Fast Automatic Sizing of a Charge Pump Phase-Locked Loop based on Behavioral Models

J. Zou, D. Mueller,
H. Graeb, U. Schlichtmann,
E. Hennig, R. Sommer

Institute of Electronic Design Automation, TU Munich,
Infineon Technologies AG
Munich, Germany

Outline

- Analog Hierarchical Design Methodology
- Phase-Locked Loops - Introduction
  - Architecture Selection
  - PLL Building Blocks
  - PLL Specifications
- Automatic Hierarchical Sizing of a Charge Pump PLL
  - Definition & Propagation of Specifications
  - Automatic Behavioral Level Sizing
  - Automatic Circuit Level Sizing
  - Design Verification
- Conclusion & Outlook

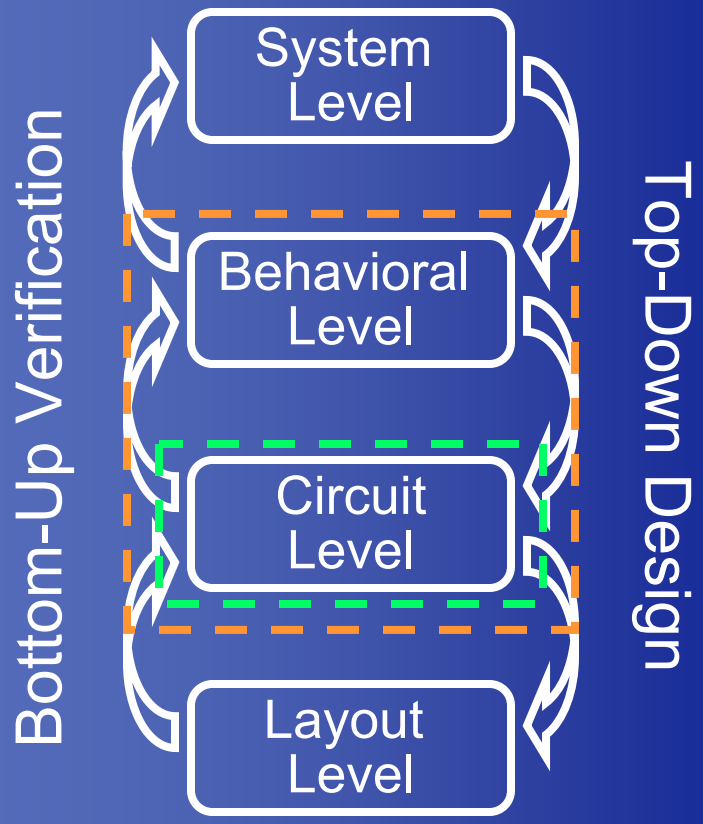
Analog Hierarchical Design Levels

Top-Down :

- Circuit Decomposition & Architecture Selection
- Specifications Propagation & Sizing Process

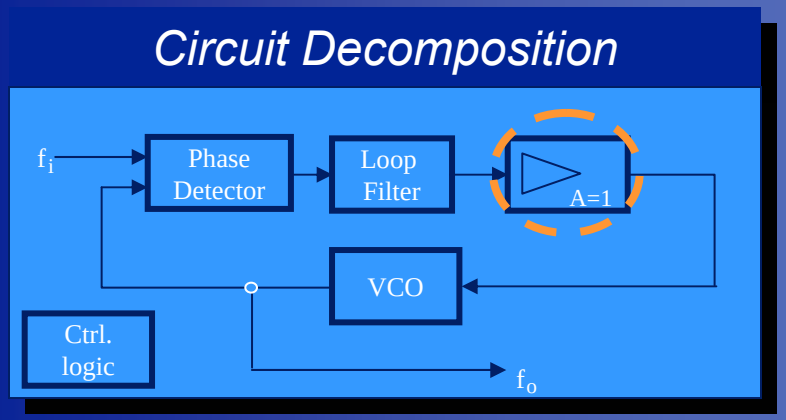
Bottom-Up :

- Verification Process
- Design Space Exploration

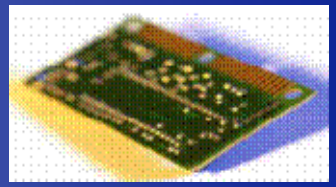


Our work focuses on the automatic sizing processes on the both levels: behavioral and circuit levels.

Elements of the Analog Hierarchical Design Methodology

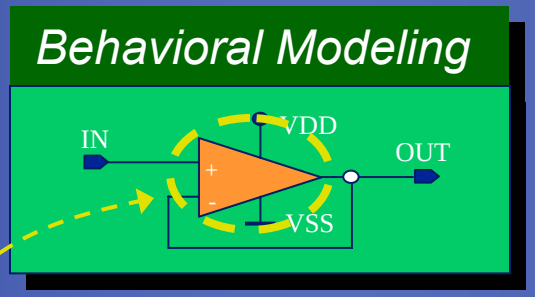


PLL Circuit



time-consuming

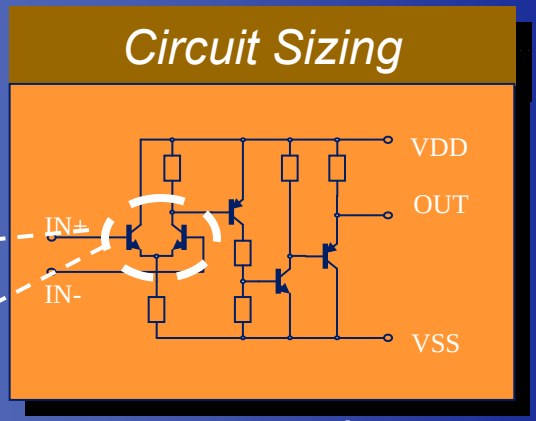
Specifications Propagation



Design Space Exploration

B=[gain, bandwidth,...]

P=[w1,w2,l1,l2,...]



