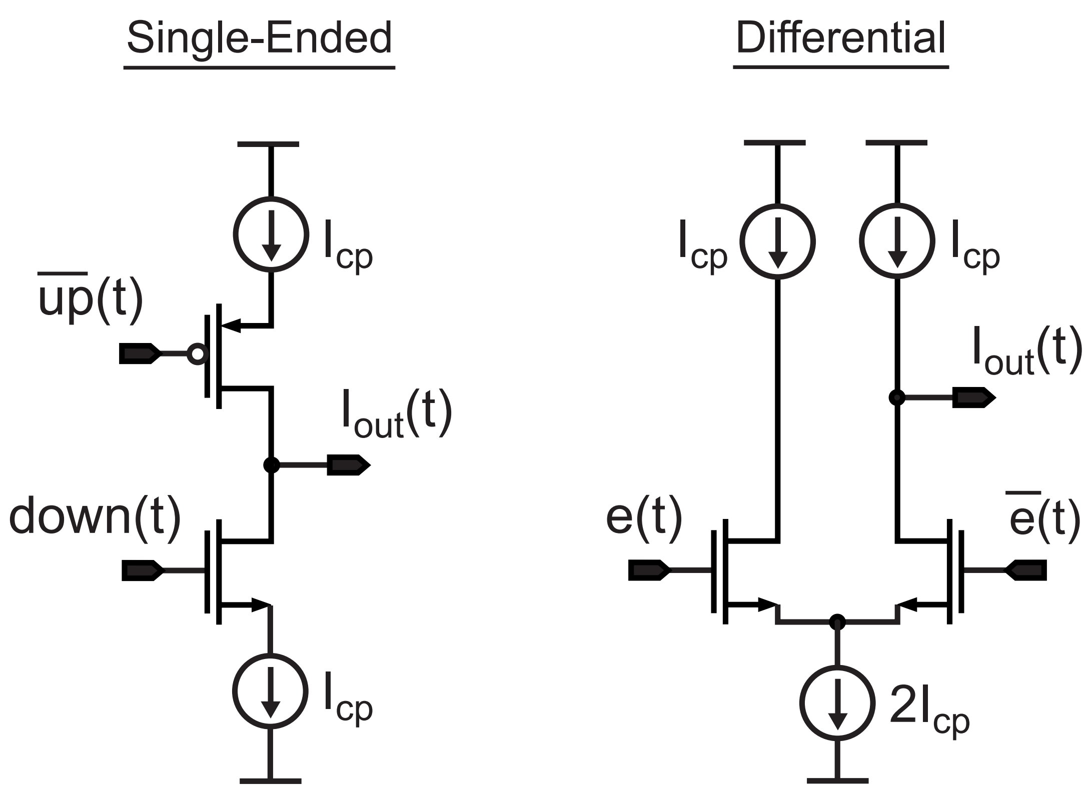
A Common Loop Filter for Type II PLL Implementation

