Problem 19.1

The XOR gate, seen in Figures 1 and 3, exhibits input-dependent skew, meaning that that delay from the input changing to the output changing is different depending on which input is changed. This can be observed in Figure 4.

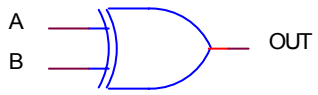


Figure 1: XOR Gate

Truth Table		
A	**B**	**A ⊕ B**
0	0	0
0	1	1
1	0	1
1	1	0

Figure 2: XOR Gate Truth Table

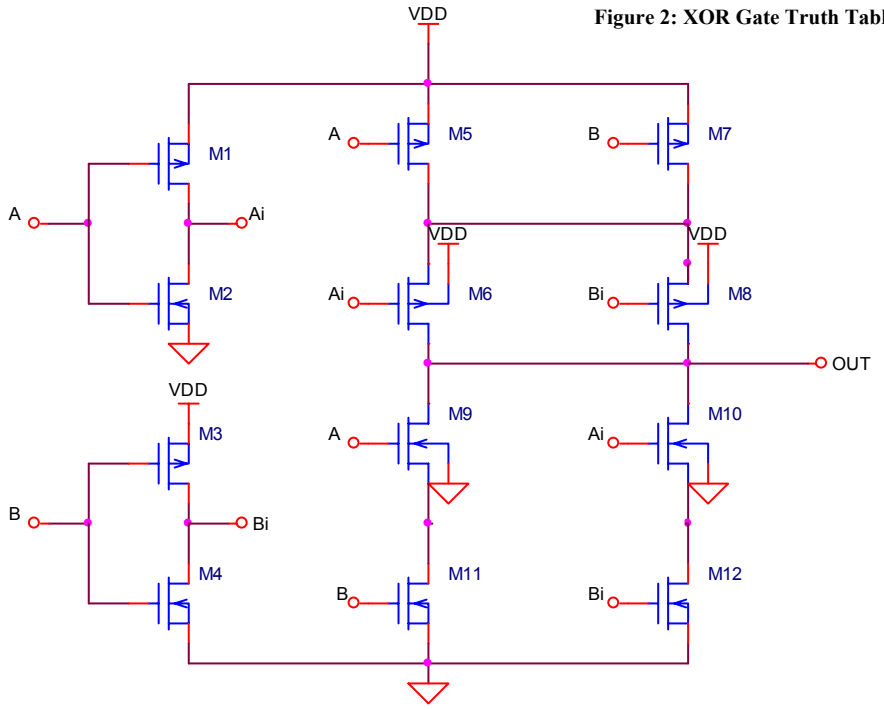


Figure 3: CMOS Implementation of an XOR Gate

```
*** XOR gate ***

.control
destroy all
run
plot out1 out2
.endc

.option scale=50n
.tran .01n 2n UIC


VDD     VDD     0       DC      1
VA      A       0       DC      0 PULSE 0 1 0 1n
VB      B       0       DC      1

X1      VDD     A       B       out1    XOR
X2      VDD     B       A       out2    XOR
```

```
.subckt XOR        VDD    A      B      out
M1      Ai         A      VDD    VDD    PMOS L=1 W=20
M2      Ai         A      0      0      NMOS L=1 W=10
M3      Bi         B      VDD    VDD    PMOS L=1 W=20
M4      Bi         B      0      0      NMOS L=1 W=10

M5      n1         A      VDD    VDD    PMOS L=1 W=20
M6      out        Ai     n1     VDD    PMOS L=1 W=20
M7      n1         B      VDD    VDD    PMOS L=1 W=20
M8      out        Bi     n1     VDD    PMOS L=1 W=20

M9      out        A      n2     0      NMOS L=1 W=10
M10     n2         B      0      0      NMOS L=1 W=10
M11     out        Ai     n3     0      NMOS L=1 W=10
M12     n3         Bi     0      0      NMOS L=1 W=10
.ends
```

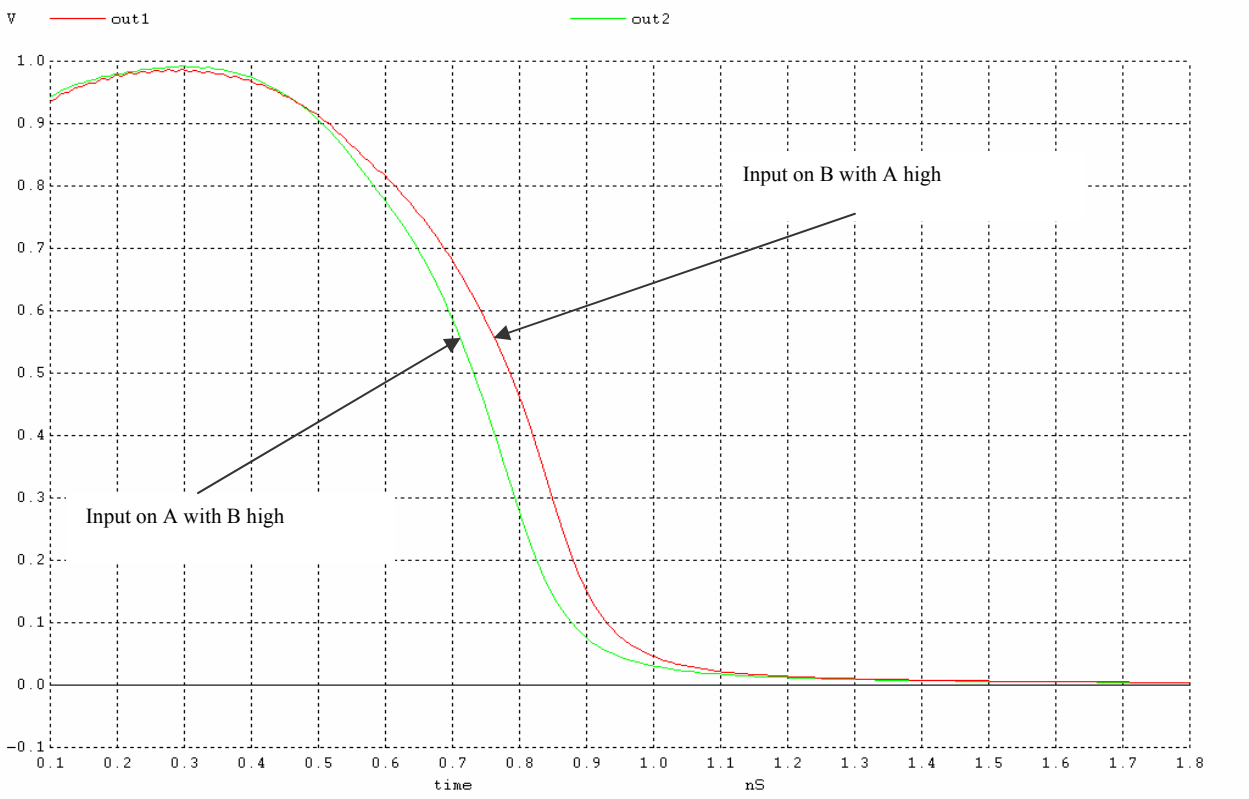


Figure 4: Showing Input-Dependent Skew

For a transition of out from high to low, the A input, with the B input held high, propagates to the output slightly faster than the B input, with the A input held high. This is because the signals see slightly different paths through the circuit from the input to the output.

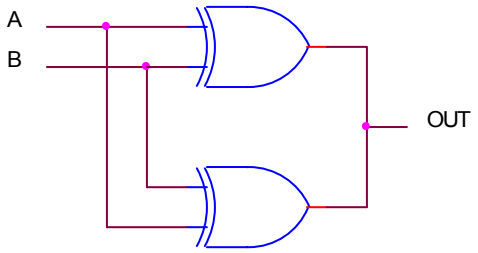


Figure 6: XOR Gates Connected in Parallel

We can put two XOR gates in parallel, as seen in Figure 6, to reduce this input-dependent skew. Then the A and the B inputs, each of which are connected to both an A and a B input on an identical XOR gate, will see the same path to OUT, and get there at the same time. This is the design we will use for the XOR phase detector.

```
*** XOR PD ***

.control
destroy all
run
plot A B+1
plot out1 out2+1
.endc

.option scale=50n
.tran .001n 100n UIC


VDD     VDD     0       DC      1
VA      A       0       DC      0 PWL 0 1 50n 1 50.01n 0 75n 0 75.01n 1 100n 1
VB      B       0       DC      0 PWL 0 1 25n 1 25.01n 0 50n 0 50.01n 1 100n 1

X1      VDD     A       B       out1    out2    XORPD

.subckt XORPD   VDD     A       B       out1    out2
M1      Ai      A       VDD     VDD     PMOS L=1 W=20
M2      Ai      A       0       0       NMOS L=1 W=10
M3      Bi      B       VDD     VDD     PMOS L=1 W=20
M4      Bi      B       0       0       NMOS L=1 W=10

M5      n1      A       VDD     VDD     PMOS L=1 W=20
M6      out1    Ai      n1      VDD     PMOS L=1 W=20
M7      n1      B       VDD     VDD     PMOS L=1 W=20
M8      out1    Bi      n1      VDD     PMOS L=1 W=20

M9      out1    A       n2      0       NMOS L=1 W=10
M10     n2      B       0       0       NMOS L=1 W=10
M11     out1    Ai      n3      0       NMOS L=1 W=10
M12     n3      Bi      0       0       NMOS L=1 W=10

M13     n1      B       VDD     VDD     PMOS L=1 W=20
M14     out2    Bi      n1      VDD     PMOS L=1 W=20
M15     n1      A       VDD     VDD     PMOS L=1 W=20
M16     out2    Ai      n1      VDD     PMOS L=1 W=20

M17     out2    B       n2      0       NMOS L=1 W=10
M18     n2      A       0       0       NMOS L=1 W=10
M19     out2    Bi      n3      0       NMOS L=1 W=10
M20     n3      Ai      0       0       NMOS L=1 W=10
.ends
```

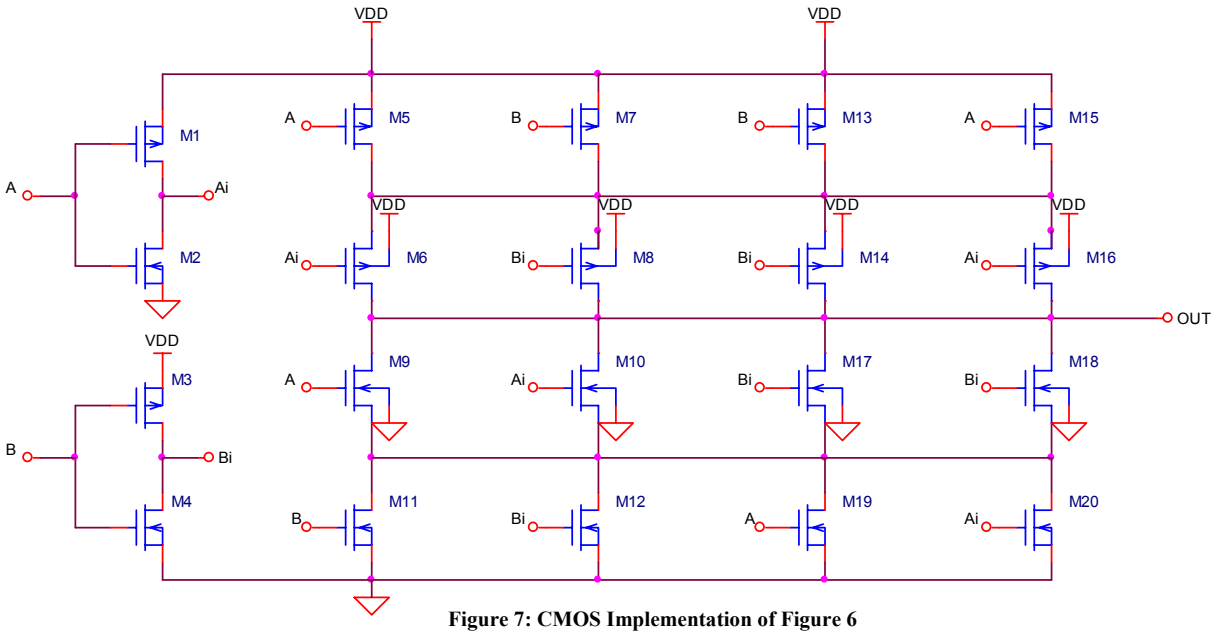


Figure 7: CMOS Implementation of Figure 6

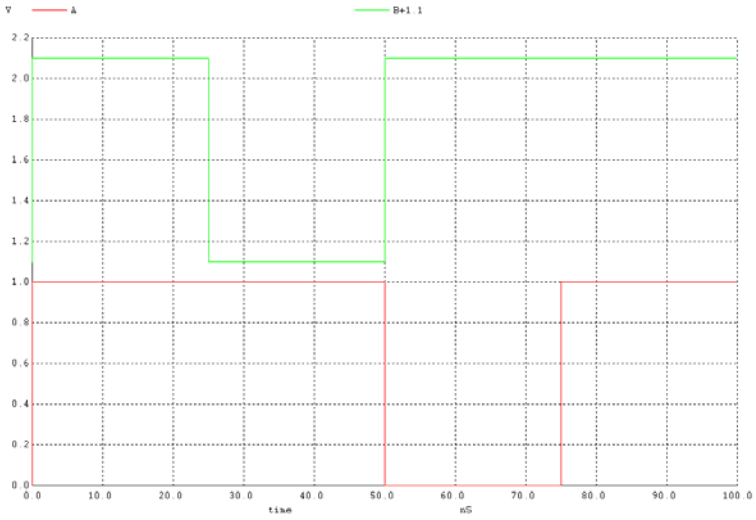


Figure 8: Inputs to Circuit in Figure 6

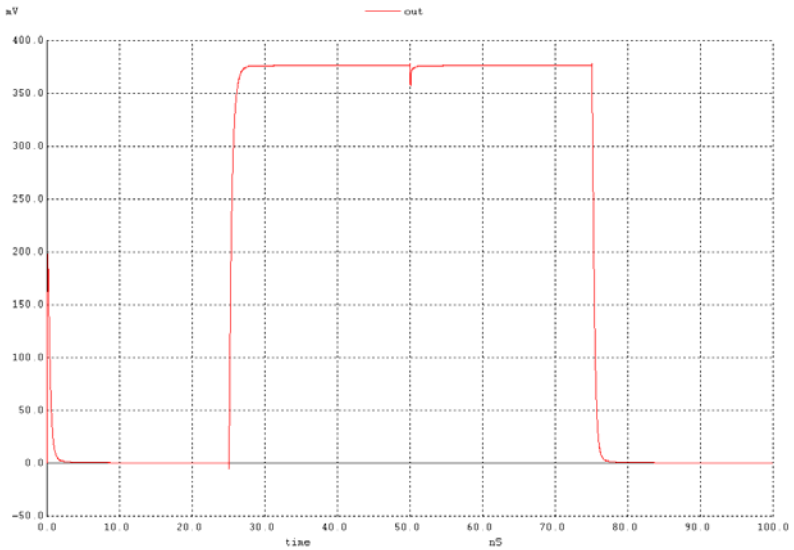


Figure 9: Output of Circuit in Figure 6

Figure 8 shows the A and B inputs to the circuit in Figure 6. First A is held high while B is changed, then B is held high while A is changed. Figure 9 shows the output of the circuit. From these two figures we can see that the circuit follows the truth table for the XOR gate, shown in Figure 2. Now we will look at the outputs of each XOR gate in the XOR PD to see if the skew has been reduced. In Figure 10 we can see that the outputs look nearly identical, and Figure 11 shows that when the output of the XOR PD goes high, the outputs of the XOR gates are nearly indistinguishable, so we have indeed reduced the skew by giving the signals the same paths to the output.