automatic

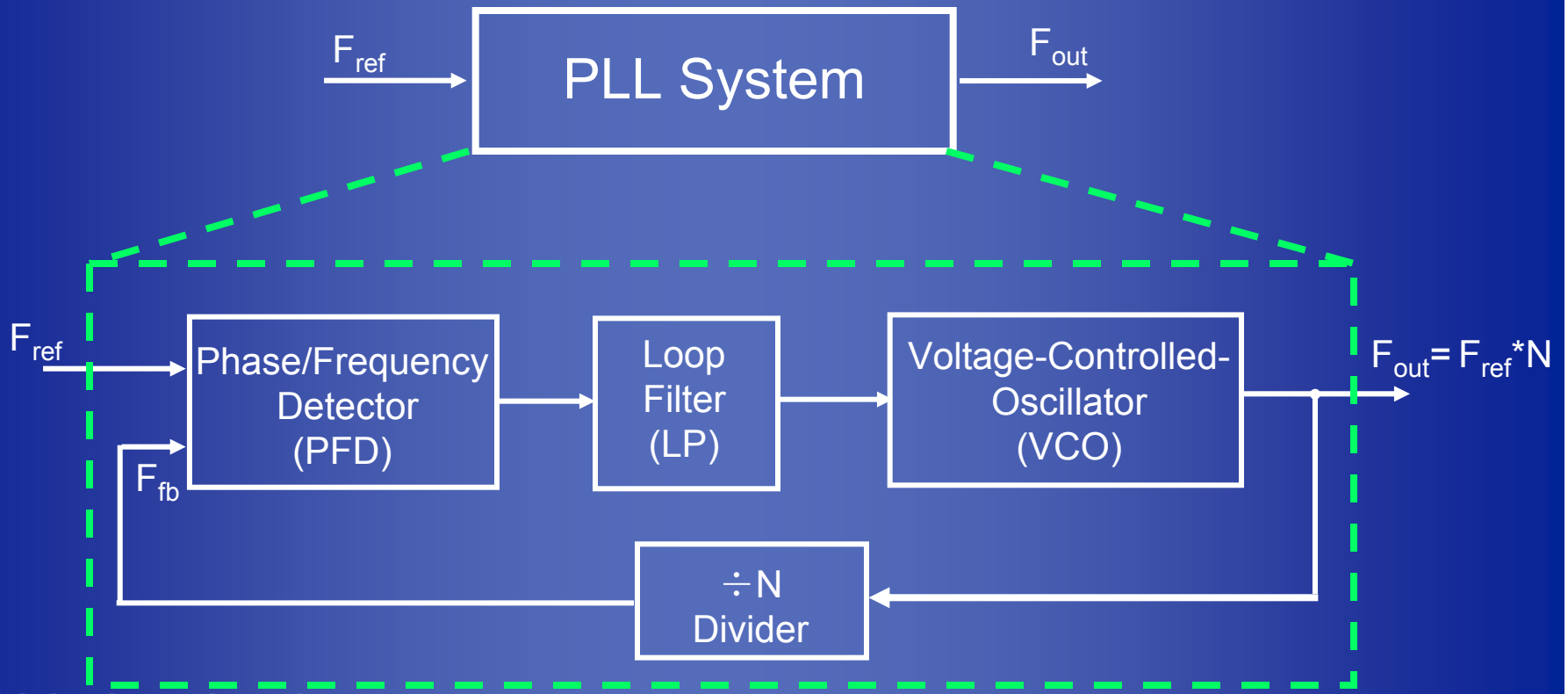
Outline

- Analog Hierarchical Design Methodology
- Phase-Locked Loops Introduction
  - Architecture Selection
  - PLL Building Blocks
  - PLL Specifications
- Automatic Hierarchical Sizing of a Charge Pump PLL
  - Definition & Propagation of Specifications
  - Automatic Behavioral Level Sizing
  - Automatic Circuit Level Sizing
  - Design Verification
- Conclusion & Outlook

PLL as Example Circuit for Analog Hierarchical Design

- PLL Function

A phase-locked loop (PLL) is a control system to generate an output signal which is synchronized in frequency and phase to an input signal.

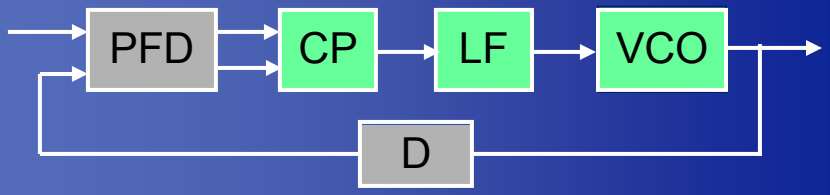


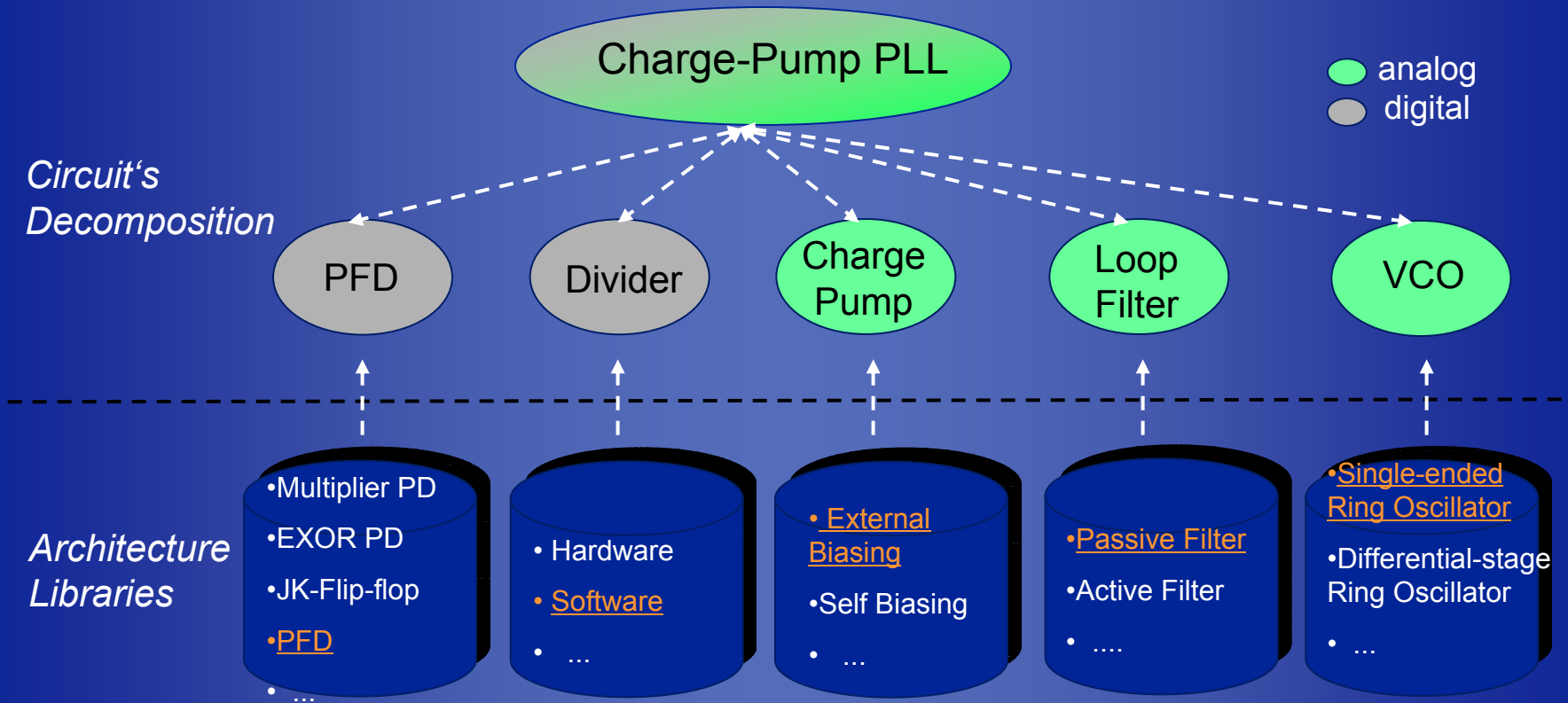
- PLL Application

Clock Recovery / Frequency Synthesis / .....

