High-Speed Low-Jitter Clock Multiplication in CMOS

Remco van de Beek

Title: High-Speed Low-Jitter Clock Multiplication in CMOS
Author: Remco C.H. van de Beek
ISBN: 90-365-1989-6

Cover Design: Erik van de Beek [Yggdrasil]
Paul Verspaget [finishing]

This work was supported by Philips Research Laboratories, Eindhoven, The Netherlands

HIGH-SPEED LOW-JITTER
FREQUENCY MULTIPLICATION IN CMOS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof.dr. F.A. van Vught,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op donderdag 8 januari 2004 om 13.15 uur

door

Remco Cornelis Herman van de Beek

geboren op 3 september 1974

te Wageningen

Dit proefschrift is goedgekeurd door

de promotor,
prof.dr.ir. Bram Nauta,

en de assistent promotor,
dr.ing. E.A.M. Klumperink.

“And what can I tell you my brother, my killer; what can I possibly say?
I guess that I miss you, I guess I forgive you; I’m glad you stood in my way.”

Leonard Cohen

To Eva and to my parents Jannie and Kees

Samenstelling van de promotiecommissie:

Voorzitter:	
Prof.dr. W.H.M. Zijm	Universiteit Twente
Secretaris:	
Prof.dr. W.H.M. Zijm	Universiteit Twente
Promotor:	
Prof.dr.ir. B. Nauta	Universiteit Twente
Assistent promotor:	
Dr.ing. E.A.M. Klumperink	Universiteit Twente
Deskundige:	
Dr. C.S. Vaucher	Philips Research Eindhoven
Leden:	
Prof.dr. J.R. Long	TU Delft
Prof.ir. A.J.M. van Tuijl	Universiteit Twente
Prof.dr.ir. C.H. Slump	Universiteit Twente

CONTENTS

LIST OF SYMBOLS

Symbol	Explanation	Unit	Page
$\mathbb{1}$	Unit step-function	-	145
A_1, A_2	Input signal amplitudes of Phase Detector	V	89
c	Design dependent proportional constant describing delay cell jitter	$A\,s$	27
C_1	Main PLL loop filter capacitor	F	23
$C_{1,opt}$	Jitter-optimal value for PLL loop filter capacitor C_1	F	57
C_2	PLL loop filter capacitor for extra filtering of VCO input	F	48
C_f	Value of DLL loop filter capacitor	F	18
C_i, C_o	Input and output capacitance of digital gate model	F	103
C_{int}, C_{ext}	Internal and external load capacitances of a CML gate. C_{int} changes with W-scaling, C_{ext} is invariant to scaling	F	74
C_L	Load capacitance of a delay cell	F	27
d_i	Delay of certain delay cell with index i	s	30
Δd_l	Jitter added by l-th VCDL delay cell (l being a 'dummy' index)	s	20
d_{tot}	Total VCDL delay	s	20
d_{tune}	Nominal delay of VCDL delay cells	s	30
e_{CP}	Random variable describing Charge Pump current source mismatch	-	33
e_i	Random variable describing delay cell mismatch	-	30
f_l, f_h	Lower and upper integration limits for determining PLL jitter from phase noise PSD	Hz	46
f_m	Offset frequency relative to carrier	Hz	46
f_{max}	Maximum input frequency of a tri-state PFD at which frequency discrimination is possible	Hz	96

Symbol	Explanation	Unit	Page
$f_{max,PD}$	Maximum input frequency of a Phase Detector at which phase detection works properly	Hz	99
f_r	VCO offset frequency at which its intrinsic phase noise is specified	Hz	53
f_{ref}	Frequency of reference clock (at input of Phase Detector)	Hz	27
f_{VCO}	VCO output frequency	Hz	46
f_{xover}	Offset frequency at which graphs of $N^2\,S_{\phi_{neq}}$ and $S_{\phi_{nVCO}}$ intersect, jitter optimal PLL bandwidth	Hz	54
h	Time domain impulse response of linear filter	-	145
H_{LP}	Transfer function from reference input to frequency divider output	-	51
H_{OL}	PLL loop gain transfer function	-	49
H_{Ref}	Transfer function from reference input to PLL output	-	51
H_{VCO}	Transfer function from VCO output to PLL output	-	52
I_{CP}	Current of Charge Pump current sources	A	20
$\overline{i_{cp}}$	Mean Charge Pump output, averaged over one reference period	A	47
I_{CP_0}	Charge pump current of a “base” CP to be scaled	A	56
I_D	MOSFET drain current	A	74
$\overline{I_D}$	Mean output currents of current mirrors in Charge Pump	A	61
I_{D_0}	MOSFET drain current for $V_{GS} = V_{GS_0}$, as used in the alpha-power law MOSFET model	A	72
i_{LF}	Total current pumped into PLL loop filter due to charge impulse q_{LF}	A	142
I_o	Value of controlled output current source in digital gate model	A	103
I_S	Static current consumption of CML gate	A	27
$I_{S,max}$	Maximum value of CML-gate’s tail current for given input pair widths and voltage swings	A	70
I_{UP}, I_{DN}	Up and down current source values of the Charge Pump	A	33
k	Boltzmann constant	J/K	26
K_d	VCDL gain	s/V	20

Symbol	Explanation	Unit	Page
K_{PD}	Phase detector/Charge Pump combination gain, as a function of input phase error	A/rad	47
K_{VCO}	VCO gain	$(\text{rad}/s)/V$	24
$\mathcal{L}_{neq}$	Single-sideband equivalent synthesizer phase noise	dBc/Hz	123
M, M_{VCDL}	Number of delay cells in VCDL	-	18
M_{VCO}	Number of delay cells in ring oscillator	-	28
N	Frequency multiplication ratio	-	3
N_r	Reference divider ratio	-	96
m	VCDL tap number index	-	20
	Index of VCO edges after rising reference edge	-	24
n	Period number of reference clock	-	19
P_{static}	Total static power consumption of all delay cells	W	29
$P_{static,VCDL}$	Static power consumption of VCDL	W	28
$P_{static,VCO}$	Static power consumption of ring oscillator	W	28
Q	Quality factor of resonator, coil or capacitor	-	29
q_{LF}	Charge pumped into loop filter by CP	$A\,s$	19
q_{noise}	Stochastic error on charge out of Charge Pump	$A\,s$	20
R_1	PLL loop filter resistor	Ω	23
$R_{1,opt}$	Value for PLL loop filter R_1 for jitter-optimized PLL	Ω	57
R_L	Value of CML load resistors	Ω	69
$R_{xx}(t_1, t_2)$	Autocorrelation function of a random variable x	-	146
s	Laplace transform variable	-	47
S_{i_n}	PSD of both the up and down current sources in the Charge Pump	A^2/Hz	60
$S_{i_{nCP}}$	Charge Pump current noise PSD	A^2/Hz	50
$S_{i_{nUP}}$, $S_{i_{nDN}}$	PSD of up and down current sources in the Charge Pump	A^2/Hz	60
$S_{v_{tune}}$	PSD of VCO control signal	V^2/Hz	147
S_η	PSD of Gaussian white noise signal	$1/\text{Hz}$	146

Symbol	Explanation	Unit	Page
$S_{\phi_{neq}}$	Equivalent synthesizer phase noise floor PSD at the input of the Phase Detector	rad^2/Hz	50
$S_{\phi_{nDiv}}$	Phase noise PSD added by frequency divider, at Phase Detector input	rad^2/Hz	50
$S_{\phi_{no}}$	PLL output phase noise PSD	rad^2/Hz	46
$S_{\phi_{neq}}$	Equivalent synthesizer phase noise floor PSD at the input of the Phase Detector	rad^2/Hz	50
$S_{\phi_{nRef}}$	Reference signal phase noise PSD	rad^2/Hz	50
$S_{\phi_{nVCO}}$	Intrinsic VCO phase noise PSD	rad^2/Hz	52
$S_{v_{nLF}}$	PSD of the VCO control voltage noise due to loop filter's intrinsic noise	V^2/Hz	53
T	Absolute temperature	K	26
t	Time	s	10
t_d	Delay of single delay cell	s	27
t_{ovl}	Overlap time of *UP* and *DN* pulses	s	33
T_{ref}	Period time of reference clock	s	20
t_{reset}	Reset-path delay of a tri-state PFD	s	94
t_{UP}, t_{DN}	Amount of time Charge pump current sources are on per reference period	s	33
T_{VCO}	Period time of VCO	s	28
v_c	Voltage across DLL loop filter capacitor C_f, Voltage across PLL loop filter capacitor C_1	V V	20 24
V_{DD}	Supply voltage	V	28
V_{DS}	MOSFET drain-source voltage	V	71
V_{GS}	MOSFET gate-source voltage	V	27
V_{GS_0}	MOSFET gate-source voltage for which I_{D_0} is defined in the alpha-power law MOSFET model	V	72
V_{GT}	MOSFET overdrive voltage ($V_{GS} - V_T$)	V	72
V_{mix}	Mixer PD filtered output voltage	V	89
V_S	MOSFET source voltage	V	74

Symbol	Explanation	Unit	Page
v_{slice}	The mean of the digital voltage levels for a '0' and for a '1' in the digital gate model	V	103
V_{sw}	Differential voltage swing of CML gate	V	27
$V_{sw,out}$	Differential output voltage swing of CML gate	V	69
V_T	MOSFET threshold voltage	V	27
v_{tune}	VCO control voltage	V	24
W_b, W_t	Width of bottom and top differential pairs in a 2-input CML gate	μm	84
x	Gaussian white noise signal multiplied by step-function	-	145
y	Integral of Gaussian white noise signal	-	145
Z_{LF}	Total PLL loop filter impedance	V/A	48
α	Impedance Level Scaling factor	-	35
α_i	Model parameter of alpha-power law MOSFET model, describing dependence of drain current on overdrive voltage	-	72
$\gamma(\phi_m)$	An integration correction factor depending on the PLL phase margin	-	55
$\delta(t)$	Dirac-pulse in the time domain	-	142
Δd_{tot}	Jitter on VCDL delay	s	20
Δt_d	Deviation of delay of single delay cell due to noise	s	27
Δt_{div}	Jitter caused by frequency divider	s	24
Δt_m	Jitter at $m-th$ VCDL output tap Jitter on m-th VCO output edge after rising reference edge	s s	20 24
Δt_M	Jitter at last ($M-th$) VCDL output tap	s	20
Δt_N	Jitter on the VCO output edge that causes a rising edge at the frequency divider output	s	24
Δt_{PFD}	Input referred jitter of PFD	s	20
$\Delta t'_{PFD}$	Input referred PFD and Charge Pump jitter	s	22
Δt_{ref}	Jitter on reference clock	s	20
Δt_{synth}	Equivalent input referred synthesizer jitter (due to PFD, Charge Pump and frequency divider)	s	25

Symbol	Explanation	Unit	Page
$\Delta T_{VCO,l}$	Period error of VCO in *l*-th cycle within reference period (*l* being a 'dummy' index)	*s*	24
$\Delta\phi_q$	Phase difference between the in-phase divider signal *DivI* and the quadrature divider signal *DivQ*	rad	102
$\Delta\phi_{VCO}$	Phase error of VCO with respect to ideal clock with frequency ω_{VCO}	rad	143
ε_{DLL}	Normalized DLL loop bandwidth	-	21
ε_{PLL}	Normalized PLL loop bandwidth	-	25
$\varepsilon_{PLL,\max}$	Rule-of-thumb maximum normalized PLL loop bandwidth	-	27
$\varepsilon_{PLL,\mathrm{opt}}$	Jitter-optimized normalized PLL loop bandwidth	-	26
η	Gaussian white noise signal	-	145
ξ	Design dependent proportional constant describing delay cell jitter	-	27
σ_{t_o}	PLL output rms-jitter	*s*	46
$\sigma_{t_{o,eq}}$	PLL output rms-jitter due to the equivalent synthesizer phase noise	*s*	51
$\sigma_{t_{o,VCO}}$	PLL output rms-jitter due to VCO phase noise	*s*	52
σ_x	Standard deviation of some random variable *x* (Squaring gives variance of *x*)	$<x>$	21
$\sigma^2_{\phi_{res}}$	Residual phase deviation	rad^2	45
τ	Dummy integration variable	*s*	142
τ_{AND}	Delay of an AND-gate	*s*	104
τ_{FO-4}	FO4-delay (delay of inverter loaded with 4 identical inverters)	*s*	103
ϕ_e	Phase error at the input of the Phase Detector	rad	47
ϕ_m	PLL phase margin	rad	49
ϕ_{no}	Phase noise of the PLL output signal	rad	47
ϕ_{nRef}	Phase noise of the reference signal	rad	47
ϕ_o	Absolute phase of the PLL output signal, not considering its phase noise	rad	47
ϕ_{Ref}	Absolute phase of the reference signal, not considering its phase noise	rad	47

Symbol	Explanation	Unit	Page
ϕ_{VCO}	Absolute output phase of the VCO	rad	142
ω_c	DLL loop bandwidth	rad/s	21
	PLL loop bandwidth	rad/s	25
$\omega_{c,opt}$	Optimal PLL open-loop bandwidth with respect to jitter	rad/s	54
ω_{fr}	VCO free running frequency ($v_{tune} = 0$)	rad/s	24
ω_p	Angular frequency of PLL loop filter pole	rad/s	49
ω_{VCO}	VCO output frequency	rad/s	24
ω_z	Angular frequency of PLL loop filter zero	rad/s	49

1

Introduction

This introductory chapter describes the contents and the motivation of this thesis. The concept of frequency and clock multiplication is introduced, and the most significant applications and architectures of frequency and clock multipliers are shown.

INTRODUCTION CHAPTER 1

Ever since Gordon Moore anticipated an exponential growth of the number of components on an integrated circuit (IC) in 1965 [1], the semiconductor industry has followed this legendary prediction, almost making it a self-fulfilling prophecy, and ICs have become an integral part of everyday life.

The timing of internal functions inside such an IC is generally driven by a periodic signal called the clock signal. Purely digital ICs mostly need an internal "metronome" to guarantee an orderly flow of data, but also the Analog-to-Digital and Digital-to-Analog converters in a mixed-signal IC need a clock to define their conversion moments. Additionally, when two different systems are communicating, a periodic signal is usually needed that synchronizes both systems. Note that when used to up- or down-convert signals in the frequency domain, the periodic signal is referred to as Local Oscillator (LO).

The clock or LO signal is mostly synthesized based on a lower-frequency reference signal, by means of a Clock Multiplier Unit (CMU) or Frequency Synthesizer[1]. In this case there is a fixed (sometimes variable) ratio N between the output and the reference frequency.

With increasing clock rates of digital ICs such as microprocessors, the Clock Multiplier is vital because no crystals are available with a clock frequency as high as needed on-chip. Another important CMU application is in serial communication. With the bandwidth of digital communication components growing, parallel busses are being replaced with high speed serial I/O links. For chip-to-chip communication, this increases the bandwidth per I/O pin, thereby reducing chip size and cost. The timing of the individual output bits is controlled by the CMU, based on the lower-frequency clock accompanying the parallel data.

Serial optical communication systems, such as those based on SONET/SDH, utilize high speed serial data streams. It is especially this last area that requires very high quality Clock Multiplication circuits to accurately time transmitted serial data streams. Accurate timing allows for few digital repeaters in the optical path.

With the ever-growing communication bandwidths demanded by digital and RF ICs, the design of the clock multiplier building block becomes increasingly difficult and more power-hungry, especially when high timing-accuracy requirements are to be met. The trends to decrease supply voltage and to increase the number of integrated functions on one CMOS die only aggravate the effort needed to meet demands on the CMU. Also, the high $1/f$ noise corner frequencies of today's CMOS processes are a challenge to the CMU designer, as it is to any analog and RF CMOS designer.

[1] In this thesis, we use the term 'Frequency Synthesizer' or 'Frequency Multiplier' when the output signal is used 'in the frequency domain', *e.g.* in a down-converter. The terms 'Clock Multiplier' or 'Clock Multiplier Unit' are reserved when the output signal is used as a time base. This thesis mainly focusses on the latter.

This thesis deals with high-speed Clock and Frequency Multiplication. The term 'high-speed' applies to both the output and the reference frequency of the multiplier. Much emphasis is placed on analysis and optimization of the total timing inaccuracies, and on implementing a high-speed feedback mechanism that synchronizes the generated signal to the reference.

1.1 Jitter and Phase Noise

The signal generated by the CMU should ideally be a periodic signal with a constant frequency. In reality, however, the frequency of the output signal fluctuates around its mean value (which is N times the reference frequency). This fluctuation can be caused by thermal and $1/f$ noise sources in the CMU circuit, by supply noise and substrate bounce, by device mismatches in the circuit or by impurity of the reference signal.

Definition and measurement of these output signal inaccuracies can, depending on the application, both be done in the frequency or in the time domain. When observed in the time domain, the inaccuracies result in variations in the period time of the generated signal, and are referred to as *jitter*. The frequency domain variations result in spectral components at frequencies other than the intended output frequency, and are referred to as *phase noise*[2]. Given that both measures are closely linked [2], it is generally possible to calculate the jitter using the output signal's phase noise. In some cases, this is a preferred method, *e.g.* because measurements or analyses are more easily performed in the frequency domain.

In general, the jitter or phase noise on the generated output signal results in a degradation of the signal-to-noise ratio of the signals clocked by or mixed with it. Examples are the signal-to-noise reduction of a receiver using the multiplied reference frequency as its Local Oscillator (LO), or the increased bit-error-rate (BER) of received data that was transmitted using an impure clock. Because of the importance of the CMU's output signal purity, this thesis puts much emphasis on its analysis and minimization.

To be able to analyze the output 'jitter', first a quantitative definition of jitter is needed. There are many different definitions for jitter available in literature [2,3]. In this work, a very simple and intuitive definition will be used:

> *Jitter is the random or systematic deviation in time of the zero-crossings of a certain generated clock with respect to corresponding zero-crossings of an* ideal *clock. The ideal clock has zero-crossings that are separated by a constant amount of time which equals the mean period of the generated clock.*

The jitter can thus be expressed as the variance of these random variations, or as their standard deviation (rms jitter). Another measure often used is the so-called peak-to-peak jitter, a number indicating the distance (in time) between the maximum positive deviation of a clock edge and the maximum negative deviation of such an edge. This number is ill-defined when

[2] Amplitude noise will also show up as unwanted spectral components, but the distinction is often easily made in both measurements as well as in analysis.

jitter is partly due to gaussian noise (it depends on measurement duration and sheer "luck"). Therefore, this jitter measure is not used directly in this thesis. There is, however, a rule-of-thumb stating that the peak-to-peak jitter is roughly ten times the rms jitter (corresponding to 5σ in both positive and negative directions, assuming a gaussian distribution).

1.2 Applications

This section discusses the main applications of frequency and clock multipliers.

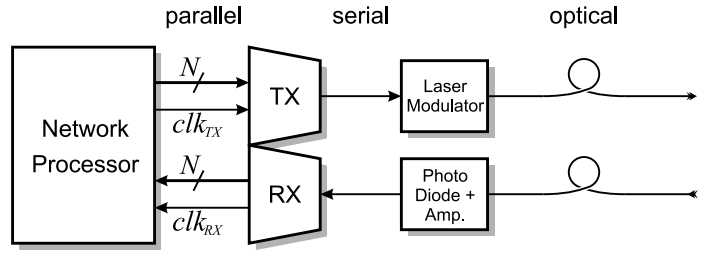
1.2.1 Serial communication

An important application area of Clock Multiplication is in serial data transmission. Figure 1.1(a) shows the basic architecture of a traditional optical transceiver [4,5] as an example. The transmitter (TX), see Figure 1.1(b), multiplexes the parallel data stream into one high-speed non-return-to-zero (NRZ) serial bitstream, which is sent to the laser driver in order to be transferred optically to the glass fiber. The receiver (RX), see Figure 1.1(c), uses the amplified current from the photo diode to recover the clock from the NRZ bitstream. This recovered clock is used to sample the bitstream and to demultiplex the data.

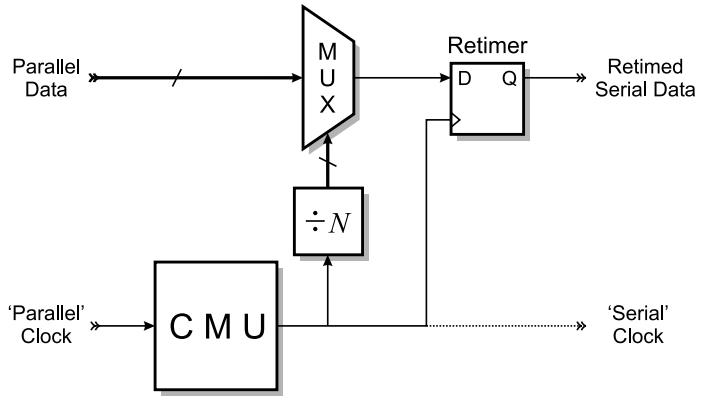
The clock that accompanies the parallel data stream in the transmitter (the "parallel" clock) does not provide enough information to accurately time the serial outgoing bitstream by itself. This means that timing information has to be derived in between the transitions of the parallel clock. The most obvious way to obtain this extra information is by using a Clock Multiplier Unit with the parallel clock used as reference input, see Figure 1.1(b). Multiplexing data streams in principle does not necessarily require a clock at the frequency of the output bit-rate. Using a binary tree-type data multiplexer [6–8], a clock at half the bit-rate frequency is sufficient to control the last 2:1 multiplexer element. Using multiphase clocks, even lower clock frequencies will suffice [9–11]. However, such solutions make the jitter of the output stream sensitive to clock duty cycle and offsets in the multiplexer block [4]. For low-jitter transmission, it is therefore generally necessary to retime the output stream using a clock frequency equal to the bit-rate [12,13], as illustrated in Figure 1.1(b).

The jitter of the output bitstream is determined by the jitter of the CMU and that added by the retimer flip-flop. Optical communication protocols, such as SONET/SDH, have stringent specifications with respect to the optically measured jitter on the serial bitstream [14], resulting from the demand that very few repeaters are needed in the optical path.

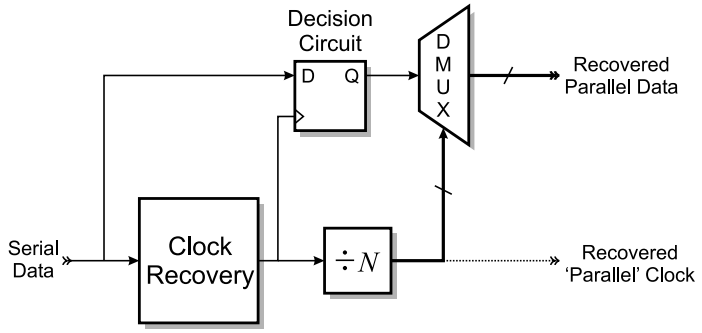
For example, a 10 Gb/s OC-192 transmitter should have less than 0.1 UI_{P-P} (unit interval, peak-to-peak) and 0.01 UI_{RMS} (unit interval, rms) jitter, measured over the bandwidth of 50 kHz to 80 MHz. This translates to 10 ps peak-to-peak and 1 ps rms jitter [15]. The jitter demands on the CMU, as a result, are very strict. Dividing the jitter specifications evenly among the CMU and the laser driver, leads to a CMU jitter specification of 7.1 ps peak-to-peak and 0.71 ps rms jitter [16]. Therefore, low jitter operation of the CMU is a center topic in this thesis.



(a) Optical transceiver architecture.



(b) Low-jitter data serializer architecture.



(c) Data deserializer architecture.

Figure 1.1: Application of a CMU in a data transmission system.

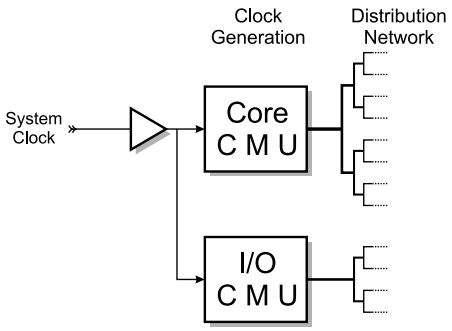


Figure 1.2: Simplified clock generation and distribution topology.

1.2.2 Digital Clock Generation

Another important application field of a Clock Multiplier is in clocked digital circuits, such as a microprocessor [17–19], in which the internal clock frequency is higher than the external system clock. A simplified clock distribution topology is shown in Figure 1.2. The reason the external clock has a lower frequency than used internally is mainly that the crystal oscillators that generate the system clock generally have a sub-gigahertz operation frequency limit. Also, distribution of a high frequency clock across a PCB is more power-hungry than distributing a clock with a lower frequency from which a high-frequency clock is generated inside the ICs by means of a Clock Multiplier Unit.

Note that the jitter specification of the Clock Multiplier in a large digital circuit, though important, is not as strict as for an optical data transmitter. This is because the internal distributed clock jitter is usually dominated by the distribution network and buffers, causing high clock skew [19,20]. However, for smaller digital ICs, the clock jitter caused by the CMU is significant, as it directly affects the maximum operation frequency of the digital circuitry. The hostile digital environment makes designing the CMU a challenging task.

1.2.3 Tuning Systems

In a typical radio receiver, efficient signal processing and demodulation is allowed for by down-converting the received antenna signal to a more convenient Intermediate Frequency (IF). Figure 1.3 *e.g.* shows a simplified direct-conversion radio receiver architecture. The Frequency Synthesizer building block is responsible for generating the Local Oscillator (LO) signal that is mixed with the antenna signal. Channel selection can be done by choosing the LO frequency.

In such tuning systems, the Signal-to-Noise ratio of the mixer outputs is partly determined by the spectral purity of the frequency synthesizer [21]. Generally, the channel spacing, the spectral purity and the LO's settling time are traded, via the reference frequency [22].

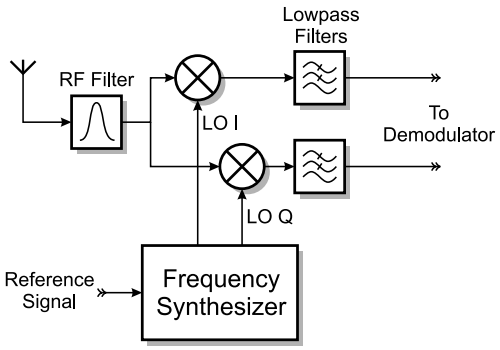


Figure 1.3: Direct-conversion receiver architecture.

1.3 Frequency and Clock Multiplying Architectures

Something all clock multipliers have in common, is that timing information has to be generated in between the edges of the reference clock. Generation of this new information can be done in different ways. This section focusses on the different methods to accomplish this task by discussing the most important architectures used to multiply the frequency of a reference signal by an *integer* number N.

1.3.1 Phase-Locked Loop

The most well-known architecture performing frequency multiplication is the Phase-Locked Loop (PLL) with a frequency divider in its feedback path. In a PLL, a Voltage Controlled Oscillator (VCO) generates the high output clock frequency. This VCO is synchronized to the reference clock by means of a feedback control loop consisting of a Phase Detector (PD) whose output is low-pass filtered by the loop filter, as shown in Figure 1.4. The PD measures the phase error of the divided VCO signal as compared to the reference signal. Because of the frequency divider in the feedback loop, the output frequency of the PLL will be exactly N times that of the reference clock (N being the frequency divider ratio). If the frequency divider is designed such that it divides by a constant integer, this PLL architecture is called an integer-N architecture.

Apart from synchronizing the oscillator to the reference signal such that its output frequency is stable and predictable, the PLL can also clean up the jitter generated in the oscillator, thereby reducing the jitter that would have been generated by a free-running oscillator.

1.3.2 Delay-Locked Loop

The VCO of the PLL is said to ‘accumulate’ its jitter [23, 24] (which is obvious in a ring oscillator structure, but also holds for a resonator type oscillator). In a Delay-Locked Loop (DLL) [25], the VCO is replaced by a delay line into which the reference clock is injected,

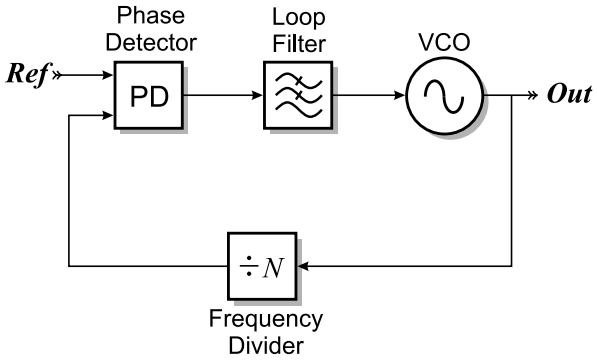


Figure 1.4: The integer-N PLL architecture.

thus preventing the accumulation of jitter [23, 24]. The obvious conclusion, therefore, is that the output jitter of a DLL is lower than that of a PLL[3], where the VCO jitter is only slowly removed by the feedback loop. For this reason, architectures based on a Delay Locked Loop (DLL) have been successfully used as Clock Multipliers [26–28].

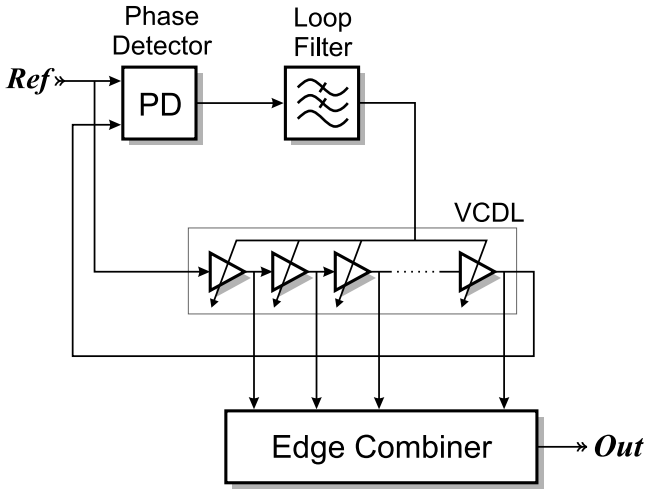
The DLL based clock multiplier is shown schematically in Figure 1.5(a). The feedback loop controls the delay of the Voltage Controlled Delay Line (VCDL) such that it is equal to one reference period (although a delay of multiple reference periods is also a valid solution of the loop, which can be a problem in practice [27]). The extra timing information needed to generate the high frequency clock is obtained by using a VCDL that consists of several tuneable delay cells, in this way generating multiple phases of the low frequency clock. These phases are combined into one high frequency clock using a circuit that is referred to as Edge Combiner.

The feedback mechanism of a DLL consists of basically the same elements as used in a PLL: a Phase Detector detects the phase error that is due to the VCDL generating the wrong delay, and a loop filter that, for a DLL, usually consists of a simple capacitor. In a PLL such a simple filter would lead to stability problems because of the integrating function of the VCO used in a PLL [29]. The 'guaranteed' stability[4] of a DLL is sometimes an argument for choosing this architecture [25, 30].

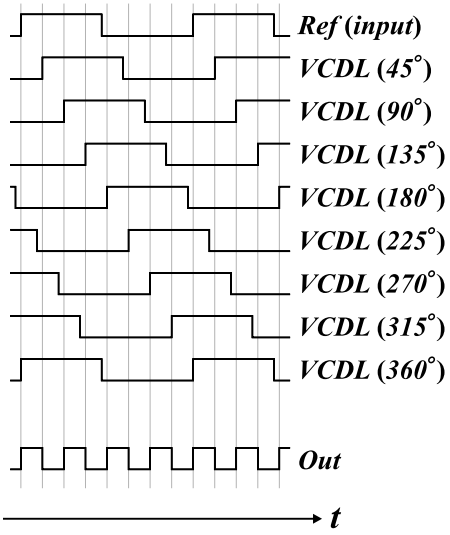
The edge combination process that generates the high frequency output clock from the different phases of the input clock is illustrated in Figure 1.5(b), where the frequency multiplication factor N equals 4.

[3]Note that in Chapter 2 this is examined in much more detail, leading to a, somewhat surprising, opposite conclusion.

[4]Chapter 2 demonstrates the exact stability conditions for a DLL, that can in fact become unstable.



(a) The DLL plus edge combiner



(b) The edge combination process for a multiplication factor $N = 4$

Figure 1.5: The DLL-based multiplying architecture.

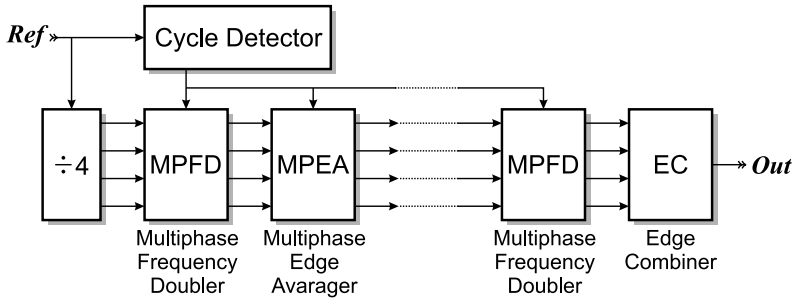


Figure 1.6: Clock Multiplier based on Direct Clock Cycle Interpolation Interpolation.

1.3.3 Clock Interpolation

In some applications, the fact that a clock multiplier based on a feedback loop takes many reference clock periods to reach a stable output clock, can be problematic. An example is a "Clock on Demand" clock synthesizer, that can be started and stopped in order to save power. Although an adaptive bandwidth PLL shows great improvement in the settling time of a frequency synthesizer [31], while maintaining low-jitter operation when in lock, still several reference cycles are needed to reach a stable output signal. Figure 1.6 shows a clock multiplier [32] that does not use a global feedback loop to synthesize its output clock, and, as a result, obtains a stable clock after just over one reference clock cycle.

The reference clock frequency is first divided by four in order to generate a 4-phase clock. The MultiPhase Frequency Doubler (MPFD) blocks generate a 4-phase clock with the a frequency twice that of the 4-phase clock applied at their inputs. The necessary extra timing information is obtained by using interpolator circuits that create transitions between already present transitions. The jitter introduced on the MPFD output clocks due to mismatches in the digital gates and interpolators used, are reduced by another interpolating stage called Multi-Phase Edge Averager (MPEA). The Cycle Detector block roughly detects the input frequency and adjusts internal capacitors based on that information.

Although this type of clock multipliers is useful when the settling time of the output clock is a crucial specification, it is less suitable for low-jitter operation, due to the large number of gates and interpolators in series, introducing jitter due to both thermal noise and supply and substrate noise. Also, this structure can only generate multiplication ratios that are a power of two (although in a data transmission this is generally required).

1.4 Motivation

Designing a Clock or Frequency Multiplier involves making choices on different hierarchical design levels, ranging from choosing the most promising architecture for a given application, to dimensioning the CMU's building block parameters, to implementation on transistor level and chip layout. This thesis describes important considerations, mostly relating to designing a

low-jitter Clock Multiplier, but much of the gained insight applies to Frequency Synthesizers as well.

The recent shift toward DLL-based multipliers led to examine the statements used to found this choice, namely that of the higher PLL output jitter due to the jitter accumulation effect occurring in oscillators. The analysis in [23] does not investigate frequency multiplying circuits (the output frequency being equal to the reference frequency). To make a sound choice for a frequency multiplying architecture, this should however be considered, as well as another source of jitter: device mismatch. One of the first goals of this thesis, therefore, is to establish a solid analysis of both PLL and DLL-based Clock Multipliers, taking into account all important sources of output jitter.

Designing a PLL-based clock multiplier with a relatively high reference frequency (which is beneficial for the multiplier's output jitter as discussed in this thesis) involves careful design of the Phase Detector building block. Because the most popular solution (the tri-state Phase Frequency Detector [33]) has an inherent speed limitation due to an internal feedback loop, part of this thesis focusses on high-speed phase detection that avoids this feedback loop.

Designing the digital building blocks of the CMU (such as the Phase Detector and the frequency divider) is preferably done in the MOS Current Mode Logic family (MCML or just: CML), because of its low generation of supply and substrate noise, and its low sensitivity to that same type of noise. Because there was no simple 'manual' available that a sub-micron CMOS circuit designer can use to easily dimension the transistors in a CML gate, this thesis describes an easy method of dimensioning the transistors of a high-speed CML gate.

The implementation aspects examined in this thesis are mainly focussed on CMOS technologies (although many of the earlier derivations apply to frequency multiplication in general, without focussing on a specific technology). The most important reason to focus on CMOS is the high level of integration that can be achieved, implementing both the digital and analog parts of a complete system on the same die. This eliminates area and power-hungry chip-to-chip interfacing. Also, the low supply voltage used in modern CMOS processes often allows low power systems. On the other hand, it should be noted that high $1/f$ corner frequencies and a low f_T as compared to modern bipolar processes make the designer's task difficult.

1.5 Thesis Outline

Chapter 2 of this thesis investigates the first choice to be made when designing a clock multiplier: its architecture. The most important architectures eligible for a high-speed and low-jitter CMU, the PLL and the recently popular DLL, are compared analytically with respect to output jitter for a given power consumption. Both jitter due to stochastic noise sources as well as due to mismatched components are discussed and compared, leading to the conclusion that a PLL performs better when used as a multiplier than a DLL.

Because it is concluded that a PLL is the most promising architecture for a low-jitter CMU, Chapter 3 discusses some important PLL design considerations: loop filter design and jitter optimization. Frequency domain analysis is used in place of the more direct time-domain

analysis for various reasons, but most importantly because $1/f$ noise sources and complex loop filters are difficult to examine in the time-domain. Chapter 3 also examines the dependence of the PLL output jitter as a function of the reference frequency applied at its input.

Seeing that a considerable part of most frequency-multiplying PLLs consists of digital building blocks, Chapter 4 discusses the MOS Current Mode Logic (CML) family of digital circuits. This logic family generates little supply disturbances while also being relatively insensitive to such disturbances. This is an important consideration in choosing to apply the Current Mode logic family, as PLL building block disturbances result in output jitter and should therefore be minimized. Because CML gates are in general not available in digital gate libraries, and information on how to dimension these gates properly is scarce, Chapter 4 discusses a simple approach to dimension CML gates. The speed of the gates is considered in relation to some degrees of freedom in the CML gate.

Because an important conclusion of Chapter 3 is that it is beneficial for the multiplying PLL to utilize a high-frequency reference signal (both with respect to generated jitter as well as the size of the integrated loop filter components), Chapter 5 focusses on high-speed Phase Detectors. A new Phase Detector is proposed that overcomes the speed limitation of a conventional tri-state Phase-Frequency Detector by getting rid of the internal reset feedback loop.

As a demonstrator, a 2.5-to-10-GHz Clock Multiplier is discussed in Chapter 6, that was realized in a standard 0.18μm CMOS process. The Clock Multiplier employs the fast Phase Detector that was introduced in Chapter 3, in combination with a Frequency Detector that ensures correct locking without disturbing the loop once phase lock has been achieved. The rms jitter of the generated 10 GHz clock is only 0.22 ps (corresponding to 2.2 mUI_{RMS}), the peak-to-peak jitter is 2.2 ps (corresponding to 22 mUI_{P-P}). These numbers are well below the OC-192 SONET specification.

Finally, Chapter 7 summarizes the most important conclusions that were arrived at in this thesis and gives an overview of the original contributions and the publications originating from the work described in this thesis.

Comparing DLL and PLL

This chapter describes a comparison between PLL- and DLL-based clock multipliers. The analysis is based upon a mathematical analysis of output jitter due to both random noise sources and device mismatch. The analyses are performed in the time-domain, to accurately describe the time-discrete nature of the CMU designs and to correctly model the delay of the DLL delay line.

2.1 Introduction

In the previous chapter, two important clock multiplier architectures were described: the frequency-multiplying Delay Locked Loop (DLL), that employs an Edge Combiner to generate the output clock, and the more conventional integer-*N* Phase Locked Loop (PLL). When having to design a CMU circuit, one of the first considerations is the choice of architecture, based on the design specifications. This chapter compares both architectures, mostly on grounds of output jitter [34].

The DLL-based clock multiplier is said to benefit from the periodically 'resetting' of the internal Voltage Controlled Delay Line (VCDL) with respect to jitter generated in the VCDL every time a new reference edge is applied at the input [23]. The integrating character of the VCO in the PLL, on the other hand, fully relies on the loop to remove the noise and preventing VCO random-walk. However, in [23], the consequences of frequency multiplication are not taken into consideration: the loops considered have an output frequency equal to the input frequency.

The analysis presented in this chapter complements that of [23] in several ways. First, by taking the effects of frequency multiplication into account. In our analysis, structures are examined where the output frequency is an integer multiple of the reference frequency. In this way, a PLL-based clock multiplier solution can be compared to a DLL-based clock multiplier, and new design considerations are obtained. Second, all important noise sources are included in the jitter analyses, as opposed to including merely the VCDL-noise in the DLL and the VCO-noise in the PLL, as is done in [23].

The noise analysis is performed in the time domain, taking account of the sampled nature of the phase detection in both the PLL and the DLL. Conventionally, such an analysis is performed in the frequency domain, first modelling the loop by a linear time-continuous system [35]. This method, however, does not take into account the sampled nature of the loops. But more importantly, this time-continuous approximation makes modelling the DLL delay-line delay time troublesome and inconvenient. This delay, as well as the sampled nature of the phase detection process, is modelled naturally using a time-discrete time-domain analysis.

Jitter due to various noise sources is treated first in this chapter. Section 2.2 covers DLL jitter; PLL jitter is discussed in section 2.3. A comparison follows in section 2.4. There is, however, another important possible source of timing errors, namely device mismatch, most importantly in the VCDL delay cells and in the Charge Pump. Device mismatch may be measurable as systematic output jitter at the CMU output clock, and will appear as spurious signals in the output frequency spectrum [36]. In section 2.5, the consequence of mismatch on the output jitter is analyzed.

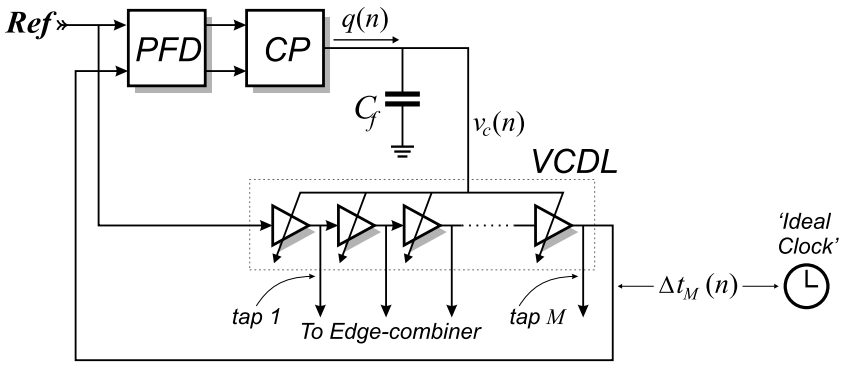


Figure 2.1: The DLL-model that is used; the 'ideal clock' illustrates the jitter definition used here.

Because the mismatch parameters depend on the chip area of the devices, the effect of scaling on the delay cell mismatch is analyzed in section 2.6, using a technique called Impedance Level Scaling or W-scaling [37]. This design technique proves useful in decoupling the noise and mismatch properties of a circuit from other properties such as speed or linearity.

Section 2.7 discusses simulations concerning both jitter due to noise and jitter due to mismatch. A summary of the most important conclusions is given in section 2.8.

2.2 Analysis of DLL jitter due to noise

In this section, the effect of different noise sources on DLL output jitter is analyzed. First, a mathematical DLL model, based on difference equations, is derived, which is then used to calculate the output jitter due to different noise sources in the architecture and due to the jitter on the reference clock.

In the coming DLL analysis, we assume that only the rising edges of the different clock phases are used (Figure 1.5(b) being an example of this). Thus the number of delay cells M in the VCDL equals:

$$M = 2N \tag{2.1}$$

where N is the ratio between the output frequency of the edge combiner and the incoming reference frequency.

2.2.1 Mathematical Model of the DLL with Noisy Building Blocks

For the stochastic DLL jitter analysis, the model shown in Figure 2.1 is used. Naturally, the 'ideal clock' is no part of the actual DLL; it is merely being shown to illustrate the concept of jitter that is being used here.

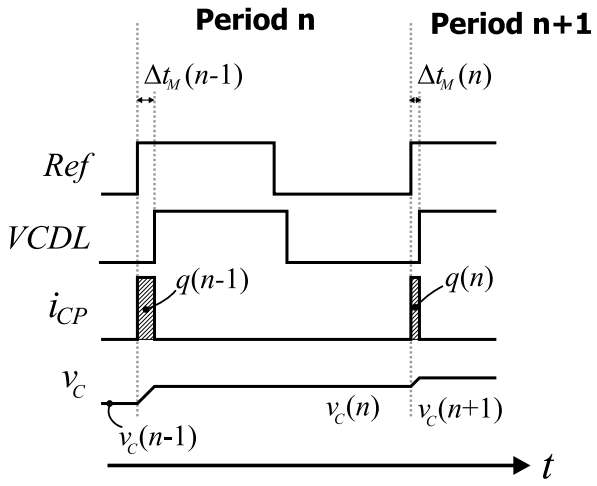


Figure 2.2: Conventions used in the difference equations.

The PFD[1] compares the zero-crossing times of the reference to those of the last tap of the VCDL. The Charge Pump (CP) converts the measured time difference into a charge $q_{LF}(n)$ which is pumped into the loop filter (a simple capacitor), thus integrating this charge. Note that parameter n indicates the period number of the input clock; this variable is used in the difference equations that are derived shortly. The conventions used in the difference equations to be derived are clarified in Figure 2.2.

The DLL noise analysis depends on a number of assumptions which are listed below:

1. The loop has successfully locked to the state in which the VCDL delay equals the period time of the reference clock. This implies that the loop is stable.

2. The mean VCDL control voltage in lock equals 0 Volts. This simplifies analysis, while the results of the jitter calculations do not depend on this assumption because of the linearity of the system.

3. The current the CP delivers can be modelled by charge pulses with a Dirac-pulse shape, which is allowed if the jitter is small compared to the reference period time.

4. All noise sources are white. This implies assuming no correlation between the noise contribution of a noise source in a certain period of the reference clock and previous contributions of the same source. A general statement about the validity of this assumption is hard to make. In theory, $1/f$ noise in the Charge Pump for example yields infinite jitter if integrated starting from DC. In practice however, there will be a lower limit on

[1]The use of a tri-state Phase Frequency Detector (PFD) is assumed in this chapter, as this is the most popular phase detector in use, see chapter 5.

the frequency from which to integrate the phase noise, depending on measurement time or system specifications, bounding the jitter. Using conventional continuous modelling of the DLL behavior and a reasonable lower integration limit, it can be shown that the $1/f$ corner frequency should be one to two decades below the DLL bandwidth for the white noise to be dominant (*e.g.* if the phase noise is to be integrated from 1 kHz up to 10 MHz and the $1/f$ corner frequency is at 1 MHz, the white noise energy is already dominant).