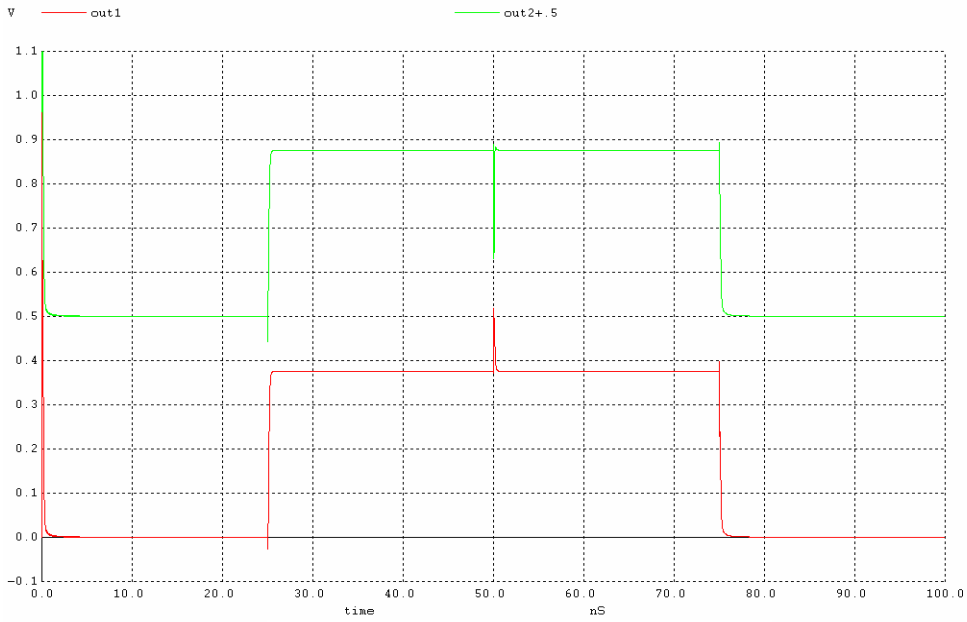


Figure 10: Outputs of each XOR gate in Figure 6

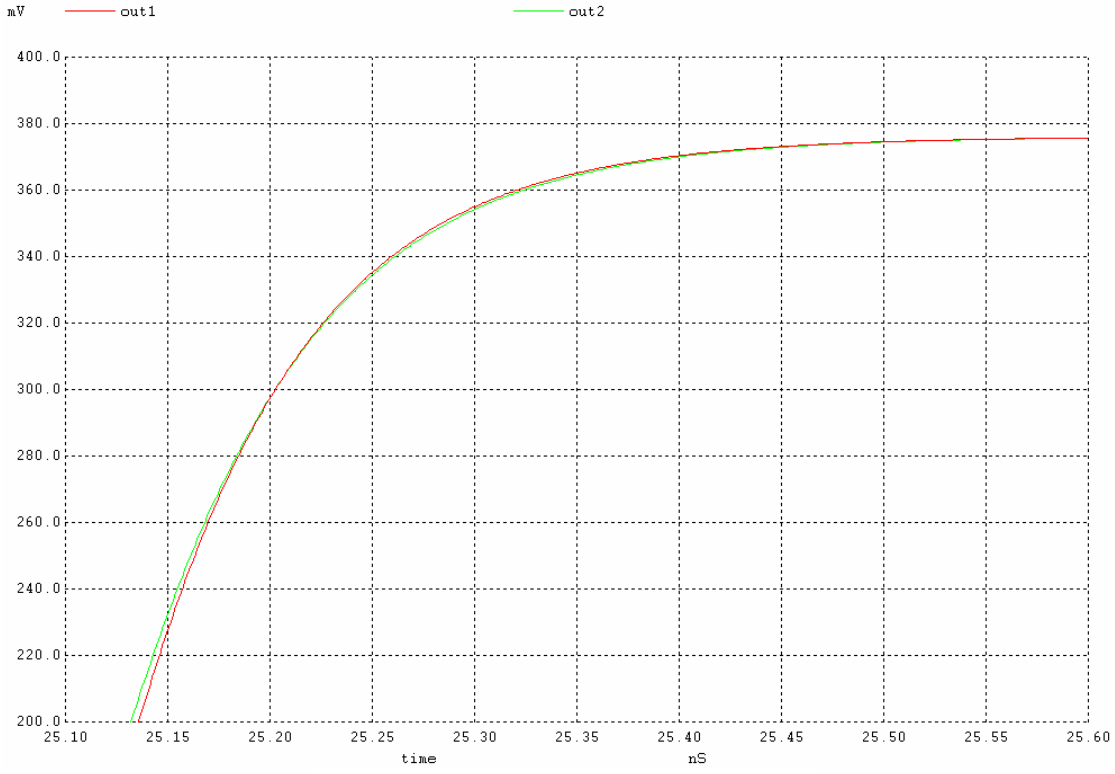


Figure 11: Outputs of each XOR gate in XOR PD

Problem 19.2 Miles Wiscombe

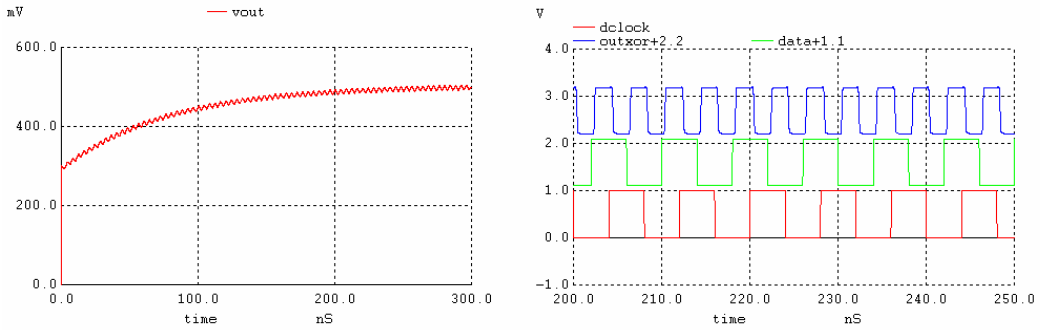
Question:

Verify, using simulations, that a locked PLL using an XOR PD will exhibit, after RC filtering, an average value of VDD/2. Show, using simulations and hand calculations, the filter's average output if the XOR PD sees a phase difference in its inputs of $-\pi/4$.

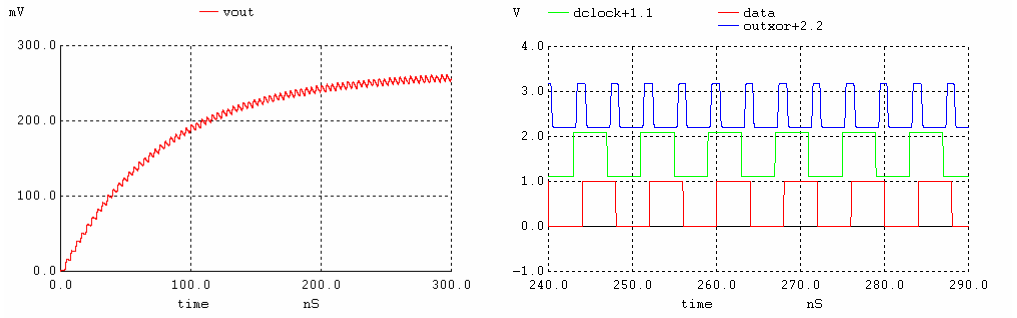
Answer:

An XOR PD is simply a XOR gate. When the XOR PD is in lock, the output of the XOR will be a pulse train with a 50 percent duty cycle. This is due to the fact that this type of phase detector locks on the center of the data. Due to this, when in lock the output will be high for half the time and low for half the time. The RC low-pass filter (**See figure 19.5**) then integrates (averages) the output of the XOR PD so that this signal can be used to control the VCO. The VCO, through feedback to the PD is used to generate the dclock signal that is being aligned with the data. (**See figure 19.3**)

Since the filter is simply averaging the output of the XOR, the result can be calculated by VDD*($\Delta\Phi/\pi$) (**Equation 19.5**). In this equation $\Delta\Phi$ is the phase difference in radians of data and dclock. Therefore, when dclock is aligned with the middle of the data the output of the filter is VDD/2 because $\Delta\Phi = \pi/2$. The following simulation results illustrate this situation. Vout is converging to VDD/2 or 0.5 V. An initial condition of .3V was used to shorten simulation time.



If the XOR PD sees a phase difference in its inputs of $-\pi/4$ the output would be VDD/4 (using equation 19.5). This circumstance is illustrated in the simulation results below. It can be seen that Vout is converging to VDD/4 or 0.25 V.



Since the output of the filter is simply an average of the XOR output, the XOR PD can not discriminate between harmonics. For example, if the dclock was twice the frequency

of the data and the rising edge was aligned in the middle of the bit, the output would still be VDD/2. Because of this limitation of the XOR PD the VCO should be designed with an operating frequency range much less than 2fclock and much greater than 0.5fclock where fclock is the nominal clock frequency for the proper lock with a XOR PD.

Netlist:

```
*** problem 9.2 ***
.control
destroy all
run
plot outxor vout data dclock
.endc

.option scale=50n
.tran 5p 300n 0n 5p UIC

VDD       VDD     0       DC      1
Vdclock   dclock  0       DC      0       pulse 0 1 2.95n 50p 50p 3.95n 8n
Vdata     data    0       DC      0       pulse 1 0 0n 50p 50p 3.95n 8n

Xxor      VDD     data    dclock  outxor  exclusiveor

R1        outxor  vout    2.5k
C1        vout    0       30p

.subckt inverter    VDD     in      out
M1        out     in      0       0       NMOS    L=1     W=10
M2        out     in      VDD     VDD     PMOS    L=1     W=20
.ends

.subckt inverter2   VDD     in      out
M1        out     in      0       0       NMOS    L=1     W=200
M2        out     in      VDD     VDD     PMOS    L=1     W=400
.ends

.subckt exclusiveor VDD     data    dclock  outxor

M1        vs3     data    VDD     VDD     PMOS    L=1     W=20
M2        vs3     dclock  VDD     VDD     PMOS    L=1     W=20
M3        out     dataf   vs3     VDD     PMOS    L=1     W=20
M4        out     dclockf vs3     VDD     PMOS    L=1     W=20
M5        out     data    vd7     0       NMOS    L=1     W=10
M6        out     dataf   vd8     0       NMOS    L=1     W=10
M7        vd7     dclock  0       0       NMOS    L=1     W=10
M8        vd8     dclockf 0       0       NMOS    L=1     W=10
Xin       VDD     data    dataf   inverter
Xin2      VDD     dclock  dclockf inverter
Xin3      VDD     out     outf    inverter
Xin4      VDD     outf    outxor  inverter2

.ends
```

Jared Fife

Problem 19.3

This problem discusses the importance for the VCO's center frequency to match the input NRZ data rate when a passive loop filter is used, and why using the active loop filter eliminates this requirement.

Since a passive loop filter does not have memory, when the data is not changing, the output of the filter (which is V_{inVCO}) will wander back to VDD/2. This causes a static phase error. The biggest problem with this static phase error is that a small amount of jitter on of the input frequency will now cause the clock to not lock up on the data.

The integral part of the active loop filter (proportional + integral) gives the loop memory. This enables V_{inVCO} to move away from VDD/2 which gets rid of the static phase error.

Fig. 1 below is the DPLL in Ex. 2 from the text with a passive loop filter. Here we have data followed by five zeros, and a small amount of noise on VDD (see the netlist details below). Compare this plot to Fig. 2 below which is similar except an active loop is used. With the same data followed by five zeros and noise on VDD this loop locks up with static phase error, whereas Fig. 1 with a passive loop can no longer lock on the data with the same amount of noise on VDD.

Figure 1 DPLL with passive loop filter, noise on VDD causes loop not to lock.