Charge Pump Phase-Locked Loop (CP PLL)

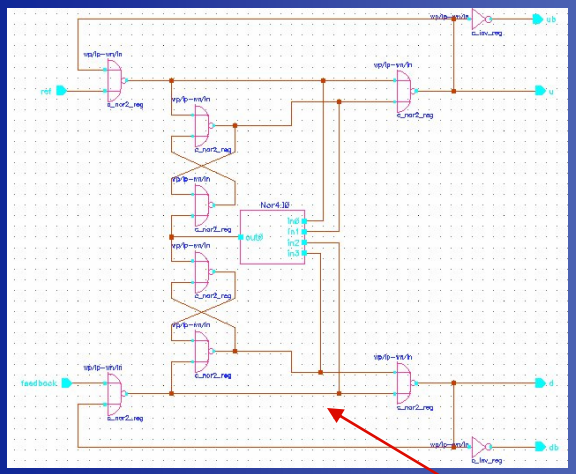
Charge-pump PLL can provide zero phase error, an extended frequency range of operation and low cost.



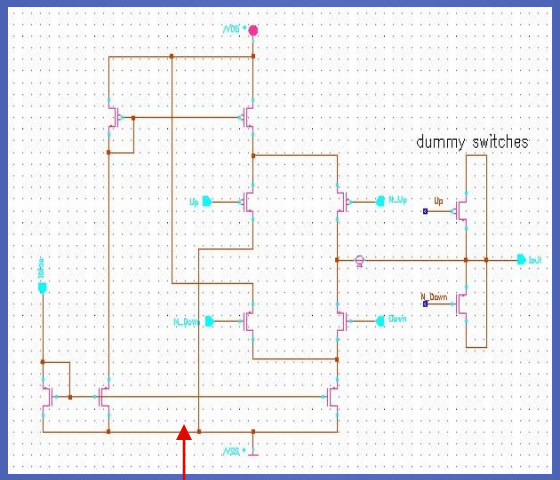


CP PLL Building Blocks on Circuit Level

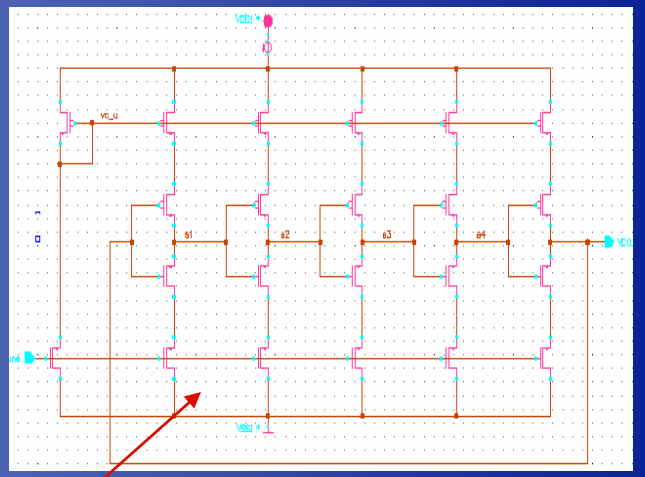
digial PFD

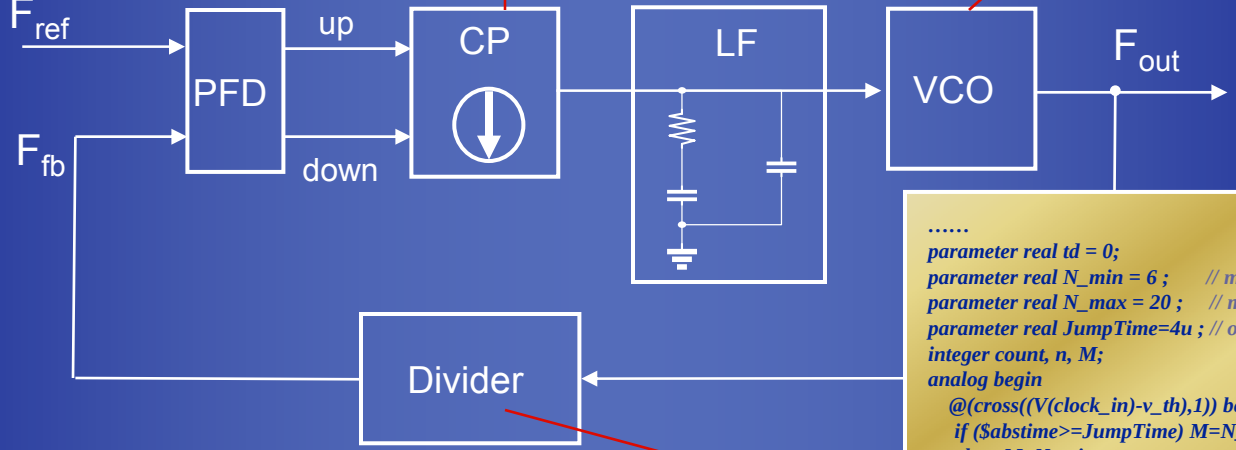


external biased CP



five-stage single-ended VCO






```
……
parameter real td = 0;
parameter real N_min = 6 ;      // minimum divider value
parameter real N_max = 20 ;     // maximum divider value
parameter real JumpTime=4u ; // output frequency jump time
integer count, n, M;
analog begin
   @(cross((V(clock_in)-v_th),1)) begin
    if ($abstime>=JumpTime) M=N_max;
    else   M=N_min;
    count=count+1;
   if (count>=M) count=0;
   n = (2*count >= M);
   end
V(clock_out)<+ transition(n? v_high : v_low, td, tt);
……
```