5. All noise sources are uncorrelated to the other noise sources in the loop.
6. The jitter contributed in a certain period of the input clock by a certain delay cell is not correlated to that delivered by another delay cell.
7. The variance of the jitter of every delay cell is equal. This is reasonable if all delay cells are realized equally and if the input signal shape of every delay cell is the same.
8. The loop behavior is linear, meaning that the output jitter contributions of every noise source can be calculated separately. The total jitter can then be calculated by adding the different contributions power-wise. This assumption is reasonable as long as the jitter remains low.

The tuning voltage v_c determines the delay of the VCDL d_{tot} according to:

$$d_{tot} = T_{ref} - K_d v_c + \Delta d_{tot} \tag{2.2}$$

where T_{ref} equals the period time of the clock, K_d is the gain of the VCDL, expressed in [s·V^{-1}], and Δd_{tot} is the jitter added by the VCDL.

The charge that is pumped into the loop filter capacitor by the CP is given by:

$$q_{LF}(n) = I_{CP}\left\{\Delta t_M(n) - \Delta t_{ref}(n) + \Delta t_{PFD}(n)\right\} + q_{noise}(n) \tag{2.3a}$$

where $q_{LF}(n)$ is the charge that the CP pumps into the loop filter after input period number n with q_{noise} denoting the part of that charge caused by a noisy CP, I_{CP} is the Charge Pump current, $\Delta t_M(n)$ is the jitter at the last (M-th) output tap of the VCDL after the n-th input period and $\Delta t_{PFD}(n)$ is the detection error that the PFD makes due to its input referred voltage noise, which will be discussed in more detail later. The term $\Delta t_{ref}(n)$ denotes the timing error in the reference edge that appears at the PFD input after input period number n.

Knowing the charge that is pumped into the filter, the VCDL control voltage during the n-th input period is given by:

$$v_c(n) = v_c(n-1) + \frac{q_{LF}(n-1)}{C_f} \tag{2.3b}$$

with C_f the value of the loop filter capacitor.

The final difference equation describes the timing error of the output taps Δt_m (m being the tap number), using (2.2):

$$\Delta t_m(n) = -\frac{m}{M} K_d v_c(n) + \sum_{l=1}^{m} \Delta d_l(n) + \Delta t_{ref}(n-1) \tag{2.3c}$$

The first term of this equation is the jitter due to tuning voltage v_c deviations, the second jitter term is due to $\Delta d_l(n)$, the jitter added by the l-th delay cell in input period number n. The jitter appearing on the output taps during period number n are partly directly due to reference jitter at the input of the VCDL at the end of the previous period, the third term $\Delta t_{ref}(n-1)$.

2.2.2 DLL output jitter due to noise

In this section the jitter that will result at the different output taps of the VCDL due to its own jitter is analyzed first, using the set of difference equations (2.3). Then, in a similar fashion, the output jitter due to the PFD and CP noise are calculated as well as the output jitter due to the reference jitter. The general calculation method is demonstrated in Appendix A.

Deviations in the tuning voltage, as well as jitter added by the delay cells will result in jitter on the taps of the VCDL. Also jitter present on the reference clock that is fed into the VCDL causes jitter on the output taps. Using the assumptions given before, the effect of both the tuning voltage errors and the jitter added by the delay cells will be worst at the last output tap of the VCDL, which means the jitter variance will be highest at the last output tap.

To isolate the effect of the delay cell noise, the other noise sources are neglected, using Assumption 8 of section 2.2.1. Following the method described in Appendix A, we can find the variance of the signal Δt_M, which is the jitter variance of the last output tap of the DLL:

$$\sigma^2_{\Delta t_M} = \sigma^2_{\Delta d} \cdot M \frac{2}{2 - \varepsilon_{DLL}} \tag{2.4}$$

with the so called *normalized loop bandwidth* ε_{DLL} defined as [23]:

$$\varepsilon_{DLL} \equiv \frac{I_{CP} K_d}{C_f} \approx \omega_c T_{ref} \tag{2.5}$$

The approximation shows the relation between the value of ε_{DLL} and the DLL loop bandwidth ω_c [38].

Note that (2.4) is in agreement with the result achieved in [23].

It is important to note that the jitter is lowest for low values of the DLL normalized loop bandwidth ε_{DLL}, in which case the jitter would be equal to that of a VCDL that is not controlled by a loop. This shows that the function of the control loop is not to remove jitter from the VCDL but merely to tune the total delay of the VCDL to the desired value. This conclusion was confirmed by measurements in [39].

Another observation from (2.4) is that the DLL can indeed become unstable, as mentioned before, contrary to common belief. This happens for $\varepsilon_{DLL} \geq 2$.

Apart from the jitter that is generated by the VCDL, the loop components that take care of the feedback mechanism also introduce jitter. First, the PFD that has to detect zero-crossings is realized using noisy elements. The internal noise of the PFD can be calculated back to the input as a voltage noise, which influences the moment in time that the PFD generates its

output signals and thus the charge that is integrated on the loop capacitor, assuming that the incoming edges are not infinitely steep. Also, the CP generates jitter as the charge that is pumped into the loop capacitor is noisy, because the switched current sources inside the CP are noisy in a realistic implementation. Both building blocks thus cause noise on the VCDL control voltage, resulting in output jitter.

To simplify calculations, the CP noise is calculated back to the input of the PFD as an equivalent time error:

$$\Delta t'_{PFD}(n) \equiv \Delta t_{PFD}(n) + \frac{q_{noise}(n)}{I_{CP}} \tag{2.6}$$

Using a method similar to the calculation of the jitter due to VCDL noise as described in Appendix A, the variance of the output jitter due to the PFD and CP noise can be calculated. This results in:

$$\sigma^2_{\Delta t_M} = \sigma^2_{\Delta t'_{PFD}} \frac{\varepsilon_{DLL}}{2 - \varepsilon_{DLL}} \tag{2.7}$$

Finally, there is the jitter contribution due to the reference signal itself being polluted by jitter. Applying the same method to analyze the jitter at the DLL output resulting from this jitter source yields:

$$\sigma^2_{\Delta t_m} = \sigma^2_{\Delta t_{ref}} \cdot \left\{ 1 + \frac{4\,\varepsilon_{DLL}}{2 - \varepsilon_{DLL}} \right\} \tag{2.8}$$

showing that a DLL can never decrease the jitter of the input reference, as is possible when using a PLL, because the jitter that is at the input of the VCDL will also be at the output of the taps. In fact, the deviations in the control voltage of the VCDL that are caused by the reference jitter will even increase the DLL output jitter.

From these equations it is again apparent that a small value of ε_{DLL} is beneficial for the DLL output jitter, for all sources of jitter mentioned here. The gain of the VCDL K_d should however be large enough to compensate for process spread and temperature variations; the Charge Pump current can not be chosen too small because of the jitter resulting from mismatch in the CP. This means that the loop filter capacitor should be made large at the cost of area, in order to maintain a reasonably low loop bandwidth. Other practical issues such as settling behavior may also set a lower limit on the value of the loop bandwidth.

2.3 Analysis of PLL jitter due to noise

In this section, an analysis is presented, similar to that of DLL jitter, which applies to an integer-*N* PLL-based clock multiplier. The analysis starts by deriving difference equations describing the architecture, which are then used to calculate the PLL output jitter due to different noise sources in the time domain directly.

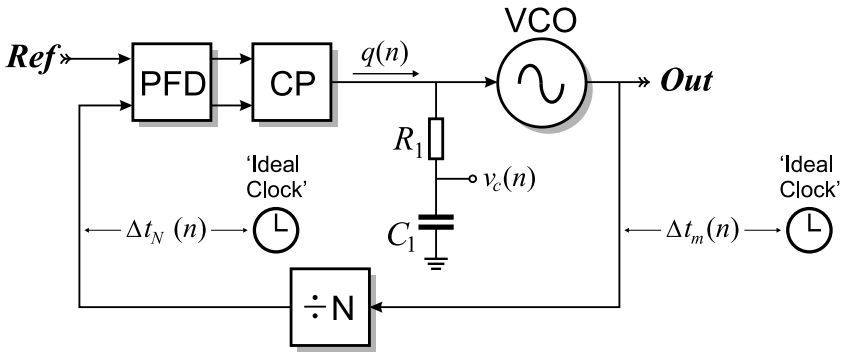


Figure 2.3: The PLL architecture.

2.3.1 Mathematical model of the PLL with Noisy Building Blocks

The difference equations describing the behavior of the PLL mathematically are derived using the PLL model shown in Figure 2.3. Again, the 'ideal clocks' are merely shown to show the jitter concept used here.

The PLL noise analysis depends on a number of assumptions similar to those made for the DLL:

1. The PLL is in lock. This implies that the loop is stable.
2. The mean VCO control voltage in lock equals 0 Volts. This means that the free-running frequency of the VCO is exactly equal to N times the reference frequency. The results of the jitter calculations do not depend on this assumption because of the linearity of the system.
3. The current the CP delivers can be modelled by charge pulses with a dirac-pulse shape.
4. All noise sources are white. From conventional PLL noise analysis one can conclude that this assumption is reasonable for a wide-band PLL. See also the remarks under Assumption 4 of section 2.2.1.
5. All noise sources are uncorrelated to other noise sources in the loop.
6. The loop behavior is linear, meaning that the superposition principle holds.

The variable n shown in the PLL model denotes the period number of the reference clock and is used in the difference equations that are to be derived.

As soon as the loop is in lock, the Charge Pump (CP) delivers current to the loop filter only just before and after a rising edge of the reference input signal, making the PLL behave much like a sampled system. To be able to model the behavior of this system, it is important to know the response of the loop filter and the VCO to a charge pulse from the CP. As stated in

Assumption 3, this charge pulse is modelled by a Dirac current pulse, which is reasonable in most cases as the actual duration of this charge pulse is much shorter than one VCO period in practice [40, 41].

The angular frequency of the VCO is controlled by the VCO's control voltage v_{tune} such that:

$$\omega_{VCO}(t) = \omega_{fr} + K_{VCO} v_{tune}(t) \tag{2.9}$$

where ω_{fr} is the free-running angular frequency of the VCO and K_{VCO} the VCO gain. The results of the jitter calculations do not depend on the value of the free-running frequency, due to the linearity of the loop.

In Appendix B, the VCO timing errors as a result of the CP current pulses are derived. Using this derivation, together with the PLL model of Figure 2.3, we can construct the following set of difference equations describing the discrete PLL behavior:

$$q_{LF}(n) = I_{CP} \cdot \left\{ \Delta t_N(n) + \Delta t_{PFD}(n) + \Delta t_{div}(n) - \Delta t_{ref}(n) \right\} + q_{noise}(n) \tag{2.10a}$$

$$v_c(n) = v_c(n-1) + \frac{q_{LF}(n-1)}{C_1} \tag{2.10b}$$

$$\Delta t_m(n) = \Delta t_N(n-1) - v_c(n) \frac{m K_{VCO} T_{ref}^2}{2\pi N^2} - \frac{K_{VCO} R_1 T_{ref}}{2\pi N} q_{LF}(n-1) + \sum_{l=1}^{m} \Delta T_{VCO,l}(n) \tag{2.10c}$$

In these equations, I_{CP} denotes the Charge Pump current, the jitter introduced by the frequency divider is denoted by $\Delta t_{div}(n)$, $\Delta t_{ref}(n)$ is the deviation of the reference input compared to an ideal clock with a period time of T_{ref}, $q_{noise}(n)$ is the charge noise of the CP and $\Delta T_{VCO,l}(n)$ is the period error of the VCO in its l-th cycle within a reference period, both due to internal noise of the VCO and the voltage noise on the control line of the VCO generated by the resistor of the loop filter.

The jitter variance Δt_m will be highest for the edge that causes a rising edge at the output of the divider, indicated by Δt_N. This is because that edge is used by the loop to correct the VCO, so the timing error of the very next edge will be less. The edges following will again be more and more polluted by jitter as the loop is 'dead' until the next comparison action. Thus, in the following analysis, only the jitter of this particular edge is calculated, although it is very well possible to calculate the jitter of all the intermediate edges, using the approach described in Appendix A.

2.3.2 PLL output jitter due to noise

In this section, the jitter caused by random VCO period variations is analyzed first using the set of difference equations given by (2.10). All other sources of jitter are assumed to be zero in this analysis. The effect of the other sources of jitter in a PLL are then discussed briefly.

Using a procedure similar to the example in Appendix A leads to the following value of the

jitter variance of the PLL output signal due to VCO jitter:

$$\sigma^2_{\Delta t_N} = E\left(\Delta t_N^2\right) = \sigma^2_{\Delta T_{VCO}} \frac{N}{\varepsilon_{PLL}\left\{2 - \varepsilon_{PLL}\left(1 + \frac{T_{ref}}{2R_1C_1}\right)\right\}} \tag{2.11}$$

where $\sigma^2_{\Delta T_{VCO}}$ symbolizes the variance of the VCO period jitter, as it would occur for a free-running VCO.

Again, ε_{PLL} denotes the *normalized loop bandwidth*. This quantity is a design variable that is defined in the case of a PLL as [23]:

$$\varepsilon_{PLL} \equiv \frac{I_{CP}K_{VCO}R_1T_{ref}}{2\pi N} \approx \omega_c T_{ref} \tag{2.12}$$

where ω_c now denotes the PLL bandwidth. Note that this definition is different from the one used for the DLL; in both cases, however, ε denotes the normalized loop bandwidth of the structure.

In practical PLL designs, the position of the loop filter zero is much smaller than the reference frequency. This means that $\frac{T_{ref}}{2R_1C_1}$ can be considered to be negligible to 1, reducing (2.11) to:

$$\sigma^2_{\Delta t_N} = E\left(\Delta t_N^2\right) = \sigma^2_{\Delta T_{VCO}} \frac{N}{\varepsilon_{PLL}\left(2 - \varepsilon_{PLL}\right)} \tag{2.13}$$

which agrees with [23], where the same assumption was used.

It is interesting to see that for $\varepsilon_{PLL}<1$ the maximum output jitter of a PLL is smaller with a large normalized loop bandwidth ε_{PLL} (provided that the jitter is most dominantly due to internal VCO noise). This observation corresponds with the well-known fact that VCO noise can be cleaned up with a wide-band PLL.

Note that the VCO noise is not the only source of output jitter. The internal noise of the building blocks other than the VCO will cause variations on the VCO tuning voltage, and thus output jitter. To ease calculations, the noise of the other PLL building blocks is calculated back to the input of the PFD according to:

$$\Delta t_{synth}(n) \equiv \Delta t_{PFD}(n) + \Delta t_{div}(n) + \frac{q_{noise}(n)}{I_{CP}} \tag{2.14}$$

the variance of which is referred to as $\sigma^2_{\Delta t_{synth}}$.

The PLL jitter due to these noise sources can now be shown to be (setting the other noise sources to zero):

$$\sigma^2_{\Delta t_N} = \sigma^2_{\Delta t_{synth}} \frac{\varepsilon_{PLL} + \frac{T_{ref}}{2R_1C_1}\left(2 + \varepsilon_{PLL}\right)}{2 - \varepsilon_{PLL}\left(1 + \frac{T_{ref}}{2R_1C_1}\right)} \tag{2.15}$$

Very similarly, the jitter on the reference signal will cause jitter on the PLL output signal, according to:

$$\sigma^2_{\Delta t_N} = \sigma^2_{\Delta t_{ref}} \frac{\varepsilon_{PLL} + \frac{T_{ref}}{2R_1C_1}\left(2 + \varepsilon_{PLL}\right)}{2 - \varepsilon_{PLL}\left(1 + \frac{T_{ref}}{2R_1C_1}\right)} \tag{2.16}$$

Observing these equations leads to the conclusion that, contrary to the VCO induced jitter, a large value of ε_{PLL} (corresponding to a large PLL bandwidth) will raise the PLL output jitter due to the noise of the other loop components.

Finally, the loop filter resistor will cause thermal noise at the input of the VCO, which is measurable at the PLL output as jitter. Using the fact that the thermal noise of the resistor is integrated by the VCO during every VCO period (which lasts approximately $\frac{T_{ref}}{N}$), the variance of the VCO period deviation caused by this thermal noise can be shown to be, using the theory of Appendix C:

$$\sigma^2_{\Delta T_{VCO}} = kTR_1 \frac{K^2_{VCO} T^3_{ref}}{2\pi^2 N^3} \tag{2.17}$$

where k is the Boltzmann constant and T the absolute temperature.

Substituting this in (2.13) yields:

$$\sigma^2_{\Delta t_N} \approx kT \frac{K_{VCO} T^2_{ref}}{\pi N \cdot I_{CP}} \frac{1}{(2 - \varepsilon_{PLL})} \approx kT \frac{K_{VCO} T^2_{ref}}{2\pi N \cdot I_{CP}} \tag{2.18}$$

where the last approximation holds for small values of the normalized PLL loop bandwidth.

2.3.3 PLL optimization

As was shown before, a larger value of ε_{PLL} will *lower* the output jitter due to VCO phase noise while *raising* the jitter contribution of the other synthesizer noise sources. It is thus to be expected that there will be an optimum value for ε_{PLL}. To be able to compare the DLL jitter characteristics with those of the PLL, the PLL should first be optimized.

To simplify things, we assume that ε_{PLL} is much smaller than 2 and that $\frac{T_{ref}}{2R_1C_1}$ is negligible to 1. Then the total PLL output jitter can be approximated by combining (2.13) and (2.15):

$$\sigma^2_{\Delta t_N} \approx \sigma^2_{\Delta T_{VCO}} \frac{N}{2\,\varepsilon_{PLL}} + \sigma^2_{\Delta t_{synth}} \frac{\varepsilon_{PLL}}{2} \tag{2.19}$$

The smallest amount of jitter is found for:

$$\varepsilon_{PLL,\text{opt}} = \frac{\sqrt{N \cdot \sigma^2_{\Delta T_{VCO}}}}{\sqrt{\sigma^2_{\Delta t_{synth}}}} \tag{2.20}$$

for which the total jitter can be approximated by:

$$\sigma^2_{\Delta t_N} = \sqrt{N \cdot \sigma^2_{\Delta T_{VCO}} \cdot \sigma^2_{\Delta t_{synth}}} \tag{2.21}$$

It is important to note that if the PLL bandwidth equals $\varepsilon_{PLL,\text{opt}}$ the jitter due to the VCO equals the jitter that is caused by the other loop components.

2.4 Comparing DLL and PLL jitter due to noise

In practical PLL-based clock multipliers, the VCO is often realized by a ring-oscillator as opposed to an oscillator using an *LC*-tank for frequency stability. An important reason for this is the area consumption of the on-chip inductor. For oscillation frequencies lower than roughly 2 GHz, realizing an on-chip inductor is especially troublesome. But also portability to newer processes and oscillator pulling effects are arguments against an *LC*-oscillator. An important disadvantage of a ring-oscillator is the relatively high jitter it produces, which is to be cleaned up by using a wide-band PLL [42, 43]. The maximum bandwidth of a PLL is in practice limited by stability considerations to about one tenth of the reference frequency that is used at the input of the PFD [40, 42]. Expressed in terms of the normalized loop bandwidth ε_{PLL}, this leads to:

$$\varepsilon_{PLL,\max} \approx \frac{2\pi f_{ref}}{10} T_{ref} = \frac{1}{5}\pi \approx 0.63 \tag{2.22}$$

Because of better supply noise and substrate bounce rejection, differential delay cells are often used in the ring-oscillator of the PLL. To compare the output jitter of an integer-*N* PLL to the DLL-based architecture, we assume that both the VCDL and the VCO consist of delay cells of similar topology: each delay cell consists of an NMOS differential pair with resistive load. Later, the results will be briefly discussed for other implementations.

2.4.1 Delay cell jitter

The jitter of a single delay cell can be predicted using the analysis presented in [44]. An important result from this work is:

$$\frac{\sigma_{\Delta_d}}{t_d} = \sqrt{\frac{kT}{C_L}} \frac{\xi}{V_{GS} - V_T} \tag{2.23}$$

in which $\sigma_{\Delta t_d}$ is the RMS-jitter of the cell, t_d is the delay of the cell, C_L is the load capacitance of one delay cell, ξ is a factor determined by the design and $V_{GS} - V_T$ the overdrive voltage of the NMOS differential pair transistors.

Knowing that the delay of one cell can be written as [44]:

$$t_d = V_{sw} \frac{C_L}{I_S}, \tag{2.24}$$

with V_{sw} the voltage swing of the delay cell and I_S the static current it consumes, we can rewrite (2.23) as:

$$\sigma^2_{\Delta t_d} = \left\{ \frac{\xi^2 kT V_{sw}}{(V_{GS} - V_T)^2} \right\} \frac{t_d}{I_S} = c \frac{t_d}{I_S} \tag{2.25}$$

with c a design dependent constant with unit $[A \cdot s]$ representing the bracketed part.

2.4.2 PLL jitter

Using equation (2.25), it is easy to show that the period jitter of a ring oscillator constructed using these delay cells is:

$$\sigma^2_{\Delta T_{VCO}} = cV_{DD}M_{VCO}\frac{T_{VCO}}{P_{static,VCO}} = cV_{DD}\frac{M_{VCO}}{N}\frac{T_{ref}}{P_{static,VCO}} \tag{2.26}$$

where V_{DD} is the supply voltage of the oscillator, M_{VCO} the number of delay cells used in the VCO, T_{VCO} the period time of the oscillator and $P_{static,VCO}$ the static power used in the VCO.

For simplicity we first assume that the VCO is the most dominant source of jitter in the PLL (the other jitter sources will be included later in the comparison for completeness). Then, using (2.13) we can write:

$$\sigma^2_{\Delta t_N} = c \cdot M_{VCO}T_{ref}\frac{V_{DD}}{P_{static,VCO}}\frac{N}{\varepsilon_{PLL}(2-\varepsilon_{PLL})} \tag{2.27}$$

2.4.3 DLL jitter

The jitter of a DLL used to multiply the reference by the same factor N can be estimated by (2.4), which reduces to:

$$\sigma^2_{\Delta t_M} = \sigma^2_{\Delta d_l} \cdot M_{VCDL} \tag{2.28}$$

for small values of the normalized DLL loop bandwidth.

Again, the jitter per delay cell can be predicted using (2.25), yielding:

$$\sigma^2_{\Delta t_M} = cM_{VCDL}\frac{t_d}{I_S} = 2c\,NT_{ref}\frac{V_{DD}}{P_{static,VCDL}} \tag{2.29}$$

2.4.4 Comparison and Discussion

Now if we allow an equal power usage in both the VCO and the VCDL, comparing (2.27) to (2.29) yields:

$$N > \frac{M_{VCO}}{2\varepsilon_{PLL}(2-\varepsilon_{PLL})} \Rightarrow \left(\sigma^2_{\Delta t}\right)_{DLL} > \left(\sigma^2_{\Delta t}\right)_{PLL} \tag{2.30}$$

If we assume a VCO consisting of 3 delay cells and a PLL with a normalized loop bandwidth given by (2.22) this leads to the conclusion that if the frequency multiplication factor N is higher than about 1.74, the DLL output jitter will be higher than the PLL output jitter. Because N is in practice an integer number, we can draw the conclusion that under the assumptions given in this section a PLL-based clock multiplier yields less output jitter than a DLL-based clock multiplier. This is because spending the same amount of power in the VCO as in the VCDL yields more power in the VCO *per delay cell* and thus less jitter per cell. This effect is larger than the jitter accumulation factor discussed in [23] (and expressed in (2.13) by the term $1/\{\varepsilon_{PLL}(2-\varepsilon_{PLL})\}$), which is not much larger than 1 for a wide-band PLL.

This important conclusion is based on a ring oscillator type VCO and a VCDL consisting of similar delay-cells. When the VCO has an improved quality factor Q, such as an LC-oscillator, this conclusion will even be stronger, in favor of the PLL-based CMU. This is because the total jitter will be smaller in that case due to the decreased phase noise of the oscillator.

Although it is possible to add a resonator network to the edge combiner of a DLL [26], this will never be as effective as the LC-tank in a VCO, because the latter is tuned to the oscillation frequency and will thus operate at its peak Q automatically. The LC-tank in a DLL will have a lower Q (a less sharp resonation peak) as it is not tuned to the correct frequency.

It is possible to get rid of the jitter accumulation in a PLL by periodically aligning the VCO with the reference signal, as shown in [45, 46]. This makes the loop behave more like a DLL in which the delay cells are re-used within one cycle of the reference clock, enabling more power usage *per cell*. This frequency multiplication technique does not need an edge combiner to increase the frequency. A disadvantage of this principle is that the injection of the reference clock should be timed very accurately, which might require calibration. This required timing accuracy might make the technique unsuitable for very high frequency clocks.

For completeness, an equation is derived that is valid for a PLL with additional jitter sources. The simplest way of doing this, is to realize that if the PLL bandwidth has been optimized with respect to jitter, the total output jitter is twice the jitter due to the VCO, as noted before in section 2.3.3. If we again assume that the VCO power consumption equals that of the VCDL of the DLL, equation (2.30) can be rewritten as:

$$N > \frac{M_{VCO}}{\varepsilon_{PLL,\mathrm{opt}}\left(2-\varepsilon_{PLL,\mathrm{opt}}\right)} \Rightarrow \left(\sigma^2_{\Delta t}\right)_{DLL} > \left(\sigma^2_{\Delta t}\right)_{PLL} \tag{2.31}$$

Using the results of the PLL optimization in this equation leads to the following conclusion:

$$N > \frac{1}{2}\sqrt{\frac{M_{VCO}P_{static}}{c\,T_{ref}V_{DD}}\,\sigma^2_{\Delta t_{synth}}} \Rightarrow \left(\sigma^2_{\Delta t}\right)_{DLL} > \left(\sigma^2_{\Delta t}\right)_{PLL} \tag{2.32}$$

where both the power used by the VCO and by the VCDL are equal to P_{static}. We can conclude that the more dominant the noise sources other than the VCO are in the PLL, the higher the frequency multiplication factor N is that is needed for the PLL to be superior to the DLL with respect to jitter. Note that reference jitter is not included in this equation; the PLL is always superior to the DLL with respect to jitter transfer.

We have assumed that the dominant power usage of the delay cells is static and that the jitter of the cells is mostly due to thermal noise. For practical implementations, these assumptions are often reasonable. However, if the delay cells consist of for example CMOS inverters, where power usage does not depend on delay line length and consequently not on the frequency multiplication factor, the DLL will perform somewhat better than the PLL, due to jitter accumulation. This also holds when the jitter is mostly caused by supply or substrate noise [3] as the jitter can not be lowered by raising the power then. In both cases, the dif-

ference is small however, as the accumulation factor of a wide-band PLL is not much larger than 1.

2.5 Jitter due to mismatch

Apart from noise sources in the CMU circuit, timing imperfections can also be caused by mismatches in the circuit [36]. An obvious cause of jitter would be the mismatch in the VCDL delay cells of the DLL-based CMU circuit [34]. Another CMU building block that deserves attention with respect to matching is the Charge Pump, as will be discussed in section 2.5.2.

Although mismatch is caused by a stochastic process, the jitter that originates from it is deterministic, because once the chip has been processed, the mismatch properties are more or less fixed. Knowing the stochastic properties of the mismatch, predictions can be made *a priori* about the deterministic jitter.

In section 2.5.1, the jitter variance due to delay cell mismatch is evaluated. Section 2.5.2 discusses the results of a mismatch in the current sources of the Charge Pump. In general, it will be made plausible that mismatch is usually not a severe problem in PLLs, while it may severely disturb the performance of DLL-based clock multipliers.

2.5.1 DLL output jitter due to delay cell mismatch

Because of stochastic component mismatch, the delay of different delay cells in the VCDL of a DLL will not be exactly equal for a certain tuning voltage, which will result in jitter as all the intermediate edges on the different output taps are not corrected by the loop. The amount of jitter caused by this effect is calculated here.

The delay mismatch can be described mathematically as follows:

$$d_i = \{1 + e_i(v_c)\}\, d_{tune} \tag{2.33}$$

where d_i is the particular delay of delay cell number i, d_{tune} is some nominal delay which is controlled by the VCDL tuning voltage v_c and $e_i(v_c)$ is a random variable, describing the delay cell mismatch for a certain value of v_c. For simplicity, this dependency on v_c will not be shown explicitly in the remaining equations. The variable e_i is assumed to have zero mean. This is reasonable as any common change of delay in the cells is removed by the loop. The delay mismatch of different cells is assumed to be uncorrelated.

The total delay of the VCDL will be equal to one period of the input clock after lock has been achieved. This results in the following equation for the individual delay of the delay cells:

$$d_i = T_{ref} \frac{1 + e_i}{M + \sum_{i=1}^{M} e_i} \tag{2.34}$$

where M denotes the number of delay cells in the VCDL and T_{ref} the period time of the reference signal.

Now an expression for the total systematic jitter of the signal on the m-th tap (at the output of the m-th delay cell) can be derived. If all the delay cells would be perfectly matched, the delay between the input and the m-th tap would be $\frac{m}{M}T_{ref}$. In case of mismatch, the systematic jitter after m cells can then be calculated to be:

$$\Delta t_m = \sum_{i=1}^{m} d_i - \frac{m}{M}T_{ref} = T_{ref}\left(\frac{m + \sum_{i=1}^{m} e_i}{M + \sum_{i=1}^{M} e_i} - \frac{m}{M}\right), \tag{2.35}$$

the variance of which can be shown to be:

$$\sigma^2_{\Delta t_m} = E\left\{(\Delta t_m)^2\right\} \approx T^2_{ref}\frac{m\,(M-m)}{M^3}\sigma^2_{e_i} \tag{2.36}$$

assuming uncorrelated values of e_i with zero mean. A first order Taylor expansion has been used, assuming $\sigma^2_{e_i}$<<1.

It is interesting to note that the variance of Δt_m is highest for m=$\frac{1}{2}M$, *i.e.* halfway the VCDL. This is to be expected: the loop controls the VCDL such that the time error at its output is zero, while the error at the input of the VCDL is also zero. The highest timing uncertainty will be in the middle of the VCDL, where the distance to these clean points is highest. This is comparable to mismatch in resistors in a resistor string based A/D converter, where the highest deviation is also found in the middle of the string [47].

The sigma value of the phase time error halfway the VCDL can be approximated, using (2.36), to be:

$$\sigma_{\Delta t_{\frac{1}{2}M}} \approx \sigma_{e_i} \cdot \frac{T_{ref}}{2\sqrt{M}} \tag{2.37}$$

Equation (2.36) has been verified using numerical statistical analysis for a constant value of the nominal delay of a single delay cell (50 ps), the results of which are shown in Figure 2.4. This figure shows a very good agreement between the predicted time deviations and the simulations. It also clearly shows the peak of the time deviation variance at the middle of the VCDL.

The jitter due to delay cell noise is also shown in the figure, for an arbitrary value of $\sigma_{\Delta t_d}$, the RMS-jitter of a single delay cell due to noise. Using the fact that DLL output jitter due to delay cell noise is approximately equal to the jitter of the *uncontrolled* VCDL [23] yields:

$$\sigma_{\Delta t_m} \approx \sqrt{m}\,\sigma_{\Delta t_d}, \tag{2.38}$$

showing that the effect of delay cell noise is highest on the *last* output tap, as opposed to mismatch induced jitter.

If we define a measure of relative jitter, where the sigma value of the maximum time deviation is related to the output period of the clock multiplier, the following result is obtained:

$$\frac{\sigma_{\Delta t_{\frac{1}{2}M}}}{\left(\frac{T_{ref}}{N}\right)} \approx \sigma_{e_i} \cdot \frac{\sqrt{N}}{2\sqrt{2}} \tag{2.39}$$

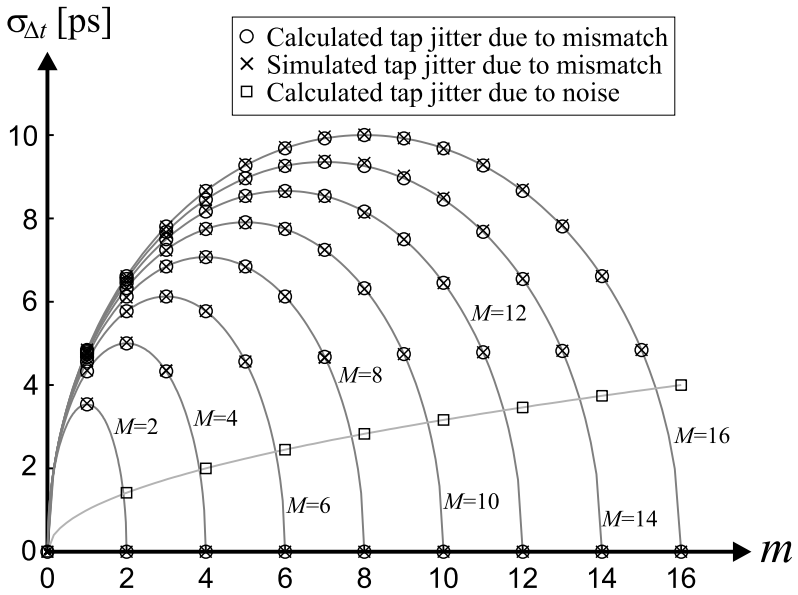


Figure 2.4: Numerical statistical simulation results of the DLL jitter due to delay cell mismatch (σ_{e_i}=10%).

using (2.1), which shows that the relative jitter of the output signal is proportional to the square root of the frequency multiplication factor N. This dependency on N was also shown for RMS-jitter due to delay cell noise.

2.5.2 DLL jitter due to CP mismatch

If the up and down current sources of the Charge Pump are not exactly matched, it will have to be compensated by a phase difference at the input of the clock multiplier, as will be explained later. Especially in the case a DLL is used, this phase difference will lead to jitter at the CMU output. When a PLL is used, this is in most cases a second order effect, and will only be a problem when the output *spectrum* of the PLL is important, like in radio reception or transmission [21].

To understand the phenomenon qualitatively, one has to realize first that the Charge Pump together with the loop filter capacitor forms an integrator. In the locked situation, the average current from the CP into the loop filter will have to be zero, disregarding leakage effects. In a practical CP, both current sources conduct current to the loop filter for a certain minimum amount of time, in order to prevent dead-zone [29]. If there is a mismatch in the currents of both CP current sources, this will have to be compensated by the smaller current source being 'on' longer than the other, such that the total charge pumped into the loop filter is zero. Thus, current mismatch will be translated into phase error at the PFD input.

It is important to understand that this CP current mismatch does not only originate from device mismatch (such as transistor threshold or area mismatch). Because the output impedances of both current sources will be finite in practice, the tuning voltage of either the VCDL or the VCO will influence the values of both current sources. A high tuning voltage will in general decrease the *UP* current while increasing the *DN* current[2]. So, matching of the two currents for a certain value of the tuning voltage generally means a mismatch at a different tuning voltage.

Using a very simple 'digital' model for the Charge Pump, where the current sources are either completely 'on' or completely 'off', depending on the *UP* and *DN* pulses, it is straightforward to calculate the input phase error resulting from current source mismatch. The current mismatch is described as:

$$I_{UP} = (1 + e_{CP})\, I_{DN} \tag{2.40}$$

where I_{UP} and I_{DN} are the current source values and e_{CP} a random variable indicating the mismatch, either due to device mismatch or output impedance.

Because of the necessary charge balance, the following holds:

$$I_{UP} \cdot t_{UP} = I_{DN} \cdot t_{DN} \tag{2.41}$$

where t_{UP} and t_{DN} refer to the 'on'-time of the current sources per input period. Using (2.40) yields:

$$\frac{t_{DN}}{t_{UP}} = 1 + e_{CP} \tag{2.42}$$

Now we assume that both the *UP* and *DN* pulse at the CP input are high at least an amount of time t_{ovl}. This is the overlap time of the *UP* and *DN* pulses (using a PFD, this is the time needed to reset the internal PFD flip-flops [29] after both have been set by their input signals). We can distinguish two cases:

$$I_{UP} > I_{DN} \Rightarrow t_{UP} = t_{ovl};\ t_{DN} = t_{ovl}\,(1 + e_{CP}) \tag{2.43a}$$

$$I_{DN} > I_{UP} \Rightarrow t_{DN} = t_{ovl};\ t_{UP} = \frac{t_{ovl}}{1 + e_{CP}} \tag{2.43b}$$

Both cases reduce to:

$$t_{UP} - t_{DN} \approx -e_{CP} \cdot t_{ovl} \tag{2.44}$$

using a first order Taylor approximation under the assumption that the mismatch is small.

Finally, if we realize that both the *UP* and the *DN* signal are falling simultaneously, the Charge Pump current mismatch can be translated into a edge-alignment error at the PFD input (the positive zero-crossing of the VCDL or frequency divider output is misaligned with respect to the corresponding reference zero-crossing):

$$\Delta t_{PFD} \approx -e_{CP} \cdot t_{ovl} \tag{2.45}$$

[2]Note that this statement assumes a positive K_{VCO}, such that the *UP* source pumps charge *into* the loop filter and the *DN* source sinks charge from the filter. For a negative K_{VCO}, a similar statement holds.

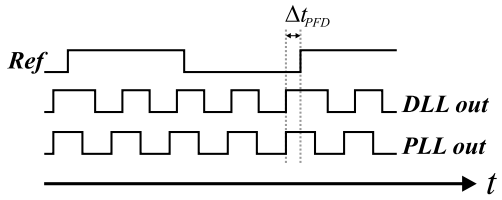


Figure 2.5: The effect of CP mismatch on DLL and PLL output.

Figure 2.5 shows the consequence of the current mismatch on the output clocks of both the DLL- and the PLL-based Clock Multiplier. In this figure, the delay of the Edge Combiner and that of the frequency divider are neglected for simplicity. As is obvious from this figure, the resulting output clocks are very different. In the case of the PLL, the output clock is merely shifted in phase with respect to the reference clock. The CP mismatch will cause a slight disturbance on the VCO control voltage, but the effect on the output jitter is a second order effect as compared to the effect the CP mismatch has on the DLL output signal.

The VCO control voltage can be filtered by a capacitor across the loop filter, as is done in practice. For the DLL, no matter how big the loop filter capacitor, the CP mismatch effect can not be filtered, because the disturbance on the loop filter capacitor is not the main reason for the output jitter, the DC voltage is. To compensate for the mismatch, the total delay of the line will differ from the reference period time, the ideal value. This causes one period of the output clock to have a different period time from the others in the cycle. The effect is a high-frequency disturbance of the clock, possibly causing bit errors at the receive side if used *e.g.* in an optical transmitter.

Note that it is exactly the ‘resetting’ of the VCDL in the DLL, that is often assumed to be beneficial for the DLL output jitter, that causes this problem. To minimize this problem, care should be taken in the design of the Charge Pump (see *e.g.* [48]), but as is apparent from (2.45), one should also try to minimize the reset-time of the PFD without introducing the hazard of dead-zone.

2.6 Impedance Level Scaling

It is a well-known fact that increasing the area of on-chip MOS-transistors improves the matching properties of those transistors [49]. The same also goes for the matching of resistors and capacitors on an IC [50]. This leads us to investigate the effect of increasing the area of a complete circuit in a systematic manner that we call *Impedance Level Scaling*, also known as *W-scaling* [37].

The concept of Impedance Level Scaling is fairly simple, yet leads to very useful design considerations. This technique enables a decoupled optimization of the noise and mismatch properties of a circuit independent of other properties such as speed and linearity, thus simplifying the task of the designer.

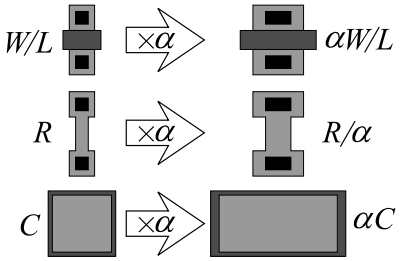


Figure 2.6: Concept of Impedance Level Scaling.

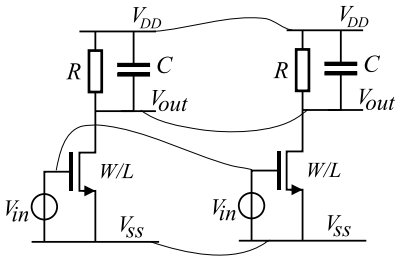


Figure 2.7: Impedance Level Scaling presented as putting identical circuits in parallel.

Starting from a circuit that has been optimized with respect to specifications other than noise and mismatch, one can scale the *width* of every component of that circuit by a certain factor α (hence, the term W-scaling). This is shown conceptually in Figure 2.6, where the effect on the component values is also shown.

Using the analogy that scaling is similar to putting identical circuits in parallel, as illustrated in Figure 2.7 where $\alpha = 2$, it is easy to deduce that the node voltages of the scaled circuit are equal to those of the original circuit, provided the circuit is not heavily loaded externally. From this analogy it is also clear that the scaling will not change linearity and speed of the circuit.

A fact that is familiar to many designers is that Impedance Level Scaling will improve the Signal to Noise ratio of the circuit at the cost of increased power usage. More precisely, scaling the circuit by a factor α will decrease the RMS-value of the noise voltages by a factor $\sqrt{\alpha}$ while increasing the power usage by a factor α, meaning there is a direct trade-off between power usage and noise.

A less familiar but important property of Impedance Level Scaling is the effect it has on the mismatch errors of a circuit. Assume the relative change in the value of a certain component changes some circuit parameter (*e.g.* the offset voltage, or the delay of a delay cell) *linearly*. This is reasonable as long as mismatch changes the value of a component just slightly. The same *relative* change of the corresponding component in the scaled circuit will result in the same change of the output parameter, which can again be understood by the scaling analogy

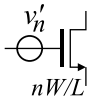
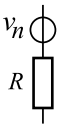
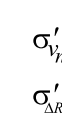
Starting circuit	After Impedance Level Scaling
 $\sigma_{v_n} = \sqrt{\frac{4kT\gamma}{g_m}\Delta f}$ $\sigma_{\Delta\beta/\beta} = \frac{A_\beta}{\sqrt{W \cdot L}}$ $\sigma_{\Delta V_T} = \frac{A_{V_T}}{\sqrt{W \cdot L}}$	$g'_m = g_m \cdot \alpha$  $\sigma'_{v_n} = \sigma_{v_n} \cdot \frac{1}{\sqrt{\alpha}}$ $\sigma'_{\Delta\beta/\beta} = \sigma_{\Delta\beta/\beta} \cdot \frac{1}{\sqrt{\alpha}}$ $\sigma'_{\Delta V_T} = \sigma_{\Delta V_T} \cdot \frac{1}{\sqrt{\alpha}}$
 $\sigma_{v_n} = \sqrt{4kTR\Delta f}$ $\sigma_{\Delta R/R} = \frac{A_R}{\sqrt{\text{Area}}}$	 $\sigma'_{v_n} = \sigma_{v_n} \cdot \frac{1}{\sqrt{\alpha}}$ $\sigma'_{\Delta R/R} = \sigma_{\Delta R/R} \cdot \frac{1}{\sqrt{\alpha}}$
 $\sigma_{\Delta C/C} = \frac{A_C}{\sqrt{\text{Area}}}$	 $\sigma'_{\Delta C/C} = \sigma_{\Delta C/C} \cdot \frac{1}{\sqrt{\alpha}}$

Table 2.1: The effect of Impedance Level Scaling on Component Properties.

depicted in Figure 2.7. But the mismatch of the component value of the scaled circuit will reduce by a factor $\sqrt{\alpha}$ (see Table 2.1), which means the sensitivity of circuit parameters such as offset and delay errors will be $\sqrt{\alpha}$ times less in the scaled circuit than in the starting circuit, at the cost of increased power usage.

Summarizing these results gives:

1. Impedance Level Scaling does not alter node voltages and frequency dependency (or speed) of the circuit.
2. There is a direct trade-off between power usage and noise errors.
3. There is a direct trade-off between power usage and mismatch errors.

For a delay cell, the implication of the Impedance Level Scaling is that increasing the power by a factor α yields a jitter reduction of $\sqrt{\alpha}$, when jitter due to noise is concerned (which also follows from the jitter analysis in [44]). Also the mismatch of the delay between different cells will improve by a factor $\sqrt{\alpha}$, and with that, the jitter caused by this mismatch.

2.7 Simulation results

In this section, results of high-level DLL and PLL simulations are presented first. These simulations were performed to verify the equations that were derived for the output jitter due

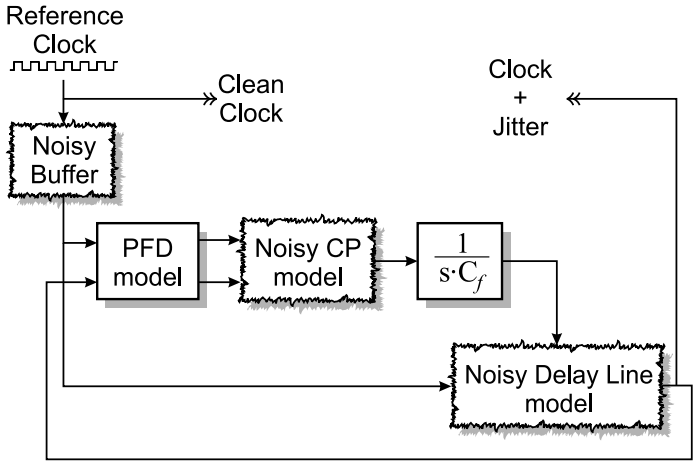


Figure 2.8: The simulation model for the DLL.

to various noise sources. Then, results of Monte Carlo simulations of a Delay Line are shown. These were done to verify the predictions done about Impedance Level Scaling and to give an indication of the severity of the mismatch induced jitter compared to jitter due to thermal noise.

2.7.1 Noise simulations

To verify the noise calculations and jitter predictions that are described in the previous section, high-level simulation models of a DLL and a PLL have been used in Simulink (which is a MATLAB[3] simulation shell). These models are depicted in Figure 2.8 and Figure 2.9. Although these simulations were time consuming, enhancement of simulation speed using techniques such as described in [51] were not used, as these techniques do not apply to systems with additive noise.

The most important noise sources used in the analyses can be applied independently. The delay cell noise is modelled by random uncorrelated delay variations with zero mean. The Charge Pump noise is modelled by adding white noise to the Charge Pump current sources. The variance of the charge that is pumped into the filter is then roughly proportional to the PFD reset time (this is the overlap time of the up- and down-current sources that is present in realistic PFD designs [29]). The reference buffer that is used is comparable to the delay cells used in the Delay Line, *i.e.* it adds jitter to the reference signal that is uncorrelated from period to period.