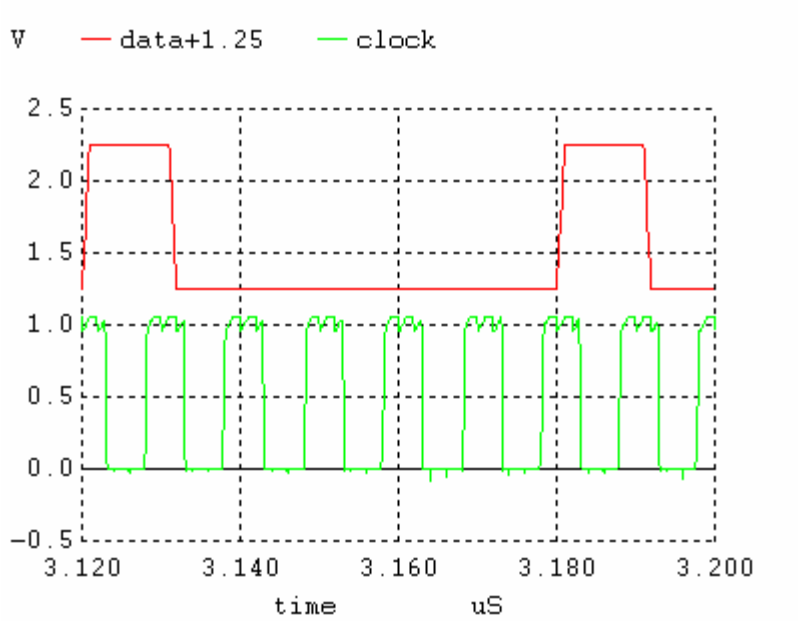


Figure 2 DPLL with active loop filter, loop can still lock with noise on VDD.

Netlist

Fig 1:
To create this plot, use the netlist for Fig 19.28 in the text except replace the following two lines:

```
VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 80n
```

With the following:

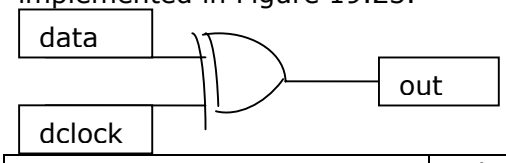
```
VDD     VDD     0       DC      1       pulse .95 1.05 0 0 0 1n 2n
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 60n
```

Fig. 2:
Repeat the replacement above in the netlist for Fig. 19.31 in the text.

19.4 Suppose, for a robust high-speed PLL using an XOR PD, that it is desirable to adjust the PD's gain. Show how a charge pump can be used towards this goal. Should the loop filter have two outputs? Discuss the PD's gain and how it would be adjusted. What topology would be used for a passive loop filter? for an active filter?

Solution:

The XOR PD is simply an exclusive OR gate phase detector shown in Figure 19.4 and implemented in Figure 19.23.



Data	Dclock	Out
0	0	0
0	1	1
1	0	1
1	1	0

Figure 19.4 XOR gate with truth table.

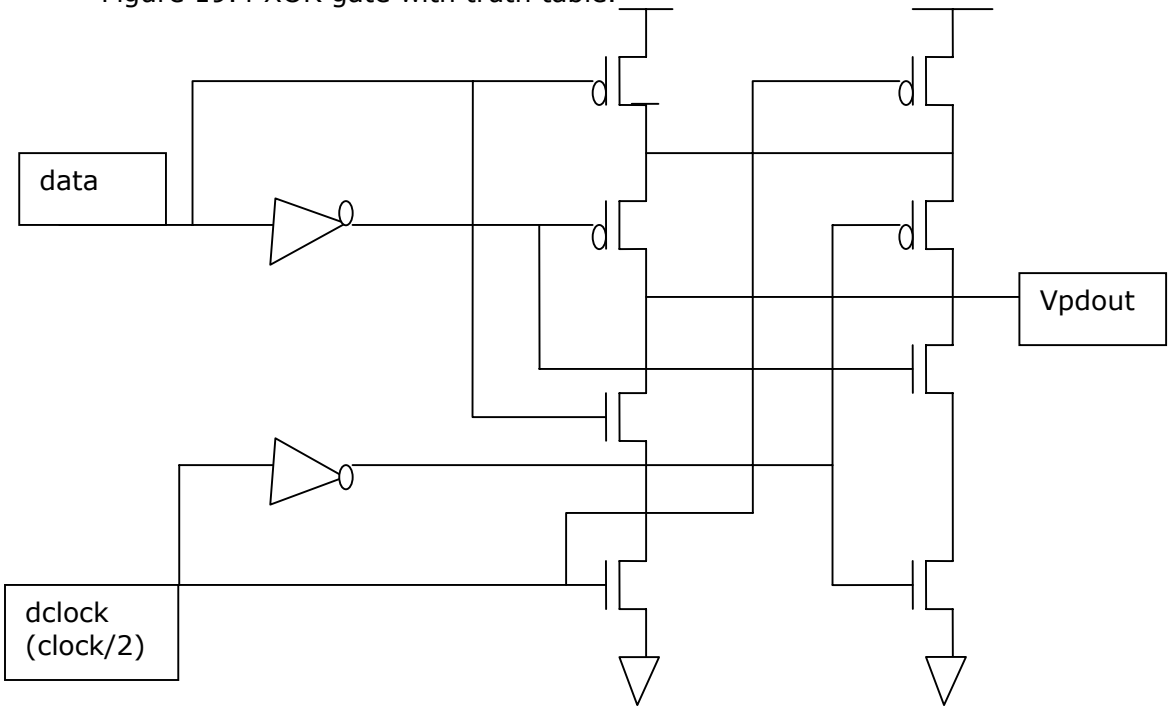


Figure 19.23 XOR phase detector (PD).

When the loop of XOR DPLL is locked, the clock rising edge is centered on the data and the time difference between the dclock rising edge and the beginning of the data is Tclock/2 or Tdclock/4. Thus the phase difference between clock and data when in lock is ∏.

The average voltage out of the XOR phase detector is

$$V_{pdout} = VDD*\Delta\Phi/\Pi \qquad (19.5)$$

where $\Delta\Phi = \Phi data - \Phi clock$ is the phase difference between the dclock and data and

$K_{pd} = VDD/\Pi$ is the gain of the XOR PD in unit of volts/radian.

The gain of XOR PD is fixed with fixed VDD. In order to adjust the gain we need to add charge pump circuit with adjustable current source as shown in Figure 3.

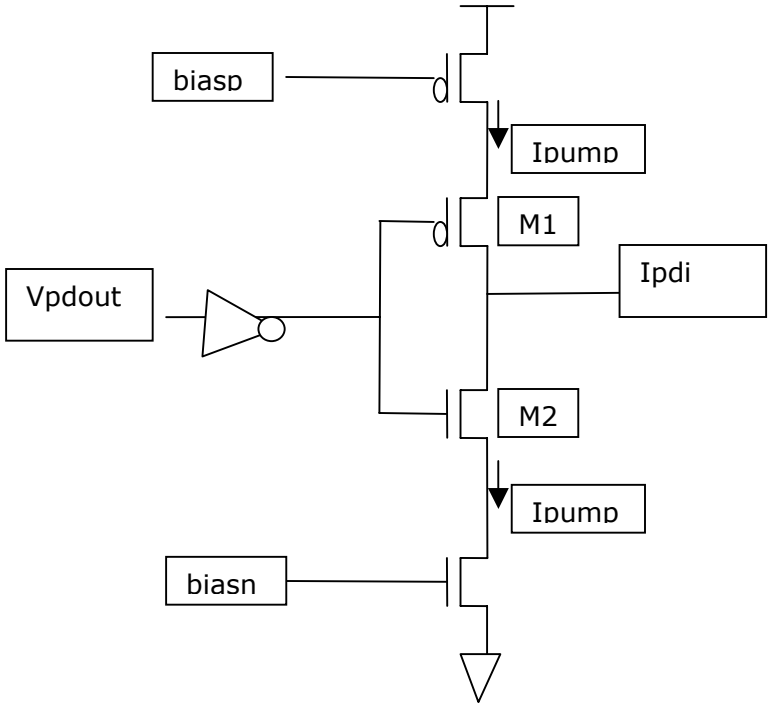


Figure 3 Charge pump to convert Vpdout to Ipdout.

Vpdout changes from 0 (Ipump flows out of Ipdi through M1) to VDD (Ipump flows to Ipdi through M2) and the output current Ipdi is

Idpi = (-Ipump-Ipump)*ΔΦ/(2∏) (2)

where Kpdi = -Ipump/∏ is the gain in units of amps/radian adjustable by Ipump and Ipump is adjustable through the bias voltages (vbiasp ans vbiasn) from diode connected MOSFETs to form current mirrors.

The basic current mirror is shown in Figure 20.1.

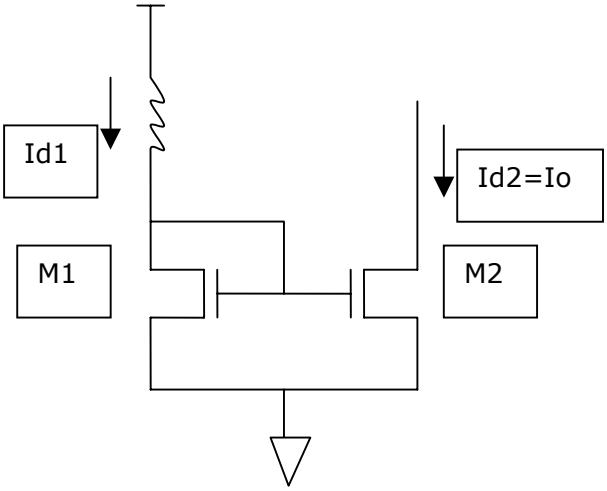


Figure 20.1 Basic current mirror schematic.

The output current Io=Id2 can be adjusted through the W/L ratio of the devices M1 and M2:

Io = Id2 = Id1((W2/L2)/(W1/L1)) (20.3)

The loop filter still has one output Vinvco to the input of voltage-controlled oscillator (VCO). However, since the input of loop filter will be current instead of voltage, the topology of the passive loop filter will be different from those shown in Figure 19.29, shown in Figure 19.13.

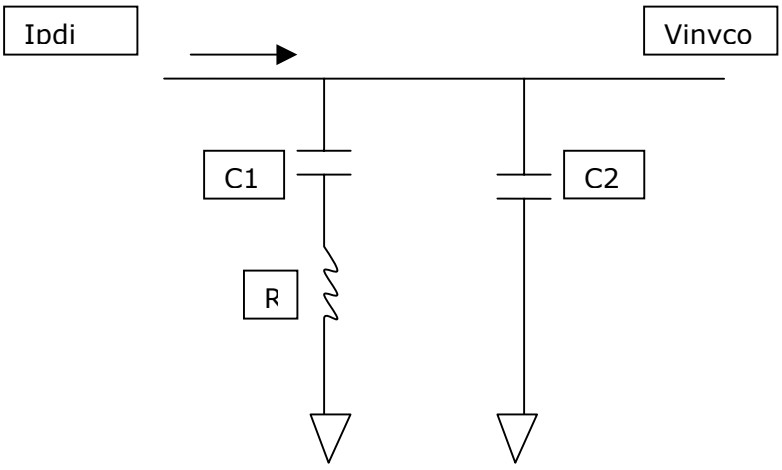


Figure 19.13 Passive loop filter topology.

The impedance of the passive filter is:

$Z = 1/(1/(1/j\omega C1+R)+j\omega C2) = (1+j\omega C1R)/(j\omega(C1+C2)(1+j\omega RC1C2/(C1+C2)))$

$Vinvco = Ipdi*Z = Kf*Ipdi$ (19.13)

where Kf = Z is the gain of passive loop filter.

For slow variations in the phase ($\omega << 1/(RC1)$), the current, Ipump, linearly charges C1 and C2, $Kf = 1/j\omega(C1+C2)$, thus giving an averaging effect. For fast variations ($\omega >> /(RC1)$) and assume C2 is small ($C2 << 1/(\omega R)$), the charge pump simply drives the resistor R, Kf = R, thus eliminates the averaging and allows the VCO to track quickly the moving variations in the input data.

The active proportional-integral (PI) loop filter will not work with charge pump.

Simulation:

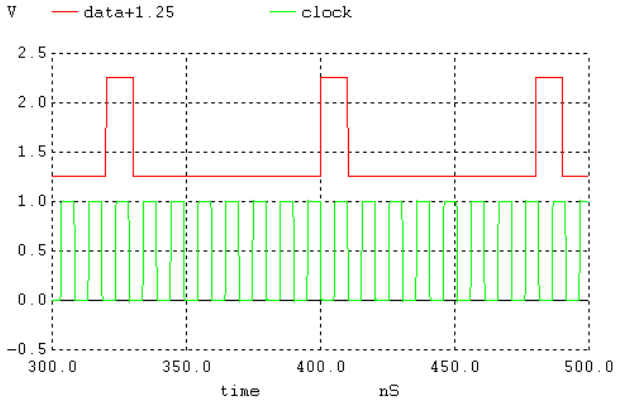


Figure 6 DPLL locks up to the center of data.

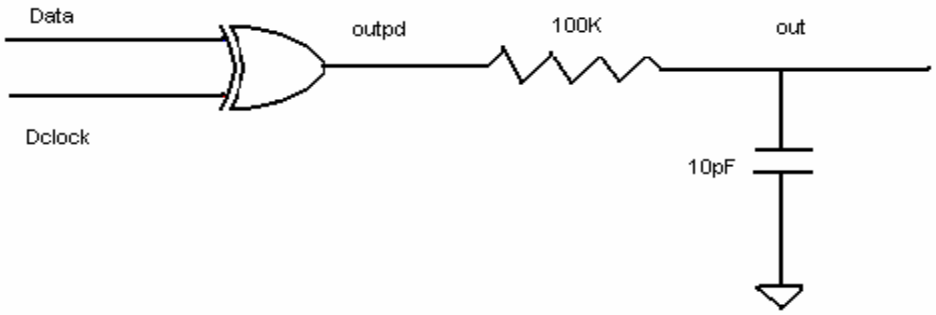
Netlist:

```
Xinv        VDD     outpd   outpdi  inverter
Xchgpp      VDD     outpdi  vinvco  chgpp
R1          Vinvco  vrc     20k
C1          vrc     0       20p     IC=350m
C2          vinvco  0       1p      IC=350m

.subckt     chgpp   VDD     outpdi  vinvco
M1          n1      Vp      VDD     VDD     PMOS L=1 W=20
M2          vinvco  outpdi  n1      VDD     PMOS L=1 W=20
M3          Vinvco  outpdi  n2      0       NMOS L=1 W=10
M4          n2      vn      0       0       NMOS L=1 W=10

Mp          Vp      vp      VDD     VDD     PMOS L=1 W=20
Ibias       vp      vn      DC      10u
Mn          vn      vn      0       0       NMOS L=1 W=10
.ends
```

Mingke Yu

19.5 Demonstrate, using simulations, how the outputs of the XOR PD can be equivalent when the loop is true or false locking (locking on a harmonic).

