# eCDRPLL

A radiation hard clock and data recovery circuit

EP-ESE Electronics Seminar, 13 December 2016

Pedro Leitao, EP-ESE-ME, CERN

# Agenda

## eCDRPLL Macrocell

- Overview
- Motivation
- PLL Topology
  - Working principle
  - Source of mismatches
  - Built-in-self-test (BIST) for mismatch estimation
- CDR topology
  - Overview
  - Reference-less calibration method
- Experimental results
  - PLL
  - CDR
- Summary

# eCDRPLL Macrocell - Architecture

## Radiation-hard and phase deterministic CDR & PLL

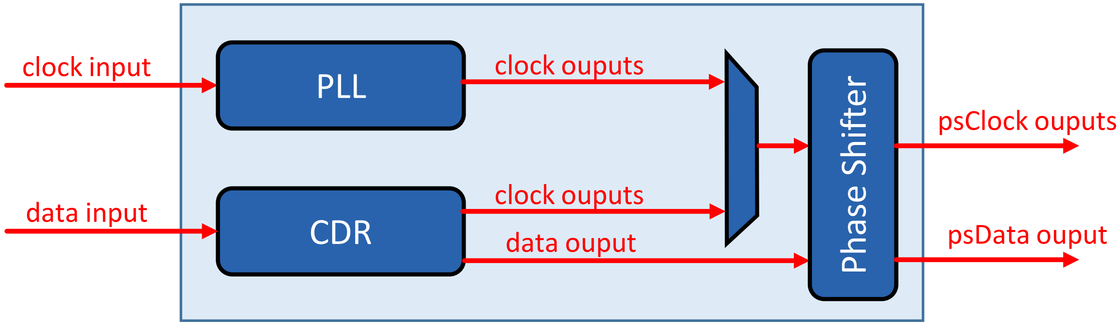
- Sub 10 ps rms jitter

## Two Frequency Multiplier modes

- VCO@240MHz - 40, 60, 120 and 240 MHz simultaneous output clocks
  - Full 360˚ phase shift with 260 ps resolution
  - Targets GBT-FPGA/backend applications
- VCO@320 MHz – 40, 80, 160 and 320 MHz simultaneous output clocks
  - Full 360˚ phase shift with 195 ps resolution
  - Targets TTC/e-link applications
- Built-in-self-test for mismatch characterization (to be run at production time)

## Clock and Data Recovery

- VCO@320 MHz
  - 40, 80, 160 and 320 MHz simultaneous output clocks
  - 40, 80, 160 and 320 Mb/s selectable output recovered data
  - Uses reference-less VCO calibration (no external clock required)
  - Full 360˚ phase shift with 195 ps resolution
  - data pushed-out on clock's falling edge



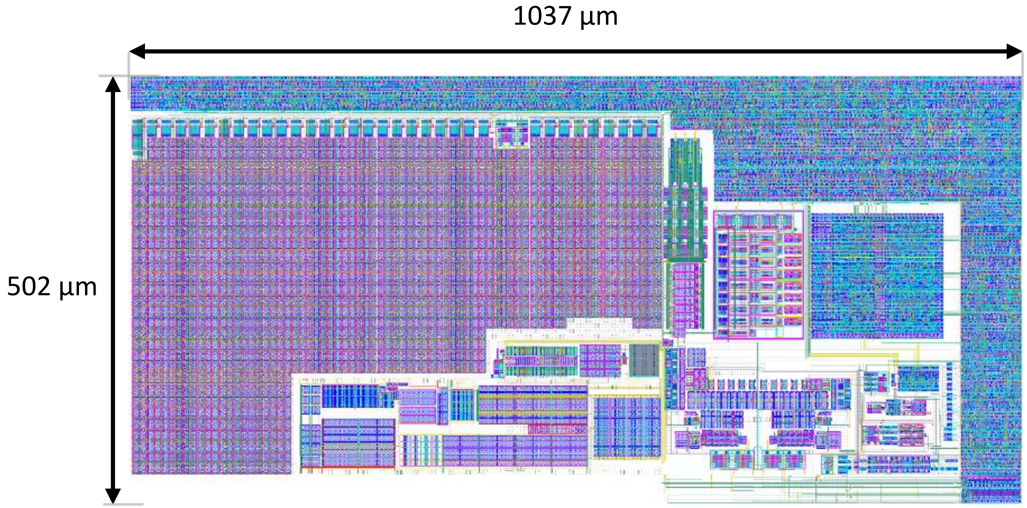
# eCDRPLL Macrocell - Implementation

## Radiation-hard and phase deterministic CDR & PLL

- 130nm CMOS technology (up to M6 metal stack)
- 1.2 V
- 0.52 mm$^2$ footprint

## Silicon proven

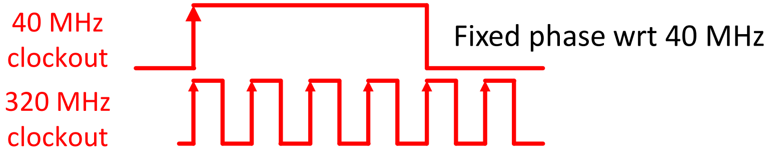
- Now available to the community as a macrocell
- Cliosoft repository
- Functional verilog model
- Digital core triplicated using TMRG tool and highVT cells
- Digital on top w/ timing and abstract files included allows a digital flow integration
- Documentation available at www.cern.ch/proj-gbt/eCDR-PLL

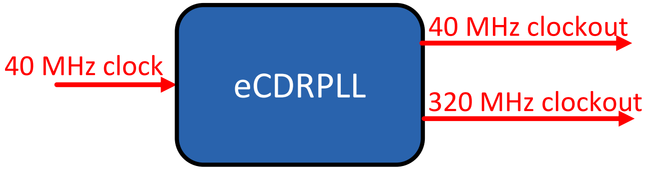


# Motivation

Developed within the GBT framework as a macrocell for the front-end ASICs

- Front-End (FE) application example
  - eCDRPLL with 40 MHz reference generating 40 and 320 MHz clocks

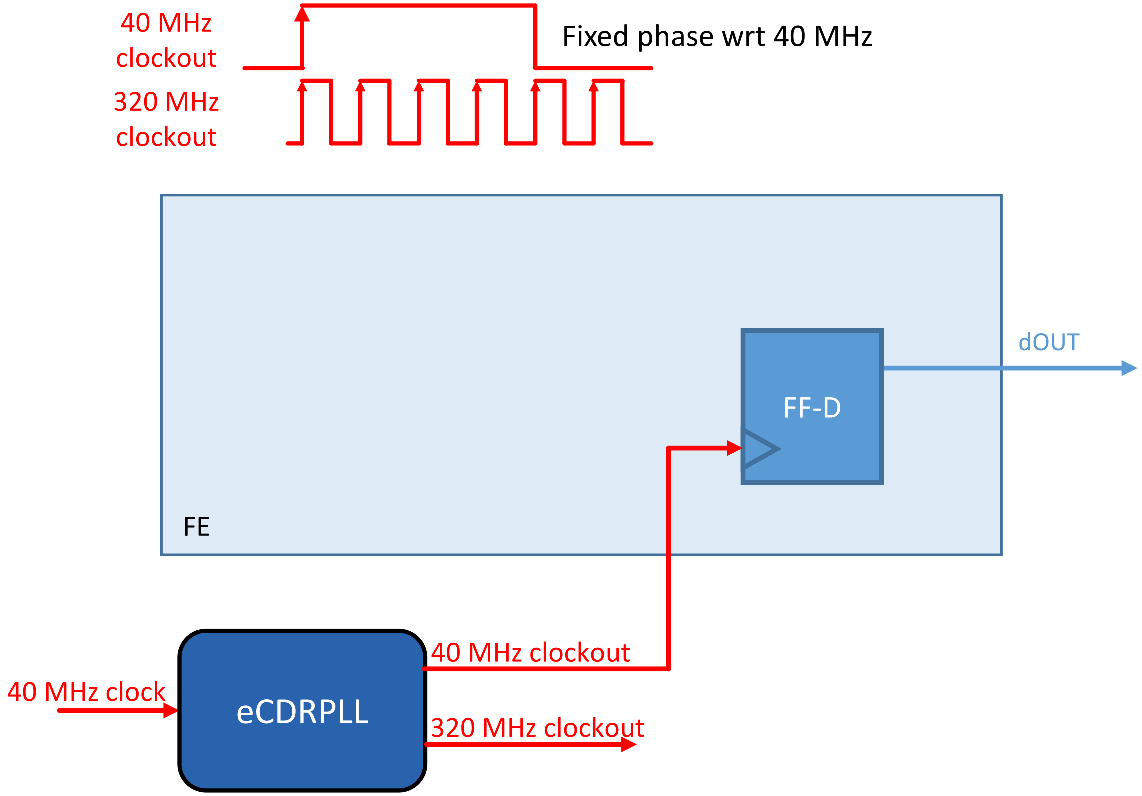


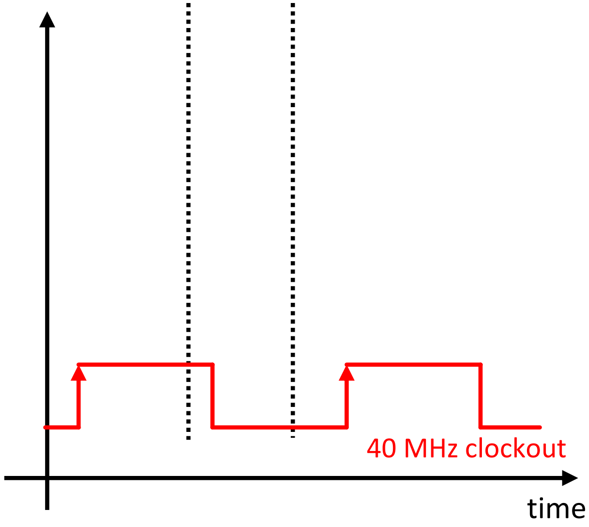


# Motivation

Developed within the GBT framework as a macrocell for the front-end ASICs

- Front-End (FE) application example
  - eCDRPLL with 40 MHz reference generating 40 and 320 MHz clocks



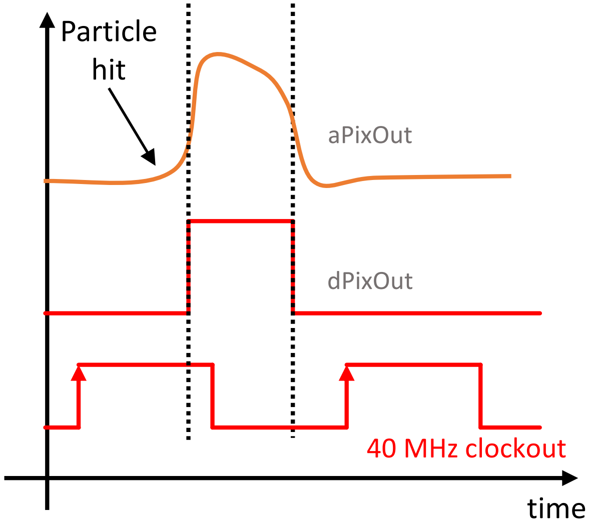


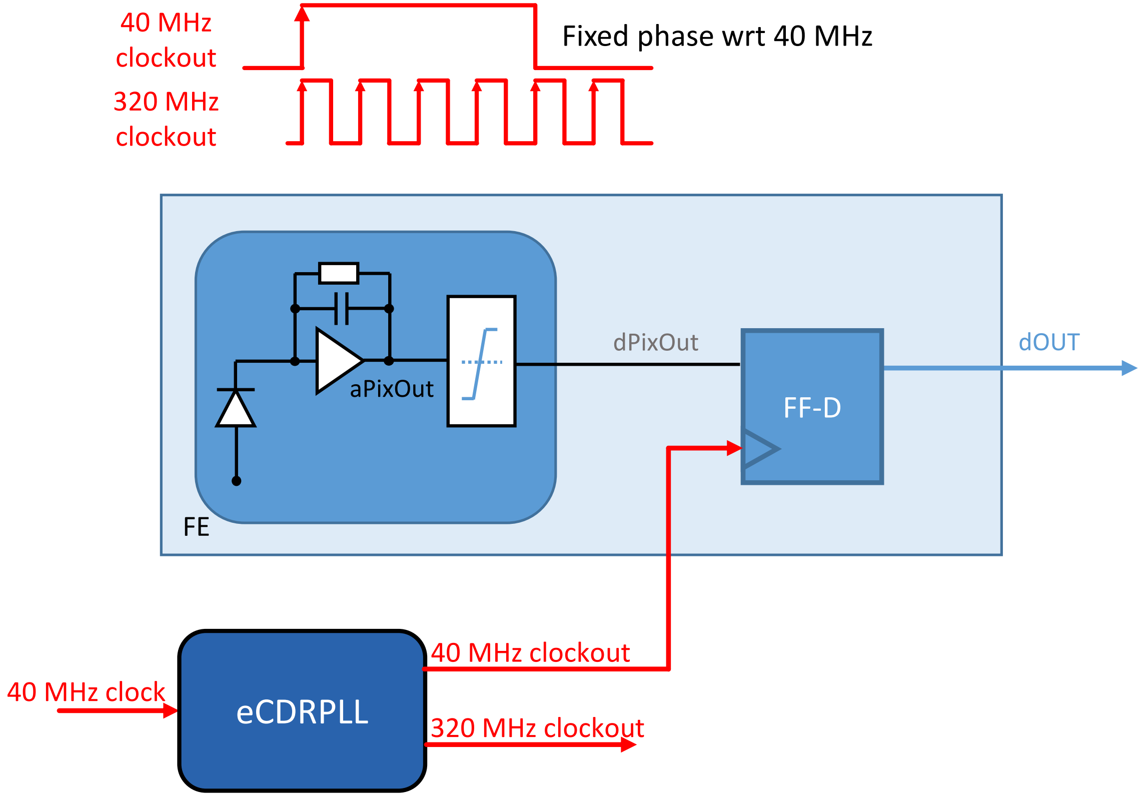
Continuous readout

# Motivation

Developed within the GBT framework as a macrocell for the front-end ASICs

- Front-End (FE) application example
  - eCDRPLL with 40 MHz reference generating 40 and 320 MHz clocks
  - The front-end analog output is out of phase



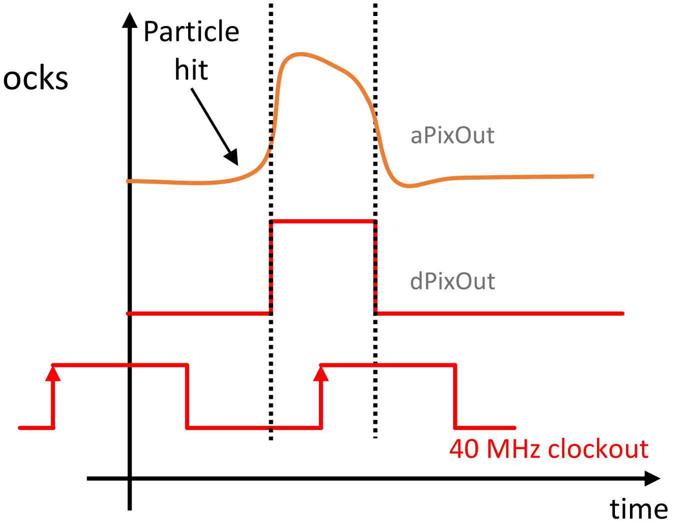


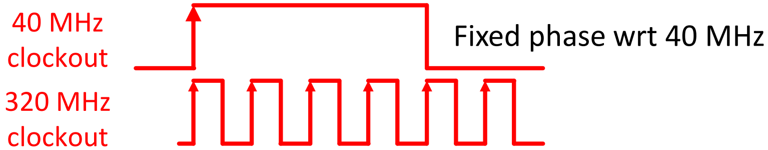
Continuous readout

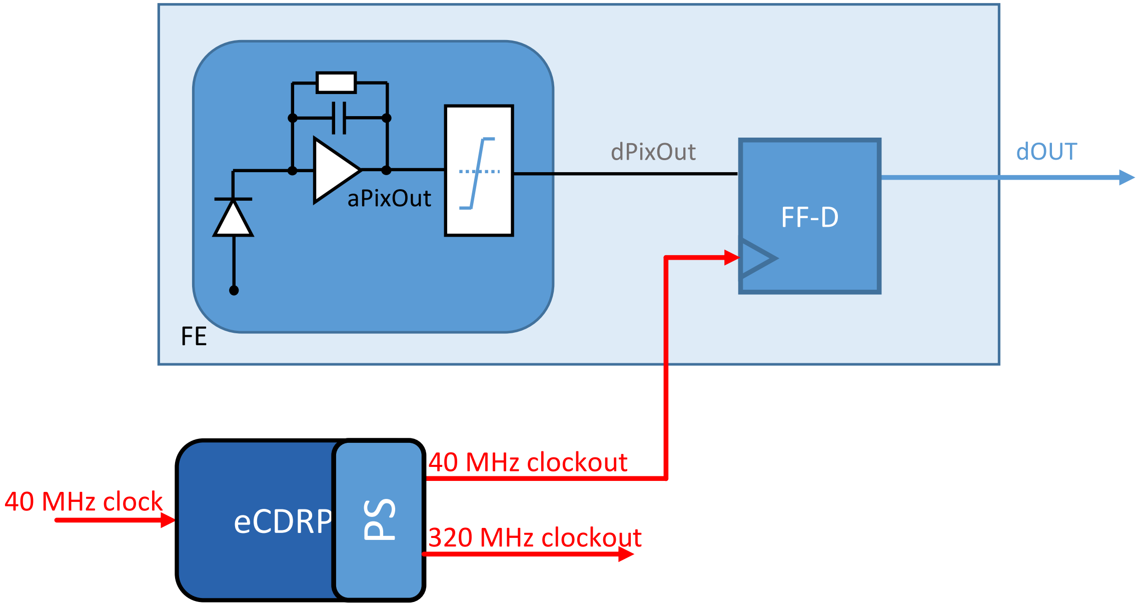
# Motivation

## Developed within the GBT framework as a macrocell for the front-end ASICs

- Front-End (FE) application example
  - eCDRPLL with 40 MHz reference generating 40 and 320 MHz clocks
  - The front-end analog output is out of phase
  - Solution: phase shift the 40 MHz clock





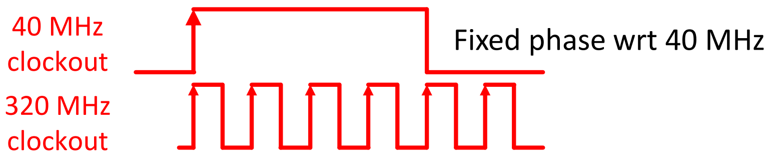


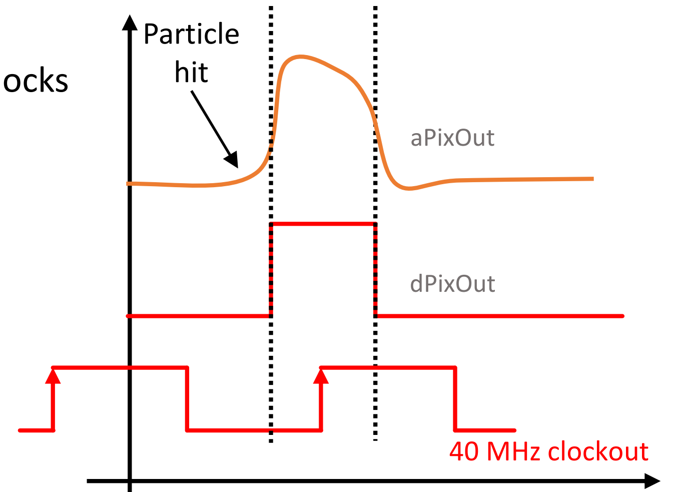
Continuous readout

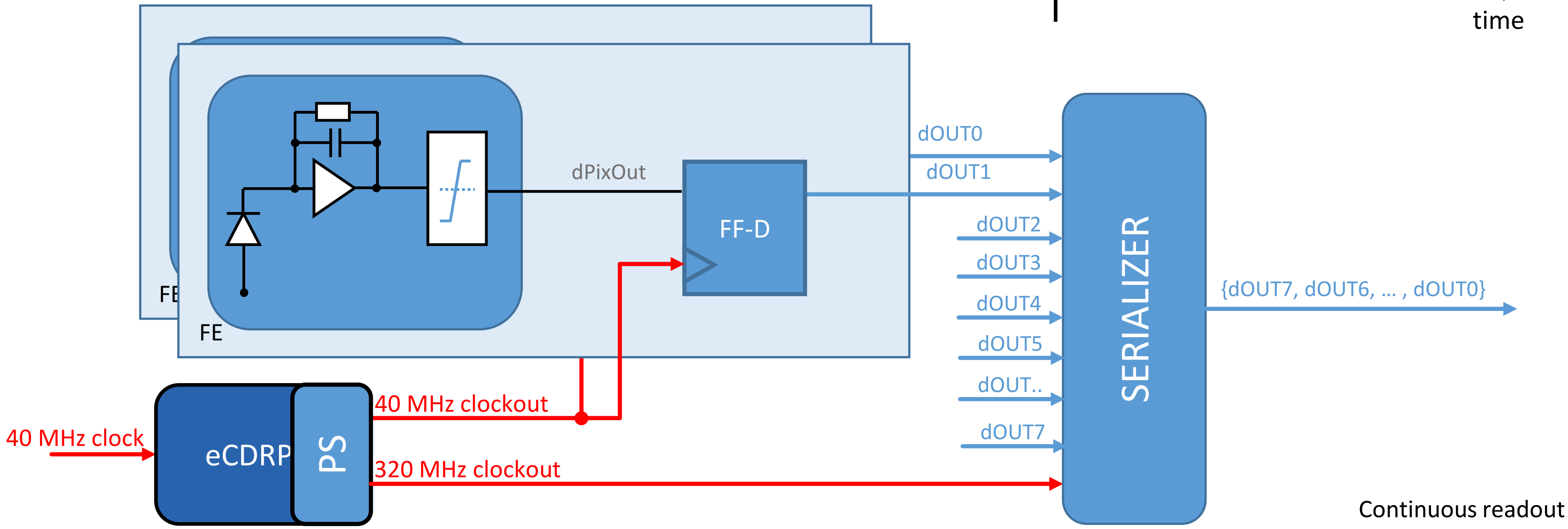
# Motivation

Developed within the GBT framework as a macrocell for the front-end ASICs

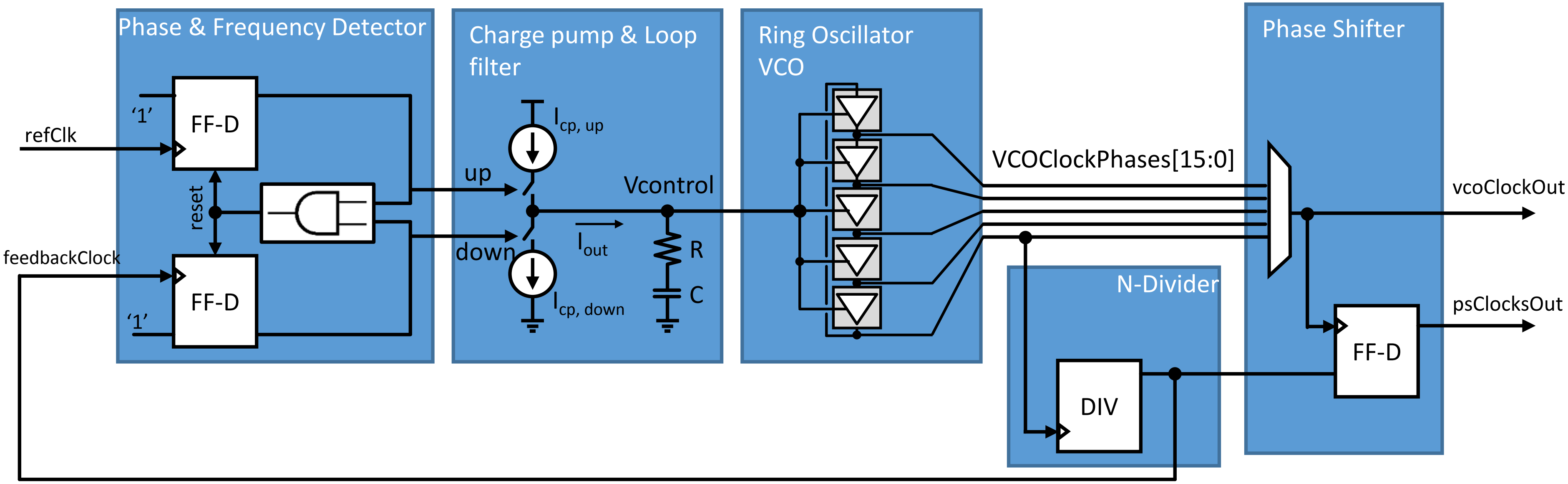
- Front-End (FE) application example
  - eCDRPLL with 40 MHz reference generating 40 and 320 MHz clocks
  - The front-end analog output is out of phase
  - Solution: phase shift the 40 MHz clock
  - Clock resources for the readout (clock multiplication)