To evaluate the simulated jitter, the clean positive zero crossings of the reference generator (before polluting it with jitter by the reference buffer) are compared with those of the DLL

[3]MATLAB is a registered trademark of The MathWorks, Natick, MA.

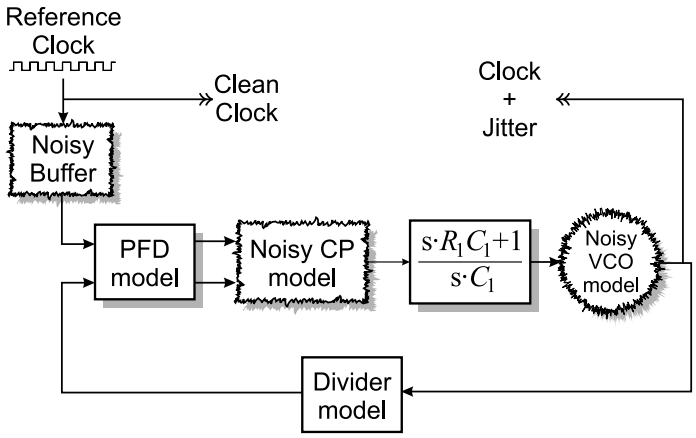


Figure 2.9: The simulation model for the PLL.

and PLL output signals. The jitter is then calculated as the variance of the time differences.

The graphs shown in Figure 2.10 and Figure 2.11 show simulation results for a clock multiplication factor N of 8, meaning that the VCDL consists of 16 delay cells. The VCO consists of 3 delay cells. The jitter of the VCO delay cells was related to that of the VCDL delay cells according to (2.26) and (2.29).

First, simulations were done with only one noise source turned on with the variances of the other sources set to zero. The graphs show good agreement between the predicted and the simulated points. Then, all noise sources were turned on simultaneously to prove that the superposition principle, that was used as an important assumption throughout the analysis, was valid (meaning the jitter contribution of the different noise sources could be added power-wise). The result of these simulations is also shown, again showing good agreement with expectations. The deviations at low normalized bandwidths are caused by the fact that the simulation time was short compared to the settling time at those bandwidths.

The simulation results give confidence in predictions of DLL and PLL output jitter based on the equations derived in this chapter. They show that the PLL shows an optimum for the normalized loop bandwidth, while the DLL benefits from having a low loop bandwidth, and they confirm the prediction that the PLL would have lower output jitter in its optimum than the DLL clock multiplier (in this case the total optimized PLL RMS-jitter is roughly half that of the DLL).

2.7.2 Mismatch simulations

Monte Carlo simulations have been performed in PStar (a Spice-like simulation tool) on a Delay Line in order to verify the effect of Impedance Level Scaling on the delay mismatch and to compare the jitter due to mismatch to the jitter caused by circuit noise. The delay

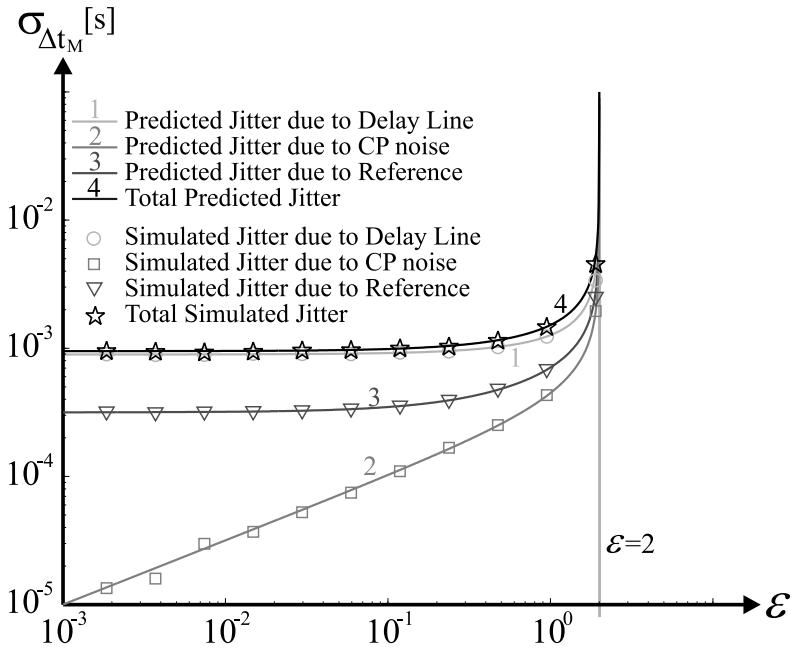


Figure 2.10: DLL simulation results: output jitter versus loop bandwidth normalized to reference frequency.

cells were "realized" as differential NMOS pairs with a resistive load, in a modern 0.18μm CMOS process. The delay of a single cell was about 50 ps; the differential voltage swing was 500 mV. The length of the delay cell transistors was minimal (0.18μm), because we are interested in high-speed DLLs.

The delay cell mismatch spread was simulated for various values of the scale factor α. The results of the simulations are presented in Figure 2.12, where the results are used in combination with (2.37) with M=16 and T_{ref}=800 ps. The upper solid line through these points has been calculated by applying the scaling theory on the simulation point at P=5.8 mW. The graph shows good agreement between theory and simulations.

Using results presented in [44], it is possible to estimate the jitter of one delay cell due to circuit noise. This has been done using operation point information obtained from simulations of the cells at P=5.8 mW. Using (2.38) leads to:

$$\sigma_{\Delta t_M} \approx \sqrt{M}\sigma_{\Delta t_d} \tag{2.46}$$

where $\sigma_{\Delta t_d}$ is the RMS jitter of a single delay cell as calculated in [44]. The calculated jitter due to noise is also shown in Figure 2.12, where the solid line represents the extrapolation of this calculation according to the scaling theory.

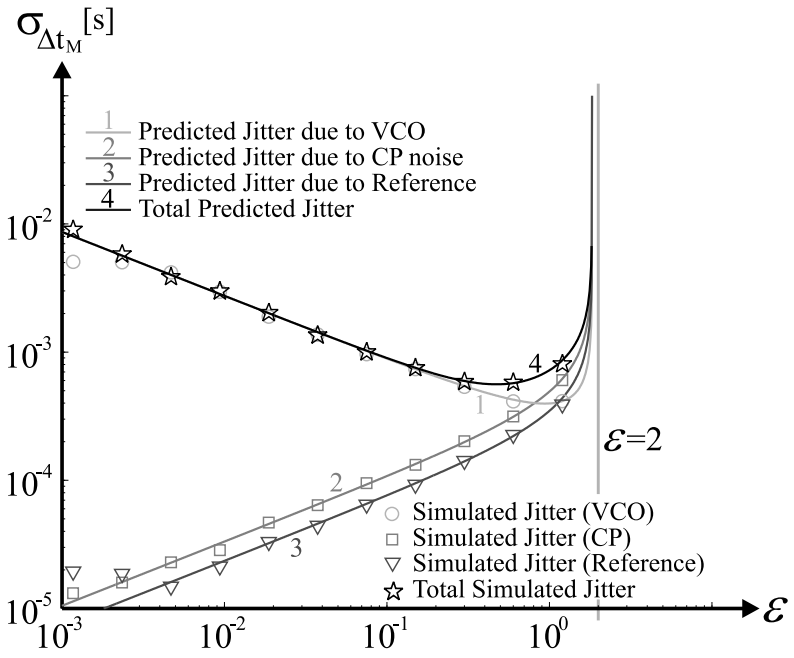


Figure 2.11: PLL simulation results: output jitter versus loop bandwidth normalized to reference frequency.

It is obvious from the graph that jitter due to mismatch is in this case[4] dominating the jitter behavior of the delay line. Another important observation is that increasing the power has the same effect on both the jitter due to noise and the jitter due to mismatch (increasing the power per delay cell with a factor α, decreases the jitter by a factor of $\sqrt{\alpha}$). Because higher power usage leads to lower total jitter, it is in theory possible to meet strict jitter specifications with a DLL-based architecture. This might however lead to unrealistic power usage of the structure.

2.8 Summary of Conclusions

Although a DLL-based clock multiplier at first glance seems a better choice than a PLL based architecture because of the jitter accumulation effects in the PLL, the fact that the structures should perform clock multiplication leads to a drastically different conclusion. In practical implementations of clock multipliers (based on either a DLL architecture or an integer-*N* PLL), the fact that the VCDL of the DLL needs more delay sections to perform the same task yields a lower power budget *per delay cell* for the VCDL than for the VCO and thus less jitter per delay cell. This effect is stronger than the jitter accumulation that the VCO of a PLL

[4] Area increase has been achieved by scaling the *width* of components. Naturally, scaling lengths will also increase matching and jitter, but we chose minimal lengths for high-speed operation.

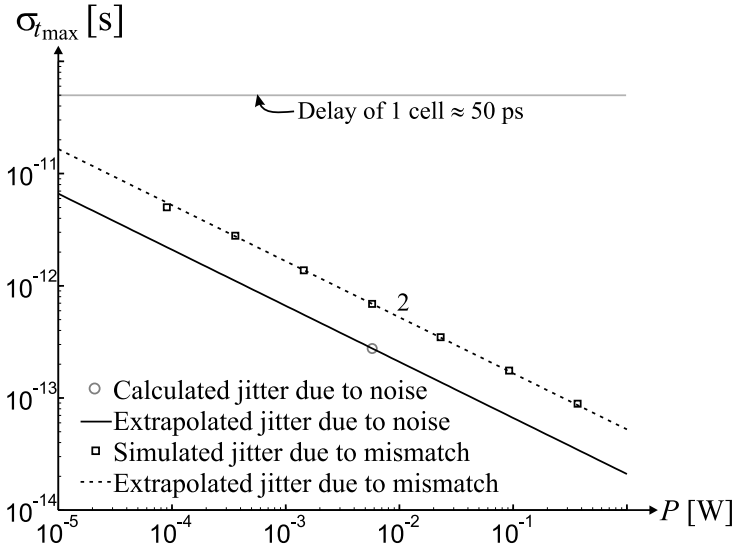


Figure 2.12: Relation between power per delay cell and DLL jitter, due to noise and mismatch; M=16, T_{ref}=800 ps.

suffers from, leading to the conclusion that a wide-band PLL used for clock multiplication produces less output jitter than a DLL-based implementation of the same function. This conclusion is based on a wide-band PLL that uses a differential ring oscillator built using delay elements similar to those used in the VCDL of the DLL. The conclusion will even be stronger in favor of the PLL if the VCO is realized using an *LC*-oscillator, due to its superior phase noise quality.

Another very important source of jitter should be taken into consideration for the DLL-based architecture: the stochastic mismatch of the delay cells in the VCDL. Monte Carlo simulations with a modern CMOS process indicate that this type of jitter is dominant in a DLL where intermediate clock phases of the VCDL are also used, due to the clock skew that is caused by the mismatch.

It has been shown, using the concept of Impedance Level Scaling, that there is a direct trade-off between power usage and output jitter of the frequency multiplier. This trade-off is identical for both jitter due to noise and due to mismatch. The amount of output jitter is limited directly by the power budget of the circuit. It was shown that if the delay cell mismatch is the most dominant jitter source for a certain circuit, it will still be dominant in an impedance level scaled version of this circuit.

The comparative analysis revealed essential differences between the PLL and the DLL. The DLL output jitter can be minimized by minimizing the DLL loop bandwidth. The function of the control loop is not to filter out jitter (as is the case for a PLL), but merely to tune the value of the mean delay of the VCDL to be equal to the reference period. For a very small

loop bandwidth, the DLL behaves as if uncontrolled with respect to jitter. In contrast, for an integer-*N* PLL, the PLL loop bandwidth shows a certain optimum, where the output jitter is minimized.

3

Low-Jitter PLL Design Issues

This chapter describes some basic PLL design considerations, mostly aimed at low-jitter PLL design. The analyses are performed in the frequency domain, allowing for a very simple jitter optimization method, including complex filter transfer and 1/f noise sources. The concept of optimal loop filter capacitor is introduced. The influence of the reference frequency on the PLL output phase noise and jitter is discussed.

3.1 Introduction

The main conclusion from the previous chapter that it is favorable to choose a PLL-based clock multiplier architecture with respect to jitter for a given power budget. This chapter describes some important design considerations for low-jitter PLL design.

First, the PLL output jitter due to various components, that was analyzed in the time-domain before, is examined in the frequency-domain, for reasons that are clarified in Section 3.2. Section 3.3 summarizes some important results obtained in [22] concerning PLL phase noise calculation due to various noise sources. Section 3.4 discusses PLL bandwidth optimization with respect to total output jitter. A common case is examined in section 3.4.1, where the PLL jitter is dominated by VCO phase noise and by Charge Pump noise, leading to the concept of optimal loop filter capacitor size. Finally, the influence of the PLL reference clock frequency on the CMU output jitter is examined in section 3.5.

3.2 Time-Continuous PLL Analysis

In the previous chapter, where DLL- and PLL based Clock Multipliers were compared, the output jitter of both architectures was analyzed in the time domain directly, thereby taking account of the time-discrete character of the phase detection process and the delay of the VCDL. Analyzing PLL jitter generation in this way has some disadvantages, however. The most important problems with time-domain analysis are:

- The time-discrete method makes $1/f$ noise sources very hard to handle. This is due to the large number of terms needed to accurately describe the $1/f$ noise auto-correlation function [52, 53].
- It is hardly possible to take into account the frequency integration limits usually specified for the CMU's jitter generation.
- Introducing a second loop filter capacitor, making a third order PLL, makes the time domain analysis a lot more complex.

The most obvious alternative method of analyzing PLL output jitter is by making a time-continuous approximation of the PLL in the phase domain [35]. Using this continuous model, it is possible to calculate the PLL phase noise power spectral density (PSD) due to various noise sources. Integration of this phase noise PSD gives the residual phase deviation $\sigma^2_{\phi_{res}}$

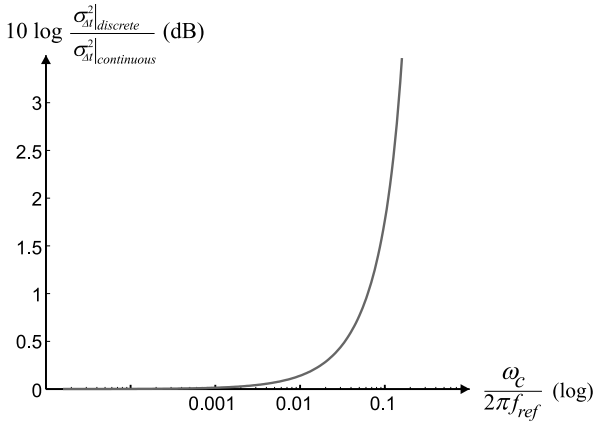


Figure 3.1: Discrete versus continuous analysis as a function of bandwidth.

[22], which is proportional to the PLL's rms-jitter squared, according to:

$$\sigma^2_{t_o} = \frac{\sigma^2_{\phi_{res}}}{4\pi^2 f^2_{VCO}} = \frac{1}{4\pi^2 f^2_{VCO}} \int_{f_l}^{f_h} S_{\phi_{no}}(f_m)\, d f_m \tag{3.1}$$

where $\sigma^2_{t_o}$ is the PLL rms-jitter, f_{VCO} the PLL output frequency and $S_{\phi_{no}}(f_m)$ the PSD of the PLL output phase noise as a function of offset frequency f_m. The integration range is between f_l and f_h.

This continuous analysis method does not have the disadvantages mentioned before. Because the analysis is performed in the frequency domain, the inclusion of $1/f$ noise is natural. As can be seen in (3.1), the specified integration limits are automatically accounted for. And finally, the extra loop filter capacitor is easily incorporated in the transfer functions of the various noise sources to the PLL output.

Of course, modelling the PLL with a time-continuous system raises the question whether doing so is valid; we in fact know that the phase-detection is usually edge-triggered, making the PLL behave like a sampled system. It can be shown, however, that if the PLL bandwidth is "small enough" with respect to the reference frequency, the time-continuous model describes the PLL behavior well. Figure 3.1 shows the results of PLL calculations using a time-continuous system compared to the results presented in the previous chapter, applied to the same system. For a realistic PLL bandwidth (rule of thumb limits the PLL bandwidth to one tenth of the reference frequency [40]), the continuous analysis method proves reliable.

3.2.1 The Time-Continuous PLL Model

Figure 3.2 shows a linear time-continuous PLL model with the various sources of noise [22], that can be used to analyze the PLL behavior in the frequency domain. The signal domains

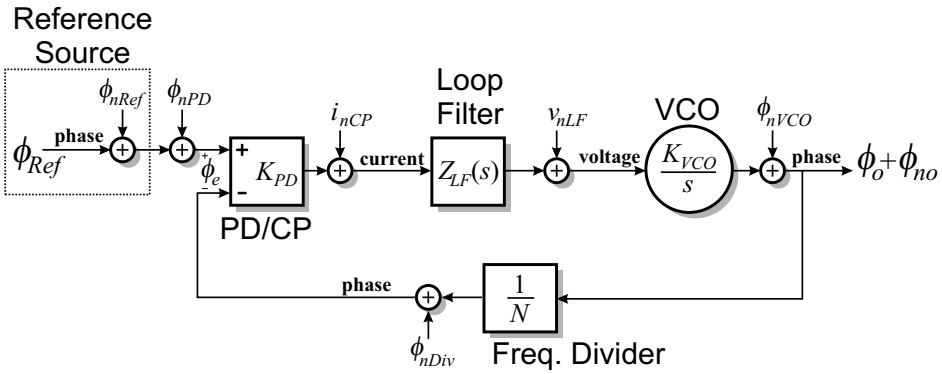


Figure 3.2: The time-continuous PLL model plus the noise sources.

(phase, current or voltage) of the various nodes in the model are also mentioned in the figure.

The model as shown in Figure 3.2 applies to a Charge Pump PLL [40]. The building block marked 'PD/CP' is the combination of the Phase Detector (PD) and the Charge Pump (CP). Its total gain is defined as:

$$K_{PD} \equiv \frac{d\overline{i_{cp}}}{d\phi_e} \tag{3.2}$$

where $\overline{i_{cp}}$ is the mean CP output current, averaged over one reference period time, and ϕ_e is the phase error at the input of the PD. The derivative is taken with respect to ϕ_e, and is evaluated for the static phase error at which the PLL will lock[1]. This is illustrated by Figure 3.3, where the well-known tri-state Phase Frequency Detector (PFD) [33] combined with a CP is taken as an example. It can be seen that the gain of the combination is

$$K_{PD} = \frac{I_{CP}}{2\pi}, \tag{3.3}$$

where I_{CP} is the current of both CP current sources. When using a Charge Pump, K_{PD} is expressed in $A \cdot \text{rad}^{-1}$.

The signal at the input of the PLL model, the reference signal, is the phase of the reference clock. This is a ramp ϕ_{Ref} with a slope corresponding to the reference frequency, with the reference phase noise ϕ_{nRef} superposed on it. The output signal ϕ_o is the phase of the generated clock, a ramp with a slope N times that of the reference signal, with the PLL output phase noise ϕ_{no} superposed on it.

3.2.2 Loop Filter Design

The most common implementation for the PLL loop filter, as used in a Charge Pump PLL, is shown in Figure 3.4. The capacitors integrate the Charge Pump current, thus creating a

[1] Compare this to the g_m of a transistor, that is taken in its biasing point.

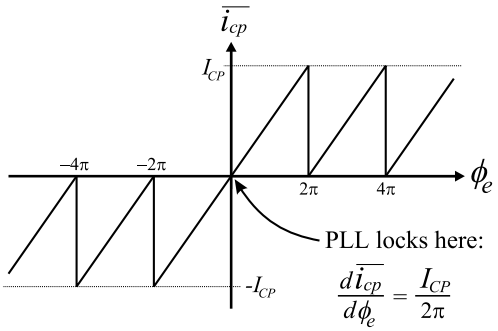


Figure 3.3: Determination of K_{PD} of a PFD/CP combination.

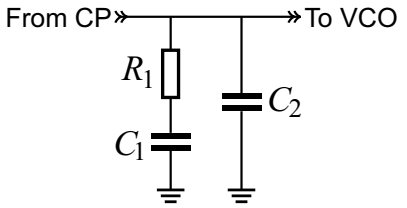


Figure 3.4: Common Loop Filter implementation.

type II system. This means there are two pure integrators[2] in the forward part of the feedback loop, the other one due to the integrating action of the VCO.

The importance of having a type II system follows from basic control theory: it can be shown that a type II feedback system with a ramp-type signal applied at its input will have zero tracking error [54]. This applies to the PLL, because a constant input frequency means a ramp-like phase-domain input signal. Having zero tracking error (a phase error of zero at the Phase Detector's input, not taking account of mismatch and other non-ideal effects) usually results in low reference feed-through due to low PD/CP activity.

Resistor R_1 adds a zero to the PLL loop gain, which is needed for an acceptable phase margin [29]. The main purpose of capacitor C_2 is extra filtering of the VCO control line, to prevent excessive reference feed-through. This might for example be caused by CP current source mismatch (see chapter 2), or by CP leakage [22].

The impedance of the loop filter of Figure 3.4 can be expressed as:

$$Z_{LF}(j\omega) = \frac{1}{j\omega(C_1 + C_2)} \cdot \frac{j\omega R_1 C_1 + 1}{j\omega R_1 \frac{C_1 C_2}{C_1 + C_2} + 1} \tag{3.4}$$

[2] In fact, the finite output impedance of the Charge Pump will be parallel to the loop filter, making a *lossy* integrator. The pole frequency is usually low enough, however, to neglect this fact.

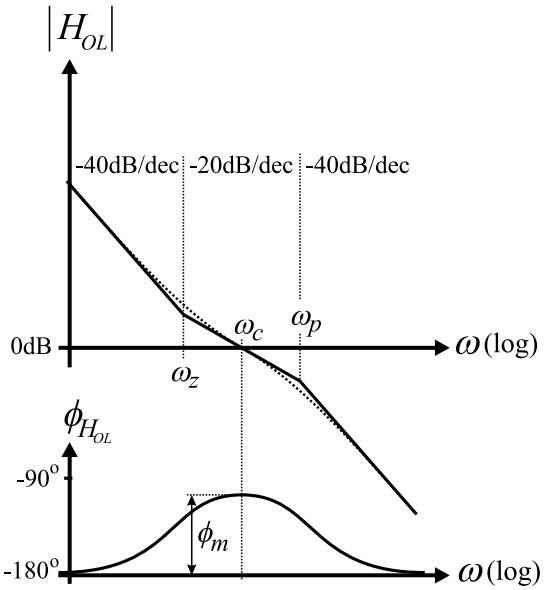


Figure 3.5: Bode-plot of the PLL loop gain.

Using the depicted loop filter, the general equation for the PLL loop gain will be of the form:

$$H_{OL}(j\omega) = -\frac{\omega_c\,\omega_z}{\omega^2} \cdot \frac{1 + j\frac{\omega}{\omega_z}}{1 + j\frac{\omega}{\omega_p}} \tag{3.5}$$

the Bode-plot of which is shown in Figure 3.5. The angular frequency of the loop gain zero, introduced by R_1 in series with C_1, is referred to as ω_z. The third pole that originates from adding the extra capacitor C_2 to the loop filter is at an angular frequency of ω_p. The so-called open-loop bandwidth ω_c [22] is the 0dB cross-over frequency of the PLL loop gain. The PLL phase margin is defined at this open-loop bandwidth, and is referred to as ϕ_m:

$$\phi_m = \arctan\left(\frac{\omega_c}{\omega_z}\right) - \arctan\left(\frac{\omega_c}{\omega_p}\right) \tag{3.6}$$

The PLL phase margin is an important parameter, as this determines the PLL settling time (minimum settling time occurs for a phase margin of about 50° [22]) and it determines the amount of peaking of the PLL noise transfer functions, and, thus, influences the total PLL output jitter (the higher the phase margin, the lower the peaking [22]).

Using the linear PLL model of Figure 3.2, the PLL loop gain can be written in terms of the PLL building block parameters:

$$H_{OL}(j\omega) = K_{PD}Z_{LF}(j\omega)\,\frac{K_{VCO}}{j\omega} \cdot \frac{1}{N} \tag{3.7}$$

We can combine this equation with the general PLL loop gain equation (3.5) and the equation for the loop filter impedance (3.4), to solve for the loop filter component values:

$$R_1 = \frac{N\omega_c}{K_{PD}K_{VCO}\left(1-\frac{\omega_z}{\omega_p}\right)} \approx \frac{N}{K_{PD}K_{VCO}} \cdot \omega_c \tag{3.8a}$$

$$C_1 = \frac{1}{\omega_z R_1} = \frac{K_{VCO}K_{PD}\left(1+\frac{\omega_z}{\omega_p}\right)}{N\omega_c\,\omega_z} \approx \frac{K_{VCO}K_{PD}}{N\omega_c\,\omega_z} \tag{3.8b}$$

$$C_2 = \frac{K_{VCO}K_{PD}}{N\omega_c\,\omega_p} = C_1 \cdot \frac{\omega_z}{\omega_p - \omega_z} \approx C_1 \cdot \frac{\omega_z}{\omega_p} \tag{3.8c}$$

where the right-hand side approximations apply to the case that $\omega_p \gg \omega_z$, which is generally the case because for a sufficient phase margin, the zero and pole frequency should be far enough apart. For example, for a phase margin of 60°, the third pole frequency should be at least 14 times higher than the frequency of the zero [22].

From the previous statement, it is obvious that for a sufficient phase margin, the capacitance of C_1 should be much larger than that of C_2. Using the same example, for a phase margin of 60°, the ratio C_1/C_2 should be at least 13 according to (3.8c). For an integrated loop filter, this means that C_1 will consume almost all the filter area.

3.3 PLL Output Phase Noise

All noise sources in Figure 3.2 will cause the output phase to deviate from its ideal value of N times the reference phase. This means there is a certain amount of jitter in the generated clock, according to (3.1). Because jitter is an important specification of a Clock Multiplier, it is important to analyze the effect of the various noise sources illustrated in Figure 3.2. Although this was partly done in the time domain in Chapter 2, this section describes the results of frequency domain analysis, for reasons mentioned earlier in section 3.2. A thorough frequency domain phase noise analysis is presented in [22]. The most important results of that analysis are briefly summarized here for convenience.

3.3.1 Equivalent Synthesizer Phase Noise Transfer

Noise sources ϕ_{nDiv} (the phase noise of the frequency divider) and i_{nCP} (the current noise generated in the CP) can be easily referred back to the reference input of the PD, such that the total phase noise at that node becomes:

$$S_{\phi_{neq}}(f_m) = S_{\phi_{nRef}}(f_m) + S_{\phi_{nPD}}(f_m) + S_{\phi_{nDiv}}(f_m) + \frac{S_{i_{nCP}}(f_m)}{K_{PD}^2} \tag{3.9}$$

with $S_{\phi_{neq}}(f_m)$ the so-called equivalent synthesizer phase noise floor at the input of the phase detector.

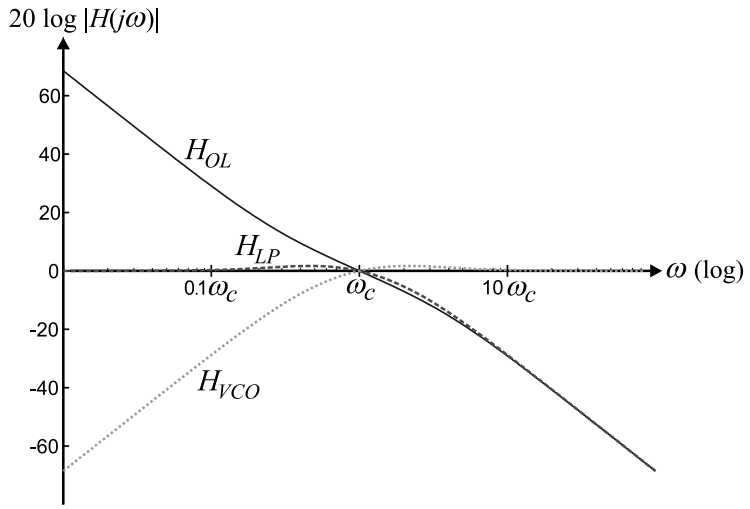


Figure 3.6: Transfer functions H_{OL}, H_{LP} and H_{VCO}.

This phase noise component is transferred to the PLL output according to the transfer function H_{Ref}:

$$H_{Ref}(s) = \frac{K_{PD}\, Z_{LF}(s)\, \frac{K_{VCO}}{s}}{1 + H_{OL}(s)} = N \frac{H_{OL}(s)}{1 + H_{OL}(s)} \equiv N H_{LP}(s) \tag{3.10}$$

using (3.7). H_{OL} is the PLL loop gain and H_{LP} is defined as the closed loop transfer from the reference input of the PD to the frequency divider output. Note that H_{LP} has a low-pass character, as illustrated in Figure 3.6, with a bandwidth equal to ω_c. This means that the equivalent synthesizer phase noise will mainly show up in the PLL output spectrum at offset frequencies lower than the PLL open-loop bandwidth. We will call the PLL output phase noise at offset frequencies lower than ω_c the *in-band phase noise*.

The PLL output phase noise spectrum due to the equivalent synthesizer phase noise is now expressed as:

$$S_{\phi_{no}}(f_m) = S_{\phi_{neq}}(f_m) \cdot N^2 \, |H_{LP}(j2\pi f_m)|^2 \tag{3.11}$$

The PLL output jitter due to the equivalent synthesizer phase noise can be calculated using the integral of (3.1):

$$\sigma^2_{t_{o,eq}} = \frac{N^2}{4\pi^2 f^2_{VCO}} \int_{f_l}^{f_h} |H_{LP}(j2\pi f_m)|^2 \, S_{\phi_{neq}}(f_m)\, df_m \tag{3.12}$$

As is shown in Appendix D, if the equivalent synthesizer phase noise PSD is white in the frequency band of interest and as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$, the PLL output jitter due to the

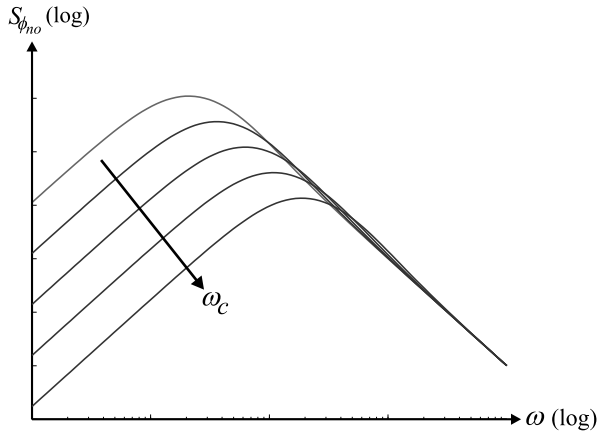


Figure 3.7: PLL phase noise due to intrinsic VCO phase noise.

equivalent synthesizer phase noise is proportional to ω_c for a constant PLL phase margin. A similar result was also obtained in Chapter 2 in the time domain.

3.3.2 VCO Phase Noise Transfer

The transfer of the intrinsic VCO phase noise ϕ_{nVCO} to the PLL output is easily found to be:

$$H_{VCO}(s) = \frac{1}{1 + H_{OL}(s)} \tag{3.13}$$

which has a high-pass characteristic, as illustrated in Figure 3.6. This means that the VCO phase noise will mainly show up at the PLL output at offset frequencies above the PLL open-loop bandwidth, as the equivalent synthesizer phase noise dominates at low offset frequencies.

The PLL output phase noise spectrum due to the VCO phase noise can now be written as:

$$S_{\phi_{no}}(f_m) = S_{\phi_{nVCO}}(f_m) \cdot |H_{VCO}(j2\pi f_m)|^2 \tag{3.14}$$

Figure 3.7 shows the PLL phase noise due to the intrinsic VCO phase noise for different values of ω_c, where it has been assumed that the VCO phase noise has a $1/f^2$ character. Because, according to (3.1), the PLL output jitter due to VCO phase noise is proportional to the integral of such a curve,

$$\sigma^2_{t_{o,VCO}} = \frac{1}{4\pi^2 f^2_{VCO}} \int_{f_l}^{f_h} |H_{VCO}(j2\pi f_m)|^2 \, S_{\phi_{nVCO}}(f_m) \, df_m \tag{3.15}$$

it is obvious that increasing the PLL bandwidth reduces the jitter due to the VCO phase noise. In Appendix D it is shown that as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$ and the VCO phase noise PSD

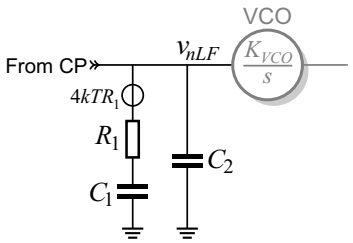


Figure 3.8: PLL Loop Filter with thermal noise.

has a $1/f^2$ shape, the PLL output jitter due to VCO phase noise is inversely proportional to the open-loop bandwidth. This result was also obtained in Chapter 2 in the time domain.

3.3.3 Loop Filter Noise Transfer

The resistor that is needed in the loop filter for PLL stability, will pollute the VCO control voltage due to the resistor's thermal noise. This disturbance causes jitter at the PLL output.

The PSD of the VCO control voltage noise caused by the resistor's thermal noise is easily established from Figure 3.8:

$$S_{v_{nLF}}(f_m) = 4kTR_1 \left(\frac{C_1}{C_1 + C_2}\right)^2 \left|\frac{1}{1 + j2\pi f_m/\omega_p}\right|^2 \tag{3.16}$$

The resulting phase deviations at the output of the VCO, in a closed-loop situation is, thus, found to be:

$$S_{\phi_{no}}(f_m) = S_{v_{nLF}}(f_m) \frac{K_{VCO}^2}{4\pi^2 f_m^2} \cdot |H_{VCO}(j2\pi f_m)|^2 = \tag{3.17}$$

$$= kTR_1 \left(\frac{C_1}{C_1 + C_2}\right)^2 \frac{K_{VCO}^2}{\pi^2 f_m^2} \cdot \frac{|H_{VCO}(j2\pi f_m)|^2}{\left|1 + j2\pi f_m/\omega_p\right|^2} \tag{3.18}$$

We will not discuss (3.18) in too much detail. Instead, in the following we assume that the PLL design is such that the VCO's internal phase noise will dominate the phase noise resulting from the loop filter resistor. This can usually easily be achieved by choosing the Phase Detector gain such that R_1 can be made small enough not to dominate the VCO phase noise. The maximum value for R_1 is analyzed in [22]:

$$R_1 < \left(\frac{C_1 + C_2}{C_1}\right)^2 \frac{\pi^2 S_{\phi_{nVCO}}(f_r)\, f_r^2}{kTK_{VCO}^2} \tag{3.19}$$

where an intrinsic VCO phase noise PSD with a $1/f^2$ shape is assumed, described by:

$$S_{\phi_{nVCO}}(f_m) = S_{\phi_{nVCO}}(f_r) \frac{f_r^2}{f_m^2} \tag{3.20}$$

with f_r the offset frequency at which the phase noise PSD is specified.

3.4 PLL Bandwidth Optimization

As was stated before, increasing the PLL bandwidth will result in a higher jitter contribution from the equivalent synthesizer phase noise, while lowering the jitter caused by the VCO. This observation brings up the notion of the existence of an optimum bandwidth, for which the PLL output jitter reaches a certain minimum. And indeed, in Chapter 2 it was already shown that such an optimum bandwidth exists.

Using the frequency domain analysis of PLL output phase noise, it is possible to determine the optimal bandwidth with a simple graphical method, as described in [35] and refined in [22]. This section briefly summarizes this method.

Basically, the graphical optimization involves drawing two graphs:

- The graph of the equivalent synthesizer phase noise PSD $S_{\phi_{neq}}(f_m)$, multiplied by N^2.
- The graph of the intrinsic VCO phase noise PSD $S_{\phi_{nVCO}}(f_m)$.

The first graph will, for frequencies above a certain $1/f$ corner frequency, have a flat, frequency independent spectrum in almost all cases. The second graph will have a $1/f^2$ shape, for careful VCO design [55] with a low transition frequency between the $1/f^3$-part and the $1/f^2$-part of the phase noise spectrum. For a certain frequency, which is called f_{xover}, these graphs will intersect. It is shown in [22] that we should choose ω_c equal to $2\pi f_{xover}$ in order to minimize the total PLL output jitter.

The graphical method is illustrated in Figure 3.9. This figure shows the resulting PLL output phase noise PSD $S_{\phi_{no}}$ when the optimum bandwidth is chosen (for a phase margin of 60°), and also when the bandwidth is chosen too high or too low. The optimum choice for ω_c clearly yields the lowest output jitter.

As is stated in [22], if the open-loop bandwidth ω_c is chosen optimally, the PLL output jitter due to the equivalent synthesizer phase noise sources will be exactly equal to the jitter contribution from the VCO. This only holds if the equivalent phase noise spectrum is flat at frequencies higher than f_l and the VCO spectrum has a $1/f^2$ character at offset frequencies below f_h. If *e.g.* the $1/f$ corner frequency of the equivalent synthesizer noise is above f_l, the jitter contribution of the VCO will be less than that of the other PLL building blocks. Note that making both contributions equal is not optimal in such a case.

If, however, the equivalent synthesizer phase noise *is* white, and the VCO phase noise *does* have a $1/f^2$ shape, the optimal open-loop bandwidth $\omega_{c,opt}$ can easily be found to be:

$$\omega_{c,opt} = \frac{2\pi}{N} \cdot \sqrt{\frac{S_{\phi_{nVCO}}(f_r)\, f_r^2}{S_{\phi_{neq}}}} \tag{3.21}$$

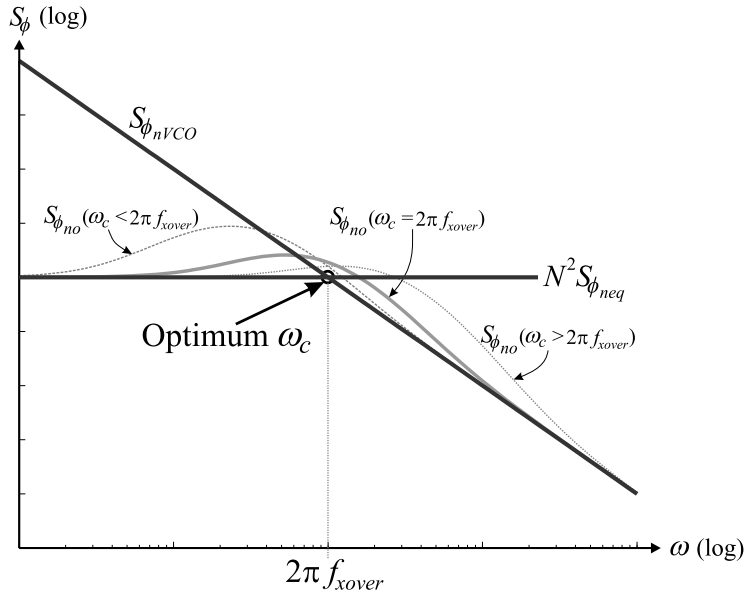


Figure 3.9: Graphical PLL optimization method.

and the total optimized jitter can then be expressed as:

$$\sigma_{t_o}^2 = \frac{\sigma_{\phi_{res}}^2}{4\pi^2 f_{VCO}^2} = \frac{\sqrt{S_{\phi_{neq}} \cdot S_{\phi_{nVCO}}(f_r)\, f_r^2}}{2\pi^2 f_{ref} f_{VCO}} \cdot \gamma^2(\phi_m) \tag{3.22}$$

with $\gamma(\phi_m)$ a factor depending on the PLL phase margin, that ranges roughly from 1.3 to 1.7 for acceptable values of ϕ_m [22].

From (3.8a) we can rewrite the PLL open-loop bandwidth ω_c in terms of the PLL building block parameters:

$$\omega_c = \frac{K_{PD} K_{VCO} R_1}{N}\left(1 - \frac{\omega_z}{\omega_p}\right) \approx \frac{K_{PD} K_{VCO} R_1}{N} \tag{3.23}$$

and optimizing the output jitter comes down to solving the equation:

$$\frac{K_{PD} K_{VCO} R_1}{N} = 2\pi f_{xover} \tag{3.24}$$

In the case of a PLL with a charge pump controlled by a PFD, using (3.3) yields:

$$I_{CP} R_1 = \frac{4\pi^2 N}{K_{VCO}}\, f_{xover} \tag{3.25}$$

The equation is written in this form, as K_{VCO} and N are often more or less fixed in the design by other considerations than optimization of output jitter.

Note that solving this equation might not always be as easy as the appearance of the equation suggests. Firstly, the value of f_{xover} is not always straightforward to determine as the noise spectra of the PLL building blocks are not easy to analyze by simulation (because of the time-varying and sampled nature of the noise sources). Secondly, changing the value of the charge pump current to influence the PLL open-loop bandwidth might also change the equivalent synthesizer noise, and, thus, the value of f_{xover} itself. Section 3.4.1 will briefly discuss this.

3.4.1 Optimal Loop Filter Capacitor Size

If we combine the knowledge about W-scaling (see section 2.6) with that of the determination of the optimal PLL open-loop bandwidth, we get an interesting result for a common PLL design. The equations derived in this section are valid for a PLL in which the following points hold:

- The given PLL power budget is mostly reserved for the high-frequency building blocks: the VCO and the high-speed section of the frequency divider. This means that the VCO can not be scaled to decrease its phase noise contribution.
- The equivalent synthesizer phase noise is dominated by the CP noise, which *can* be influenced by W-scaling.
- To reach the optimum bandwidth, only the loop filter and charge pump current can be altered.

Assuming an equivalent synthesizer phase noise that is dominated by CP current noise, yields:

$$S_{\phi_{neq}}(f_m) = \frac{S_{i_{nCP}}(f_m)}{K_{PD}^2} = 4\pi^2 \frac{S_{i_{nCP}}(f_m)}{I_{CP}^2} \tag{3.26}$$

if a tri-state Phase Frequency Detector is used, in which case K_{PD} is expressed by (3.3).

If we assume a white noise contribution from the CP, $S_{i_{nCP}}(f_m)$ is frequency independent. According to the W-scaling theory, described in section 2.6, the PSD is proportional to a scaling factor α that is used to scale the CP circuit. The value of the CP current I_{CP} itself is also proportional to the scale-factor α. These observations lead us to use the following notation for the PSD of the Charge Pump current noise:

$$S_{i_{nCP}}(f_m) = S_{i_{nCP}}(I_{CP_0}) \frac{I_{CP}}{I_{CP_0}} \tag{3.27}$$

with $S_{i_{nCP}}(I_{CP_0})$ the current noise PSD of the "base" CP circuit (before scaling, α=1), and I_{CP_0} the Charge Pump current of that particular circuit. I_{CP} is the CP current after W-scaling this "base" circuit. Substituting this in (3.26) results in:

$$S_{\phi_{neq}}(f_m) = 4\pi^2 \frac{S_{i_{nCP}}(I_{CP_0})}{I_{CP}\, I_{CP_0}} \tag{3.28}$$

showing that the equivalent synthesizer phase noise PSD is inversely proportional to the CP current I_{CP}, and, thus, to the W-scaling factor α.

Applying (3.21) to find the optimum open-loop bandwidth results in:

$$\omega_{c,opt} = \frac{1}{N}\sqrt{\frac{S_{\phi_{nVCO}}(f_r)\, f_r^2 I_{CP}\, I_{CP_0}}{S_{i_{nCP}}(I_{CP_0})}} \tag{3.29}$$

where the VCO phase noise is assumed to have a $1/f^2$ shape, such that it can be expressed mathematically by (3.20).

To obtain this open-loop bandwidth, the loop filter resistor is determined using (3.3) and (3.8a):

$$R_{1,opt} = \frac{2\pi}{K_{VCO}} \cdot \frac{\omega_p}{\omega_p - \omega_z}\sqrt{\frac{S_{\phi_{nVCO}}(f_r)\, f_r^2 I_{CP_0}}{I_{CP}\, S_{i_{nCP}}(I_{CP_0})}} \tag{3.30}$$

Now, an interesting result is obtained when the value of C_1, the largest loop filter capacitor, is calculated using (3.8b).

$$\begin{aligned} C_{1,opt} &= \frac{1}{\omega_z R_1} = \left(\frac{\omega_c}{\omega_z}\right)\frac{1}{\omega_c R_1} = \\ &= \underbrace{\frac{K_{VCO}N}{2\pi}}_{\text{I}} \cdot \underbrace{\left(\frac{\omega_c}{\omega_z} - \frac{\omega_c}{\omega_p}\right)}_{\text{II}} \cdot \underbrace{\frac{S_{i_{nCP}}(I_{CP_0})}{I_{CP_0}}}_{\text{III}} \cdot \underbrace{\frac{1}{S_{\phi_{nVCO}}(f_r)\, f_r^2}}_{\text{IV}} \end{aligned} \tag{3.31}$$

This last equation, consisting of four factors, leads to some interesting conclusions.

Firstly, the absence of the Charge Pump current I_{CP} in (3.31) means that the value of $C_{1,opt}$ is independent of the scaling factor that was applied to the "base" CP circuit. The value of $C_{1,opt}$ depends on the *quality* of the CP circuit with respect to noise, represented by factor III, and on the VCO phase noise quality, represented by factor IV. The quality of the CP design is not influenced by W-scaling the design; it is a measure of "how well" the CP is designed with respect to generated current noise, for a certain amount of power consumed in the CP. The value of the factor II is directly related to the PLL phase margin ϕ_m according to (3.6).

Another important conclusion from (3.31) is the dependence of the value of $C_{1,opt}$ on N, the multiplication factor of the PLL. This is an indication that in order to generate a certain output clock frequency, the use of a high reference clock frequency would yield a smaller loop filter capacitor area than when using a lower reference frequency. Note that (3.31) suggests that the value of $C_{1,opt}$ would be inversely proportional to the reference frequency for a certain VCO design and output frequency. This, however, need not necessarily be so, as the CP current noise PSD, represented in factor III of (3.31), in practice often depends on the reference frequency as well [22]. This point is discussed in more detail in section 3.5.

The effect of W-scaling the Charge Pump on PLL parameters is summarized in Figure 3.10, where it is shown that the product of $\omega_{c,opt}$ and $R_{1,opt}$ is unaffected by scaling, as long as the PLL open-loop bandwidth is optimized with respect to output jitter.