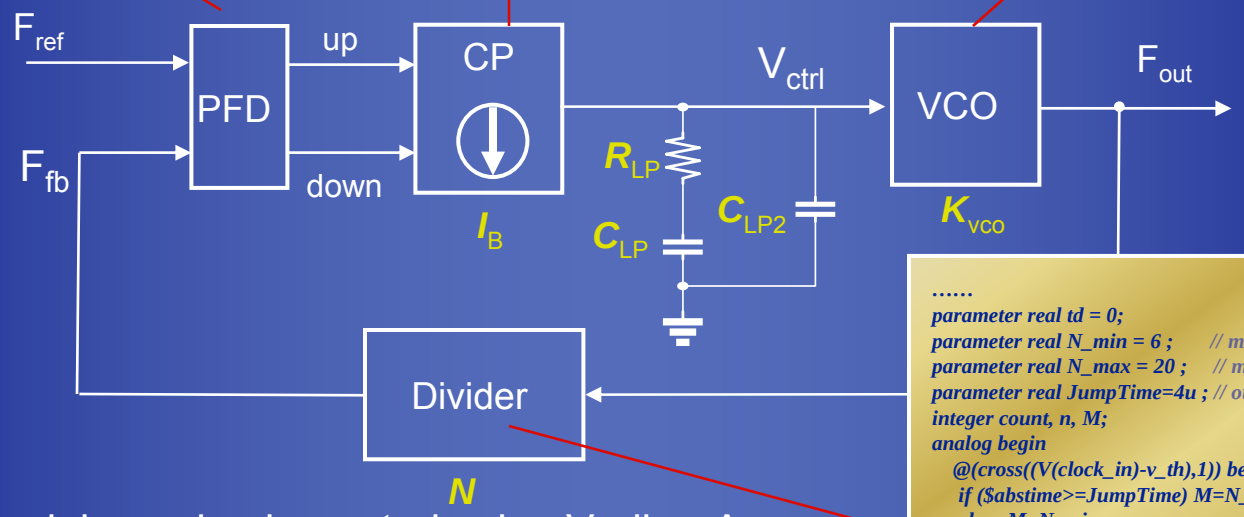
CP PLL Building Blocks on Behavioral Level

```
module pfd (ref, fb, u, d);
    output u, d; input ref, fb;
    electrical u, d, ref, fb;
    parameter real v_high=1.5;
    parameter real ttol=10p;
    parameter tt=120p;
    ….
    integer state;
analog begin
      @(cross(V(ref)-v_high/2), +1,ttol)
         if (state <1)  state = state +1;
      @(cross(V(fb)-v_high/2), +1,ttol)
         if (state >-1)  state = state - 1;

  V(u) <+ transition((state==+1) ? v_high : 0.0, td_u, tt);
  V(d) <+ transition((state==-1) ? 0.0 : v_high, td_d, tt);
end
endmodule
```

```
module ChargePump (Iout, Down, N_Down, N_Up, Up, V_bias_n, V_bias_p);
   output Iout; input Down, N_Down, N_Up, Up, V_bias_n, V_bias_p;
   electrical Iout, Down, N_Down, N_Up, Up, V_bias_n, V_bias_p;
   parameter real v_high = 1.5, v_low = 0;
   parameter real v_th =(v_high-v_low)/2;
   parameter real TransTime=10p from (0:1000000);
   parameter real Delay=1p from (0:1000000);
   parameter real Ip=25.0e-6;              // the value of output current
   parameter real v_max=1.3, v_min=0.2;    // limit the output voltage
   Parameter real Mis=0.05; // mismatch of charge and discharge current
  integer state;
analog begin
  @(cross(V(Up)-v_th, 1))       begin   state = -1;   end
  @(cross(V(Down)-v_th, 1))     begin   state = 1;    end
  @(cross(V(Up)-v_th, -1))      begin   state =0;     end
  @(cross(V(Down)-v_th, -1))    begin   state =0;     end
  @(cross(V(Iout)-v_max, 1))    begin   state =0;     end
  @(cross(V(Iout)-v_min, -1))   begin   state =0;     end
  I(Iout)<+transition(Ip*state*(1+state*Mis), Delay, TransTime);
 ……
```

```
 ……
     parameter real Vmin=0;
     parameter real Vmax=Vmin+1 from(Vmin:10e5);
     parameter real Fmin=1 from(0:10e9);
     parameter real Fmax=2*Fmin from(0:10e9);
     parameter real vlo=0, vhi=1.5;
     parameter real tt=0.01/Fmax from(0:10e3);
     parameter real ttol=1u/Fmax from(0:1/Fmax);
     real freq, phase;
     real kvco;
analog begin
     kvco= (Fmax-Fmin)/(Vmax-Vmin)
     freq=(V(V_tune)-Vmin)*kvco+Fmin;
     if(freq>Fmax) freq=Fmax;
     if(freq<Fmin) freq=Fmin;
     phase=idtmod(freq, 0.0, 1.0, -0.5);
     @(cross(phase-0.25,1,ttol))begin    Vout=vhi;    end
     @(cross(phase+0.25,1,ttol))begin    Vout=vlo;    end
     V(VCO_out)<+transition(Vout,0,tt);
 ……
```




```
……
parameter real td = 0;
parameter real N_min = 6 ;      // minimum divider value
parameter real N_max = 20 ;     // maximum divider value
parameter real JumpTime=4u ; // output frequency jump time
integer count, n, M;
analog begin
   @(cross((V(clock_in)-v_th),1)) begin
    if ($abstime>=JumpTime) M=N_max;
    else   M=N_min;
    count=count+1;
   if (count>=M) count=0;
   n = (2*count >= M);
   end
V(clock_out)<+ transition(n? v_high : v_low, td, tt);
……
```

All behavioral models are implemented using Verilog-A.

K. Kundert: *Predicting the phase noise and jitter of PLL-based frequency synthesizers*

PLL Specifications for Frequency Synthesis Application

- Nonlinear Acquisition Process

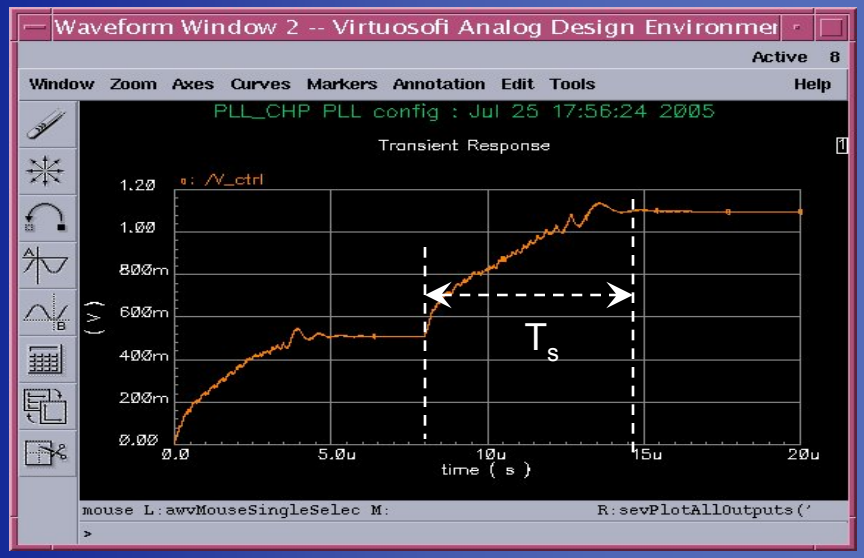
The locking time T_s is defined as the time taken by the PLL to synchronize with or to lock the new frequency.

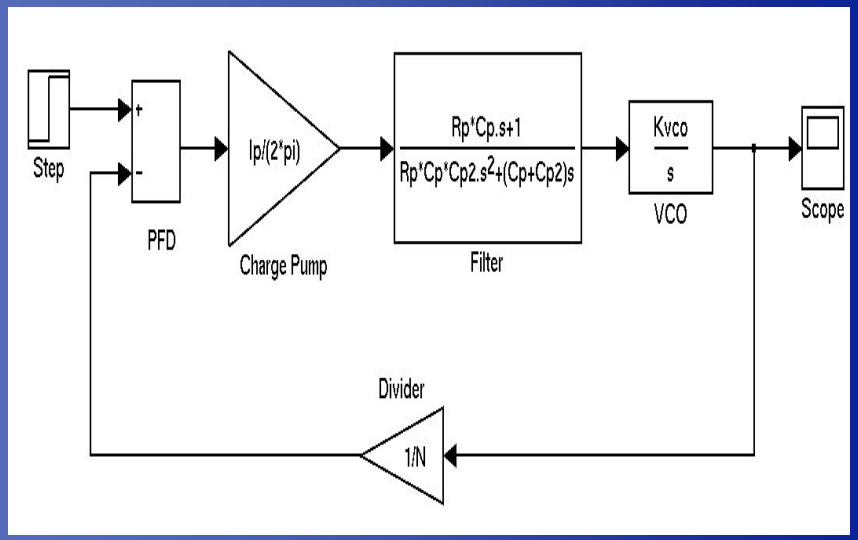
On circuit level,
simulation: time-consuming (hours/days)
optimization for T_s: impossible

- Linear Process in the Lock-In State

Damping factor, Nature Frequency, Phase Margin, Unity-Gain-Bandwidth, …

Matlab models,
can help analyze the PLL system
cannot automatically size the parameters





Our goal: to automatically size the PLL circuit, considering the performances in nonlinear and linear processes at the same time !