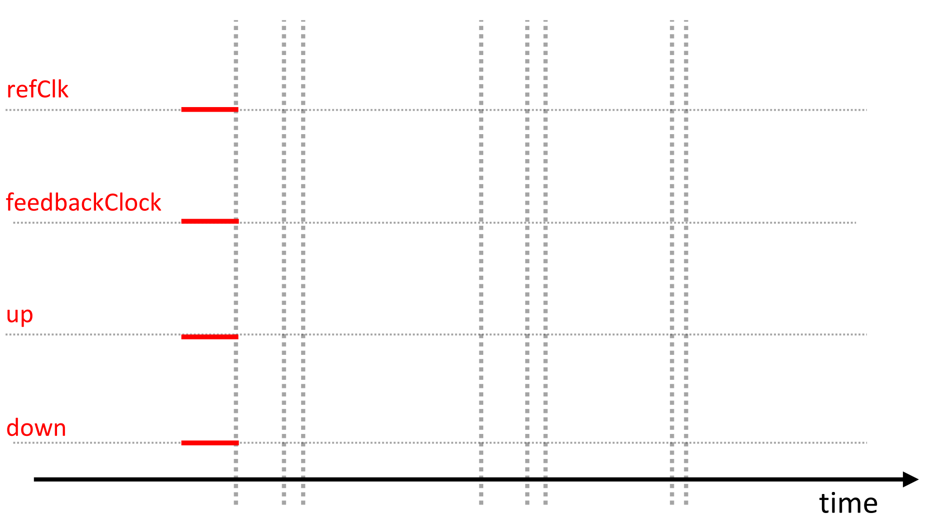


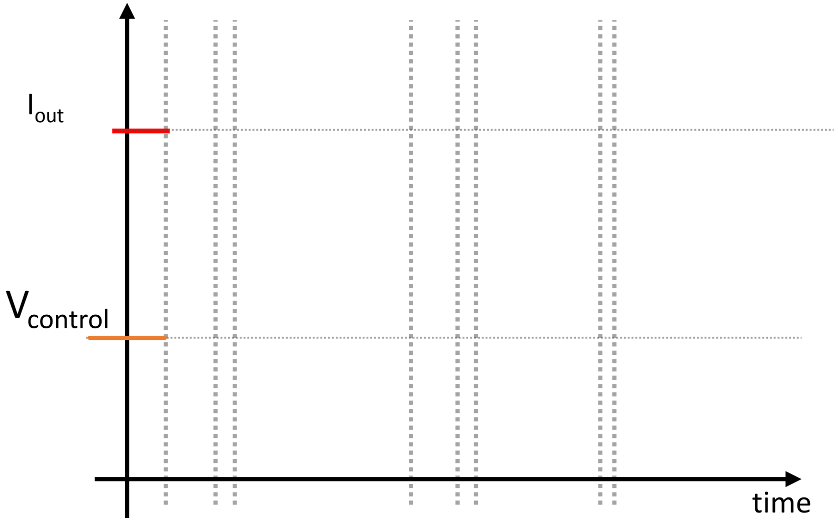




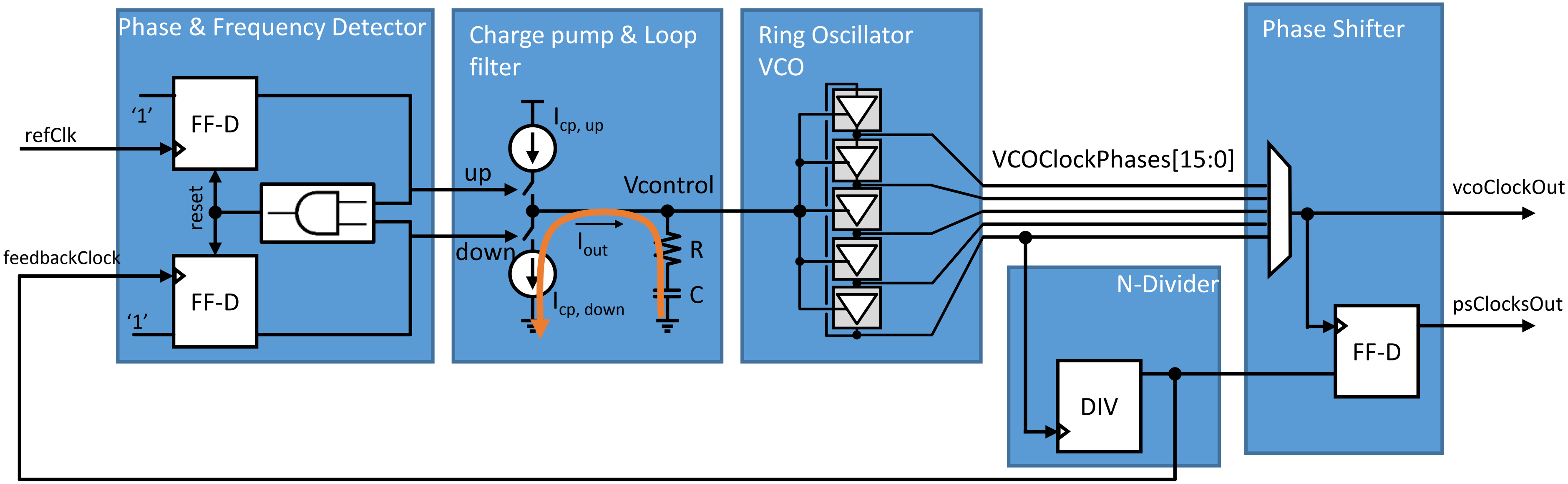
# eCDRPLL’s simplified PLL topology

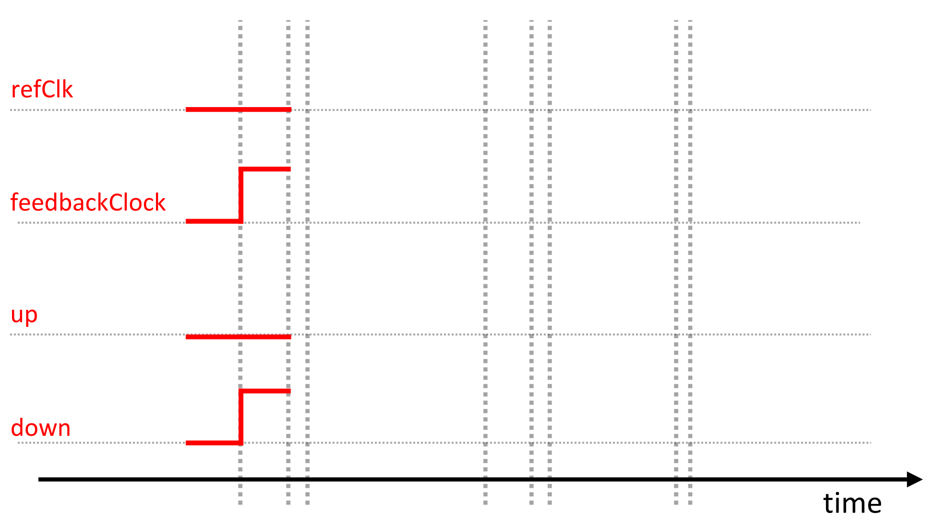


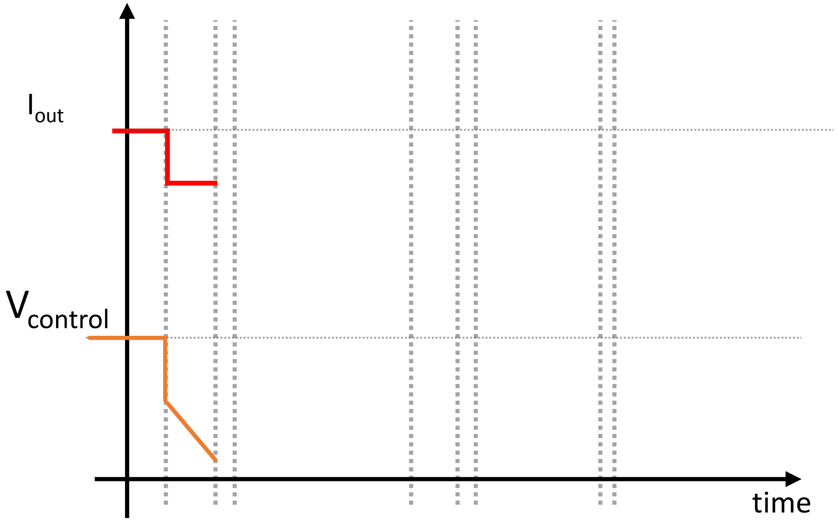




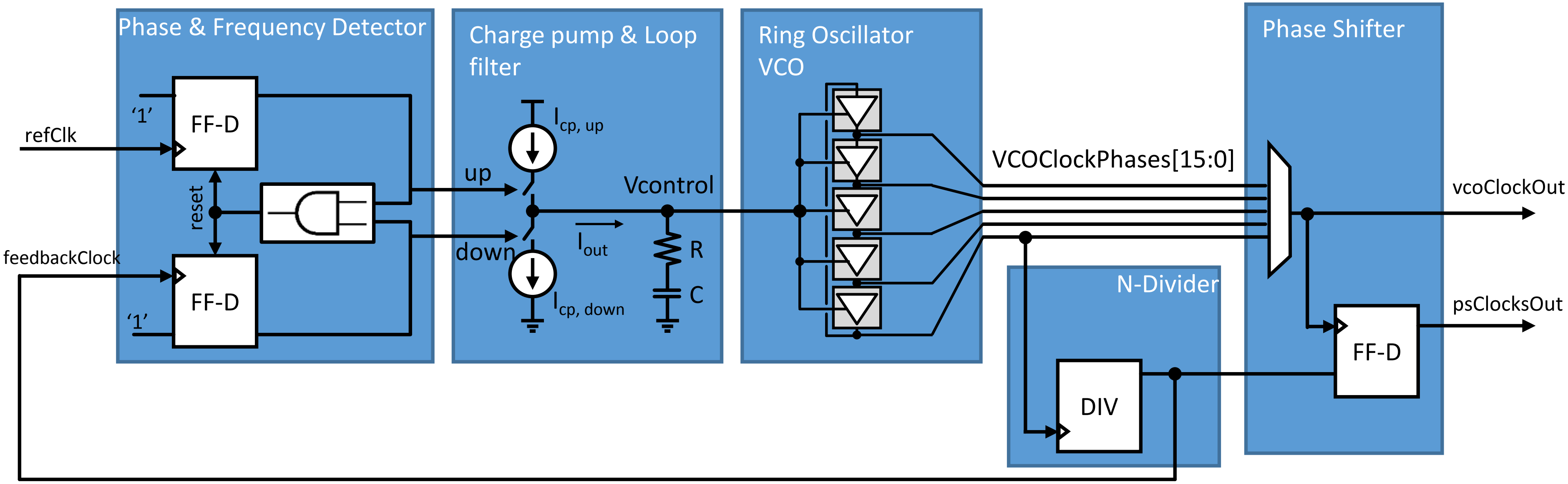
# eCDRPLL's simplified PLL topology

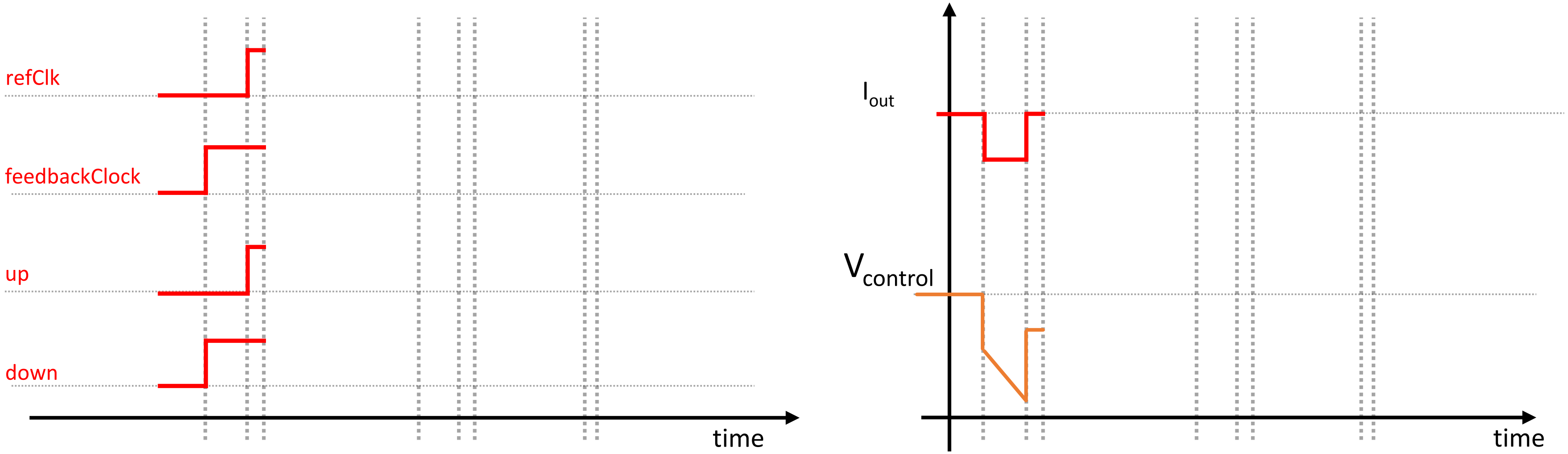




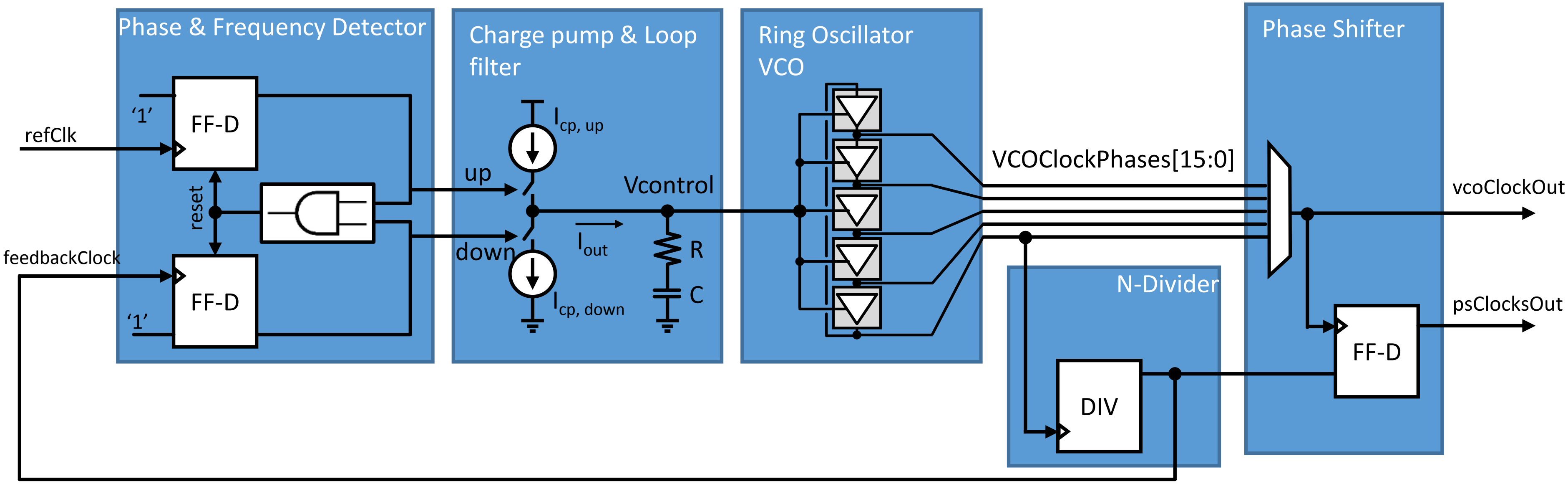


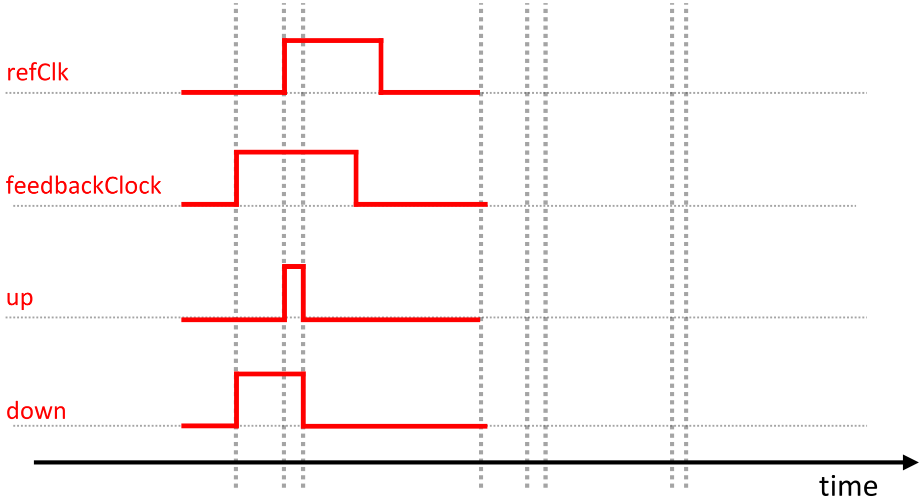
# eCDRPLL’s simplified PLL topology

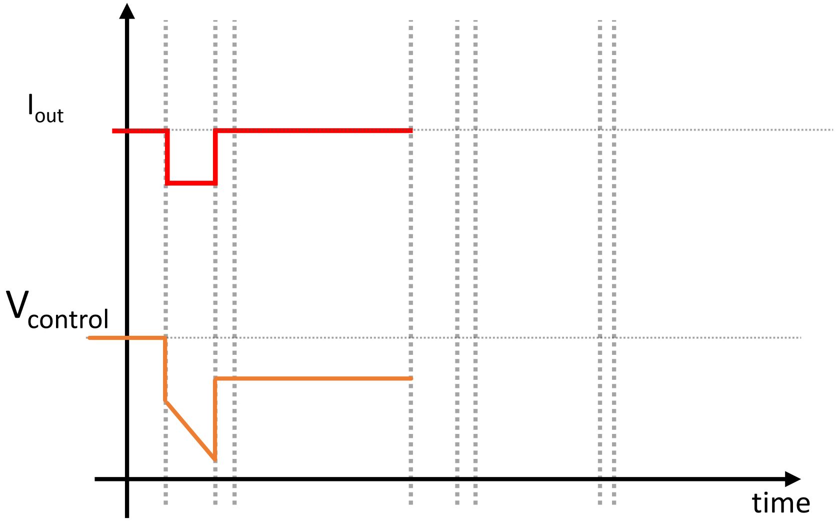




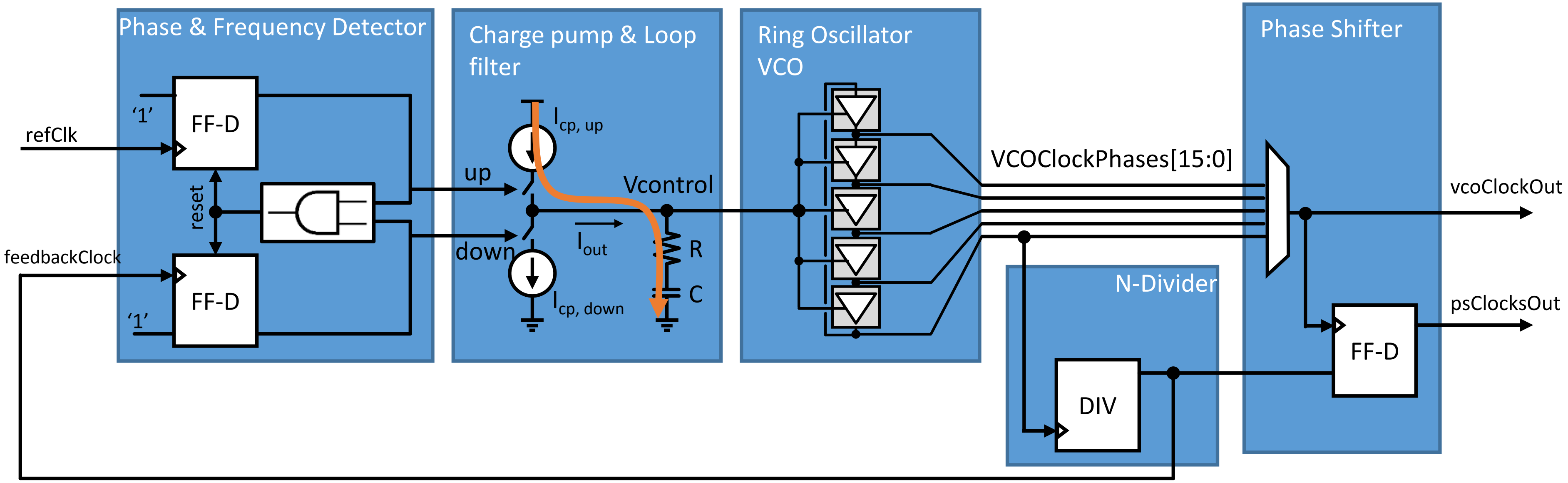
# eCDRPLL’s simplified PLL topology

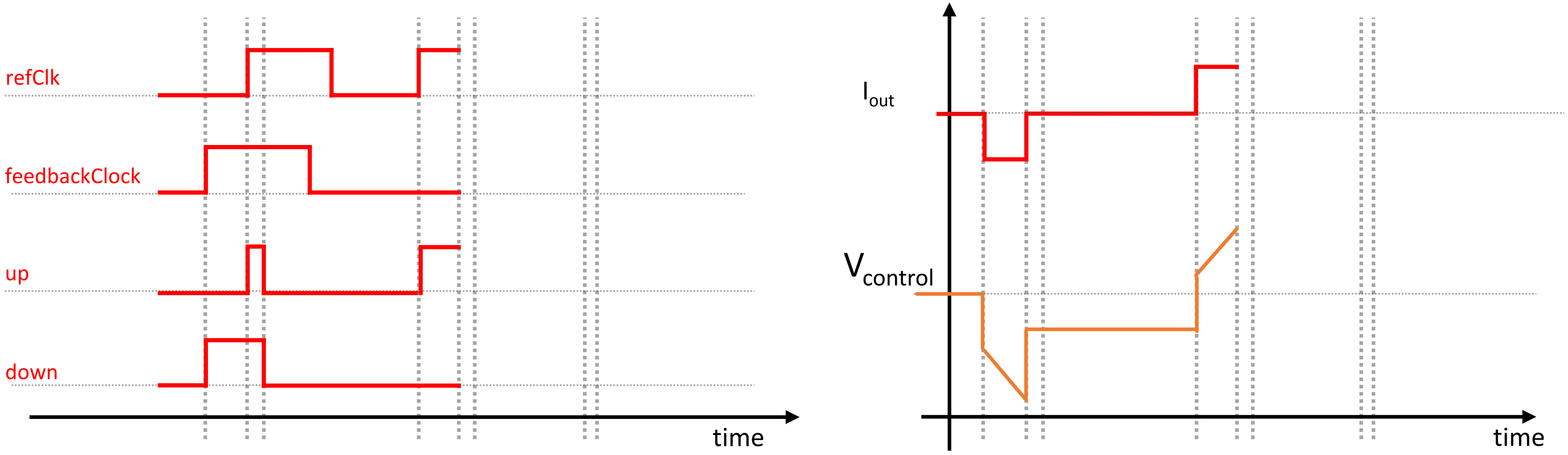




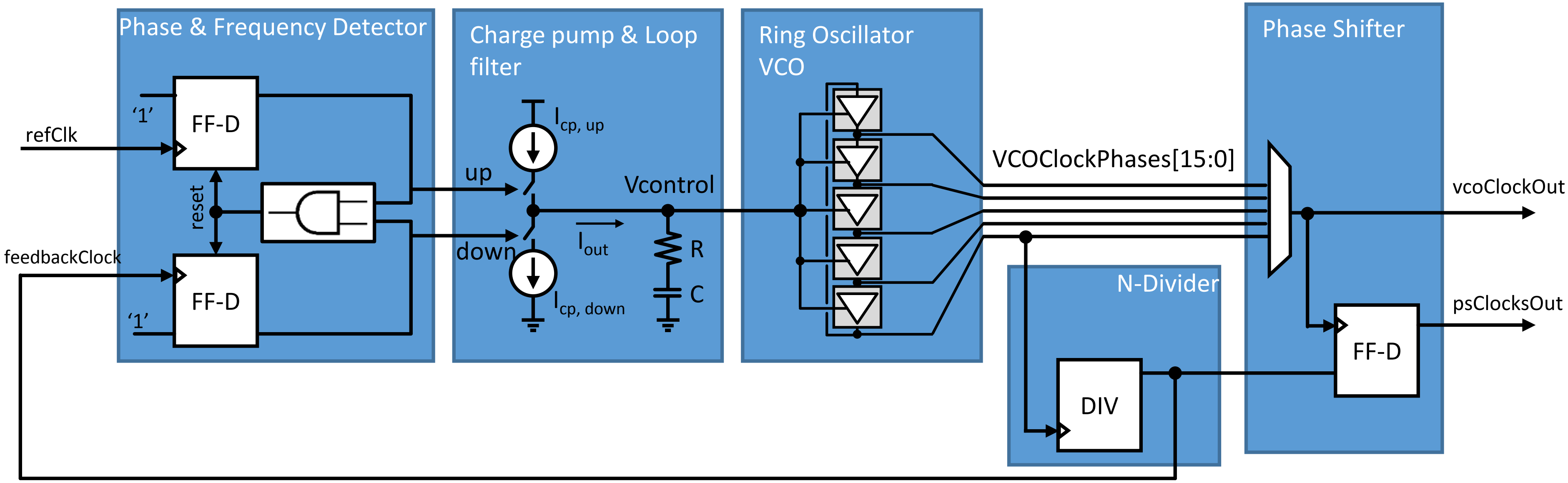


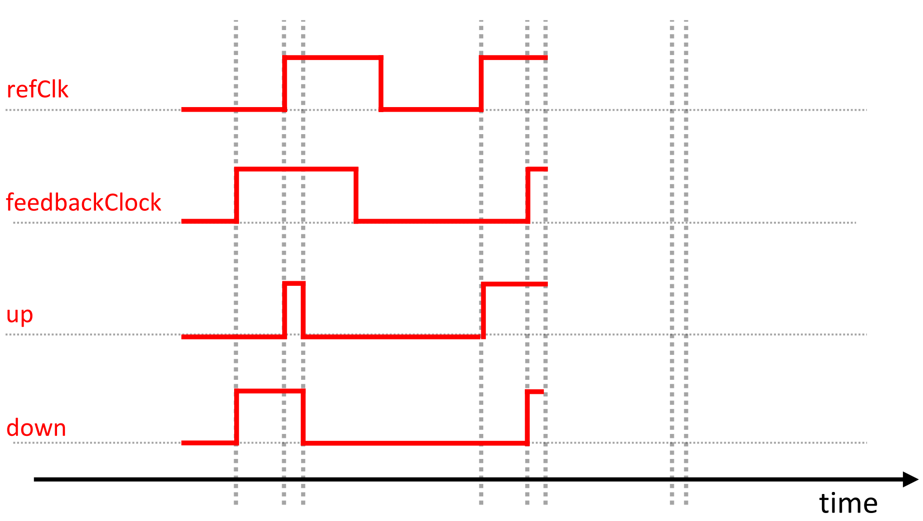
# eCDRPLL's simplified PLL topology

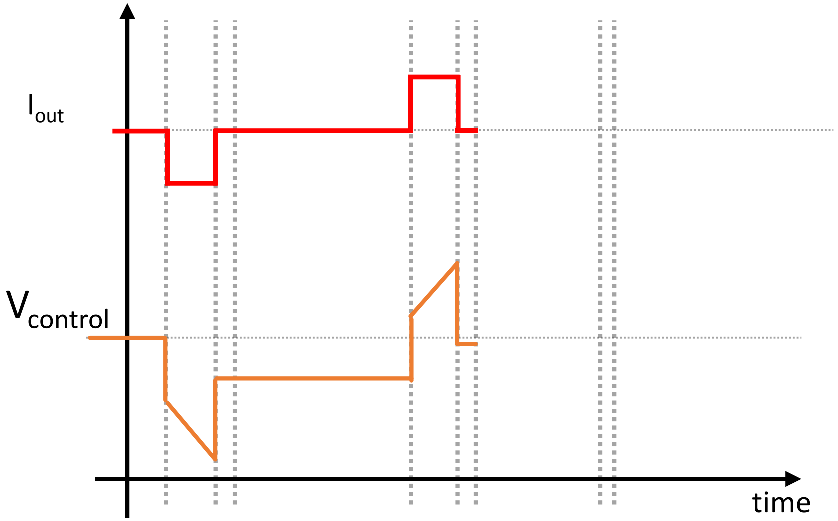




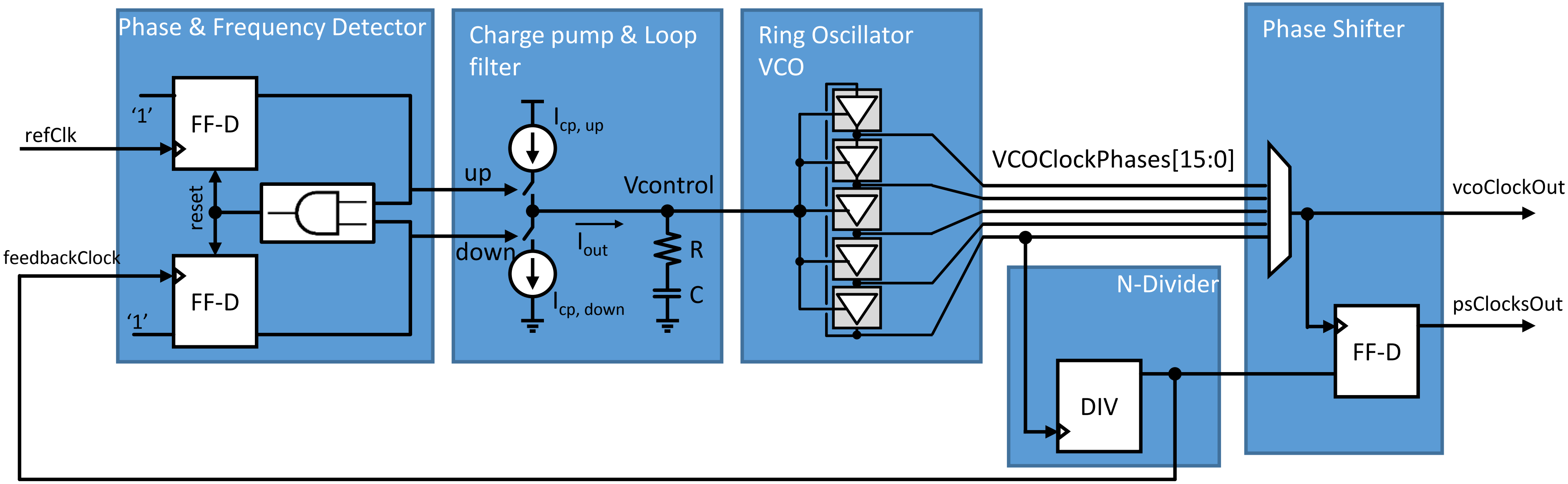
# eCDRPLL’s simplified PLL topology

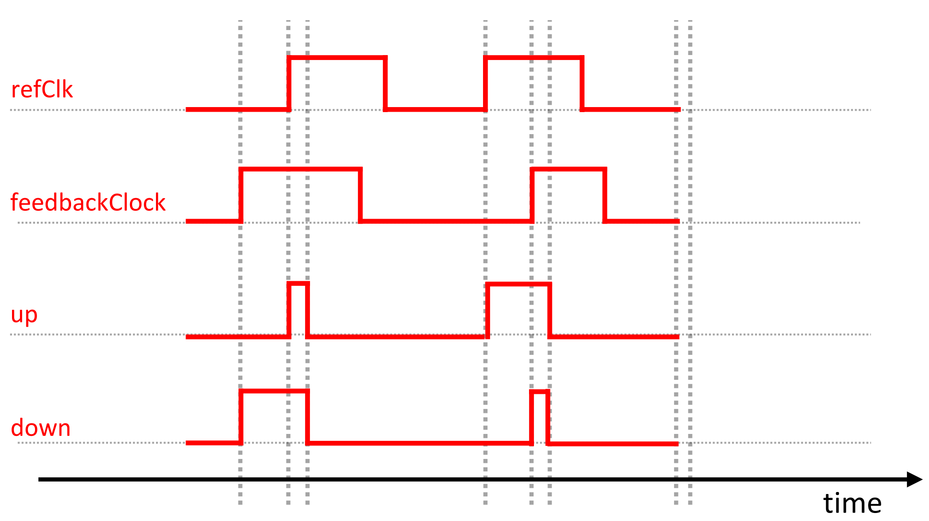


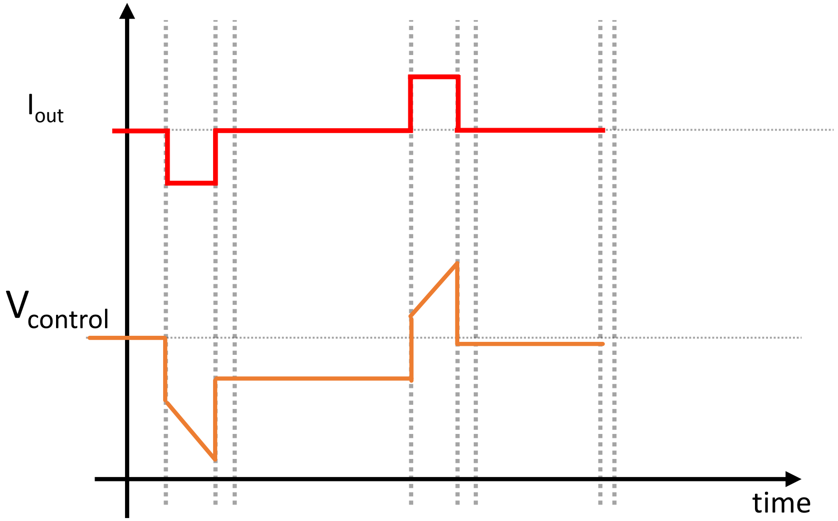




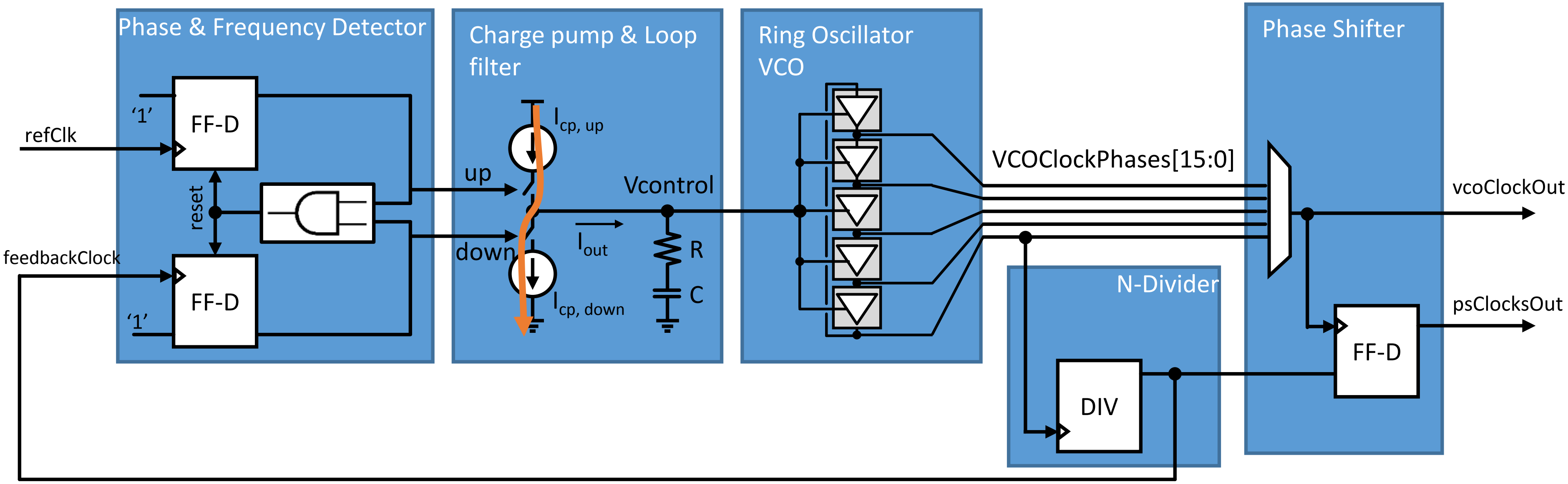
# eCDRPLL’s simplified PLL topology

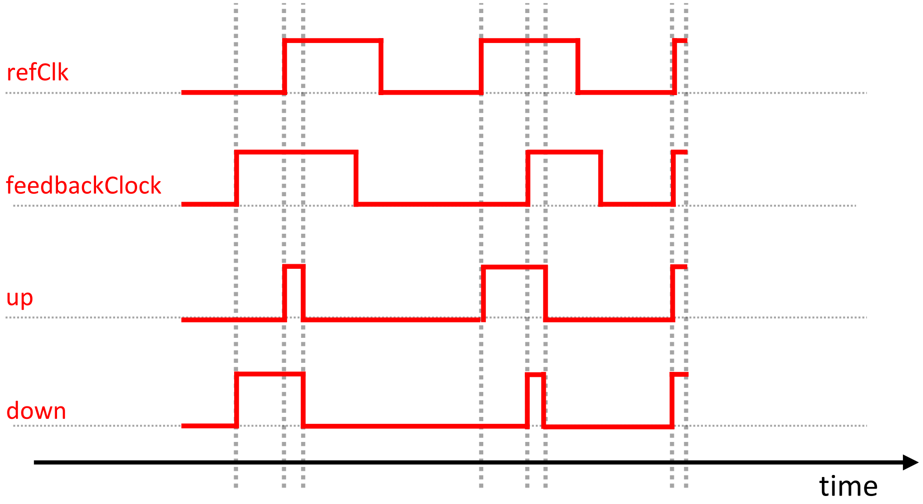


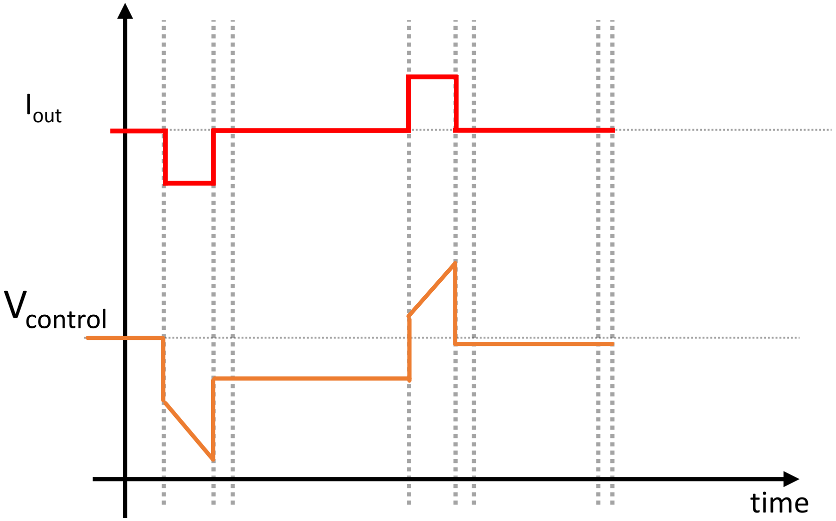




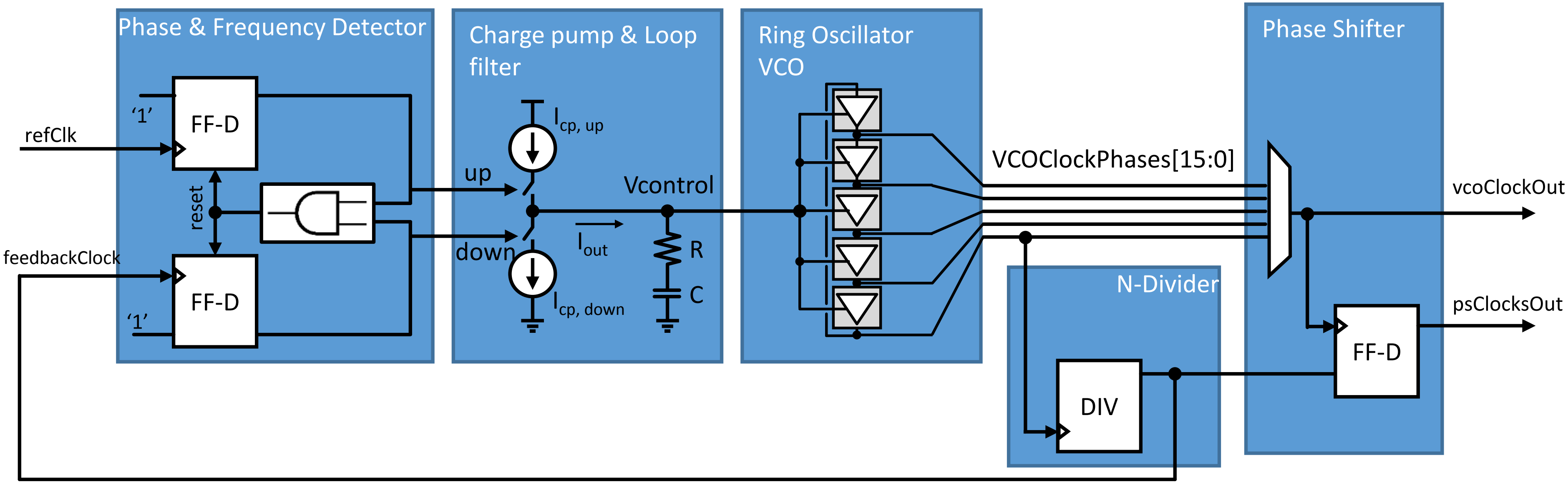
# eCDRPLL’s simplified PLL topology

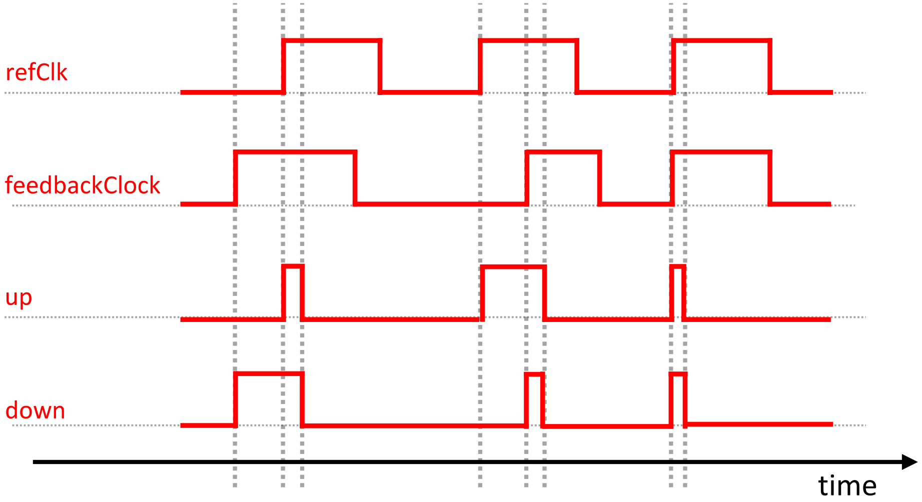


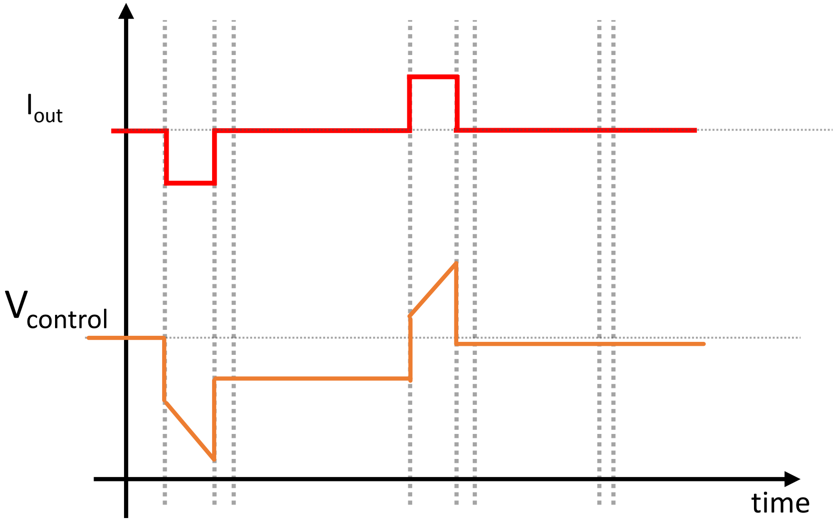




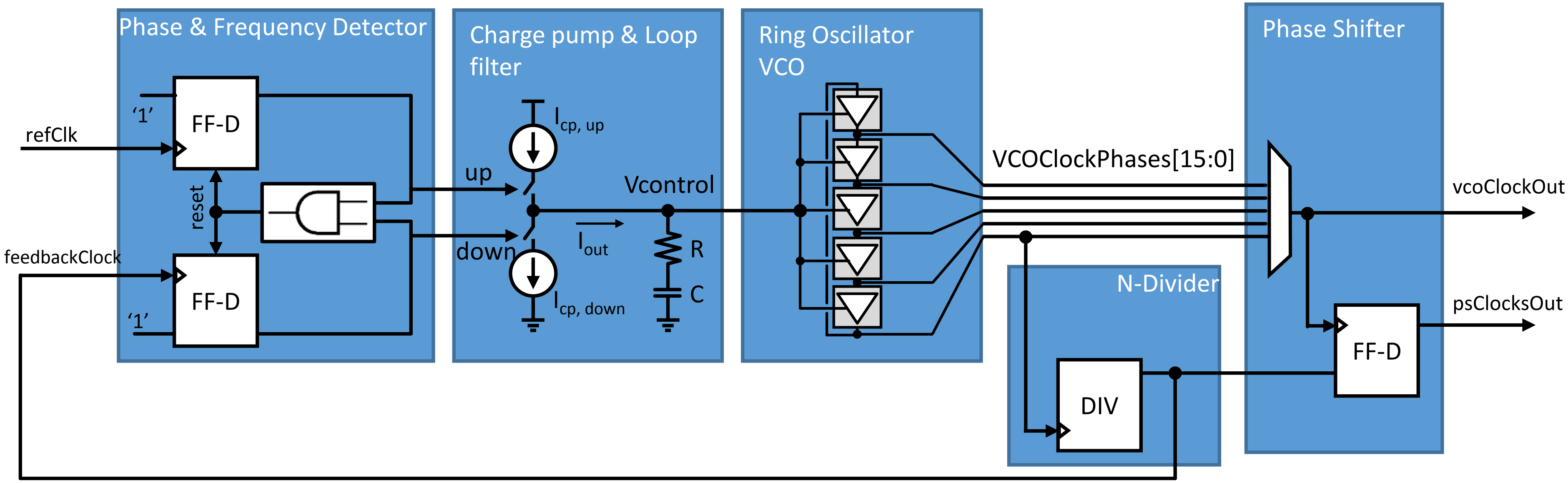
# eCDRPLL's simplified PLL topology





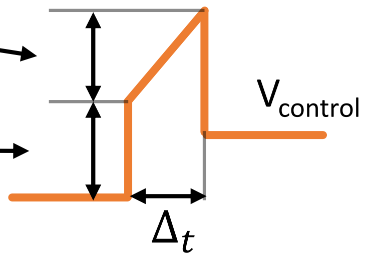


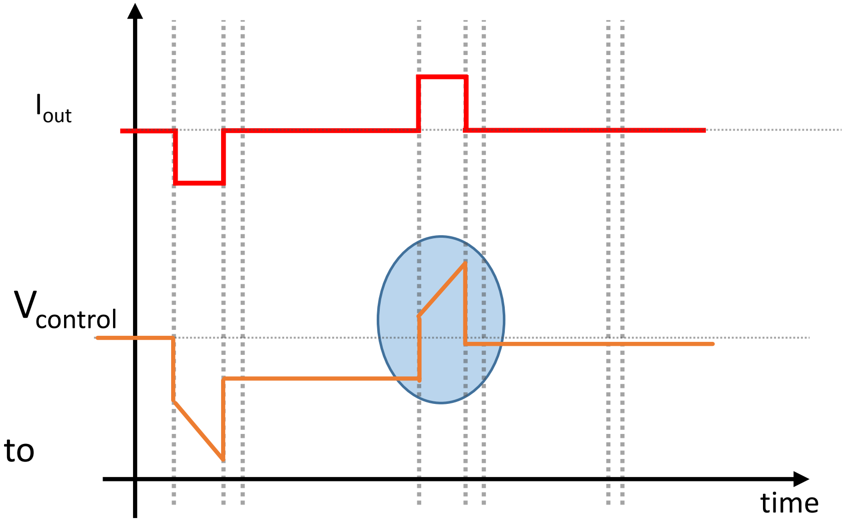
# eCDRPLL's simplified PLL topology



$$\Delta V_I = I_{cp} \cdot \frac{\Delta_t}{C}$$

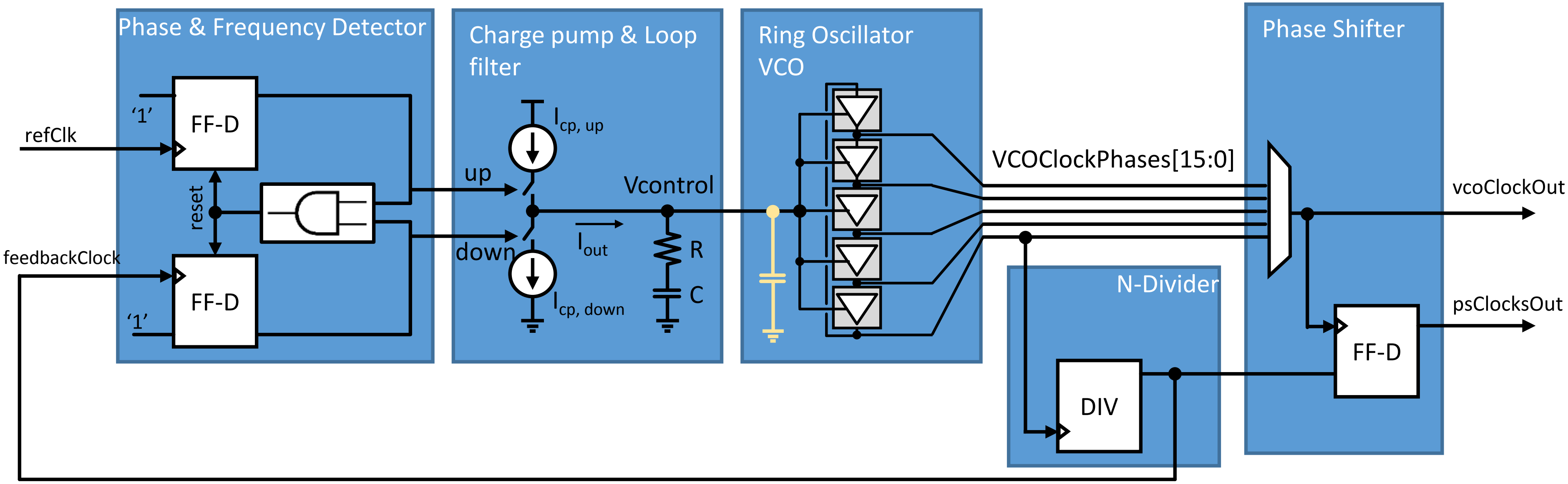
$$\Delta V_p = I_{cp} \cdot R$$





- Behaves as a 2nd order negative feedback loop
- $I_{cp}$, R and C control the loop dynamics
- Can be optimized to achieve best performance with relation to the input refClk

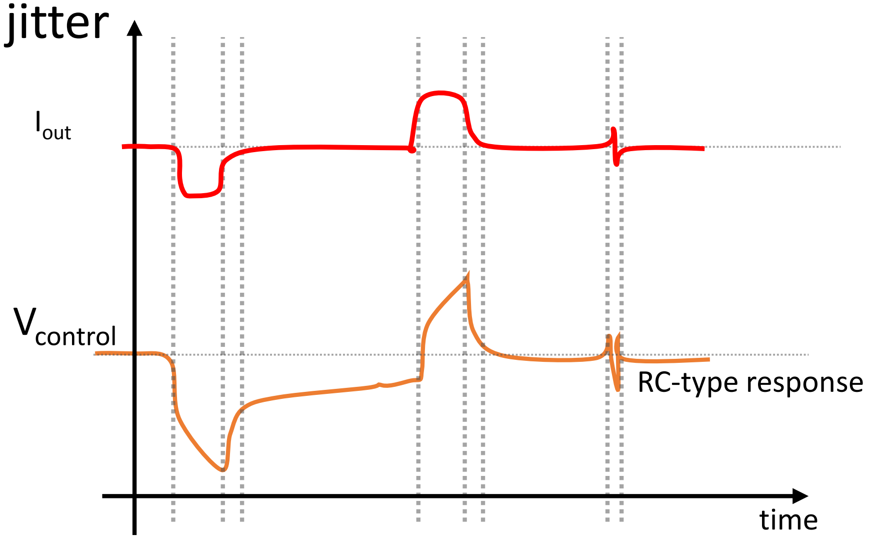
# eCDRPLL’s sources of noise



## Mismatches in the PLL loop filter will create jitter

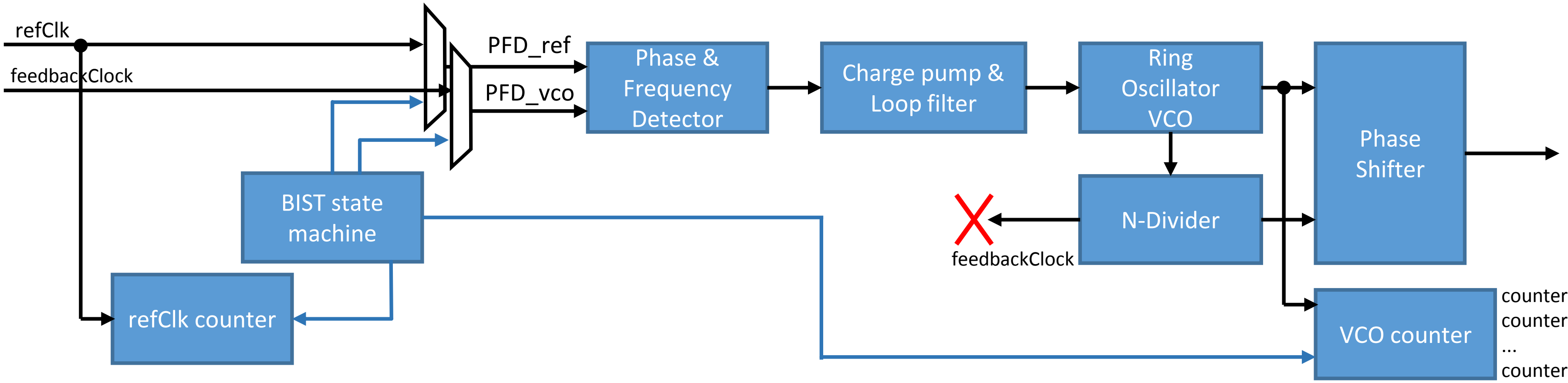
- $I_{cp,up}$ not exacly the same as $I_{cp,down}$
- Leakage in the loop filter
- Parasitic capacitances in the loop filter
- Noise in the VCO
- Process variation

- The closed loop will try to self-correct mismatches
- The mismatches can be revealed by open-loop operation