Solution:



By definition, when the output of the XOR is a pulse train with a 50 percent duty cycle, the DPLL is said to be in lock.

Consider the following two conditions:
In both conditions, we have the same in coming data pattern as "10000000" at 10ns per bit – 100MHz.

Case 1:
While Dclock is a clock signal of **100MHz**. And the rising edge of the Clock match with the rising edge of Data signal. – Dclock and Data are now locked. (Dclock has the same frequency as Data frequency)
The final voltage after RC loop filter = Vout1

Case 2:
While Dclock is a clock signal of **200MHz**. And the rising edge of the Clock match with the rising edge of Data signal. – Dclock and Data are now locked. (Dclock has the 2x frequency as Data frequency)
The final voltage after RC loop filter = Vout2

According to the SPICE simulation result:
Vout1 = Vout2

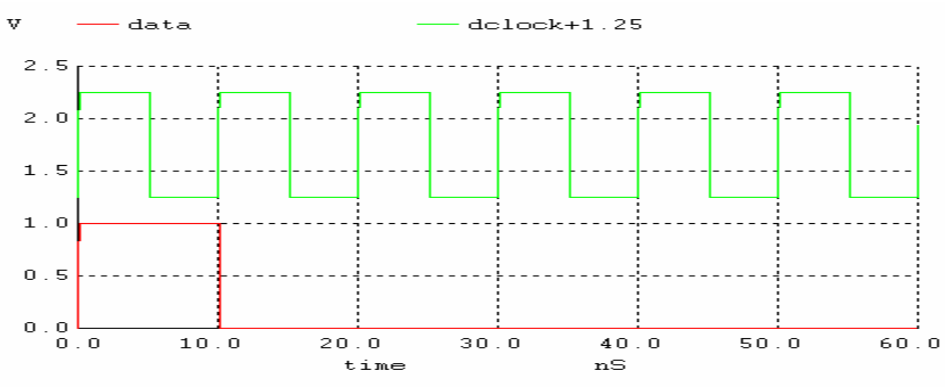
Conclusion:

Applying XOR gate to be the PD, it is possible to lock in harmonic frequency. The output of the XOR PD can be equivalent when loop is locking on a harmonic.

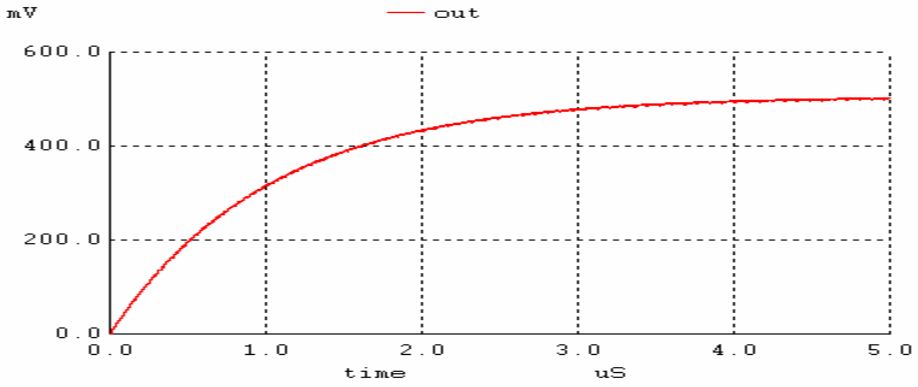
Simulation Result:

Case 1: Dclock frequency = 100MHz

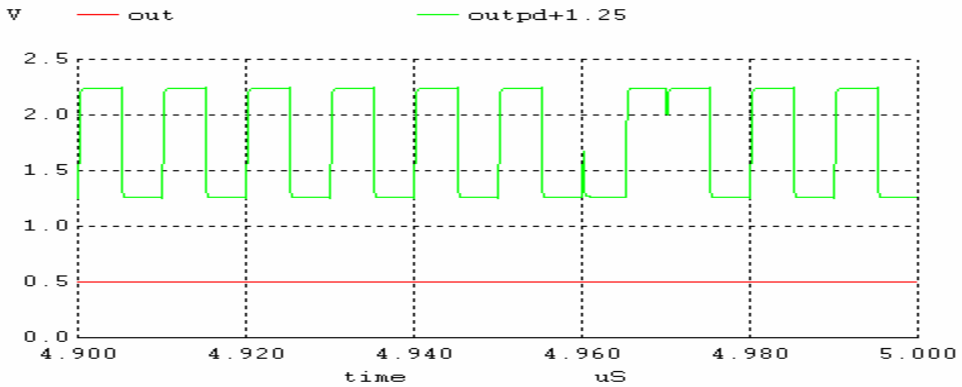
Data VS. Dclock



Vout1 -- final voltage after loop filter (= 500mv)

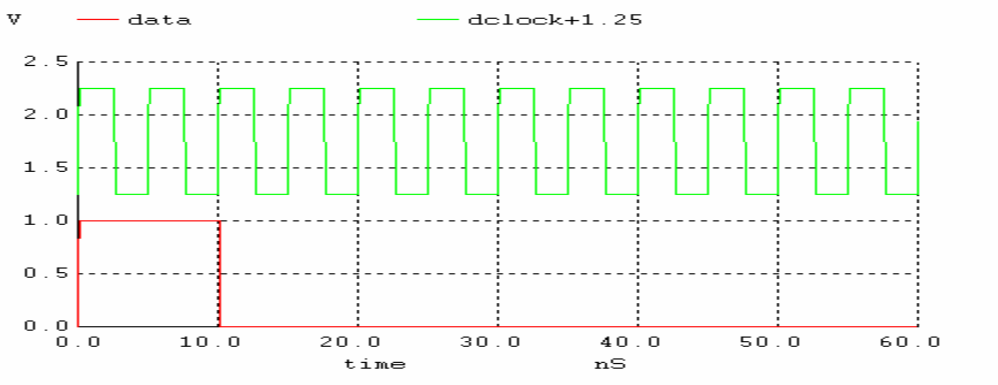


Vout1 – zoom in the end of the 5us simulation time

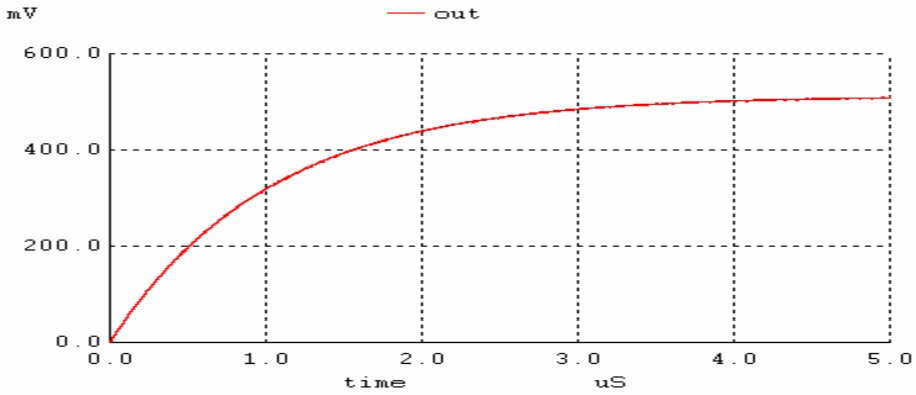


Case 2: Dclock frequency = 200MHz

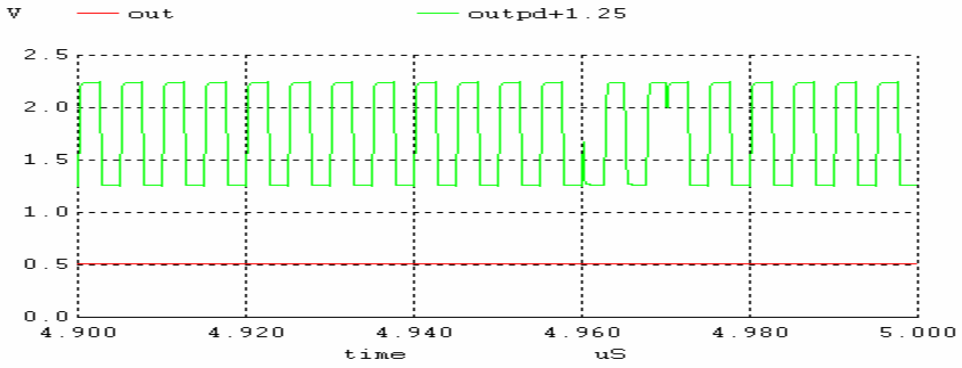
Data VS. Dclock



Vout2 -- final voltage after loop filter (= 500mv)



Vout2 – zoom in the end of the 5us simulation time



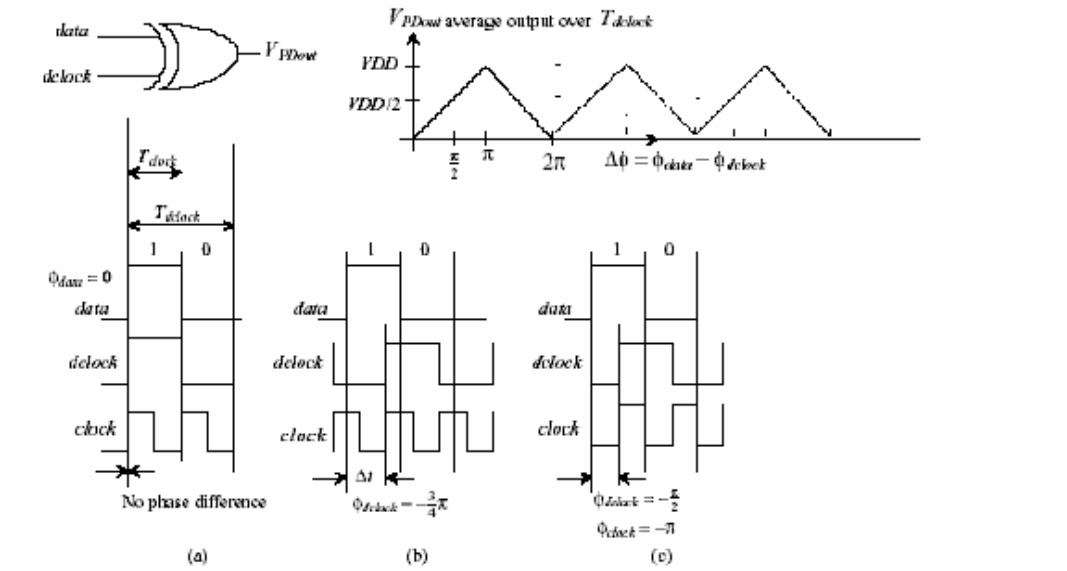
Netlist:

```
* EE497 Final -- Problem 19.5

.control
destroy all
run
plot out
plot out outpd+1.25 xlimit 4.9u 5u
plot data dclock+1.25 xlimit 0n 60n
.endc

.lib 50nm_models.cir
.option scale=50n
.tran .1n 5u UIC

VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0n 0 0 10n 80n
*vdclk  dclock  0       DC      0       pulse 0 1 0n 0 0 5n 10n
vdclk   dclock  0       DC      0       pulse 0 1 0n 0 0 2.5n 5n

XPD     VDD     data    dclock  outpd   XORPD

Rfilter outpd   out     100k
Cfilter out     0       10p

.subckt XORPD   VDD     A       B       outpd
M1      Ai      A       VDD     VDD     PMOS L=1 W=20
M2      Ai      A       0       0       NMOS L=1 W=10
M3      Bi      B       VDD     VDD     PMOS L=1 W=20
M4      Bi      B       0       0       NMOS L=1 W=10

M5      n1      A       VDD     VDD     PMOS L=1 W=20
M6      outpd   Ai      n1      VDD     PMOS L=1 W=20
M7      n1      B       VDD     VDD     PMOS L=1 W=20
M8      outpd   Bi      n1      VDD     PMOS L=1 W=20

M9      outpd   A       n2      0       NMOS L=1 W=10
M10     n2      B       0       0       NMOS L=1 W=10
M11     outpd   Ai      n3      0       NMOS L=1 W=10
M12     n3      Bi      0       0       NMOS L=1 W=10
.ends

.end
```

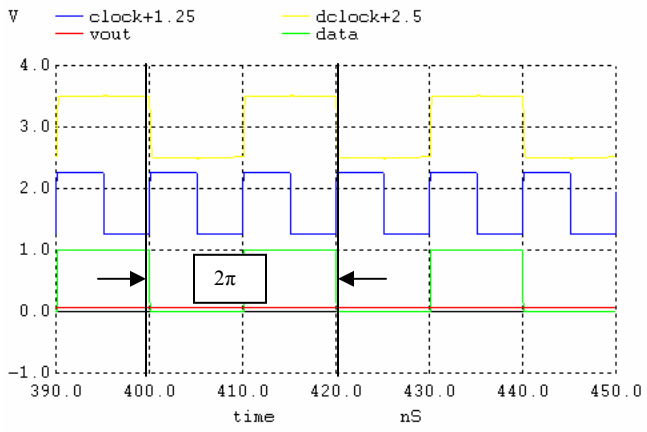
Jing Plaisted

Problem 19.6 Describe, in your own words and using simulations, what the dotted line in Fig19.8 indicates.

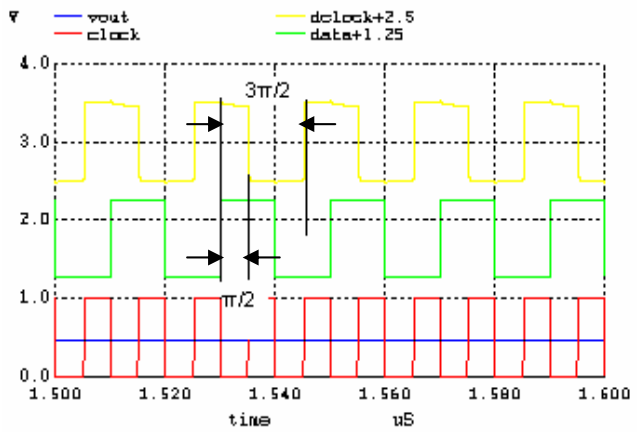
The dotted line indicates different phase shift. Since the data is repeated with period of 2π, as shown in figure below, when there is no phase difference, we can also say the phase different is 2π (figure 19.8(a) and (d)). When the phase is $\pi/2$, look at the different point, we also see $3\pi/2$ (figure (c) and (e). The result is valid only when the data and dclock frequency is the same.