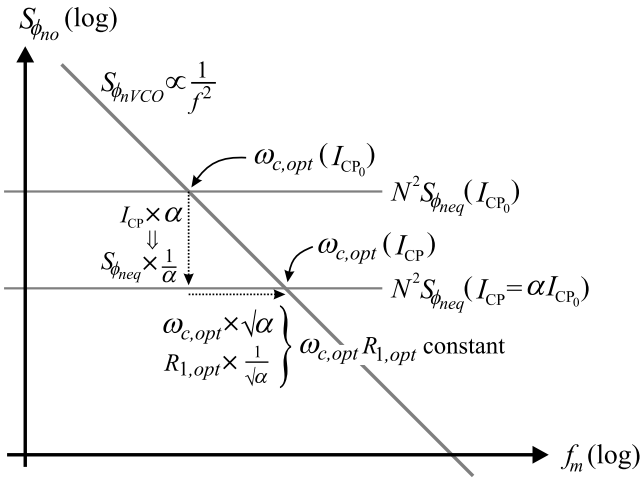


Figure 3.10: Effect of W-scaling the CP on PLL parameters.

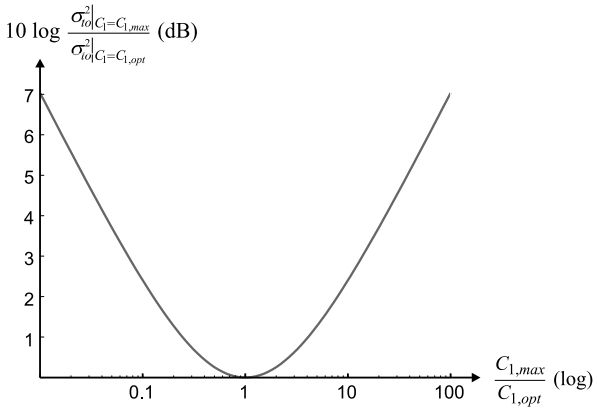


Figure 3.11: Jitter penalty due to non-optimal size of C_1.

Note that the above analysis also holds if the equivalent synthesizer phase noise is dominated by the Phase Detector phase noise, and the Phase Detector is scaled the same amount as the CP, to accommodate for the load the CP presents. In that case, the equivalent synthesizer phase noise is still inversely proportional to I_{CP}.

If the value for C_1 calculated with (3.31) is higher than acceptable, due to chip area constraints, and a smaller value for the capacitor was chosen, it is not possible to obtain the optimal PLL open-loop bandwidth with respect to jitter. The effect of decreasing the size of C_1 on the output jitter is analyzed now.

Decreasing the value of C_1 means that the value of R_1 needs to be increased to maintain a certain PLL phase margin, thus also increasing the PLL loop bandwidth. Suppose the maximum acceptable value for C_1 is referred to with $C_{1,max}$, the optimal value for R_1 then is[3]:

$$R_1 = R_{1,opt}\sqrt{\frac{C_{1,opt}}{C_{1,max}}} \tag{3.32}$$

resulting in an increase in both the PLL open-loop bandwidth ω_c and the zero-frequency ω_z of a factor $\sqrt{C_{1,opt}/C_{1,max}}$. Using the analysis of Appendix D, one can show that the jitter penalty on choosing a non-optimal value for C_1 is:

$$\frac{\sigma^2_{t_o}\big|_{C_1=C_{1,max}}}{\sigma^2_{t_o}\big|_{C_1=C_{1,opt}}} = \frac{1}{2}\frac{C_{1,max}+C_{1,opt}}{\sqrt{C_{1,opt}C_{1,max}}} \tag{3.33}$$

Figure 3.11 shows the graph of this jitter penalty. It should be noted that the jitter penalty shows a relatively flat optimum (*e.g.* decreasing C_1 with a factor of 10 results in a jitter increase of 2.4 dB). This means that in practice it is possible to trade PLL chip area for output jitter.

3.5 Influence of Reference Frequency on Phase Noise and Jitter

As was briefly noted before, the equivalent synthesizer phase noise spectrum at the PD input is influenced by the PLL reference frequency (or, more precise, the comparison frequency at the PD input). This is *e.g.* caused by altered CP activity, noise sampling effects or a different slope of the reference clock edges. In this section, the effect of changing the reference frequency on the equivalent synthesizer phase noise PSD is examined.

Referring to (3.12), the PLL output jitter can be written as:

$$\sigma^2_{t_{o,eq}} = \frac{S_{\phi_{neq}}}{4\pi^2 f^2_{ref}}\int_{f_l}^{f_h}|H_{LP}(j2\pi f_m)|^2\,d\,f_m \tag{3.34}$$

provided that the equivalent synthesizer phase noise PSD is white in the frequency band of interest. We use the notation $S_{\phi_{neq}}$ for this white spectrum, which we now assume to be dependent on f_{ref}.

From (3.34) we see that if the dependence of $S_{\phi_{neq}}$ on f_{ref} is a weaker function than proportional to f^2_{ref}, the PLL output jitter benefits from increasing the reference frequency.

[3]This is under the assumption that the PLL phase margin is to be kept constant, *e.g.* to bound phase noise peaking.

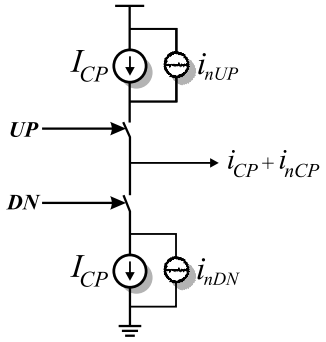


Figure 3.12: Noisy Charge Pump model.

3.5.1 Charge Pump Noise as a function of f_{ref}

Increasing the reference frequency will in general raise the noise at the CP output (indicated by $S_{i_{nCP}}$), due to increased CP activity. Here we will examine whether this dependency is indeed weaker than proportional to f_{ref}^2, such that increasing the reference frequency would yield lower CP noise induced output jitter.

Figure 3.12 shows the most elementary Charge Pump circuit with noisy current sources. Assuming white current noise spectra, the CP output noise PSD is given by:

$$S_{i_{nCP}}(f_m) = \frac{t_{ovl}}{T_{ref}} S_{i_{nUP}} + \frac{t_{ovl}}{T_{ref}} S_{i_{nDN}} = 2\, t_{ovl}\, f_{ref}\, S_{i_n} \tag{3.35}$$

where t_{ovl} is the width of the *UP* and *DN* pulses at 0° phase error. Usually, the modulation of the pulse-width due to phase errors is a small fraction of t_{ovl} when in lock, such that the multiplicative effect on the noise can be neglected. S_{i_n} is the noise spectrum of both the *UP*- and the *DN*-current source; we assume both current sources have equal noise spectra.

Using (3.9) and assuming a PFD is used yields an equivalent synthesizer phase noise PSD of:

$$S_{\phi_{neq}} = S_{i_n} \frac{8\pi^2 t_{ovl}\, f_{ref}}{I_{CP}^2} \tag{3.36}$$

showing a proportional relation to the reference frequency, which is weaker than f_{ref}^2. And, indeed, the PLL output jitter due to the CP noise is, using (3.34):

$$\sigma_{t_o}^2 = S_{i_n} \frac{2\, t_{ovl}}{f_{ref}\, I_{CP}^2} \int_{f_l}^{f_h} |H_{LP}(j2\pi f_m)|^2\, d f_m \tag{3.37}$$

This shows that the in-band phase noise PSD at the PLL output (due to CP noise) is inversely proportional to the reference frequency, assuming the simple CP model of Figure 3.12.

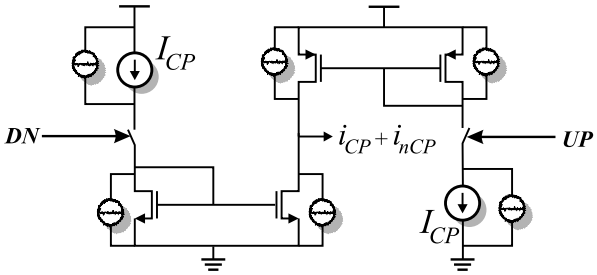


Figure 3.13: Charge Pump with noisy current mirrors.

According to (3.36), the equivalent synthesizer phase noise PSD is proportional to the overlap time of the *UP* and *DN* pulses of the PD controlling the CP. This means that in a low jitter design, this overlap time should be as short as possible without introducing dead-zone. This was also seen when examining the errors introduced by CP current source mismatch, in section 2.5.2.

When implementing the CP circuit in a low-voltage process such as modern CMOS, stacking of the current sources and the switches might be a problem. To increase the maximum voltage swing at the CP output (*i.e.* the VCO control voltage), current mirrors can be used, see Figure 3.13. When using mirrors, the mirror transistors may become the dominant sources of noise.

If we increase the reference frequency, the mean output current of the mirrors will increase proportionally to the reference frequency. If we assume that the mirrors are scaled to accommodate for this increase in current, thus, keeping the overdrive voltage constant, W-scaling theory predicts that the charge pump current noise PSD is proportional to the mean mirror output currents $\overline{I_D}$. This yields:

$$S_{i_{nCP}} \propto \overline{I_D} = \frac{t_{ovl}}{T_{ref}} I_{CP} = t_{ovl} f_{ref} I_{CP} \tag{3.38}$$

as was the case for the simple CP without mirrors. The conclusion for the PLL output jitter, thus, still holds: the in-band PLL output phase noise is inversely proportional to the reference frequency, and proportional to the overlap time of the *UP* and *DN* pulses, given by t_{ovl}.

3.5.2 PD and Divider Noise as a function of f_{ref}

The PD and frequency divider usually are digital blocks that respond to zero crossings of their input signals. This time-discrete behavior causes sampling of the input referred noise at the input of these blocks. If the noise has a bandwidth wider than half the sampling frequency, sampling results in aliasing.

The easiest way to calculate the aliasing effects is to use the first-crossing method [56, 57] to convert input voltage noise into timing errors. If these timing errors are a result of white

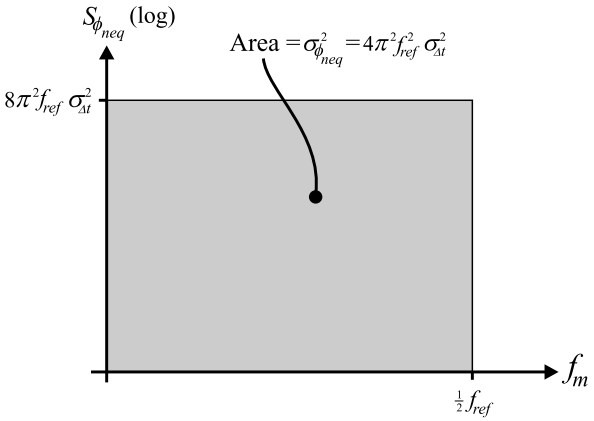


Figure 3.14: Effect of aliasing on input referred phase noise.

noise at the input, the errors are uncorrelated. The variance of the timing errors $\sigma^2_{\Delta t}$ can be converted to a phase error variance $\sigma^2_{\phi_{neq}}$:

$$\sigma^2_{\phi_{neq}} = \sigma^2_{\Delta t} \cdot 4\pi^2 f^2_{ref} \tag{3.39}$$

Using the fact that the PSD of these phase errors will be flat and band-limited by $f_{ref}/2$ due to sampling [36], the equivalent synthesizer phase noise PSD is easily shown to be:

$$S_{\phi_{neq}}(f_m) = \sigma^2_{\Delta t} \cdot 8\pi^2 f_{ref} \tag{3.40}$$

as illustrated by Figure 3.14, showing that the equivalent synthesizer noise due to PD noise and due to frequency divider noise is proportional to f_{ref}. This result was shown for a PFD in [58]. In [59] this result is confirmed for frequency dividers.

Using (3.34) finally leads us to conclude:

$$\sigma^2_{t_{o,eq}} = \frac{2\,\sigma^2_{\Delta t}}{f_{ref}} \int_{f_l}^{f_h} |H_{LP}(j2\pi f_m)|^2\, d f_m \tag{3.41}$$

again showing an in-band PLL phase noise inversely proportional to the reference frequency.

3.5.3 Consequences of Increasing f_{ref}

It has been shown that, in general, the effect of increasing the reference frequency is a lower in-band PLL phase noise level. This has two important consequences: the total PLL output jitter will be lower and the necessary size of the loop filter capacitor C_1 is smaller due to an increased optimum PLL open-loop bandwidth.

Using (3.22), we conclude that the optimum jitter achievable with a PLL is proportional to:

$$\sigma_{t_o}^2 \propto \frac{\sqrt{S_{\phi_{neq}}}}{f_{ref}} \tag{3.42}$$

Because $S_{\phi_{neq}}$ is in general proportional to f_{ref}, as discussed before, the PLL output jitter $\sigma_{t_o}^2$ will be inversely proportional to the square root of f_{ref}, an important reason to choose the reference frequency as high as possible when designing a PLL-based CMU.

Because the in-band PLL phase noise level is generally inversely proportional to the reference frequency, the optimal PLL open-loop bandwidth will be proportional to the square root of f_{ref}, provided the VCO phase noise PSD has a $1/f^2$ shape. Assuming the zero-frequency ω_z also scales with $\sqrt{f_{ref}}$, the necessary loop filter capacitor C_1 is inversely proportional to f_{ref} for a constant value of I_{CP}. Because in many fully integrated PLLs, the loop filter takes a considerable part of the total area, this can be an important benefit of using a high reference frequency.

Apart from the advantages mentioned here, the PLL also benefits from a high reference frequency because of more efficient filtering of the reference breakthrough [22]. And in some cases, the increased maximum PLL bandwidth, which is about one tenth of the reference frequency according to a rule-of-thumb for PLL stability [40], can be an argument for using a high PLL reference frequency.

3.6 Summary of Conclusions

Because time domain analysis of PLL jitter is only convenient in a simplified case (with a simple first-order loop filter and white noise sources), this chapter used phase noise analysis in the frequency domain. Jitter can be then determined by integrating the phase noise spectrum. Frequency domain analysis allows for a very simple method to determine the PLL bandwidth resulting in the lowest output jitter. When optimized, the phase noise at offset frequencies above the PLL bandwidth is dominated by the intrinsic VCO phase noise, while below the PLL bandwidth, the other loop components dominate the noise. Because half of the total output jitter is due to the components other than the VCO, it is rewarding to minimize this in-band phase noise. It has been shown that this can be done effectively by using as high a reference frequency as possible. To minimize charge pump noise, the phase detector should generate pulses with a short active duration.

Via the concept of an optimum loop filter capacitance, it has been shown that PLL area can be saved, at the price of increased output jitter. However, because of the flat optimum, this price is small as long as the actual capacitance area is not reduced too much.

4

Dimensioning Current Mode Logic

This chapter describes a simple way to dimension CMOS Current Mode Logic digital gates, based upon the maximum fully switchable tail current of a gate whose transistors have a given size. Using this approach, the speed of a gate as a function of digital voltage swing is analyzed, showing that high voltage swings are beneficial for the speed. Simulations were performed in a standard 0.18μm CMOS process to verify gained insight.

4.1 Introduction

A considerable part of a PLL consists of digital building blocks. The frequency divider section usually is composed of D-flip-flops and digital gates [21]. Also, the phase detector block generally consists of digital gates.

Although, from a designer's point of view, standard CMOS digital blocks would be easiest to use (using standard digital gate libraries), this logic family comes with some important disadvantages. Most importantly, standard CMOS digital gates are notorious for generation of delta-*I* noise [60–62], sharp current pulses due to charging and discharging of the gate's parasitic capacitances. This delta-*I* noise is directly measurable as disturbances on the power supply and the substrate, which can be an important drawback when designing a mixed analog-digital system. For example, in a PLL, modulating the power supply is generally directly measurable at the output of the VCO [63].

Adopting the MOS Current Mode Logic style [64, 65] (MCML, CML), sometimes referred to as Source Coupled Logic (SCL), greatly reduces the generated switching noise by about two orders of magnitude [66, 67]. This reduction is due to the differential and current steering nature of the logic style. The low delta-*I* noise generation makes the CML style very suitable for mixed signal designs.

Apart from the low *generation* of delta-*I* noise, the CML style is also less sensitive to already present supply and substrate noise, due to the differential signalling used [61, 68]. This means that critical digital blocks such as used in a PLL are less sensitive to disturbances caused by standard CMOS digital blocks integrated on the same die.

Although it is possible to use a simple *RC*-filter in combination with the standard CMOS style to both reduce generated supply noise and the gates' sensitivity to supply noise [19, 69], the resistive component of the filter will decrease the effective supply voltage of the gate, thus reducing its speed. Also, this technique will be less effective when dealing with baseband digital signals, such as NRZ-data for example (having a spectrum starting from DC [21]). This is because the part of the spectrum below the filter's corner frequency is not attenuated and will influence performance. Using the CML style, both the generation of and the insensitivity to supply and substrate variations are handled by using differential signalling, instead of by filtering, which also works for low frequencies.

The fact that the CML style generates a smaller digital voltage swing also offers some advantages over rail-to-rail CMOS logic styles. Firstly, the smaller swing reduces crosstalk between adjacent signals [61]. Secondly, the dynamic power dissipation resulting from charging and discharging parasitic capacitances is generally smaller than is the case for full-swing logic [61], which can especially be advantageous when controlling long busses. For high-

frequency operation, this effect can result in a lower total power consumption than for standard CMOS cells, even though CML gates suffer from static power consumption [68, 70, 71].

A possible disadvantage of using CML is its slightly lower speed than some rail-to-rail CMOS families [61], such as the True Single-Phase Clocking (TSPC) family [72] sometimes used in frequency dividers[1] [74, 75] or precharged CMOS in the phase detector [76, 77]. Also, the area consumption of rail-to-rail CMOS gates is generally smaller than than that of CML gates, which is beneficial for the capacitance of interconnections [74].

Another disadvantage of using Current Mode Logic is implied before: there usually are no standard CML cells in a digital library that can be used directly by the designer. Designing CML gates is more involved than designing their rail-to-rail CMOS counterparts, because of the many degrees of freedom (static current, sizes of differential pair transistors, digital swing). Although some design equations are given in [78, 79], these depend a quadratic transistor model (which is inaccurate for deep-submicron CMOS devices) and on extensive knowledge of process and transistor parameters.

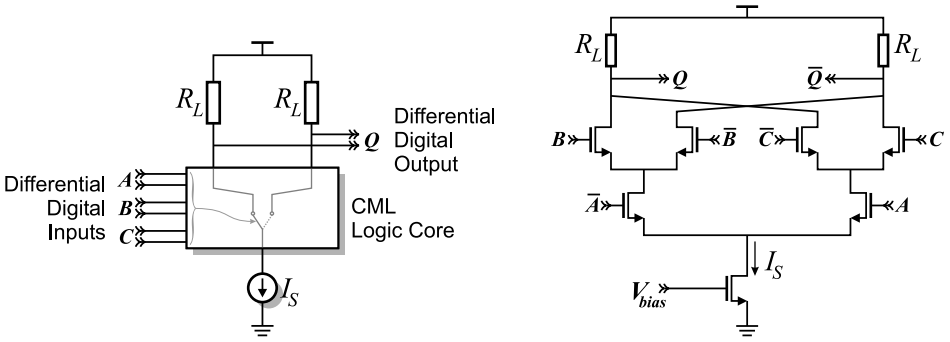
This chapter describes an easy way of optimizing the CML gates with respect to speed, without the need for "tedious simulation iterations [78]", using the concept of maximum tail current for given transistor dimensions and digital swing. Section 4.2 introduces the general structure of Current Mode Logic digital gates. In section 4.3, some basic CML gate design issues are discussed by examining the most simple and basic CML gate: the CML buffer or inverter. The operation speed of the gate as a function of digital voltage swing and transistor sizes is discussed, followed by simulation results in a standard 0.18μm CMOS technology. Section 4.4 then shows the results of gained knowledge as applied to the design of a 2-input CML gate.

4.2 CML gate structure

Every CML gate consists of a tail current source, a current steering logic core and a differential load. This is shown conceptually in Figure 4.1(a). The task of the logic core is to direct the tail current through one of the load resistors, thereby defining the differential output voltage as a function of the differential input signals. Note that in some cases, the resistors are replaced by triode PMOS devices, using replica biasing to control the digital voltage swing [38, 64, 80]. Also, some designs leave out the tail current source to increase the maximum operation speed (20% improvement is claimed in [81, 82]), but this makes for a less robust design as behavior and current dissipation now depend on the input signal level. Also, this technique will result in higher delta-*I* noise, as the total current is no longer kept constant by design.

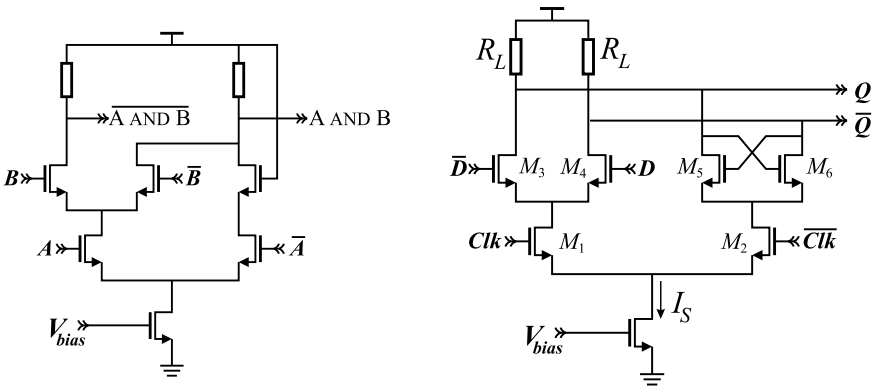
The logic core is realized using differential pairs that fully switch their tail current one way or the other. The CML logic core topology can be designed in a systematic manner, as described in [66]. Figure 4.1(b) shows an example 3-input CML gate.

[1] It is interesting to see the conflicting statements from both 'camps': [73] mentions the disadvantage of a power-hungry buffer needed when using TSPC, while [74] claims the need of such a buffer when using CML...



(a) CML concept **(b)** $Q = (A \text{ AND } C) \text{ OR } (\overline{A} \text{ AND } \overline{B})$

Figure 4.1: General Current Mode Logic circuit.



(a) CML 2-input AND **(b)** CML D-latch

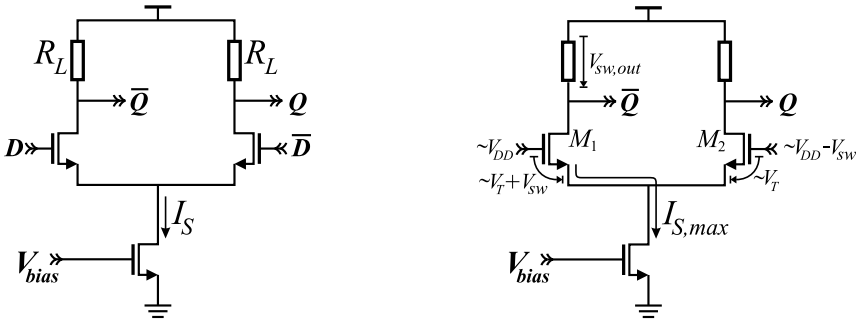
Figure 4.2: Examples of CML circuits.

All CML input and output signals are fully differential. An important inherent property of using differential signalling is that the inverted signals are always available. This means that a lot of clock skew problems are avoided using CML. This is not the case when using standard CMOS logic, where often the gate's output has to be inverted explicitly, using inverters.

Due to the full switching of the tail current, the differential output swing $V_{sw,out}$ is easily seen to be:

$$V_{sw,out} = I_S \, R_L \tag{4.1}$$

A very common CML gate structure is illustrated in Figure 4.2(a), a CML AND gate. The same CML circuit can also be used as *e.g.* a NAND, OR and NOR gate, depending on the connection of input and output signals. The transistor whose gate is connected to the V_{DD} prevents offset in the bottom differential pair.



(a) CML buffer circuit **(b)** Buffer operating at maximum tail current

Figure 4.3: CML buffer (general circuit and speed-optimized situation).

A final example is shown in Figure 4.2(b), which is a CML D-latch. If the clock signal is high, the circuit is transparent (the output Q will follow the input D). A low clock signal will make the cross-coupled circuit consisting of M_5 and M_6 active, thus latching the output signal. A CML master-slave D-flip-flop can be constructed using two D-latches in cascade, one controlled by the inverted clock signal. This circuit is applied often in the PLL's frequency divider [73, 80, 83, 84].

4.3 CML buffer

The simplest possible CML circuit is the CML buffer (or CML inverter, which has the same circuit topology, but with reversed output signal definition), as illustrated in Figure 4.3(a). This same circuit is also used as a delay-cell in the delay line of a DLL [57]. Because of the simplicity of this circuit (there is no stacking of differential pairs in the 'logic core'), the optimization with respect to speed is analyzed for the CML buffer first.

For correct operation of the CML buffer, we desire that the digital input voltage fully switches the tail current I_S one way or the other. If the input pair is not completely switched, part of the tail current is common for both input transistors and does not contribute in the differential output signal. Also, not fully switching the input pair means that the actual differential output current will be sensitive to temperature and input pair offset voltage, which is undesirable.

4.3.1 Maximum Tail Current

If we assume a given size of the input transistors and a given input signal swing V_{sw}, the condition of fully switching the input pair means there is a *maximum value* $I_{S,max}$ for the tail current. Increasing the tail current above this maximum will lead to incomplete switching, as will be explained shortly. This maximum current is important: a high value for I_S is in principle favorable for high-speed operation. This is because a high tail current means a low

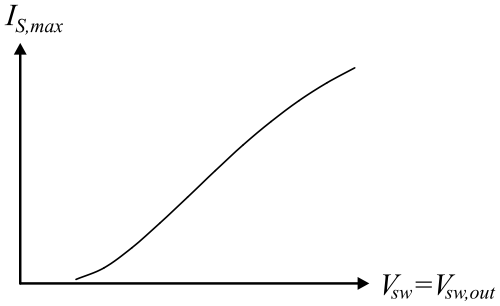


Figure 4.4: Typical plot of maximum tail current as a function of the differential swing.

value for the load resistors R_L (given the output voltage swing) according to (4.1), which leads to a low output *RC* time constant. This time constant is formed by the load resistors and the parasitic capacitance at the buffer output, partly due to the buffer's own output capacitance and partly due to the buffer's load capacitance. Because the rise- and fall times of the gate, and the gate delay, are inversely proportional to the output *RC* time [78], R_L should be minimized.

We analyze the maximum tail current $I_{S,max}$ using Figure 4.3(b). This figure shows that if I_S is equal to $I_{S,max}$, transistor M_2 will have a gate-source of about the threshold voltage V_T. Increasing I_S above $I_{S,max}$ would lead to conduction of M_2, thus, incomplete switching occurs. If I_S equals $I_{S,max}$, the overdrive voltage of M_1 will roughly equal the input swing V_{sw}, while conducting a drain current of $I_{S,max}$. Finally, the drain-source voltage of M_1 is $V_{DS_1} = V_T + V_{sw} - V_{sw,out}$.

Using these data, $I_{S,max}$ can be found using a plot of the drain current of M_1 (or M_2) versus its overdrive voltage V_{GT}, using the previously given drain-source voltage. As V_{GT} equals V_{sw}, this plot is, thus, a plot of the maximum tail current $I_{S,max}$ as a function of the input voltage swing, with the size of M_1 and M_2 and the output voltage swing $V_{sw,out}$ as parameters. A typical plot is shown in Figure 4.4, where the buffer's output swing was chosen to be equal to the input swing.

Note that in practical robust designs, dimensioning the tail current equal to the maximum tail current is risky practice. This is because in such a case, small deviations from expected component values due to process spread (*e.g.* the load resistances) might cause the CML gates to operate incorrectly due to incomplete switching. Also, offset in the input differential pair can cause incomplete switching when choosing the maximum tail current [71]. This means that sufficient safety margin should be used when the CML gates are designed, at the cost of decreased speed.

Also note that in practice, M_2 of Figure 4.3(b) *will* in fact conduct a small current, even if I_S does not exceed $I_{S,max}$, due to its behavior in moderate or weak inversion; even if its gate-source voltage is below V_T, some current will flow. This current is however a small percentage of the total tail current and is neglected here. In the simulations that are described later, switching 98% percent of the tail current is considered "full" switching.

4.3.2 Load resistor value

From (4.1) we can conclude that the load resistor value is proportional to the logic swing and inversely proportional to the tail current source value. Because the maximum switchable tail current itself is a function of the logic swing, it is interesting to see how the load resistor value depends on the logic swing with respect to operation speed. We analyse this using a simple MOS transistor model, referred to as the alpha-power law MOSFET model [85, 86]. This model replaces the usual square-law model by introducing the parameter α_i that would equal two in the square-law model, but is somewhere between 1 and 2 in a sub-micron MOSFET due to mobility saturation. The MOSFET drain current I_D in saturation is described by:

$$I_D = I_{D_0} \left(\frac{V_{GS} - V_T}{V_{GS_0} - V_T} \right)^{\alpha_i} \tag{4.2}$$

with I_{D_0} the MOSFET drain current for a gate-source voltage of V_{GS_0} and α_i the model parameter from which the MOSFET model derives its name [85]. The drain current is not specified for a MOSFET in triode in the alpha-power law model: the following equations only hold when operating the input pair in saturation.

As long as M_1 and M_2 of the CML buffer of Figure 4.3(b) operate in saturation, the maximum tail current can be approximated by:

$$I_{S,max} = I_{D_0} \left(\frac{V_{GS_1} - V_T}{V_{GS_0} - V_T} \right)^{\alpha_i} = I_{D_0} \left(\frac{V_{GT_1}}{V_{GS_0} - V_T} \right)^{\alpha_i} = I_{D_0} \left(\frac{V_{sw}}{V_{GS_0} - V_T} \right)^{\alpha_i} \tag{4.3}$$

The substitution $V_{GT_1} = V_{sw}$ is justified in Figure 4.3(b).

The value of the load resistor for a tail current equal to $I_{S,max}$ can now be seen to be, using (4.1),

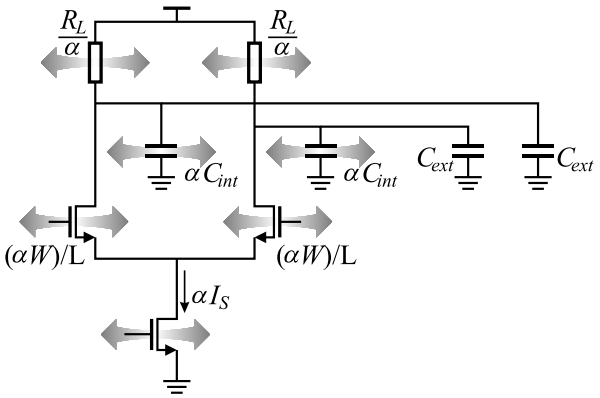
$$R_L = \frac{V_{sw,out}}{I_{S,max}} = \frac{V_{sw,out}}{I_{D_0}} \left(\frac{V_{sw}}{V_{GS_0} - V_T} \right)^{-\alpha_i} \tag{4.4}$$

In many cases, the output voltage swing will be chosen to be equal to the input swing ($V_{sw,out} = V_{sw}$) and we can write

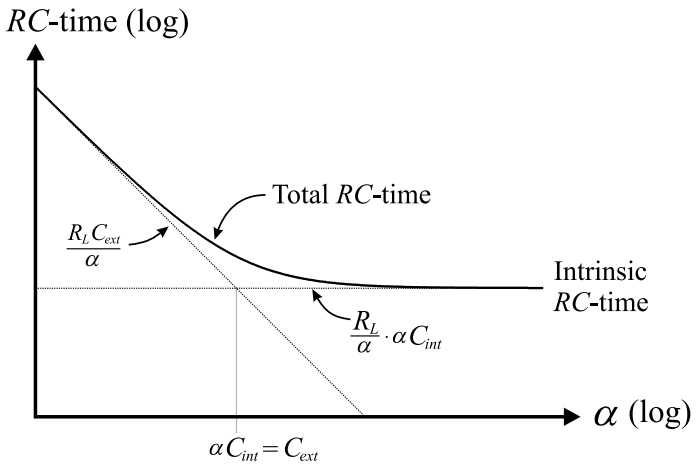
$$R_L = \frac{1}{I_{D_0}} \, (V_{GS_0} - V_T)^{\alpha_i} \, V_{sw}^{(1-\alpha_i)} \tag{4.5}$$

We see that as long as α_i is higher than one[2], the calculated value of the load resistor will drop with increasing input and output swing. Assuming a constant capacitive output load of the buffer, this means that a high digital swing is beneficial for the speed. This directly negates the common assumption that the high speed of CML circuits is due to the low voltage swing [61, 71, 78, 83].

[2]This means that the drain current is a rising function of the overdrive voltage, which can be assumed for most MOS devices.



(a) W-scaling CML buffer



(b) Scaling influence on RC-time

Figure 4.5: Relation between power usage and speed of CML gates

4.3.3 Speed versus tail current

Once the maximum tail current $I_{S,max}$ is known for a certain width of the input pair transistors, the W-scaling theory described in section 2.6 can be applied to calculate the maximum current for differently sized input transistors. This means that if the input pair transistor width is doubled for example, the value for $I_{S,max}$ will also be double, while the value of the load resistors R_L should be divided by two to maintain a constant output swing.

If the differential pair is loaded by an external capacitance (in addition to the internal parasitic output capacitance at the drains of the differential pair transistors), the above procedure of increasing the widths of the input pair and the tail current, while decreasing the load re-

sistance, will decrease the *RC*-time associated with the buffer's output. This is illustrated in Figure 4.5. Figure 4.5(a) shows the consequence of applying W-scaling to the CML gate. All transistor widths of the buffer are multiplied by α, leading to an increase of the buffer's output capacitance C_{int} by a factor of α. Because the tail current is also multiplied by α, the load resistance should be divided by α.

Figure 4.5(b) illustrates the consequence of this operation on the overall *RC*-time of the CML buffer, and thus, on the delay of the buffer, which is proportional to this *RC*-time [78]. For a small value of α, resulting in low power consumption, the output capacitance C_{ext} dominates the total capacitance. In this region, increasing power dissipation will result in an increase of speed. However, increasing the dissipation above a certain point, a further increase will not lead to a significant speed improvement due to the fact that the buffer's intrinsic parasitic capacitance dominates the total capacitance. It may, however, be fruitful to increase dissipation even further. This may be because either the jitter generated by the buffer or the differential pair offset needs to be improved.

4.3.4 Voltage headroom of the tail current source

Apart from the condition of fully switching the buffer's input pair, there is another important condition for correct buffer operation: the tail current source transistor should never enter the triode region. This would cause steep drops in the total buffer current, which is a problem as constant supply current is usually an important reason to use CML. Also, a drop in current will slow down the operation of the CML buffer.

The voltage across the tail current will be lowest when the differential buffer input voltage is zero, so both transistors of the differential pair conduct half the tail current. If this tail current equals $I_{S,max}$, we can write for the drain currents of both input transistors, using (4.3) and the alpha-power law MOSFET model [85, 86]:

$$I_D = I_{D_0}\left(\frac{V_{GS} - V_T}{V_{GS_0} - V_T}\right)^{\alpha_i} = \frac{1}{2} I_{S,max} = \frac{1}{2} I_{D_0}\left(\frac{V_{sw}}{V_{GS_0} - V_T}\right)^{\alpha_i} \tag{4.6}$$

If the differential input voltage of the CML buffer is zero (the previous CML stage being in its zero crossing), the gate voltage of the input transistors equals $V_{DD} - \frac{1}{2} V_{sw}$. Using this to solve (4.6) for V_S, which equals the voltage across the tail current, yields:

$$V_S = V_{DD} - V_T - V_{sw}\left\{\frac{1}{2} + \left(\frac{1}{2}\right)^{\frac{1}{\alpha_i}}\right\} \tag{4.7}$$

showing a linear decrease of the voltage V_S across the tail current source with the digital voltage swing V_{sw}. This means that in a low-voltage CMOS process, the tail current source voltage headroom limits the CML swing used.

4.3.5 CMOS Simulation Results of CML Buffer

Simulations with a CML buffer, loaded with four identical buffers, were performed in a standard 0.18μm CMOS process. The buffer's digital output swing was chosen equal to the applied input swing in every case, by choosing R_L depending on the CML tail current according to (4.1). The supply voltage was 1.8 V.

Figure 4.6(a) shows the maximum tail current that can be fully switched by the input pair, as a function of the input digital voltage swing. The width of the minimum-length input pair transistors was 20μm. Above approximately 800 mV input (and output) swing, the input pair enters the triode region. Note that, as described earlier, switching 98% percent of the tail current is considered "full" switching.

In Figure 4.6(b), simulation results of the buffer dimensioned with the maximum tail current values as plotted in Figure 4.6(a) are shown. Both the buffer's delay time (which is the time between the zero-crossing of the differential input signal and that of the differential output signal) and the rise and fall time of the differential output signal (the rise time is the time it takes the signal to increase from 10% to 90% of its final value) were obtained for different values of the input (and output) voltage swing.

Figure 4.6(b) shows a decrease in both the delay time and the rise and fall time of the CML buffer for large input voltage swings. This is due to the predicted lower value of the load resistance for a high input voltage swing (also plotted in Figure 4.6(b)), see equation (4.5). When the output swing is high enough to put the input pair into triode, we see that the speed does not improve significantly, and this should generally be avoided to save power usage of the digital cells[3].

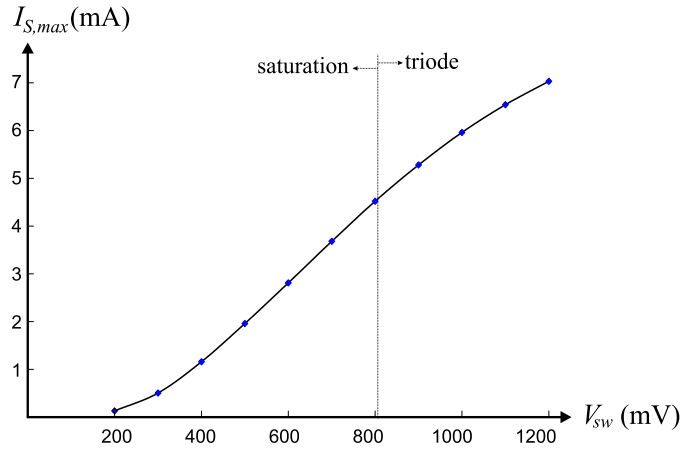
The effect of decreasing the tail current while maintaining equal input and output swings is illustrated in Figure 4.7, for a swing of 500 mV and 900 mV. This figure clearly shows the expected speed penalty resulting from a tail current that is too small, which is due to the higher value of the load resistance needed to maintain the output swing. It was mentioned before, however, that a certain safety margin should be considered when choosing the CML tail current, so a small loss in speed should in general be accepted.

Figure 4.7 also shows that the ratio between the rise time and the buffer's delay increases for large tail currents. This is mainly due to the fact that the input window of the differential pair is larger for high tail currents, causing the output signal to rise before the input signal crosses zero.

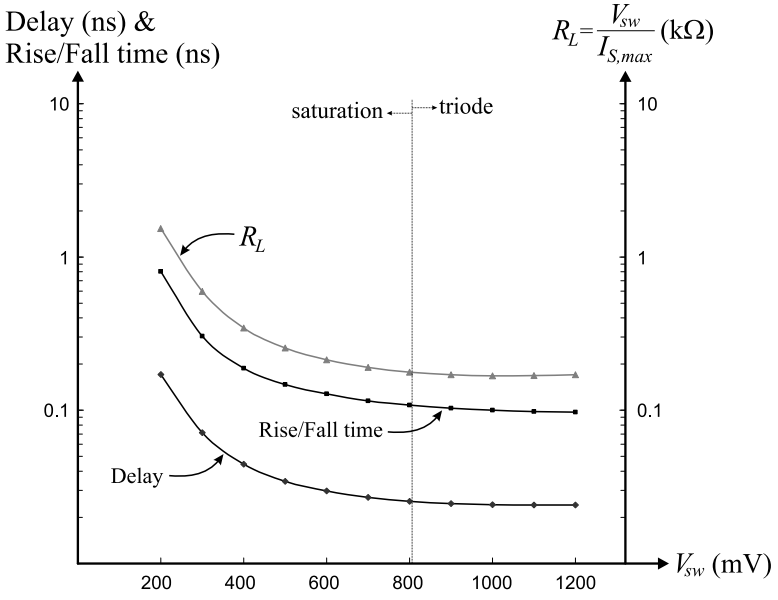
From the simulation results we conclude that for optimum behavior with respect to speed, the input signal swings should be kept high and the tail currents should not be chosen too small (or, conversely, the width of the input pair transistors should be kept close to minimal).

Figure 4.8 shows the simulated voltage across the tail current source as a function of the input voltage swing (where the value of the tail current depends on the input swing as in Figure 4.6(a)). The supply voltage used in this (and every other) simulation was 1.8 V. The

[3]Note that operating the differential pair in the triode region is not as devastating for the speed as it is in the bipolar ECL family, where saturating the bipolar differential pairs takes long recovery times.

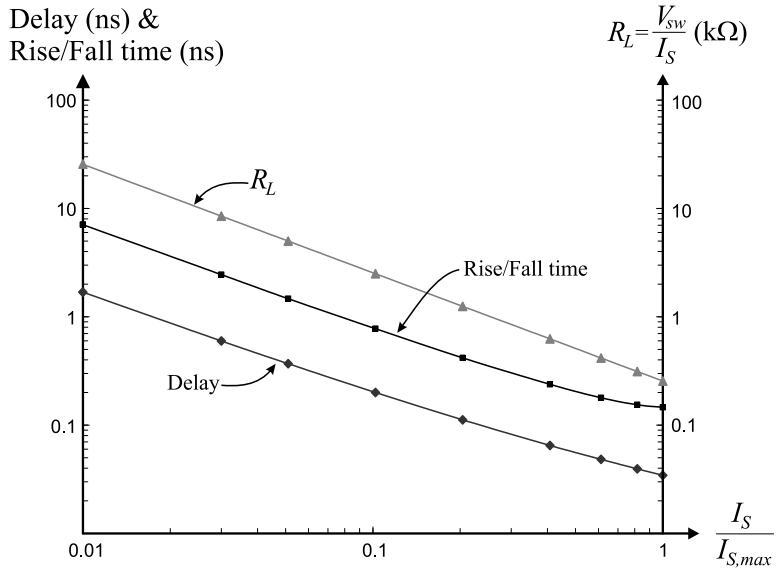


(a) Maximum tail current versus input and output swing (W/L input pair is 20/0.18)

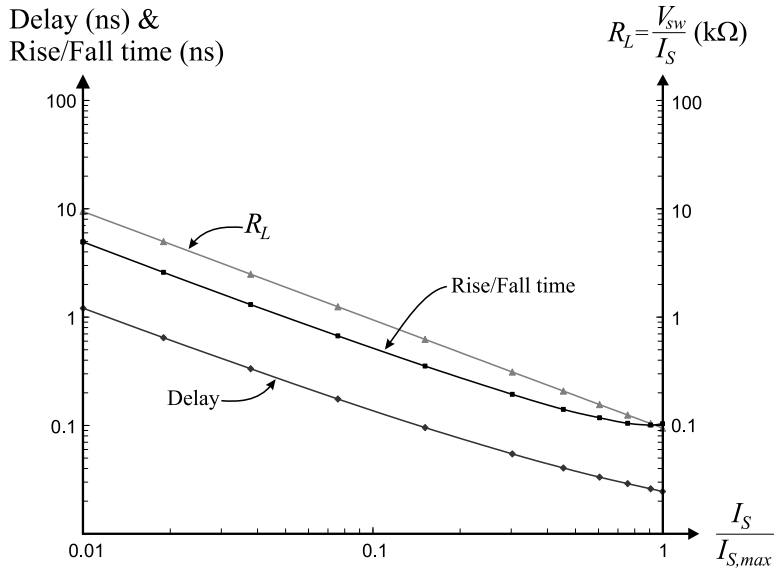


(b) Buffer speed versus input and output swing

Figure 4.6: Buffer $I_{S,max}$ and speed at $I_{S,max}$ as a function of CML digital input and output swing.



(a) V_{sw}=$V_{sw,out}$=500 mV



(b) V_{sw}=$V_{sw,out}$=900 mV

Figure 4.7: Influence of I_S on CML buffer speed.

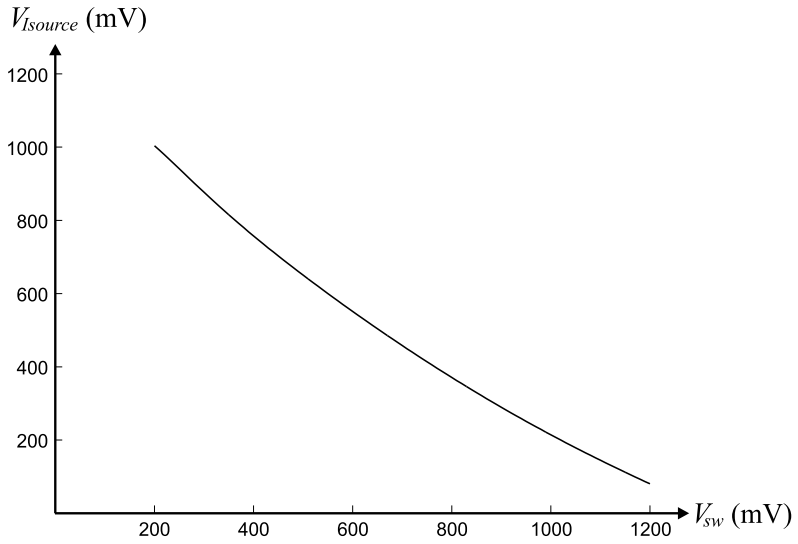


Figure 4.8: Influence of input and output swing on the voltage across the tail current source.

line is not linear with V_{sw} as suggested by (4.7) due to the body effect of the input transistors.

From Figure 4.8 we see that the input voltage swing will in practice be limited by the saturation condition of the tail current source. If we want to have a high current source output resistance, to prevent the delivered current from fluctuating with the input signal of the buffer or the power supply, its length should be chosen larger than minimal. This means that a considerable overdrive voltage is needed for the current source, which also helps in minimizing the current noise. Because a high overdrive leads to a high saturation voltage, this will limit the CML voltage swing.

4.4 2-input CML gate

The 2-input CML gate, such as illustrated in Figure 4.2(a), will generally be a slower circuit than the simple CML buffer. This is mainly because the maximum tail current $I_{S,max}$ will be lower due to the bottom differential pair operating in triode (assuming equal voltage levels are used to control this bottom pair). Also, when switching the bottom pair, the parasitic capacitance associated with the sources of the top pair needs to be charged or discharged, causing extra delay.

The 2-input gate is slightly more difficult to dimension properly because there is one more degree of freedom: the ratio of the widths of the top and the bottom differential pair[4]. For

[4]Of course, there is also the lengths of the input pairs, but for high speed operation, these should be minimal.