# eCDRPLL's BIST

The BIST allows to estimate mismatches from the open-loop response



## BIST

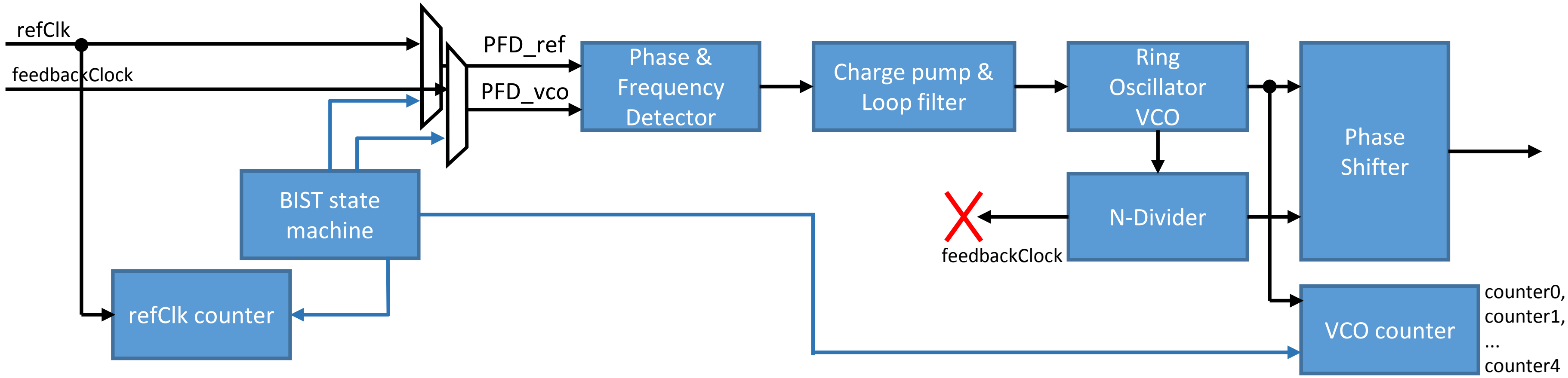
- To be run at production time (wafer probing), saving time in the long run
- Correspondence between the PLL's jitter and the BIST results
- Removes the need for lenghly jitter measurements with high-end equipment

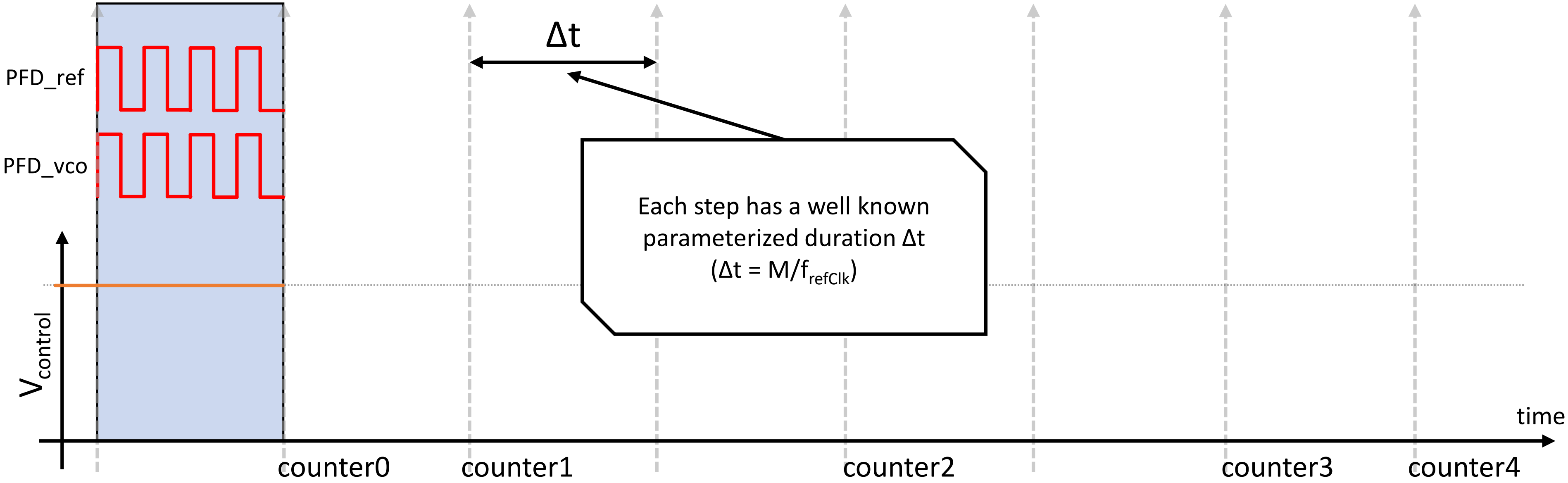
- The BIST state machine controls the PFD inputs with a predefined stimuli
  - The VCO frequency is forced to go up and down during the amont of time
  - Differences between starting and finishing VCO frequencies will reveal any mismatches in the loop
- The VCO frequency is monitored by the VCO counter and stored at each step
- The refClk counter is used as a reference counter to calculate the VCO frequency

*Investigations for Minimum Invasion Digital Only Built-In "Ramp" Based Test Techniques for Charge Pump PLL's*, Martin John Burbidge *et al*, ETW'02

# eCDRPLL’s BIST

The BIST allows to estimate mismatches from the open-loop response

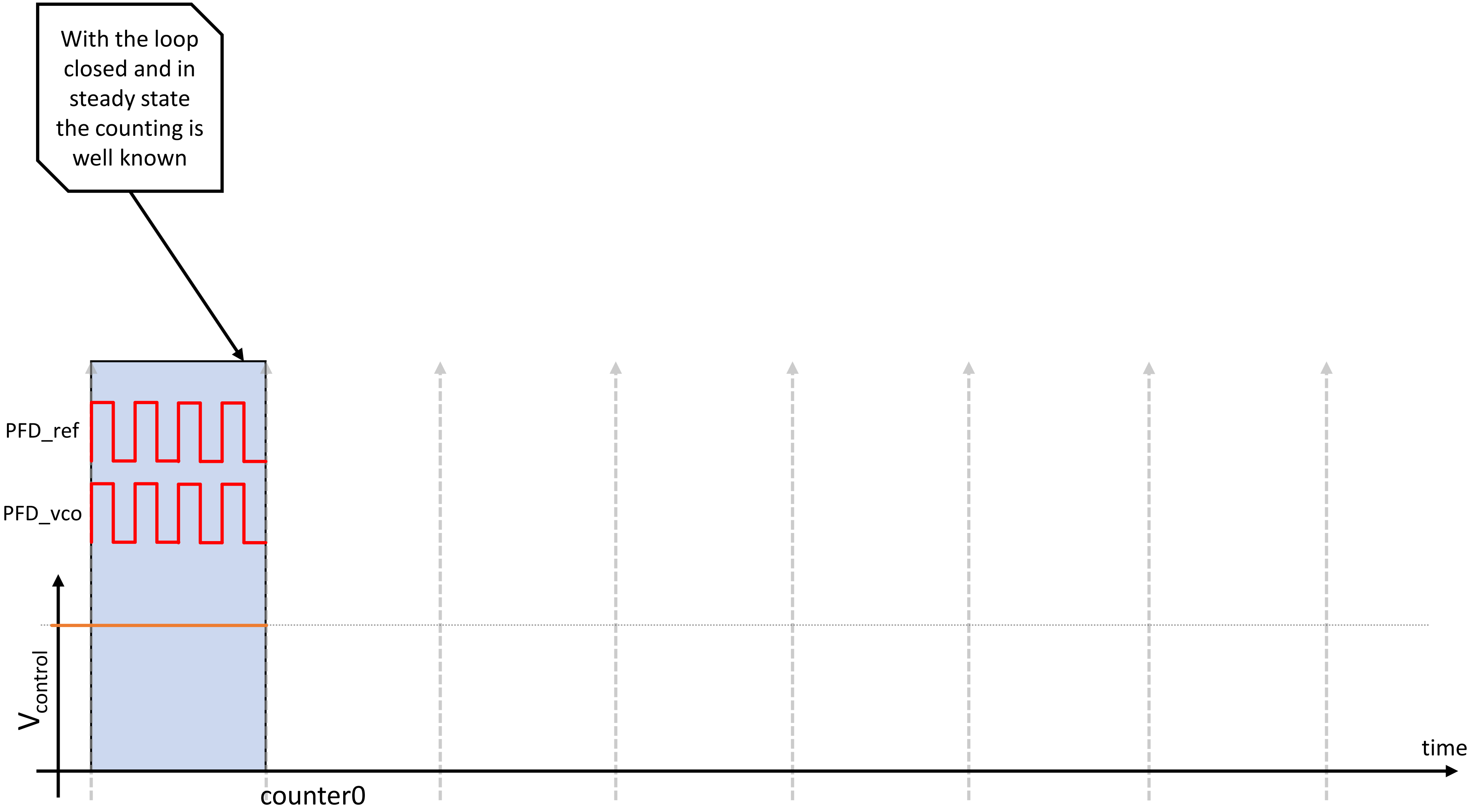




Each step has a well known parameterized duration Δt ($\Delta t = M/f_{refClk}$)

# eCDRPLL’s BIST

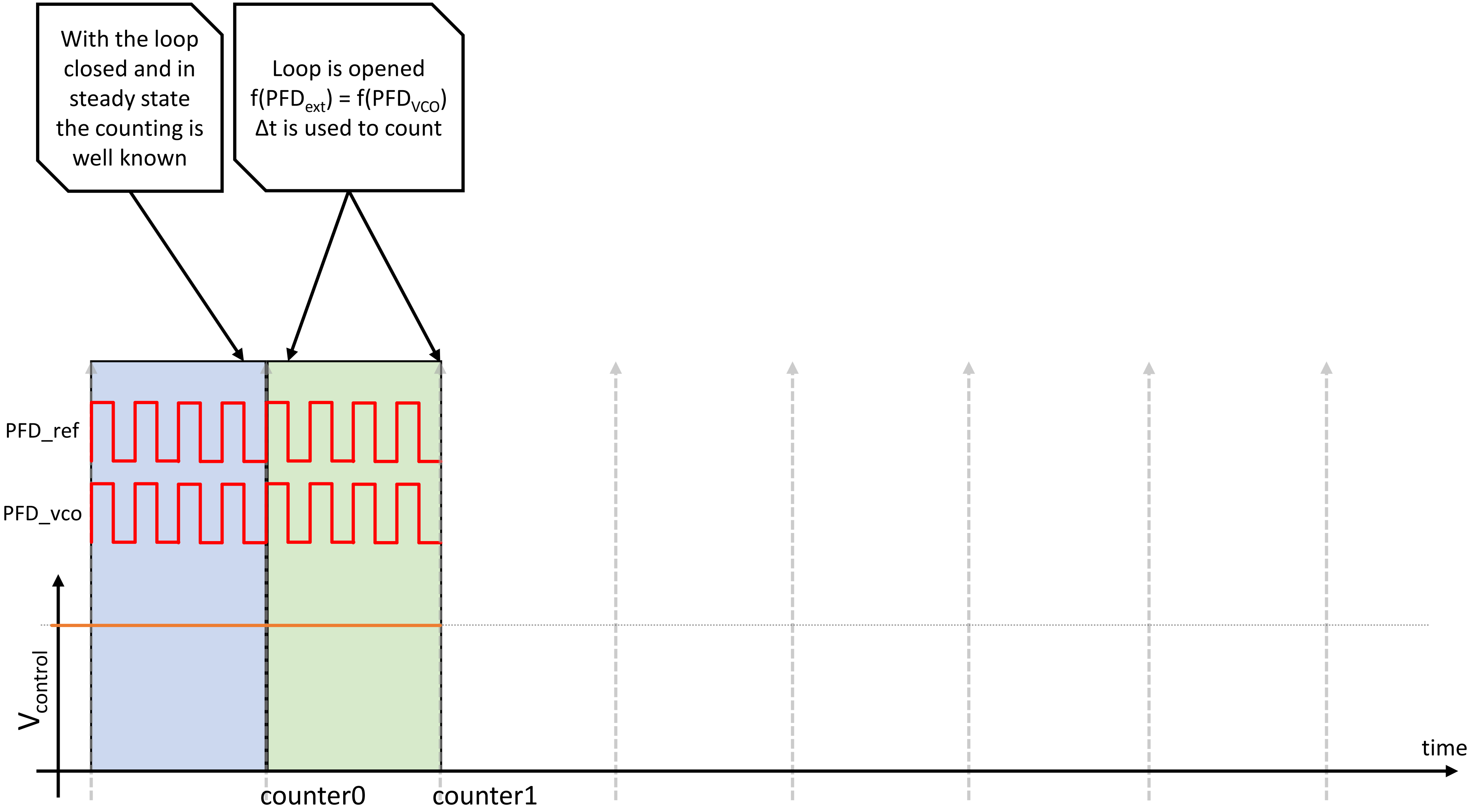
The BIST allows to estimate mismatches from the open-loop response



With the loop closed and in steady state the counting is well known

# eCDRPLL’s BIST

The BIST allows to estimate mismatches from the open-loop response

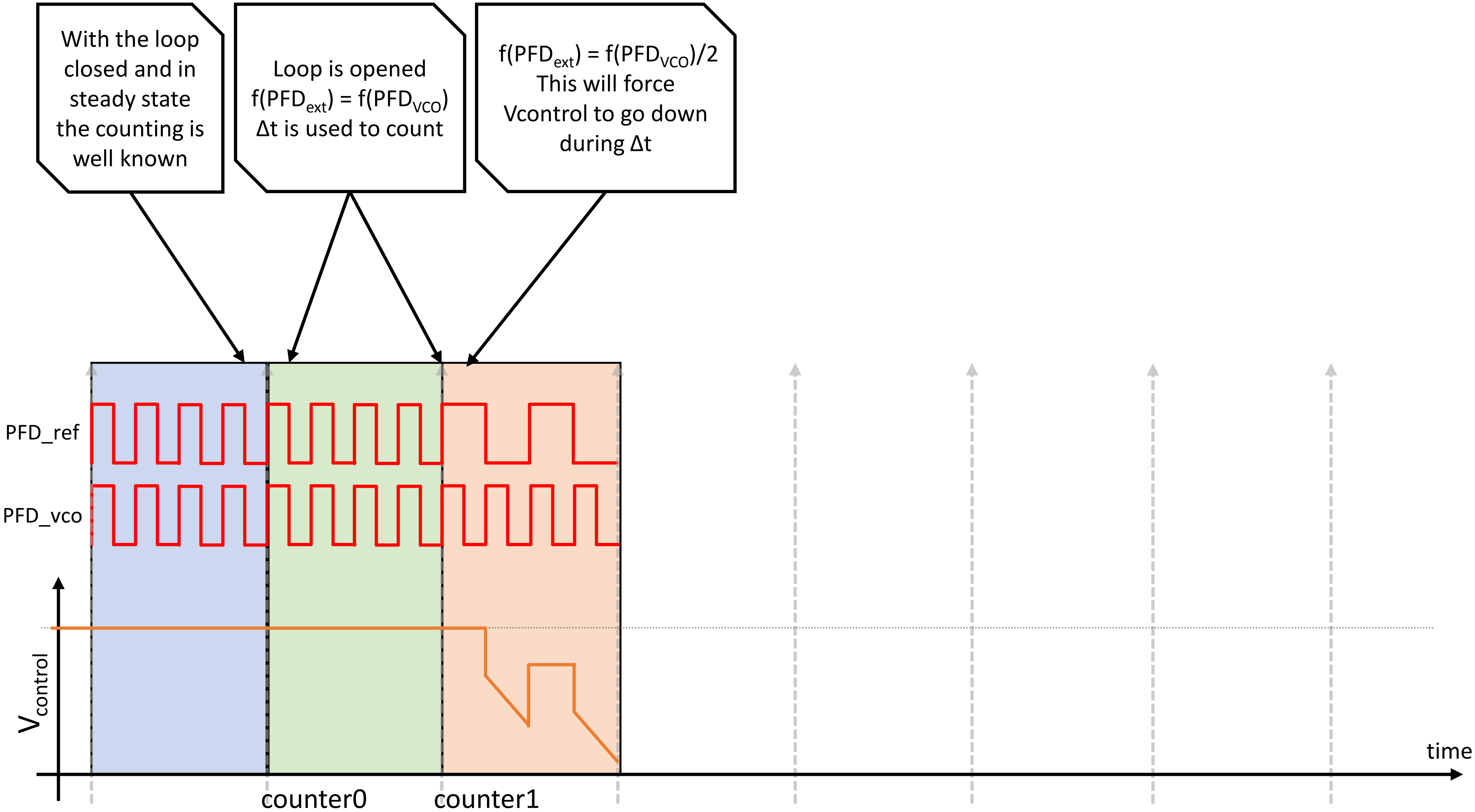


With the loop closed and in steady state the counting is well known

Loop is opened
$f(PFD_{ext}) = f(PFD_{VCO})$
Δt is used to count

# eCDRPLL’s BIST

The BIST allows to estimate mismatches from the open-loop response



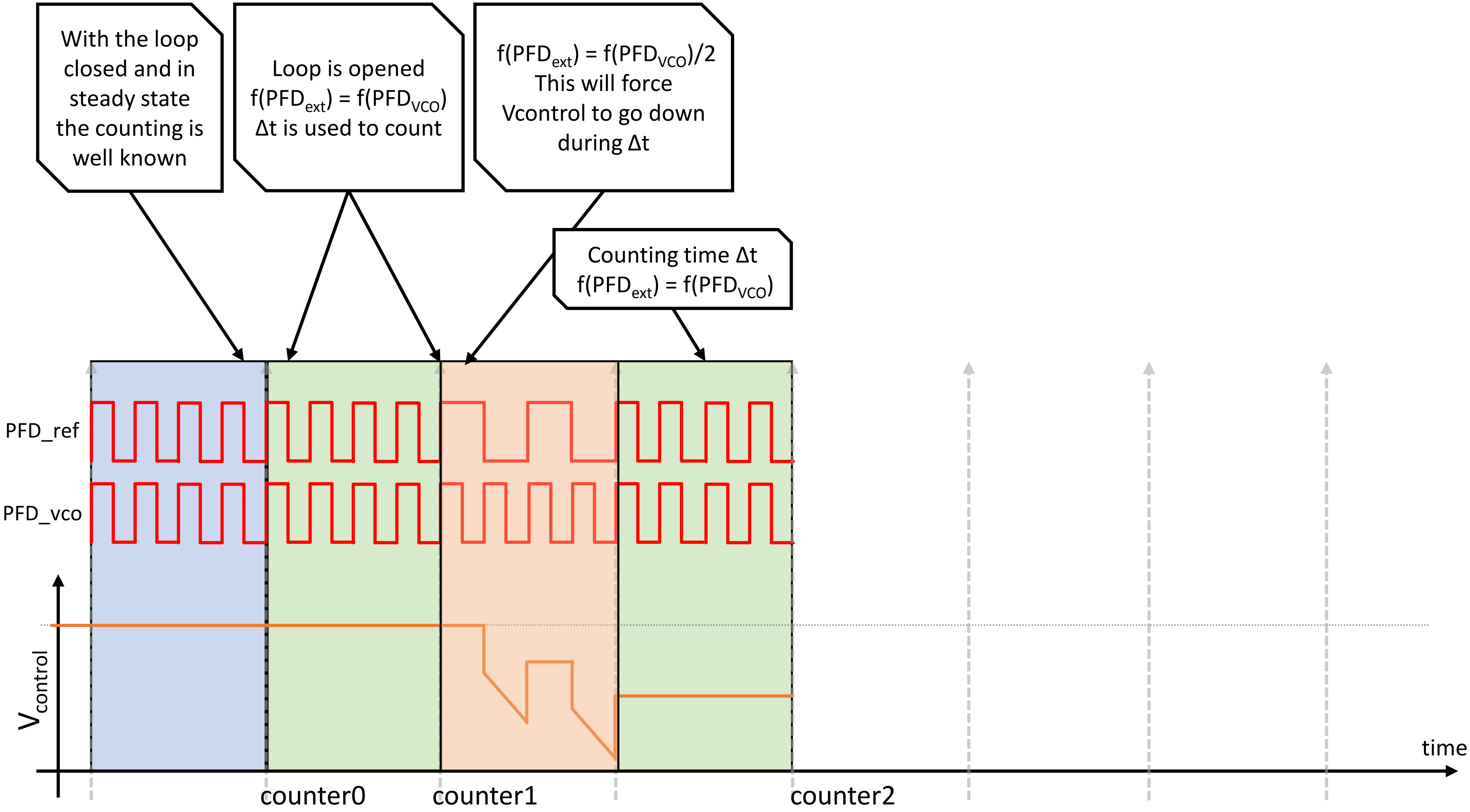
With the loop closed and in steady state the counting is well known

Loop is opened $f(PFD_{ext}) = f(PFD_{VCO})$ Δt is used to count

$f(PFD_{ext}) = f(PFD_{VCO})/2$ This will force Vcontrol to go down during Δt

# eCDRPLL's BIST

The BIST allows to estimate mismatches from the open-loop response



With the loop closed and in steady state the counting is well known

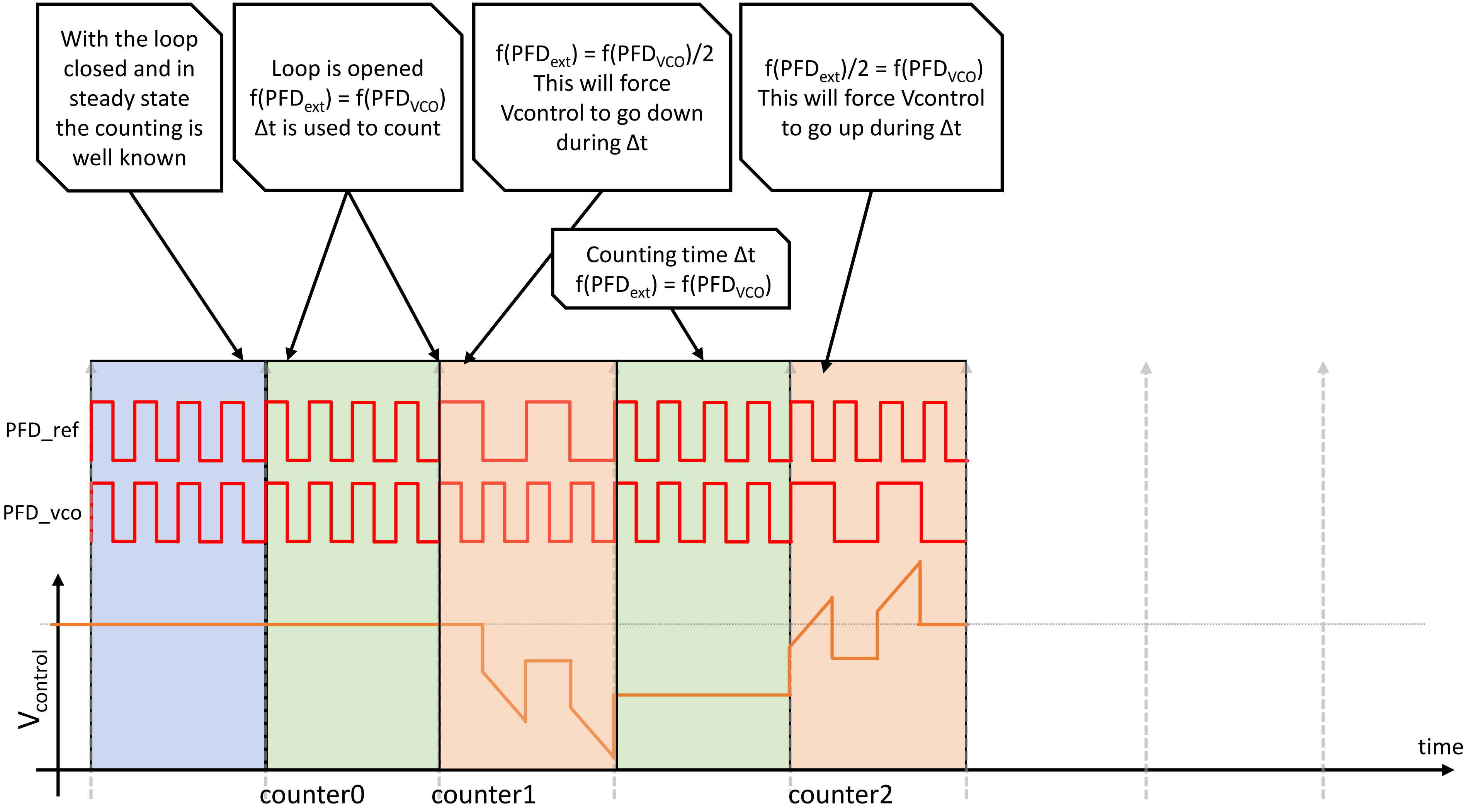
Loop is opened $f(PFD_{ext}) = f(PFD_{VCO})$ Δt is used to count

$f(PFD_{ext}) = f(PFD_{VCO})/2$ This will force Vcontrol to go down during Δt

Counting time Δt $f(PFD_{ext}) = f(PFD_{VCO})$

# eCDRPLL's BIST

The BIST allows to estimate mismatches from the open-loop response



With the loop closed and in steady state the counting is well known

Loop is opened
$f(PFD_{ext}) = f(PFD_{VCO})$
Δt is used to count

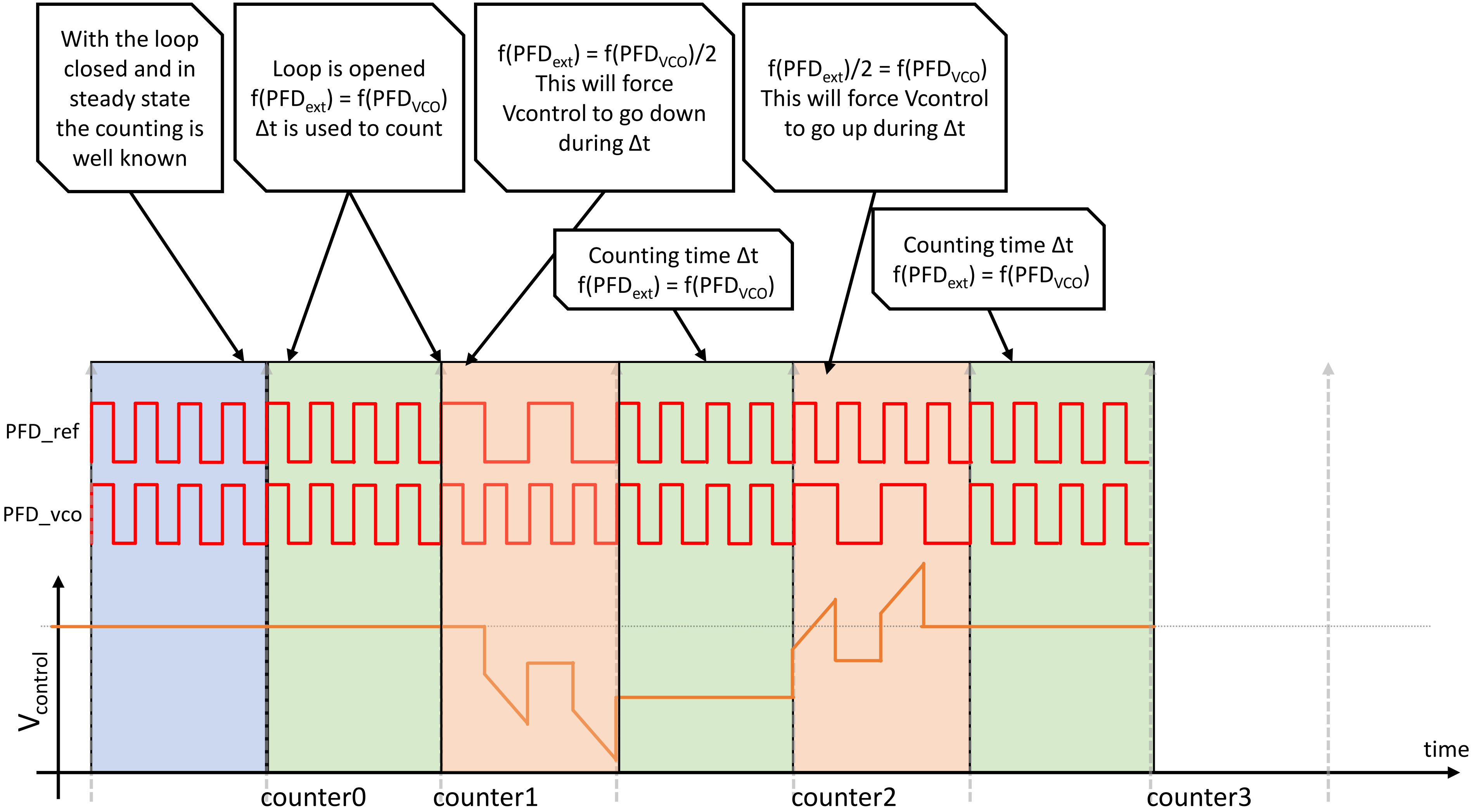
$f(PFD_{ext}) = f(PFD_{VCO})/2$
This will force Vcontrol to go down during Δt

$f(PFD_{ext})/2 = f(PFD_{VCO})$
This will force Vcontrol to go up during Δt

Counting time Δt
$f(PFD_{ext}) = f(PFD_{VCO})$

# eCDRPLL’s BIST

The BIST allows to estimate mismatches from the open-loop response



With the loop closed and in steady state the counting is well known

Loop is opened $f(PFD_{ext}) = f(PFD_{VCO})$ Δt is used to count

$f(PFD_{ext}) = f(PFD_{VCO})/2$ This will force Vcontrol to go down during Δt

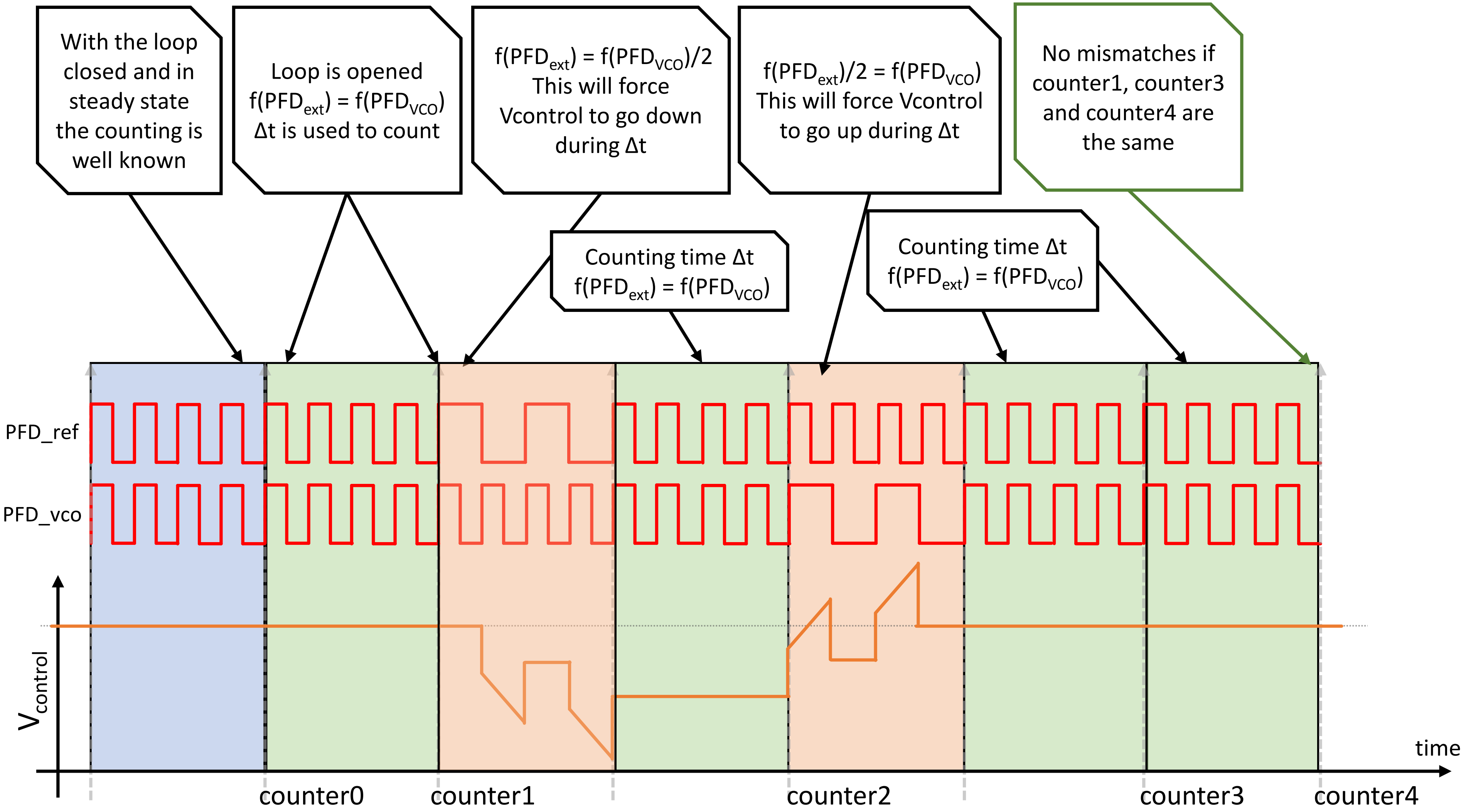
$f(PFD_{ext})/2 = f(PFD_{VCO})$ This will force Vcontrol to go up during Δt

Counting time Δt $f(PFD_{ext}) = f(PFD_{VCO})$

Counting time Δt $f(PFD_{ext}) = f(PFD_{VCO})$

# eCDRPLL's BIST

The BIST allows to estimate mismatches from the open-loop response



With the loop closed and in steady state the counting is well known

Loop is opened $f(PFD_{ext}) = f(PFD_{VCO})$ Δt is used to count

$f(PFD_{ext}) = f(PFD_{VCO})/2$ This will force Vcontrol to go down during Δt

$f(PFD_{ext})/2 = f(PFD_{VCO})$ This will force Vcontrol to go up during Δt

No mismatches if counter1, counter3 and counter4 are the same

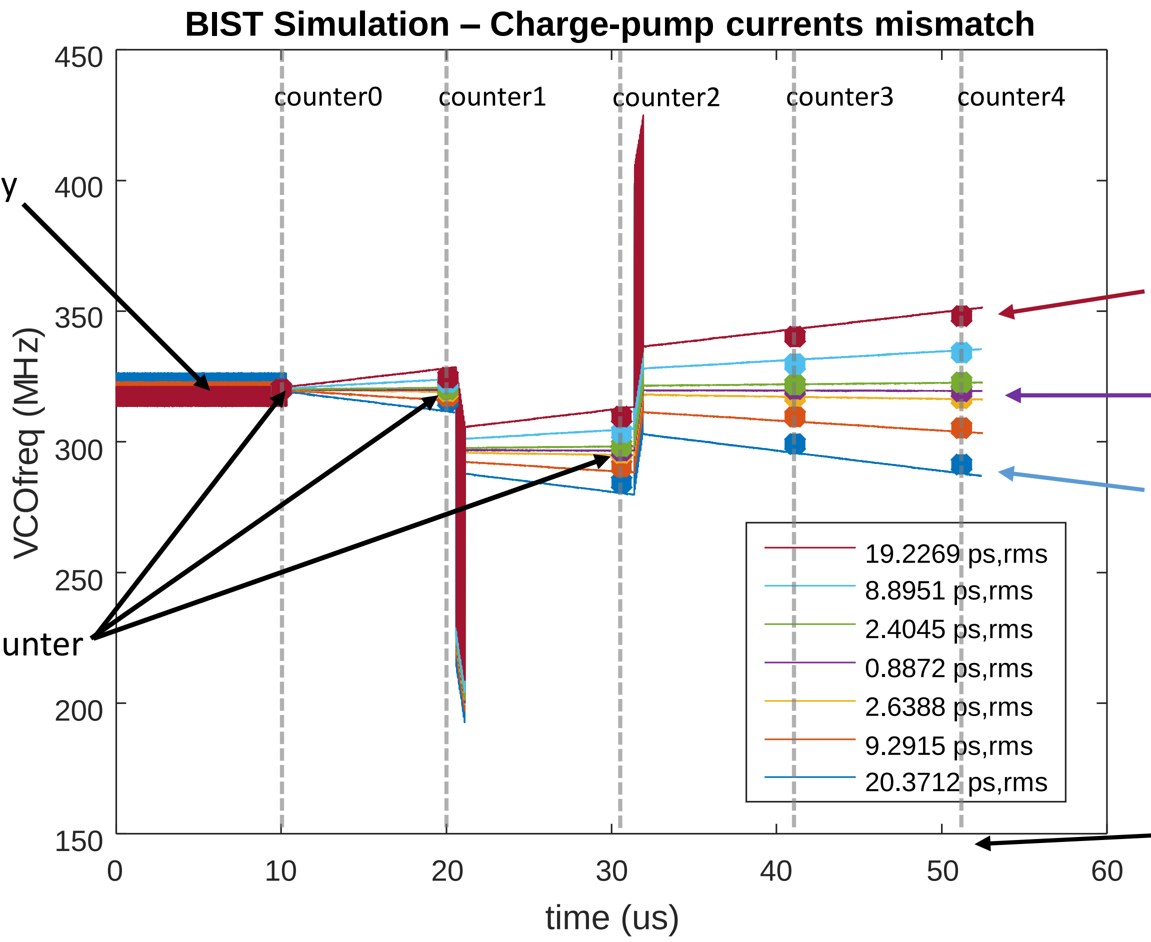
Counting time Δt $f(PFD_{ext}) = f(PFD_{VCO})$