The following figures showed the simulations. Figure (d) showed no phase different or the phase different is 2π, the output is 90 mV, which is very close to the ground.



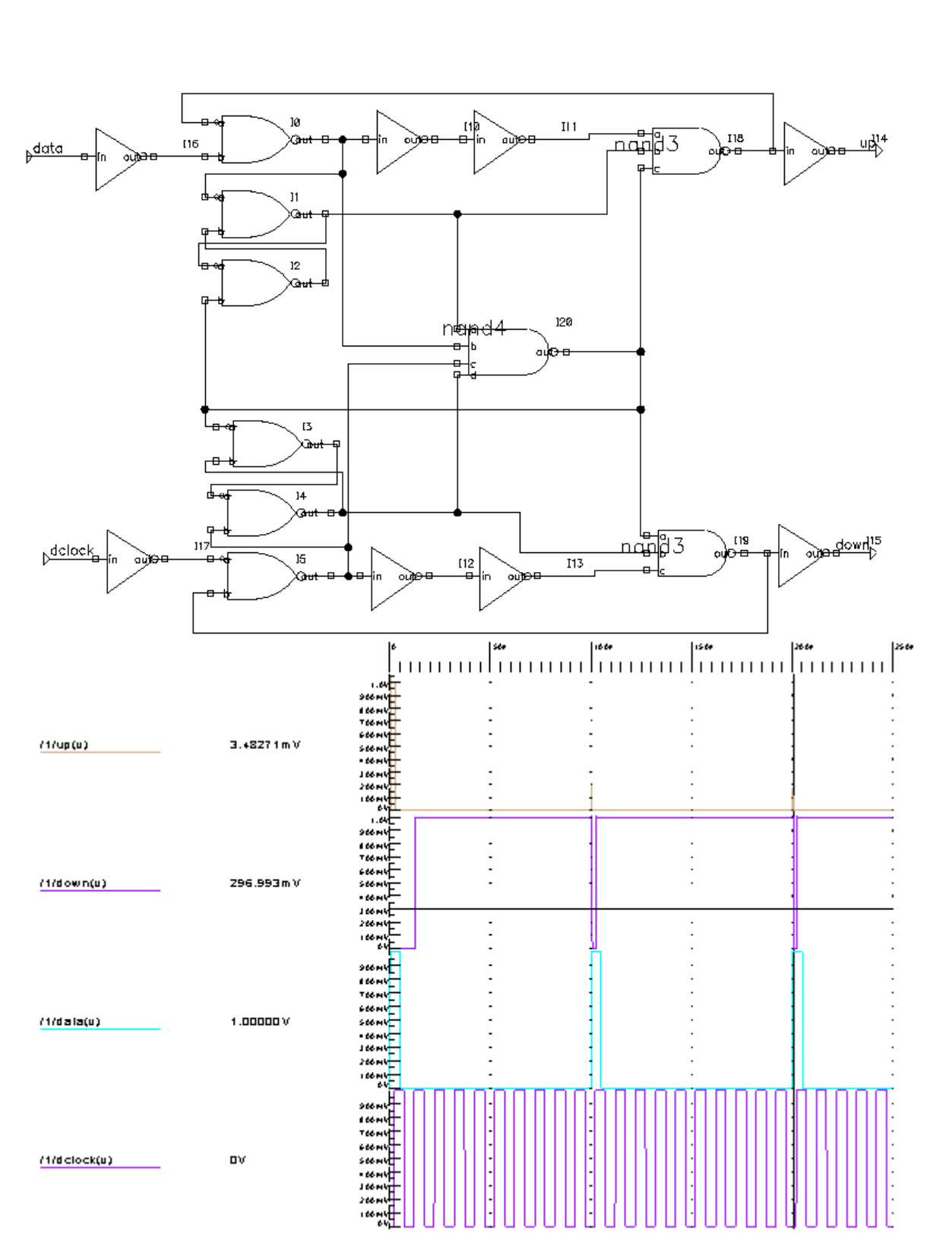
(d)



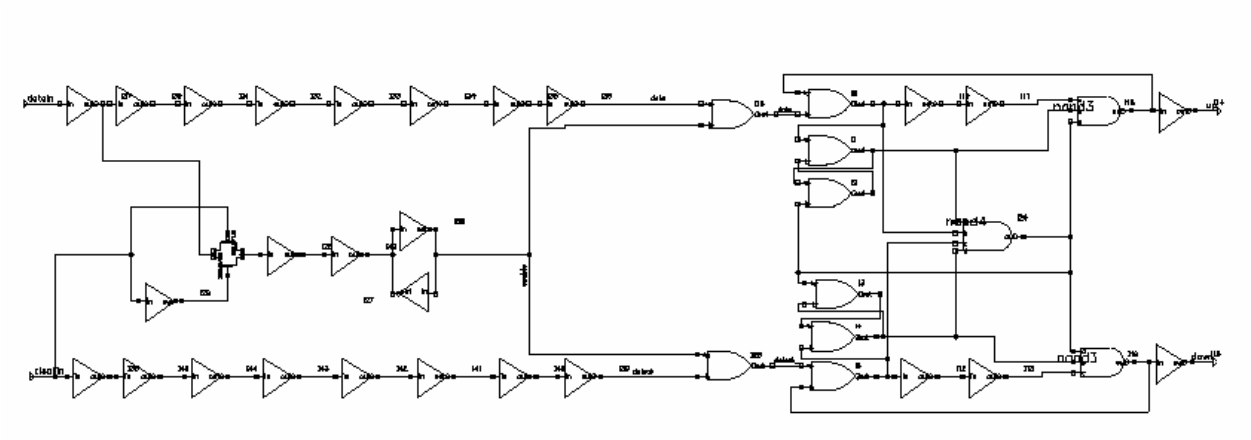
(e)

Figure (e) showed that phase difference between data and dclock is $\pi/2$ (or $3\pi/2$) the loop is locked and output is VDD/2, which is 0.5V.

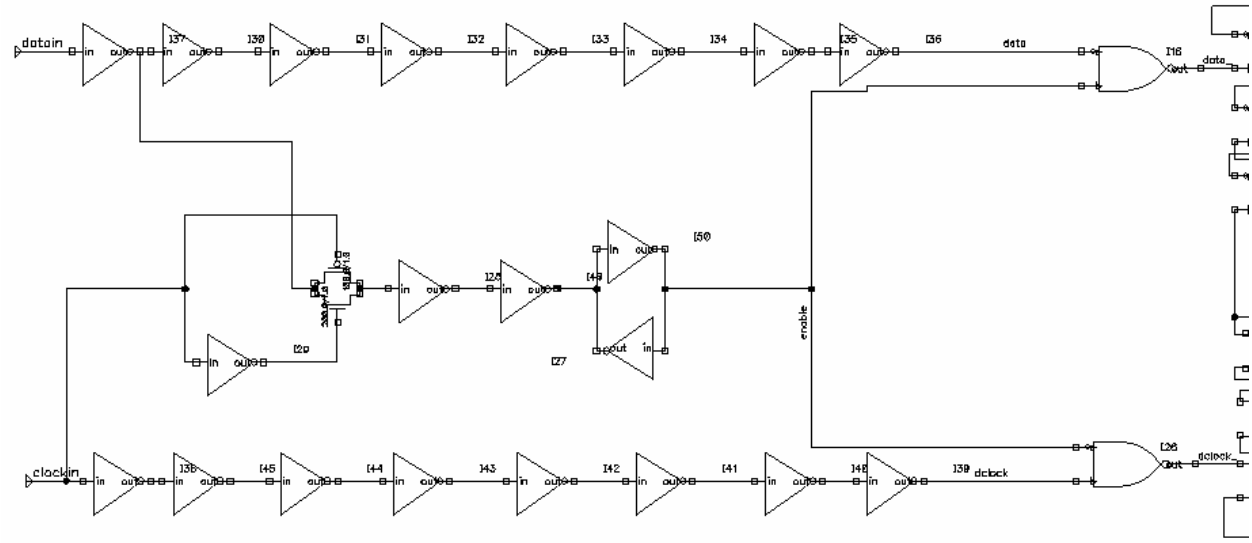
19.7 With PFD in figure 19.33, when data is 0, it is still trying to look for an edge of the data so the down signal will be always high and up signal always low.



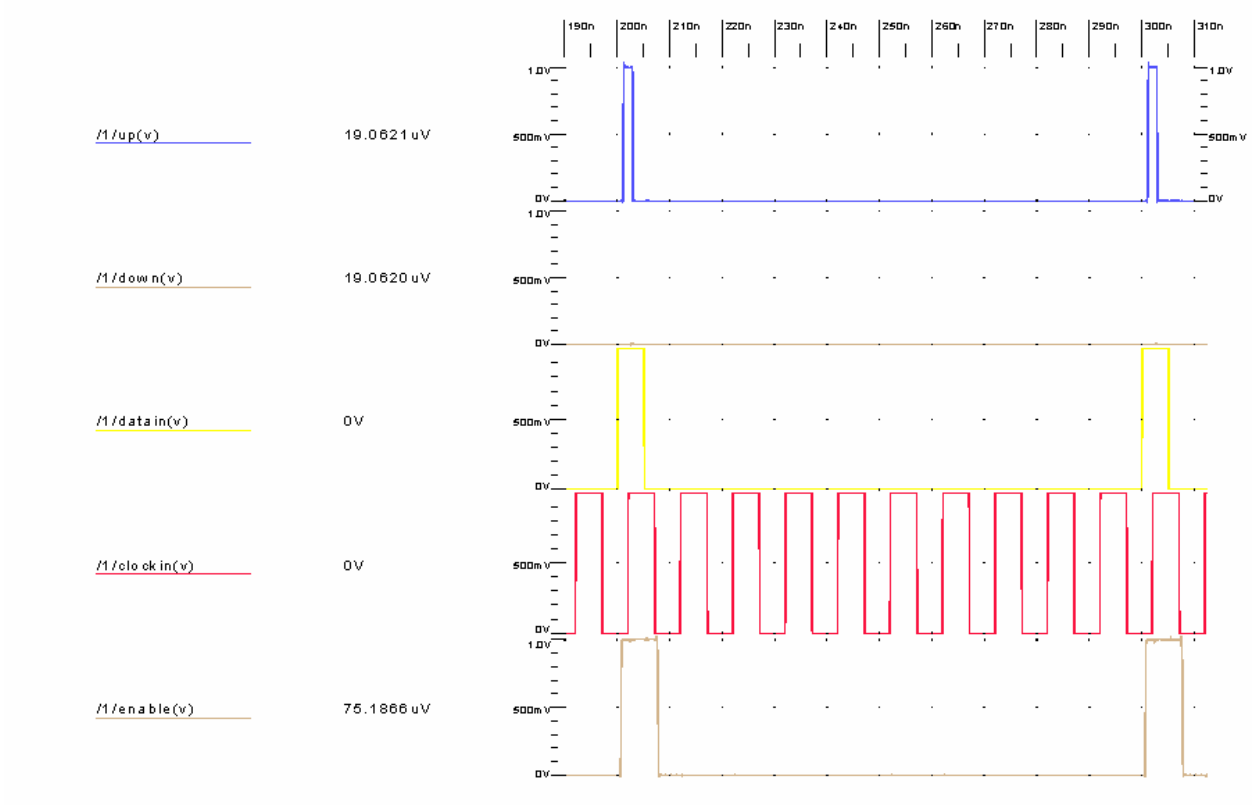
To avoid this from happening, we can add a clock swallowing circuit in front of the pfd. Pfd is enabled only when a rising edge of data is detected. The idea is to create an enable signal that controls clock and data that go in the pfd circuit. Every clock cycle, this circuit looks at data and see if it is a "1", if it is, the pfd is enabled, if not pfd is disabled. Since the enable signal is generated off the same detain edge as that we use to compare with clock, we need to delay the actual signal after enable goes high.



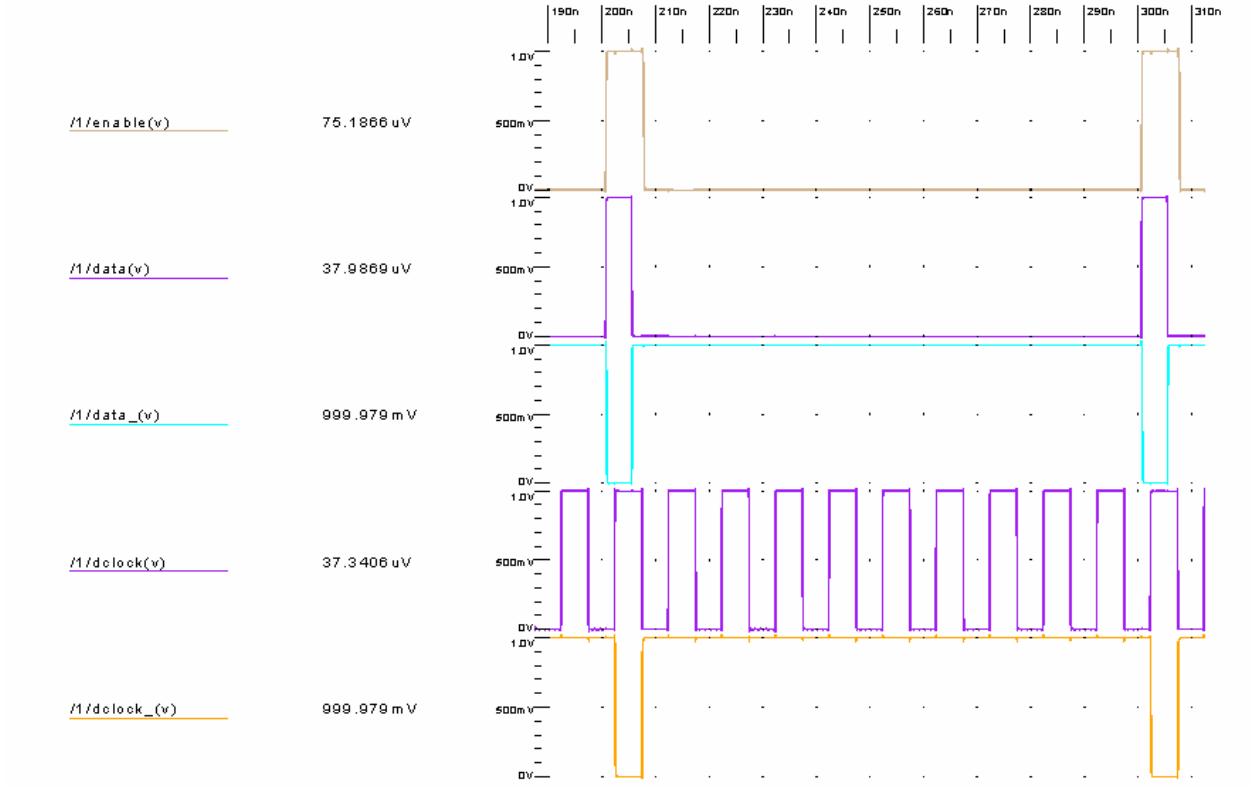
This is the zooming part of the clock swallowing circuit,



The simulation results are as below when the clock is behind data.