- **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**



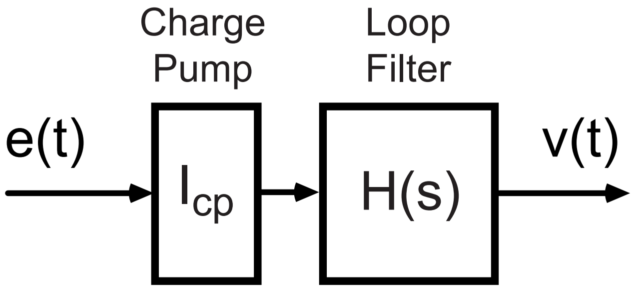
Charge Pump Implementations

- **Switch currents in and out:**



Modeling of Loop Filter/Charge Pump

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



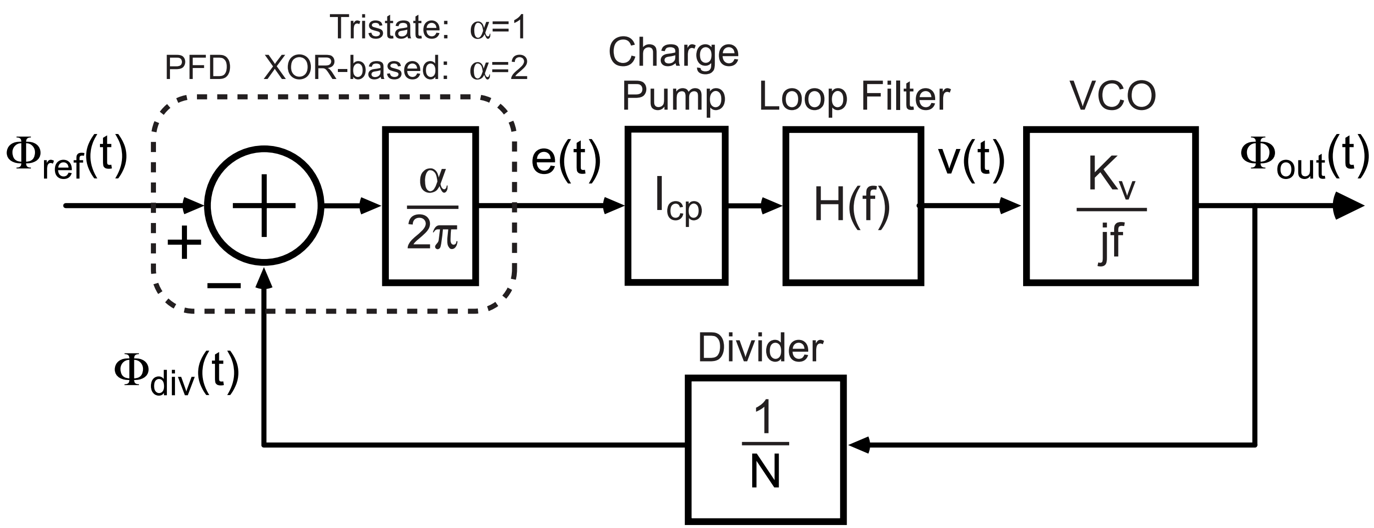
- **Example: lead/lag network from previous slide**