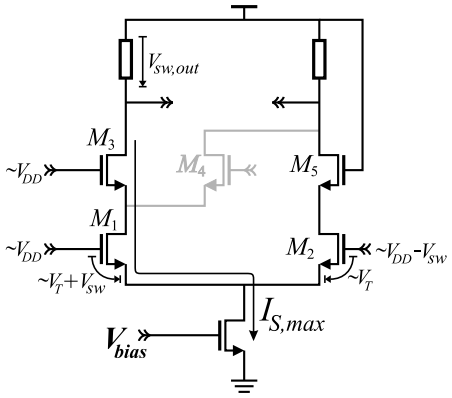


Figure 4.9: Gate operating at maximum tail current.

simplicity, we will assume first that these pairs are dimensioned having equal widths. Later it will be shown that this is as good as optimal.

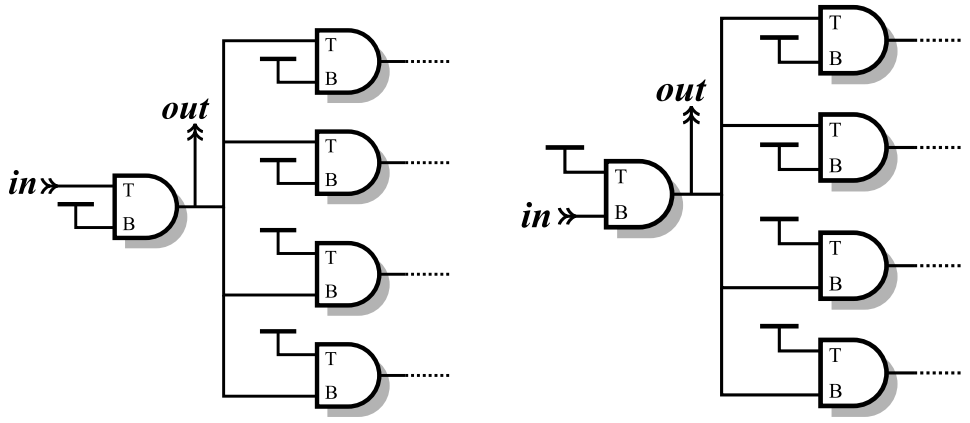
4.4.1 Maximum Tail Current

As was the case for the CML buffer, there will be a certain maximum value $I_{S,max}$ for the tail current, for given dimensions of the differential pairs and the voltage swing V_{sw}. As was stated before, the bottom differential pair will in general be operating in triode. This results in a lower maximum tail current than was the case when using a CML buffer with similar dimensions and voltage swing.

Figure 4.9 illustrates the situation of the 2-input gate operating at the maximum tail current $I_{S,max}$. Increasing the tail current would result in transistor M_2 conducting part of the current, meaning incomplete switching. The tail current at which this happens can be determined in a similar manner as described for the simple CML buffer, considering the overdrive voltage V_{GT} of M_1 equals V_{sw} in Figure 4.9. Plotting the drain currents of M_3 and M_1 connected in series versus V_{GT} results in a plot of $I_{S,max}$ as a function of V_{sw}.

Because M_1 and M_3 are in series and carry the same drain current in Figure 4.9, we can gain some insight by considering those two transistors as one composite transistor with a length that is double the minimum transistor length. This double length means that the maximum tail current will be roughly a factor two lower than the maximum tail current of a CML buffer (assuming equal transistor widths). Half the tail current means that R_L should be doubled for a given output swing, resulting in a speed that is only half that of the CML buffer.

Note that an easier way to determine the maximum tail current is by using simulations, increasing the tail current until the condition of full switching is no longer satisfied. However, knowing that $I_{S,max}$ can be found by plotting drain current versus overdrive voltage, can give insight in the optimization of a CML gate.



(a) Controlled on top pair **(b)** Controlled on bottom pair

Figure 4.10: Simulation configuration concept: Gate controlled on top (T) and on bottom (B) input pair.

4.4.2 CMOS Simulation Results of 2-input CML Gate

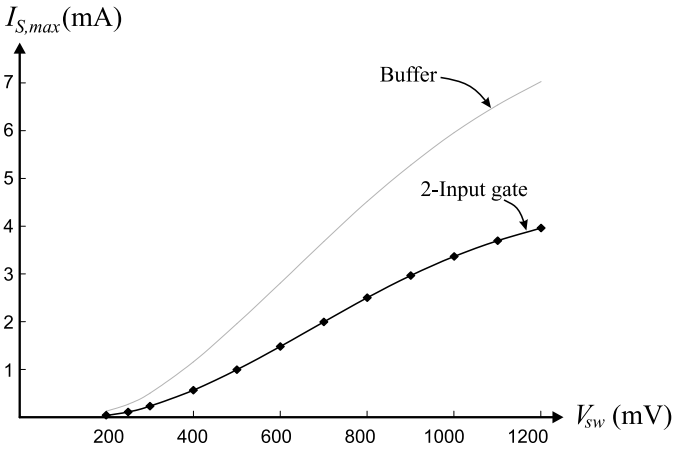
Using simulations in a standard 0.18μm CMOS process, previous statements about the maximum tail current will now be verified. Also, the extra degree of freedom, the ratio of transistor widths of the input pairs, is examined.

Because both the inputs of the 2-input gate will have different delays toward the gate's output, simulations were performed controlling the top input pair and controlling the bottom pair. All gates were loaded with four identical gates. This concept is illustrated in Figure 4.10.

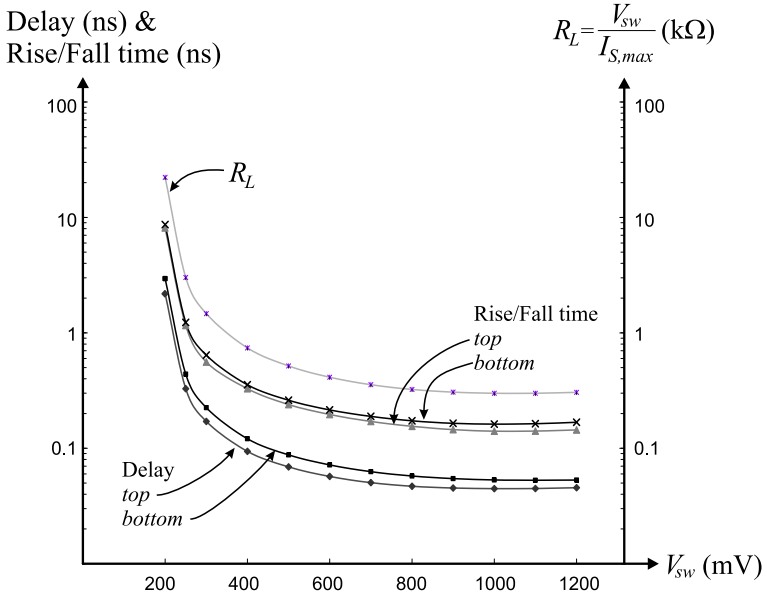
Figure 4.11(a) shows the simulated maximum tail current of a 2-input gate, above which the condition of full switching is no longer satisfied, as a function of the input (and output) swing V_{sw}. The widths of all differential pair transistors is 20μm. We see that, as explained before, the maximum tail current of a 2-input gate is roughly a factor two lower than that of a simple buffer, for the same input and output swing.

The influence this has on the speed can be seen by comparing Figure 4.11(b), showing the speed of the 2-input gate, with Figure 4.6(b) the corresponding figure for the CML buffer. It can be seen that the 2-input gate delay is about twice that of the simple buffer, when controlled at the top differential pair. Controlling the gate at the bottom pair even gives about 25% more delay. This is due to the extra delay taken by charging and discharging the parasitic capacitance at the sources of the top input pair.

Figure 4.12 shows the speed penalty resulting from choosing the tail current too low. This graph shows that the gate delay is inversely proportional to the value of the tail current I_S. Although the speed is maximized by choosing $I_S = I_{S,max}$, this is not wise with respect to robust operation of the gate, as mentioned before. Offset in the differential pair and possible process spread might cause the differential pair to switch incompletely, thus degrading its

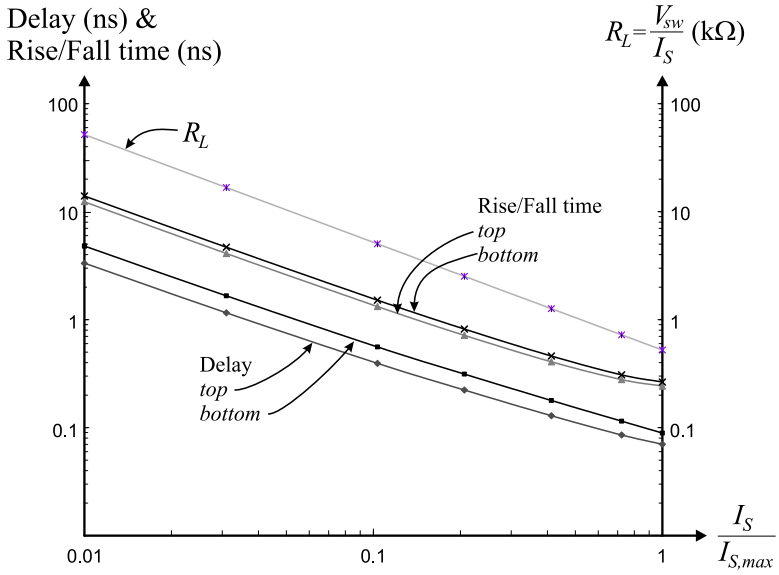


(a) Maximum tail current versus input swing (W/L input pair is 20/0.18)

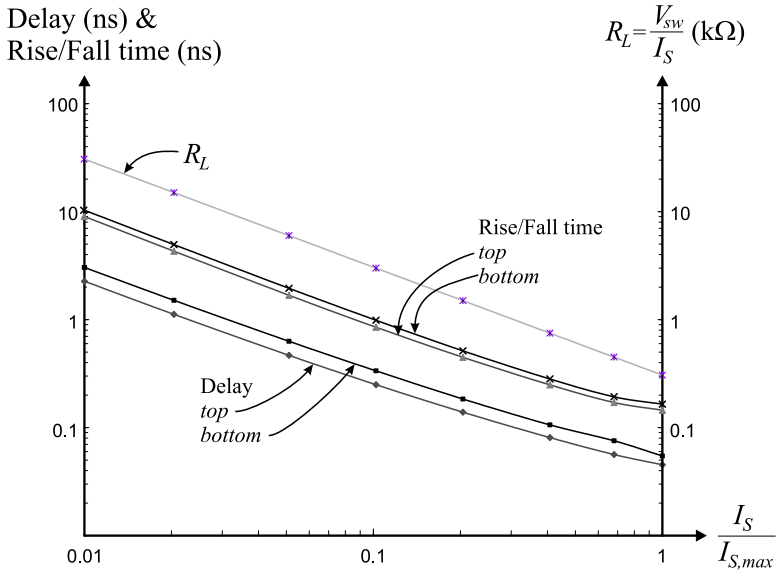


(b) 2-Input gate speed versus input swing

Figure 4.11: 2-Input gate $I_{S,max}$ and speed as a function of CML digital input swing.

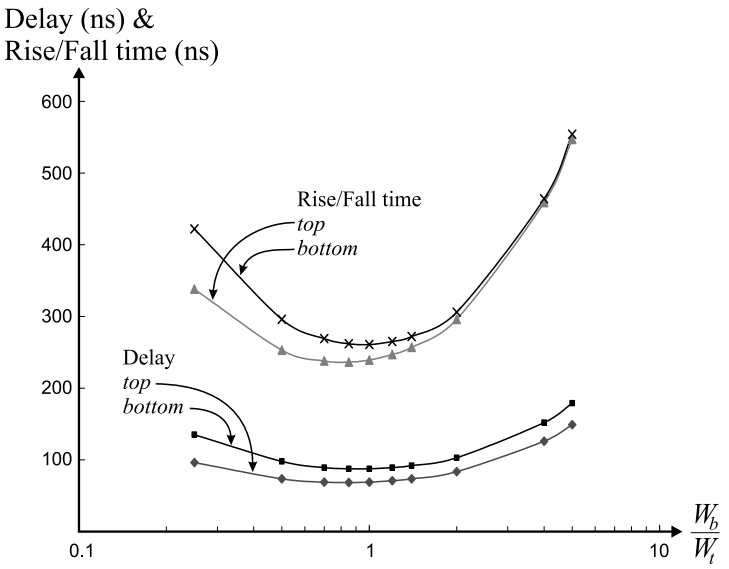


(a) V_{sw}=$V_{sw,out}$=500 mV

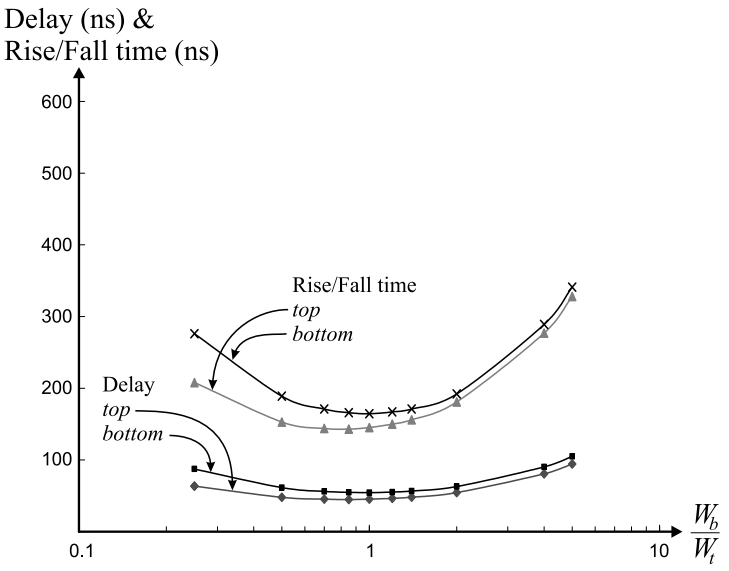


(b) V_{sw}=$V_{sw,out}$=900 mV

Figure 4.12: Influence of I_S on CML 2-input gate speed.



(a) $V_{sw}=V_{sw,out}$=500 mV



(b) $V_{sw}=V_{sw,out}$=900 mV

Figure 4.13: Influence of differential pair width ratio on CML 2-input gate speed.

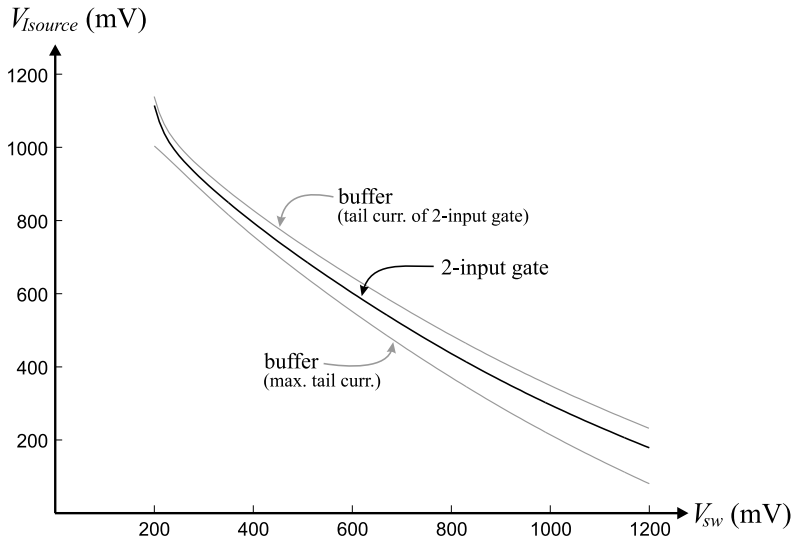


Figure 4.14: Influence of input and output swing on the voltage across the tail current source.

output swing.

Simulations were performed to establish the optimal ratio of the top and bottom pair transistor widths. Results are shown in Figure 4.13, where the delay and rise/fall time of the gate are plotted versus the ratio of the bottom pair transistor widths W_b and the top pair transistor widths W_t. From this figure we can conclude that the optimum ratio is in fact somewhat smaller than one (meaning that the bottom input pair should be dimensioned with a slightly smaller width than the top pair). However, the optimum is very flat. In fact, the optimum is only about 1% faster than the situation where $W_b = W_t$. Furthermore, the exact location of the optimum was shown by simulations to depend on the fan-out of the gate. Thus, it is safe to say that all transistors of the differential pairs should be given equal widths.

Finally, the voltage across the gate's tail current source was simulated, using the maximum allowable tail current $I_{S,max}$. The results are plotted in Figure 4.14. At first glance, it may be surprising to see that the voltage across the tail current source is higher than when using a simple CML buffer at its maximum tail current, while the 2-input gate has stacked input transistors. This, however, is due to the fact that the maximum tail current of the buffer is about twice as high as that of the 2-input gate. Figure 4.14 also shows that a buffer operating with the same tail current as a 2-input gate does in fact have more voltage room for the tail current source, as expected.

4.5 Summary of Conclusions

Current Mode Logic is often used in mixed signal designs, due to both the low generation of and the insensitivity to supply noise and substrate bounce. This chapter demonstrated a simple method of dimensioning the elements of CML gates, based upon determination of the maximum switchable tail current. Using W-scaling, this approach can also be used to determine the minimum transistor widths of the input pairs that fully switch a certain tail current. It was shown that, contrary to common belief, using large signal swings improves the speed of CML circuits. However, the maximum swing is in practice limited by the voltage overhead needed by the tail current source.

Using the concept of maximum tail current, it was explained and demonstrated that a 2-input CML gate can operate at a maximum frequency that is half that of a simple CML buffer. Simulations showed that controlling the bottom differential pair results in a gate delay 25% higher than when controlling the top differential pair. Finally, it was shown that the two layers of differential pairs can be dimensioned equally. Although this theoretically is not optimal, the optimum is extremely flat and the speed penalty of this simple rule-of-thumb is negligible.

5

High-speed Phase Detection

This chapter describes Phase Detector circuits that are eligible for use in a high-speed clock multiplier. A comparison between the structures is made, based on high-speed and low-jitter behavior. A new fast and simple Phase Detector circuit is proposed that consists of just two AND gates and is able to control a Charge Pump circuit.

5.1 Introduction

In Chapter 3 it was shown that a low-jitter PLL-based clock multiplier design benefits from a high reference frequency. Firstly, this is because a high reference frequency allows for low PLL in-band phase noise. Secondly, high reference frequencies allow a high PLL bandwidth to be used, which is sometimes needed to clean up the VCO phase noise. An example of this is described in [42], where the trade-off between step-size and phase noise is decoupled using a dual-loop PLL architecture.

This high-speed demand, however, has some consequences for the choice of the Phase Detector. This chapter starts by reviewing the most well-known Phase Detectors in section 5.2. Their most important advantages and disadvantages are discussed. Section 5.3 discusses the operation speed limitation of the popular tri-state Phase-Frequency Detector with respect to frequency discrimination. A possible solution is offered in section 5.3.1, based on a distinction between the capture-phase and the lock-state, in which low-jitter performance is important. However, the speed limitation is not restricted to the frequency discrimination ability of the PFD. The phase detection mechanism also fails when the reference frequency exceeds a certain limit. A Phase Detector circuit is shown in section 5.4, that overcomes this speed problem, while still producing pulses similar to those generated by the PFD. This is demonstrated using both high- and transistor-level simulations in section 5.5. A possible speed improvement, resulting from combining the Phase Detector and the Charge Pump circuits, is discussed briefly in Section 5.6. Section 5.7 finalizes by summarizing the most important conclusions.

5.2 Traditional Phase Detectors

This section discusses the most popular state-of-the-art phase-detectors, focussing on high-speed and low-jitter operation in a PLL.

5.2.1 Mixer

A very simple and well-known Phase Detector circuit is the mixer (multiplier) circuit, usually implemented using a double-balanced Gilbert cell. If the frequencies of both inputs are equal, the baseband part of the mixer output for sinusoidal input signals is a function of the phase difference at the input:

$$V_{mix} \propto A_1 A_2 \cos\phi_e \tag{5.1}$$

(see Figure 5.1), with A_1 and A_2 the amplitudes of the input signals, and ϕ_e the phase difference between the input signals. The transfer equation immediately reveals the most important

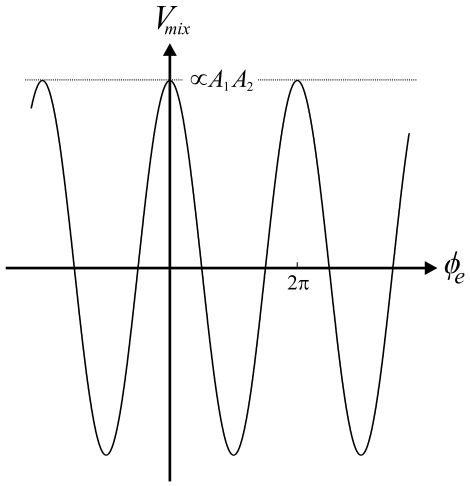


Figure 5.1: Transfer of mixer PD

shortcomings of using a mixer as Phase Detector.

Firstly, the gain of this detector (the derivative of the output voltage to the phase error) is proportional to the amplitudes of both input signals. This means that the detector is not only sensitive to phase difference, but also to amplitude changes. This also means that not only the phase noise, but also the amplitude noise of the signals is converted to PLL output phase noise, via modulation of the VCO control voltage.

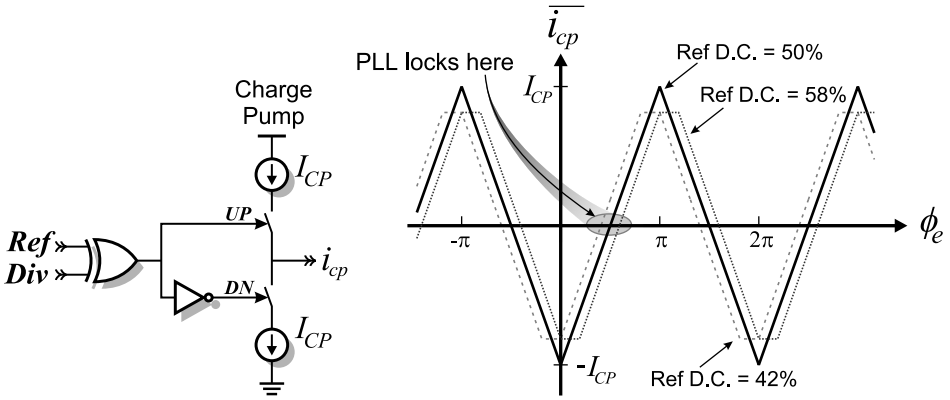
Secondly, the gain of the detector also depends on the static phase error at the PD input, due to the cosine shape of the transfer. According to Figure 5.1, this static phase error depends on the VCO control voltage (which is equal to V_{mix} using a simple first-order loop filter), and, as a result, on the VCO frequency. Finally, the gain of the mixer PD depends on the shape of the input signals.

This uncertainty of the mixer PD gain is an important disadvantage of the mixer PD, as this makes the PLL behavior variable. This makes optimization of the PLL bandwidth a very difficult task.

5.2.2 Exclusive OR Gate

A well-known alternative to the mixer PD is the exclusive OR (XOR) gate. In fact, if the input signals of the mixer are large, the mixer will behave as an XOR gate, and it should come as no surprise that the physical implementation of a CML XOR is a Gilbert cell mixer.

Using an XOR solves the problem of the gain uncertainty of the mixer PD. The gain is not dependent on the input amplitudes, nor on the static phase error. However, a disadvantage of the XOR Phase Detector is that its transfer function depends on the duty cycle of the



(a) XOR plus CP **(b)** Influence of Duty Cycle on PD/CP transfer

Figure 5.2: XOR controlling a Charge Pump.

input signals. This is illustrated in Figure 5.2, where the XOR Phase Detector controls a Charge Pump circuit to eliminate the dependence of the static phase error on the VCO control voltage. Figure 5.2(b) shows that the static PLL phase error depends on the Duty Cycle of the reference clock *Ref*. It also depends on the Duty Cycle of the divider output signal *Div* (not shown). This Duty Cycle dependence can especially be troublesome if the frequency divider generates an output signal with a duty cycle significantly different from 50%, such as usually is the case when using a programmable divider.

The unpredictable static phase error may be a problem when using a frequency detector in the loop, needed due to the lack of frequency discrimination of the XOR itself. To prevent disturbance of the loop because of frequency detector activity, switching off of the frequency detector after achieving phase lock is desirable. With an unpredictable static phase error, however, reliable lock detection is hard to realize.

Finally, the unpredictable static phase error can be a problem when controlling a data multiplexer in a robust manner, as the phase relation between the incoming data and the multiplexer clocking is unknown.

The fact that the static phase error is not 0° results in ripple on the VCO control voltage causing high reference breakthrough, with a strong component at twice the reference frequency. If, however, the duty cycles of the input signals differ from 50%, there will also be a component at the reference frequency.

5.2.3 Edge-triggered SR-flip-flop

The XOR Phase Detector shows duty-cycle dependent behavior, which is due to the fact that the PD is level-sensitive. An edge-triggered Phase Detector on the other hand, would solve

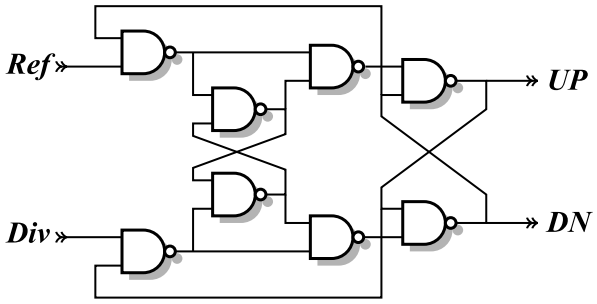
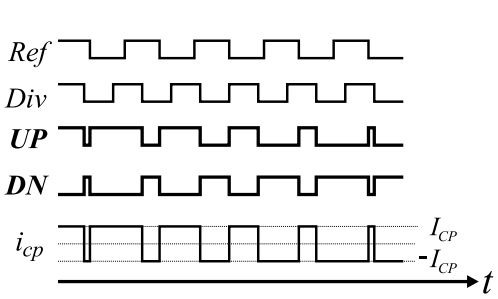
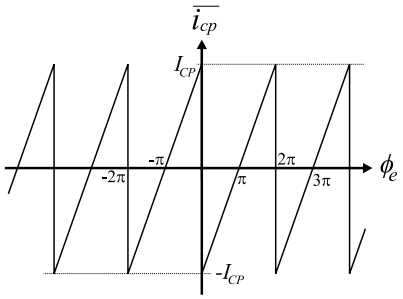


Figure 5.3: Edge-triggered Set-Reset flip-flop implementation.



(a) Time-domain behavior of the Edge-Triggered SR-FF PD



(b) Transfer of SR-FF PD combined with a CP

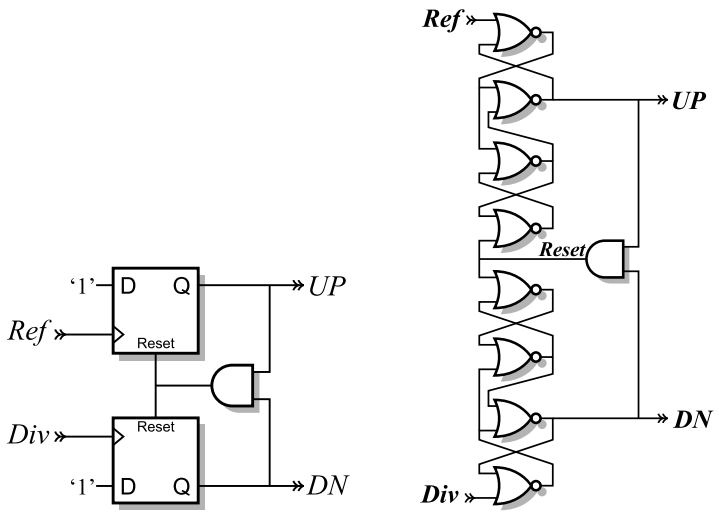
Figure 5.4: Behavior of the Edge-Triggered SR-FF Phase Detector.

this problem by responding only to the rising (or falling) edges of the input signals. Because the phase of the input signals is in fact only well-defined in either the rising or falling edges, it would conceptually be better to use a Phase Detector that only responds to those edges.

The edge-triggered Set-Reset flip-flop [29], also called edge-triggered JK master/slave flip-flop [87], can be used as a Phase Detector operating only on the falling edges of the input signals. The digital implementation of this PD is shown in Figure 5.3 [87]. Note that because this PD responds to falling edges only, the signal shape and duty cycle of the input signals do not influence the PD transfer.

Figure 5.4(a) illustrates the time-domain behavior of the PD. The *UP* signal goes high after a falling edge of the reference signal, the *DN* signal goes high after a falling edge of the divider output signal.

The edge-triggered SR Phase Detector will lock to 180° static phase error when driving a CP. The CP will generate a current with no DC value in lock, but a high harmonic content at the reference frequency, causing high reference breakthrough. Also, the fact that at every moment in time one of the CP current sources is on, will introduce a high phase noise component at



(a) PFD DFF implementation **(b)** PFD gate implementation

Figure 5.5: PFD implementation.

the PLL output due to the continuous noise injection of the CP current sources.

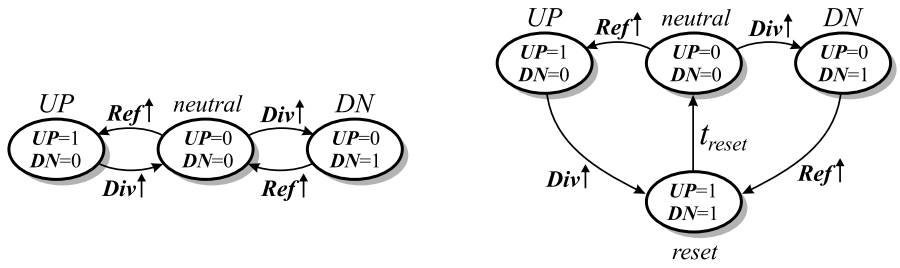
Although for extreme frequency differences between the input signals this PD shows frequency detection behavior [87], this PD does not detect small frequency differences (in the order of ten percent). A PLL employing a SR Phase Detector would, therefore, need a frequency detector to guarantee correct locking.

5.2.4 Tri-State Phase-Frequency Detector

All Phase Detectors mentioned before have a common drawback: there is no inherent means of frequency detection, which limits the capture range of the PLL. The tri-state Phase Frequency Detector [33] (PFD) solves this problem by offering both phase error as well as frequency error detection. Figure 5.5(a) shows the well known D-flip-flop implementation of this very popular PD. Figure 5.5(b) shows a fast implementation in digital gates, using four NOR-gates to implement each D-flip-flop [29].

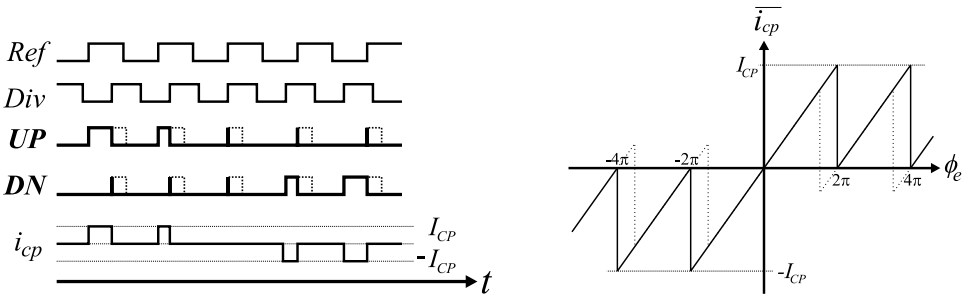
The operation of the PFD is best explained using a state diagram, as depicted in Figure 5.6. The state diagram of a PFD that uses ideal (infinitely fast) components is shown in Figure 5.6(a). The PFD enters the '*UP*' or '*DN*' state depending on which input signal (*Ref* or *Div*) leads the other. This leads to the generation of an *UP* or *DN* pulse, until the other input signal experiences a rising edge, resulting in the PFD returning to its '*neutral*' state, in which both *UP* and *DN* signals are inactive.

In a practical PFD, returning to the neutral state does not happen immediately after the second



(a) State diagram of infinitely fast PFD **(b)** State diagram of real PFD

Figure 5.6: PFD state diagram.



(a) Time-domain behavior of the PFD close to lock **(b)** Transfer of PFD combined with a CP

Figure 5.7: Behavior of the PFD.

rising edge, as shown in Figure 5.6(b), due to the reset-path delay. Instead, the PFD enters a fourth state (the '*reset*' state), during which both the *UP* and *DN* signals are asserted. After a time t_{reset} the PFD returns to the neutral state. Note that this '*reset*' state is in fact a necessity to avoid the infamous dead-zone problem (where the gain of the Phase Detector drops for small values of ϕ_e) [29].

Figure 5.7(a) shows the time-domain behavior of the PFD close to lock. Figure 5.7(b) shows the transfer of the PFD as a function of input phase error. The dashed lines in these figures apply to high-frequency input signals, where the reset-path delay is a significant portion of the reference period time [88].

The PFD shares one important property with the previously mentioned SR-FF Phase Detector: it responds to transitions of the input signals (the rising edges in this case). This means that the PFD is not sensitive to the duty cycle of the input signals.

Apart from the frequency detection the PFD offers, it solves another drawback of previously mentioned Phase Detectors. When the PFD is used to control a Charge Pump, which is almost invariably the case, the resulting *UP* and *DN* currents will cancel when the PLL is in lock. This means that the disturbance on the VCO control line is small, resulting in low

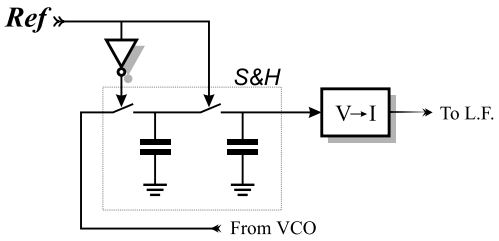


Figure 5.8: Conceptual schematic of S&H Phase Detector

reference-breakthrough.

Because usually the reset-delay path delay t_{reset} is a relatively small fraction of the reference period, the duty cycle of the CP current sources will be small. As was already shown before in (3.35), the CP noise contribution is proportional to this duty cycle. This means that usually the PLL jitter contribution of the CP controlled by a PFD will be less than when using any of the PDs described before.

In some cases, the fact that the PLL locks to a static phase error of 0° might be a reason to use the PFD [89], although probably not in the case of a CMU, because the delay of the frequency divider causes the VCO phase not to exactly align with the reference signal [45].

5.2.5 Sample&Hold PD

A PD that is especially suitable for high-speed operation is the Sample&Hold Phase Detector [90,91]. The Phase Detectors described thus far (with the exception of the mixer) all generate digital *UP* and *DN* signals, of which the duration is a measure of the phase difference. This duration is a fraction of the reference period. The Sample&Hold (S&H) detector, on the other hand, generates an analog signal that is proportional to the phase difference. Because this PD is not based on duty-cycle modulation of a digital-valued signal, it is suitable for high operation speeds.

Figure 5.8 conceptually shows the S&H PD. Using two cascaded Track&Hold circuits (shown here in their most basic form: a switch and a hold capacitor), the analog value of the VCO output signal or divider signal is sampled by rising edges of the reference. Because around the VCO or divider zero-crossings, this analog value is proportional to the phase error ϕ_e, the S&H effectively measures ϕ_e.

In Clock Recovery PLLs, the S&H PD may be a good alternative to digital detectors, as Clock Recovery Phase Detectors usually run at the VCO frequency. The S&H PD has, however, several disadvantages, making this Phase Detector unpopular in low-jitter frequency multiplying PLLs, where the PD runs at a fraction of the VCO frequency.

Firstly, like most of the before mentioned Phase Detectors, the S&H circuit does not provide frequency error information, although in [90] an efficient addition to the S&H circuit is

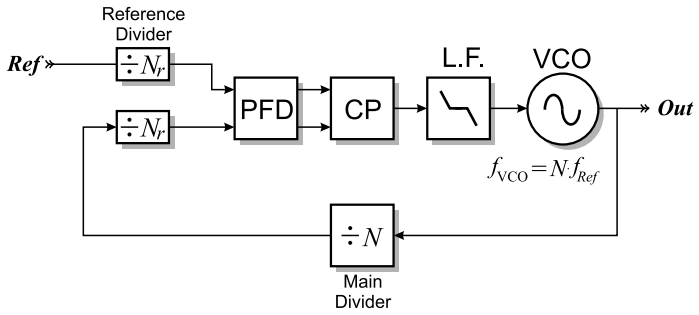


Figure 5.9: PLL with Reference Divider.

shown, that does provide frequency detection, needing *I*&*Q* VCO or divider signals.

A second difficulty with the S&H Phase Detector arises when it is used in a frequency multiplying PLL. In this case, the S&H circuit will sample the divider output. In [91], where the VCO signal is sampled, the PD gain is reasonably well defined due to the sinusoidal nature of the VCO signal, whose amplitude and frequency are within certain well-defined limits in practice. The divider output signal, however, usually has a slope that is ill-defined, making the PD gain uncertain.

Finally, in some cases, the possibility exists that the VCO will lock to a harmonic of the reference clock. This, of course, is only possible if the tuning range of the VCO is wider than one octave, *e.g.* when a ring oscillator is used.

5.3 PFD speed limitation

Concluding from the previous summary of Phase Detectors, the tri-state Phase Frequency Detector has some important advantages that make it a very popular. These include the PFD's frequency discrimination ability and the generation of relatively short and mostly simultaneous *UP* and *DN* pulses, leading to low CP noise injection and low reference breakthrough.

The reset-path delay of the PFD is of concern, however. As demonstrated in [88], a PFD has an upper limit for the reference frequency at which frequency discrimination is possible. This upper limit is directly linked to the reset time of the PFD by:

$$f_{max} = \frac{1}{2\,t_{reset}} \tag{5.2}$$

where t_{reset} is the reset time of the PFD (see Figure 5.6(b)). This time includes the delay of the AND-gate and the propagation time inside the flip-flops. Starting up of the PLL, with a higher reference frequency than f_{max} can lead to permanent frequency lock at wrong operation frequencies.

An obvious work-around for this speed limitation is to keep the PFD input frequency below

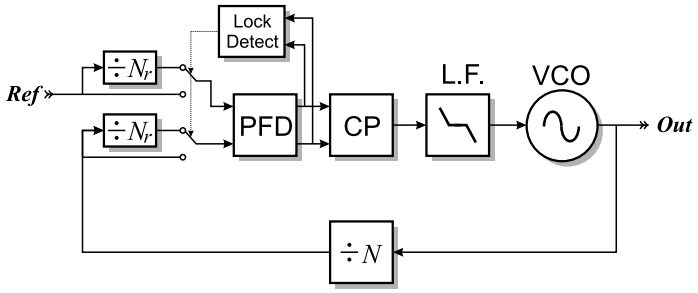


Figure 5.10: Switched Reference Divider.

f_{max} by preceding the PLL with a frequency divider (the *reference divider*) in order to divide the reference frequency down by a certain factor N_r, as shown in Figure 5.9. Note that the divider ratio of the frequency divider inside the loop (the *main divider*) needs to be multiplied by the same factor N_r to maintain a certain output frequency. This approach has some disadvantages, though.

Firstly, decreasing the comparison frequency lowers the maximum PLL open-loop bandwidth, which is, according to a rule-of-thumb approximation, ten times lower than the comparison frequency [40]. This may result in a PLL that cannot be dimensioned optimally with respect to noise, especially if a low-Q oscillator is used, such as a ring oscillator.

More importantly, however, lowering the comparison frequency raises the PLL in-band phase noise, as described in section 3.5. This means that the PLL output jitter increases due to the divider addition. Also, the optimal loop bandwidth decreases, which, for a noise-optimized loop means an increased loop filter capacitor size. This jitter penalty is an important motivation to seek an alternative solution for the PFD speed limitation.

5.3.1 Switched Reference Divider

Remembering that the extra dividers were only introduced to enable frequency discrimination of the PFD before the PLL has achieved phase-lock, there is a simple solution [92] to avoid the drawbacks of using a reference divider, while still being able to achieve frequency lock with the PFD. This is shown schematically in Figure 5.10.

The principle of operation is as follows: some time after phase (and frequency) lock is achieved (a condition that is confirmed by the lock detect circuit), the two extra dividers (which were added to decrease the comparison frequency) are removed from the loop using the switches. This enables an increase in the loop bandwidth and a decrease in the divider ratio of the main divider by a factor N_r. After phase-lock is reached, the only function of the PFD is to maintain the phase-lock, without the necessity for frequency discrimination. This means that the reset time of the PFD is allowed to be somewhat higher than half the period time of the reference signal. Therefore, the maximum operation frequency of a given PFD is extended with the proposed technique.

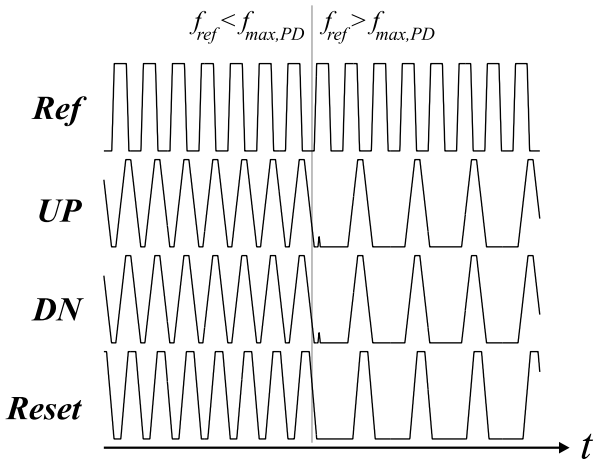


Figure 5.11: Speed limitation of PFD circuit.

The advantage of effectively removing the extra dividers when in lock is that the close-in phase noise power density drops, as discussed in section 3.5. Removing the extra dividers also enables an increase of the loop bandwidth resulting in a more effective suppression of the VCO jitter as compared to the situation when the divider N_r is in the loop.

Note that the removing of the extra dividers "on-the-run" must not cause phase disturbances in the loop, as that might force the loop out of lock. The timing of the switches is easily linked to the divider states, so that switching can be timed to occur just after the active divider edges reach the PFD. Switching noise on the PFD inputs can be prevented by careful circuit design.

Note that the maximum theoretical speed-gain that can be obtained using this 'switched reference divider' technique is a factor of two. This is because according to (5.2), the maximum operation frequency is achieved when the reset-path delay is half the reference period. This means that the PFD will in any case fail to detect phase differences at an input frequency that is twice f_{max}, as the reset-delay path delay would then equal the reference period and the duty cycle of the generated pulses would be 100%, leaving no room for modulating the phase error information on these pulses.

5.3.2 PFD phase detection speed limitation

As discussed in the previous section, not only the PFD's frequency discrimination capability is limited by an upper frequency. There also exists a frequency limit for the phase detection itself. This is in fact an even more important limitation in case of a low-jitter PLL; the frequency detection can also be performed by a separate circuit, possibly at a lower speed. The highest frequency for which phase detection is possible determines the upper limit for f_{ref}.

Although it was said before that the absolute maximum frequency would be twice the maximum reference frequency at which frequency detection is possible, it is to be expected that the actual limit is significantly lower. This is because when the *UP* and *DN* signals approach a 100% duty cycle, the time that these signals are *inactive* becomes very short, which generally is just as difficult to achieve as signals being *active* for a short time.

The implementation shown in Figure 5.5(b) was simulated using Simulink, a MATLAB simulation shell, to study the high frequency phase detection behavior of this implementation. The limited output speed of the gates was modelled with a current source charging a capacitor, resulting in a slewing output. Figure 5.11 shows a plot of the PFD output signals for a slowly increasing reference frequency. The simulation was performed with zero phase difference at the PFD input.

This plot shows that when increasing the PFD input frequency above a certain limit, the PFD starts responding to only half the input edges, thus implicitly dividing the input frequencies by two. This means that, although the reference frequency is high, the actual comparison frequency is two times lower. This will undo the advantages of increasing the reference frequency. The highest frequency for which the PFD still responds to all rising edges is called $f_{max,PD}$ here, where *PD* refers to the phase detection limit.

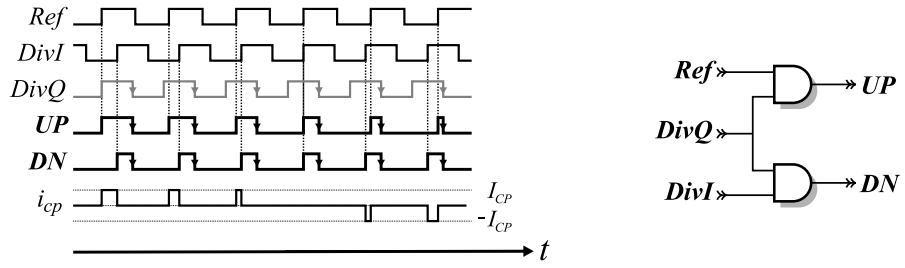
It is interesting to note that for input frequencies just below $f_{max,PD}$, the duty cycle of the *UP* and *DN* pulses is 50%, for the implementation of Figure 5.5(b). This means that the frequency limit for phase detection and that for frequency discrimination are the same for that particular implementation. We conclude that the 'switched reference divider' approach would not offer a solution when trying to increase the PFD operation frequency in this case.

When the desired PLL reference frequency exceeds $f_{max,PD}$, a fundamental change needs to be made on the circuit level in order to increase the comparison frequency of the Phase Detector, while still maintaining the concurring *UP* and *DN* pulses. A solution to this will be proposed in the next section.

5.4 A Simple and fast two-AND-gate PD

Having established that the main speed limitation of the PFD is the relatively slow reset path, led to investigate the possibility of removing the PFD reset loop altogether. It is, however, desirable to preserve the basic operation of the PFD: the generation of *concurrent UP* and *DN* pulses that, when applied to a Charge Pump, deliver linear phase error information.

Observing the basic functionality of the PFD building block reveals that the reset signal is needed to put the detector in its '*neutral*' state. In the case of phase lock, this signal will always go high a small amount of time after the rising edge of both PFD input signals. When using a static divide-by-two frequency divider in the PLL, which inherently generates *I&Q* signals, we see that the quadrature divider signal *DivQ* has a transition some time after the rising edge of the in-phase divider signal *DivI* (see Figure 5.12(a)). To avoid generation of the reset signal in a feedback loop, the possibility of using the quadrature divider signal as a reset signal was examined.



(a) Desired PD Response **(b)** Circuit implementation

Figure 5.12: PD Response and Implementation.