Outline

- Analog Hierarchical Design Methodology
- Phase-Locked Loops Introduction
  - Architecture Selection
  - PLL Building Blocks
  - PLL Specifications
- Automatic Hierarchical Sizing of a Charge Pump PLL
  - Definition & Propagation of Specifications
  - Automatic Behavioral Level Sizing
  - Automatic Circuit Level Sizing
  - Design Verification
- Conclusion & Outlook

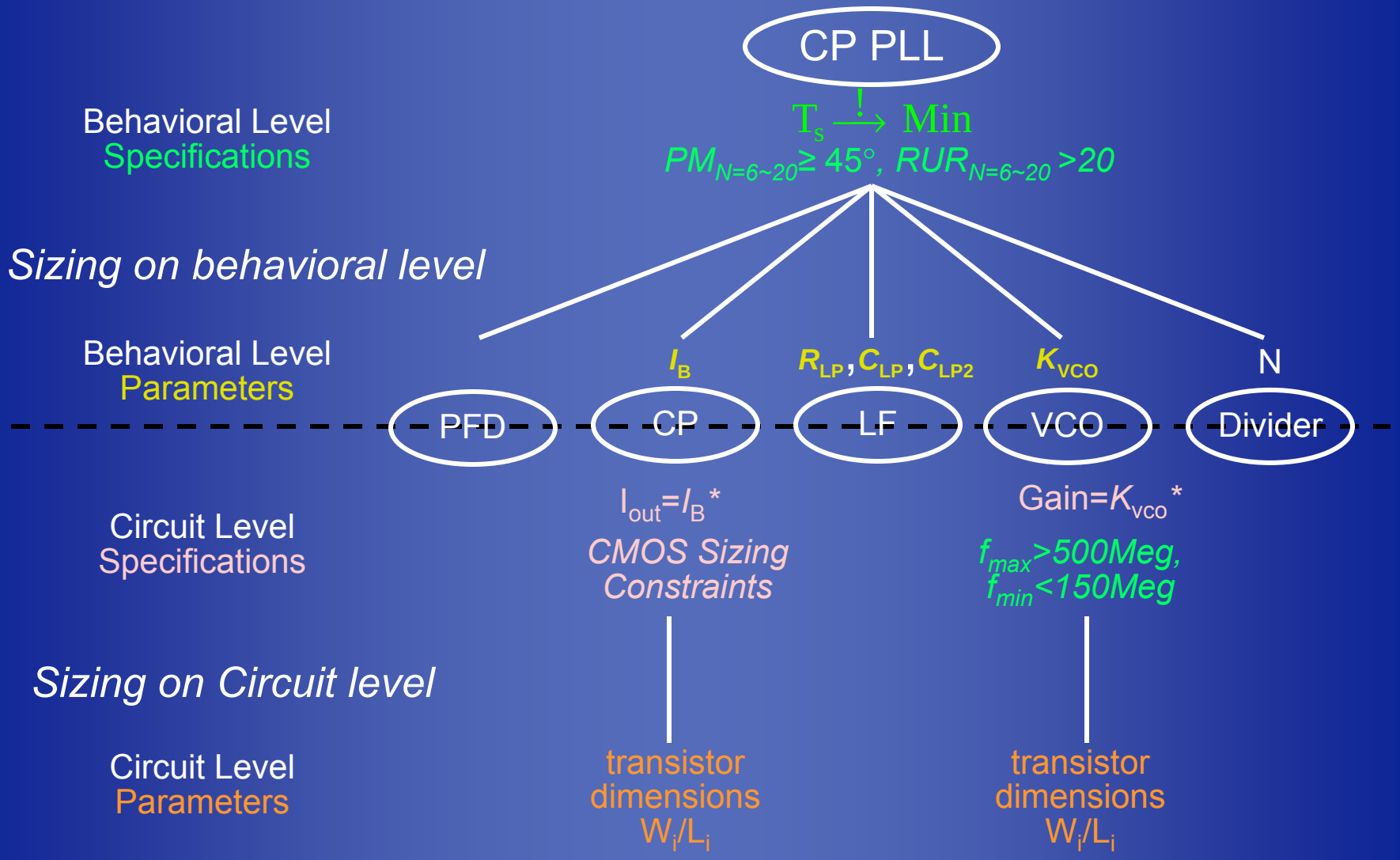
Definition of PLL Specifications

Type	*Specifications*	*Values*
Performances	Output frequency range	150MHz to 500 MHz
	Locking time T_s ✦	minimal
	Phase margin for all *N*	$\geqslant$ 45°
	RUR* for all *N*	$\geqslant$ 20
Operation	Reference frequency	25MHz
	Supply voltage	1.5V
Technology	Infineon	130nm

✦ *We define the locking time in the worst case, i.e. output frequency directly jumps from 150MHz to 500MHz. This determine the divider value N jumping from 6 to 20.*

** RUR = the reference input frequency / the unity-gain-bandwidth of PLL*

Performance Propagation on CP PLL



Behavioral Level Parameters Variation Range Definition

Parameters set $\underline{\mathbf{p}} = [\, I_B,\ C_{LP},\ C_{LP2},\ R_{LP}, K_{VCO}]$