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

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

PLL Design with Lead/Lag Filter

- **Overall PLL block diagram**

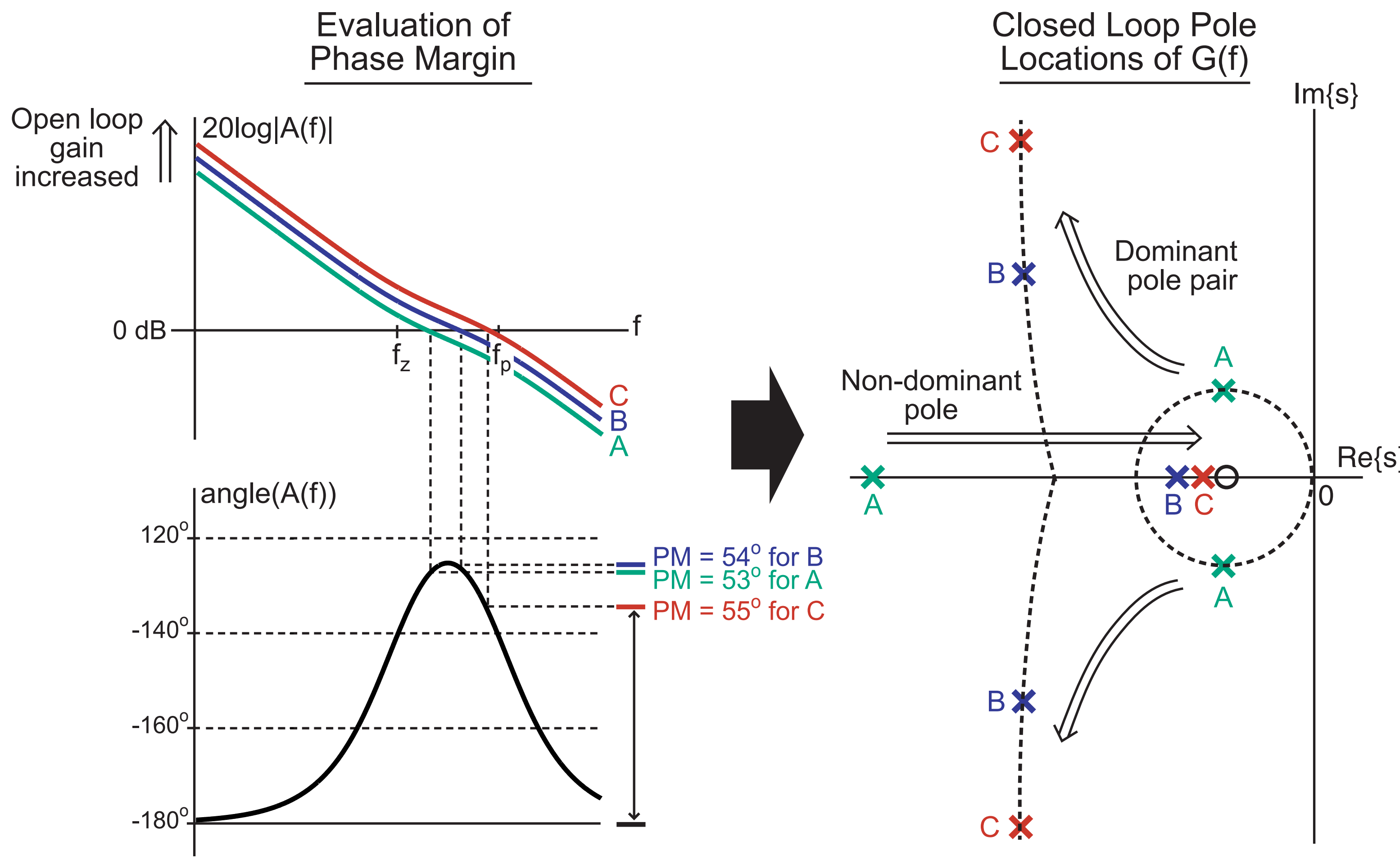


- **Loop filter**

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

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

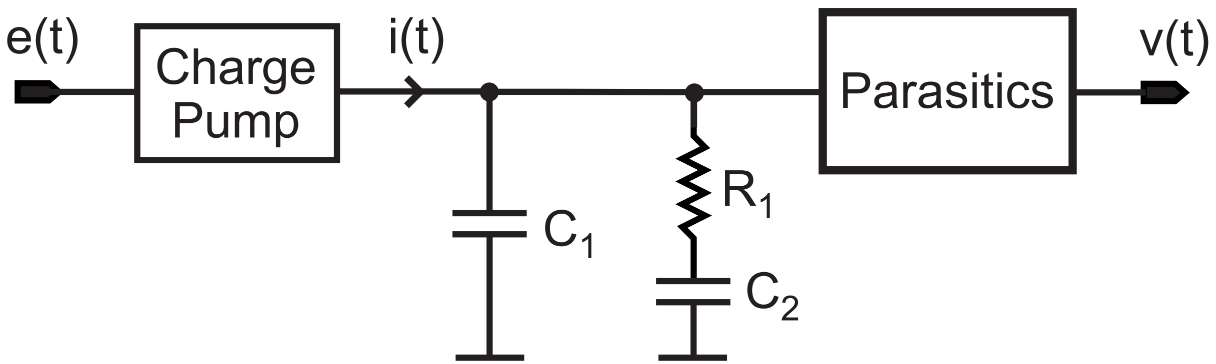
Closed Loop Poles Versus Open Loop Gain



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

Impact of Parasitics When Lead/Lag Filter Used

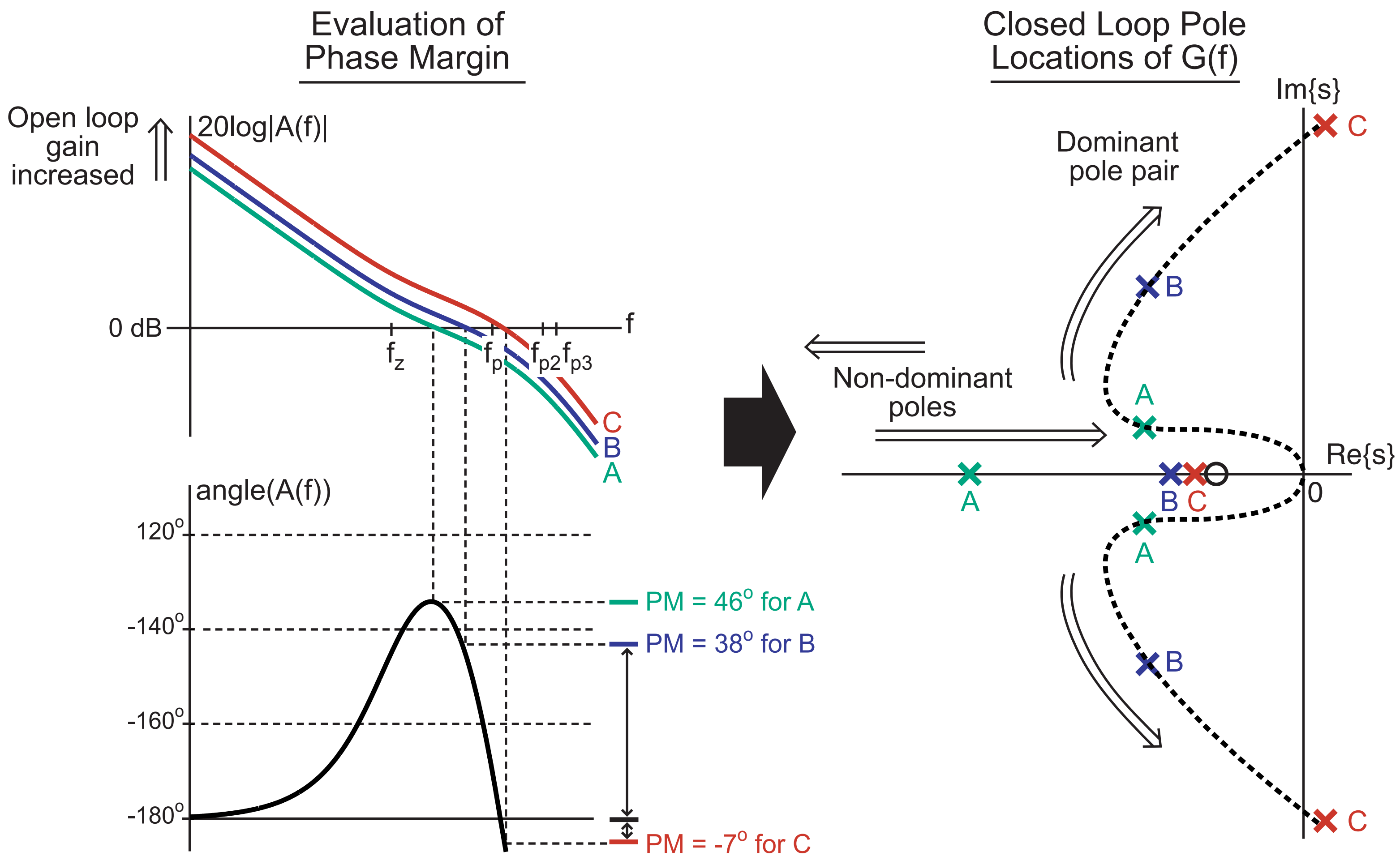
- **We can again model impact of parasitics by including them in loop filter transfer function**