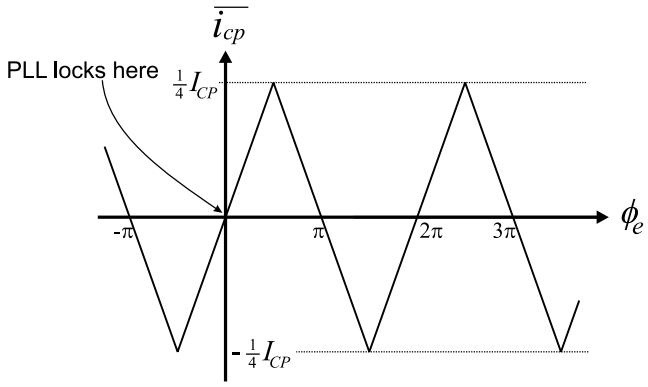


Figure 5.13: Mean CP current as a function of the phase error.

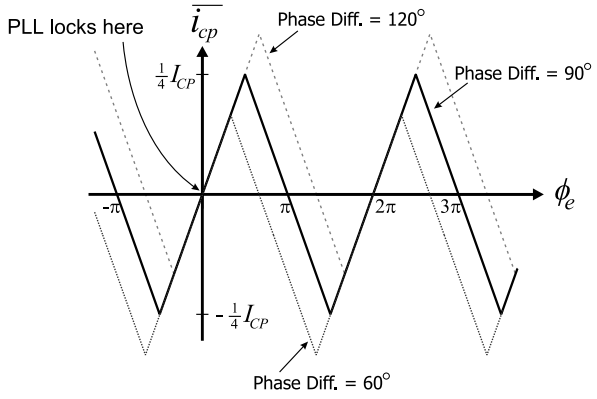
We can derive a set of simple rules for a fast PD able to control a Charge Pump, using the quadrature divider output signal as reset signal:

1. The *UP* signal will go high after a rising reference edge (*Ref*).
2. The *DN* signal will go high after a rising in-phase divider edge (*DivI*).
3. Both *UP* and *DN* signals will be reset after the next quadrature divider output edge (*DivQ*).

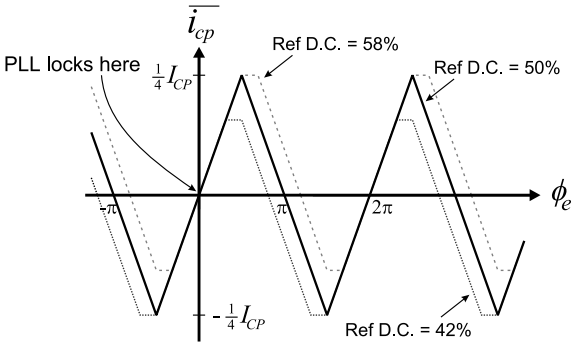
The effect of these rules are shown graphically in Figure 5.12(a), for the case that the PLL is close to lock. Observing the PD input and output signals, we conclude that the simplest implementation of this behavior can be described as:

UP = *Ref* AND *DivQ*
DN = *DivI* AND *DivQ*

with 'AND' being the boolean And operation. The Phase Detector circuit implementing this behavior is shown in Figure 5.12(b) [93].



(a) Imperfect quadrature



(b) Reference Duty Cycle

Figure 5.14: Influence of imperfect quadrature and reference duty cycle on PD/CP response.

Note that the width of the *UP* pulse responds *linearly* to the time overlap of *DivQ* and *Ref*. The width of the *DN* pulse depends on the time-overlap of *DivQ* and *DivI*. In lock, with coinciding rising edges of *Ref* and *DivI*, the *UP* pulse has the same width as the *DN* pulse.

When the *UP* and *DN* signals are supplied to a Charge Pump, the net charge pumped in the PLL loop filter is linearly dependent on the phase difference detected, for phase differences close to zero degrees. This can be concluded from Figure 5.13, where the mean Charge Pump current is plotted as a function of the PLL input phase difference. Note that this graph applies to infinitely fast AND-gates and an ideal CP transfer. Figure 5.13 shows that the gain of the PD/CP combination is $\frac{I_{CP}}{2\pi}$, where I_{CP} is the maximum Charge Pump current. Because of the integrating action of the CP and the loop filter, the PLL will lock to a phase error of 0 degrees, as indicated in Figure 5.13. Because the *UP* and *DN* pulses and the corresponding Charge Pump currents will cancel in lock, the reference breakthrough will be small resulting in low

spurious peaks, similar to the case in which a conventional PFD is used.

In Figure 5.14(a), the dependence of the PD/CP response on the 90° quadrature quality is shown. Although the range in which the PD stays linear is affected by the phase angle between *DivI* and *DivQ*, the PD gain is unaffected, as is the locking point of the PLL. The linear region ranges from $\Delta\phi_q - \pi < \phi_e < \Delta\phi_q$, where $\Delta\phi_q$ is the phase difference between *DivI* and *DivQ* and ϕ_e is the phase error of the PLL. In the case of ideal quadrature (when the phase difference between *DivI* and *DivQ* is $\pi/2$), the linear range is between $-\pi/2$ and $\pi/2$.

Figure 5.14(b) shows that the gain and the PLL phase error in lock also do not depend on the Duty Cycle of the reference. The same holds for the dependence on the Duty Cycles of *DivI* and *DivQ* (not plotted here). The linear range is again affected by Duty Cycle deviations. For a reference Duty Cycle lower than 50%, the linear region is between $-\pi/2 < \phi_e < 2\pi\,(D.C. - 1/4)$, with *D.C.* being the reference Duty Cycle. If $D.C. > 50\%$, the linear region is between $-2\pi\,(3/4 - D.C.) < \phi_e < \pi/2$. Note that for correct operation of the PD, the reference Duty Cycle can vary amply between 25% and 75%. In the case where the Duty Cycle is 50% the linear range is between $-\pi/2$ and $\pi/2$.

Note that in [45], a PD is used that at first sight looks similar in design to the proposed PD. However, in [45], three-input AND-gates are used, which are generally slower than two-input AND-gates as used in the proposed PD. More importantly, in that work, the length of the *UP* and *DN* pulses approaches zero when the PLL is in lock. Because neither the output voltages of the AND-gates nor the current sources of the CP are infinitely fast, the structure proposed there will have a dead-zone problem. This means that the gain of the PD/CP combination will drop significantly as the phase difference approaches zero. The PD shown in this section does not suffer from this problem as the *UP* and *DN* signals maintain a Duty Cycle of about 25% when the PLL is locked, owing to the overlap existing between the input signals of the AND-gates.

During the presentation of the proposed PD at the ISSCC 2003 [94], it was discovered that a similar PD structure had been proposed in an M.Sc. thesis [95]. In that work, extra digital circuitry was added to reduce the minimal width of the *UP* and *DN* pulses. At operating frequencies far lower than the maximum operation frequency, this technique could be used in a CMU to reduce unwanted side effects of both Charge Pump current sources being on simultaneously, such as extra noise injection into the loop filter or mismatch-induced reference feed-through. However, at high reference frequencies this technique is not useful, as the rise- and fall times of the output pulses limit the minimum pulse width. Trying to apply this technique to reduce the pulse-width will only lead to increased complexity and power usage.

5.5 Phase Detector Speed Comparison

To assess the value of the proposed two-AND-gate Phase Detector, its maximum operation speed is compared to that of the other digital Phase Detectors described before. This is done by expressing the $f_{max,PD}$ of the different Phase Detectors detectors in terms of the FO-4 delay metric [96].

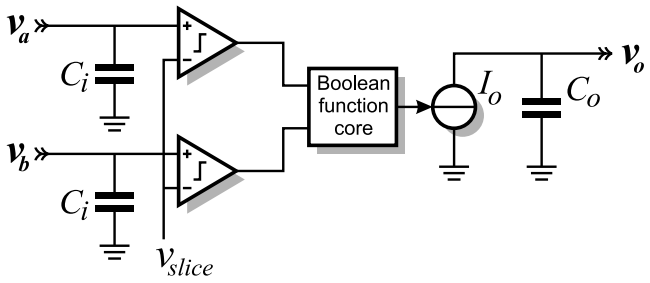


Figure 5.15: Finite-speed Digital Gate Model.

The FO-4 delay (fan-out-of-4 delay) is defined as the delay of an inverter loaded by four identical inverters. Normalizing the speed of a circuit with respect to the FO-4 delay largely removes the process, voltage and temperature dependence. In this way, the speed of digital circuits can be effectively compared.

To be able to predict the maximum operation speed of the different Phase Detector implementations in terms of FO-4 delays, first an abstract digital gate model is introduced. This makes it possible to analyze the circuits irrespective of the logic family used to implement them. Using this model it is also possible to perform simulations on a high level to remove the dependence of the results on the technology that is chosen.

5.5.1 Digital Gate Model

The digital gate model that is used to simulate and analyze the Phase Detectors is shown in Figure 5.15. The gate has an input capacitance C_i that loads the controlling gates and an output capacitance C_o. The comparators determine the logic levels at the gate inputs; v_{slice} is the mean of the digital voltage levels for a '0' and for a '1'. The actual logic function is modelled using an infinitely fast logic core that controls a current source. This current source, combined with C_o, causes a slewing output and, thus, a certain gate delay. Note that the current source is automatically switched off when either one of the extreme output voltages is reached (not shown in Figure 5.15).

The FO-4 delay of this gate model can be expressed as:

$$\tau_{FO-4} = \frac{C_o + 4 \cdot C_i}{2I_o} \tag{5.3}$$

which is invariant to Impedance Level Scaling as C_o, C_i as well as I_o are proportional to the scale factor α.

This digital gate model behaves much like a realistic CMOS Rail-to-Rail or CML gate. The slewing behavior can be seen in both these gates. Some deviations might occur due to the fact that a realistic gate does not perform the input slicing in the abrupt manner modelled in the high-level gate model. Rather, the on- and off-'switching' of the current source is more

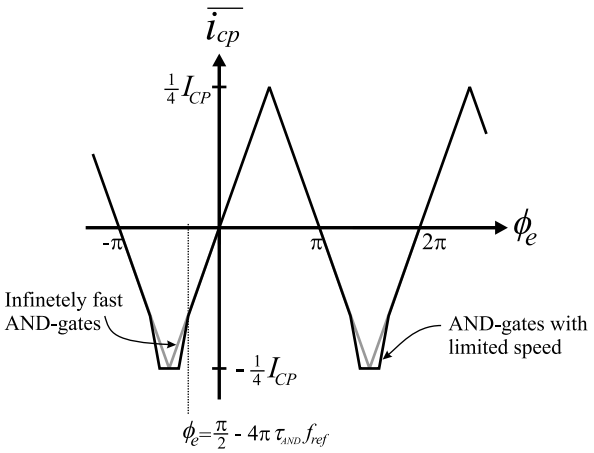


Figure 5.16: 2-AND-gate PD plus CP transfer with digital gate model.

fluent in a realistic gate. Nevertheless, the gate model offers insight in the basic operation of a digital gate. The results obtained by this model will be verified using transistor-level simulations later.

5.5.2 High-Level Speed Comparison

To be able to determine the maximum operation frequency $f_{max,PD}$ of the different Phase Detectors using the digital gate model, we first have to define what is considered 'correct PD operation'. This is clear for the tri-state PFD, as it was shown that this Phase Detector starts to implicitly divide the input frequency by skipping input edges. Simulations show the same phenomenon for the edge-triggered Set-Reset flip-flop.

The criterion for the two-AND-gate Phase Detector can be derived by looking at the PD transfer in case AND-gates with limited speed are used. Plotting the mean Charge Pump output current as a function of the phase error at the detector's input, using the digital gate model, yields a curve as shown in Figure 5.16. We see that when the *UP* pulses become narrow (close to $-\pi/2$ input phase error), the curve deviates from the ideal transfer curve, also shown in Figure 5.16. This is due to the output of the AND-gate not having enough time to completely reach the voltage associated with a digital '1'. As long as the gain in the PLL locking point of zero degrees phase error is unaffected, the PD is said to operate correctly. According to Figure 5.16, the maximum input frequency of the two-AND-gate PD (the reference frequency for which the change in gain would coincide with the origin) is

$$f_{max,PD} = \frac{1}{8\,\tau_{AND}} \tag{5.4}$$

with τ_{AND} the gate delay of the AND-gates.

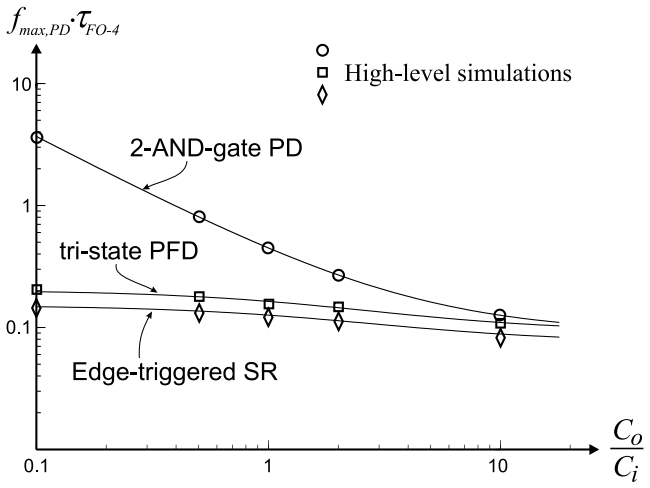


Figure 5.17: Maximum operation frequencies normalized to FO-4 delay.

Such a limitation also holds when using the XOR-Phase Detector: when a predictable PD/CP gain is required, the maximum operation frequency is one-eighth the inverse of the XOR gate delay.

Figure 5.17 shows the maximum operation frequencies (normalized to FO-4 delay) of the digital Phase Detectors that were described before, as a function of the ratio of the output capacitance of the gate C_o and the input capacitance of the gate C_i. The fact that the delay of a two-input gate is about 1.4 times the FO-4 delay [96] is taken into account. The tri-state PFD is realized according to Figure 5.5(b), as it is indicated in [88] that this is the fastest implementation.

From Figure 5.17, it can be seen that if the gate output capacitance is by far dominant over the input capacitance of the following gates, the maximum operation frequency of the two-AND-gate Phase Detector is not significantly higher than that of the PFD. If, indeed, the use of the simple two-AND-gate PD does not offer an operation speed advantage, the duty cycle of the output signals is still half that of the PFD (25% versus 50%). This means that the in-band PLL phase noise due to the Charge Pump noise will be 3dB lower when using the simple PD, according to (3.35). Also, the input referred noise of the two-AND-gate PD due to noise of the PD circuit itself will be lower than that of a PFD circuit, due to the lower number of gates determining the length of the output pulses.

However, simulations in a standard 0.18μm CMOS process show that C_o and C_i are roughly equal. This means that the simple two-AND-gate Phase Detector implementation is able to run at an operation frequency about 2 times higher than the $f_{max,PD}$ of a tri-state PFD.

Apart from the proposed PD's ability to run at very high reference frequencies, the previously mentioned advantages still hold: the short duration of the *UP* and *DN* pulses that lead to a

low CP jitter contribution and the low input referred PD jitter.

5.5.3 CMOS simulation results

To verify the high-level simulation results, predicting a high operation speed for the two-AND-gate PD compared to the tri-state PFD, simulations in a standard 0.18μm CMOS process were performed. The results of those simulations are briefly summarized here.

Again, the tri-state PFD is realized according to Figure 5.5(b), as it is indicated in [88] that this is the fastest implementation. Simulations were performed for both standard CMOS digital gates and for a Current Mode Logic implementation. The two-AND-gate PD is simulated for both these logic families as well.

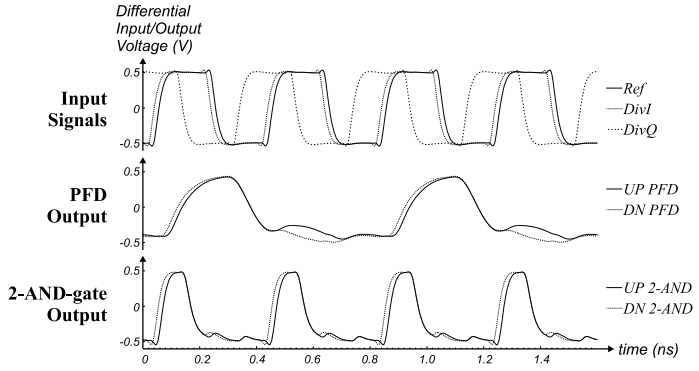
Figures 5.18(a) and 5.18(b) show the simulated response of the CML and standard CMOS implementations of the tri-state PFD and of the two-AND-gate Phase Detector, illustrating the implicit divide-by-two effect of the PFD. The transistor-level simulations confirm the speed improvement that can be gained by employing the simple two-AND-gate Phase Detector.

A short summary of the simulated maximum operation frequencies of the Phase Detectors is given in Table 5.1, showing that the transistor level simulations support the conclusion that the proposed two-AND-gate Phase Detector can operate at about twice the maximum PFD operation frequency. Expressing the speed of the PDs in FO-4 delays yields roughly the same numbers.

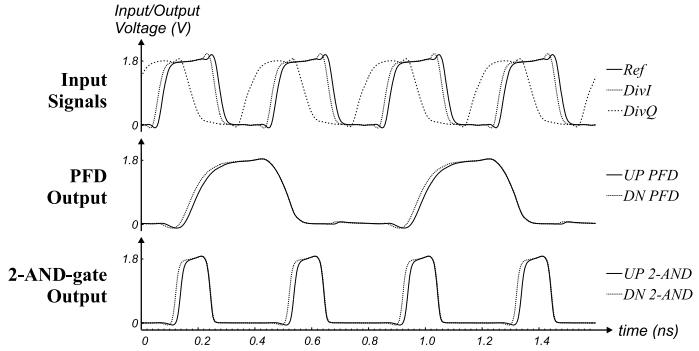
In [97], a very fast implementation for a tri-state PFD is presented. This high-speed operation is due to clever design on the transistor-level implementation (the design basically still consists of two D-flip-flops and an AND-gate). The simulated speed of this circuit (just over 4 GHz) exceeds that given in Table 5.1 for the standard CMOS version of the PFD, and even that of the two-AND-gate PD. However, using the same transistor-level fine tuning technique for the two-AND-gate PD, see Figure 5.19, results in a similar improvement in operation speed (just over 9 GHz).

Logic family	PD type	$f_{max,PD}$ (GHz)	$f_{max,PD}$ ($1/\tau_{FO\text{-}4}$)
CML	tri-state PFD	2.1	0.11
	2-AND-gate	4.4	0.22
Standard CMOS	tri-state PFD	1.6	0.10
	2-AND-gate	3.0	0.19

Table 5.1: Summary of simulated maximum operating frequencies.



(a) Current Mode Logic



(b) Standard CMOS

Figure 5.18: Simulated response of PFD and 2-AND-gate PD at 2.5GHz

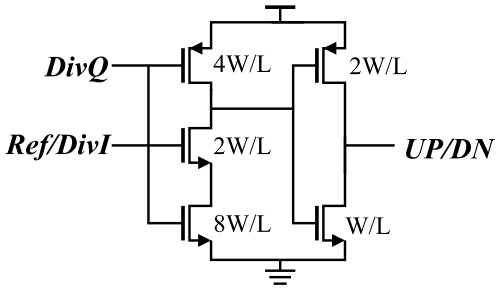


Figure 5.19: Pre-charged AND-gate implementation for two-AND-gate PD.

5.6 Combining PD and CP by mirroring PD output current

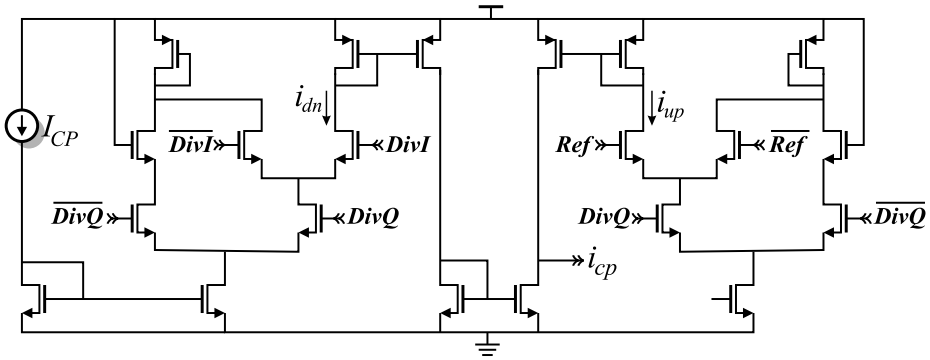
In all previous implementations of the Phase Detector circuits, *UP* and *DN* signals were generated in the voltage domain. For simple Phase Detector circuits, where the *UP* and *DN* pulses are not needed to drive one or more digital gates as part of a reset loop, the conversion to the voltage domain is not fundamentally necessary.

When an XOR or a two-AND-gate PD is employed, it is possible to identify *UP* and *DN* signals in the *current* domain inside the digital PD. Using a simple mirror structure, it is then possible to lead these currents into the PLL loop filter, thus combining the Phase Detector and Charge Pump functionality [45]. Because the *UP* and *DN* signals need not be generated in the voltage domain now, this technique avoids high-speed high-swing internal nodes. Such a technique could prove useful in a high-speed low-power PD/CP design.

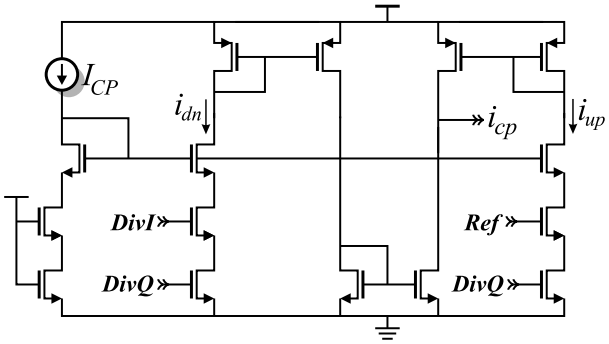
Figure 5.20(a) illustrates the technique in a CML implementation. If *DivQ* and *DivI* are both active, a current I_{CP} is drawn from the loop filter (connected to the output marked i_{cp}). If *DivQ* and *Ref* are both active, a current I_{CP} is pumped into the loop filter. Compared to using CML AND-gates that generate *UP* and *DN* pulses in the voltage domain, the circuit of Figure 5.20(a) is roughly 30% faster. Also, its power usage will be lower, because the input differential pair of the Charge Pump is omitted here.

Figure 5.20(a) shows a similar circuit for use in a standard CMOS digital environment. Note that, comparing to the circuit of Figure 5.19, the stack of NMOS transistors is higher now, in order to generate predictable output currents. This is the reason that the speed improvement of the PD/CP combination is not significant.

A disadvantage of the technique of combining the PD and the CP function is that the design of the PD and that of the CP part are not orthogonal. As can be seen from Figure 5.20, the internal currents of the PD part need to be equal to I_{CP} (assuming mirror-factors of 1). This non-orthogonality may complicate the designer's task. Tuning of the Charge Pump current to adapt the PLL loop bandwidth is also complicated when using this technique.



(a) Current Mode Logic (Concept)



(b) Standard CMOS (Concept)

Figure 5.20: Concept of combining the two AND-gate PD and the CP function.

5.7 Summary of Conclusions

Comparing different Phase Detectors, we concluded that digital Phase Detectors were to be preferred over Phase Detectors with an analog output in a clock multiplying PLL, due to the predictable value of K_{PD}. The output signals of a tri-state PFD are particularly convenient because of the low net Charge Pump activity in lock and the predictable PLL static phase error. The inherent feedback loop present in this type of PD was shown to cause a maximum operation frequency for both the frequency discrimination as well as the phase detection mechanism. It was shown that a simple and fast solution to the phase detection limit can be achieved, using the readily available *I&Q* signals of the last frequency divider stage in the clock multiplying PLL. The proposed Phase Detector consists of two AND gates. It generates concurrent *UP* and *DN* pulses with a short duration as compared to the output of a PFD operating close to its frequency limit.

A 2.5-to-10 GHz CMU in 0.18μm CMOS

This chapter describes the design of a low-jitter Clock Multiplier Unit that multiplies a reference clock signal of 2.5 GHz by four, resulting in an output clock of 10 GHz. The Clock Multiplier employs the two-AND-gate Phase Detector described in the previous chapter to lock a high-quality 10 GHz integrated LC-oscillator to the 2.5 GHz reference clock. The Clock Multiplier was realized in a standard 0.18μm CMOS process.

A 2.5-TO-10 GHz CMU IN 0.18μM CMOS CHAPTER 6

6.1 Introduction

The previous chapters described some important aspects of designing a high-speed low-jitter clock multiplier. As a demonstrator, a clock multiplier was integrated in a standard 0.18μm CMOS process that multiplies a 2.5 GHz input clock by four, resulting in a low-jitter 10 GHz output signal [94].

The target application of the CMU is a data serializer, multiplexing four parallel streams of 2.5 Gb/s into a 10 Gb/s serial data stream (see Figure 1.1 in chapter 1). The multiplexer itself was already shown to be feasible in 0.18μm CMOS [71]. Therefore, the aim was designing the low-jitter clock multiplier.

The jitter specifications of the CMU were derived from the SONET OC-192 standard, being an optical 10 Gb/s system, albeit with a different reference clock frequency ($\simeq$625 MHz). Our choice of an increased reference frequency is a step toward high performance, high frequency clock multipliers for *next* generation wireline communication systems, such as 40 Gb/s optical systems, where reference frequencies of 2.5 GHz are likely to be used.

From the SONET jitter specifications, only the jitter *generation* is important. If the data multiplexer is part of a serial input/serial output regenerator application, the jitter tolerance is resolved in the Clock and Data Recovery PLL and the jitter transfer can be resolved in an external low-bandwidth clean-up PLL [5], or by using external high-Q filtering of the recovered clock [15, 98]. In many cases, the outgoing data source is a network processor ASIC and the data rate is independent of the incoming data rate. In that case, both ASIC and transmitter are controlled by a separate clean reference clock [5, 15]; jitter transfer and tolerance do not apply to such a system.

The optically measured jitter generation for an OC-192 transmitter must be below 0.1 Unit Interval (UI) peak-to-peak and 0.01 UI rms [15], measured over a 50 kHz to 80 MHz bandwidth (f_l of equation (3.1) equals 50 kHz, f_h equals 80 MHz). Because the data is in a 10 Gb/s Non-Return to Zero (NRZ) format, this translates to 10 ps peak-to-peak jitter and 1 ps rms jitter. In order for the whole optical line card to satisfy these strict jitter specifications, the jitter generation of the transmitter chip should be well below these requirements.

Because a very good 10 GHz *LC*-VCO plus 50Ω output buffer and a 10 GHz divider were already provided by Philips Research to be used in the demonstrator, the focus was on designing a low-noise CMU input part (running at 2.5 GHz). Although the operation frequency is significantly lower than that of the VCO and the divider, the input still forms a big challenge because generation of low duty cycle pulses is required.

The realized CMU design is described in Section 6.2. First the PLL's lock behavior and the

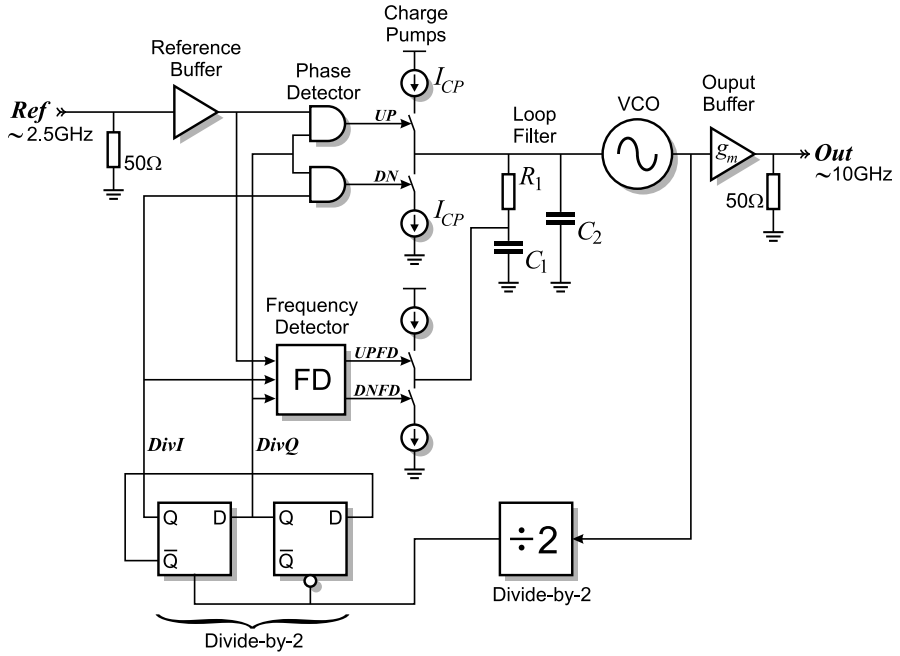


Figure 6.1: Architecture of the clock multiplier unit.

need for a Frequency Detector are explained. Then the implementation of the various PLL building blocks is discussed. Section 6.3 describes experimental results of the realized CMU chip, and of a separate test chip containing the Phase Detector, Frequency Detector and the Charge Pumps they control. The conclusions are summarized in Section 6.4.

6.2 CMU Design

Figure 6.1 shows the top-level block diagram of the CMU. The block diagram shows the use of the novel Phase Detector design in combination with a Frequency Detector (FD), the need for which will be demonstrated shortly. Both detectors control a separate Charge Pump. The VCO output signal of about 10 GHz is passed through two static divide-by-2 circuits to generate the 2.5 GHz signal that is locked to the CMU reference clock. The second divide-by-two stage provides the *I*&*Q* signals needed by the PD and the FD.

6.2.1 Lock Behavior

Before frequency lock is achieved, the phase difference between the reference and the *DivI* signal varies almost linearly with time, due to the frequency difference of both signals. This means that the mean Charge Pump output current of the CP that is controlled by the PD will

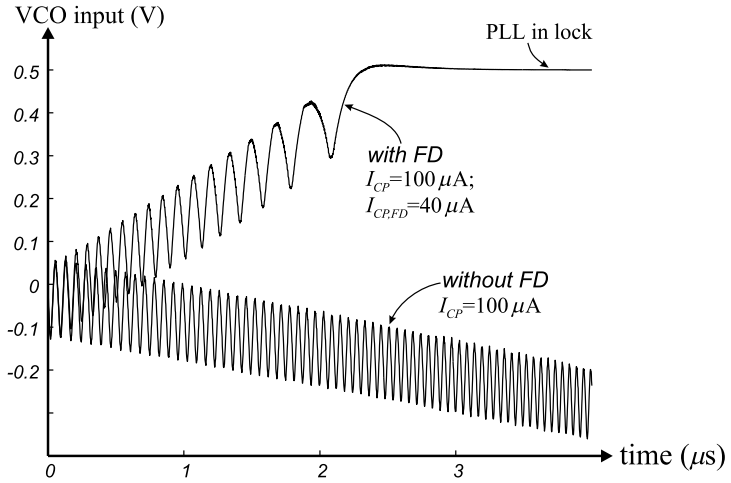


Figure 6.2: PLL lock behavior with and without FD.

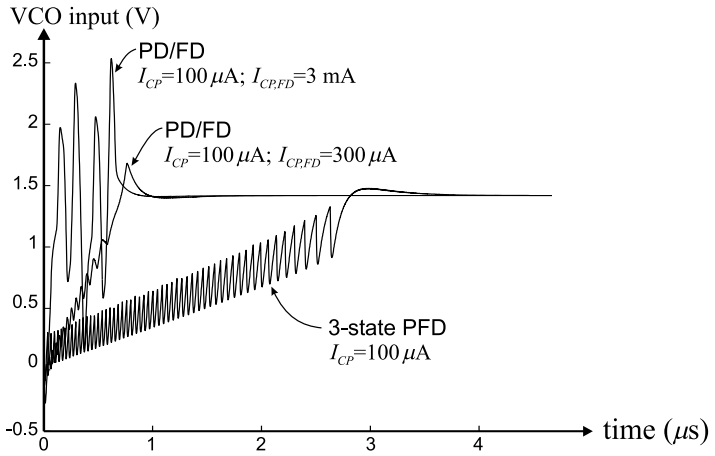


Figure 6.3: Lock behavior of PLL with PFD and of PLL with FD.

be the mean value of the PD/CP transfer curve (the graph of the low frequency component of the Charge Pump output current versus the PD's input phase error). For an ideal PD/CP combination, the mean current will be equal to 0 A. In practice, however, the mean CP output current will be non-zero for various reasons (see *e.g.* Figure 5.14 and Figure 5.16). Because of the integration of this current by the loop filter, the VCO frequency will drift, possibly away from lock.

To ensure correct locking of the loop, a Frequency Detector should be added that drives the

VCO frequency toward the desired output frequency. The mean output current of the CP that is controlled by this FD should be higher than the before mentioned drift current. In this case, the net current will always bring the PLL toward lock.

High-level simulations using Simulink were performed with the CMU structure using the two-AND-gate PD. The goal of these simulations was to study the lock behavior of a PLL with the proposed PD, with and without a Frequency Detector. The implementation of the Frequency Detector used in these simulations will be described later in this chapter.

Simulation results of the CMU architecture of Figure 6.1 are shown in Figure 6.2, where the VCO control voltage is plotted versus the simulation time. The results clearly show the need for the FD. Without the FD, the drifting effect discussed before drives the PLL away from lock ($\simeq$0.5 V in this case). With the FD enabled, the absolute value of the output current of the CP controlled by the FD is higher than that of the drifting current, driving the average VCO control voltage toward lock.

Because the CP current I_{CP} of the PD-controlled CP and that of the FD-controlled CP are orthogonal design parameters in the architecture of Figure 6.1, the lock time of the PLL can in principle be improved, compared to using a PFD. This is illustrated by Figure 6.3, showing the lock behavior of the architecture with the two-AND-gate PD and a Frequency Detector as compared to that of a PLL using a PFD. Before frequency lock is achieved, the PFD input phase error changes almost linearly in time. According to Figure 5.7(b), the mean CP output current is then $\frac{1}{2}I_{CP}$ for an infinitely fast PFD (and less for a realistic PFD, for which t_{reset} is a significant portion of the reference period[1]). The CP controlled by the Frequency Detector can be dimensioned to deliver a higher mean current, resulting in faster lock times.

Note from Figure 6.3 that this increased lock speed has its limitations. If the CP current of the FD controlled Charge Pump is dimensioned too high, the FD control mechanism experiences overshoot, resulting in lock times that may be higher than for a more conservative value of the CP current.

In the implemented CMU, the PLL lock time was not an important specification. To save power and to avoid the before mentioned FD instability, the Charge Pump current of the FD controlled CP was chosen to be 40% of I_{CP}.

6.2.2 The Frequency Detector Implementation

The FD implementation is shown in Figure 6.4. It is similar to that presented in [90], however, two AND-gates were added to generate signals that can directly control a second Charge Pump. A strong point of this detector is that as soon as phase lock has been achieved, it will not generate output signals that may disrupt the normal behavior of the loop. This means two things. First, and most importantly, the Frequency Detector will not contribute in the clock multiplier output jitter. Secondly, there is no need to switch off the Frequency Detector after

[1]The reset time t_{reset} was 25% of the reference period of 400 ps in the simulations, such that in lock the duty cycle of the generated pulses was equal to that of the two-AND-gate PD; note that this PFD speed is very optimistic, as can be concluded from Chapter 5.

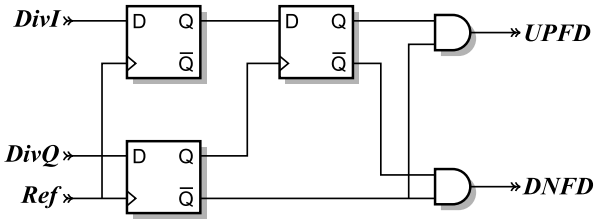


Figure 6.4: Frequency Detector design

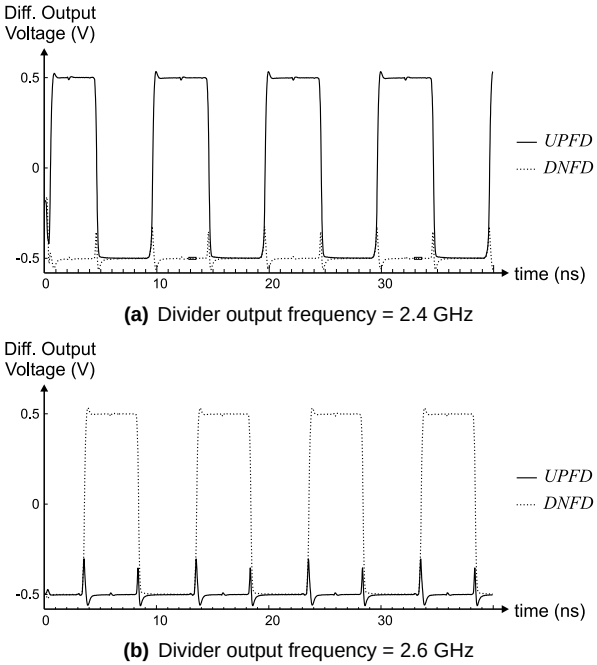


Figure 6.5: FD response for 2.5 GHz reference signal.

start-up. When the loop looses lock (*e.g.* due to a strong disturbance), and the input phase error exceeds 90°, the FD will automatically become active and bring the loop back to lock. So there is no need for a lock detector circuit.

The reason that the D-flip-flops used in this detector can operate at a higher reference frequency than the D-flip-flops used in a conventional PFD architecture is that the flip-flops in this Frequency Detector have an output frequency that is equal to the difference between the reference frequency and the frequency of *DivI* and *DivQ*. Also, the flip-flops do not need to be reset asynchronously, which makes their design easier and faster.

The Frequency Detector can in principle also be used as a high-speed Phase Detector, as de-

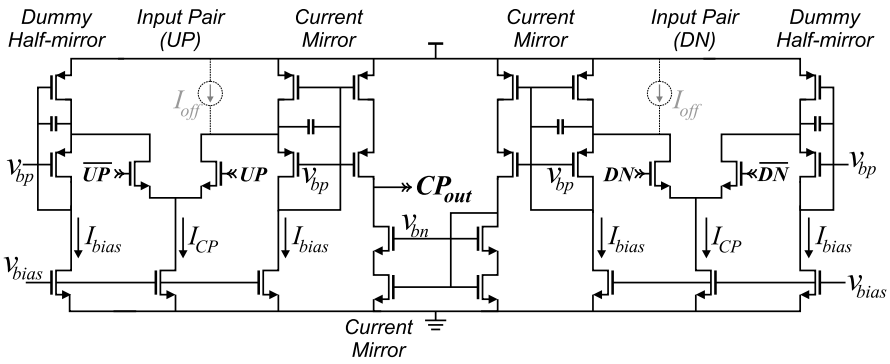


Figure 6.6: Charge Pump Circuitry.

scribed in [90] and in Section 5.2.5. However, the gain of this Phase Detector is not well defined: it depends on the speed of operation and on the waveform of the incoming signals, making stability and loop behavior not as well defined as with the proposed two-AND-gate Phase Detector. Using a separate PD simplifies designing the D-flip-flops because their internal loop gain need not be close to 1, as is needed in [90] to create a sample-and-hold circuit.

Figure 6.5 shows transistor-level simulated output signals of the FD for a reference frequency of 2.5 GHz. The FD was implemented in Current Mode Logic; an output voltage of +0.5V means that the corresponding Charge Pump current source is on, -0.5V means off. The upper graph represents a divider output frequency of 2.4 GHz, resulting in $UPFD$ pulses. The lower graph resulted from a divider frequency of 2.6 GHz, in which case the $DNFD$ pulses become active. Note that when the FD is active, its output signals have a Duty Cycle of 50%. This means that the Charge Pump current source values should be at least twice the worst-case drift current discussed earlier (Section 6.2.1) to guarantee locking.

The theoretical input range of the FD is ±25% of the desired VCO frequency. This is high enough in this CMU design, where an *LC*-oscillator is used of which the tuning range plus the expected process spread plus temperature variations are expected to be well below that.

6.2.3 The Charge Pumps

The Charge Pump controlled by the PD should be able to process the *UP* and *DN* signals at the speed of the reference signal (2.5 GHz).

Figure 6.6 shows the CP-circuit that is used, based on [42]. Both the *UP* and the *DN* signal are processed by an NMOS differential pair. This ensures input symmetry and high speed. The tail current source transistors have a high overdrive voltage to decrease their white noise contribution. The transistors are large enough to ensure a negligible $1/f$-noise contribution to the CMU's output jitter.

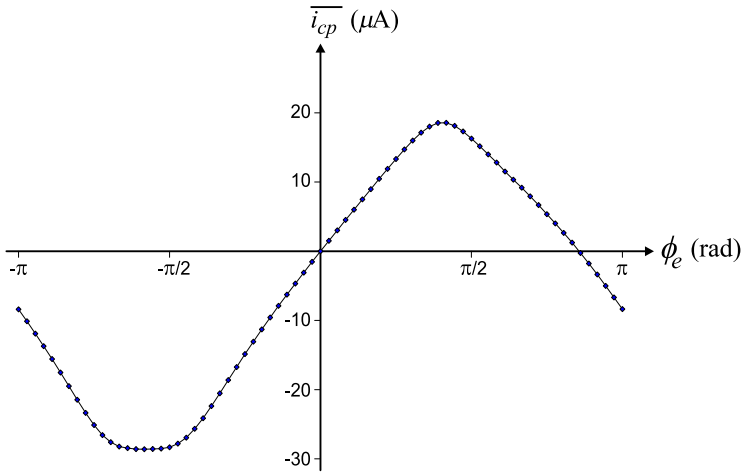


Figure 6.7: Simulated PD/CP response as a function of phase error.

The resulting *UP* and *DN* currents are processed by current mirrors. Although the shape of the output current of these mirrors is different from the input current shape due to the mirror poles, it can be shown that the total *charge* put into the mirror is copied to the output accurately [42]. As long as the mirror pole frequencies are much higher than the PLL loop bandwidth, the mirror poles will not disrupt loop behavior.

Low voltage PMOS mirrors were used to ensure correct operation of the NMOS differential pairs. Simple two-transistor PMOS mirrors would drive the NMOS transistors of the input stages out of saturation because of the high voltage drop the input PMOS transistor would require. The NMOS transistor being in triode means that either the differential output swing of the PD or the NMOS widths have to be increased to fully switch the tail current. Both methods decrease the maximum operation speed of the PD/CP combination. A third method of solving the problem would be to use level-shifters to connect the PD to the CP, also decreasing performance.

Another advantage of using these low-voltage mirrors is the high output impedance obtained by cascoding. This increases the *UP* and *DN* current matching, as the CP output current is less sensitive to the VCO control voltage than without a cascoded CP output stage.

The capacitors in the PMOS mirrors are necessary to provide a fast feed-forward path from the input of the PMOS mirror to the gates of the upper mirror transistors. Without the capacitor, the high-frequency input impedance of the PMOS mirror would be high. This would cause a voltage drop on the input node every time the NMOS differential pair receives an input pulse, causing the NMOS transistor to go out of saturation. The capacitor also provides frequency compensation of the feedback loop present in the PMOS mirror to prevent ringing.

A disadvantage of the use of low-voltage mirrors is the need of an extra bias-current, thus increasing the CP noise. However, because this bias current is just a fraction of the CP

current, this was not a problem.

Figure 6.7 shows the simulated transfer of the two-AND-gate Phase Detector (implemented in Current Mode Logic), combined with the described Charge Pump. The figure shows the simulated mean Charge Pump output current as a function of the phase difference between the reference and the *DivI* signal, at 2.5 GHz input frequencies. There is a nearly flat response around input phase differences of about $-\pi/2$, due to the fact that the time overlap of the reference and the *DivQ* signal becomes very small here. Although the cause of this problem is similar to what normally causes PD/CP dead-zone (narrow pulses), in this case it will not cause the problems usually associated with dead-zone. This is because the PLL will not lock anywhere near this flat part. One can see that there is no gain degradation around zero degrees phase error, showing that the PD/CP combination does not have a dead-zone problem.

For the Frequency Detector CP, two pull-up current sources were added (shown gray in Figure 6.6) to completely switch off the CP output current when in lock, in order to decrease the noise contribution of this CP when the PLL is in lock (and no FD output pulses are generated), which would otherwise be caused by the mirroring of the bias currents.

6.2.4 The VCO

The *LC*-VCO has to operate at 10 GHz at a low-phase noise level, demanding careful design. An important issue is the design of a monolithic planar inductor. Shielding the inductor toward the substrate reduces noise coupling from the substrate and, more importantly, increases the inductor's quality factor. Normally, the shield is patterned in order to prevent circular currents (so called "eddy currents") from flowing [99]. The pattern is made with grounded poly-silicon bars. This grounding can be done from the outer side or inner side of the bars. Measurements reveal that inner grounding is the best option to get the maximum quality factor at high frequencies, see Figure 6.8 [100]. The reason that outer shielding performs worse than no shielding at all is that the outer shield connection forms a ring that absorbs energy from the coil.

In recent work, a relation between the inductance value and the phase noise in the voltage-limited and current-limited region of the VCO is established [101]. The value of the inductor is chosen so that the oscillator operates at the edge of the voltage-limited region. The geometry of the inductor is a single turn circle, realized in metal layers 3, 4 and 5.

The varactor used is a differential PMOS P+ drain/source diffusion in an N-well. The quality factor is approximately 6, making it the limiting element in this design. The C_{max}/C_{min} ratio is in the order of 1.2. To increase the tuning range, digital tuning by means of MOS capacitors has been applied. The digital tuning can cope with fabrication spread, without increasing the gain of the VCO. The overall schematic of the VCO can be seen in Figure 6.9. Figure 6.10 shows the phase noise spectrum of the free-running VCO, measured with a dedicated HP3048A phase noise measurement system.

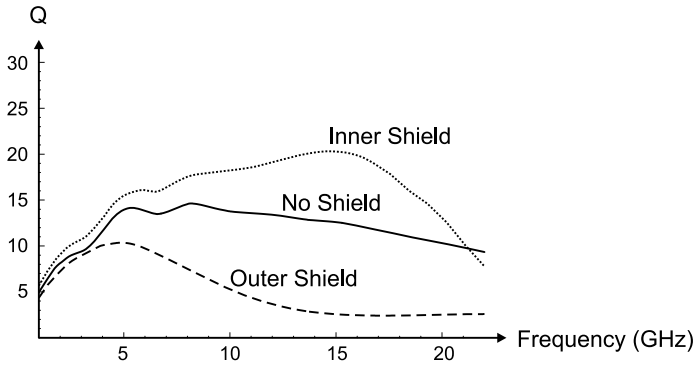


Figure 6.8: Measured influence of shielding on the quality factor of the inductor.