$5 < I_B$ in µA < 100

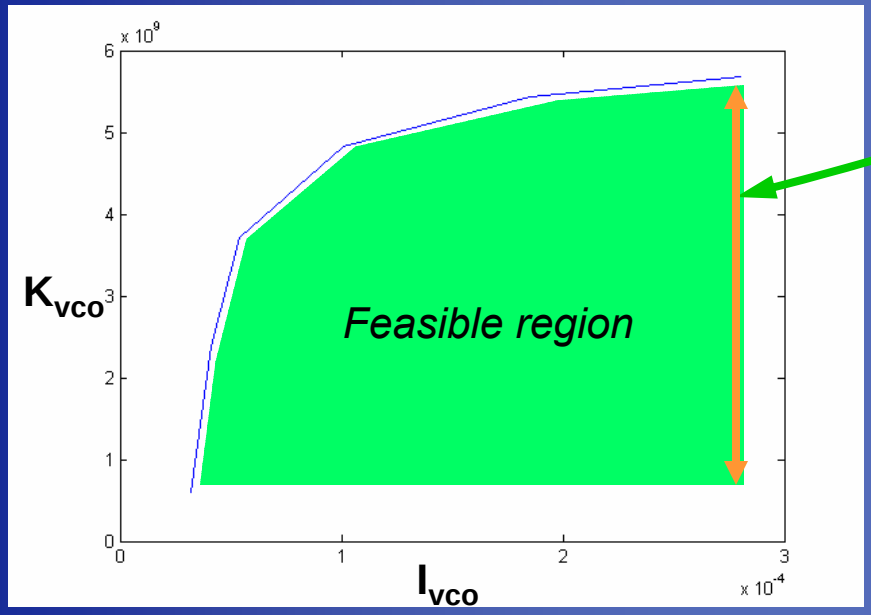
$20 < C_{LP}$ in pF < 200

$1 < C_{LP2}$ in pF < 15

$1 < R_{LP}$ in kΩ < 100

$0.5 < K_{vco}$ in GHz/V < 5.5

Design Space Exploration on VCO Circuit



Feasible region of K_{vco}

s.t. $F_{max} > 500$MHz & $F_{min} < 150$MHz

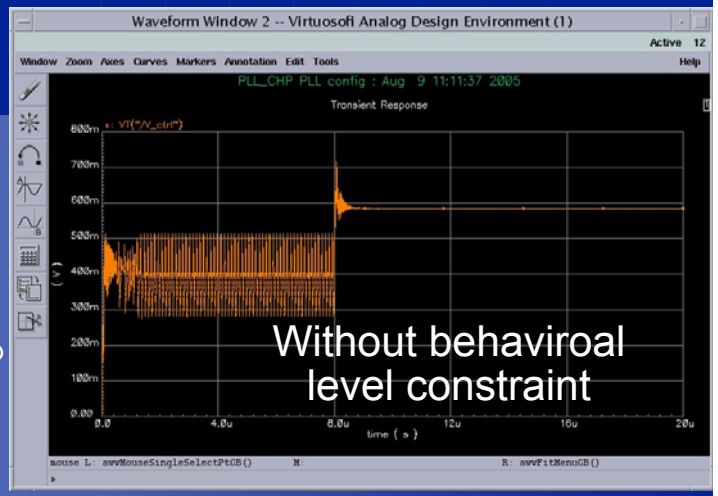
D. Müller, etc.: *Deterministic Approaches to Analog Performance Space Exploration*

Automatic Behavioral Level Sizing

- Optimization process

$$\min_{\mathbf{p}}(T_s)$$

$$\text{s.t.}\ RUR \geq 20;\ PM_{N_{\max}} \geq 45°;\ PM_{N_{\min}} \geq 45°$$



- Optimization results

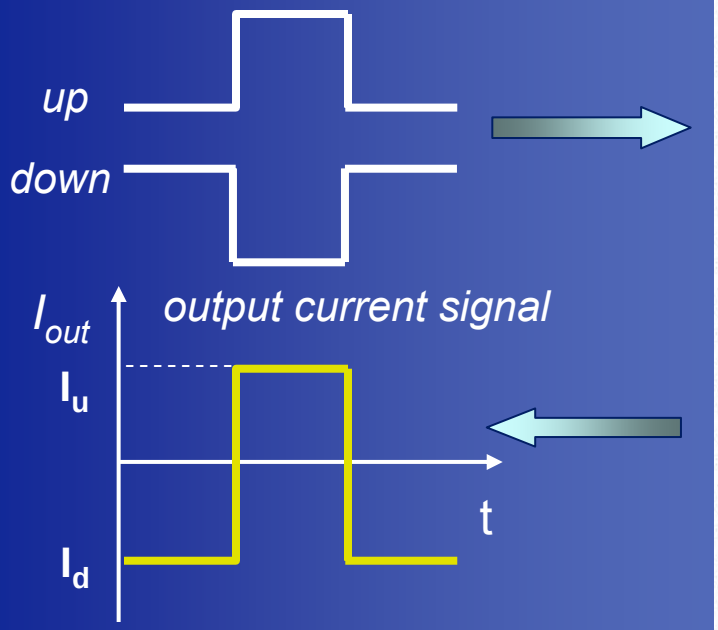
	Parameters	Performances
Initial	C_p=50pF; C_{p2}=10pF; I_p=9μA; R_p=30kΩ; K_{vco}=550MHz/V	PM_{Nmax}=43.5°; PM_{Nmin}=41.7°; RUR= 56; T_s=6.4μs
Optimized	C_p=54.2pF; C_{p2}=8.185pF; I_p=38μA; R_p=9.05kΩ; K_{vco}=1.07GHz/V	PM_{Nmax}=45.01°; PM_{Nmin}=49.6°; RUR=20; T_s=1.3μs

- Optimization time — About 250 simulations about 12min
- Optimization tool — WiCkeD of MunEDA

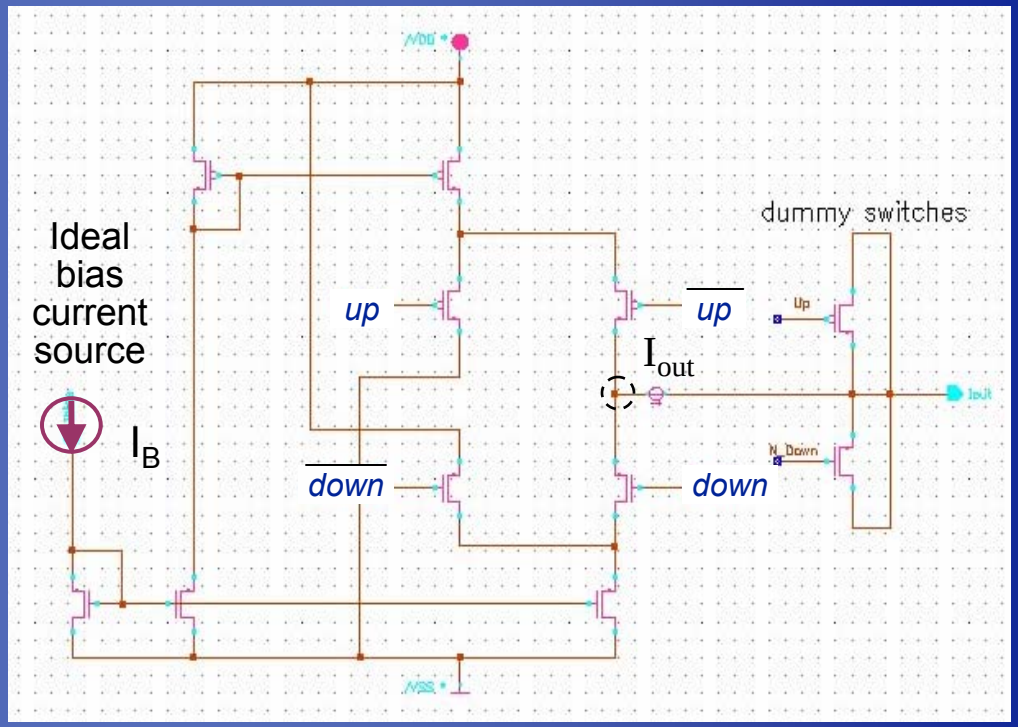
Automatic Sizing Process for CP

- Test bench

Two input voltage-pulse signals



External-biased charge-pump schematic



- Optimization process

$$\min_{\mathbf{d}_{CP}} \left(I_u - I_B\right)^2 + \left(I_d + I_B\right)^2$$

s.t. CMOS sizing rules

- Optimization time

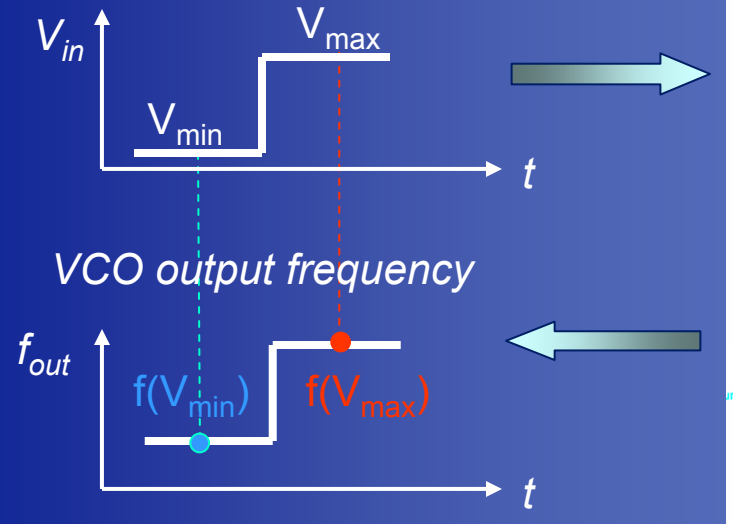
about 2min

Parameters set in CP: $\underline{\mathbf{d}}_{cp}$ = [*w*1,*l*1,*w*2,*l*2,…,*wn*,*ln*]