Counting time Δt $f(PFD_{ext}) = f(PFD_{VCO})$

# eCDRPLL’s BIST

The BIST allows to estimate mismatches from the open-loop response

VCO frequency



frequencyCounter

+10%*$I_{cpLSB}$, 19 ps rms jitter

0%*$I_{cpLSB}$, 0.9 ps rms jitter

-10%* $I_{cpLSB}$, 20 ps rms jitter

Around 50 µs test time

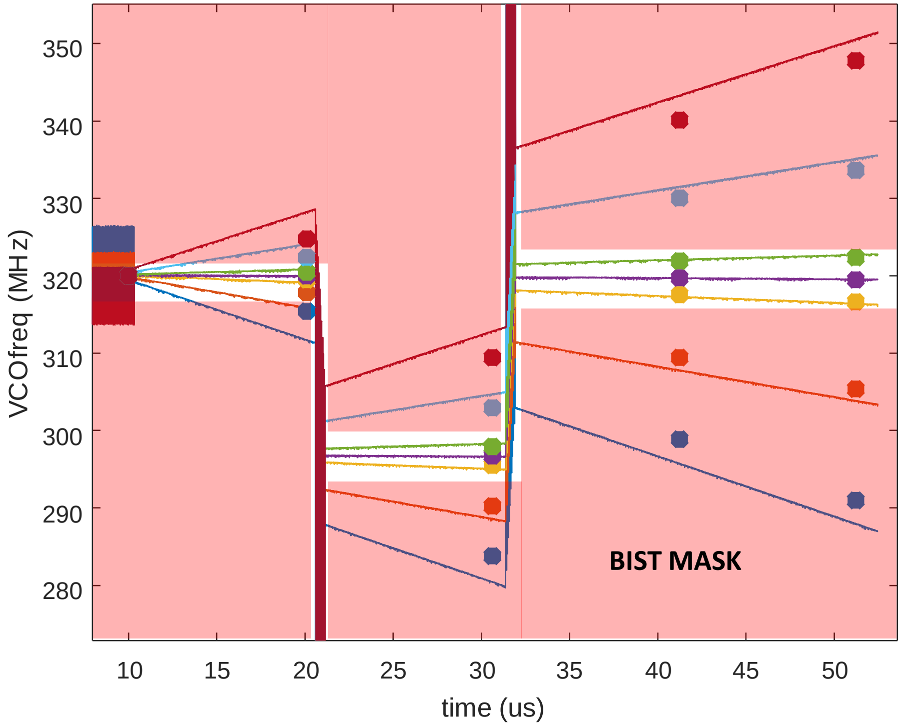
To be run at production time; relies on a statistical method

refClk=320MHz;VCO=320MHz

# eCDRPLL’s BIST

## The implemented BIST allows to estimate mismatches from the open-loop response

**BIST Simulation – Charge-pump currents mismatch**



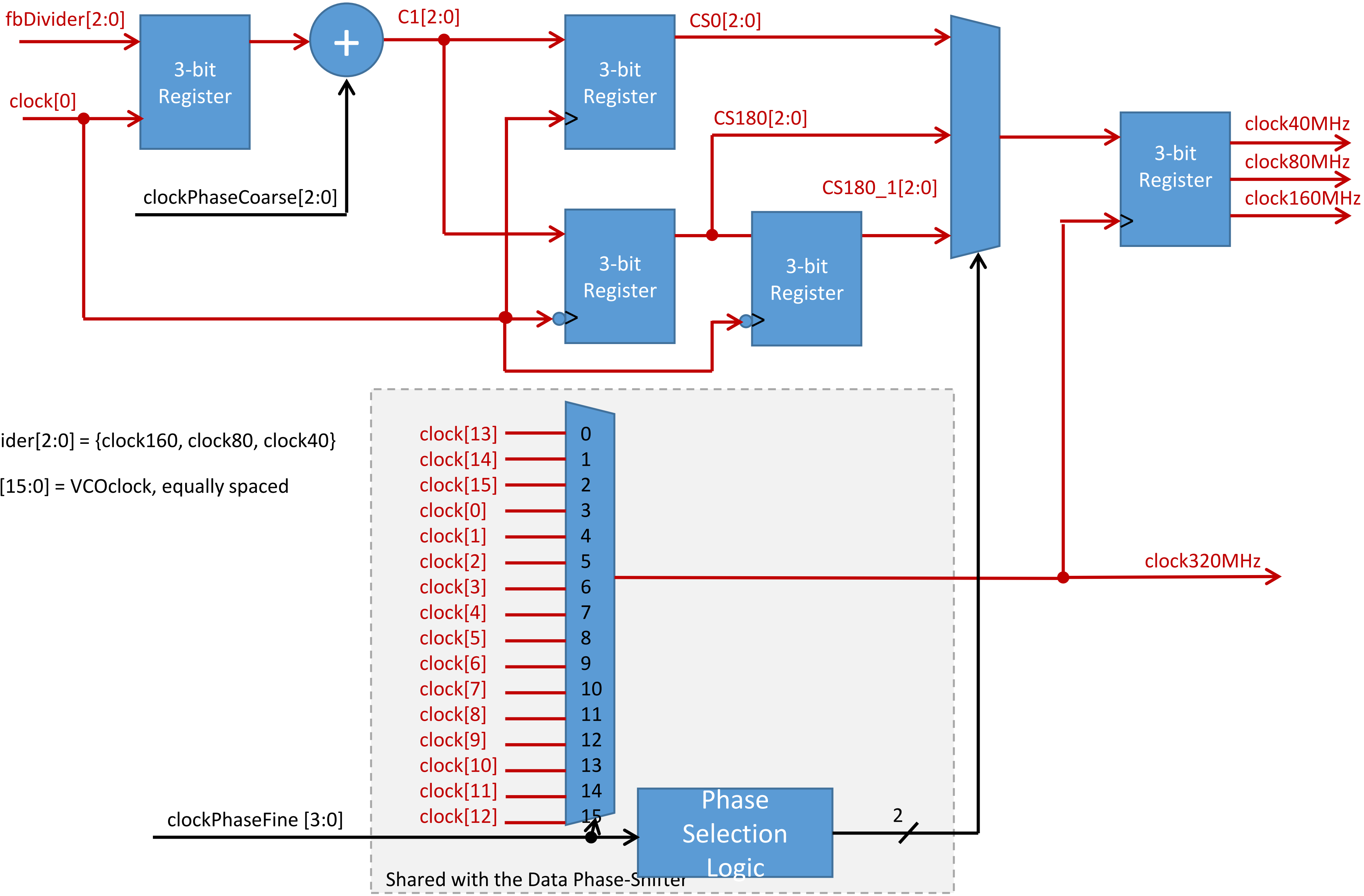
– Will be necessary to characterize some devices to create the BIST mask

| Mismatch | Store0 | Store1 | Store2 | Store3 | Store4 | Jitter rms (ps) |
|---|---|---|---|---|---|---|
| +1% | 3216 | 3220 | 2994 | 3234 | 3240 | 2.6 |
| -1% | 3216 | 3210 | 2968 | 3191 | 3182 | 2.4 |

To be run at production time; relies on a statistical method

refClk=320MHz;VCO=320MHz

# eCDRPLL Clock Phase Shifter



fbDivider[2:0] = {clock160, clock80, clock40}

clock[15:0] = VCOclock, equally spaced

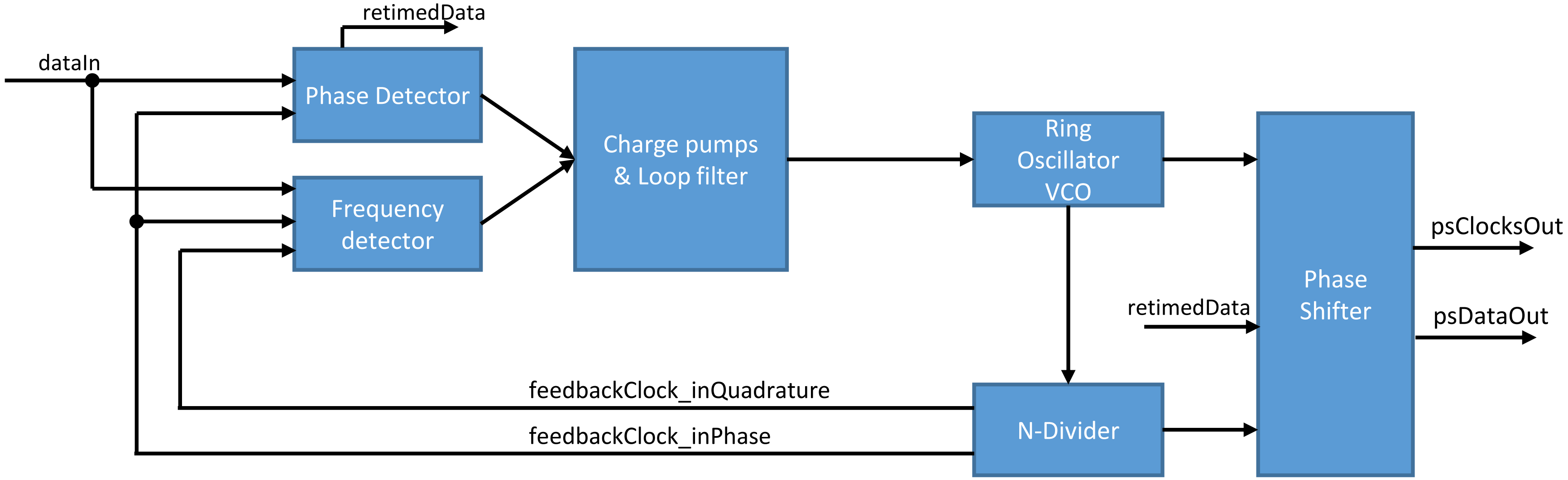
# eCDRPLL’s CDR topology

Clock and Data Recovery (CDR) can be used to recover the bit rate clock from the data transmision

- Useful for FEs when the mass budget limits the number of links

The CDR is PLL based, however

- The CDR needs to lock to random data, rendering the PFD useless
- To overcome this the eCDRPLL employs a Phase Detector
  - The phase detector only takes a decision when there are dataIn transtitions
- A Frequency Detector is added to aid frequency locking (±25% of data rate)
- The remaining blocks are shared

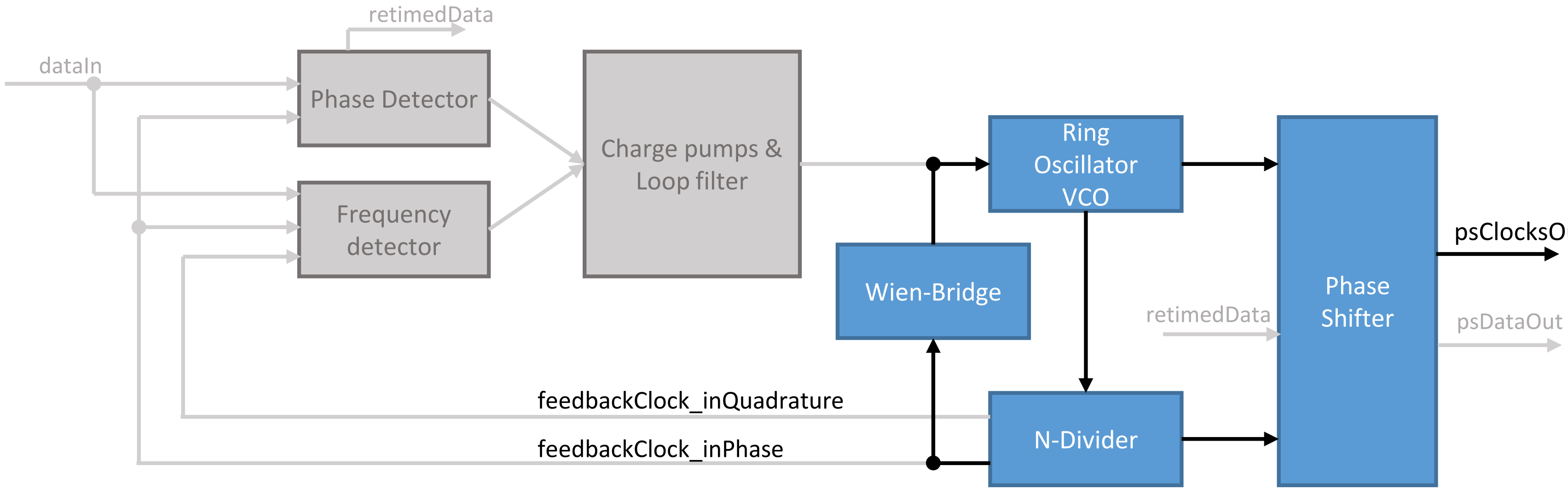


# eCDRPLL's CDR topology

## Locking mechanism

- Necessary to bring the VCO within a few percent of the data rate
  - Frequency detector has limited locking range
- Two methods are implemented
  - Initiate the loop in PLL mode by supplying an external clock
  - Reference-less calibration method by means of a Wien-Bridge
    - Necessary to disable the phase detectors, charge pumps and the loop filter

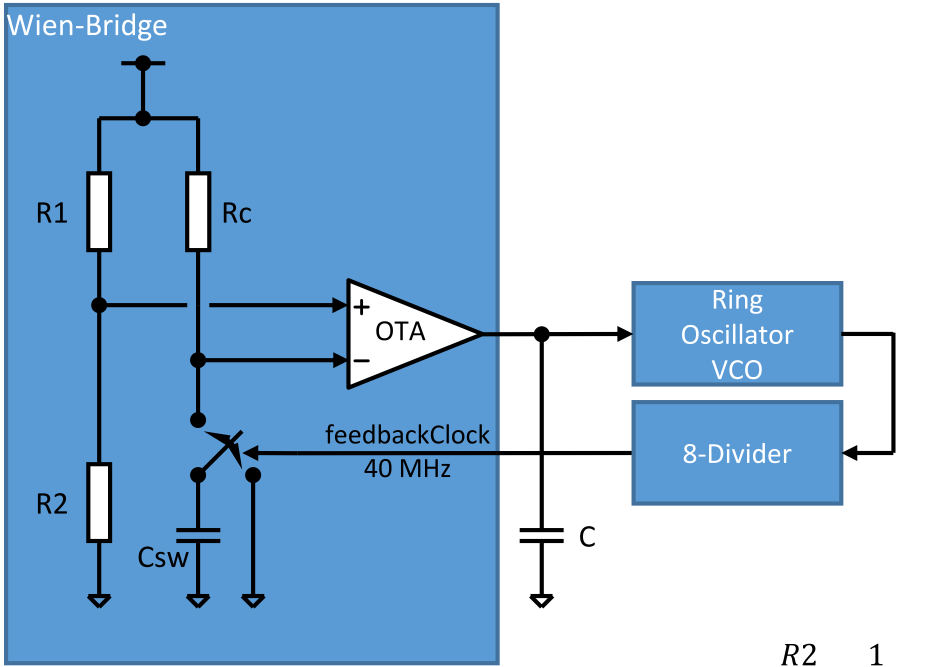
- After locking mechanism has finished
  - The FD will have to acquire frequency lock, and only after, the PD will be able to acquire lock



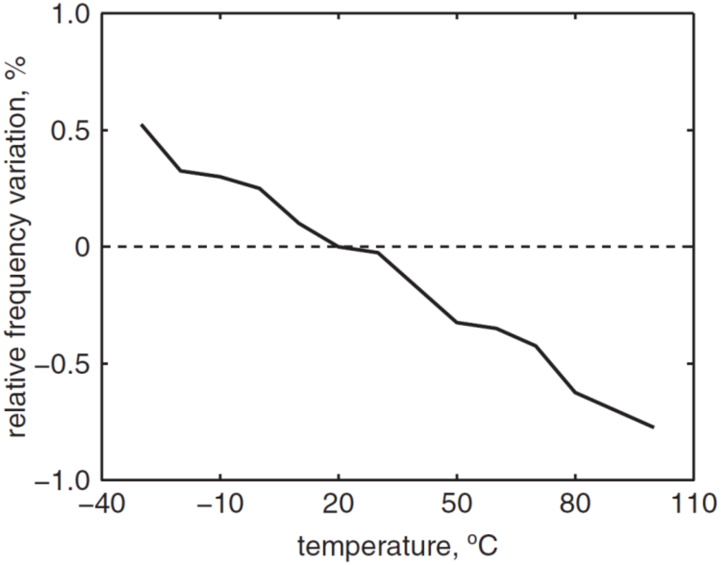
# eCDRPLL’s CDR Reference-less locking

## Reference-less locking mechanism

- The method is based on balancing the DC voltage of a Wien-Bridge
- A switched capacitor acts as a resistance inversely proportional to the frequency
- There is no voltage supply dependance as the the equilibrium voltage is given by R1/R2
- The loop makes the circuit temperature and supply tolerant
- Radiation also shown to have a negligible impact on the circuit

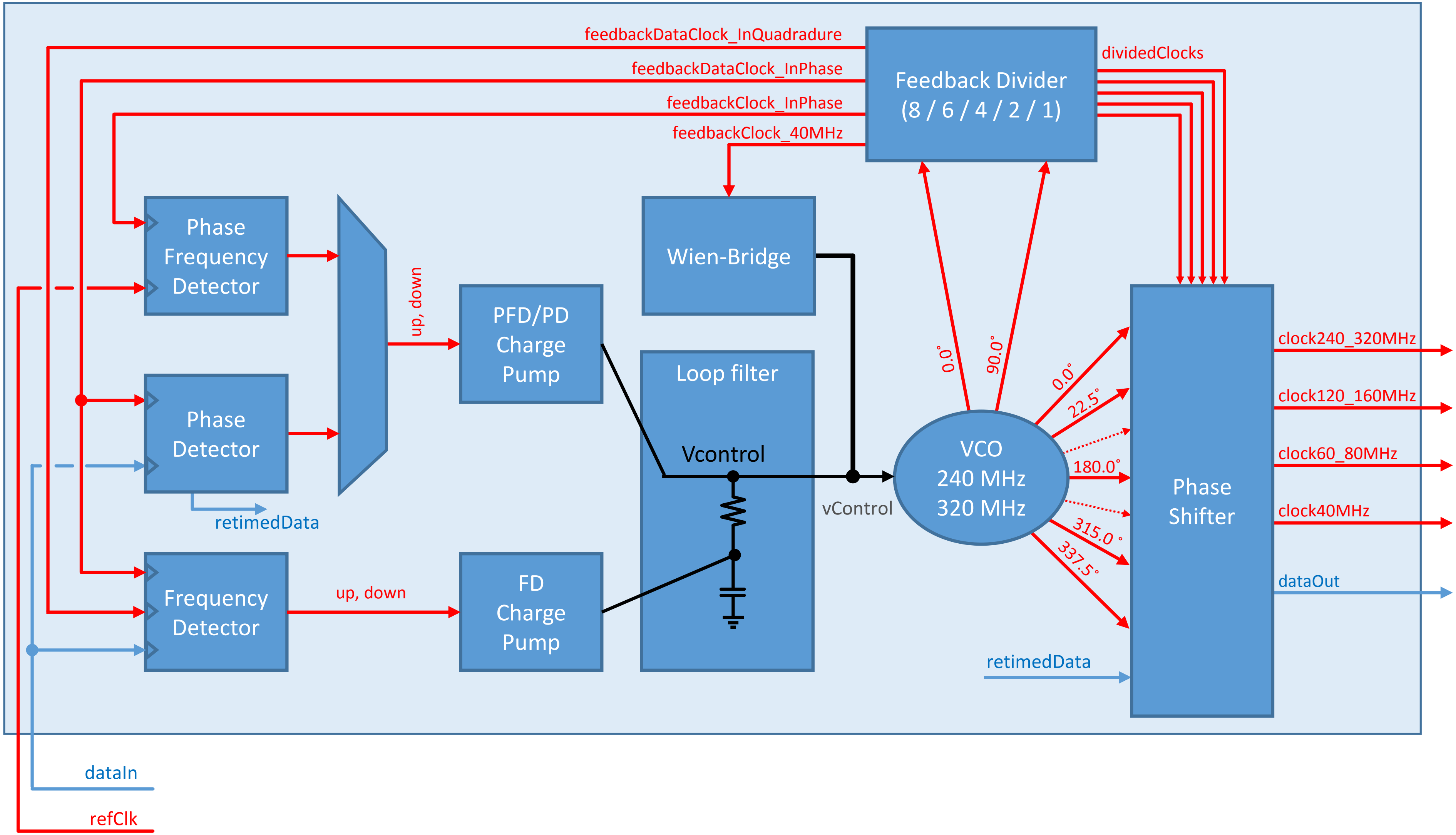


$$f_{sw,eq} = \frac{R2}{R1}\frac{1}{R_c \cdot C_{sw}}$$

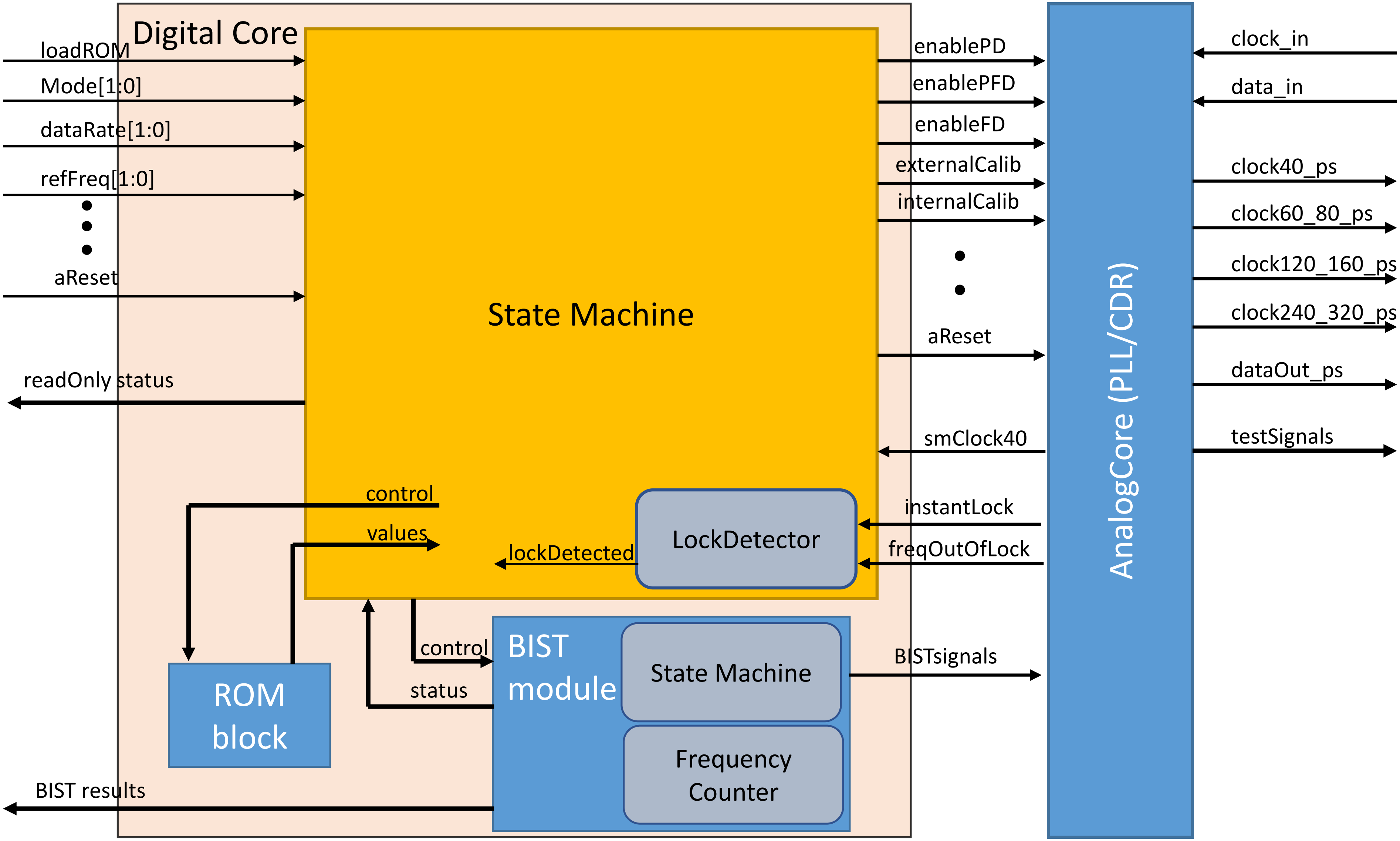


F. Tavernier, Wien bridge-based calibration circuit for CDR applications

# eCDRPLL Analog Block Diagram



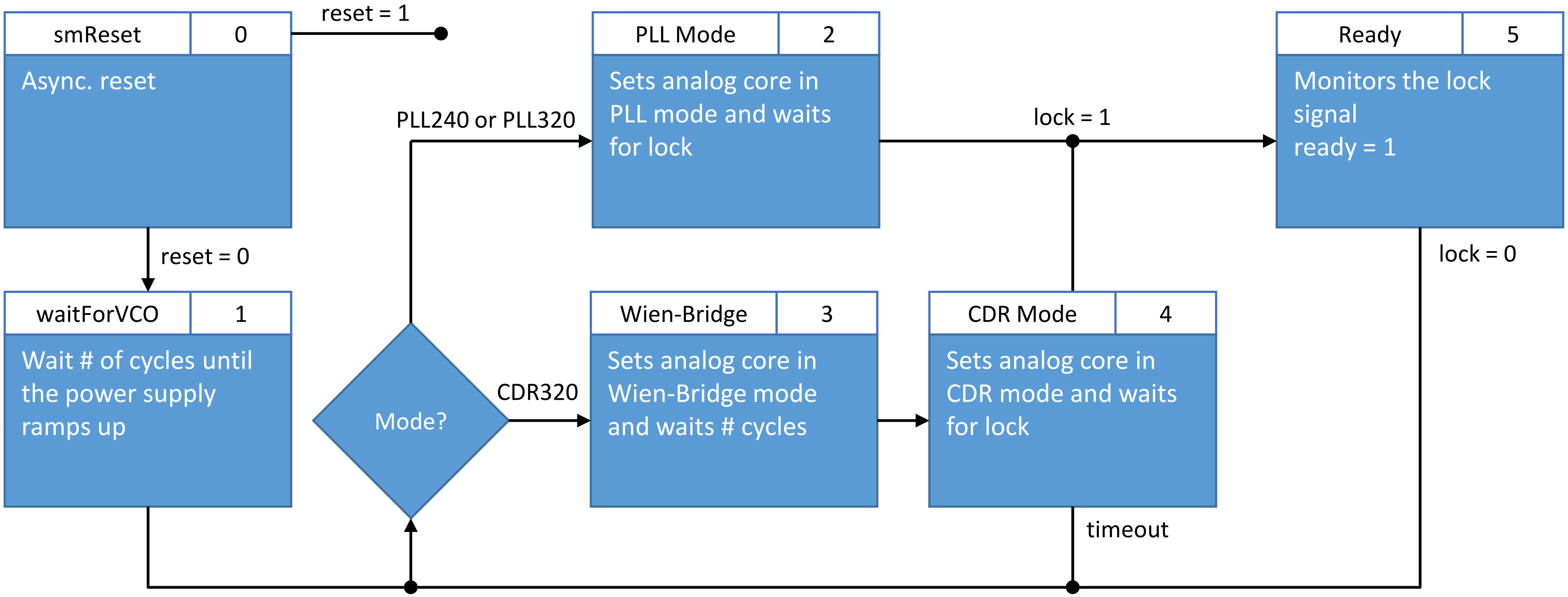
# eCDRPLL High Level Block Diagram



# eCDRPLL (Digital Core)

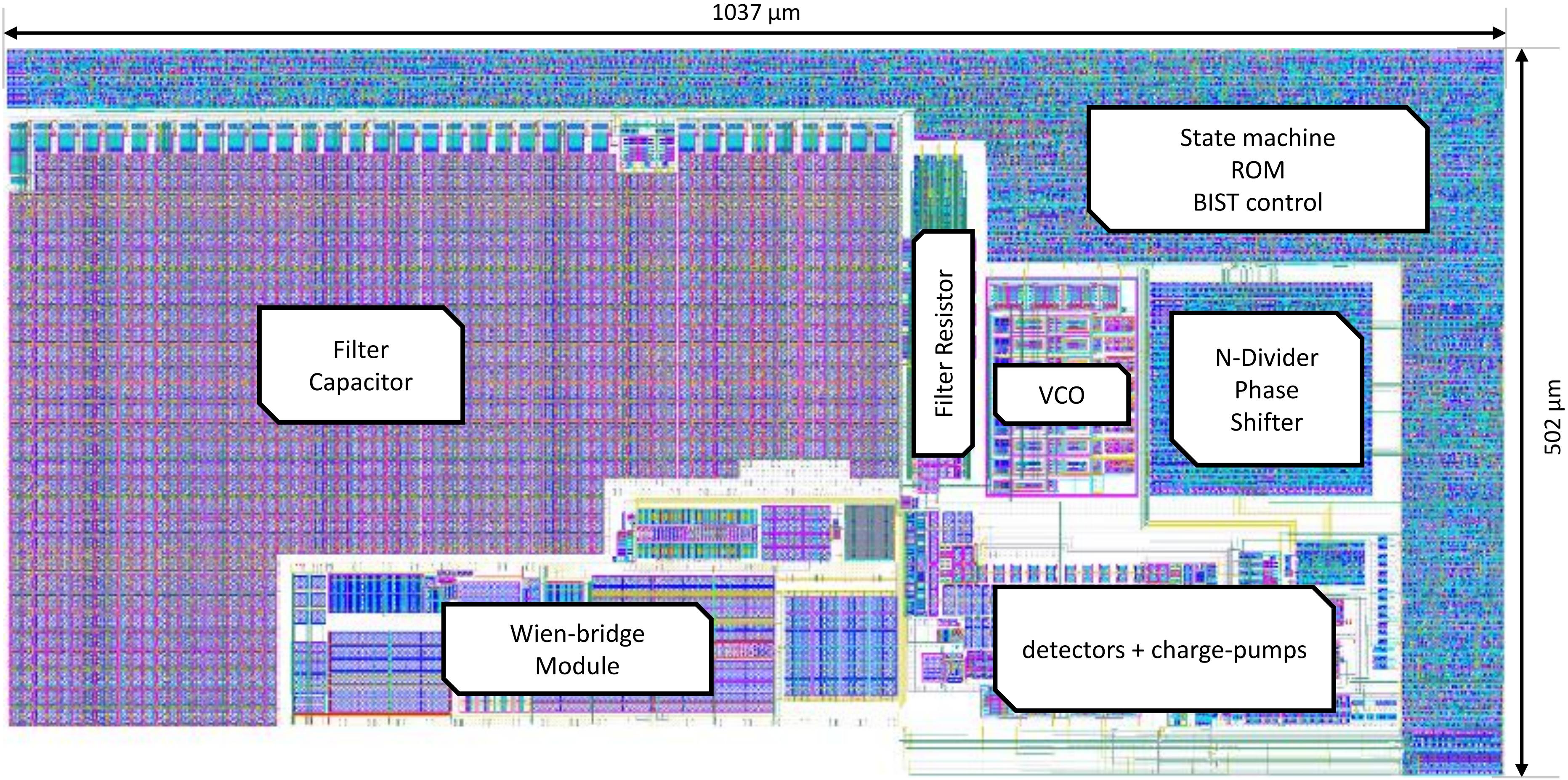
The digital core is comprised of

- State machine
  - Power up state machine
  - If the user has not preconfigured the loop filter parameters, the state machine will load the read-only values at power up
  - Enables the reference-less locking when in CDR mode
  - Has built-in watchdog to recover from loses of lock



smReset 0
Async. reset
reset = 1
reset = 0
waitForVCO 1
Wait # of cycles until the power supply ramps up
Mode?
PLL240 or PLL320
CDR320
PLL Mode 2
Sets analog core in PLL mode and waits for lock
Wien-Bridge 3
Sets analog core in Wien-Bridge mode and waits # cycles
CDR Mode 4
Sets analog core in CDR mode and waits for lock
lock = 1
timeout
Ready 5
Monitors the lock signal
ready = 1
lock = 0

# eCDRPLL floorplan



- Digital on Top (highVT library used)
- Power mesh grid comprised of M5 and M6

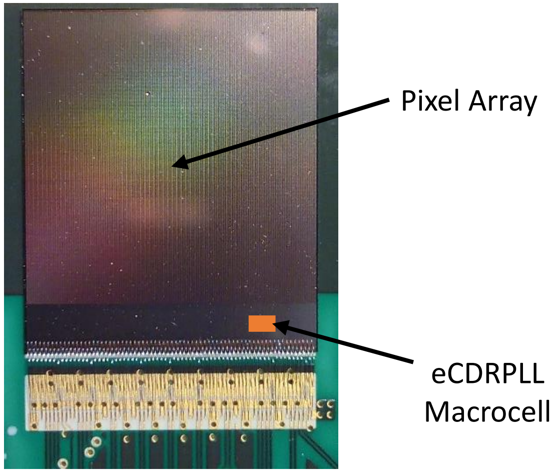
# The eCDRPLL – Experimental Results

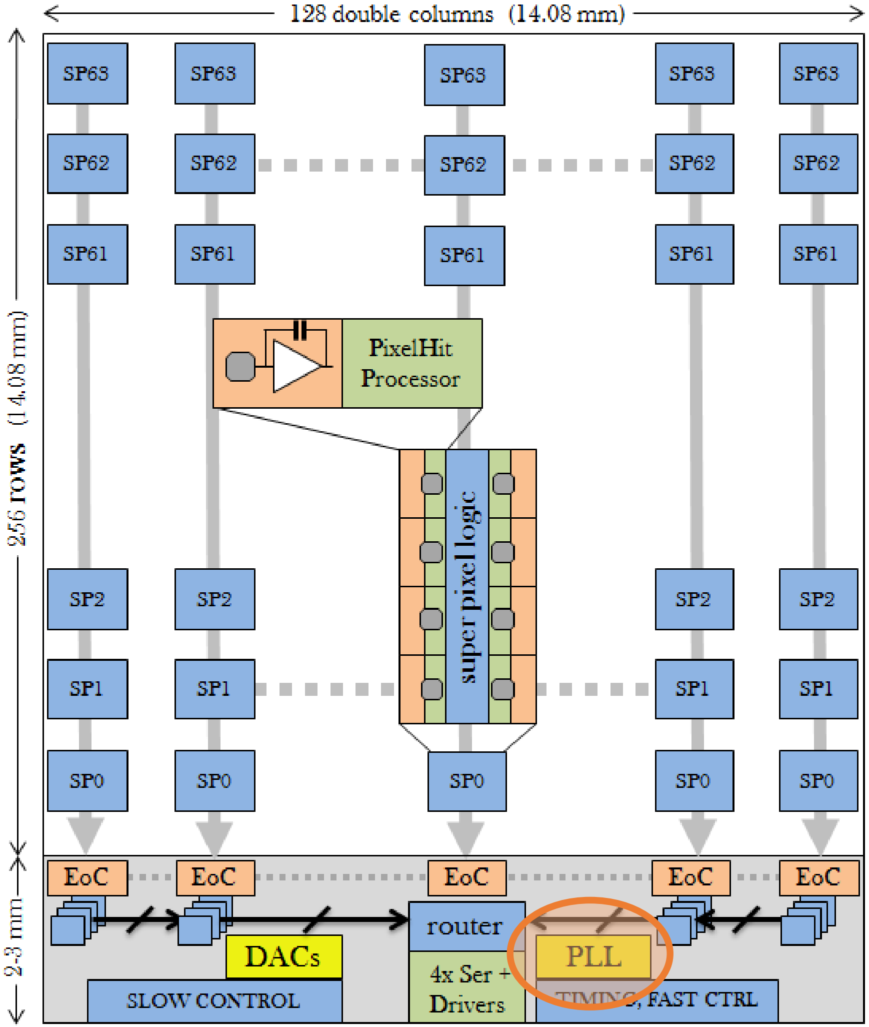
# eCDRPLL

## Velopix ASIC

- The eCDRPLL was instantiated in the Velopix ASIC
  - Used to synthesize 40, 80, 160 and 320 MHz clocks
  - PLL mode (VCO@320MHz)
  - LHC-40 MHz reference clock
  - Separated power domain for the macrocell

- The eCDRPLL macrocell was characterized as a sub-circuit of the Velopix ASIC

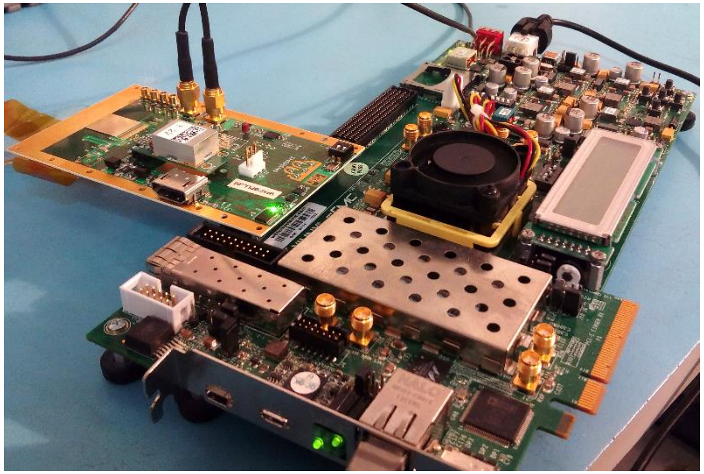


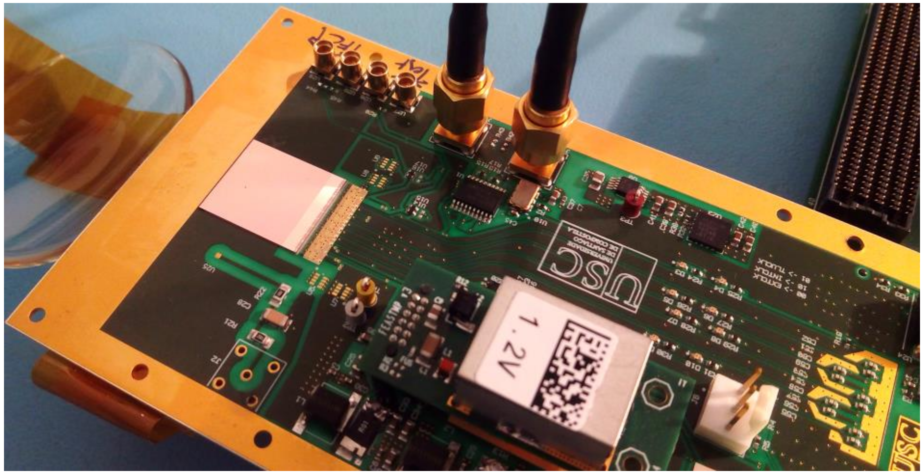


# eCDRPLL

## Test board

- External LVDS clock input
- Dedicated SLVS test clock output
- Uses a FPGA-based readout system (SPIDR) developed at NIKHEF (NL) to control the Velopix
- Chipboard developed at Univ. Santiago de Compostella
- All power supplies externally tied together to an 1.2V FEASTMP





# Some Results

## Some Results

- PLL Mode works across all frequency division ratios
- CDR Mode works across all data rate division ratios and locking methods
- PLL BIST
  - BIST passed behavioural test but there is not enough statistics yet
- Radiation campaign
  - Scheduled to Dec 2016 in a Velopix parasitic run

| | PLL<br>40 MHz input (÷8) | PLL<br>320 MHz input (÷1) | CDR<br>80 Mb/s input (÷4) | CDR<br>320 Mb/s input (÷1) |
|---|---|---|---|---|
| Phase shifter | Max {INL, DNL} < 0.6 LSB on all modes | | | |
| Locking range | 7 to 77 (MHz) | 27 to 436 (MHz) | 70 – 82 (Mb/s)[1] | 216 – 370 (Mb/s)[1] |
| Power consumption (mW) | 13.4 | 15.4 | To be tested | To be tested |
| Jitter (rms)<br>(320 MHz clock output/dataout) | 7.3 ps[2] | 1.9 ps[2] | 48.9 ps clock out[3]<br>4.2 mUI data out[3] | 13.1 ps clock out[3]<br>4.0 mUI data out[3] |

[1] Reference-less locking method

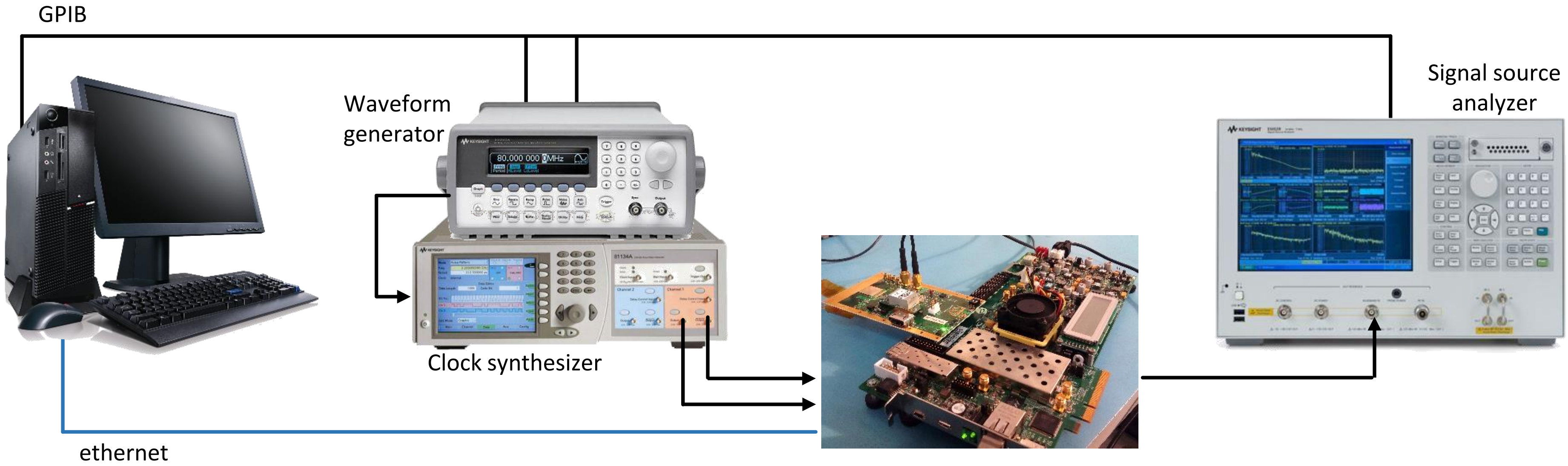
[2] 1-ps total jitter rms input clock; phase noise measurement with 10kHz to 10MHz BW

[3] 1mUI rms PRBS-7 data input; time domain measurement with jitter analysis using 2nd order PLL recClk with default BW