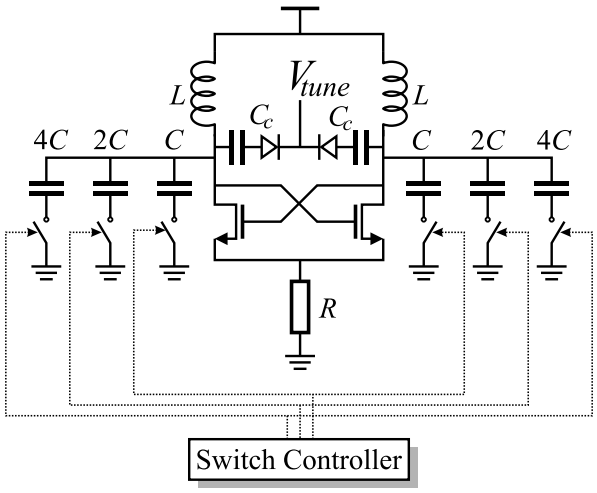


Figure 6.9: The overall VCO schematic.

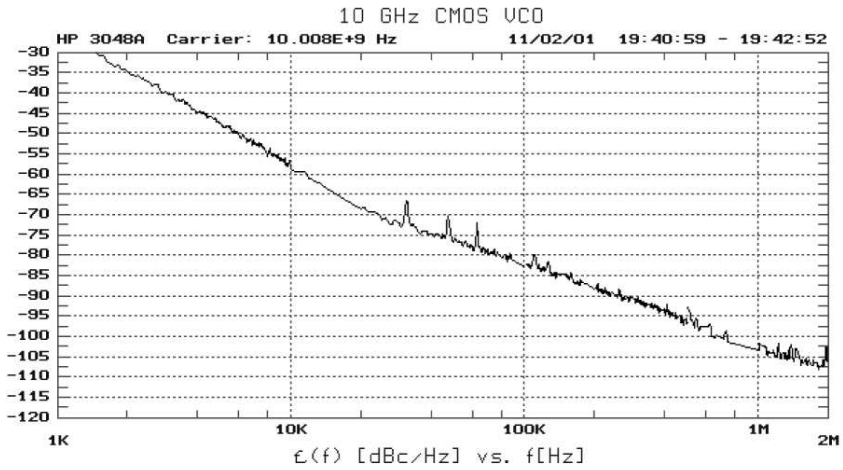


Figure 6.10: Measured phase noise spectrum of free-running VCO.

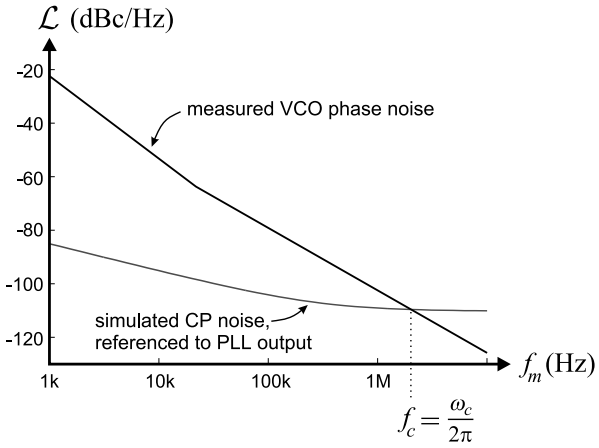


Figure 6.11: Determination of optimal loop bandwidth.

6.2.5 Phase Noise Optimization

To optimize the PLL open loop bandwidth ω_c with respect to output jitter, as described in section 3.4, we used the assumption that the in-band phase noise would be dominated by the CP noise. The input referred synthesizer phase noise due to the Charge Pump noise is calculated according to (3.26). The CP noise spectrum was estimated by using a steady-state AC noise analysis, using the fact that the CP current duty cycle is roughly 25%.

Because Figure 6.10 gives the *single-sideband* VCO phase noise, determination of the opti-

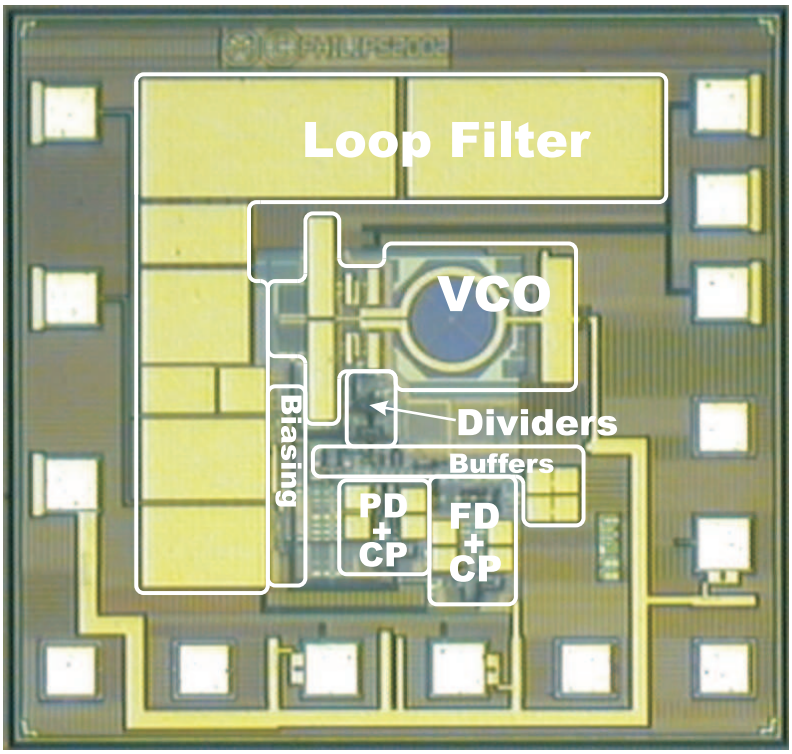


Figure 6.12: CMU die micrograph.

mum ω_c requires a plot of the one-sided equivalent synthesizer phase noise according to:

$$\mathcal{L}_{neq} \equiv \frac{1}{2} S_{\phi_{neq}}(f_m) \tag{6.1}$$

The graph of $\mathcal{L}_{neq}$ is drawn together with the measured phase noise spectrum of the free-running VCO. The optimum loop bandwidth can now be determined from the point of intersection of these two lines, as shown in Figure 6.11. This works out to be about 2 MHz.

6.2.6 Layout

The clock multiplier was realized in a standard 0.18μm CMOS process with 5 metal layers, one poly layer and a substrate resistivity of 10Ω·cm. The process did not provide the possibility of triple well isolation. A die micrograph of the complete CMU is shown in Figure 6.12. The area consumed is about 1x1 mm^2, including the bond pads. The 'active area', including the loop filter, is 0.83x0.86 mm^2.

All digital circuitry (PD, FD, frequency dividers and buffers) was implemented using current

mode logic (CML) to minimize both the sensitivity to and the generation of supply noise and substrate bounce. Both the input and the output buffers were terminated with 50Ω on-chip. The input buffer is AC coupled on chip to adapt to the correct CML levels.

A large portion of the total chip area is taken by the loop filter capacitors (the largest of which has a capacitance of 50 pF). This chip area could have been drastically reduced by using techniques presented in [102]. This was not done however due to a lack of correct capacitor models for these more advanced and dense capacitors; the capacitors were realized with ordinary 5-metal parallel plate capacitors.

6.3 Experimental Results

Two ICs were fabricated, one with the complete Clock Multiplier Unit and the other containing the PD/CP and the FD/CP combinations, for open loop measurements. The results of on-wafer measurements on these chips are presented here.

6.3.1 CMU chip measurements

The 10 GHz output signal generated by the CMU chip is shown on the Agilent 86100B oscilloscope screen-dump of Figure 6.13, together with the 2.5 GHz reference clock, which was derived from a Marconi 2042 signal generator. The reference clock was also used as the trigger signal for the oscilloscope. The measured CMU power consumption was 99 mW including the 10 GHz output buffer. The estimated consumption without the buffer is 81 mW.

The oscilloscope could in principle be used to measure the peak-to-peak jitter of the generated clock signal. In this case, however, the generated clock jitter was below the oscilloscope noise floor. To determine the CMU output clock jitter, we used a more sensitive phase noise measurement, the results of which were integrated to determine the RMS output jitter. Measuring the complete phase noise spectrum also has the advantage that it gives a more detailed picture of the possible sources of jitter and the optimal setting of the PLL loop bandwidth.

The CMU output phase noise was measured using the HP3048A phase noise measurement setup in PLL-configuration. The 2.5 GHz reference to the CMU was derived from a Marconi 2042 signal generator in low noise mode. An HP83731B generator was controlled by the measurement setup to follow the CMU output clock at 10 GHz. The measured phase noise plot of the CMU is shown in Figure 6.14. This graph clearly shows the open loop VCO phase noise (at offset frequencies higher than 1 MHz) and the in-band phase noise, which is dominated by the Charge Pump noise.

Integration of the phase noise spectrum using the integration limits defined for OC-192 SONET systems (50 kHz up to 80 MHz), yields an rms-jitter of 0.22 ps (equivalent to 2.2 mUI rms), which is almost a factor 5 lower than the SONET recommendation of 10 mUI rms. A usual rule-of-thumb approximation of the peak-to-peak jitter of a factor of ten times the rms-jitter shows a jitter of 2.2 ps peak-to-peak (22mUI p-t-p). A performance summary of the CMU chip is given in Table 6.1.

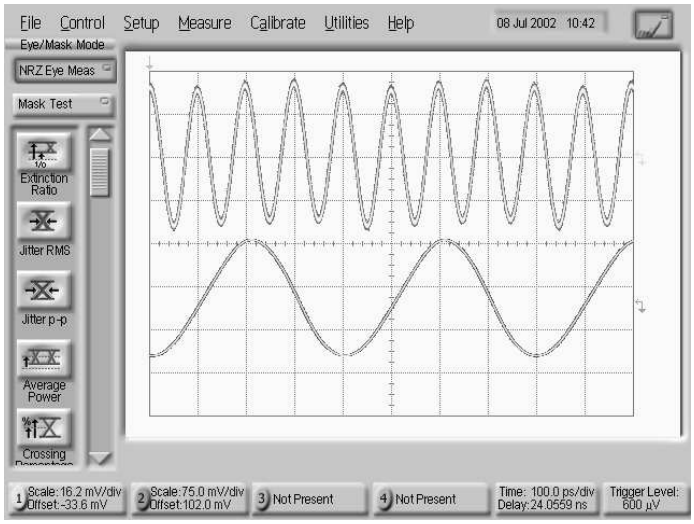


Figure 6.13: Oscilloscope screen-dump of reference clock and CMU output.

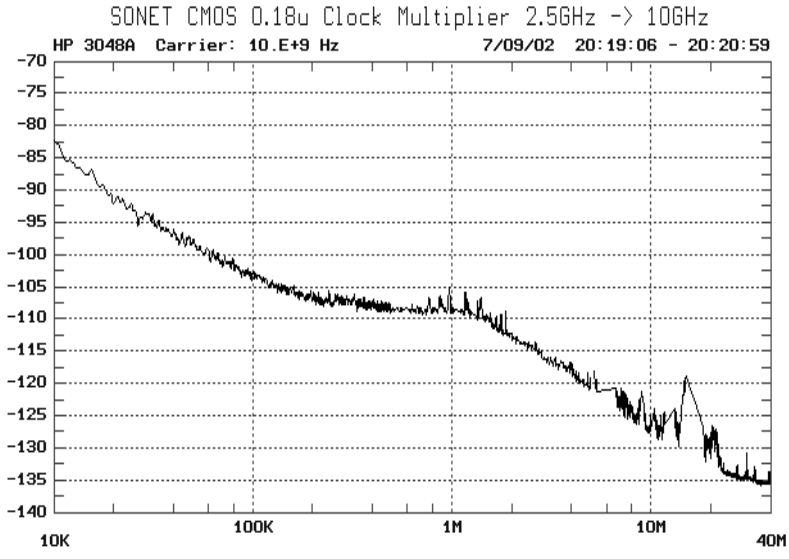


Figure 6.14: Measured generated CMU phase noise spectrum.

Property	Value
Output Frequency	9.953 GHz
Reference Frequency	2.488 GHz
Technology	0.18μm standard CMOS
Jitter Generation (RMS)	0.8º rms, equiv. to 0.22 ps rms
Jitter Generation (peak-to-peak)	2.2 ps (SONET spec. = 10 ps)
Supply Voltage	1.8 V
Chip Size	0.83 x 0.86 mm^2
Power Consumption	VCO: 25 mW Dividers: 32 mW Output buffer: 18 mW PD,FD,CPs: 10 mW Misc. buffers: 14 mW **Total: 99 mW**

Table 6.1: CMU performance summary.

6.3.2 Stand-alone PD and FD chip measurements

The PD speed was measured using the chip containing the open-loop PD and CP combination. The measurement setup is shown in Figure 6.15. The two generators deliver two independent signals, close in frequency. The delay line in series with the second generator creates a phase difference between *DivI* and *DivQ*. This phase difference is frequency dependent as such:

$$\phi_e = 2\pi f_2 \Delta\tau \tag{6.2}$$

with f_2 being the frequency of the second generator and $\Delta\tau$ the difference in delay time of both the cables connecting *DivI* and *DivQ*. The frequency should be controlled such that the mean output current of the CP is about zero and falling with increasing input frequency. In that case, the phase difference between *DivI* and *DivQ* is close to 90°.

The PD's input phase difference increases linearly due to the constant frequency difference of the generators. The CP output current will, thus, be a periodic signal with a frequency equal to the difference frequency of both generators and the signal on the digital oscilloscope can be considered a plot of the mean Charge Pump output current versus PD input phase error.

Figure 6.16 shows some of the measured output plots for different PD input frequencies. The first two plots, at 1 GHz and 2.5 GHz, show the expected linear behavior around zero degrees phase offset (the origin of the oscilloscope dump), and the flat area in the third quadrant due to the narrow *UP*-pulses described before.

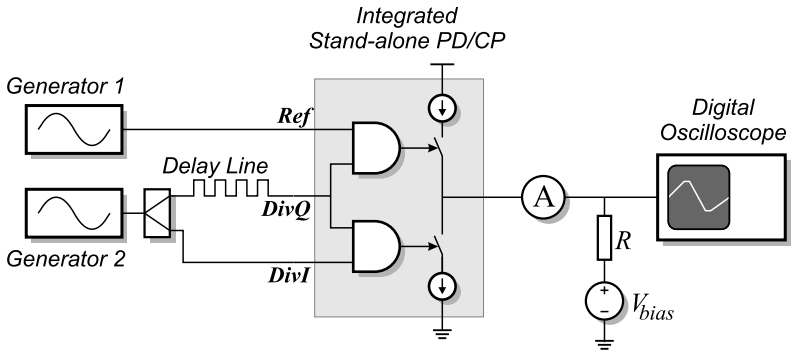


Figure 6.15: Setup for PD/CP transfer measurements.

It should be noted that the actual value of the PD input phase error is not known[2], in contrast to the value of the CP output current. The actual input phase for which the CP output current equals zero can deviate from zero degrees due to device mismatch errors on the chip, both of the PD AND-gates and the CP current sources. Of this mismatch, only the CP current mismatch will influence reference breakthrough. During layout, effort has been taken to minimize CP current mismatch. The fact that cascoded current sources were used greatly reduces the influence of VCO control voltage on CP current mismatch.

The plots measured at higher input frequencies show a slight degradation of both the gain in the origin and the size of the linear range of the PD/CP combination. This is due to the fact that both the *UP* and *DN* pulses become too narrow to fully control the CP.

Measurements performed on a stand-alone FD/CP combination implemented on a single chip showed that the FD had an input range of 23%. This is sufficient, taking into account the combined VCO tuning range together with its expected process spread.

6.4 Summary of Conclusions

Using the two-AND-gate Phase Detector described in chapter 5, an experimental 2.5-to-10 GHz Clock Multiplier Unit in a standard 0.18μm CMOS process has been designed. The Phase Detector operates at the reference frequency of 2.5 GHz, without the need of a reference divider, using the readily available output signals of the main divider.

Initial PLL locking is achieved with a frequency detector that automatically becomes inactive while in lock, therefore not influencing generated output jitter and spurs. This technique makes a lock-detect circuit superfluous. The CMU achieves an output jitter of 0.22 ps rms, almost a factor 5 lower than the 10 Gbps SONET recommendation, while consuming 100 mW (including the 10 GHz output buffer). Comparing to state-of-the-art [103], this design realized

[2]This would require knowing the exact cable lengths, the effect of the probes that were used and internal wiring and buffering on the chip.

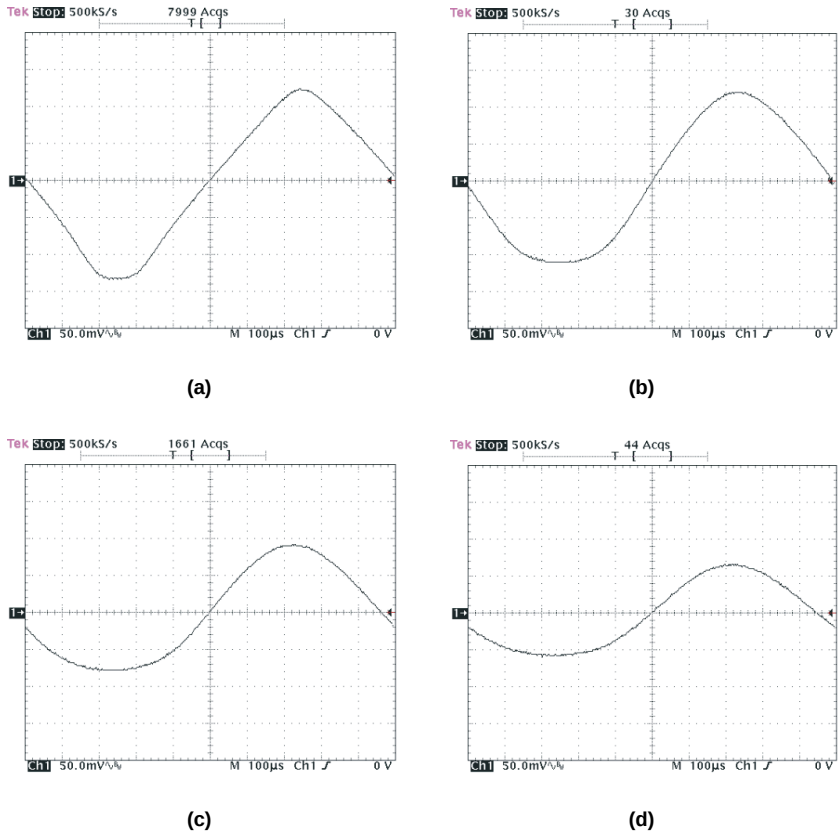
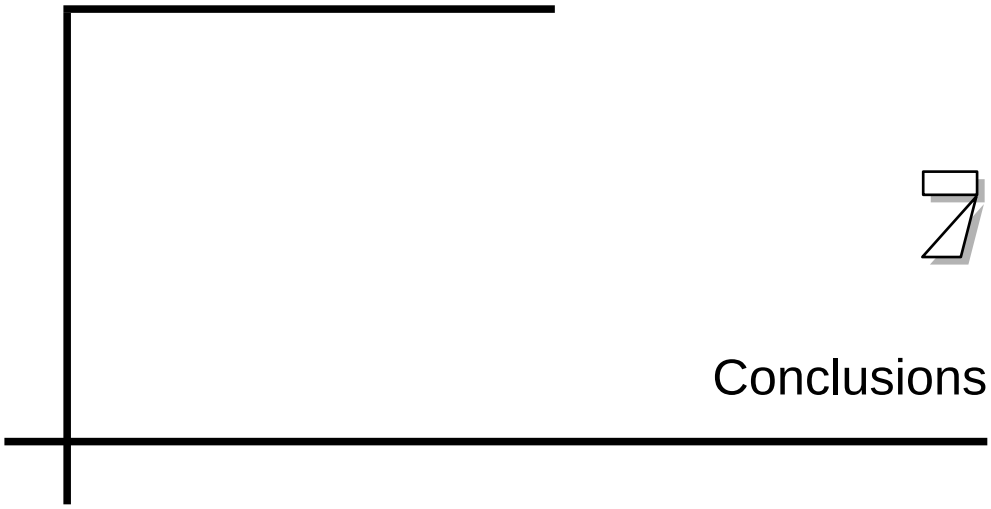


Figure 6.16: PD/CP measurement results, measured at: (a)1.0 GHz; (b)2.5 GHz; (c)4.0 GHz; (d)5.0 GHz;

an approximate factor 3 improvement in peak jitter performance, and 2 to 3 times less power dissipation in a comparable technology.



7 Conclusions

This chapter summarizes the conclusions reached in this thesis and the original contributions proposed here.

7.1 Summary of Conclusions

7.1.1 Chapter 1: Introduction

Chapter 1 is an introductory chapter. It describes the main applications of clock and frequency multiplication (serial communication, clocked digital and mixed signal ICs and tuning systems) and gives a definition of jitter that is used throughout this thesis. Apart from the well known PLL-based multiplier, the DLL-based multiplier and the clock interpolator are discussed. Chapter 1 briefly discusses the reason the DLL-based solution is of interest: the jitter accumulation of the PLL counterpart, which does not occur in the DLL based multiplier. Note that this is studied in detail in chapter 2. Chapter 1 motivates the remainder of the work presented in this thesis, which is focussed on jitter analysis and optimization, and on achieving high-speed operation of PLL building blocks. The focus on CMOS technologies is accounted for mainly by the high level of integration that these technologies allow.

7.1.2 Chapter 2: Comparing DLL and PLL

The choice between a PLL and a DLL-based Clock Multiplier is examined in chapter 2, based on the output jitter of both structures. Earlier analysis found in literature mentions jitter accumulation effects in the VCO of a PLL, which would presumably result in higher output jitter compared to a DLL-based solution. The fact that frequency multiplication is required, however, results in a DLL delay line consisting of more cells than a ring oscillator of a PLL in general. This leads to a lower power consumption per delay cell (assuming equal total consumption of the VCO and the VCDL), which in turn leads to higher thermal noise induced jitter of the delay cells in the VCDL. This effect is stronger than the jitter accumulation that the VCO of a PLL suffers from, leading to the conclusion that a wide-band PLL used for clock multiplication produces less output jitter than a DLL-based implementation of the same function. The conclusion will even be stronger in favor of the PLL if the VCO is realized using an *LC*-oscillator, due to its superior phase noise quality.

Chapter 2 also takes into account another important source of (deterministic) jitter: delay cell mismatch and charge pump current source mismatch, which are both an issue in DLL-based multipliers. Monte Carlo simulations with a modern CMOS process indicate that jitter due to delay cell mismatch is dominant in a DLL where intermediate clock phases of the VCDL are also used, due to the clock skew that is caused by the mismatch.

It has been shown, using the concept of Impedance Level Scaling, that there is a direct trade-off between power usage and output jitter of the frequency multiplier. This trade-off is identical for both jitter due to noise and due to mismatch. The amount of output jitter is limited directly by the power budget of the circuit. It was shown that if the delay cell mismatch is

the most dominant jitter source for a certain circuit, it will still be dominant in an impedance level scaled version of this circuit.

The comparative analysis revealed essential differences between the PLL and the DLL. The DLL output jitter can be minimized by minimizing the DLL loop bandwidth. The function of the control loop is not to filter out jitter (as is the case for a PLL), but merely to tune the value of the mean delay of the VCDL to be equal to the reference period. For a very small loop bandwidth, the DLL behaves as if uncontrolled with respect to jitter. In contrast, for an integer-*N* PLL, the PLL loop bandwidth shows a certain optimum, where the output jitter is minimized. It has been shown that, contrary to common belief, the DLL is not unconditionally stable.

7.1.3 Chapter 3: Low-Jitter PLL Design Issues

The PLL based multiplier, being the most promising architecture according to the analysis of chapter 2, is focussed on in the remainder of the thesis. Whereas chapter 2 used time-domain analysis, mainly to easily model the VCDL delay, this type of analysis is abandoned for frequency-domain analysis, thus making analysis of higher-order loop filters and $1/f$ noise sources more convenient. Jitter can be then determined by integrating the phase noise spectrum.

Roughly half of the total PLL output jitter is due to building blocks other than the VCO. Therefore, it is rewarding to minimize this jitter. It has been shown that this can be done effectively by using as high a reference frequency as possible, as the PLL output jitter $\sigma_{t_o}^2$ will be inversely proportional to the square root of f_{ref}. To minimize Charge Pump noise, the phase detector should generate pulses with a short active duration. Both a high reference frequency and low Charge Pump activity help reduce the integrated loop filter area.

Via the concept of an optimum loop filter capacitance, it has been shown that PLL area can be saved, at the price of increased output jitter. However, because of the flat optimum, this price is small as long as the actual capacitance area is not reduced too much.

7.1.4 Chapter 4: Dimensioning Current Mode Logic

Current Mode Logic is an important logic family in mixed-signal designs because of low Delta-*I* noise generation and sensitivity. A disadvantage of using CML is that generally no standard gate libraries are available, and neither is an easily applicable method of dimensioning the transistors. Chapter 4 demonstrates a simple method of dimensioning the elements of CML gates, based upon determination of the maximum switchable tail current for given dimensions of the differential pairs. It was shown that, contrary to common belief, using large signal swings improves the speed of CML circuits. However, the maximum swing is in practice limited by the voltage overhead needed by the tail current source.

It was explained and demonstrated that a 2-input CML gate can operate at a maximum frequency that is half that of a simple CML buffer. Simulations showed that controlling the

bottom differential pair results in a gate delay 25% higher than when controlling the top differential pair. Finally, it was shown that the two layers of differential pairs can be dimensioned equally. Although this theoretically is not optimal, the optimum is extremely flat and the speed penalty of this simple rule-of-thumb is negligible.

7.1.5 Chapter 5: High-speed Phase Detection

Chapter 5 starts by giving an overview of Phase Detectors eligible for controlling a high-reference-frequency PLL. Comparing these Phase Detectors, we concluded that linear digital PDs (generating digital output signals with the phase difference information represented by duty-cycle) are preferred over analog PDs, as the latter generally have an unpredictable gain. The output signals of a tri-state PFD are particularly convenient because of the low net Charge Pump activity in lock and the predictable PLL static phase error. However, the inherent feedback loop present in the tri-state PFD was shown to limit its operation frequency, and with that, the reference frequency.

It was shown that a simple and fast solution to the phase detection limit can be achieved, using the readily available *I*&*Q* signals of the last frequency divider stage in the clock multiplying PLL. The proposed Phase Detector consists of just two AND gates. It generates concurrent *UP* and *DN* pulses with a short duration as compared to the output of a PFD operating close to its frequency limit. Simulations showed a significant speed improvement (two times the maximum PFD reference frequency). For low-jitter operation, this two-AND-gate PD is preferred over both the XOR-gate and the tri-state PFD when a high reference frequency is available.

7.1.6 Chapter 6: A 2.5-to-10 GHz CMU in 0.18μm CMOS

In chapter 6, a test-chip is described that implements a high-speed low-jitter Clock Multiplier Unit in a standard 0.18μm CMOS process. The clock multiplier converts a 2.5 GHz reference clock into a 10 GHz output clock, using a PLL based architecture, that controls a high-quality integrated *LC*-oscillator.

The core of the PLL's control loop is the two-AND-gate Phase Detector that was described in chapter 5, operating at the 2.5 GHz reference frequency directly for best jitter performance. The PD uses the output signals of the PLL's main divider to lock the VCO to the reference clock. The output pulses control a high-speed Charge Pump circuit.

As the two-AND-gate PD does not provide frequency discrimination, a separate Frequency Detector is needed to guarantee initial locking of the PLL. A simple frequency detector circuit was used that automatically becomes inactive as soon as lock has been achieved, therefore not influencing generated output jitter and spurs. A lock detect circuit is not needed.

All digital circuitry (PD, FD, frequency dividers and on-chip buffering) was designed in Current Mode Logic to minimize generation of supply and substrate noise and sensitivity to that type of noise.

Experimental results show that the CMU achieves an output jitter of 0.22 ps rms, which is almost a factor 5 lower than the 10 Gbps SONET recommendation, while consuming 100 mW (including a 50 Ω terminated 10 GHz output buffer). Comparing to state-of-the-art [103], this design realized an approximate factor 3 improvement in peak jitter performance, and 2 to 3 times less power dissipation in a comparable technology.

7.2 Original Contributions of this Thesis

- A time-discrete analysis of stochastic DLL and PLL output jitter, including all important noise sources.
- The notion that from the requirement of frequency multiplication follows that the PLL jitter accumulation is in general less than the DLL jitter increase due to a lower power availability per delay cell.
- A description of deterministic DLL jitter due to Delay Cell and Charge Pump mismatch, and the observation that this type of mismatch hardly influences PLL output jitter.
- The introduction of the concept of optimal loop filter capacitor value, via Impedance Level Scaling the Charge Pump.
- A very fast and simple linear Phase Detector, consisting of just two-AND-gates, that generates *UP* and *DN* pulses that are concurrent in lock like those generated by a tri-state Phase Frequency Detector. The two-AND-gate PD, however, can operate at much higher frequencies than the PFD can.
- A Frequency Detector circuit that can control a charge pump using the same input signal as the two-AND-gate PD and that switches off and on automatically depending on whether or not the PLL is locked.
- A fast and accurate Charge Pump circuit suitable for low-voltage operation. The Charge Pump is controlled by differential signals with CML compatible levels.
- A simple method of dimensioning CML gates based on maximum switchable tail current.

7.3 Publication List

What follows is a list of publications arising from the work presented in this thesis.

7.3.1 Patents

- B. Nauta, R.C.H. van de Beek and C.S. Vaucher, “Phase-Locked-Loop with reduced Clock Jitter.” Patent Nr. WO03065586, 2003.

- R.C.H. van de Beek, E.A.M. Klumperink, B. Nauta and C.S. Vaucher, "A very fast Duty Cycle independent Phase Detector with low reference breakthrough." Patent Nr. PHNL020803, 2003.

7.3.2 Papers and Conference Contributions

- R.C.H. van de Beek, E.A.M. Klumperink, C.S. Vaucher and B. Nauta, "Analysis of Random Jitter in a Clock Multiplying DLL Architecture." *ProRisc 2001, Veldhoven*.
- R.C.H. van de Beek, E.A.M. Klumperink, C.S. Vaucher and B. Nauta, "On Jitter due to Delay Cell Mismatch in DLL-based Clock Multipliers." *ISCAS 2002, Phoenix*.
- R.C.H. van de Beek, E.A.M. Klumperink, C.S. Vaucher and B. Nauta, "Low-jitter clock multiplication: a comparison between PLLs and DLLs," *IEEE Trans. Circuits Syst.—II*, vol. 49, pp. 555–566, August 2002.
- R.C.H. van de Beek, E.A.M. Klumperink, C.S. Vaucher and B. Nauta, "Jitter in DLL-Based Clock Multipliers caused by Delay Cell Mismatch." *ProRisc 2002, Veldhoven*.
- R.C.H. van de Beek, C.S. Vaucher, D.M.W. Leenaerts, N. Pavlovic, K. Mistry, E.A.M. Klumperink and B. Nauta, "A 2.5 to 10GHz Clock Multiplier Unit with 0.22ps RMS Jitter in a 0.18μm CMOS Technology," *ISSCC 2003, San Francisco*.
- R.C.H. van de Beek, C.S. Vaucher, D.M.W. Leenaerts, E.A.M. Klumperink and B. Nauta, "A 2.5-to-10 GHz Clock Multiplier Unit with 0.22 ps RMS jitter in standard 0.18μm CMOS," submitted to the IEEE Journal of Solid-State Circuits.
- R.C.H. van de Beek, C.S. Vaucher, D.M.W. Leenaerts, E.A.M. Klumperink and B. Nauta, "A Low-Jitter 2.5-to-10-GHz Clock Multiplier Unit in CMOS," *ProRisc 2003, Veldhoven*.

APPENDICES

NOISE TRANSFER IN A TIME-DISCRETE SYSTEM: AN EXAMPLE

APPENDIX A

To demonstrate how to obtain the output jitter of a system described by difference equations, the calculation of the output jitter of a DLL with a VCDL that consists of noisy delay cells is shown in this Appendix. This is done using the set of difference equations given by (2.3) describing the DLL behavior mathematically as an example. For this analysis, we use the assumptions given in section 2.2.1 of this paper.

First, we assume that the noisy delay cells are the only source of jitter. The quantity of interest is the variance of the signals Δt_m (the jitter variance of the different VCDL output taps). To be able to calculate these variances, it proves easiest to first forget about the intermediate output taps and to calculate the jitter variance of the last output tap only (the one numbered $m = M$). After having done that, calculating the jitter of the other output taps proves straightforward.

Setting all noise sources to zero, except for the delay cell noise, and substituting $m = M$ in the set of difference equations given by (2.3), gives us the following set of equations:

$$v_c(n) = v_c(n-1) + \frac{I_{CP}}{C_f}\Delta t_M(n-1) \tag{A.1a}$$

$$\Delta t_M(n) = -K_d v_c(n) + \sum_{l=1}^{M} \Delta d_l(n) \tag{A.1b}$$

substituting (2.3a) in (2.3b).

The statistical mean of Δt_M is zero, as this is a linear system and the noise sources have zero mean. This means the variance of Δt_M can be written as:

$$\begin{aligned}\sigma^2_{\Delta t_M} = E\left(\Delta t_M^2\right) &= E\left\{\left(-K_d v_c(n) + \sum_{l=1}^{M}\Delta d_l(n)\right)^2\right\} = \\ &= K_d^2 \cdot E\left(v_c^2(n)\right) - 2K_d \cdot E\left(v_c(n)\sum_{l=1}^{M}\Delta d_l(n)\right) + E\left\{\left(\sum_{l=1}^{M}\Delta d_l(n)\right)^2\right\}\end{aligned} \tag{A.2}$$

Because the variance of the tuning voltage does not depend on the period number n in the locked situation (in this situation, the output jitter is the result of a stationary process), this equation can now be reduced to:

$$\sigma^2_{\Delta t_M} = E\left(\Delta t_M^2\right) = K_d^2 \cdot E\left(v_c^2\right) + M \cdot E\left(\Delta d^2\right) \tag{A.3a}$$

taking into account the variables in (A.2) that are uncorrelated. We also assume that the jitter of every delay cell has the same statistical properties, meaning that $E\left(\Delta d_l^2\right)$ does not depend on l and can be written as $E\left(\Delta d^2\right)$.

This equation shows that in order to relate the variance of Δt_M directly to the delay cell noise variance, the variance of the tuning voltage v_c needs to be known. This variance can be found by using (A.1a). The following equation can be derived from it by squaring both the left and the right hand side, followed by equating the expected value of both sides, taking into account the uncorrelated variables:

$$E\left(v_c^2\right) = E\left(v_c^2\right) + 2\frac{I_{CP}}{C_f} E\left\{v_c(n)\,\Delta t_M(n)\right\} + \frac{I_{CP}^2}{C_f^2} E\left(\Delta t_M^2\right) \tag{A.3b}$$

Note that all expected values are independent of the value of n; if the equation still features this variable it is only to clarify the time relationship between two different variables.

Now there are two equations with three unknowns. To solve this problem, a new equation can be derived by adding $K_d v_c(n)$ on both sides of (A.1b). Squaring this equation and equating the expected value of both the left and the right hand side results in the needed new independent equation, making it possible to solve for the tuning voltage variance:

$$K_d^2 \cdot E\left(v_c^2\right) + 2K_d \cdot E\left\{v_c(n)\,\Delta t_M(n)\right\} + E\left(\Delta t_M^2\right) = M \cdot E\left(\Delta d^2\right) \tag{A.3c}$$

Finally, solving the set of equations (A.3) results in:

$$E\left(v_c^2\right) = E\left(\Delta d^2\right) \cdot \frac{I_{CP}}{K_d C_f} \cdot \frac{2M}{2 - \frac{I_{CP}K_d}{C_f}} \tag{A.4a}$$

$$E\left(\Delta t_M^2\right) = E\left(\Delta d^2\right) \cdot \frac{2M}{2 - \frac{I_{CP}K_d}{C_f}} \tag{A.4b}$$

What is left now, is to calculate the jitter of the intermediate VCDL output taps. Having found the variance of the tuning voltage, this is straightforward. Using (2.3c), we can write:

$$\begin{aligned}\sigma_{\Delta t_m}^2 &= E\left\{\left(-\frac{m}{M}K_d v_c(n) + \sum_{l=1}^{m}\Delta d_l(n)\right)^2\right\} = \\ &= \frac{m^2}{M^2}K_d^2 \cdot E\left(v_c^2\right) - 2\frac{m}{M}K_d \cdot E\left(v_c(n)\sum_{l=1}^{m}\Delta d_l(n)\right) + m \cdot E\left(\Delta d^2\right) = \\ &= \frac{m^2}{M^2}K_d^2 \cdot E\left(v_c^2\right) + m \cdot E\left(\Delta d^2\right)\end{aligned} \tag{A.5}$$

again taking into account uncorrelated variables.

Substituting the variance of the tuning voltage given by (A.4a) yields:

$$\sigma^2_{\Delta t_m} = \frac{m}{M}\left\{\frac{m\frac{K_d I_{CP}}{C_f}}{2 - \frac{K_d I_{CP}}{C_f}} + M\right\} \tag{A.6}$$

An approach similar to the one used in this Appendix can be used on any of the difference equation sets given in Chapter 2.

In this appendix, the VCO output phase error resulting from a charge pulse into the loop filter is calculated. This phase error is then used to calculate the time error of the zero-crossings following the charge injection. This leads to an equation that is used in Chapter 2 to derive the difference equation describing the combined behavior of the loop filter and the VCO.

The part of the PLL that is under investigation here, is the loop filter and the VCO, illustrated for convenience in Figure B.1. The current pulse that is pumped into the loop filter is assumed to be Dirac-pulse shaped and occurring at the moment in time labelled $t = 0$, so the total current pumped into the loop filter is equal to:

$$i_{LF}(t) = q_{LF}\,\delta(t) \tag{B.1}$$

with q_{LF} the charge contained in the current pulse.

The VCO angular frequency is given by:

$$\omega_{VCO}(t) = \omega_{fr} + K_{VCO}\,v_{tune}(t) \tag{B.2}$$

with ω_{fr} the free-running VCO angular frequency, K_{VCO} the VCO gain and $v_{tune}(t)$ the VCO control voltage.

The VCO output phase is the integral of the angular frequency:

$$\phi_{VCO}(t) = \int_{-\infty}^{t} \omega_{VCO}(\tau)\,d\tau = \phi_{VCO}(0^-) + \int_{0^-}^{t} \omega_{VCO}(\tau)\,d\tau =$$

$$= \phi_{VCO}(0^-) + \omega_{fr}t + K_{VCO}\int_{0^-}^{t} v_{tune}(\tau)\,d\tau \tag{B.3}$$

using (B.2), with $\phi_{VCO}(0^-)$ the VCO output phase just before $t = 0$.

The VCO control voltage depends on the current $i_{LF}(t)$ pumped into the loop filter and the initial voltage on the loop filter capacitor:

$$v_{tune}(t) = i_{LF}(t)\,R_1 + \frac{1}{C_1}\int_{0^-}^{t} i_{LF}(\tau)\,d\tau + v_c(0^-) \tag{B.4}$$

Substituting this in (B.3) yields:

$$\phi_{VCO}(t) = \phi_{VCO}(0^-) + \omega_{fr}t + K_{VCO}\left\{\frac{1}{C_1}\iint_{0^-}^{t} i_{LF}(\tau)\,d\tau^2 + R_1\int_{0^-}^{t} i_{LF}(\tau)\,d\tau + v_c(0^-)\,t\right\} \tag{B.5}$$

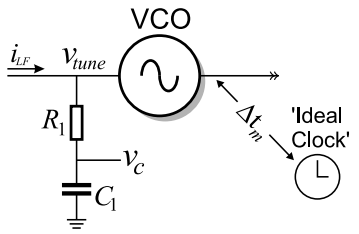


Figure B.1: Loop Filter and VCO.

This rather awkward equation can be easily simplified by realizing the Dirac-shaped nature of the input current, described by (B.1):

$$\phi_{VCO}(t) = \phi_{VCO}(0^-) + \left\{\omega_{fr} + K_{VCO}v_c(0^-)\right\}t + K_{VCO}R_1\left(1 + \frac{t}{R_1C_1}\right)q_{LF} \tag{B.6}$$

Now we define an ideal clock having a phase of exactly $\omega_{fr}t$, as was also done in Chapter 2. The fact that this ideal clock runs at the free-running VCO frequency is justified in Chapter 2 and does not influence the jitter calculations performed there due to superposition. To calculate the VCO jitter, we define its phase error as the deviation from the ideal clock phase:

$$\Delta\phi_{VCO}(t) \equiv \omega_{fr}t - \phi_{VCO}(t) \tag{B.7}$$

This VCO phase error is now easily worked out using (B.6), which gives us:

$$\Delta\phi_{VCO}(t) = \Delta\phi_{VCO}(0^-) - K_{VCO}v_c(0^-)\,t - K_{VCO}R_1\left(1 + \frac{t}{R_1C_1}\right)q_{LF} \tag{B.8}$$

with $\Delta\phi_{VCO}(0^-)$ the phase error just before $t = 0$.

A small simplification can be made by realizing that:

$$v_c(0^+) = v_c(0^-) + \frac{q_{LF}}{C_1} \tag{B.9}$$

where $v_c(0^+)$ is voltage across the loop filter capacitor just *after* the charge is dumped into it. Substituting this in (B.8) yields:

$$\Delta\phi_{VCO}(t) = \Delta\phi_{VCO}(0^-) - K_{VCO}v_c(0^+)\,t - q_{LF}K_{VCO}R_1 \tag{B.10}$$

Note that this is just a minor simplification leading to a somewhat more convenient difference equation.

As long as the jitter remains low compared to the VCO period, the zero-crossing time error of the VCO output can be estimated well by sampling the VCO phase at the *ideal* zero-crossing moments (which are the positive zero-crossing moments of the ideal clock defined before).

Using the fact that the ideal positive zero-crossing moments are spaced by T_{ref}/N (where T_{ref} is the period time of the reference clock the PLL locks to, and N the frequency multiplication factor), the VCO phase error can be converted into a timing error:

$$\Delta t \approx -\frac{\Delta\phi_{VCO}}{2\pi}\frac{T_{ref}}{N}, \tag{B.11}$$

Now the time error of the m-th positive zero-crossing of the VCO after the charge injection can be estimated well by substituting $t = mT_{ref}/N$ into (B.10) and using (B.11):

$$\Delta t_m \approx \Delta t(0^-) - v_c(0^+)\,\frac{mK_{VCO}T_{ref}^2}{2\pi N^2} - \frac{K_{VCO}R_1T_{ref}}{2\pi N}\,q_{LF} \tag{B.12}$$

This final equation is used in Chapter 2 to derive difference equation (2.10c).

In this appendix, the period time deviations of a VCO with white frequency-noise are evaluated. This is done by first examining mathematically the variance of a so-called random walk signal. Then, this theory is applied to oscillators.

C.1 Variance of a 'Random Walk' signal

If a white noise signal is integrated with a pure integrator (having a DC-gain of infinity), the resulting signal is called a random walk signal. This name probably stems from the similarity with the Brownian movement of a small particle in a gas-filled space. Because the output phase of a VCO is related to the integral of its control signal (see equation (B.3)), application of a white noise signal at the VCO input leads to random walk of the output phase. To be able to calculate the effect of this random walk on period deviations, a mathematical analysis of the variance of a random walk signal is shown first.

We want to determine the variance of a random walk signal y after some time t. First, we define the signal y, being the integral of the white noise signal, as follows:

$$y(t) \equiv \begin{cases} \int_0^t \eta(\tau)\, d\tau & t \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{C.1}$$

with $\eta(\tau)$ the white noise that is integrated.

To get rid of this 'case'-notation, we define the following:

$$x(t) \equiv \mathbb{1}(\tau)\, \eta(\tau) \tag{C.2}$$

with $\mathbb{1}(t)$ the unit-step function.

Thus, we are able to rewrite (C.1) as

$$y(t) = \int_{-\infty}^{t} \mathbb{1}(\tau)\, \eta(\tau)\, d\tau = \int_{-\infty}^{t} x(\tau)\, d\tau \tag{C.3}$$

If we consider x as being the 'input' signal and y the 'output' signal of a system with an impulse response of

$$h(t) = \mathbb{1}(t) \tag{C.4}$$

which is the impulse response of an integrator, we can write the output autocorrelation function $R_{yy}(t_1, t_2)$ in the form of a convolution integral [104]:

$$R_{yy}(t_1, t_2) = \int_{v=-\infty}^{\infty} \int_{u=-\infty}^{\infty} R_{xx}(t_1 - u, t_2 - v)\, h(u)\, h(v)\, du\, dv \tag{C.5}$$

with $R_{xx}(t_1, t_2)$ the autocorrelation function of the input signal x.

If S_η is the single-sided power spectral density (PSD) of the white-noise input signal $\eta(t)$, its autocorrelation function is:

$$R_{\eta\eta}(t_1, t_2) = \frac{S_\eta}{2}\, \delta(t_1 - t_2) \tag{C.6}$$

Using (C.2), the autocorrelation function of the signal x then becomes:

$$R_{xx}(t_1, t_2) = \frac{S_\eta}{2}\, \delta(t_1 - t_2) \cdot \mathbb{1}(t_1) \cdot \mathbb{1}(t_2) \tag{C.7}$$

Substituting (C.7) and (C.4) in (C.5) yields the following equation:

$$\begin{aligned} R_{yy}(t_1, t_2) &= \int_{v=-\infty}^{\infty} \int_{u=-\infty}^{\infty} \frac{S_\eta}{2}\, \delta(t_1 - u - t_2 + v)\, \mathbb{1}(t_1 - u)\, \mathbb{1}(t_2 - v)\, \mathbb{1}(u)\, \mathbb{1}(v)\, du\, dv \\ &= \frac{S_\eta}{2} \cdot \int_{v=0}^{t_2} \int_{u=0}^{t_1} \delta(t_1 - u - t_2 + v)\, du\, dv \end{aligned} \tag{C.8}$$

in which the integration limits could be adapted thanks to the unit-step functions.

This integral can be worked out to be:

$$R_{yy}(t_1, t_2) = \frac{S_\eta}{2} \min(t_1, t_2) \tag{C.9}$$

where the ‘min’-function returns the smallest of its two arguments.