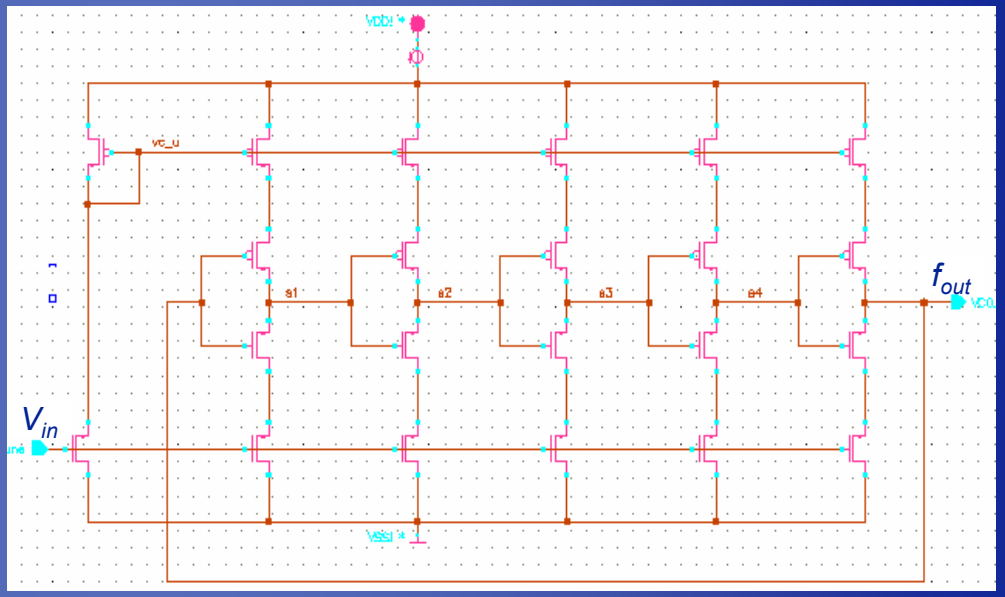
Automatic Sizing Process for VCO

- Test bench

Two-stage Input voltage signal



Five-stage single-ended VCO schematic



- Optimization process

$$\min_{\mathbf{d}_{VCO}} \left(gain - K_{vco}\right)^2$$

s.t. $f(V_{max}) > 500MHz$; $f(V_{min}) < 150MHz$

where $gain = \left(f(V_{max}) - f(V_{min})\right) / (V_{max} - V_{min})$

Parameters set in VCO: $\underline{\mathbf{d}}_{vco}$ = [*w*1,*l*1,*w*2,*l*2,…,*w*n,*l*n]

- Optimization time

about 8min

Simulation Time on Both Levels and the whole Sizing Time

- Comparison of the time for single transient simulation on both levels

Level	**simulation time** for 20 μ s transient simulation
Behavioral	38.5sec
Circuit (except divider)	1h 32min 11sec ≈ 92min

- Time for the whole Hierarchical Design Process

$t_{hierarchical\text{-}sum} = t_{behavioral\text{-}level} + t_{circuit\text{-}level} + t_{design\text{-}verification}$