# eCDRPLL’s PLL performance

## Test setup for Jitter measurement

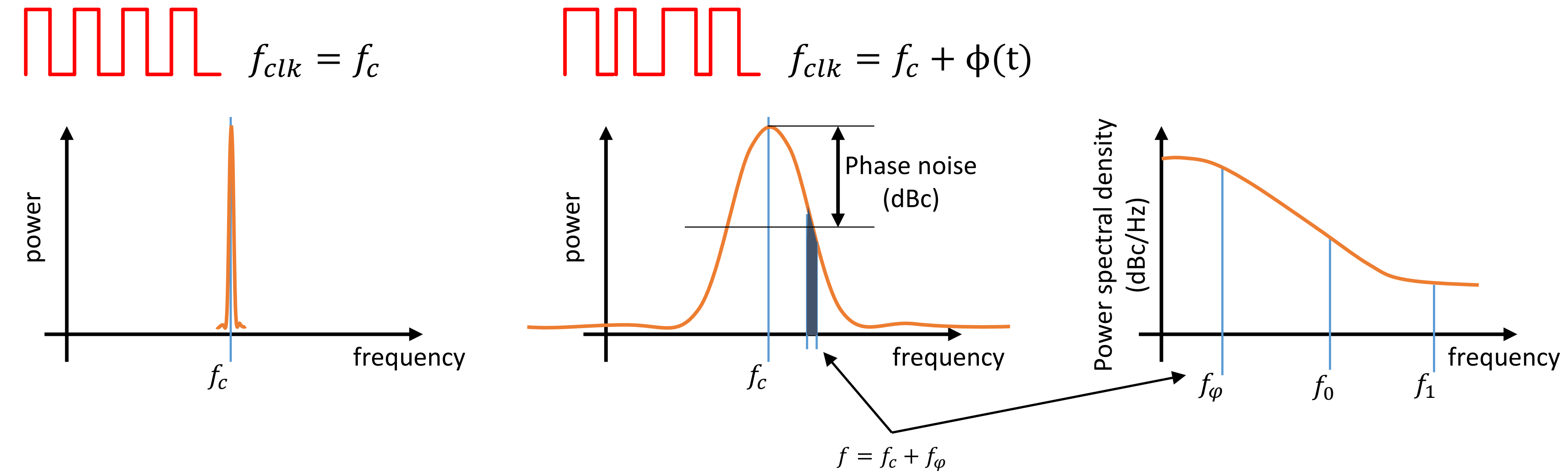
- The jitter measurement setup was done using
  - Clock source: Agilent 81133A, 3.35GHz Pulse Pattern Generator (sub 1-ps rms jitter output)
    - White noise jitter was injected by modulating the output clock phase
    - An arbitrary waveform generator (Agilent 33250A) was used to inject white noise with different amplitudes
  - Signal Source Analyzer: Agilent E5052B
    - The jitter was measured by integrating the phase noise measurements within a 10kHz to 10MHz bandwidth
- The measurements were automatized with a full parameter scan in mind



# Review – what is phase noise?

## What is phase noise?

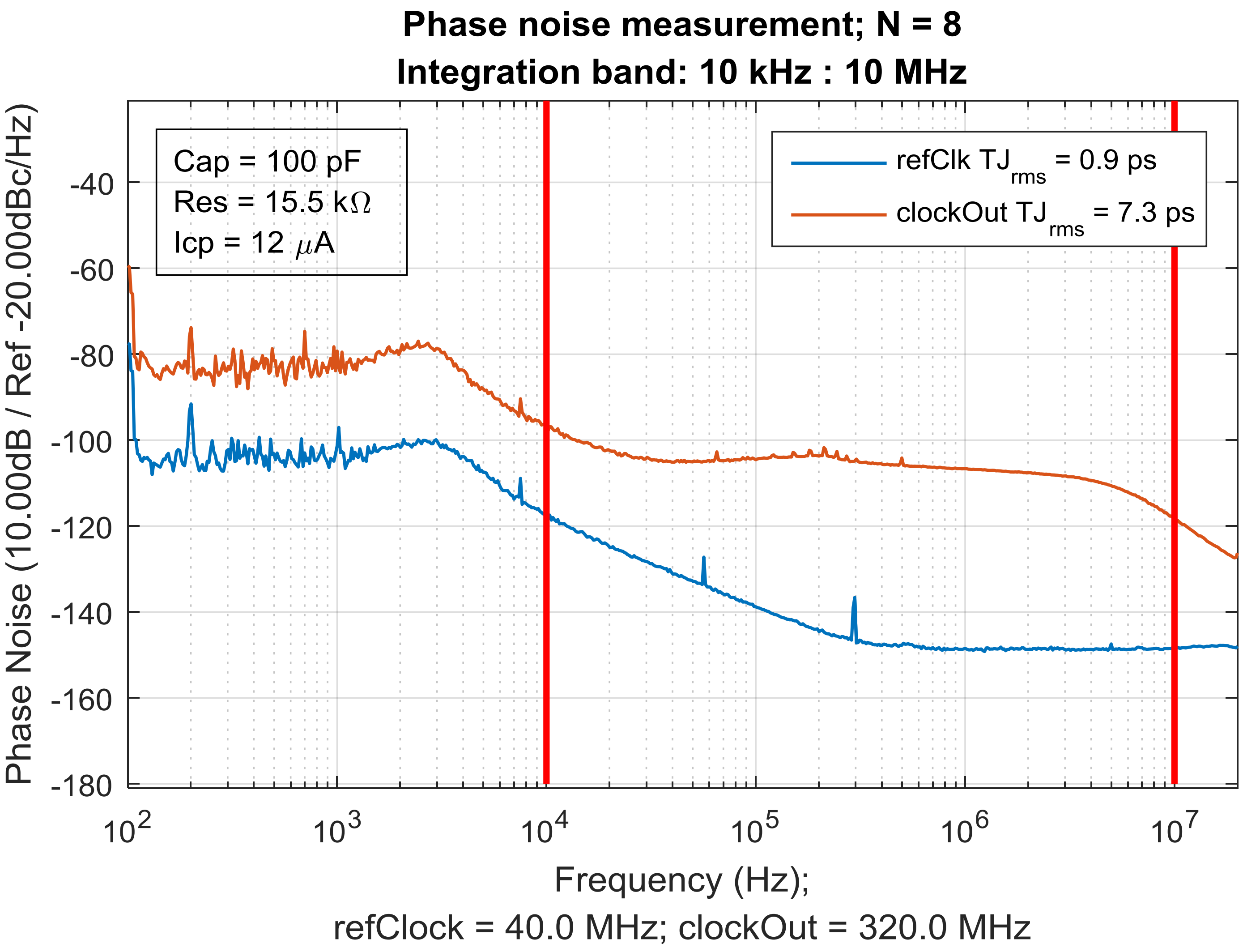
- The phase noise is the frequency domain representation of the fluctuations in the phase of a signal - it measures the phase stability of a signal
- It is equivalent to a jitter measurement (which is done in time domain) after integrating the phase noise
- Phase noise is specified in dBc/Hz at a given offset in relation to the carrier frequency



$$TotalJitter_{rms} = \frac{\int_{f0}^{f1} PhaseNoise \cdot df}{2\pi f_{carrier}\ [Hz]}\ (s)$$

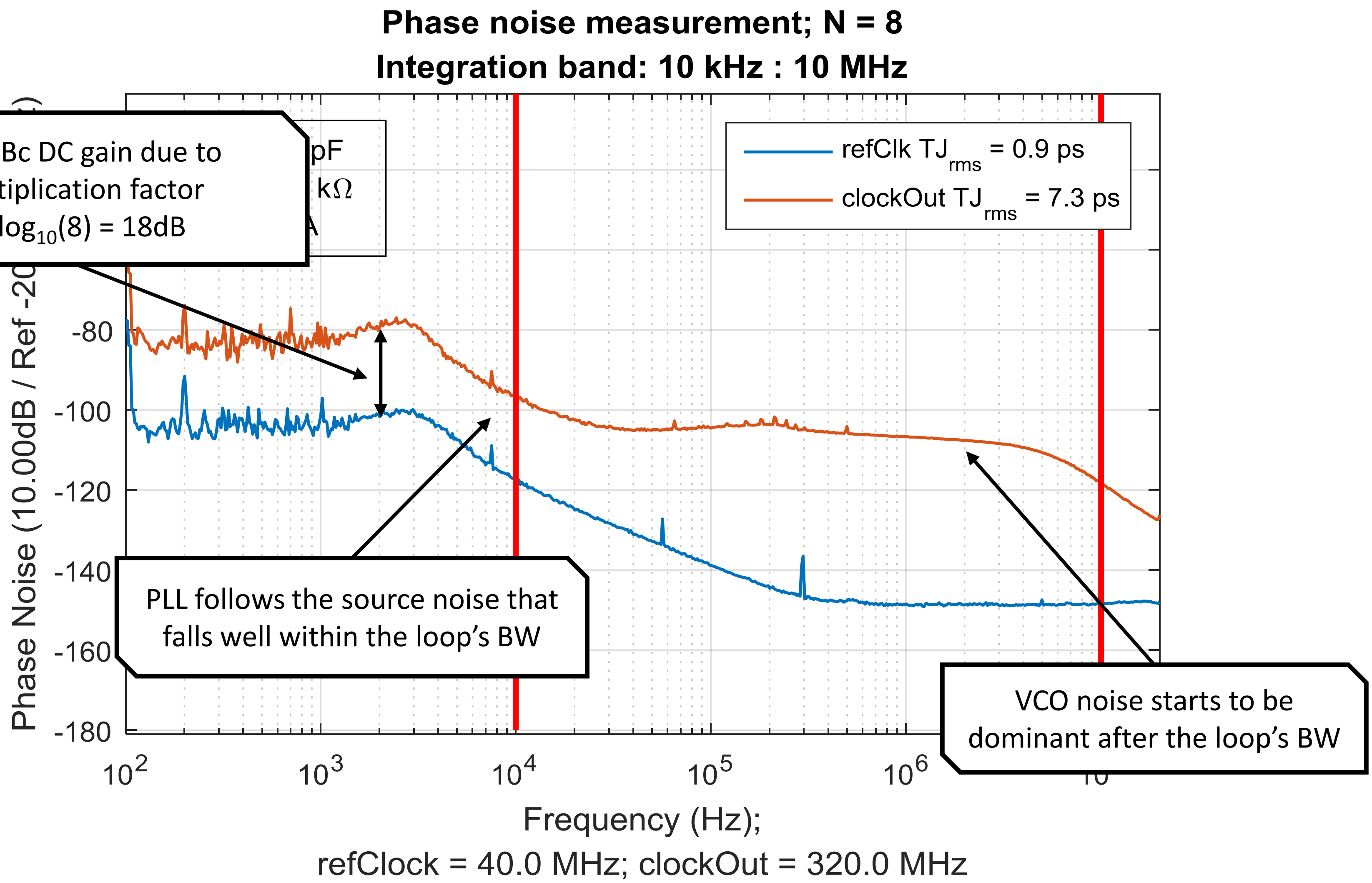
dBc = level in dB relative to the carrier

# eCDRPLL PLL Jitter Characterization



DUT: refClk=40MHz, VCO@320MHz, clkOut=320MHz

# eCDRPLL PLL Jitter Characterization



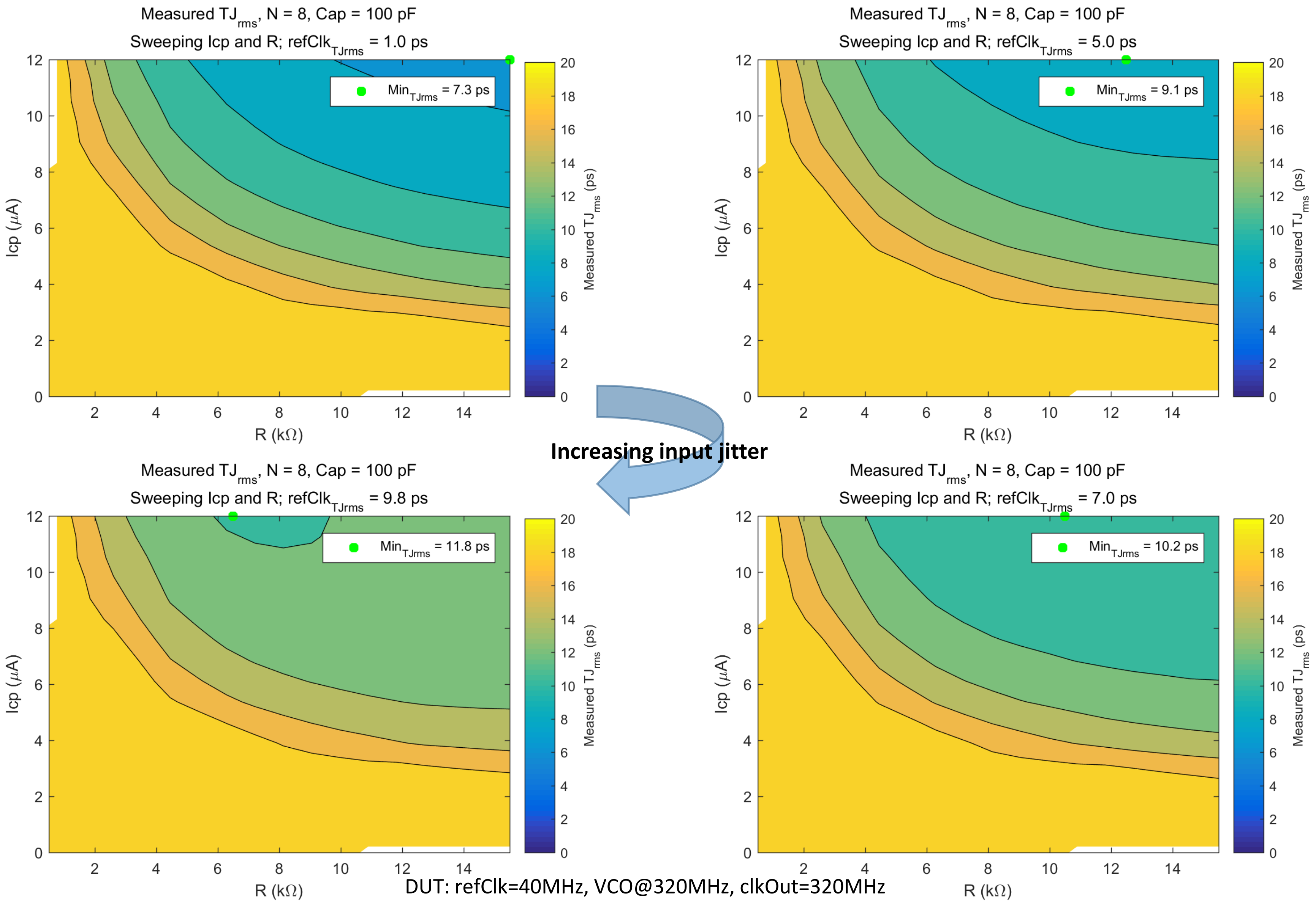
+18 dBc DC gain due to multiplication factor $20\log_{10}(8) = 18\text{dB}$

PLL follows the source noise that falls well within the loop's BW

VCO noise starts to be dominant after the loop's BW

DUT: refClk=40MHz, VCO@320MHz, clkOut=320MHz

# eCDRPLL PLL Jitter Characterization



Measured $TJ_{rms}$, N = 8, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 1.0 ps

Measured $TJ_{rms}$, N = 8, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 5.0 ps

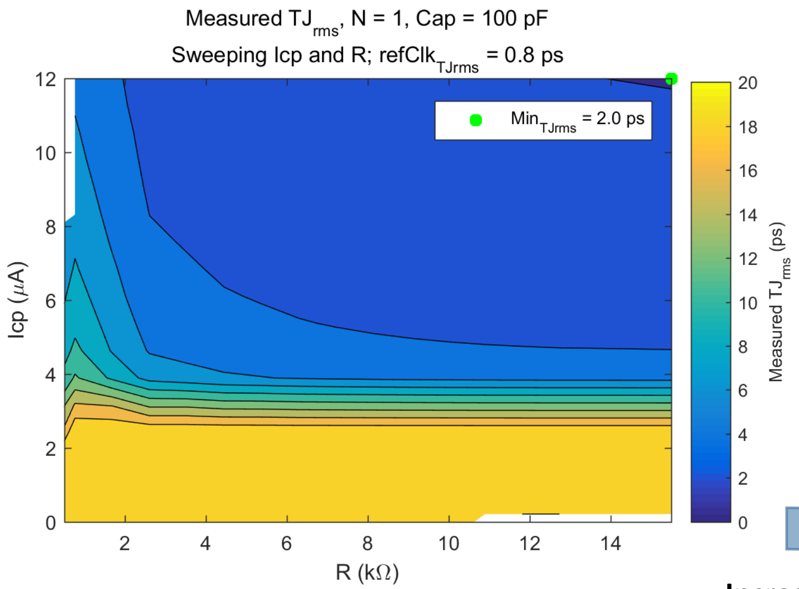
Measured $TJ_{rms}$, N = 8, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 9.8 ps

Measured $TJ_{rms}$, N = 8, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 7.0 ps

**Increasing input jitter**

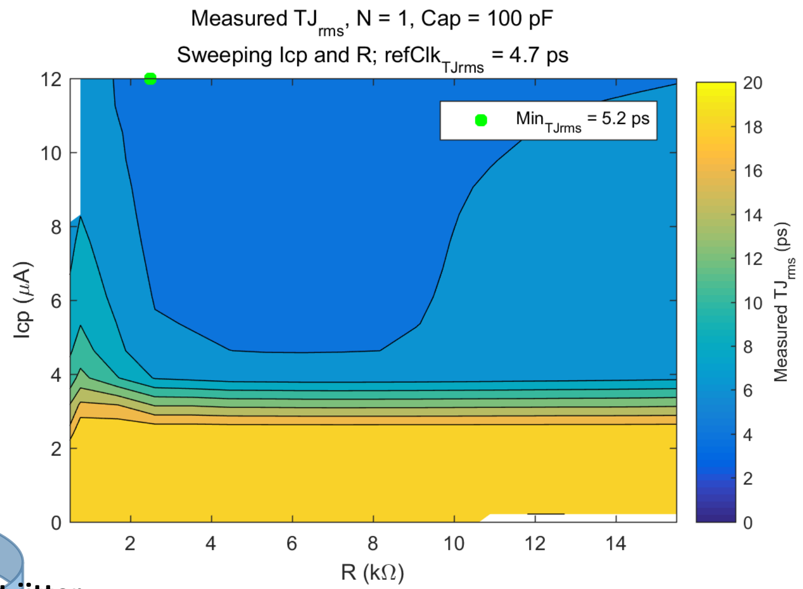
DUT: refClk=40MHz, VCO@320MHz, clkOut=320MHz

# eCDRPLL PLL Jitter Characterization



Measured $TJ_{rms}$, N = 1, Cap = 100 pF

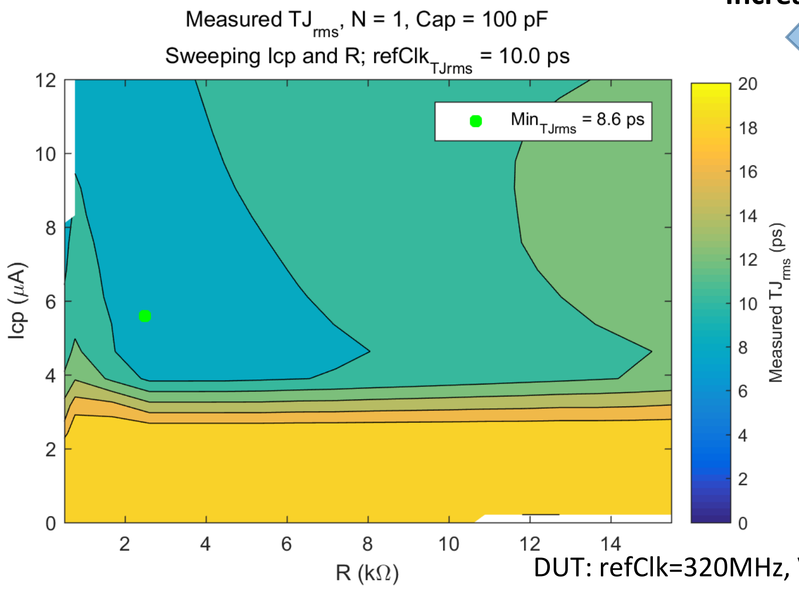
Sweeping Icp and R; $refClk_{TJrms}$ = 0.8 ps



Measured $TJ_{rms}$, N = 1, Cap = 100 pF

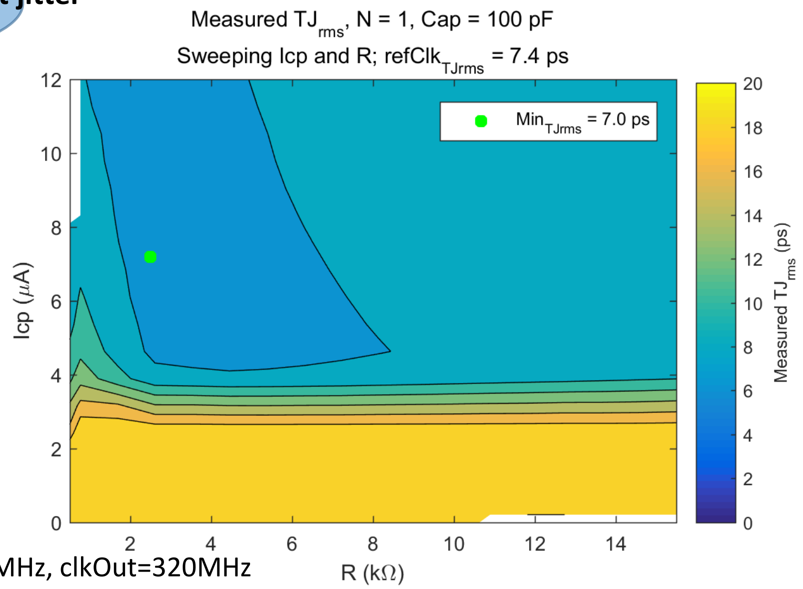
Sweeping Icp and R; $refClk_{TJrms}$ = 4.7 ps

**Increasing input jitter**



Measured $TJ_{rms}$, N = 1, Cap = 100 pF

Sweeping Icp and R; $refClk_{TJrms}$ = 10.0 ps

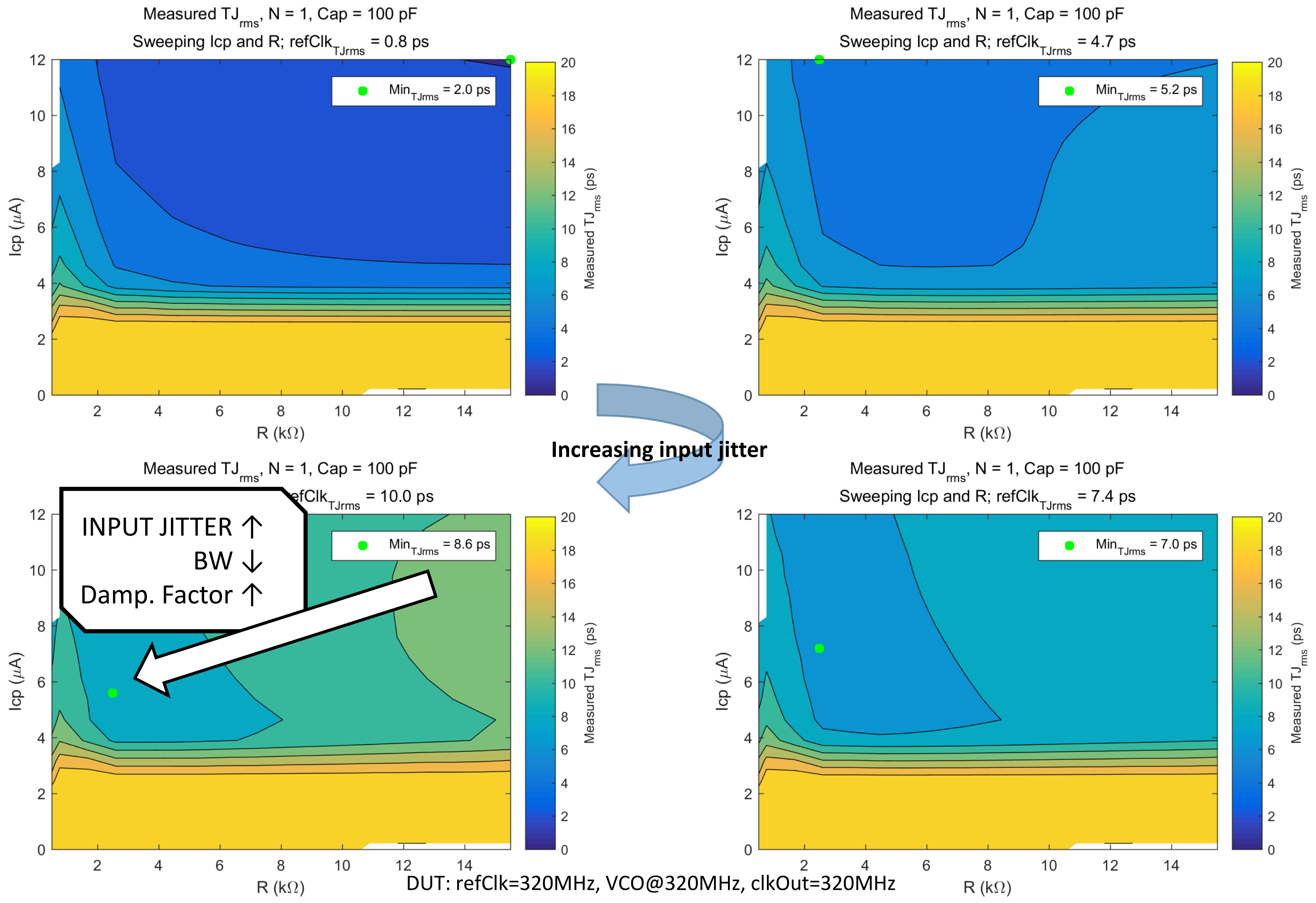


Measured $TJ_{rms}$, N = 1, Cap = 100 pF

Sweeping Icp and R; $refClk_{TJrms}$ = 7.4 ps

DUT: refClk=320MHz, VCO@320MHz, clkOut=320MHz

# eCDRPLL PLL Jitter Characterization



Measured $TJ_{rms}$, N = 1, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 0.8 ps

Measured $TJ_{rms}$, N = 1, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 4.7 ps

**Increasing input jitter**

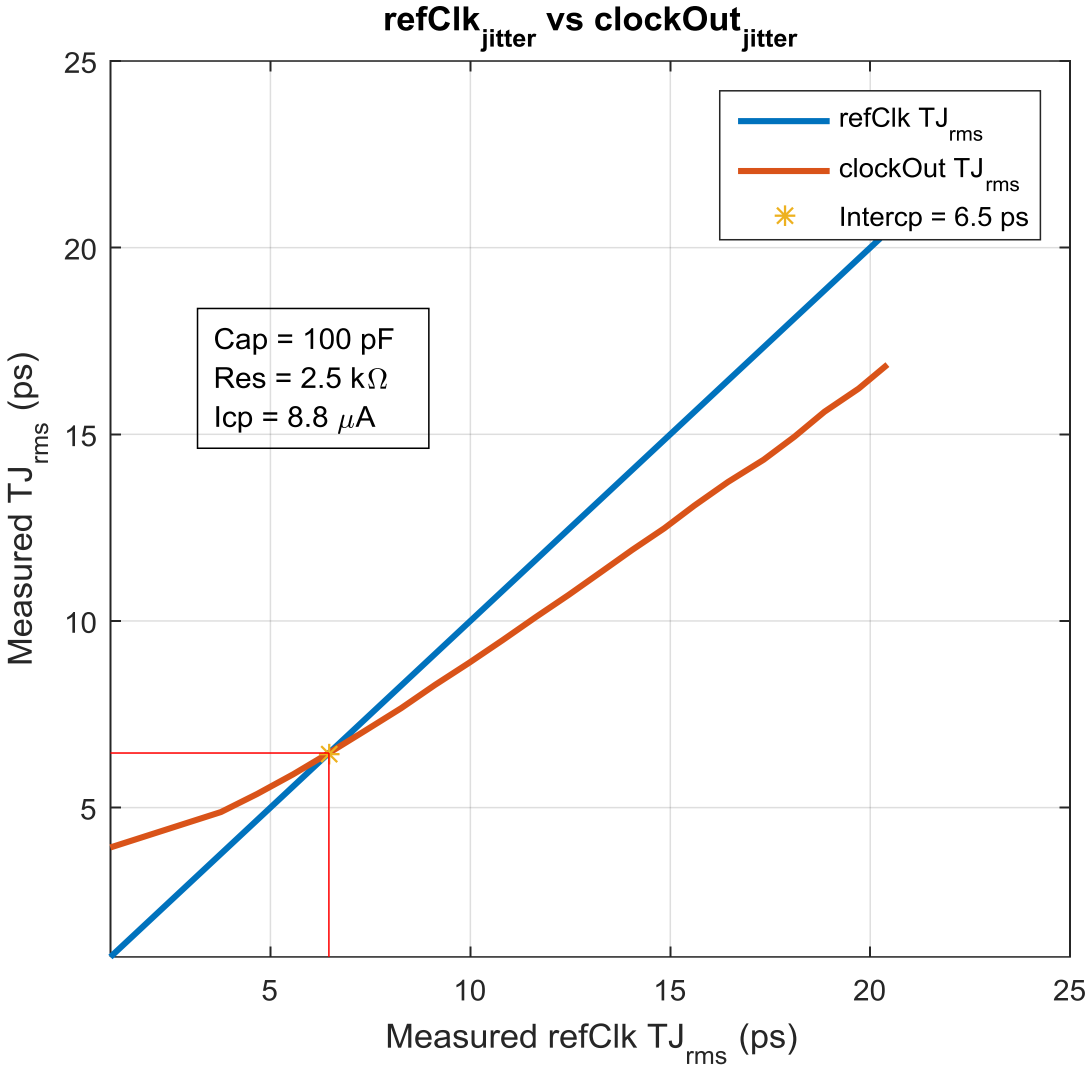
Measured $TJ_{rms}$, N = 1, Cap = 100 pF
…efClk$_{TJrms}$ = 10.0 ps

INPUT JITTER ↑
BW ↓
Damp. Factor ↑

Measured $TJ_{rms}$, N = 1, Cap = 100 pF
Sweeping Icp and R; $refClk_{TJrms}$ = 7.4 ps

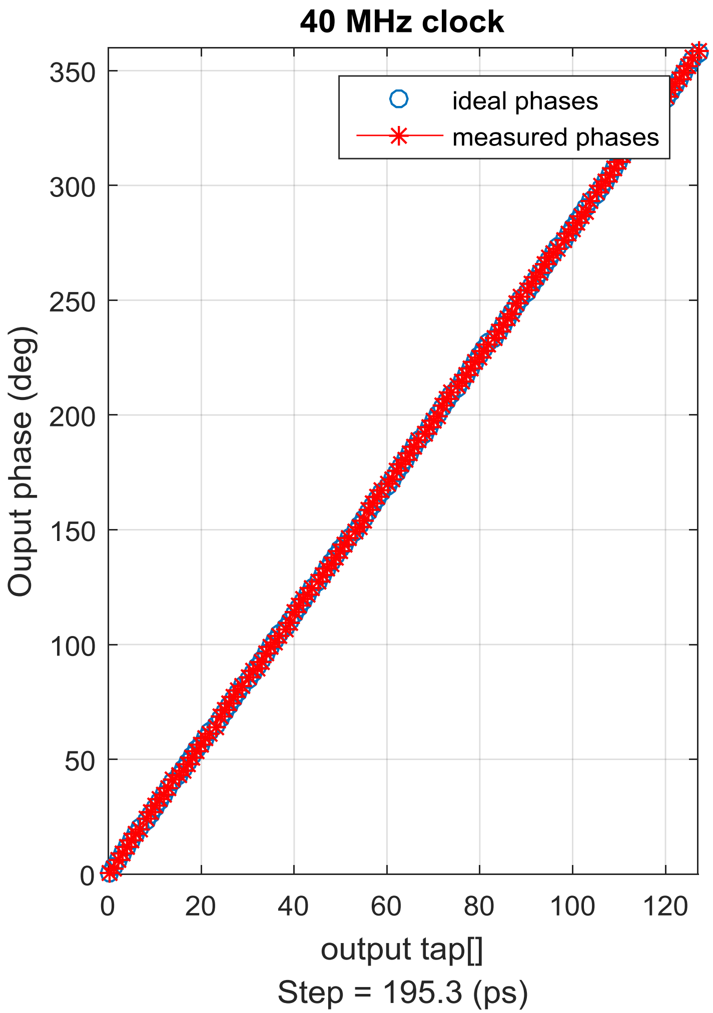
DUT: refClk=320MHz, VCO@320MHz, clkOut=320MHz

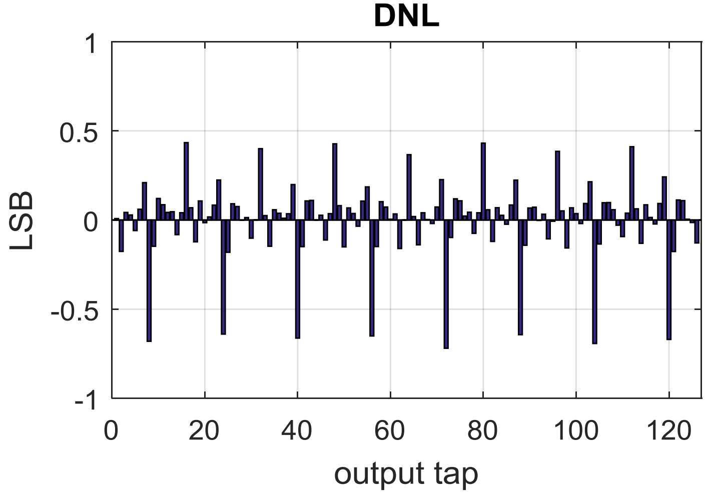
# eCDRPLL PLL Jitter Characterization

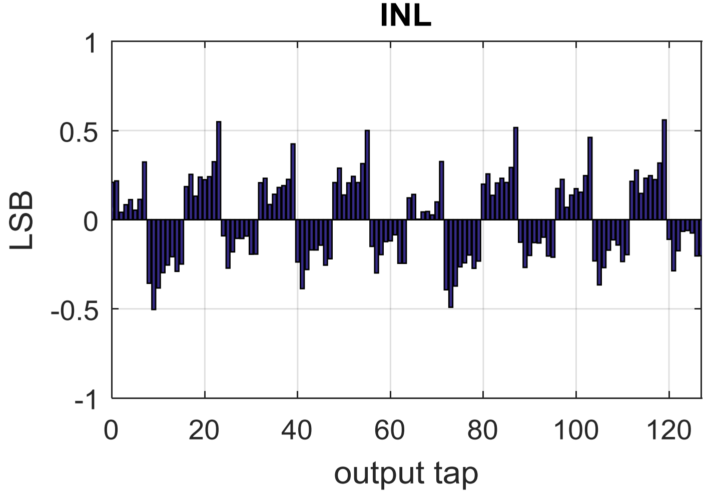


DUT: refClk=320MHz, VCO@320MHz, clkOut=320MHz

# eCDRPLL PLL Phase Shifter







Phase shifter has
128 taps

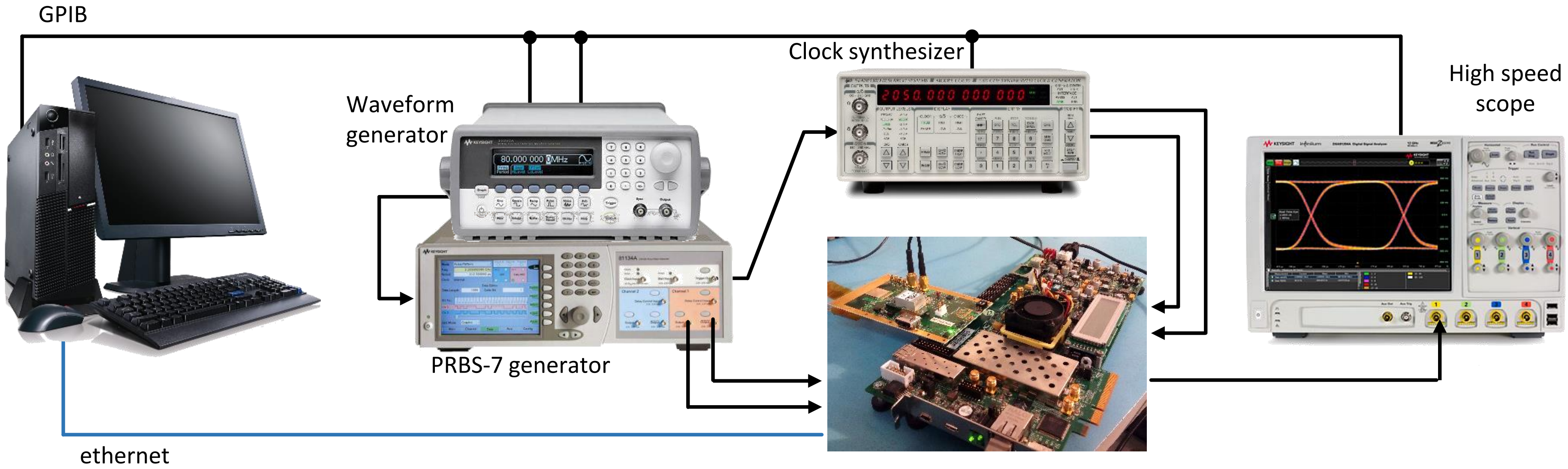
DUT: refClk=40MHz, VCO@320MHz, clkOut=40MHz

# eCDRPLL CDR Performance

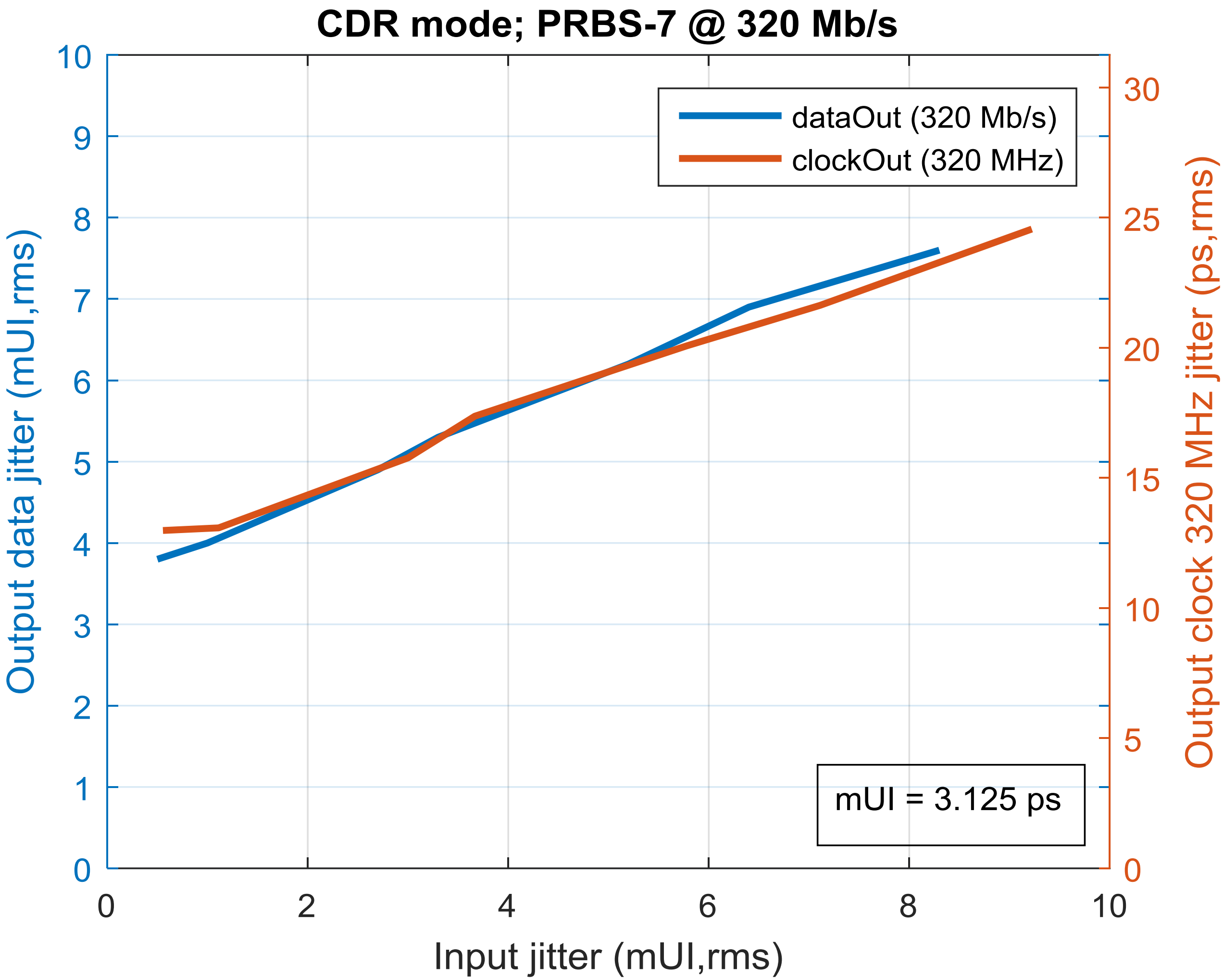
## Test setup – CDR jitter performance

- The measurement setup was done using
  - Data source: Agilent 81133A, 3.35GHz Pulse Pattern Generator (sub 1mUI rms jitter output)
    - Generating PRBS-7 data input
    - An arbitrary waveform generator (Agilent 33250A) was used to inject white noise with different amplitudes
  - Reference Clock: CG635, 2.05 GHz Synthesized clock generator
    - Synchronized timebase with Agilent 81133A to ensure SPIDR is synchronized with the Velopix
  - High speed scope with recClk application: Agilent DSA91204A Digital Analyzer
    - EZJIT Complete app, 2nd order PLL, default bandwidth