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

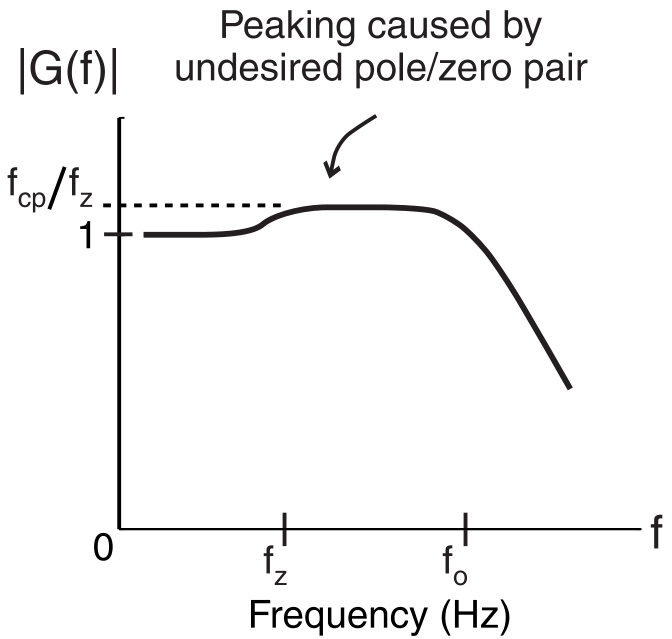
$$H(f) = \left(\frac{1}{sC_{sum}}\right)\frac{1+jf/f_z}{1+jf/f_p}\left(\frac{1}{1+jf/f_{p2}}\right)\left(\frac{1}{1+jf/f_{p3}}\right)$$

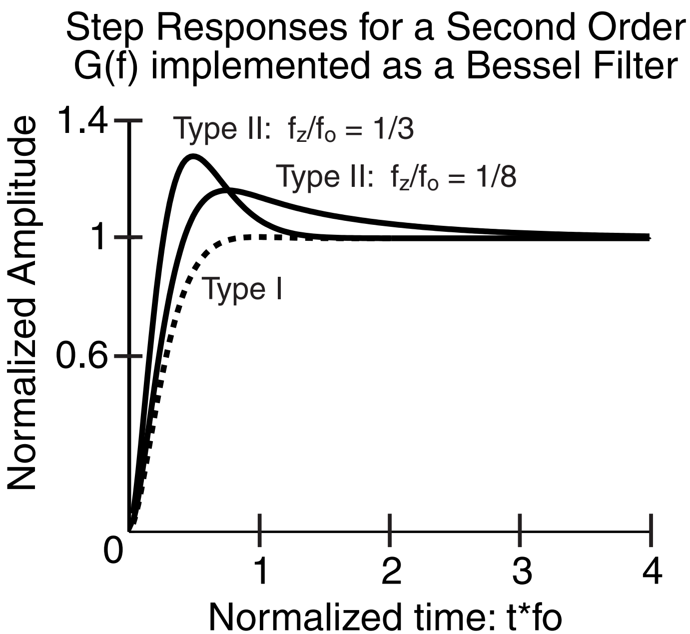
Closed Loop Poles Versus Open Loop Gain



- **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**
    - A big issue for CDR systems, but not too bad for wireless
  - **Extended settling time due to parasitic "tail" response**
    - Bad for wireless systems demanding fast settling time