Data and clock are the signals right before PFD circuit that go into the nand gate with enable signal. Data_ and dclock_ are those after the nand gate.



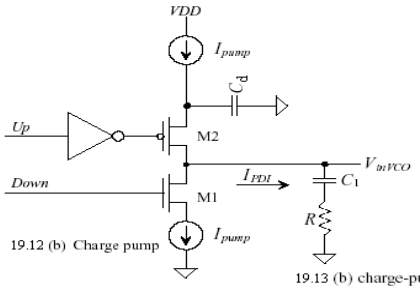
The drawback of this circuit is that it is hard to make perfect delay for clockin and datain. If there is process variations, the phase difference of clock and data won't be the same as clockin and datain.

Below is the netist:

```
*netlist of 19.7
.global vcc vdd
.options post
.options measout
.options scale=50n parhier=local


Vcc Vcc 0 DC 1
Vdatain datain 0  PULSE 0 1 0 0 0 5n 100n
Vdclockin clockin 0  PULSE 0 1 2n 0 0 5n 10n


.subckt nand4 a b c d out gnd vcc
MN0 net24 b net21 gnd N l=1.0 w=10.0 GEO=0.0
MN1 net21 c net022 gnd N l=1.0 w=10.0 GEO=0.0
MN3 net022 d gnd gnd N l=1.0 w=10.0 GEO=0.0
MN8 out a net24 gnd N l=1.0 w=10.0 GEO=0.0
MN2 out a vcc vcc P l=1.0 w=20.0 GEO=0.0
MP0 out b vcc vcc P l=1.0 w=20.0 GEO=0.0
MP1 out c vcc vcc P l=1.0 w=20.0 GEO=0.0
MP2 out d vcc vcc P l=1.0 w=20.0 GEO=0.0
.ends nand4

.subckt nand3 a b c out gnd vcc
MN0 net24 b net21 gnd N l=1.0 w=10.0 GEO=0.0
MN1 net21 c gnd gnd N l=1.0 w=10.0 GEO=0.0
MN8 out a net24 gnd N l=1.0 w=10.0 GEO=0.0
MN2 out a vcc vcc P l=1.0 w=20.0 GEO=0.0
MP0 out b vcc vcc P l=1.0 w=20.0 GEO=0.0
MP1 out c vcc vcc P l=1.0 w=20.0 GEO=0.0
.ends nand3

.subckt inv in out gnd vcc
MN2 out in gnd gnd N l=1.0 w=10.0 GEO=0.0
MP7 out in vcc vcc P l=1.0 w=20.0 GEO=0.0
.ends inv

.subckt nand a b out gnd vcc
MN0 net020 b gnd gnd N l=1.0 w=10.0 GEO=0.0
MN8 out a net020 gnd N l=1.0 w=10.0 GEO=0.0
MN2 out a vcc vcc P l=1.0 w=20.0 GEO=0.0
MP0 out b vcc vcc P l=1.0 w=20.0 GEO=0.0
.ends nand

* top cell:  pfdcs2

MN2 net115 net131 net079 gnd N l=1.0 w=100.0 GEO=0.0
MP7 net115 clockin net079 vcc P l=1.0 w=200.0 GEO=0.0
XI10 net167 net147 gnd vcc inv
```

```
XI11 net147 net145 gnd vcc inv
XI12 net157 net143 gnd vcc inv
XI13 net143 net141 gnd vcc inv
XI14 net170 up gnd vcc inv
XI15 net154 down gnd vcc inv
XI27 enable net80 gnd vcc inv
XI28 net079 net0102 gnd vcc inv
XI29 clockin net131 gnd vcc inv
XI30 net115 net129 gnd vcc inv
XI31 net129 net127 gnd vcc inv
XI32 net127 net125 gnd vcc inv
XI33 net125 net123 gnd vcc inv
XI34 net123 net121 gnd vcc inv
XI35 net121 net119 gnd vcc inv
XI36 net119 data gnd vcc inv
XI37 datain net115 gnd vcc inv
XI38 clockin net113 gnd vcc inv
XI39 net111 dclock gnd vcc inv
XI40 net109 net111 gnd vcc inv
XI41 net107 net109 gnd vcc inv
XI42 net105 net107 gnd vcc inv
XI43 net103 net105 gnd vcc inv
XI44 net101 net103 gnd vcc inv
XI45 net113 net101 gnd vcc inv
XI49 net0102 net80 gnd vcc inv
XI50 net80 enable gnd vcc inv
XI0 net170 data_ net167 gnd vcc nand
XI1 net167 net166 net164 gnd vcc nand
XI16 data enable data_ gnd vcc nand
XI2 net164 net163 net166 gnd vcc nand
XI26 enable dclock dclock_ gnd vcc nand
XI3 net163 net160 net158 gnd vcc nand
XI4 net158 net157 net160 gnd vcc nand
XI5 dclock_ net154 net157 gnd vcc nand
XI18 net145 net164 net163 net170 gnd vcc nand3
XI19 net163 net160 net141 net154 gnd vcc nand3
XI20 net164 net167 net157 net160 net163 gnd vcc nand4

.temp 25c
.tran 100ps 2000ns
```

19.8) First, to demonstrate the charge sharing between the charge pump (Fig19.12b) and loop filter (Fig19.13b), shown below, the C_2 capacitance is taken out. A drain capacitance C_d = 100fF is added at the drain of the current source and M2's source to intensify and better illustrate the charge sharing on the output, V_{inVCO}. Charge sharing is a design problem because it can cause static phase error or jitter.

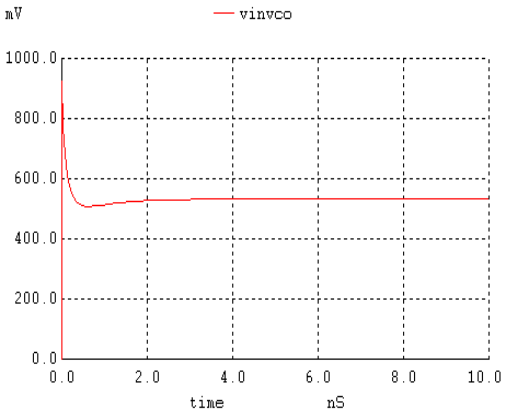


19.12 (b) Charge pump

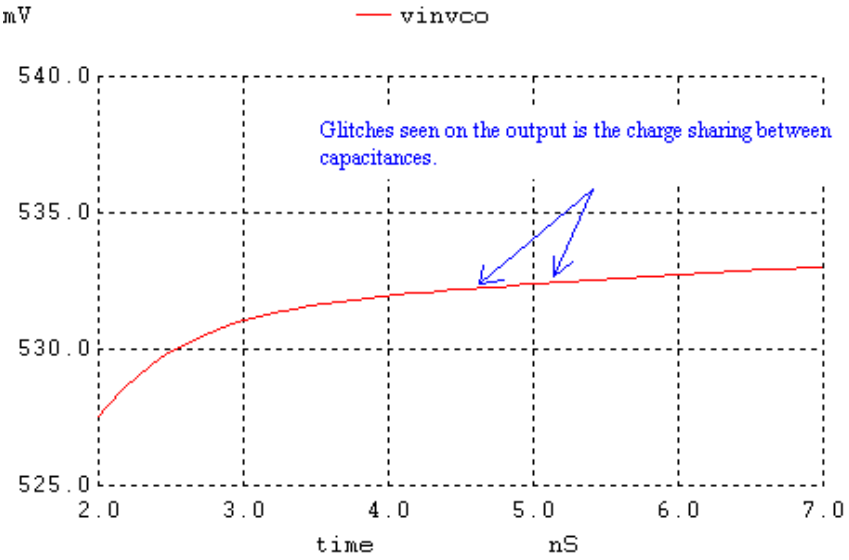
19.13 (b) charge-pump output

Netlist and Simulation Results:

```
.control
destroy all
run
plot vinvco
.endc
.option scale=50n
.tran .1n 10n UIC

VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 20n
vdclock dclock  0       DC      0       pulse 1 0 0 0 0 10n 20n

Mpb     Vp      Vp      VDD     VDD     PMOS L=2 W=100
Ibias   Vp      Vn      DC      10u
Mnb     Vn      Vn      0       0       NMOS L=2 W=50
Cp      Vp      0       100f

M4      upi     vdata   VDD     VDD     PMOS    L=1 W=20
M3      upi     vdata   0       0       NMOS    L=1 W=10
M2      Vinvco  upi     Vp      VDD     PMOS L=1 W=20
M1      vinvco  vdclock  Vn      0       NMOS L=1 W=10
C1      vinvco  vr      10p IC=. 5
R1      vr      0       20k
```



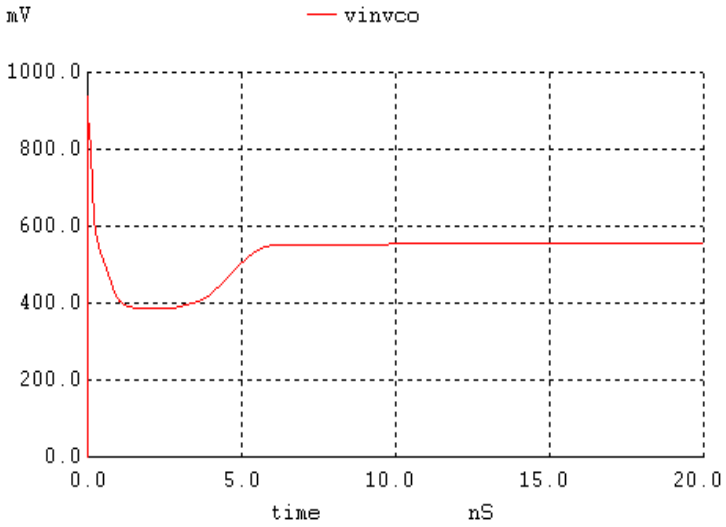
V_{invco} output.



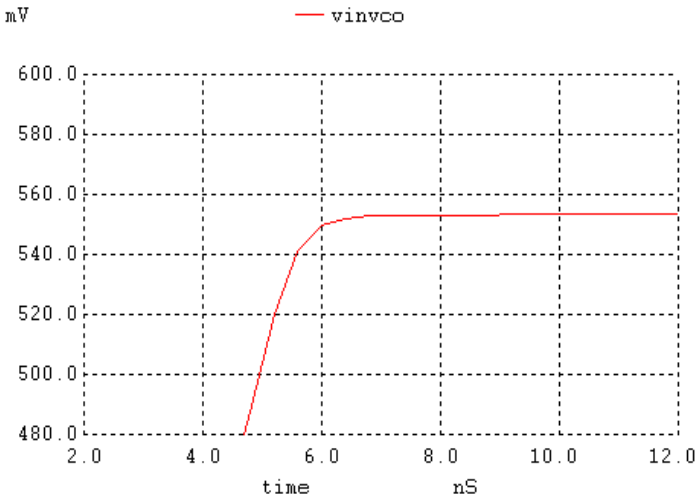
Closer look at output, showing charge sharing.

19.8) (Cont'd) Now we will show how using Fig19.37 with and without the x1 amplifier helps with charge sharing.

With the (x1) amplifier in Fig19.37: What is seen on the output V_{inVCO} when you use the x1 amplifier is the glitches or small spikes are eliminated (as shown below). Since the amplifier sets the drains of M1L and M2L in Fig19.37 to the same potential as V_{inVCO} (loop filter) node.

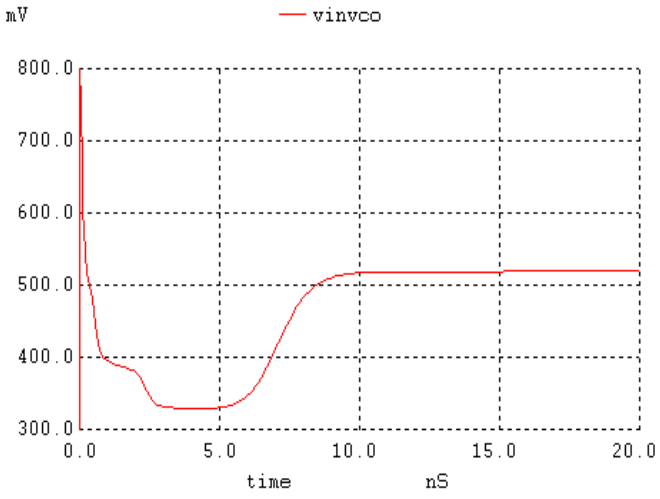


Output shown with the x1 amplifier.