  - No parameters sweep was performed



# eCDRPLL CDR @ 320 Mb/s



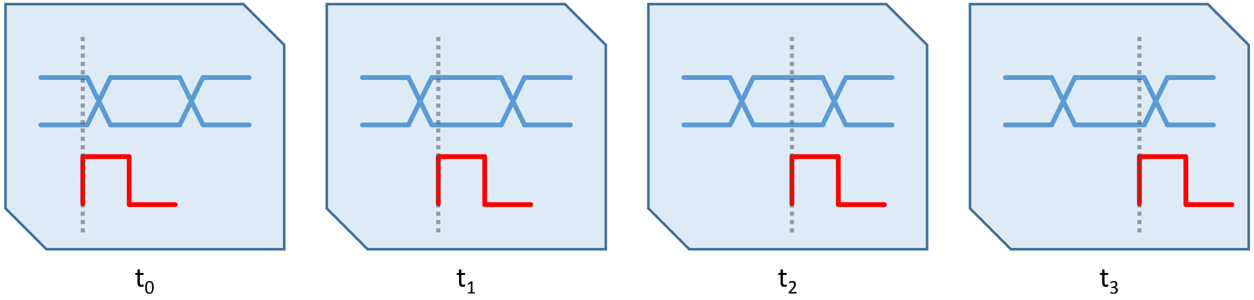
Cap = 500 pF
Res = 0.5 kΩ
$I_{PD}$ = 0.8 µA
$I_{FD}$ = 12.8 µA
1mUI = 3.125ps

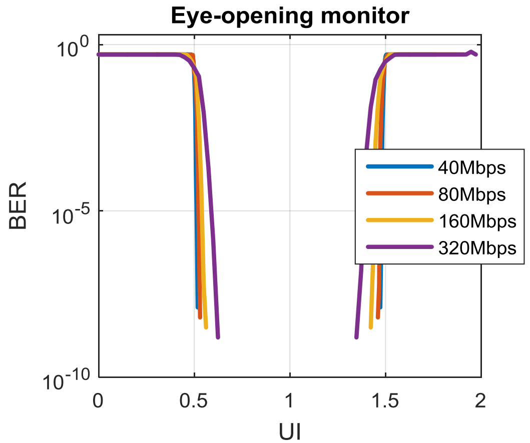
DUT: dataIn=320Mb/s PRBS-7, VCO@320MHz, clkOut=320MHz

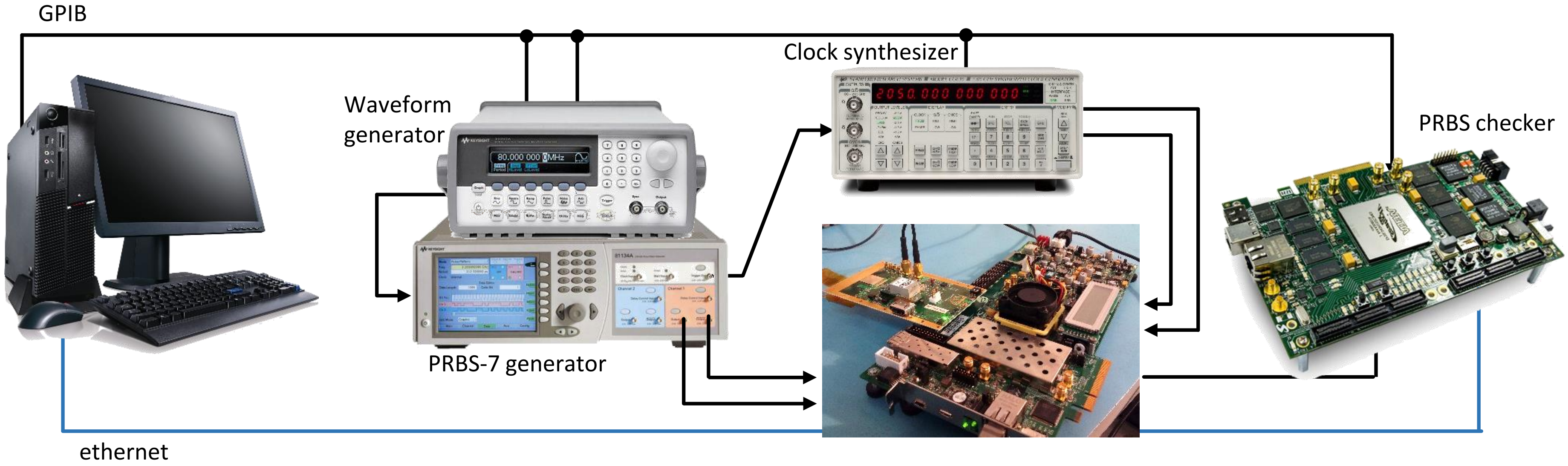
# eCDRPLL (CDR results)

## Test setup – CDR BER data transmision test

- A PRBS-7 checker was implemented in a FPGA to check correct data transmission
- Only one test output pin available on the Velopix



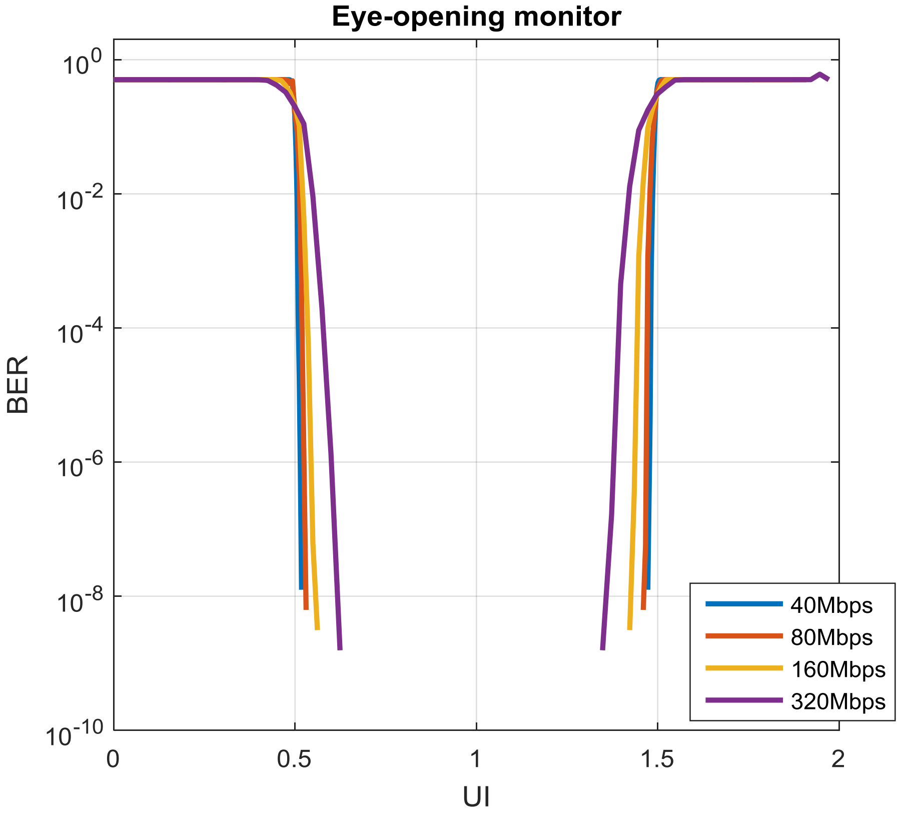




# eCDRPLL (CDR results)

## Measurement – CDR BER data transmision test (eye-opening scan)

- A PRBS-7 checker was implemented in a FPGA to check correct data transmission
- The FPGA's sampling clock was swept with 78 ps resolution
- A 2 seconds durantion BER was performed per sampling point



$$BER = \frac{\sum errors}{\sum transData}$$

# Summary

## A Radiation-hard and phase deterministic CDR & PLL was implemented and tested

- 130nm CMOS technology, (up to M6 metal stack)

## Two Frequency Multiplier modes

- VCO@240MHz - 40, 60, 120 and 240 MHz simultaneous output clocks
  - Full 360˚ phase shift with 260 ps resolution
- VCO@320 MHz – 40, 80, 160 and 320 MHz simultaneous output clocks
  - Full 360˚ phase shift with 260 ps resolution
- Built-in-self-test for mismatch characterization (to be run at production time)

## Clock and Data Recovery

- VCO@320 MHz
  - 40, 80, 160 and 320 MHz simultaneous output clocks
  - 40, 80, 160 and 320 Mbps selectable output recovered data
  - No external clock required
  - Full 360˚ phase shift with 195 ps resolution
  - data pushed-out on clock's falling edge

## Silicon proven

- All working modes have passed
- Sub 10 ps rms jitter
- Macrocell available to the community
- Verilog model and timing files available
- Compatible with a digital on top flow

## Contributions (alphabetically)

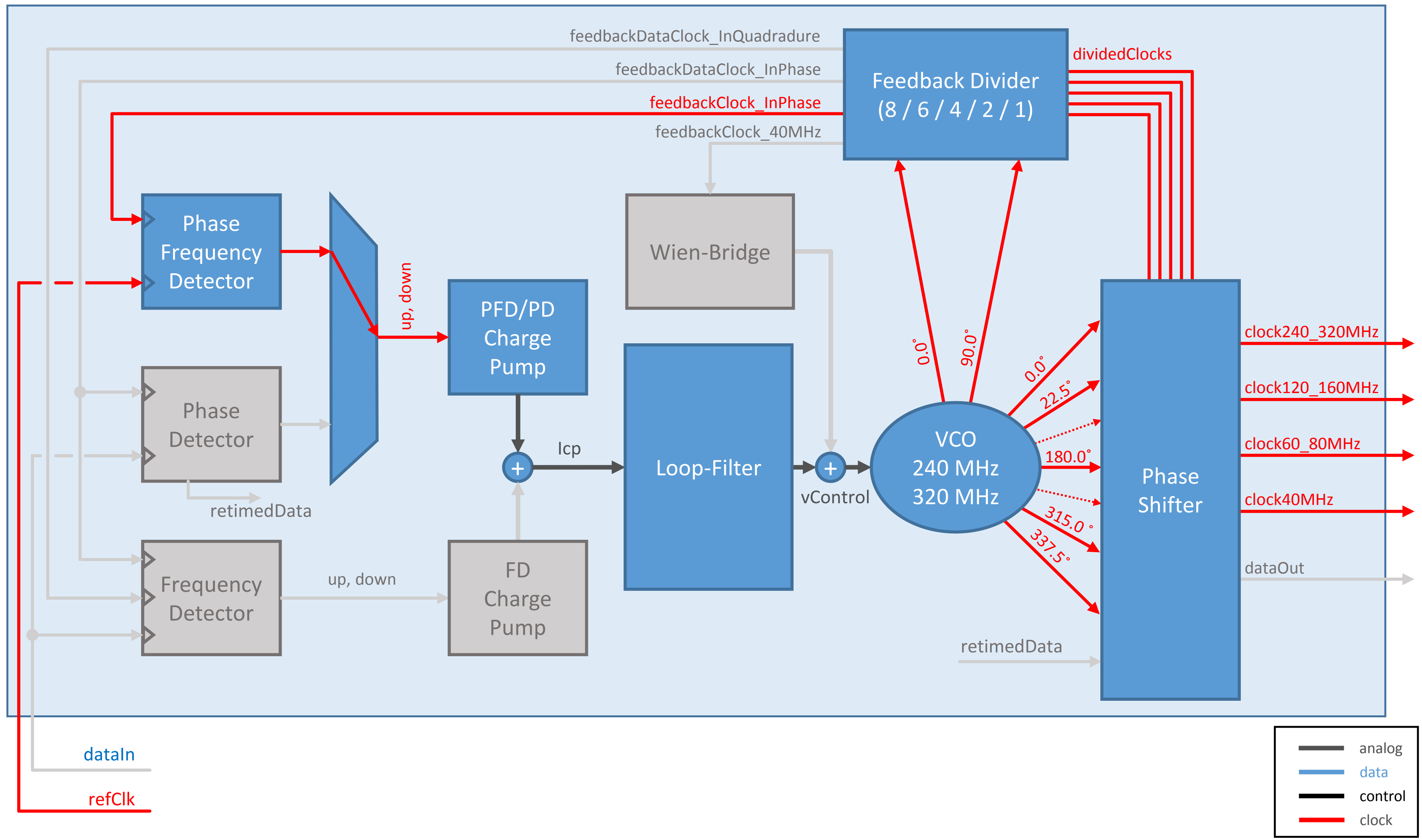
- eCDRPLL Macrocell
  - Paulo Moreira
  - Pedro Leitao
  - Rui Francisco
  - Xavi Llopart
- Velopix team

# Backups

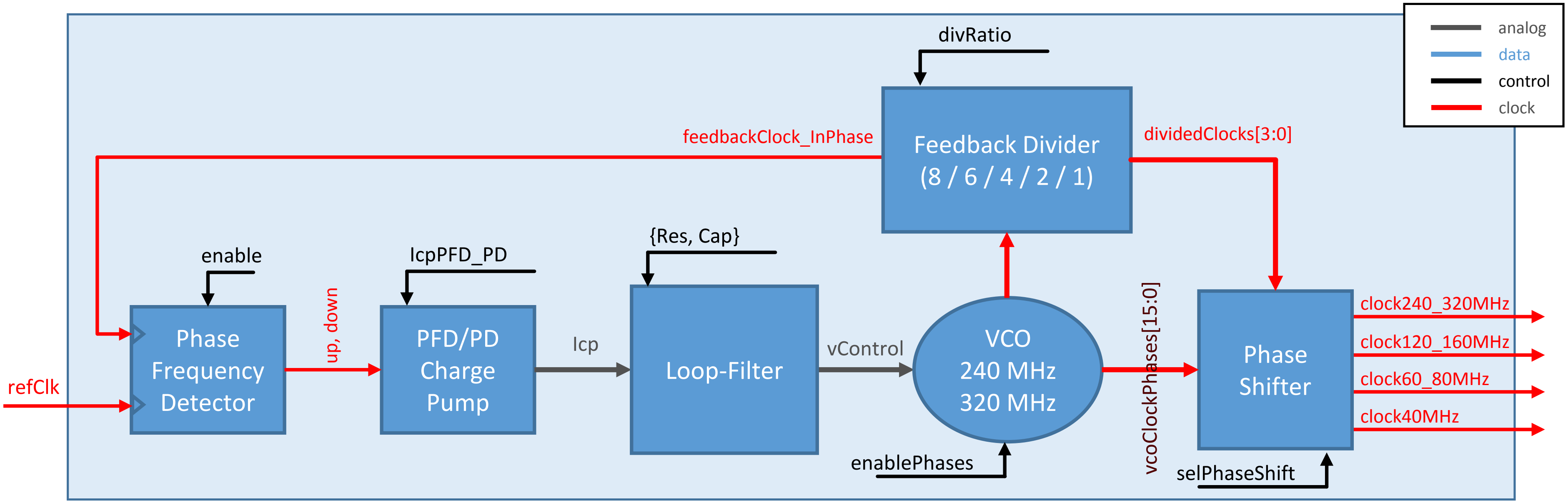
# ePLL vs eCDR vs eCDRPLL

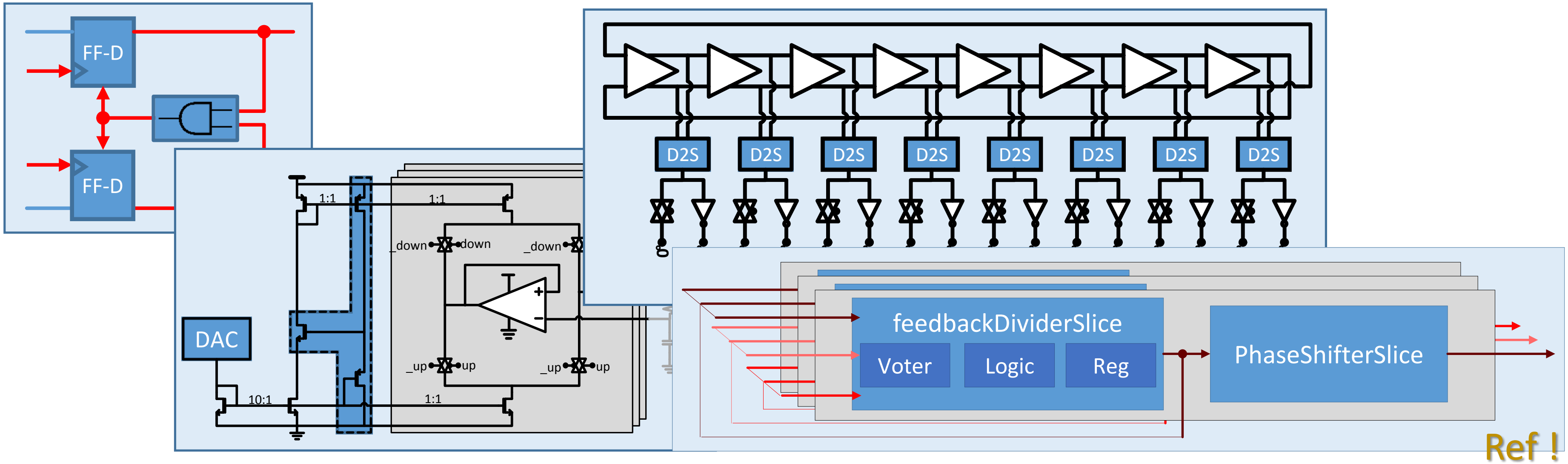
| | ePLL Macrocell (2012) | eCDR Macrocell (2013) | eCDRPLL Macrocell |
|---|---|---|---|
| Tech. | 130nm CMOS | 130nm CMOS | 130nm CMOS |
| Modes of operation | PLL | PLL (initial locking mechanism)<br>CDR | PLL240, PLL320<br>CDR320 |
| Startup Locking Mechanism | Reference (PLL) | Reference (acts as a PLL)<br>Reference-less (uses Wien-Bridge) | Reference (PLL240, PLL320, CDR320)<br>Reference-less (CDR320) |
| Supply Voltage | 1.5 V | 1.5 V | 1.2 V |
| Nominal VCO frequency | 320 MHz | 320 MHz | 320 MHz (PLL320, CDR320)<br>240 MHz (PLL240) |
| Phase Shifting resolution | $f_{VCO}/16$ | $f_{VCO}/16$ | $f_{VCO}/16$ |
| Input Frequencies | 40, 80, 160 MHz | 40, 80, 160, 320 MHz | 40, 80, 160, 320 MHz (PLL320)<br>40, 60, 120, 240 MHz (PLL240) |
| Input Data Rates | - | 40, 80, 160, 320 Mb/s | 40, 80, 160, 320 Mb/s (CDR320) |
| Outputs | 160 and 320 MHz | 40, 80, 160, 320 MHz and data | 40, 80, 160 and 320 MHz (PLL320)<br>40, 60, 120 and 240 MHz (PLL240)<br>40, 80, 160, 320 MHz and data (CDR320) |
| Filter Programmability | Yes, no state machine | Yes, no state machine | Yes, with power-up state machine |
| Power consumption (max) | 41.5 mW | 34.5 mW | < 15 mW (PLL)<br>< 28 mW (CDR, budget) |
| Size | 324 x 387 µm | 425 x 930 µm | 420 x 960 µm |

# eCDRPLL (PLL circuit operation)



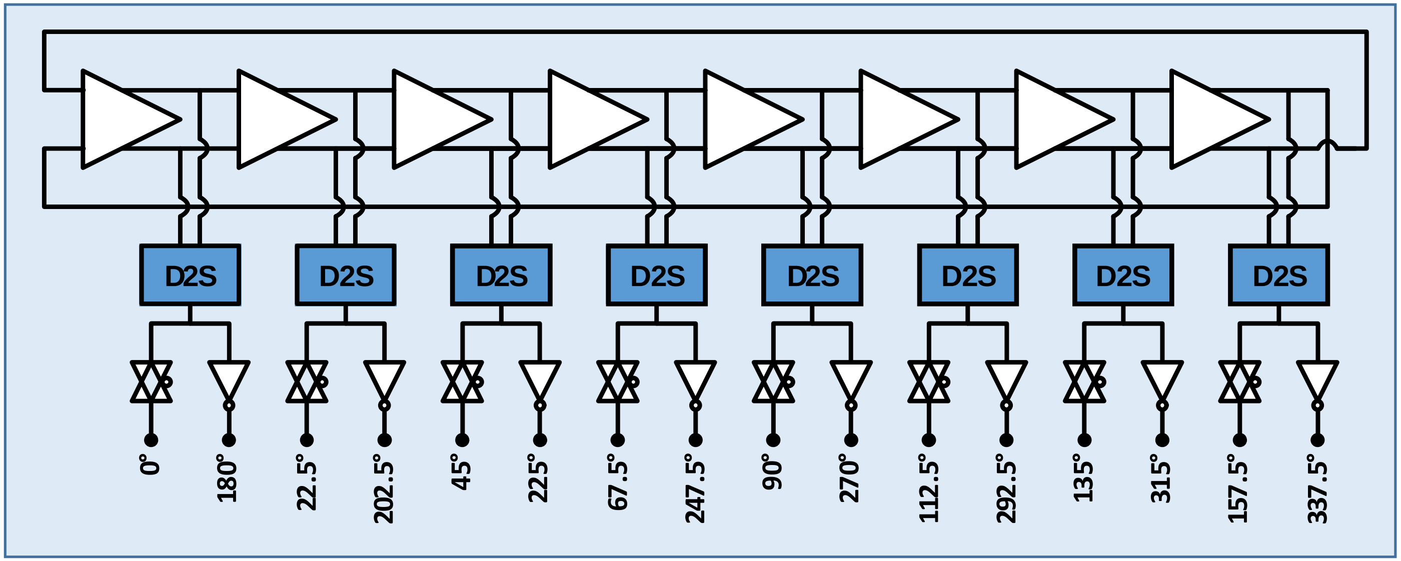
# eCDRPLL (PLL circuit topology)



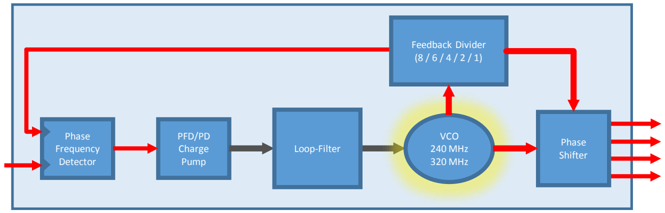


Ref !

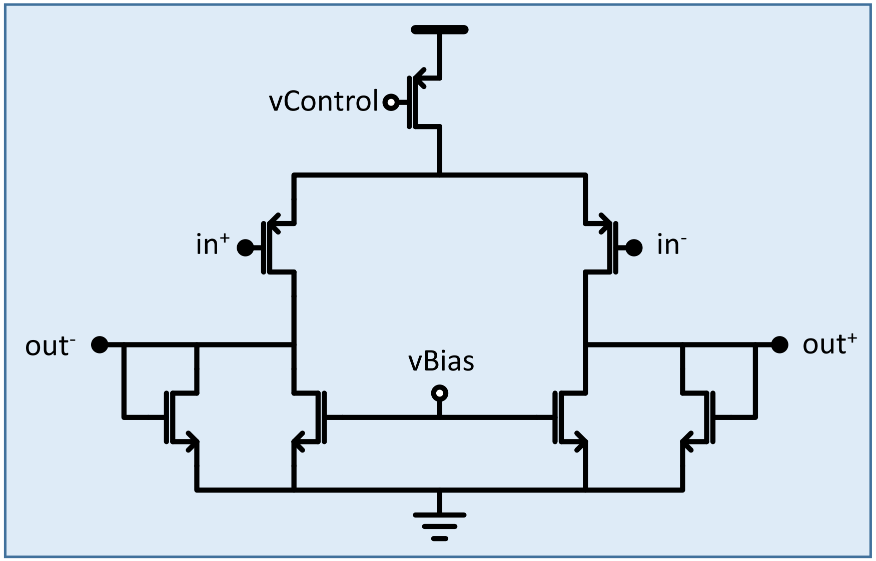
# eCDRPLL (VCO)



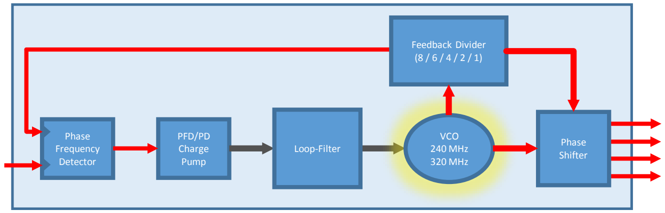
- 8-stage differential ring-oscillator
- 8x D2S converters to generate full-swing output phases ($f_{VCO}/8$)
- Inverters are used to generate the additional 8 phases in order to save power
- Additional 8x D2S converters could have been used if the VCO phase equalization was critical
- Sub 5-ps pk-pk jitter VCO



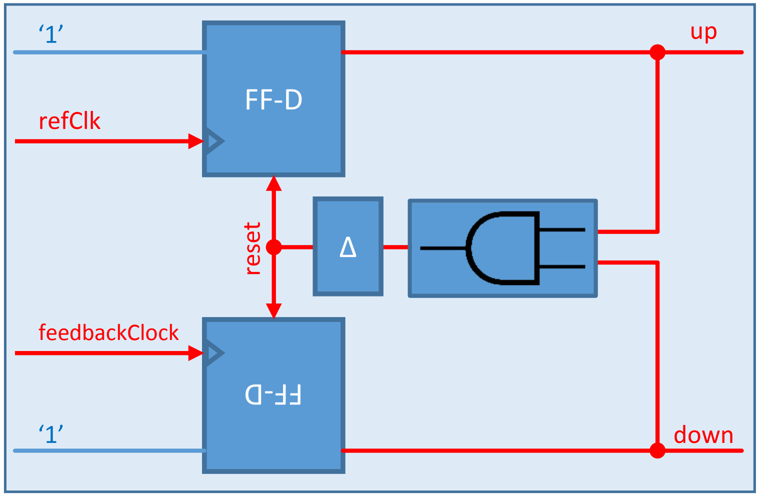
# eCDRPLL (VCO – delay cell)

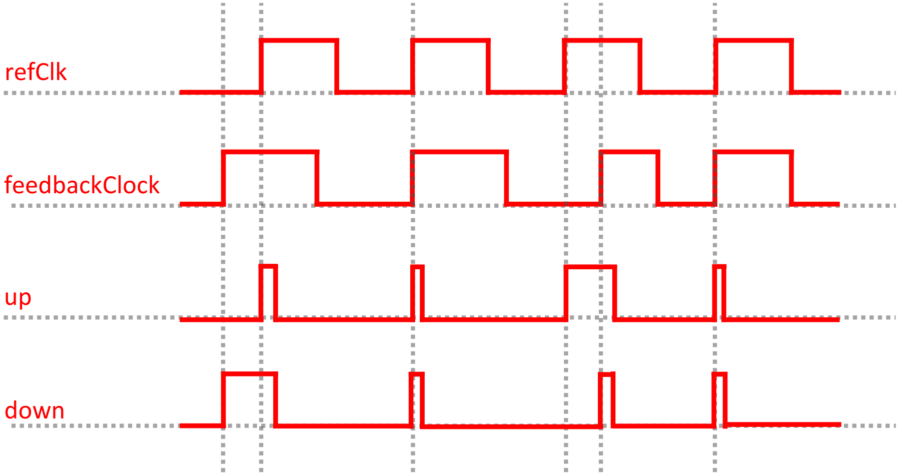


- pMOS differential pair with pMOS current source → control voltage (vControl) referred to the supply voltage
  - vControl higher → VCO slower
- Active load consisting of a current source and a diode to avoid the need of a common-mode feedback circuit and still have enough small-signal gain
- Gate voltage (vBias) of the nMOS current sources derived by means of a replica biasing circuit

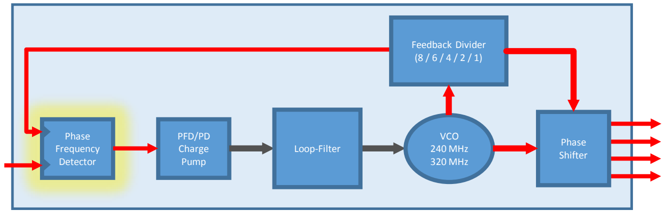


# eCDRPLL (Phase Frequency Detector)

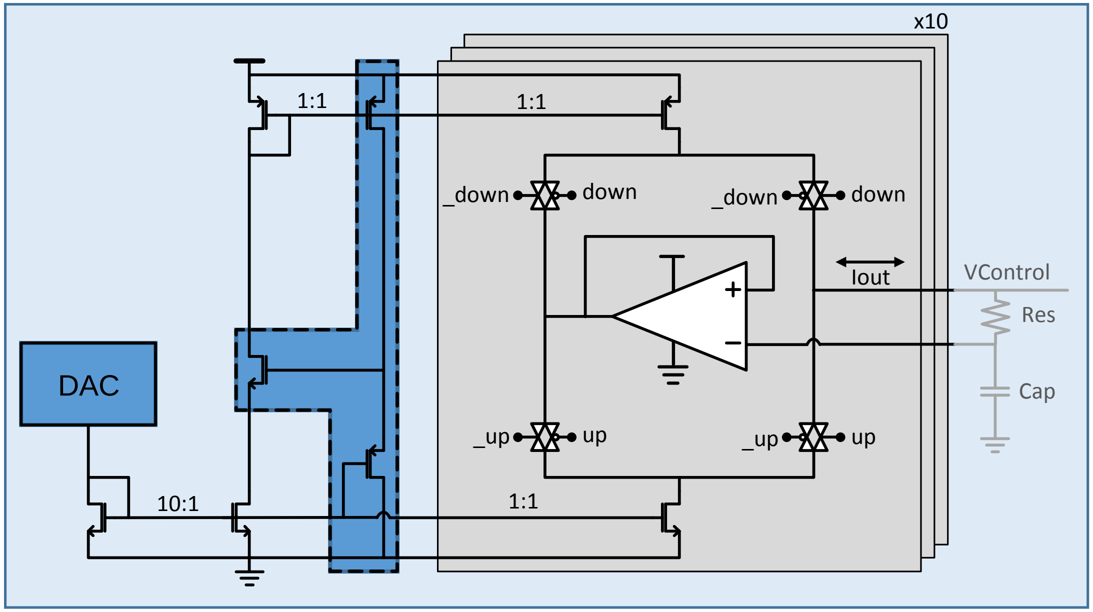


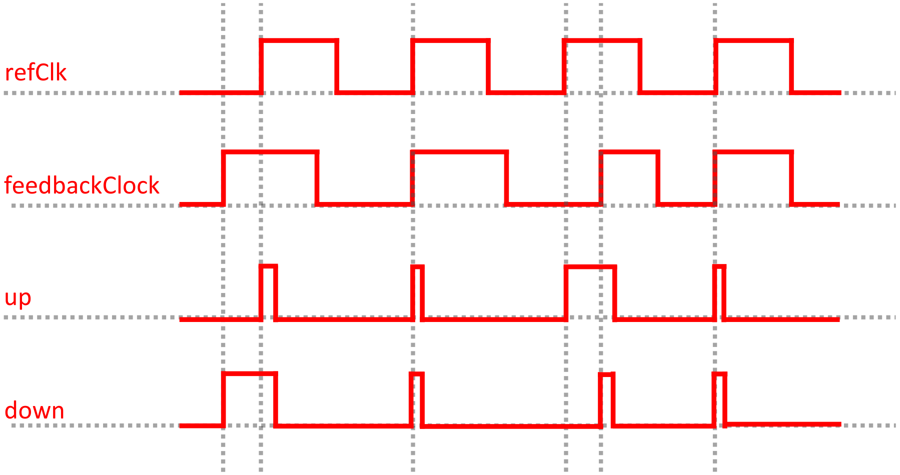


- Linear relationship between the duration of up/down pulses and the phase error
- PFD is sensitive to phase and frequency errors
- Sequencial PFD; delay cell is used to avoid dead-zone at expense of gain linearity
  - {up=0, down=0} → no current is sourced/sink from the loop filter → maintain vControl
  - {up=0, down=1} → VCO is too fast; decrease VCO frequency → increase vControl
  - {up=1, down=0} → VCO is too slow; increase VCO frequency → decrease vControl
  - {up=1, down=1} → no current is sourced/sink from the loop filter → maintain vControl

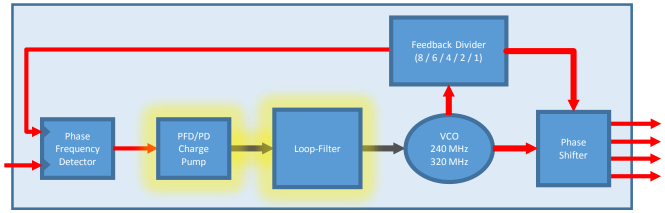


# eCDRPLL (Charge Pump Loop Filter)

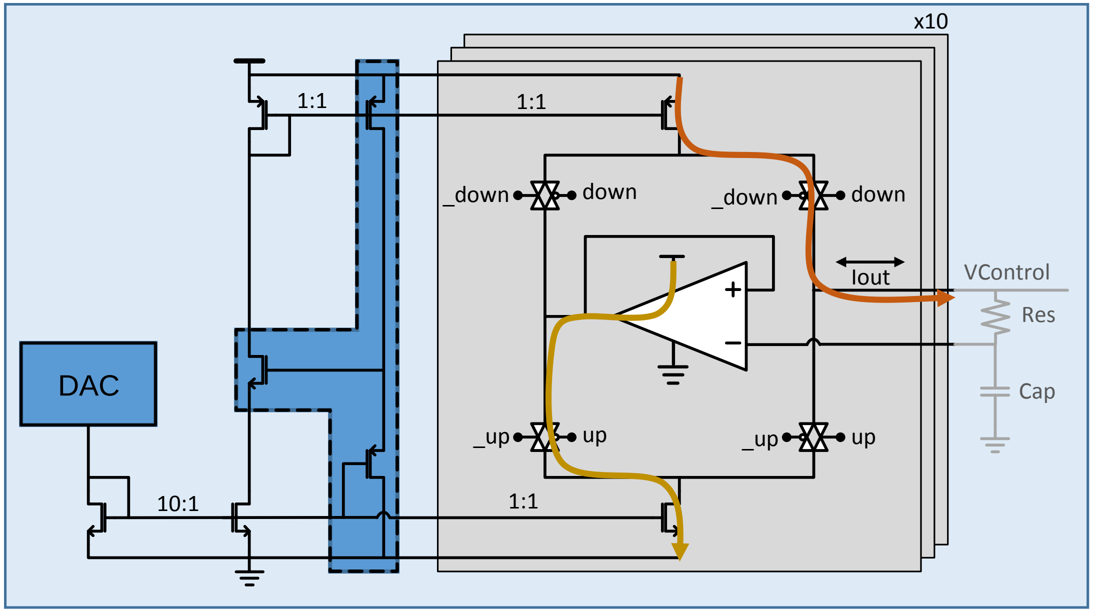




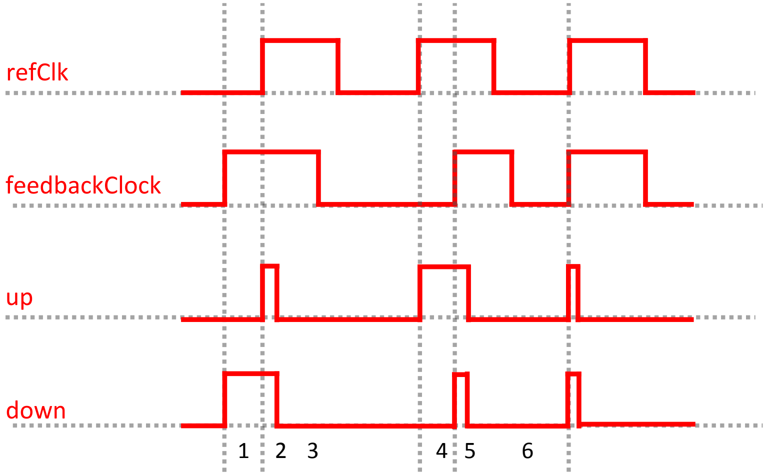
- Non-linearities
  - Fast turn-on/off times
    - Current steering technique will keep bias current on at all times
  - Unequal sink and source currents;
    - The loop filter is charged ou discharged by the current difference when UP and DOWN are both high every clock cycle
    - Solution. Current sources with long length and consequently high output impedance; added 2 source followers to improve the accuracy of the current copy
  - Parasitic capacitance at the drain of the current sources
    - Charge sharing takes place eveytime one of the current sources is connected to the loop filter
    - Solution. Unity-gain amp to equalize the drain voltage of the current sources to the control voltage