about 12min

CP: about 2min
VCO: about 8min

about 92min

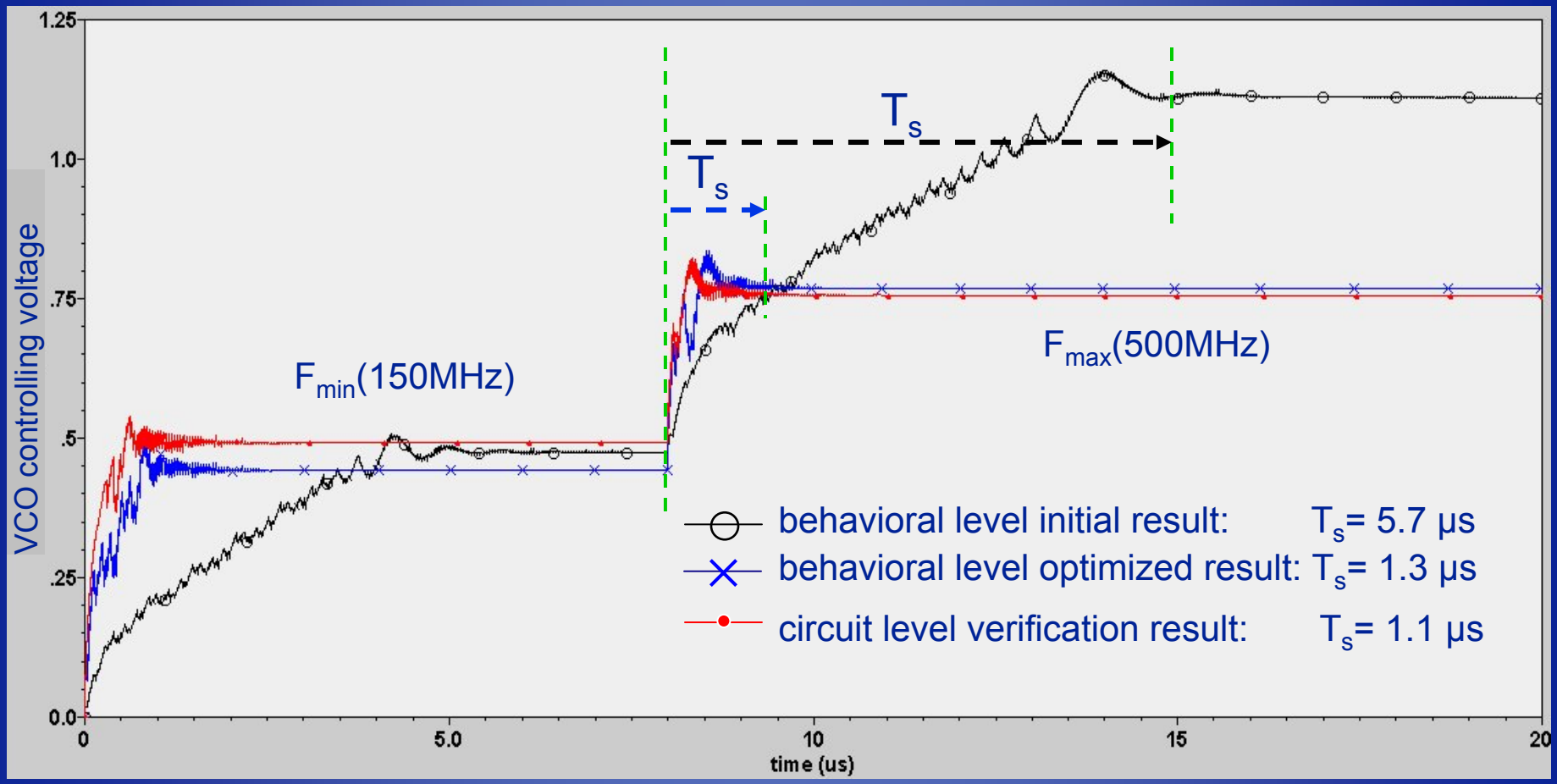
verification on circuit level

$t_{hierarchical\text{-}sizing} \approx 20min$

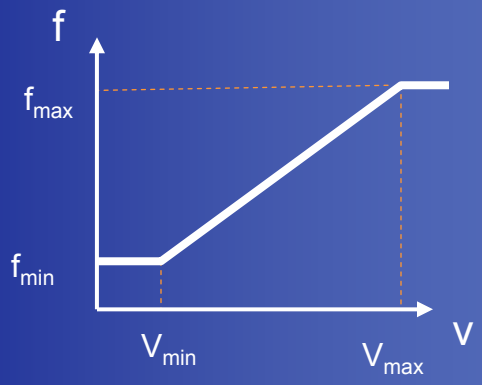
$t_{hierarchical\text{-}sum} \approx 112min$

Design Verification based on Circuit Level Simulation

- Simulation results before & after optimization

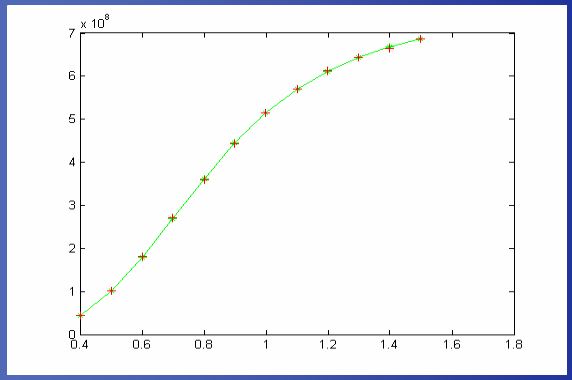


Design Verfication based on VCO Nonlinear Model



Linear and limited VCO model

```
module vco(V_tune, VCO_out);
    ……
    real freq, phase, kvco;
analog begin
    kvco=(Fmax-Fmin)/(Vmax-Vmin);
    freq=(V(V_tune)-Vmin)*kvco+Fmin;
   if(freq>Fmax) freq=Fmax;
   if(freq<Fmin) freq=Fmin;
    …..
end
endmodule
```



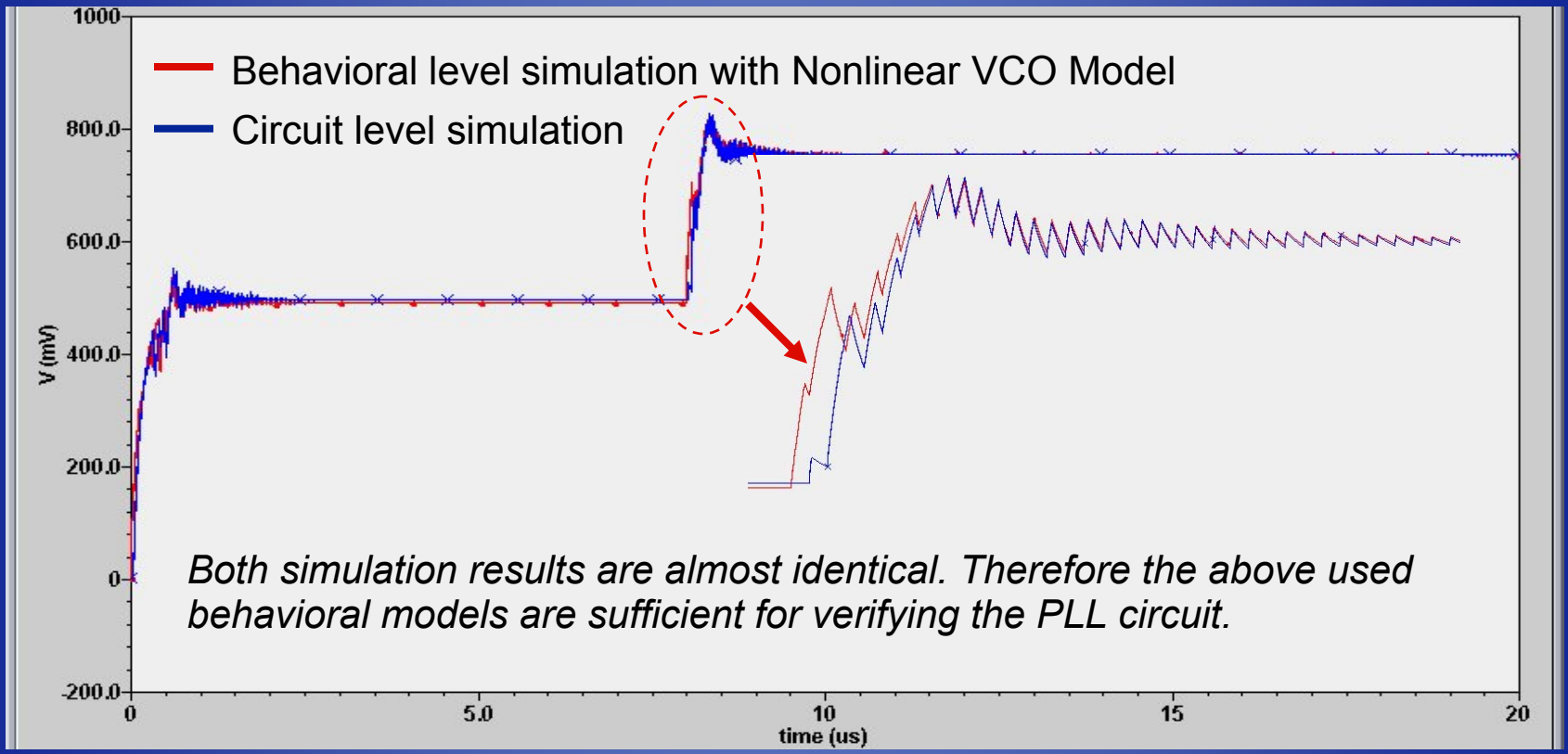
Nonlinear VCO model
polynomial model (4th-order)

```
. . . . .
freq =
k4*pow(V(V_tune),4)+k3*pow(V(V_tune),3) +
k2*pow(V(V_tune),2) +k1*V(V_tune) + k0;
// fitting curve's coefficients
. . . . . .
```

Top-Down Phase

Bottom-Up Phase

Simulation Results of the Nonlinear Model and Circuit Level



$t_{hierarchical\text{-}sum} = t_{hierarchical\text{-}sizing} + t_{Verification\text{-}with\text{-}Nonlinear\text{-}Model}$

about 20min

Calibration of VCO Nonlinear Model about 10min
PLL Verification about 40sec

$t_{hierarchical\text{-}sum} \approx 30min$

Conclusion & Outlook

Conclusion

- A CP PLL hierarchical sizing process has been presented.
- Automatic sizing process could be achieved in 20 minutes.
- Sizing constraints on behavioral level are indispensable .
- Design space exploration on circuit level confirms feasibility.

Outlook

- Extension to other specifications, e.g. noise
- Extension to other PLL architectures.