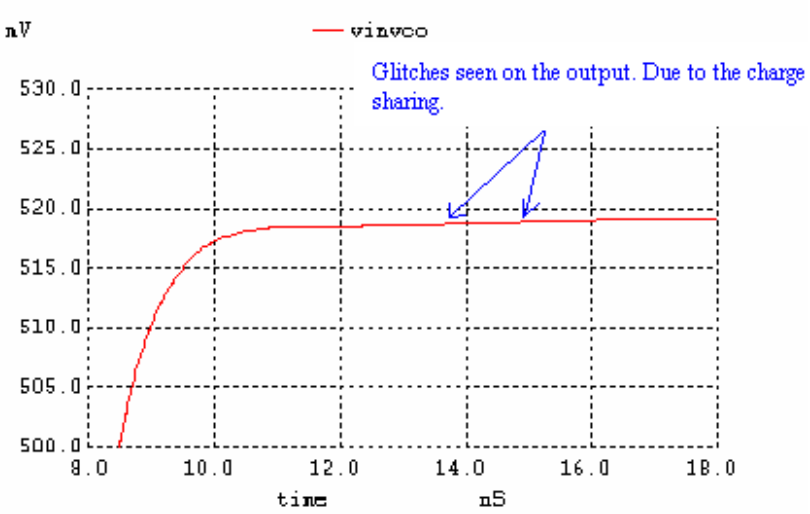


Closer look at output without any glitches.

Without the (x1) amplifier in Fig19.37: What is seen on the output V_{inVCO} when the x1 amplifier is not used results in glitches or small spikes (as shown below). Since the amplifier no longer sets the drains of M1L and M2L to the same potential as V_{inVCO}.



Output without the x1 amplifier.



Output seen with glitches (charge sharing).

19.8) (cont'd)
***Note:** Netlist included for the case when only the x1 amplifier was used.Same netlist used for the case w/o the x1 amplifier with proper modifications.

```
*** Figure 19.37 with a x1 amp ***

.control
destroy all
run
plot vinvco
.endc

.option scale=50n
.tran 1n 20n UIC


VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 20n
vdclock dclock  0       DC      0       pulse 1 0 0 0 0 10n 20n

Xinv1   VDD     vdata   upi     inverter
Xinv2   VDD     vdclock downi   inverter

M2L     vd12    vdata   Vpup    VDD     PMOS L=1 W=20
M2R     vinvco  upi     vpup    VDD     PMOS L=1 W=20
M1L     vd12    downi   vndwn   0       NMOS L=1 W=10
M1R     vinvco  vdclock vndwn   0       NMOS L=1 W=10

Mpb     Vp      Vp      VDD     VDD     PMOS L=2 W=100
Ibias   Vp      Vn      DC      10u
Mnb     Vn      Vn      0       0       NMOS L=2 W=50

Mpup    Vpup    Vp      VDD     VDD     PMOS L=2 W=100
Mndwn   Vndwn   Vn      0       0       NMOS L=2 W=50
Cd      vpup    0       100f

** Diff Amp Fig 18.17**
Mamp2   Vamp12 Vamp12 VDD     VDD     PMOS L=1 w=20
Mamp4   Vd12    Vamp12 VDD     VDD     PMOS L=1 W=20
Mamp1   Vamp12 Vinvco  Vamp5  0       NMOS L=1 W=10
Mamp3   Vd12    Vd12    Vamp5  0       NMOS L=1 W=10
Mamp5   Vamp5   Vamp12 0       0       NMOS L=1 W=10


R1      vinvco  vrc     20k
C1      vrc     0       10p IC= .5

.subckt inverter   VDD     in      out
M1      out     in      0       0       NMOS   L=1     W=10
M2      out     in      VDD     VDD     PMOS   L=1     W=20
.ends
```

Problem 19.9
Memory Circuits
Qawi Harvard
Boise State University

Using the VCO that generated the simulation data in Fig. 19.18 plot the VCO's center frequency against changes in *VDD*. Is this VCO insensitive to changes in *VDD*? Where does the sensitivity come from?

Begin by reviewing the design of the VCO in example 19.1. The delay of the VCO is determined by the capacitance located at the input and output of each stage (the input of one stage is connected to the output of another stage). Figure 1 shows the delay stage of the VCO. Using equation 19.19 you can find the total capacitance of the inverter delay stage.

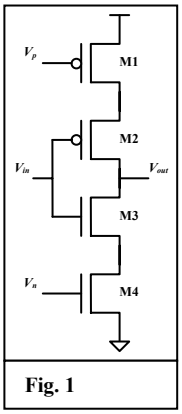


Fig. 1

$$C_{tot} = \frac{5}{2} C'_{ox} (W_p L_p + W_n L_n)$$

The derivation of C_{ox} can be found in chapter 5 of the text. Using $L_p = L_n = 1$, $W_p = 2W_n = 20$, and a scale factor of 50nm a value of $4.7fF$ is obtained for C_{tot}. Use C_{tot} and the relationship between voltage, capacitance, current and time, determine the time it takes to charge and discharge C_{tot}.

$$t_{chg} = C_{tot} \cdot \frac{V_{SP}}{I_{D1}} \quad , \; t_{dis} = C_{tot} \cdot \frac{VDD - V_{SP}}{I_{D4}}$$

t_{chg} is the time it takes to charge the capacitor from zero to V_{SP} above this voltage the inverter will pull the output high. t_{dis} is the time it takes to discharge the capacitor from *VDD* to V_{SP}. Setting $I_{D1} = I_{D4} = I_D$ the sum of t_{chg} and t_{dis} is simply an equation independent of V_{SP}. The oscillation frequency of the VCO with more than N=5 stages is found from the sum of the discharge and charge times.

$$t_{chg} + t_{dis} = \frac{C_{tot} VDD}{I_D}, \; f_{osc} = \frac{I_D}{N(t_{chg} + t_{dis})}$$

Using $I_D = 10\mu A$ (based on the $I_D - V_{GS}$, device sizes discussed in Chapter 9, $V_{inVCO} = VDD/2$), and $f_{osc} = 100MHz$ a value of N = 21 stages is calculated. Linearizing the control voltage and oscillating frequency is done by using an input transistor with a current mirror load. Figure 2 shows the linear voltage to current conversion technique used for the delay cells.

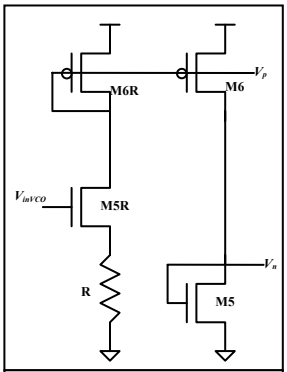


Fig. 2

Simulating the VCO with $V_{inVCO} = VDD/2$ the center oscillation frequency is found to be ~100MHz. Placing V_{inVCO} = 500mV on the input of the VCO and varying VDD a plot of the center frequency against changes in *VDD* is obtained. Figure 3 shows the result of the simulation data.

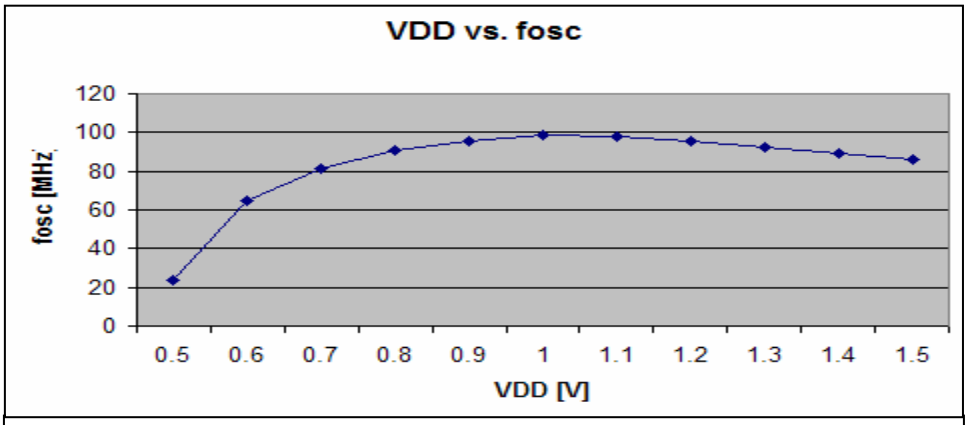


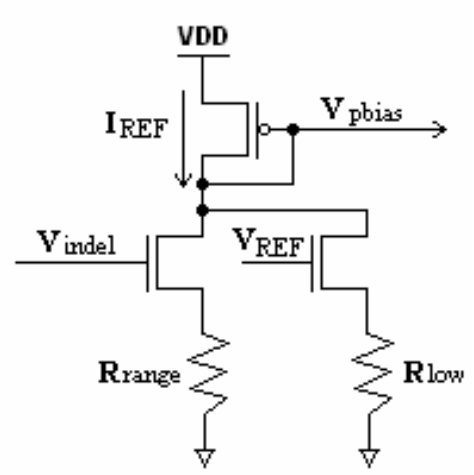
Fig. 3 Simulation data showing simulation results of the center frequency against changes in VDD

Analyzing figures 1 and 2 will give a clue as to where the variations of the center frequency come from. Simulating the current that flows in the delay stage (I_{D1} of figure 1) shows that with changes in *VDD* the extra current needed to charge C_{tot} up to a higher *VDD* increases the delay, and therefore reduces the center frequency of the VCO.

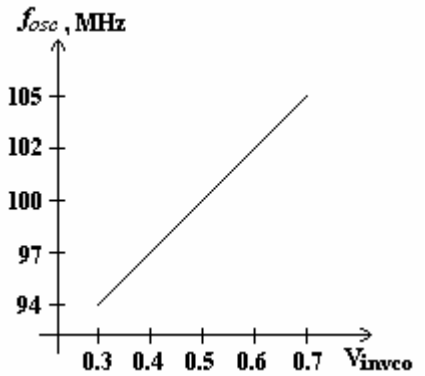
Ben Rivera

19.10) Discuss, and demonstrate with simulations, how to reduce the gain of the VCO used to generate the data in Fig. 19.18 (see Fig. 19.25 and the associated discussion). Your discussion should include some insight into the manufacturability of low-gain VCOs.

To lower the gain of the VCO we need to set the lower frequency range so when an input voltage of 0.3V is applied the frequency is set not by the lower voltage but by some other mechanism. Fig.19.25 accomplishes this by adding a resistor to the drain of the input NMOS transistor to ground. The resistor will pull a current set by the voltage on the drain of the NMOS transistor. Great results are obtained in a simulation environment but in reality changes on the power supply voltage will be seen by the resistor varying the output frequencies.
Another way of setting the lower frequency range is shown in the figure below, the gate of the NMOS is held at a generated voltage, V_{ref}, and the current being pulled away from the node is approximately $(V_{ref}-V_{thn})/R_{low}$.



Simulation results of the above schematic are shown below with Rrange=100k and Rlow=11.5k



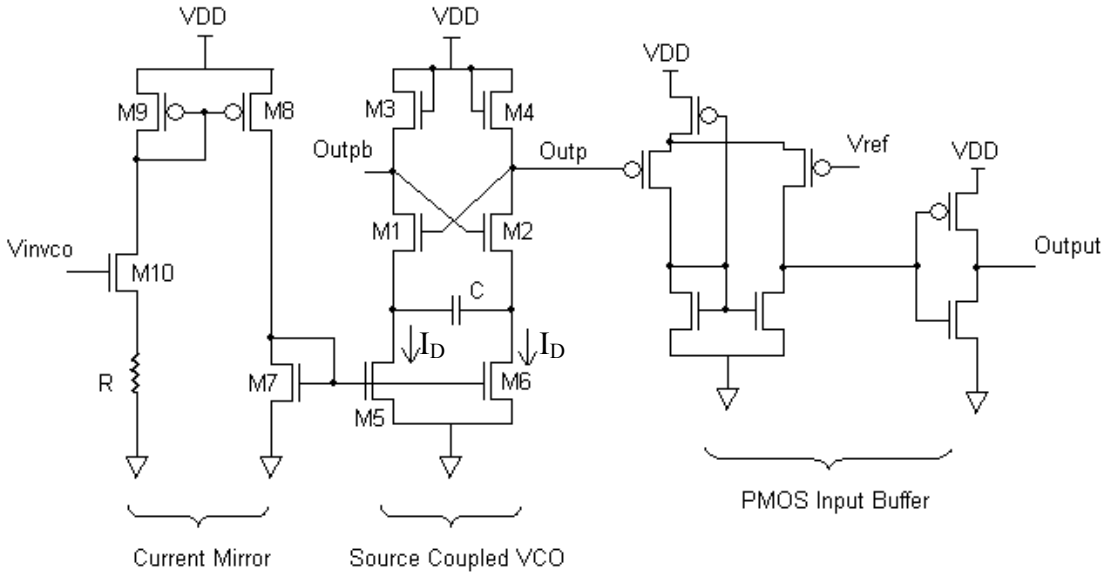
By changing the power supply voltage by 10% and then running the simulations with varying Vinvco voltages a comparison was made between Fig. 19.25 and the circuit shown here. Fig. 19.25 was found to vary by 21MHz while the circuit shown varied by only 3MHz. Again, this is to be expected because the current being pulled away is set by the Vref voltage and the resistor, Rlow.

In a manufacturing environment process shifts and operating temperatures will vary the resistances and the output frequencies will not be what is expected. To help overcome this we can add fuses to trim the resistances up or down as needed and allow us to control the output frequencies.

Solution to Problem 19.11 Johnny Yam

Design and simulate the operation of a 100MHz VCO using the topology seen in Figure 19.19a. The output of your VCO should be full logic levels. Your design should show a linearly frequency against V_{invco} curve. What is the gain of your design?

The following figure is the overall design. The purpose of the Current Mirror circuit is to mirror the desired current to the VCO and the PMOS Input Buffer is to restore full logic levels.