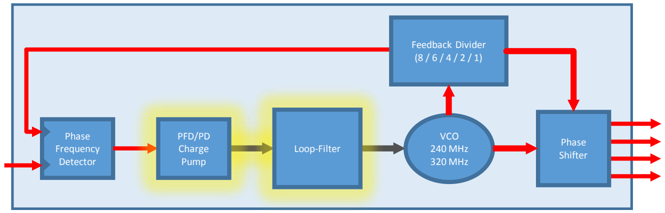


# eCDRPLL (Charge Pump Loop Filter)

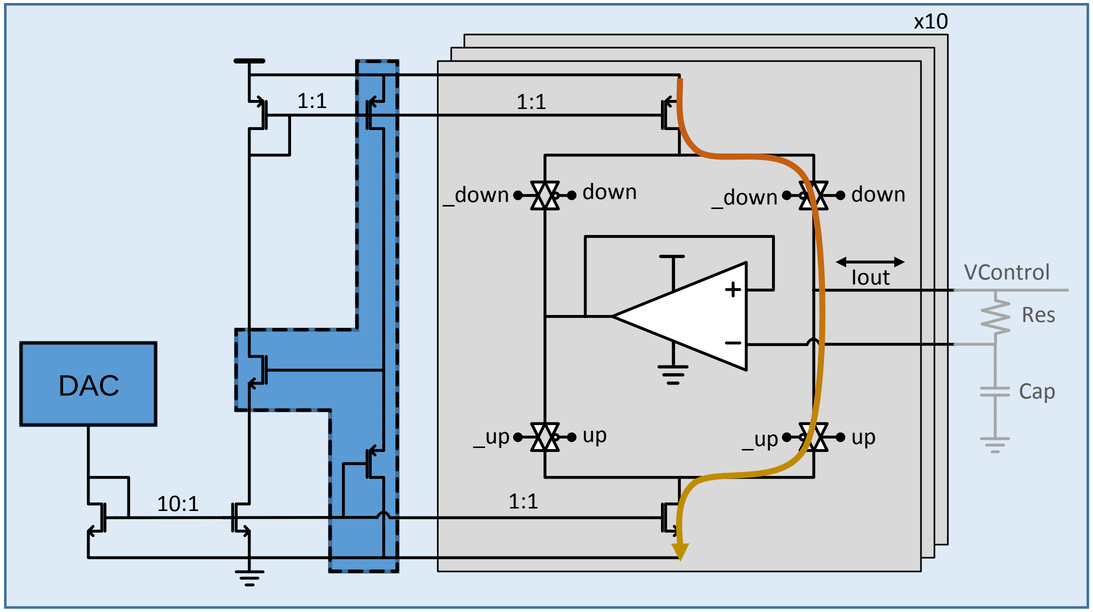


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier

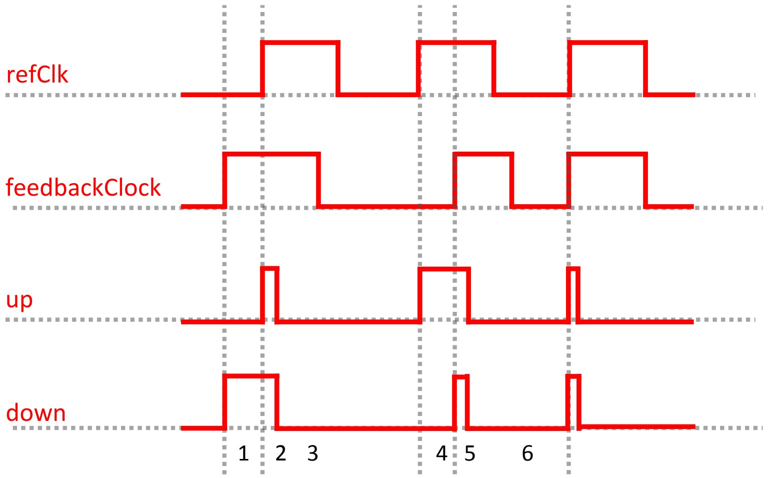


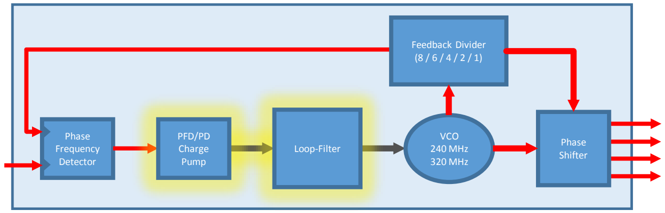


# eCDRPLL (Charge Pump Loop Filter)

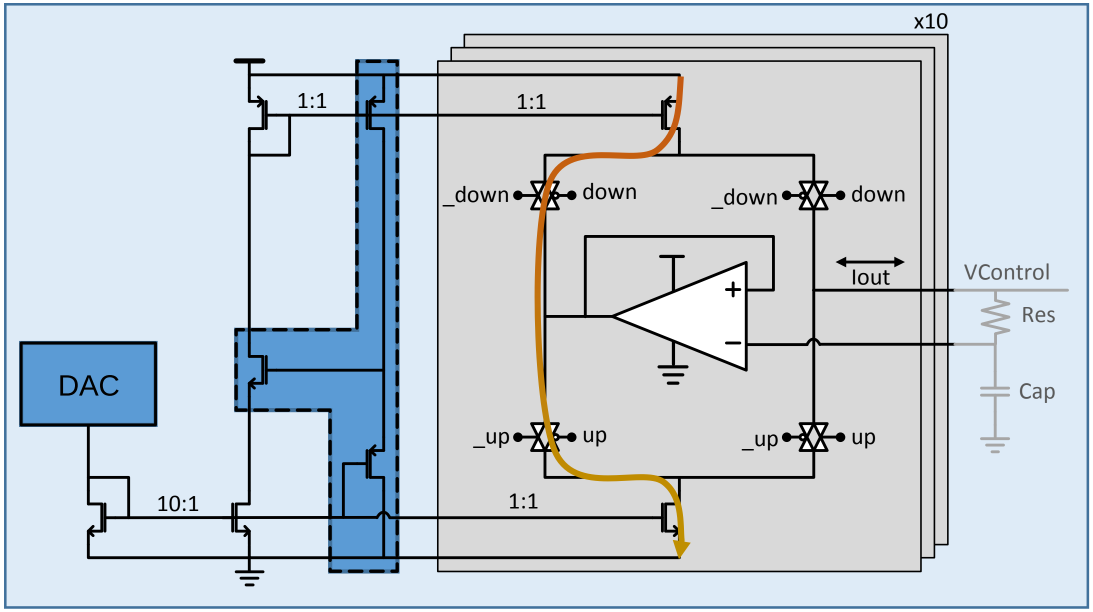


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current

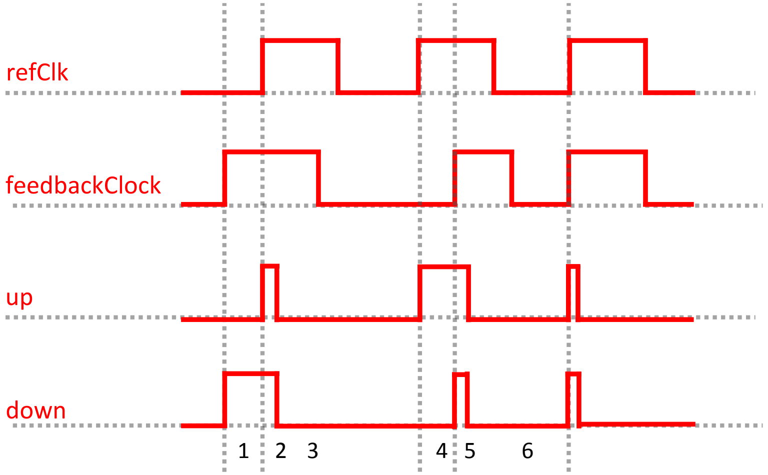


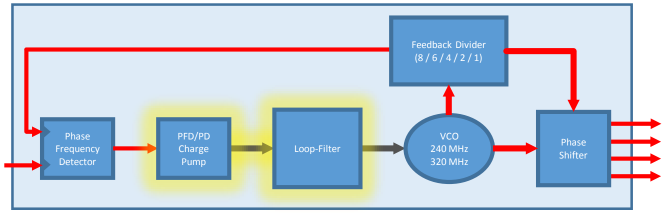


# eCDRPLL (Charge Pump Loop Filter)

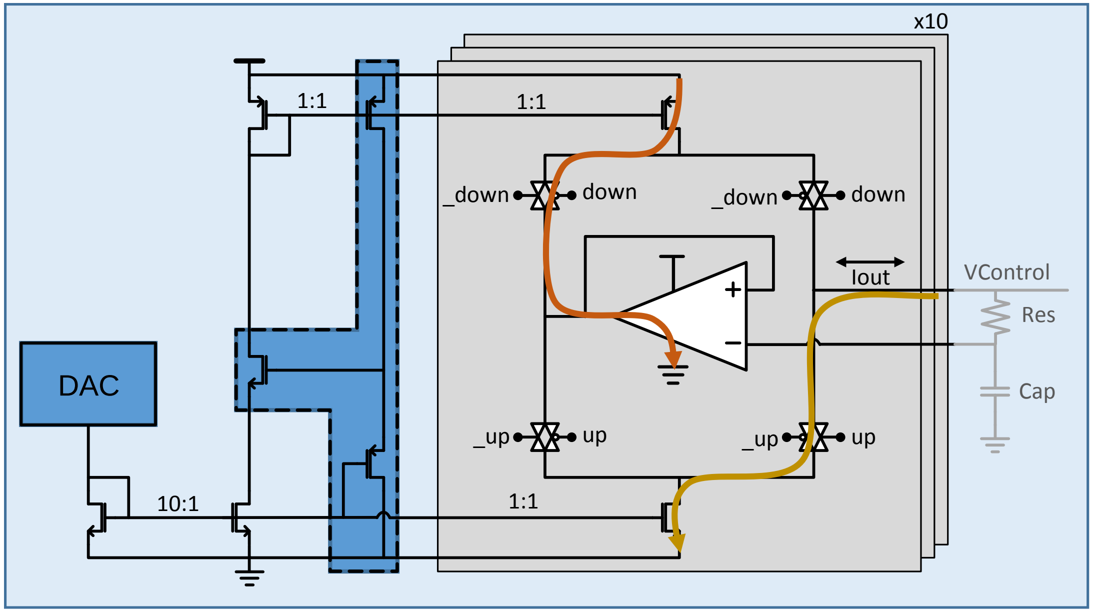


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  3. PFD in reset state
     - up = 0, down = 0
     - No current should be sourced/sink

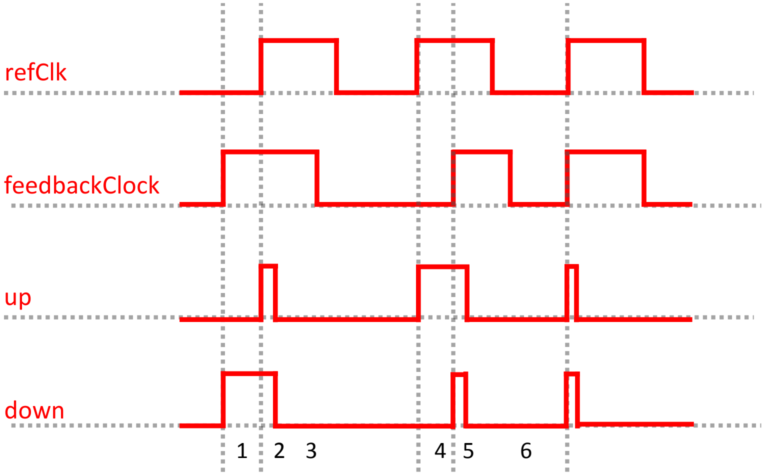


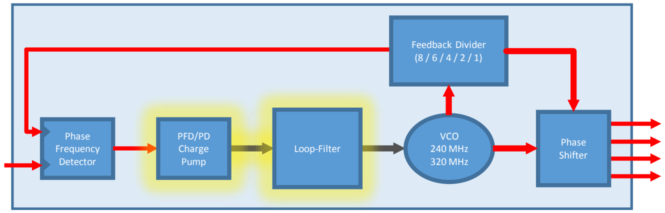


# eCDRPLL (Charge Pump Loop Filter)

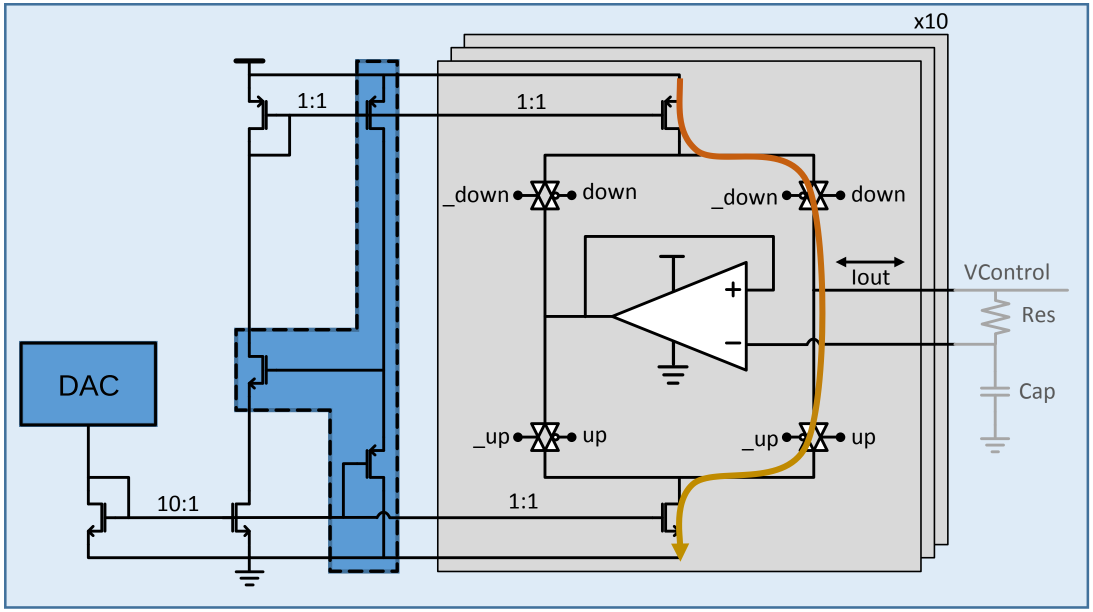


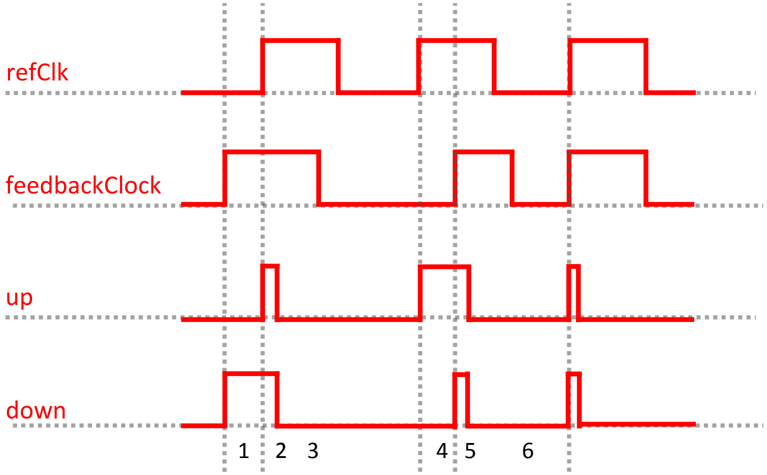
- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  3. PFD in reset state
     - up = 0, down = 0
     - No current should be sourced/sink
  4. VCO slower than refclk – VCO is late
     - up = 1, down = 0
     - Increase frequency
     - Source current flows into the amplifier
     - Sink current flows out of the filter



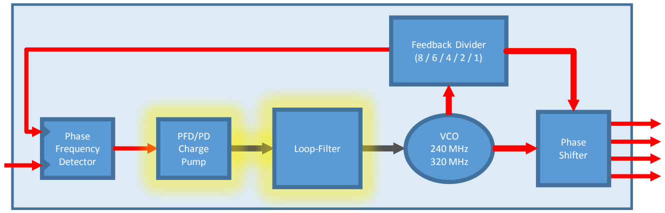


# eCDRPLL (Charge Pump Loop Filter)

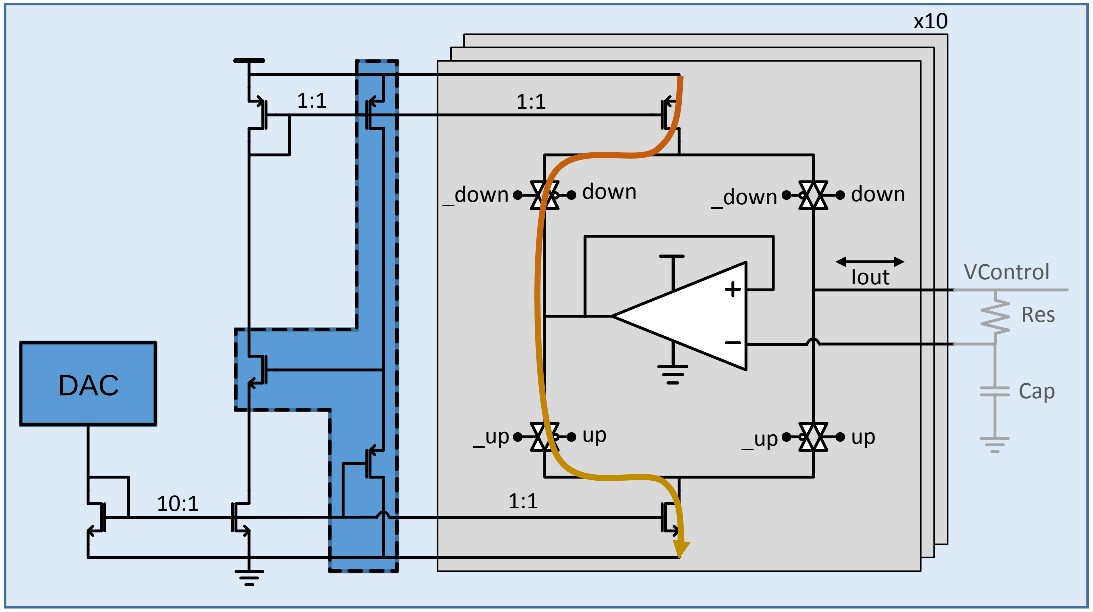


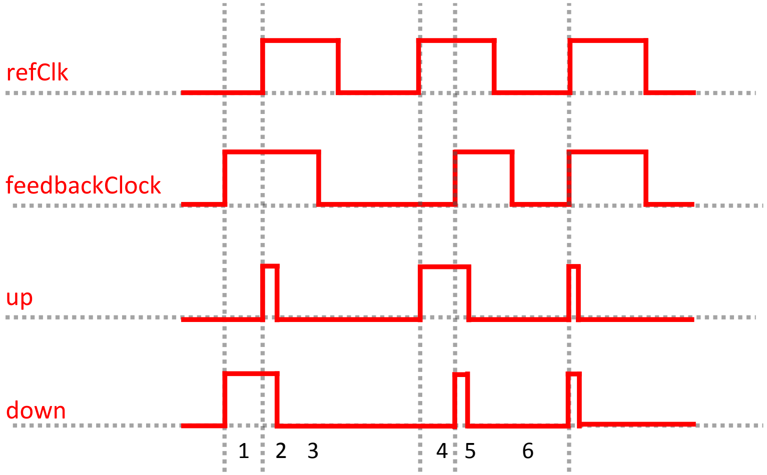


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  3. PFD in reset state
     - up = 0, down = 0
     - No current should be sourced/sink
  4. VCO slower than refclk – VCO is late
     - up = 1, down = 0
     - Increase frequency
     - Source current flows into the amplifier
     - Sink current flows out of the filter
  5. VCO has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current

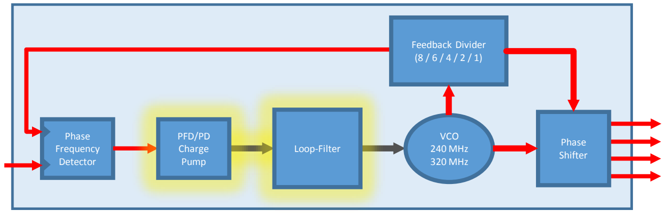


# eCDRPLL (Charge Pump Loop Filter)

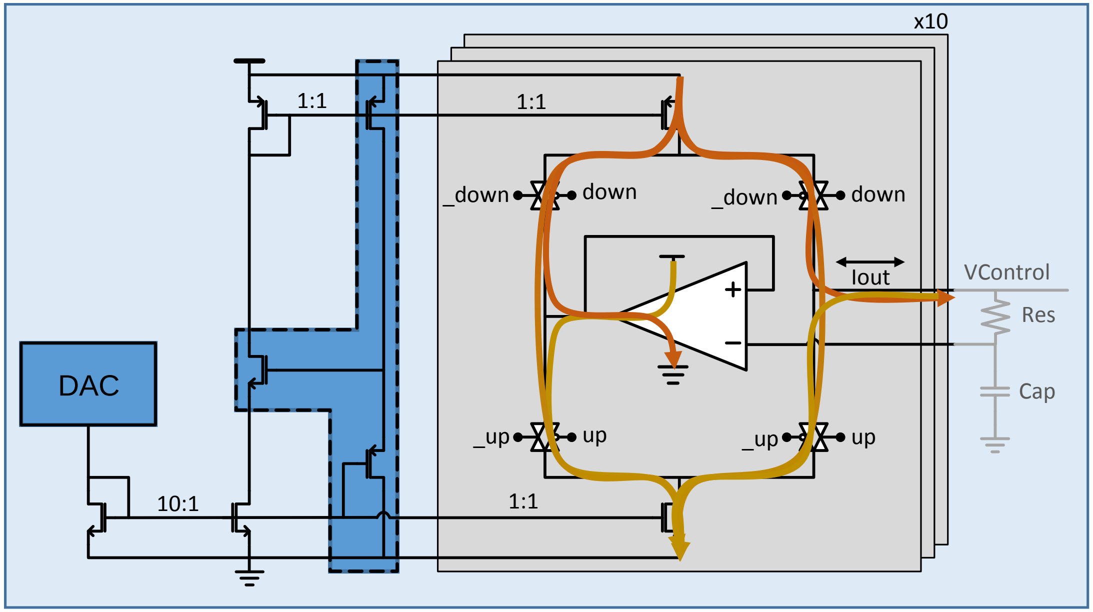


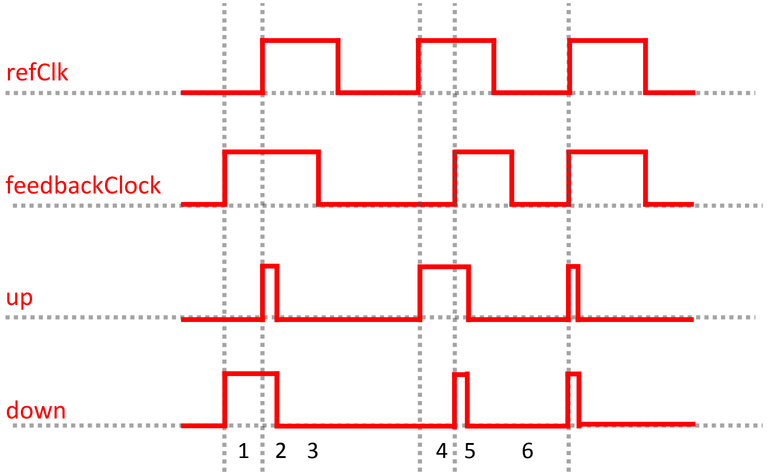


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  3. PFD in reset state
     - up = 0, down = 0
     - No current should be sourced/sink
  4. VCO slower than refclk – VCO is late
     - up = 1, down = 0
     - Increase frequency
     - Source current flows into the amplifier
     - Sink current flows out of the filter
  5. VCO has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  6. PFD in reset state
     - up = 0, down = 0

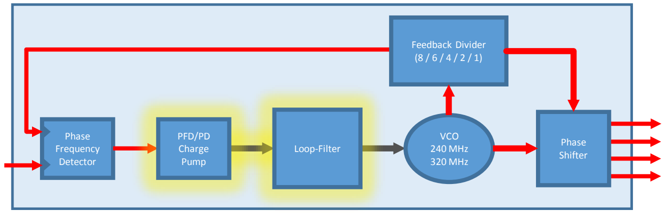


# eCDRPLL (Charge Pump Loop Filter)

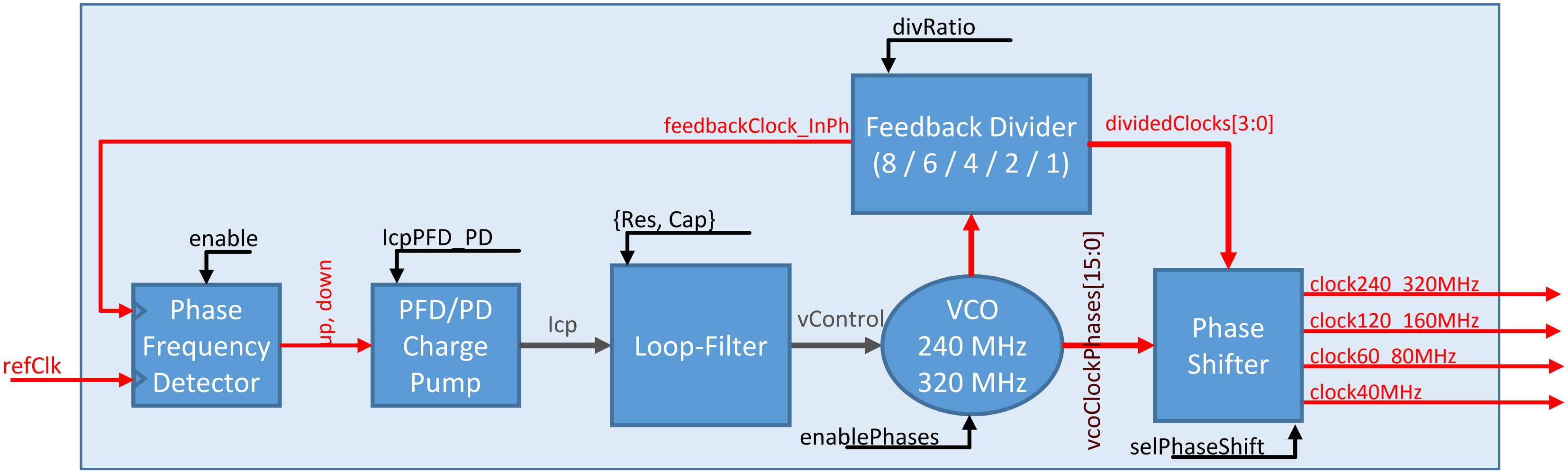


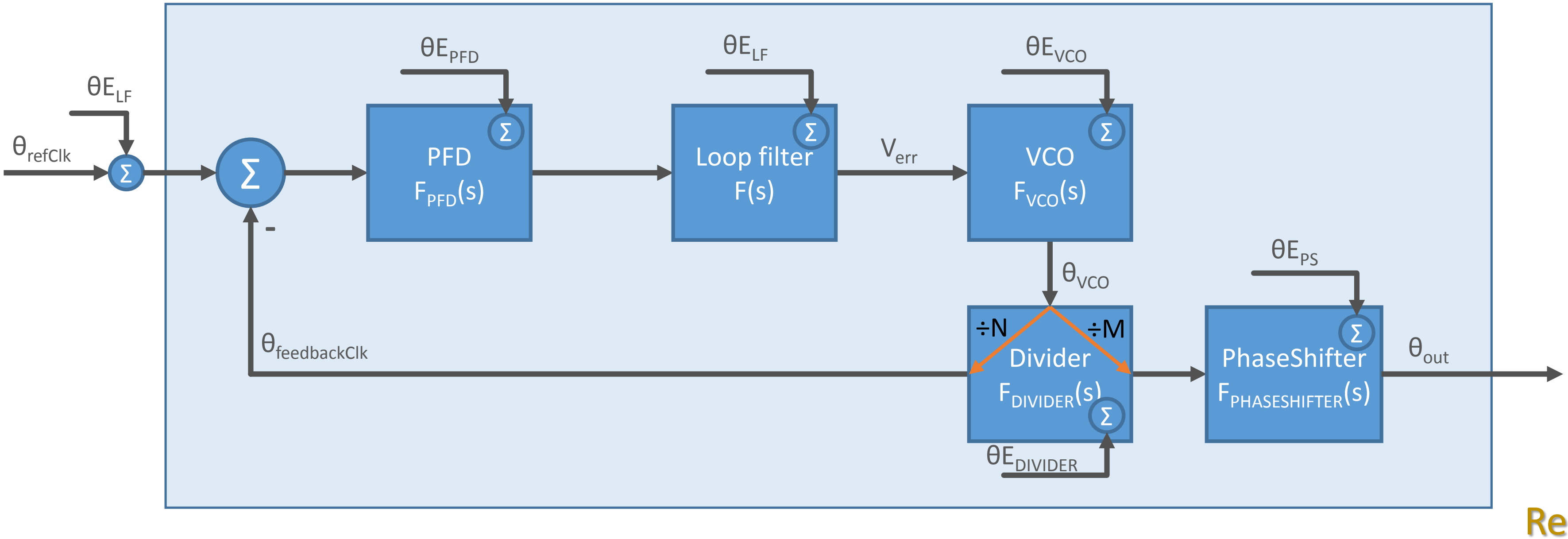


- Operating mode
  1. VCO faster than refClk – VCO is early
     - up = 0, down = 1
     - Decrease frequency
     - Source current flows into the loop filter
     - Sink current flows out of the amplifier
  2. refClk has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  3. PFD in reset state
     - up = 0, down = 0
     - No current should be sourced/sink
  4. VCO slower than refclk – VCO is late
     - up = 1, down = 0
     - Increase frequency
     - Source current flows into the amplifier
     - Sink current flows out of the filter
  5. VCO has arrived – PFD delay
     - up = 1, down = 1
     - No current flows to the loop if sink = source current
  6. PFD in reset state
     - up = 0, down = 0



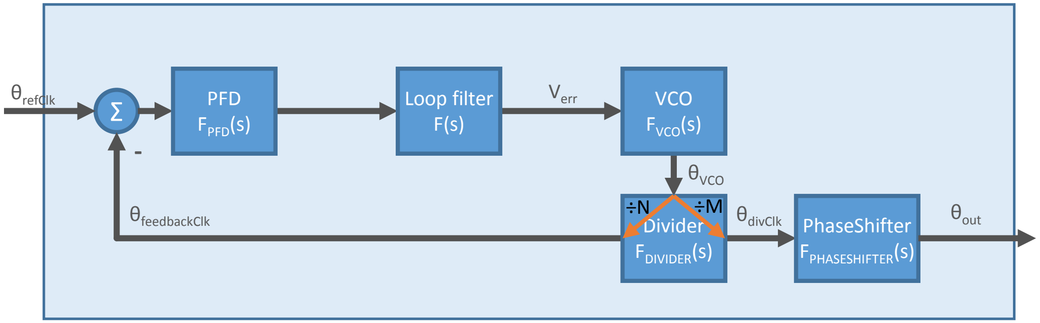
# eCDRPLL (PLL Analysis)





Ref !

# eCDRPLL (PLL Analysis)



**PFD and charge pump**

$$F_{PFD}(s) = K_{PFD} = \frac{I_{cp}}{2\pi}$$

**Loop filter**

$$F(s) = R + \frac{1}{C \cdot s}$$

**VCO**

$$\mathrm{F}_{VCO}(s) = \frac{K_{VCO}}{s}$$

**PhaseShifter**

$$F_{PS}(s) = 1$$

**Divider**

$$\frac{\theta_{feedbackClk}}{\theta_{VCO}} = \frac{1}{\mathrm{N}}$$

$$\frac{\theta_{divClk}}{\theta_{VCO}} = \frac{1}{M}$$

**PLL**

$$\frac{\theta_{out}}{\theta_{refClk}} = \frac{\theta_{VCO}}{\theta_{ref}} \cdot \frac{\theta_{out}}{\theta_{VCO}}$$

$$\frac{\theta_{out}}{\theta_{refClk}} = \frac{F_{PFD}(s)F(s)F_{VCO}(s)}{1 + F_{PFD}(s)F(s)F_{VCO}(s)\frac{\theta_{feedbackClk}}{\theta_{VCO}}} \cdot \frac{\theta_{divClk}}{\theta_{VCO}} \cdot F_{PS}\,(s)$$

Ref !

# eCDRPLL (PLL Analysis)

The eCDRPLL can be approximated to a 2nd order type-II negative feedback loop when in steady-state!

$$\frac{\theta_{out}}{\theta_{refClk}} = \frac{F_{PFD}(s)F(s)F_{VCO}(s)}{1+\frac{F_{PFD}(s)F(s)F_{VCO}(s)}{N}} \cdot \frac{1}{\mathrm{M}}$$

$$\frac{\theta_{out}}{\theta_{refClk}} = H_0 \frac{\omega_n^2(\tau s+1)}{\frac{s^2}{N}+\frac{2\zeta\omega_n}{N}s+\frac{\omega_n^2}{N}}$$

$$\tau = RC$$

$$\omega_n = \sqrt{\frac{I_{cp}K_{vco}}{2\pi CN}}$$

$$H_0 = \frac{N}{M}$$

$$\xi = \frac{\tau\omega_n}{2} = \frac{RC\omega_n}{2}$$

$R$, filter resistance (Ohm)
$C$, filter capacitance (Farad)
$I_{cp}$, charge-pump current (A)

$K_{vco}$, gain of the VCO in steady-state (rad/V)
$N$, division ratio
$M$, clock output ratio

$\omega_n$, natural frequency (rad/s)
$\tau$, settling time
$\xi$, damping factor

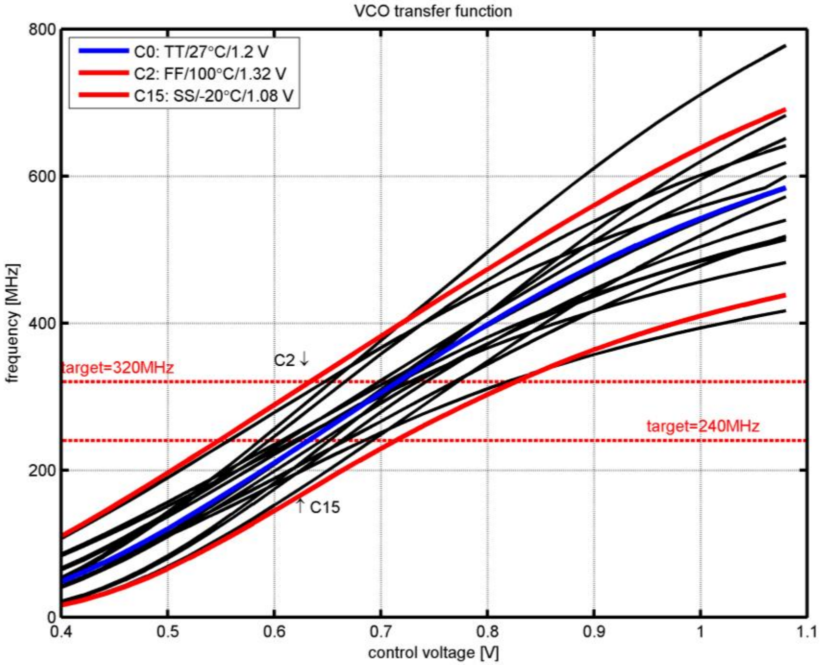
Ref !

# eCDRPLL (parameter exploration)

## Parameter exploration

- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance
- The VCO oscillating at steady state (240 MHz and 320 MHz) will give the $K_{vco}$
  - Extracted VCO transfer function
  - The 3 corners (C0, typical, C2, high gain, and C15, low gain) were used for stability studies

| Mode<br>Corner | @ 240 MHz (MHz/V) | @ 320 MHz (MHz/V) |
|---|---|---|
| C15 (low gain) | 673.4 | 512.4 |
| C0 (typical) | 960.4 | 943.9 |
| C2 (high gain) | 1163.9 | 1212.6 |



# eCDRPLL (parameter exploration)

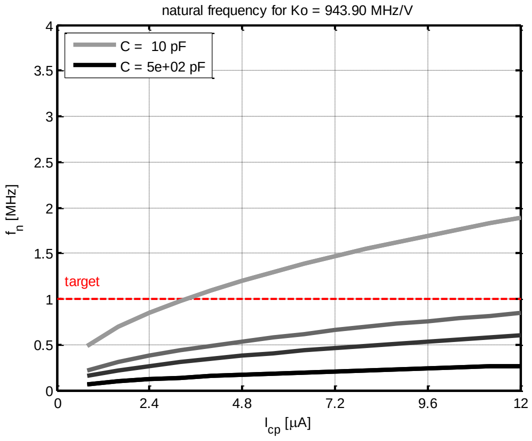
## Parameter exploration

- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance

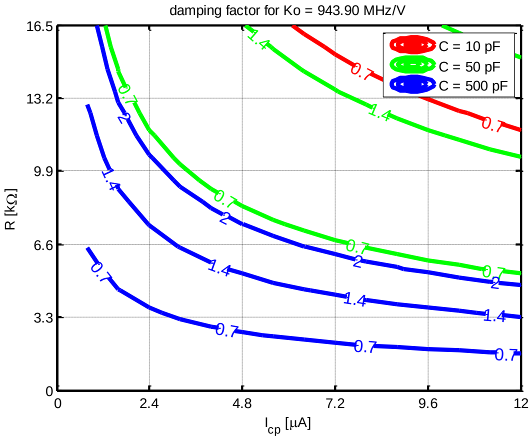
- To select the parameters' range the following contrains were selected as middle point
  - Bandwidth lower than 1 MHz
  - Damping factor between 0.7 and 1.4
    - Damping factor < 0.7 will cause ringing to transient responses

$$\omega_n = \sqrt{\frac{K_{vco} I_{cp}}{2\pi C N}} < 1\ MHz$$

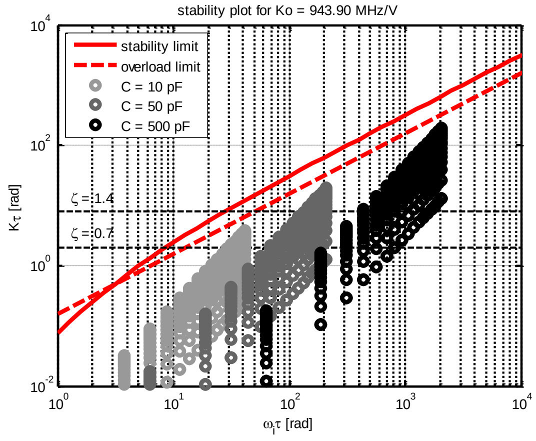
$$\xi = \frac{R C \omega_n}{2} = 1$$



natural frequency for Ko = 943.90 MHz/V



damping factor for Ko = 943.90 MHz/V



stability plot for Ko = 943.90 MHz/V

Example: VCO@320MHz, refClk@40MHz, C0

# eCDRPLL (parameter exploration)

## Parameter exploration

- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance

- The parameters range were selected as follow
  - Icp
    - Current of the phase and frequency detector charge pump can range from 0 to 12 µA with a 0.8 µA step
  - Resistor
    - The filter resistor can range from 0.5 to 15.5 kΩ with a 1 kΩ step
  - Capacitor
    - The filter capacitance can range from 10, 50, 100 to 500 pF

- The eCDRPLL's PLL performance can be optimized depending on the input reference clock
  - If the reference clock is clean, the PLL should follow the reference clock
    - BW ↑, damping factor↓
  - If the reference clock is dirty, the PLL should only follow the reference clock average
    - BW ↓, damping factor ↑

  - Higher damping factors (critically or overdamped) might improve SEU robustness

# eCDRPLL (parameter exploration)

## Parameter exploration

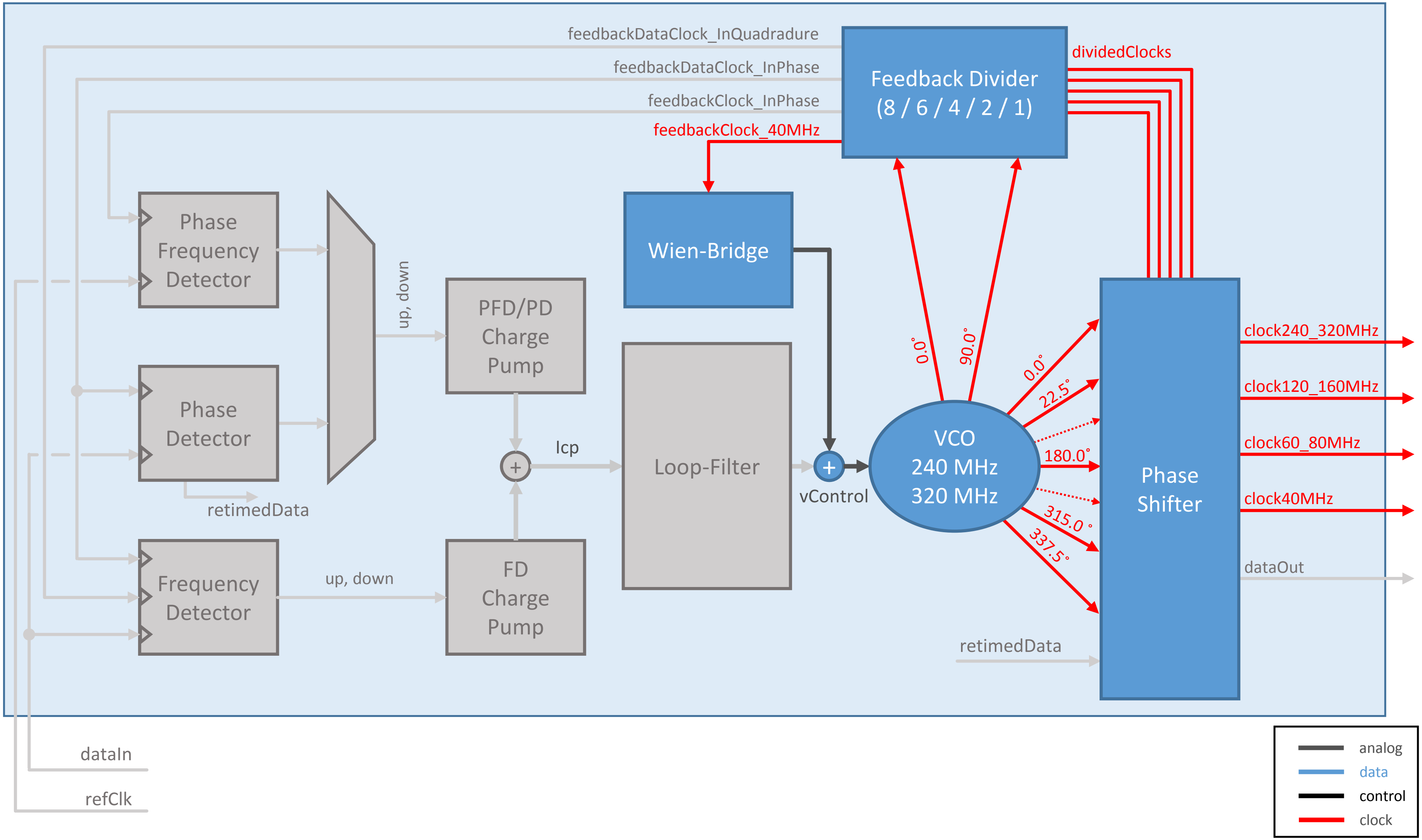
- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance

- A simplistic verilog model of the PLL was used to estimate the PLL's jitter performance
  - A reference clock with a white-noise jitter (2.5ps, 5ps and 10ps rms) was used
  - PFD's delay to avoid dead-time of 400 ps
  - Loop mismatches:
    - Charge pump with a leakage current of 1% of LSB (8 nA) and 1% mismatch between sink and source current
    - A filter type (R-C1)||C2 was used in the model where C2 simulates the parasitic capacitances of the charge-pump switches and the VCO
  - VCO with 1-ps rms jitter (normal distribution) and extracted transfer function
  - Feedback divider and phase shifter with back annotation pnr delay values

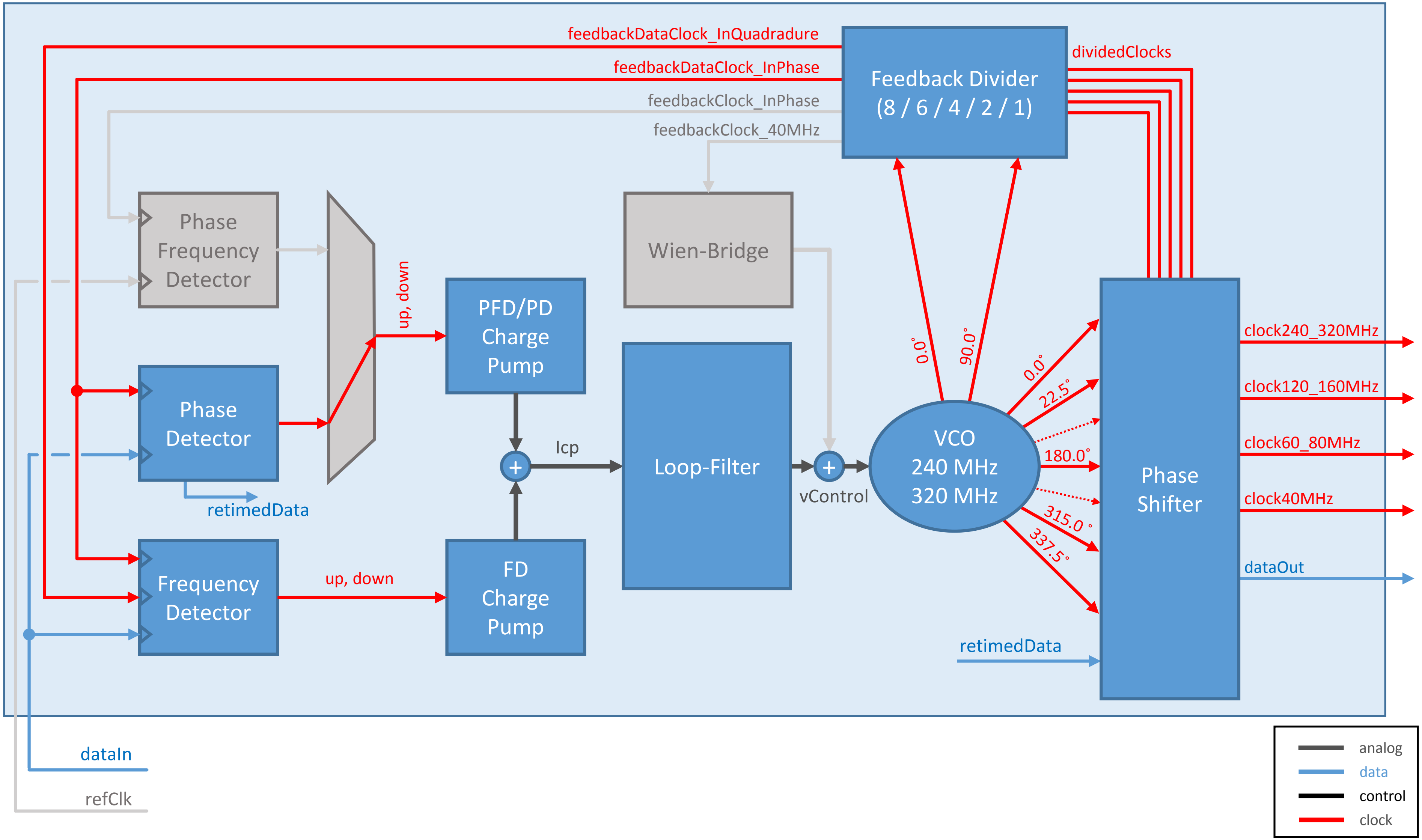
  - No power supply variations
  - No charge sharing or clock feedthrough modeling
  - Typical case only

- The loop filter parameters were swept to ensure the PLL behaviour

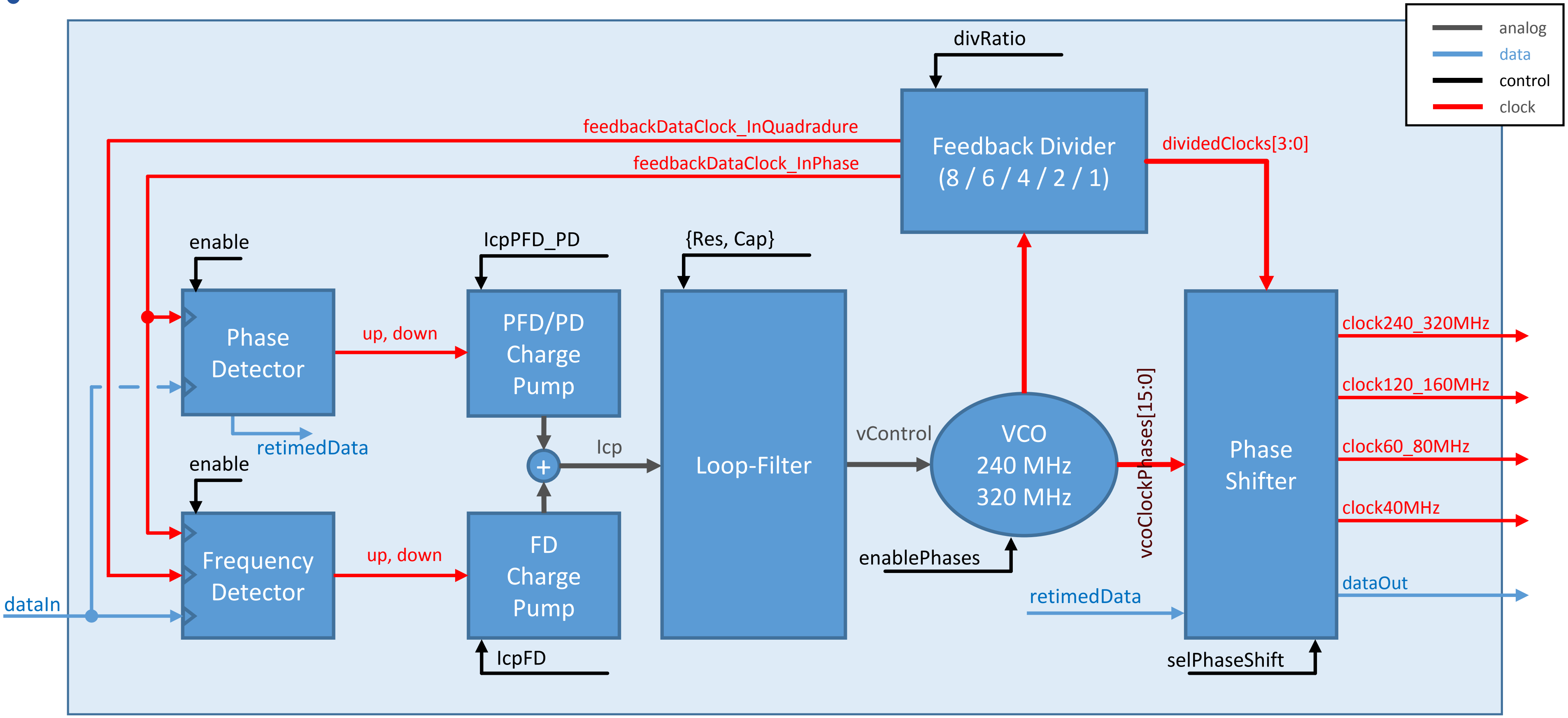
# eCDRPLL (CDR circuit operation, 1/2)



# eCDRPLL (CDR circuit operation, 2/2)



# eCDRPLL (CDR steady state)



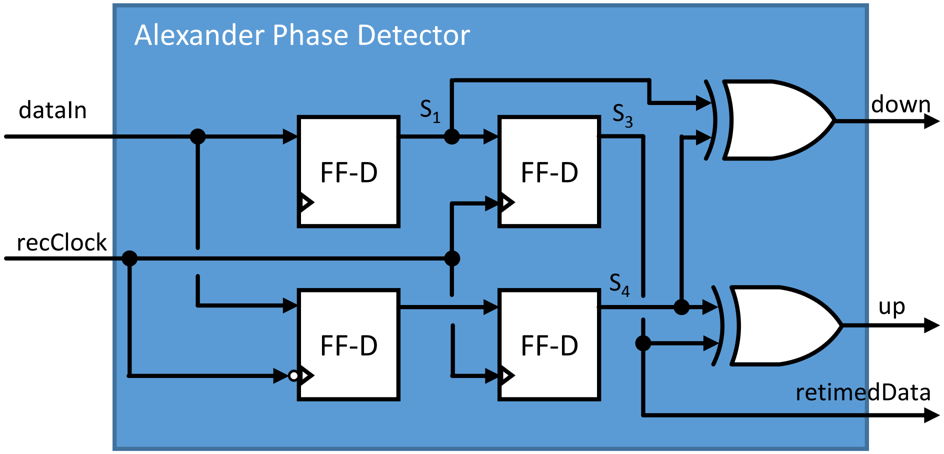
## Steady-state operation

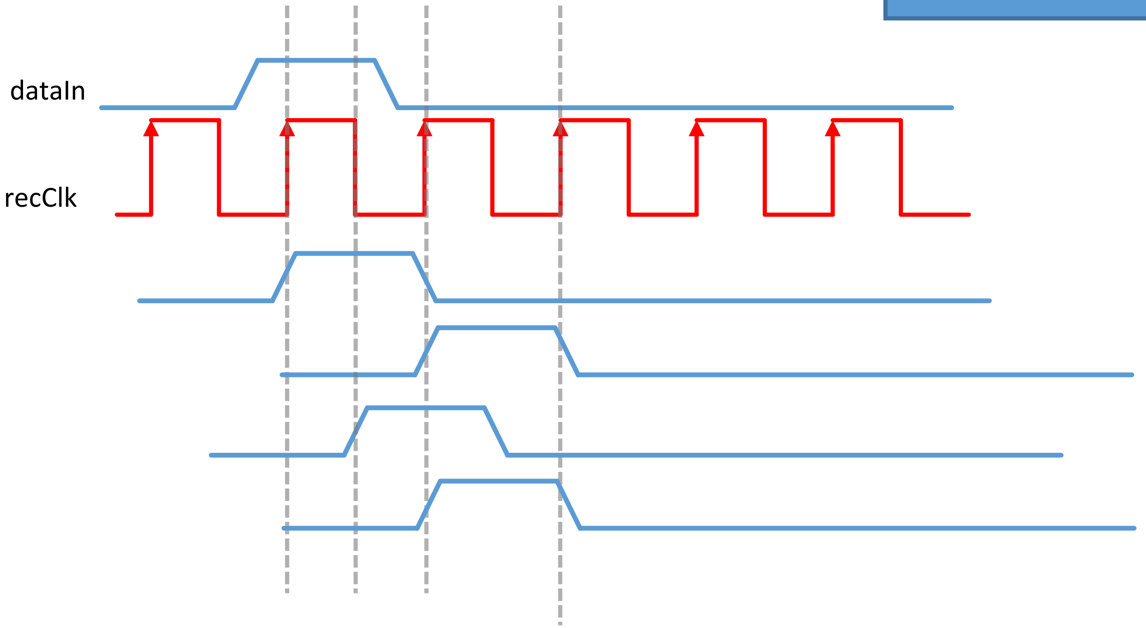
- A rotational Frequency Detector is added to increase the locking range
- Only the Alexander phase detector remains at use during steady-state
- The remaining blocks are shared with the PLL mode

Ref !

# eCDRPLL’s CDR topology

## Alexander Phase Detector





# eCDRPLL (CDR parameter exploration)

## Parameter exploration

- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance
- A bangbang CDR can be modeled as two branches [Walker model]
  - Proportional, or bang-bang branch
  - Integral branch
- To ensure stability, the phase variation due to the integral branch should be higher than the proportional branch

$$J_{BB} = N\frac{K_{VCO}RI_{cp}}{f_{vco}^{2}}$$

$$J_{INT} = \frac{I_{cp}}{C}K_{VCO}\frac{N^{3}}{f_{vco}^{3}}$$

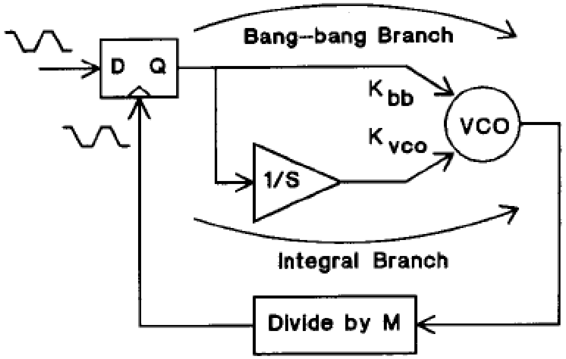


Fig. 6. Simplified clock recovery loop.

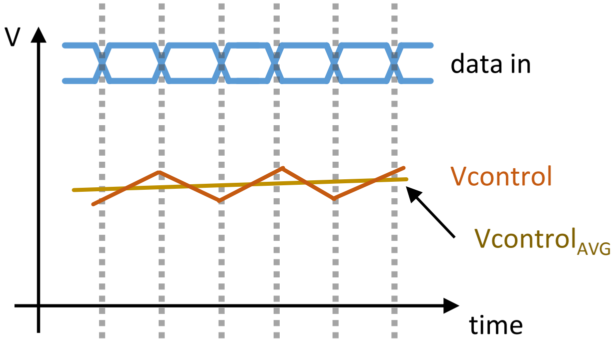
To achieve the best jitter figure

$K_{VCO}$ is fixed

$I_{cp}$ should be as low as possible

R should be as low as possible

C should be as high as possible



Ref !

$K_{VCO}$ in Hz/V

# eCDRPLL (CDR parameter exploration)

## Parameter exploration

- The loop parameters ($K_{vco}$, $I_{cp}$, R and C) will dictate the loop's performance

- A simplistic verilog model of the CDR was used to estimate the CDR's jitter and estability performance
  - PRBS-7 data input with a white-noise jitter (2.5ps, 5ps and 10ps rms) was used
  - Loop mismatches:
    - Charge pump with a leakage current of 1% of LSB (8 nA) and 1% mismatch between sink and source current
    - A filter type (R-C1)||C2 was used in the model where C2 simulates the parasitic capacitances of the charge-pump switches and the VCO
  - VCO with 1-ps rms jitter (normal distribution) and extracted transfer function
  - Feedback divider and phase shifter with back annotation delay values

  - No power supply variations
  - No charge sharing or clock feedthrough modeling
  - Typical case only

- The loop filter parameters were swept to ensure the CDR behaviour

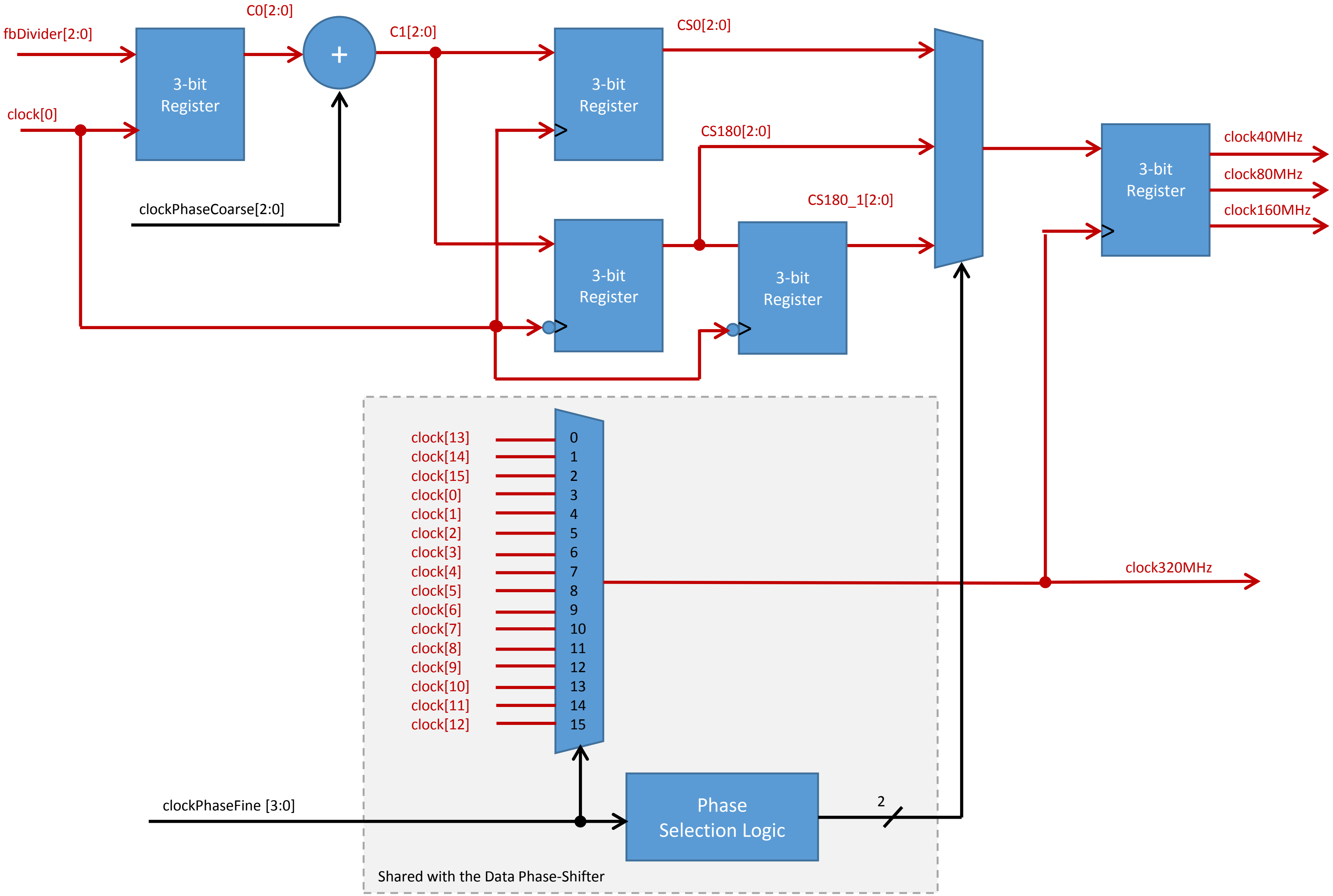
# eCDRPLL (Digital Core)

The digital core is comprised of

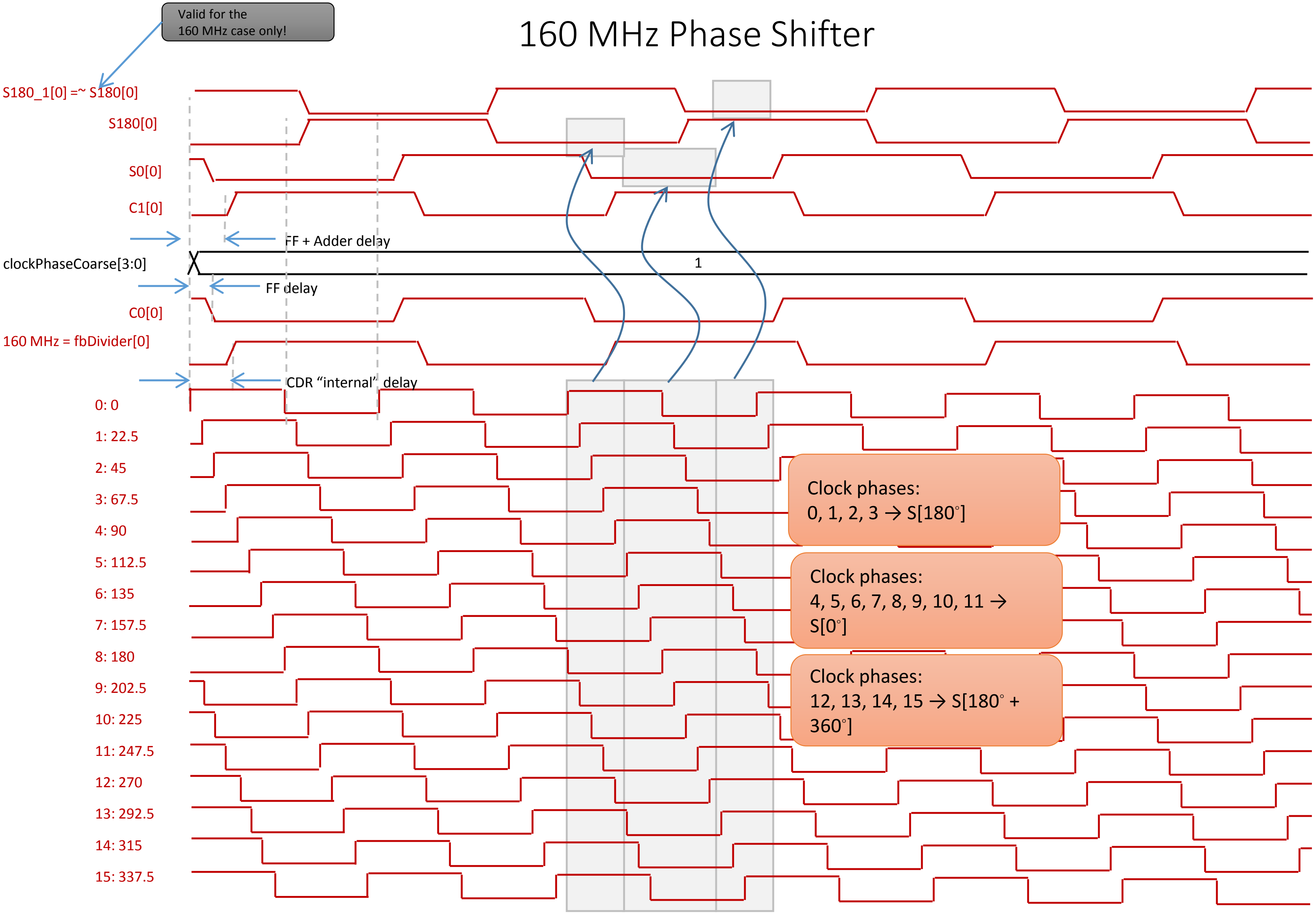
- ROM block
  - Holds known-good values for power up
  - These values can be overriden by the user
- BIST Module
  - The implemented BIST module is designed to automatically estimate relative leakage and mismatch in the forward path while in PLL mode
    - Designed to be used during production testing (wafer probing)
    - Removes the need to do lengthly jitter measurements with high-end equipment
    - Jitter estimation by measuring the forward path response while in open-loop
    - Low impact in area and power

- The digital core was triplicated using the TMRG tool

# Clock Phase-Shifter



# 160 MHz Phase Shifter



Valid for the
160 MHz case only!

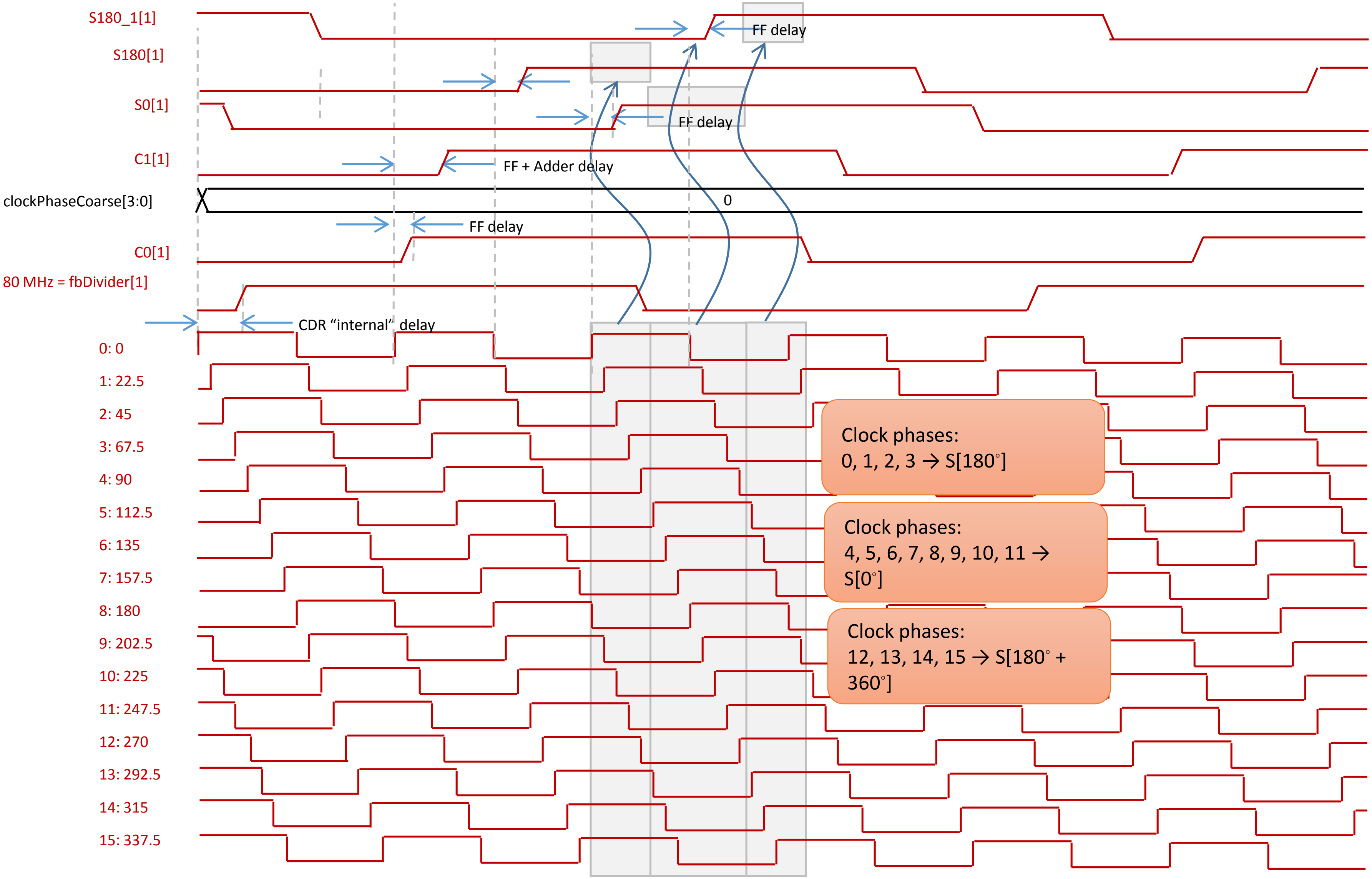
Clock phases:
0, 1, 2, 3 → S[180°]

Clock phases:
4, 5, 6, 7, 8, 9, 10, 11 → S[0°]

Clock phases:
12, 13, 14, 15 → S[180° + 360°]

The same phase selection principle applies to the 40 MHz clock

# 80 MHz Phase Shifter

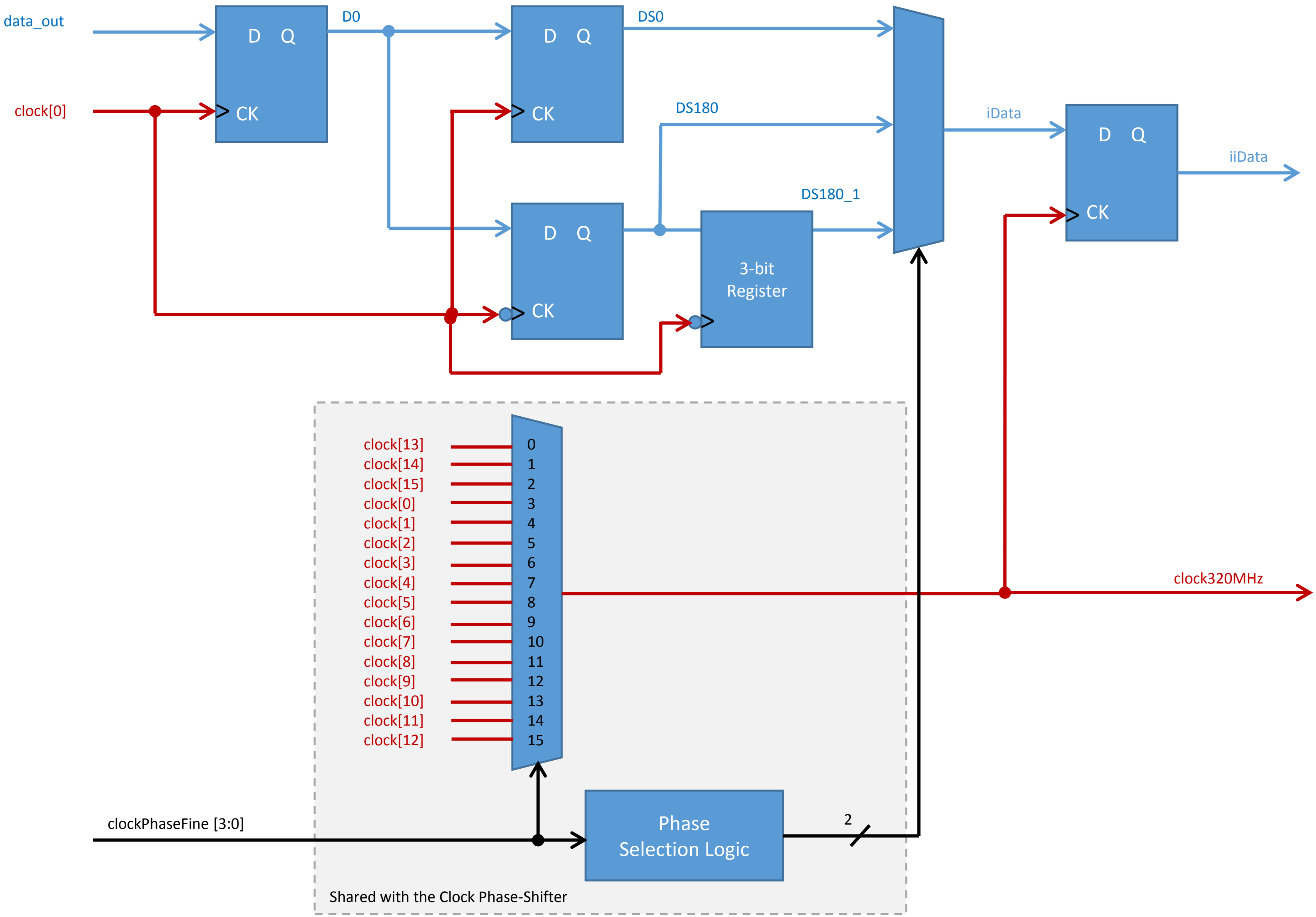


Clock phases:
0, 1, 2, 3 → S[180°]

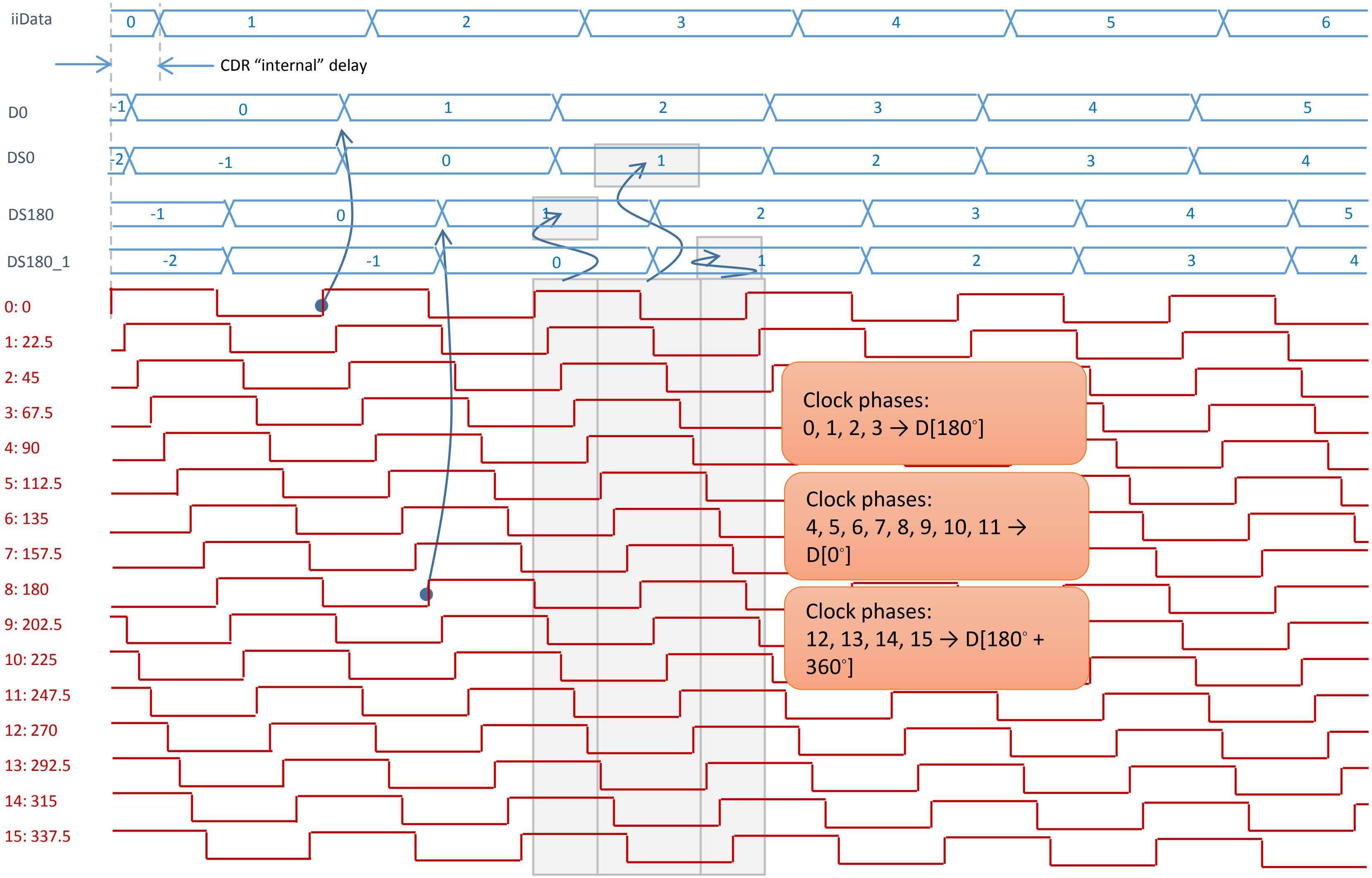
Clock phases:
4, 5, 6, 7, 8, 9, 10, 11 → S[0°]

Clock phases:
12, 13, 14, 15 → S[180° + 360°]

# Data Phase-Shifter (1/2)



# Data Phase Shifter

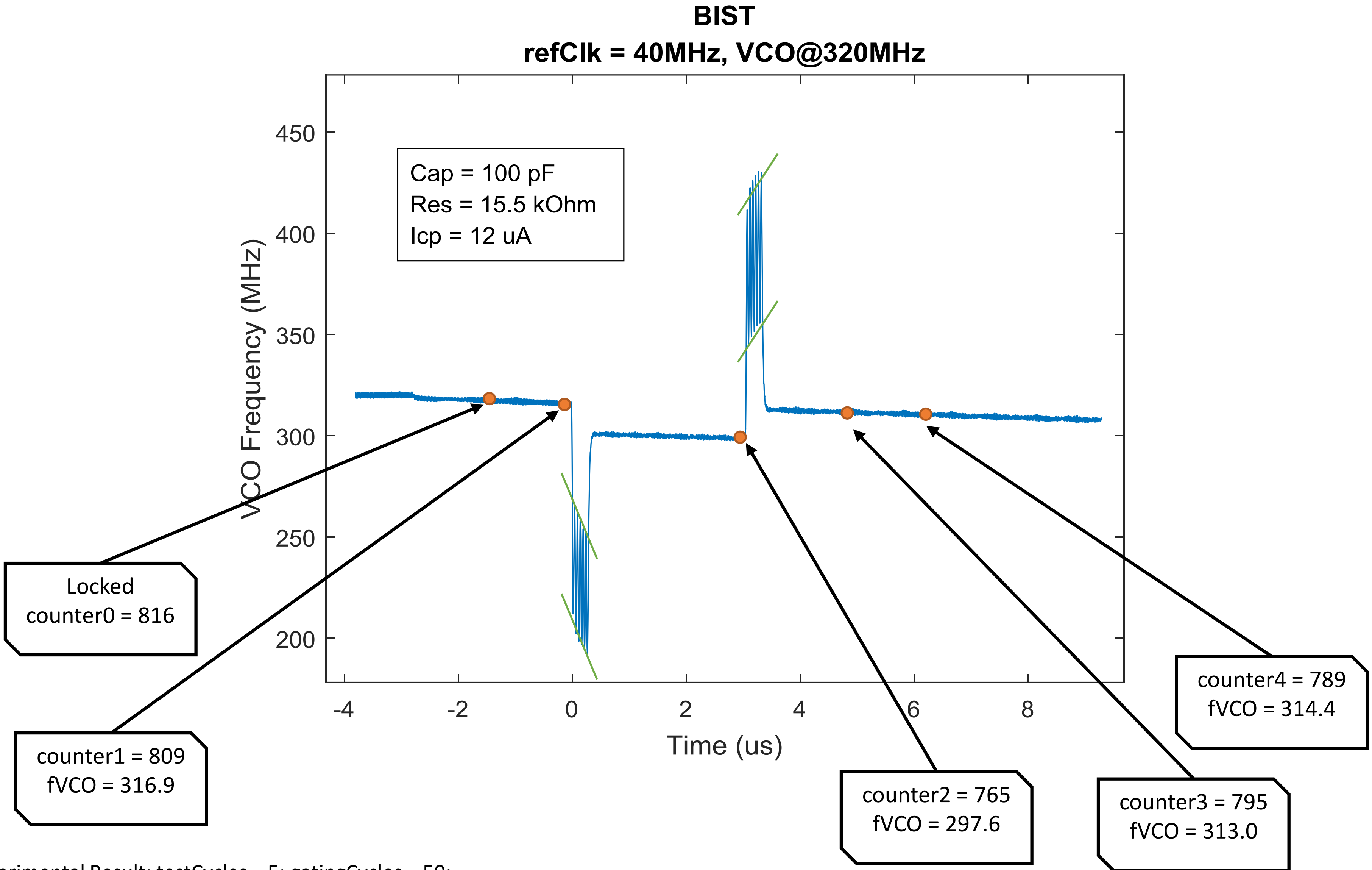


Clock phases:
0, 1, 2, 3 → D[180°]

Clock phases:
4, 5, 6, 7, 8, 9, 10, 11 → D[0°]

Clock phases:
12, 13, 14, 15 → D[180° + 360°]

# BIST



Locked
counter0 = 816

counter1 = 809
fVCO = 316.9

counter2 = 765
fVCO = 297.6

counter3 = 795
fVCO = 313.0

counter4 = 789
fVCO = 314.4

Experimental Result; testCycles = 5; gatingCycles = 50;