The variance of the random walk signal y will, thus, be:

$$\sigma_y^2(t) = R_{yy}(t, t) = \frac{S_\eta}{2} \cdot t \tag{C.10}$$

stating the well-known fact that the variance of a random walk signal is proportional to the integration time.

C.2 VCO Period Variance due to White Frequency-Noise

If we have a VCO with white noise on its control input (either due to a real noise source, or due to its own internal noise referred to the input), the output phase will be a random walk

signal superposed on a ramp with a slope of ω_{fr} (the VCO's free-running frequency). We are interested in this random walk signal because it is this deviation that causes the period errors of the VCO.

Suppose we have an ideal clock with an angular frequency of ω_{fr}; both the noisy VCO and the ideal clock are assumed to experience a positive zero-crossing at $t = 0$. We define the phase error of the noisy VCO as its phase difference with the ideal clock. Using equation (B.3), this phase error is:

$$\Delta\phi_{VCO}(t) = K_{VCO} \int_0^t v_{tune}(\tau)\, d\tau \tag{C.11}$$

If the single-sided PSD of the white noise source v_{tune} is $S_{v_{tune}}$, the variance of the phase error is, using (C.10)

$$\sigma^2_{\Delta\phi_{VCO}}(t) = \frac{K^2_{VCO} S_{v_{tune}}}{2} \cdot t \tag{C.12}$$

According to the first crossing approximation [56, 57] we can estimate zero-crossing errors of the VCO output by evaluating the phase error at the *ideal* zero-crossing moment and converting this phase error into a time error according to:

$$\Delta t = \frac{\Delta\phi_{VCO}}{2\pi} \cdot T_{VCO} = \frac{\Delta\phi_{VCO}}{\omega_{fr}} \tag{C.13}$$

with T_{VCO} the period time of the ideal clock, which is inversely proportional to ω_{fr}.

The first positive zero crossing of the ideal clock after $t = 0$ will occur at T_{VCO}. The variance of the phase error at that time will be:

$$\sigma^2_{\Delta\phi_{VCO}}(T_{VCO}) = \frac{K^2_{VCO} S_{v_{tune}}}{2} \cdot T_{VCO} \tag{C.14}$$

This means that the variance of the noisy VCO's period time $\sigma^2_{\Delta T_{VCO}}$ is, using (C.13):

$$\sigma^2_{\Delta T_{VCO}} \approx \sigma^2_{\Delta\phi_{VCO}}(T_{VCO}) \frac{T^2_{VCO}}{4\pi^2} = \frac{K^2_{VCO} S_{v_{tune}}}{8\pi^2} \cdot T^3_{VCO} \tag{C.15}$$

C.3 VCO Phase Noise and Period Jitter

Using the theory described above, it is rather straightforward to relate the measured phase noise of a VCO and its period jitter. A white noise source at the input of the VCO would cause $1/f^2$-shaped phase noise at its output. Reversely, this means that if the phase noise of the oscillator has a $1/f^2$ character, it can be input referred and be represented by a imaginary white noise source. If we know the spectrum of this noise source, the analysis above can be used to determine its period jitter.

From [2] we know that the oscillator phase noise characteristic $\mathcal{L}(f_m)$ is related to the PSD of the oscillators phase deviations $S_{\phi_{nVCO}}$ as:

$$\mathcal{L}(f_m) = \frac{S_{\phi_{nVCO}}(f_m)}{2} \tag{C.16}$$

with f_m the offset voltage with respect to the VCO center frequency.

If the VCO phase noise would be due to an input noise source v_{tune}, the PSD of phase deviations would be related to the PSD of the input noise source $S_{v_{tune}}$ according to:

$$S_{\phi_{nVCO}}(f_m) = K_{VCO}^2 \frac{S_{v_{tune}}}{4\pi^2 f_m^2} \tag{C.17}$$

This can be reversed to refer the phase noise of a VCO, originating from its internal noise sources, back to its input according to:

$$S_{v_{tune}} = \frac{4\pi^2 f_r^2 S_{\phi_{nVCO}}(f_r)}{K_{VCO}^2} = \frac{8\pi^2 f_r^2 \mathcal{L}(f_r)}{K_{VCO}^2} \tag{C.18}$$

which is indeed a white noise source (constant for all f_r) if the VCO phase noise has a $1/f^2$ character. Note that f_r denotes the offset frequency at which the VCO's phase noise is specified.

Finally, we can relate the VCO period jitter to its phase noise using (C.15):

$$\sigma^2_{\Delta T_{VCO}} = f_r^2 T_{VCO}^3 \mathcal{L}(f_r) \tag{C.19}$$

C.4 Correlation of period deviations

In Chapter 2 of this thesis, it is assumed that the deviations of the VCO period times are not correlated to one another. If the random walk mechanism is the cause of the period jitter (which is the case if the VCO phase noise has a $1/f^2$ character), it can be shown that this assumption is indeed valid.

We will show that the period deviation of the second period after $t = 0$ is not correlated to the that of the first. Proving that *all* period deviations are uncorrelated can then be done in a similar fashion.

First, we call the deviation of the first period ΔT_1, the deviation of the second ΔT_2. Using the first crossing approximation we can derive:

$$\Delta T_1 = -\frac{\Delta\phi_{VCO}(T_{VCO})}{2\pi} T_{VCO} \tag{C.20a}$$

$$\Delta T_2 = \frac{\Delta\phi_{VCO}(T_{VCO})}{2\pi} T_{VCO} - \frac{\Delta\phi_{VCO}(2T_{VCO})}{2\pi} T_{VCO} \tag{C.20b}$$

To prove these deviations are not correlated, we have to show that the expected value of their product is zero.

$$E(\Delta T_1 \cdot \Delta T_2) = \frac{T_{VCO}^2}{4\pi^2} \left[E\left\{ \Delta\phi_{VCO}(T_{VCO}) \cdot \Delta\phi_{VCO}(2T_{VCO}) \right\} \right. \\ \left. - E\left\{ \Delta\phi_{VCO}(T_{VCO}) \cdot \Delta\phi_{VCO}(T_{VCO}) \right\} \right] \quad \text{(C.21)}$$

Using (C.9) and (C.11), we can rewrite this as:

$$\begin{aligned} E(\Delta T_1 \cdot \Delta T_2) &= \frac{T_{VCO}^2}{4\pi^2} \left[R_{\Delta\phi\Delta\phi}(T_{VCO}, 2T_{VCO}) - R_{\Delta\phi\Delta\phi}(T_{VCO}, T_{VCO}) \right] = \\ &= K_{VCO}^2 \frac{T_{VCO}^2}{4\pi^2} \frac{S_{v_{tune}}}{2} \left[\min(T_{VCO}, 2T_{VCO}) - \min(T_{VCO}, T_{VCO}) \right] = \\ &= K_{VCO}^2 \frac{T_{VCO}^2}{4\pi^2} \frac{S_{v_{tune}}}{2} \left[T_{VCO} - T_{VCO} \right] = 0, \end{aligned} \quad \text{(C.22)}$$

proving that both period deviations are indeed uncorrelated.

As mentioned, a similar derivation can be used to prove that all period deviations are uncorrelated. Along these lines, it is also possible to show that all period deviations have the same variance given by (C.15).

In this appendix, the dependence of the PLL output jitter on the open-loop bandwidth ω_c is examined in the frequency domain, for both jitter due to the VCO and jitter due to the other PLL building blocks. The latter source is treated first here.

D.1 Jitter due to Equivalent Synthesizer Phase Noise

According to (3.12), the PLL output jitter due to building blocks other than the VCO and due to reference jitter can be evaluated with:

$$\sigma_{t_o}^2 = \frac{N^2}{4\pi^2 f_{VCO}^2} \int_{f_l}^{f_h} |H_{LP}(j2\pi f_m)|^2 \, S_{\phi_{neq}}(f_m) \, df_m \tag{D.1}$$

If we assume the PSD of the equivalent synthesizer phase noise is white in the frequency band of interest, this can be written as:

$$\sigma_{t_o}^2 = \frac{N^2 S_{\phi_{neq}}}{4\pi^2 f_{VCO}^2} \int_{f_l}^{f_h} |H_{LP}(j2\pi f_m)|^2 \, df_m \approx \frac{N^2 S_{\phi_{neq}}}{4\pi^2 f_{VCO}^2} \int_{0}^{\infty} |H_{LP}(j2\pi f_m)|^2 \, df_m \tag{D.2}$$

where we use $S_{\phi_{neq}}(f_m) = S_{\phi_{neq}}$, the white noise level of the equivalent synthesizer phase noise PSD. The right-hand side approximation holds as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$, as can be shown numerically [22]. The integration limits have been changed to ease the following analysis.

To examine the dependence of the integral in (D.2) on the PLL open-loop bandwidth, we start with a PLL of which we know the angular frequencies ω_c=ω_{c_0}, ω_z=ω_{z_0} and ω_p=ω_{p_0}. We then change the open-loop bandwidth and keep the PLL phase margin constant, and examine the effect this has on the integral in (D.2).

Changing the open-loop bandwidth while maintaining the value of the phase margin is described mathematically as:

$$\omega_c = \vartheta \, \omega_{c_0} \tag{D.3a}$$

$$\omega_z = \vartheta \, \omega_{z_0} \tag{D.3b}$$

$$\omega_p = \vartheta \, \omega_{p_0} \tag{D.3c}$$

with the left-hand side angular frequencies the “new” PLL open-loop bandwidth, zero and pole and ϑ the factor of change. Because all three frequencies change by the same factor, the PLL phase margin will remain constant.

Changing these three characteristic frequencies results in a new transfer function of the equivalent synthesizer phase noise, that can be written as:

$$H_{LP}(j2\pi f_m) = H_{LP_0}\left(j2\pi \frac{f_m}{\vartheta}\right) \tag{D.4}$$

with H_{LP_0} the PLL transfer function before changing the open-loop bandwidth and H_{LP} the transfer resulting from the open-loop bandwidth change. This relation can easily be verified with equations (3.5) and (3.10) of chapter 3.

Using this last equation to evaluate the integral in (D.2) results in:

$$\int_{f_m=0}^{\infty} |H_{LP}(j2\pi f_m)|^2 \, df_m = \int_{f_m=0}^{\infty} \left| H_{LP_0}\left(j2\pi \frac{f_m}{\vartheta}\right)\right|^2 df_m = \\ = \int_{u=0}^{\infty} \left|H_{LP_0}(j2\pi u)\right|^2 \vartheta \, du = \vartheta \cdot \int_{u=0}^{\infty} \left|H_{LP_0}(j2\pi u)\right|^2 du \tag{D.5}$$

where the substitution $u = f_m/\vartheta$ is used. Because u is merely a dummy integration variable, we may again substitute it, with f_m this time, which yields:

$$\int_{f_m=0}^{\infty} |H_{LP}(j2\pi f_m)|^2 \, df_m = \vartheta \cdot \int_{f_m=0}^{\infty} \left|H_{LP_0}(j2\pi f_m)\right|^2 df_m = \\ = \frac{\omega_c}{\omega_{c_0}} \cdot \int_{f_m=0}^{\infty} \left|H_{LP_0}(j2\pi f_m)\right|^2 df_m \tag{D.6}$$

using (D.3a).

From this last equation, we conclude that as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$ and the equivalent phase noise PSD in the frequency band of interest is white, the PLL output jitter due to the equivalent synthesizer phase noise is proportional to the PLL open-loop bandwidth ω_c. This holds for a fixed ratio of ω_z/ω_c and of ω_c/ω_p.

D.2 Jitter due to VCO Phase Noise

In a very similar fashion, the relation between VCO induced output jitter and PLL open-loop bandwidth can be determined. Assuming an intrinsic VCO phase noise PSD with a $1/f^2$-

shape, the output jitter caused by this phase noise according to (3.15) is:

$$\sigma_{t_o}^2 = \frac{1}{4\pi^2 f_{VCO}^2} \int_{f_l}^{f_h} |H_{VCO}(j2\pi f_m)|^2 \, S_{\phi_{nVCO}}(f_m) \, df_m =$$

$$= \frac{S_{\phi_{nVCO}}(f_r) \, f_r^2}{4\pi^2 f_{VCO}^2} \int_{f_l}^{f_h} |H_{VCO}(j2\pi f_m)|^2 \, \frac{df_m}{f_m^2} \quad \text{(D.7)}$$

using (3.20), with f_r a certain offset frequency at which the VCO phase noise PSD is specified. Again, as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$, this is approximated by:

$$\sigma_{t_o}^2 \approx \frac{S_{\phi_{nVCO}}(f_r) \, f_r^2}{4\pi^2 f_{VCO}^2} \int_{f_m=0}^{\infty} |H_{VCO}(j2\pi f_m)|^2 \, \frac{df_m}{f_m^2} \quad \text{(D.8)}$$

where the integration limits are changed to simplify the coming analysis.

Using the same procedure as before, we start with a PLL of which we know the angular frequencies $\omega_c{=}\omega_{c_0}$, $\omega_z{=}\omega_{z_0}$ and $\omega_p{=}\omega_{p_0}$. We then change the open-loop bandwidth, keeping the PLL phase margin constant according to (D.3), and examine the effect this has on the integral in (D.8).

The VCO phase noise transfer function to the PLL output after changing the open-loop bandwidth, called H_{VCO}, can be written using H_{VCO_0}, the transfer function before changing the PLL open-loop bandwidth:

$$H_{VCO}(j2\pi f_m) = H_{VCO_0}\left(j2\pi \frac{f_m}{\vartheta}\right) \quad \text{(D.9)}$$

using equations (3.5) and (3.13) of chapter 3.

The integral in (D.8) can now be evaluated:

$$\int_{f_m=0}^{\infty} |H_{VCO}(j2\pi f_m)|^2 \, \frac{df_m}{f_m^2} = \int_{f_m=0}^{\infty} \left|H_{VCO_0}\left(j2\pi \frac{f_m}{\vartheta}\right)\right|^2 \, \frac{df_m}{f_m^2} =$$

$$= \int_{u=0}^{\infty} \left|H_{VCO_0}(j2\pi u)\right|^2 \, \frac{du}{\vartheta \, u^2} = \frac{1}{\vartheta} \cdot \int_{u=0}^{\infty} \left|H_{VCO_0}(j2\pi u)\right|^2 \, \frac{du}{u^2} \quad \text{(D.10)}$$

using the substitution $u = f_m/\vartheta$. To get rid of the dummy integral variable u, we substitute f_m

back for it, yielding:

$$\int_{f_m=0}^{\infty} |H_{VCO}(j2\pi f_m)|^2 \frac{df_m}{f_m^2} = \frac{1}{\vartheta} \cdot \int_{f_m=0}^{\infty} \left|H_{VCO_0}(j2\pi f_m)\right|^2 \frac{df_m}{f_m^2} =$$

$$= \frac{\omega_{c_0}}{\omega_c} \cdot \int_{f_m=0}^{\infty} \left|H_{VCO_0}(j2\pi f_m)\right|^2 \frac{df_m}{f_m^2} \quad \text{(D.11)}$$

using (D.3a).

From this last equation, we conclude that as long as $2\pi f_l \ll \omega_c \ll 2\pi f_h$ and the VCO phase noise PSD in the frequency band of interest has a $1/f^2$-shape, the PLL output jitter due to VCO phase noise is inversely proportional to the PLL open-loop bandwidth ω_c. This holds for a fixed ratio of ω_z/ω_c and of ω_c/ω_p.

BIBLIOGRAPHY

[1] G.E. Moore, "Cramming More Components onto Integrated Circuits," *Electronics*, vol. 38, pp. 114–117, Apr. 1965.

[2] S.L.J. Gierkink, *Control Linearity and Jitter of Relaxation Oscillators*. PhD thesis, University of Twente, 1999.

[3] F. Herzel and B. Razavi, "A Study of Oscillator Jitter due to Supply and Substrate Noise," *IEEE Trans. Circuits Syst.—II*, vol. 46, pp. 56–62, Jan. 1999.

[4] B. Razavi, "Prospects of CMOS Technology for High-Speed Optical Communication Circuits," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 1135–1145, Sep. 2002.

[5] J. Cao *et al.*, "OC-192 Transmitter and Receiver in Standard 0.18-μm CMOS," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 1768–1780, Dec. 2002.

[6] H.-M. Rein and R.H. Derksen, "Integrated Bipolar 4:1 Time-Division Multiplexer for Bit Rates Up to 3 Gbit/s," *Electronics Letters*, vol. 20, pp. 546–548, June 1984.

[7] R. Reimann and H.-M. Rein, "A Bipolar 4:1 Time Division Multiplexer IC Operating up to 5.5Gb/s," in *ISSCC Dig. Tech. Papers*, pp. 186–187, Feb. 1986.

[8] M. Ida, N. Kato and T. Takada, "A 4-Gbits/s GaAs 16:1 Multiplexer/1:16 Demultiplexer LSI Chip," *IEEE Journal of Solid State Circuits*, vol. 24, pp. 928–932, Aug. 1989.

[9] J.B. Hughes *et al.*, "A Versatile ECL Multiplexer IC for the Gbit/s Range," *IEEE Journal of Solid State Circuits*, vol. SC-14, pp. 812–817, Oct. 1979.

[10] H.-M. Rein *et al.*, "A Time Division Multiplexer IC for Bit Rates Up to About 2 Gbits/s," *IEEE Journal of Solid State Circuits*, vol. SC-19, pp. 306–310, June 1984.

[11] R.C. Walker, K.-C. Hsieh, T.A. Knotts and C.-S. Yen, "A 10Gb/s Si-Bipolar TX/RX Chipset for Computer Data Transmission," in *ISSCC Dig. Tech. Papers*, pp. 302–303, Feb. 1998.

[12] T. Harada *et al.*, "Si Bipolar Multiplexer, Demultiplexer and Prescaler ICs for 10Gb/s SONET Systems," in *ISSCC Dig. Tech. Papers*, pp. 154–155, Feb. 1993.

[13] A. Momtaz *et al.*, "A Fully Integrated SONET OC-48 Transceiver in Standard CMOS," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 1964–1973, Dec. 2001.

[14] Telcordia Technologies, GR-253-CORE, "Synchronous Optical Network (SONET) transport systems: Common generic criteria." Issue 3, Sept. 2000.

[15] H.-I. Cong *et al.*, "A 10-Gb/s 16:1 Multiplexer and 10-Ghz Clock Synthesizer in 0.25-μm SiGe BiCMOS," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 1946–1953, Dec. 2001.

[16] J.D. van der Tang and C.S. Vaucher, "Design and Optimization of a Low Jitter Clock-Conversion PLL for SONET/SDH Optical Transmitters," in *IEEE International Conference on Electronics, Circuits and Systems*, vol. 1, pp. 31–34, Sep. 2001.

[17] I.A. Young, M.F. Mar and B. Bhushan, "A 0.35μm CMOS 3-880MHz PLL N/2 Clock Multiplier and Distribution Network with Low Jitter for Microprocessors," in *ISSCC Dig. Tech. Papers*, pp. 330–331, Feb. 1997.

[18] S. Tam *et al.*, "Clock Generation and Distribution for the First IA-64 Microprocessor," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 1545–1552, Nov. 2000.

[19] N.A. Kurd *et al.*, "A Multigigahertz Clocking Scheme for the Pentium® 4 Microprocessor," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 1647–1653, Nov. 2001.

[20] V. Gutnik and A.P. Chandrakasan, "Active GHz Clock Network Using Distributed PLLs," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 1553–1560, Nov. 2000.

[21] B. Razavi, *RF Microelectronics*. Prentice Hall, 1998.

[22] C.S. Vaucher, *Architectures for RF Frequency Synthesizers*. Boston,MA: Kluwer, 2002.

[23] B. Kim, T.C. Weigandt and P.R. Gray, "PLL/DLL System Noise Analysis for Low Jitter Clock Synthesizer Design," in *Proc. Int. Symp. On Circuits and Systems*, June 1994.

[24] T.H. Lee *et al.*, "A 2.5 V CMOS Delay-Locked Loop for an 18 Mbit, 500 Megabyte/s DRAM," *IEEE Journal of Solid State Circuits*, vol. 29, pp. 1491–1496, Dec. 1994.

[25] M.G. Johnson and E.L. Hudson, "A Variable Delay Line PLL for CPU-Coprocessor Synchronization," *IEEE Journal of Solid State Circuits*, vol. 23, pp. 1218–1223, Oct. 1988.

[26] G. Chien and P.R. Gray, "A 900-MHz Local Oscillator Using a DLL-Based Frequency Multiplier Technique for PCS Applications," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 1996–1999, Dec. 2000.

[27] D.J. Foley and M.P. Flynn, "CMOS DLL-Based 2-V 3.2-ps Jitter 1-GHz Clock Synthesizer and Temperature-Compensated Tunable Oscillator," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 417–423, Mar. 2001.

[28] C. Kim, I.-C. Hwang and S.-M. Kang, "Low-Power Small-Area ±7.28ps Jitter 1GHz DLL-Based Clock Generator," in *ISSCC Dig. Tech. Papers*, pp. 142–143, Feb. 2002.

[29] B. Razavi, *Monolithic Phase-Locked Loops and Clock Recovery Circuits: Theory and Design*, ch. Design of Monolithic Phase-Locked Loops and Clock Recovery Circuits — A Tutorial. IEEE Press, 1996.

[30] B.W. Garlepp *et al.*, "A Portable Digital DLL for High-Speed CMOS Interface Circuits," *IEEE Journal of Solid State Circuits*, vol. 34, pp. 632–644, May 1999.

[31] C.S. Vaucher, "An Adaptive PLL Tuning System Architecture Combining High Spectral Purity and Fast Settling Time," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 490–502, Apr. 2000.

[32] T. Saeki, M.Mitsuishi, H. Iwaki and M. Tagishi, "A 1.3-Cycle Lock Time, Non-PLL/DLL Clock Multiplier Based on Direct Clock Cycle Interpolation for "Clock on Demand"," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 1581–1590, Nov. 2000.

[33] C.A. Sharpe, "A 3-state Phase Detector can improve your next PLL Design," *EDN*, pp. 55–59, 1976.

[34] R.C.H. van de Beek, E.A.M. Klumperink, C.S. Vaucher and B. Nauta, "Low-jitter clock multiplication: a comparison between PLLs and DLLs," *IEEE Trans. Circuits Syst.—II*, vol. 49, pp. 555–566, August 2002.

[35] W.P. Robins, *Phase Noise in Signal Sources*. IEE Telecommunications series, London, 2^{nd} ed., 1996.

[36] G. Chien, *Low-Noise Local Oscillator Design Techniques using a DLL-based Frequency Multiplier for Wireless Applications*. PhD thesis, University of California, Berkeley, Jan. 2000.

[37] B. Nauta, "Analog CMOS Low Power Design Considerations." Low Power Workshop on ESSCIRC 1996 Conference, Sep. 1996.

[38] J.G. Maneatis, "Low-Jitter Process-Independent DLL and PLL Based on Self-Biased Techniques," *IEEE Journal of Solid State Circuits*, vol. 31, pp. 1723–1732, Nov. 1996.

[39] M.-J.E. Lee *et al.*, "Jitter Transfer Characteristics of Delay-Locked Loops—Theories and Design Techniques," *IEEE Journal of Solid State Circuits*, vol. 38, pp. 614–621, Apr. 2003.

[40] F.M. Gardner, "Charge-pump phase-locked-loops," *IEEE Transactions on Communications*, vol. COM-28, pp. 1849–1858, Nov. 1980.

[41] J.P. Hein and J.W. Scott, "z-Domain Model for Discrete-Time PLL's," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1393–1400, Nov. 1988.

[42] C.S. Vaucher and D. Kasperkovitz, "A Wide-Band Tuning System for Fully Integrated Satellite Receivers," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 987–997, July 1998.

[43] L. Lin, L. Tee and P.R. Gray, "A 1.4GHz Differential Low-Noise CMOS Frequency Synthesizer using a Wideband PLL Architecture," in *ISSCC Dig. Tech. Papers*, pp. 204–205, Feb. 2000.

[44] T.C. Weigandt, B. Kim and P.R. Gray, "Timing Jitter Analysis for High-Frequency, Low-Power CMOS Ring-Oscillator Design," in *Proc. Int. Symp. On Circuits and Systems*, June 1994.

[45] R. Farjad-rad *et al.*, "A Low-Power Multiplying DLL for Low-Jitter Multigigahertz Clock Generation in Highly Integrated Digital Chips," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 1804–1812, Dec. 2002.

[46] S. Ye, L. Jansson and I. Galton, "A Multiple-Crystal Interface PLL with VCO Realignment to Reduce Phase Noise," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 1795–1803, Dec. 2002.

[47] S. Kuboki, K. Kato, N. Miyakawa and K. Matsubara, "Nonlinearity Analysis of Resistor String A/D Converters," *IEEE Transactions on Circuits and Systems*, vol. CAS-29, pp. 383–390, June 1982.

[48] J.-S. Lee, M.-S. Keel, S.-I. Lim and S. Kim, "Charge Pump with perfect Current Matching Characteristics in Phase-Locked Loops," *Electronics Letters*, vol. 36, pp. 1907–1908, Nov. 2000.

[49] M.J.M. Pelgrom, A.C.J. Duinmaijer and A.P.G. Welbers, "Matching Properties of MOS transistors," *IEEE Journal of Solid State Circuits*, vol. 24, pp. 1433–1440, Oct. 1989.

[50] A. Hastings, *The Art of Analog Layout*. Prentice Hall, 2001.

[51] P. Larsson, "A Simulator Core for Charge-Pump PLL's," *IEEE Trans. Circuits Syst.—II*, vol. 45, pp. 1323–1326, Sep. 1998.

[52] M.S. Keshner, "$1/f$ Noise," *Proceedings of the IEEE*, vol. 70, pp. 212–218, Mar. 1982.

[53] F.N. Hooge and P.A. Bobbert, "On the Correlation Function of $1/f$ Noise," *Physica B*, no. 239, pp. 223–230, 1997.

[54] K. Dutton, S. Thompson, B. Barraclough, *The Art of Control Engineering*. Addison-Wesley, 1997.

[55] A. Hajimiri and T.H. Lee, "A General Theory of Phase Noise in Electrical Oscillators," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 179–194, Feb. 1998.

[56] A.A. Abidi and R.G. Meyer, "Noise in Relaxation Oscillators," *IEEE Journal of Solid State Circuits*, vol. SC-18, pp. 794–802, Apr. 1995.

[57] T.C. Weigandt, *Low-Phase-Noise, Low-Timing-Jitter Design Techniques for Delay Cell Based VCOs and Frequency Synthesizers*. PhD thesis, University of California, Berkeley, 1998.

[58] I. Thompson and P.V. Brennan, "Phase Noise Contribution of the Phase/Frequency Detector in a Digital PLL Frequency Synthesizer," *IEE Proc.-Circuits Devices Syst.*, vol. 150, pp. 1–5, Feb. 2003.

[59] V.F. Kroupa, "Jitter and Phase Noise in Frequency Dividers," *IEEE Trans. on Instrumentation and Measurement*, vol. 50, pp. 1241–1243, Oct. 2001.

[60] J.L. Gonzalez and A. Rubio, "Low delta-I Noise CMOS Circuits based on Differential Logic and Current Limiters," *IEEE Trans. Circuits Syst.—I*, vol. 46, pp. 872–876, July 1999.

[61] M. Anis, M. Allam and M. Elmasry, "Impact of Technology Scaling on CMOS Logic Styles," *IEEE Trans. Circuits Syst.—II*, vol. 49, pp. 577–588, Aug. 2002.

[62] C.S. Vaucher *et al.*, "A Family of Low-Power Truly Modular Programmable Dividers in Standard 0.35-μm CMOS Technology," *IEEE Journal of Solid State Circuits*, vol. 35, pp. 1039–1045, July 2000.

[63] P. Larsson, "Measurements and Analysis of PLL Jitter Caused by Digital Switching Noise," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 1113–1119, July 2001.

[64] B. Kim, D.N. Helman and P.R. Gray, "A 30-MHz Hybrid Analog/Digital Clock Recovery Circuit in 2-μm CMOS," *IEEE Journal of Solid State Circuits*, vol. 25, pp. 1385–1394, Dec. 1990.

[65] M. Yamashina and H. Yamada, "An MOS Current Mode Logic (MCML) Circuit for Low-Power sub-GHz Processors," in *IEICE Trans. Electron.*, pp. 1181–1187, Oct. 1992.

[66] S.R. Maskai, S. Kiaei and D.J. Allstot, "Synthesis Techniques for CMOS Folded Source-Coupled Logic Circuits," *IEEE Journal of Solid State Circuits*, vol. 27, pp. 1157–1167, Aug. 1992.

[67] J. Kundan and S.M. Rezaul Hasan, "Enhanced Folded Source-Coupled Logic Technique for Low-Voltage Mixed-Signal Integrated Circuits," *IEEE Trans. Circuits Syst.—II*, vol. 47, pp. 810–817, Aug. 2000.

[68] M. Mizuno *et al.*, "A GHz MOS Adaptive Pipeline Technique Using MOS Current-Mode Logic," *IEEE Journal of Solid State Circuits*, vol. 31, pp. 784–791, June 1996.

[69] H.P. Hendrikse, "ΔI-Noise Generation in Digital Implementations (a Comparison)," Master's thesis, University of Twente, Enschede, the Netherlands, 2003.

[70] M.W. Allam and M.I. Elmasry, "Dynamic Current Mode Logic (DyCML): A New Low-Power High-Performance Logic Style," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 550–558, Mar. 2001.

[71] A. Tanebe *et al.*, "0.18-μm CMOS 10-Gb/s Multiplexer/Demultiplexer ICs Using Current Mode Logic with Tolerance to Threshold Voltage Fluctuation," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 988–996, June 2001.

[72] J. Yuan and C. Svensson, "High-Speed CMOS Circuit Technique," *IEEE Journal of Solid State Circuits*, vol. 24, pp. 62–70, Feb. 1989.

[73] D. Pfaff and Q. Huang, "A Quarter-Micron CMOS, 1GHz VCO/Prescaler-Set For Very Low Power Applications," in *Custom Integrated Circuits Conference, San Diego*, pp. 649–652, May 1999.

[74] S. Pellerano, C. Samori, S. Levantino and A.L. Lacaita, "13.5-mW, 5-GHz WLAN, CMOS Frequency Synthesizer Using a True Single Phase Clocking Divider," in *Symp. VLSI Circuits Dig. Tech. Papers*, pp. 145–148, June 2003.

[75] C.-Y. Yang, G.-K. Dehng, J.-M. Hsu and S.-I. Liu, "New Dynamic Flip-Flops for High-Speed Dual-Modulus Prescaler," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 1568–1571, Oct. 1998.

[76] H. Kondoh *et al.*, "A 1.5-V 250-MHz to 3.0-V 622-MHz operation CMOS Phase-Locked Loop with Precharge type Phase-Detector," *IEICE Trans. Electron.*, vol. E78-C, pp. 381–388, Apr. 1995.

[77] H.O. Johansson, "A Simple Precharged CMOS Phase Frequency Detector," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 295–299, Feb. 1998.

[78] M. Alioto and G. Palumbo, "Power-Delay Trade-Offs in SCL Gates," in *Proc. Int. Symp. On Circuits and Systems*, June 2002.

[79] M. Alioto and G. Palumbo, "Design Strategies for Source coupled Logic Gates," *IEEE Trans. Circuits Syst.—I*, vol. 50, pp. 640–654, May 2003.

[80] J.F. Parker and D. Ray, "A 1.6-GHz CMOS PLL with On-Chip Loop Filter," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 337–343, Mar. 1998.

[81] J. Craninckx and M.S.J. Steyaert, "A 1.8-GHz CMOS Low-Phase-Noise Voltage-Controlled Oscillator with Prescaler," *IEEE Journal of Solid State Circuits*, vol. 30, pp. 1474–1482, Dec. 1995.

[82] J. Craninckx and M.S.J. Steyaert, "A 1.75-GHz/3-V Dual-Modulus Divide-by-128/129 Prescaler in 0.7-μm CMOS," *IEEE Journal of Solid State Circuits*, vol. 31, pp. 890–897, July 1996.

[83] H. Knapp, H.-D. Wohlmuth, M. Wurzer and M. Rest, "25GHz Static Frequency Divider and 25Gb/s Multiplexer in 0.12μm CMOS," in *ISSCC Dig. Tech. Papers*, pp. 302–303, Feb. 2002.

[84] C.-M. Hung and K.K. O, "A Fully Integrated 1.5-V 5.5-GHz CMOS Phase-Locked Loop," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 521–525, Apr. 2002.

[85] T. Sakurai and A.R. Newton, "Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas," *IEEE Journal of Solid State Circuits*, vol. 25, pp. 584–594, Apr. 1990.

[86] K.A. Bowman *et al.*, "A Physical Alpha-Power Law MOSFET Model," *IEEE Journal of Solid State Circuits*, vol. 34, pp. 1410–1414, Mar. 1999.

[87] U.L. Rohde, *Microwave and Wireless Synthesizers*. Wiley-Interscience, 1997.

[88] M. Soyuer and R.G. Meyer, "Frequency Limitations of a Conventional Phase-Frequency Detector," *IEEE Journal of Solid State Circuits*, vol. 25, pp. 1019–1022, Aug 1990.

[89] T.H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*. Cambridge University Press, 1998.

[90] A. Pottbäcker, U. Langmann and H.-U. Schreiber, "A Si bipolar phase and frequency detector IC for clock extraction up to 8 Gb/s," *IEEE Journal of Solid State Circuits*, vol. 27, pp. 1747–1751, Dec. 1992.

[91] S.B. Anand and B. Razavi, "A CMOS Clock Recovery Circuit for 2.5-Gb/s NRZ Data," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 432–439, Mar. 2001.

[92] B. Nauta, R.C.H. van de Beek and C.S. Vaucher, "Phase-Locked-Loop with reduced Clock Jitter." Patent Nr. WO03065586, 2003.

[93] R.C.H. van de Beek, E.A.M. Klumperink, B. Nauta and C.S. Vaucher, "A very fast Duty Cycle independent Phase Detector with low reference breakthrough." Patent Nr. PHNL020803, 2003.

[94] R.C.H. van de Beek, C.S. Vaucher, D.M.W. Leenaerts, N. Pavlovic, K. Mistry, E.A.M. Klumperink and B. Nauta, "A 2.5 to 10GHz Clock Multiplier Unit with 0.22ps RMS Jitter in a 0.18μm CMOS Technology," in *ISSCC Dig. Tech. Papers*, pp. 178–179, Feb. 2003.

[95] G.W. den Besten, "Precision Clock Generation by means of a CMOS Phase-Locked Loop," Master's thesis, University of Twente, Enschede, the Netherlands, 1994.

[96] M. Horowitz, C.-K Ken Yang and S. Sidiropoulos, "High-Speed Electrical Signaling," *IEEE Micro*, vol. 18, pp. 12–24, Jan. 1998.

[97] M. Mansuri, D. Liu and C.-K Ken Yang, "Fast Frequency Acquisition Phase-Frequency Detectors for GSamples/s Phase-Locked Loops," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 1331–1334, Oct. 2002.

[98] K. Ishii, K. Kishine and H. Ichino, "A Jitter Suppression Technique for a 2.48832-Gb/s Clock and Data Recovery Circuit," *IEEE Trans. Circuits Syst.—II*, vol. 49, pp. 266–272, Apr. 2002.

[99] C.P. Yue and S.S. Wong, "On-chip Spiral Inductors with Patterned Ground Shields for Si-Based RF IC's," *IEEE Journal of Solid State Circuits*, vol. 33, pp. 743–752, May 1998.

[100] D. Leenaerts and N. Pavlovic, "Design of wireless LAN circuits in RF-CMOS," in *Workshop on Advances in Analog Circuit Design (AACD)*, Mar. 2002.

[101] D. Ham and A. Hajimiri, "Concepts and methods in optimization of integrated LC VCOs," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 896–909, June 2001.

[102] R. Aparicio and A. Hajimiri, "Capacity Limits and Matching Properties of Integrated Capacitors," *IEEE Journal of Solid State Circuits*, vol. 37, pp. 384–393, Mar. 2002.

[103] M.M. Green *et al.*, "OC-192 Transmitter in Standard 0.18μm CMOS," in *ISSCC Dig. Tech. Papers*, pp. 248–249, Feb. 2002.

[104] S.S. Hastings, *An Introduction to Analog and Digital Communications*. John Wiley & Sons, 1989.

SAMENVATTING

Dit proefschrift beschrijft belangrijke overwegingen bij het ontwerpen van een klokvermenigvuldiger, een bouwblok dat een periodiek signaal genereert met een frequentie die een veelvoud is van een inkomend periodiek referentiesignaal. Klokvermenigvuldigers worden toegepast in digitale ICs, om de interne klok te genereren, en bij het serializeren van parallele datastromen. In het proefschrift wordt veel nadruk gelegd op de zuiverheid van het gegenereerde signaal (elke periode van het uitganssignaal heeft idealiter een gelijke tijdsduur). Met behulp van tijddiscrete analyse wordt de beste architectuurkeuze gemotiveerd. Omdat de periodieke zuiverheid het beste is als de frequentie van het referentiesignaal hoog is, wordt onderzocht hoe een terugkopellus te realiseren is die op een zo hoog mogelijke vergelijkingsfrequentie opereert.

Eén van eerste overwegingen bij het ontwerpen van een klokvermenigvuldiger is de meest geschikte architectuur. De keuze die het meest voor de hand ligt is de zogenaamde integer-*N* Phase Locked Loop (PLL), welke een oscillator bevat die de uitgangsklok genereert. Een regellus zorgt er enerzijds voor dat de oscillatiefrequentie een veelvoud is van de referentiefrequentie en anderzijds dat door ruis veroorzaakte tijdonnauwkeurigheden gedeeltelijk onderdrukt worden. Een andere klokvermenigvuldiger is de Delay Locked Loop (DLL) met Edge Combiner. Deze gebruikt een “delay line” die de referentieklok precies één periodetijd vertraagt. Deze vertragingstijd wordt door een op een PLL gelijkende regellus bewerkstelligd. Door de delay line op te bouwen met identieke “delay cells” worden verschillende fasen van de referentieklok verkregen, die gecombineerd worden tot de hoogfrequente uitgangsklok.

In hoofstuk 2 worden beide architecturen met elkaar vergeleken op grond van de tijdonnauwkeurigheden die optreden in de gegenereerde uitgansklok. Zowel de invloed van de belangrijkste ruisbronnen alsmede de gevolgen van “mismatch” worden analytisch beschouwd, zodat een gemotiveerde keuze voor een architectuur gemaakt kan worden. Berekeningen en ondersteunende simulaties laten zien dat een PLL veelal te verkiezen is boven een DLL. Wanneer de oscillator in de PLL een ring-oscillator is die bestaat uit delay elementen soortgelijk aan de delay line elementen van de DLL, is het vermogen dat per element gebruikt wordt i.h.a. hoger in het geval van de PLL zodat de jitter per delay cell kleiner blijft. Dit effect is typisch sterker dan de accumulatie van tijdonnauwkeurigheden die optreedt in de oscillator van de PLL. Als een *LC*-oscillator gebruikt wordt in de PLL, is de PLL nog meer in het voordeel. Ook wanneer effecten van mismatch in ogenschouw worden genomen blijkt de PLL beter te presteren.

Hoofdstuk 2 beschrijft tevens een analoge ontwerpmethode die snelheid (bandbreedte) en lineariteit van een circuit loskoppelt van de effecten van mismatch en ruis: W-schaling ofwel “impedance level scaling”. Deze methode levert het inzicht dat jitter (of deze nu veroorzaakt wordt door ruis of door mismatch) en vermogensverbruik van een “delay cell” uitruilbaar zijn.

Volgens de belangrijkste conclusie van hoofdstuk 2 is een PLL de veelbelovendste archi-

tectuur om een klokvermenigvuldiger te realiseren. Daarom worden in hoofdstuk 3 enkele belangrijke ontwerpaspecten van een PLL besproken: het lusfilter en minimalizatie van jitter. Omdat grofweg de helft van de jitter veroorzaakt wordt door de regellus die om de oscillator opgebouwd wordt (en de referentie-jitter), wordt bekeken hoe deze jitter-bijdrage zo laag mogelijk gehouden kan worden. Zowel een hoge referentiefrequentie als het gebruik van een fasedetector die zo kort mogelijke pulsen genereert blijken zowel tot lage jitter als een klein lusfilteroppervlak te leiden. Het blijkt dat lusfilteropervlak en jitter uitgeruild kunnen worden.

Een belangrijk deel van de PLL terugkoppellus (de frequentiedeler en de fasedetector) bestaat uit digitale bouwblokken. Dankzij haar lage gevoeligheid voor storingen in de voedingsspanning alsmede lage generatie van deze storing, is de differentiële CMOS current mode logic (CML) familie een geschikte kandidaat om deze digitale blokken mee te implementeren. Hoofdstuk 4 beschrijft een eenvoudige en inzichtelijke methode om digitale CML poorten te dimensioneren, gebaseerd op de maximaal stuurbare staartstroom.

Omdat hoofdstuk 3 laat zien dat een hoge referentiefrequentie en korte pulsen uit de fasedetector gunstig zijn voor de jitter in de gegenereerde klok, alsmede voor het lusfilteroppervlak, worden in hoofdstuk 5 snelle fasedetectoren onderzocht. Eerst wordt een vergelijking gemaakt tussen de belangrijkste bestaande fasedetectoren, waaruit blijkt dat de populaire "tri-state phase-frequency detector" (PFD) een gunstige keuze is. De interne terugkoppellus, nodig om de detector in zijn neutrale toestand te brengen, zorgt echter voor een snelheidsbeperking. Een eenvoudige fasedetector, bestaande uit 2 EN-poorten, wordt gepresenteerd. Deze detector genereert sturende pulsen die, net als bij een PFD, geen netto activiteit vertonen wanneer de PLL "in lock" is. Vanwege zijn eenvoud en de afwezigheid van een interne reset-lus, kan de voorgestelde detector op veel hogere snelheden opereren en genereert deze detector veel smallere pulsen dan een PFD, wat gunstig is voor de jitter in de gegenereerde klok.

Hoofdstuk 6 beschrijft de implementatie van een complete klokvermenigvuldiger-chip, geïmplementeerd in een standaard 0.18μm CMOS proces. De chip genereert een zuiver 10 GHz kloksignaal gebaseerd op een referentieklok van 2.5 GHz. De 10 GHz klok wordt verkregen uit een geïntegreerde *LC*-oscillator. De terugkoppellus is gebaseerd op de simpele 2-EN-poort fasedetector. Voor correct opstartgedrag is een frequentiedetector gebruikt die geen invloed heeft op de jitter van de uitgangsklok doordat deze vanzelf inactief wordt als de PLL "in lock" is. De rms-jitter van de 10 GHz klok bedraagt slechts 0.22 ps (2.2 mUI$_{RMS}$) en de peak-to-peak jitter bedraagt 2.2 ps (22 mUI$_{P-P}$). Dit ligt ruim beneden de OC-192 SONET specificatie.

DANK JULLIE!

Al kan het soms eenzaam werk zijn, een AIO-onderzoek doe je niet alleen. Vaak genoeg zat ik met mijn handen in mijn toch al warrige en ietwat lange haar, maar dankzij de hulp van onderstaande mensen wist ik tegenslagen als een leerzame ervaring achter me te laten. Bij deze wil ik hen dan ook graag bedanken:

- Bram Nauta, mijn promotor, in de eerste plaats omdat je me (zonder al te veel moeite) overtuigd hebt van het feit dat ik eigenlijk heel graag AIO wilde worden. Maar natuurlijk ook voor je creatieve ideeën en onze voor mij vruchtbare gesprekken.
- Eric Klumperink, mijn assistent promotor en dagelijkse begeleider, voor de enorm inspirerende discussies die ik met je gevoerd heb en de broodnodige sturing die je me gegeven hebt. Je enthousiasme was besmettelijk!
- Cicero Vaucher, voor je indrukwekkende kennis en inzicht. Je hebt me meer bijgebracht over frequentiesynthese dan welk boek ook gekund zou hebben. Jij moet overigens wel de meest relaxte Braziliaan zijn die ik ken...
- Domine Leenaerts, voor je onmisbare inspanning en geweldige hulp tijdens het layouten. Wat smaakt zo'n fles champagne nog goed dan, net bier...
- Nenad Pavlovic en Ketan Mistry, voor het ontwerpen van onmisbare bouwblokken op de multiplier chip.
- Kees en Jannie, mijn lieve ouders. Dit boekje is mede mogelijk gemaakt door deze geweldige mensen waar ik onvoorwaardelijk en altijd op kan terugvallen.
- Eva, mijn vriendin, die mijn leven zoveel meer spanning en ontspanning geeft.
- Mijn broer Erik, voor het maken van Yggdrasil op de voorkant. Ik heb veel respect voor jouw eigen carrière.
- Ilma, want hoe zou ik mijn allerliefste zusje nou kunnen vergeten?
- Mijn meest nerdy kamergenoot Simon. Ook al is *rek'ng* gevonden, het kan nooit kwaad om door te zoeken. Room 3?
- Mijn eigenwijste kamergenoot Jan Rutger 'Shredder'. Howzabout halibutt in de crap shack? Sorry voor de muziek...
- Mijn Italiaanste kamergenoot Federico. Je was me net voor!
- Het uiterst behulpzame secretariaat van vloer 3 en de evenzo onmisbare systeembeheerders.
- Henk de Vries en Gerard Wienk, voor het onmogelijke mogelijk maken in het 'meethok'.
- En natuurlijk überheerschend studentenhuis WG, waar ik de tofste tijd gehad heb die ik had kunnen wensen. Wat zullen we drinken? Ik voel me zo verdomd alleen...

OVER DE AUTEUR

Remco van de Beek werd geboren op 3 september 1974, te Wageningen. Na het behalen van zijn VWO-diploma op het "Christelijk Streeklyceum Ede" in 1992, is hij in datzelfde jaar Elektrotechniek gaan studeren op de Universiteit Twente in Enschede. Zijn afstudeeropdracht, die in samenwerking met het Philips NatLab te Eindhoven uitgevoerd werd, betrof het ontwerpen van een differentiële Track&Hold versterker, gebruikmakend van een hiërarchische top-down methode gebaseerd op 'behavioral modelling'. Na het verkrijgen van zijn M.Sc. degree in 1999 aanvaardde hij een AIO-positie in de vakgroep IC Design op de Universiteit Twente, onder leiding van prof.dr.ir. Bram Nauta. Het resultaat van deze werkzaamheden is beschreven in dit proefschrift. Remco is momenteel werkzaam in de Integrated Transceiver groep van het Philips NatLab te Eindhoven.