The gain of the source coupled VCO seen in Figure 19.19a is defined as

$f_{osc} = I_D/(4*C* V_{thn})$ (19.29)

The current (I_D) is proportional to the output frequency (Outp and Outpb). To obtain a linear relationship between the current and the output frequency, a current mirror circuit is employed. In addition M10 has to be a wide device, so the current is roughly (V_{invco} - V_{thn})/R. The size of M10 is set to be 200/1 (W/L), M5 to M9 are 100/1, and R is 55k. These settings will mirror 10uA to I_D when V_{invco} is 0.45V and also set the operating frequency. Now C can be determined by rearranging equation (19.29)

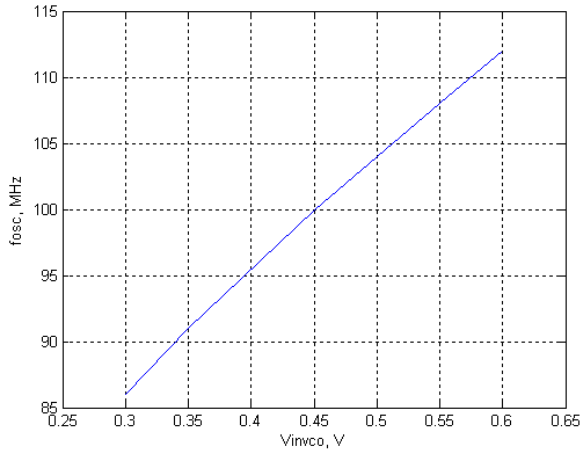
$C = I_D /(4* f_{osc} * V_{thn})$

C = 10uA/(4*100MHz*0.35V)

C = 72fF (100fF will be used in the simulation)

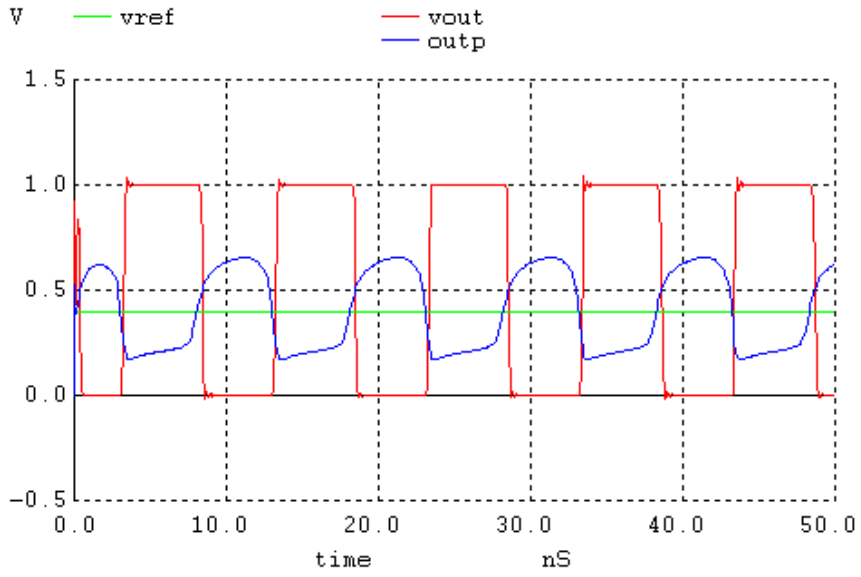
Sizing the MOSFETs in the VCO is not very critical. M1 and M2 are used as switches, so 30/1 is selected. The size of M3 and M4 is 10/10 for a relatively large resistance.

The plot below shows that Vinvco and f_{osc} are linearly related. The center frequency is 100MHz at V_{invco}=0.45. The maximum frequency is 112MHz and the minimum frequency is 86MHz. The gain of the VCO is 5.65×10^9 radians/Vs.



Either an inverter or the PMOS input buffer seen in Figure 18.21 can be used to restore full logic levels. The PMOS input buffer is chosen in the design. Since the change of V_{invco} may vary the output swing, the reference voltage (V_{ref}) ideally is at the 50% of the duty cycle of the output. To precisely adjust the V_{ref} correspondingly, a peak and valley detectors can be employed. The PMOS input buffer is more desired than a single inverter with a fixed switching point. For simplification, V_{ref} is 0.4V here.

The simulation results are shown in the following plot. A 100MHz square wave is achieved when V_{invco} = 0.45V.



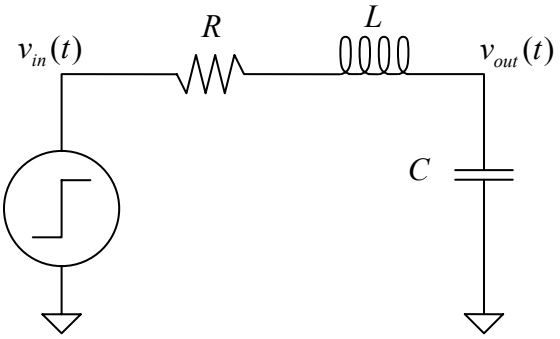
**** Netlist for Problem 19.11 ****

```
.control
destroy all
run
plot vout vref  outp
.endc

.option scale=50n
.tran 100p 50n UIC


VDD   VDD   0       DC     1
Vref  Vref  0       DC     0.4
VINVCO      VINVCO      0       DC     0.45

M1    outpb outp  X     0     NMOS L=1 W=30
M2    outp  outpb Y     0     NMOS L=1 W=30
M3    VDD   VDD   outpb  0     NMOS L=10 W=10
M4    VDD   VDD   outp   0     NMOS L=10 W=10
M5    X     N1    0     0     NMOS L=1 W=100
M6    Y     N1    0     0     NMOS L=1 W=100
M7    N1    N1    0     0     NMOS L=1 W=100
M8    VDD   N2    N1    VDD   PMOS L=1 W=100
M9    VDD   N2    N2    VDD   PMOS L=1 w=100
M10   N2    VINVCO      NR    0       NMOS L=1 w=200
R1    NR    0     55k
C1    X     Y     100f
x1    VDD   outp  vref  vout  pbuffer

**** subcircuit  Figure 18.21  ****

.subckt pbuffer       VDD   Vinm  Vinp  Vout
M1    Vom   Vinm  Vpp   VDD   PMOS L=1 W=20
M2    Vop   Vinp  Vpp   VDD   PMOS L=1 W=20
M6    Vpp   Vom   VDD   VDD   PMOS L=1 W=20

M3    Vom   Vom   0     0     NMOS L=1 W=10
M4    Vop   Vom   0     0     NMOS L=1 W=10

MI1   Vout  Vop   0     0     NMOS L=1 W=10
MI2   Vout  Vop   VDD   VDD   PMOS L=1 W=20
.ends
```

Problem 19.12

Using the step response of an RLC circuit, Figure 1 below, demonstrate how selection of the resistor, inductor, and capacitor affect the output voltage's damping factor and natural frequency. From this plot show how natural frequency and lock time are related.

Figure 1: A Second-Order Series RLC Circuit



Using a KVL, we can say that:

$$v_{in}(t) = i(t)R + L\frac{di(t)}{dt} + \frac{1}{C}\int i(t)dt$$

But our output is:

$$v_{out}(t) = \frac{1}{C}\int i(t)dt$$

Using this, we can obtain:

$$v_{in}(t) = RC\frac{dv_{out}(t)}{dt} + LC\frac{d^2v_{out}(t)}{dt} + v_{out}(t)$$

The LaPlace Transform converts into frequency domain:

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{\frac{1}{LC}}{s^2 + s\frac{R}{L} + \frac{1}{LC}}$$

This transfer function can be converted to the classical second-order form:

$$H(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2}$$

In our case then:

$$\omega_n = \frac{1}{\sqrt{LC}}$$
$$\zeta = \frac{R}{2}\sqrt{\frac{C}{L}}$$

Our natural frequency ω_n controls how quickly the system can react. The damping factor ζ dictates what type of response (under-damped, critically damped, or over-damped) will occur. Using the fact that our input is a step function:

$$v_{in}(t) = u(t) \Rightarrow V_{in}(s) = \frac{1}{s}$$

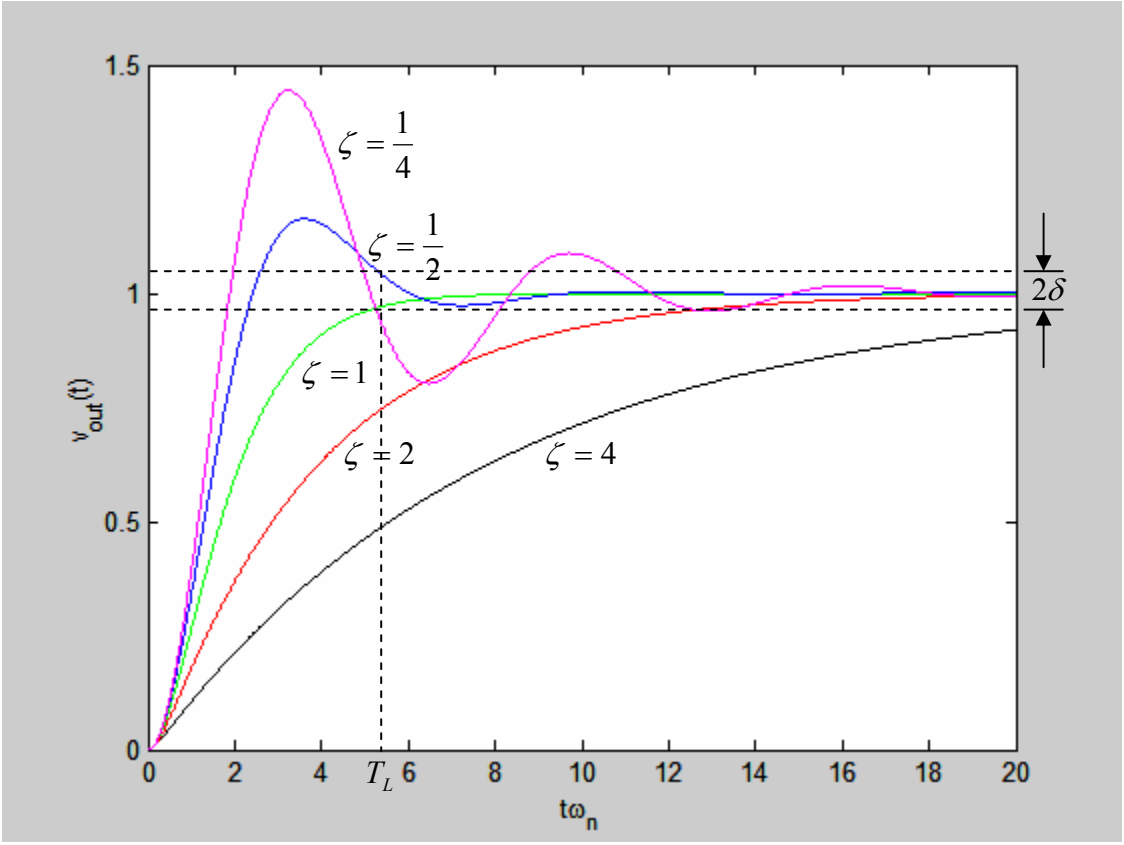
$$V_{out}(s) = H(s)V_{in}(s) = \frac{\frac{1}{LC}}{s^2 + s\left(\frac{R}{2}\sqrt{\frac{C}{L}}\right)\left(\sqrt{\frac{1}{LC}}\right) + \frac{1}{LC}}\frac{1}{s}$$

The inverse LaPlace Transform shows that the general solution form is:

$$v_{out}(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}}e^{-\zeta\omega_n t}\sin\left(\frac{\omega_n}{\sqrt{1-\zeta^2}}\right)$$

If $\zeta \geq 1$, the sine term doesn't exist in the equation. Figure 2 shows example responses of a system to a step response. Note in the figure that the lock time for $\zeta = \frac{1}{2}$ is illustrated. Also note that in Figure 2 the time index is normalized to the natural frequency. In conjunction with a step input of unity magnitude, Figure 2 can be appropriately scaled to match any second-order step input response.

Figure 2: Example Responses of a System to a Step Response



We can define 'lock time' as the time required for the response to settle within a certain percentage $\pm\delta$ of the input amplitude. If we use $\delta = 2\%$ (and our step function has an amplitude of one), this occurs when:

$$e^{-\zeta\omega_n T_L} = 0.02$$

An approximation yields:

$$\zeta\omega_n T_L = 4$$

Solving for T_L in the above equation gives:

$$T_L = \frac{4}{\zeta\omega_n}$$

So, as stated earlier, the larger the natural frequency, the more quickly the system can react to an input and reach steady state. Also note that for a given natural frequency, the quickest system is also the most oscillatory in nature.

Problem 19.13 Roger Porter

When the RC loop filter in *Figure 19.22* is replaced with the lag loop filter of *Figure 19.29*, a zero is added to the loop filter gain. The resulting loop filter gain is,

$$K_F = \frac{1 + j\omega R_2 C}{1 + j\omega (R_1 + R_2) C}$$ as seen in *Figure 19.29*

In *Figure 19.29* the equations for the natural frequency and damping factor are also given, they are:

$$\omega_n = \sqrt{\frac{K_{PD} K_{VCO}}{N (R_1 + R_2) C}}$$ (equation A)

$$\zeta = \frac{\omega_n}{2}\left(R_2 C + \frac{N}{K_{PD} K_{VCO}} \right)$$ (equation B)

A divide-by-two circuit is used in the feedback of the *DPLL in Example 19.2,* hence:

N = 2

Also found in *Example 19.2*

$$K_{PD} = \frac{VDD}{\pi}$$ Where VDD = 1V (see equation 19.6)

$K_{VCO} = 1.57 \times 10^6$ rads/V-s (see figure 19.26)

If we assume the same natural frequency and damping factor of *Example 19.2*

$\omega_n = 100 \times 10^6$ rads/s
$\zeta = 1$

then we can use equation A to find R_2C.

$$R_2 C = \frac{2\zeta}{\omega_n} \cdot \frac{N}{K_{PD} K_{VCO}} =$$ 2.86 ns

and equation B to find R_1C.

$$R_1 C = \frac{K_{PD} K_{VCO}}{N (\omega_n)^2} - R_2 C =$$ 2.25 ns

For simulation purposes, If we use a 5pF capacitor then we can find R_1 and R_2.

$R_1 = 450\ \Omega$
$R_2 = 572\ \Omega$

Looking at *Figure 19.23,* the output resistance of the XOR phase detector is not zero so we can lessen R_1.

When we simulate the circuit with the lag loop filter added we get:

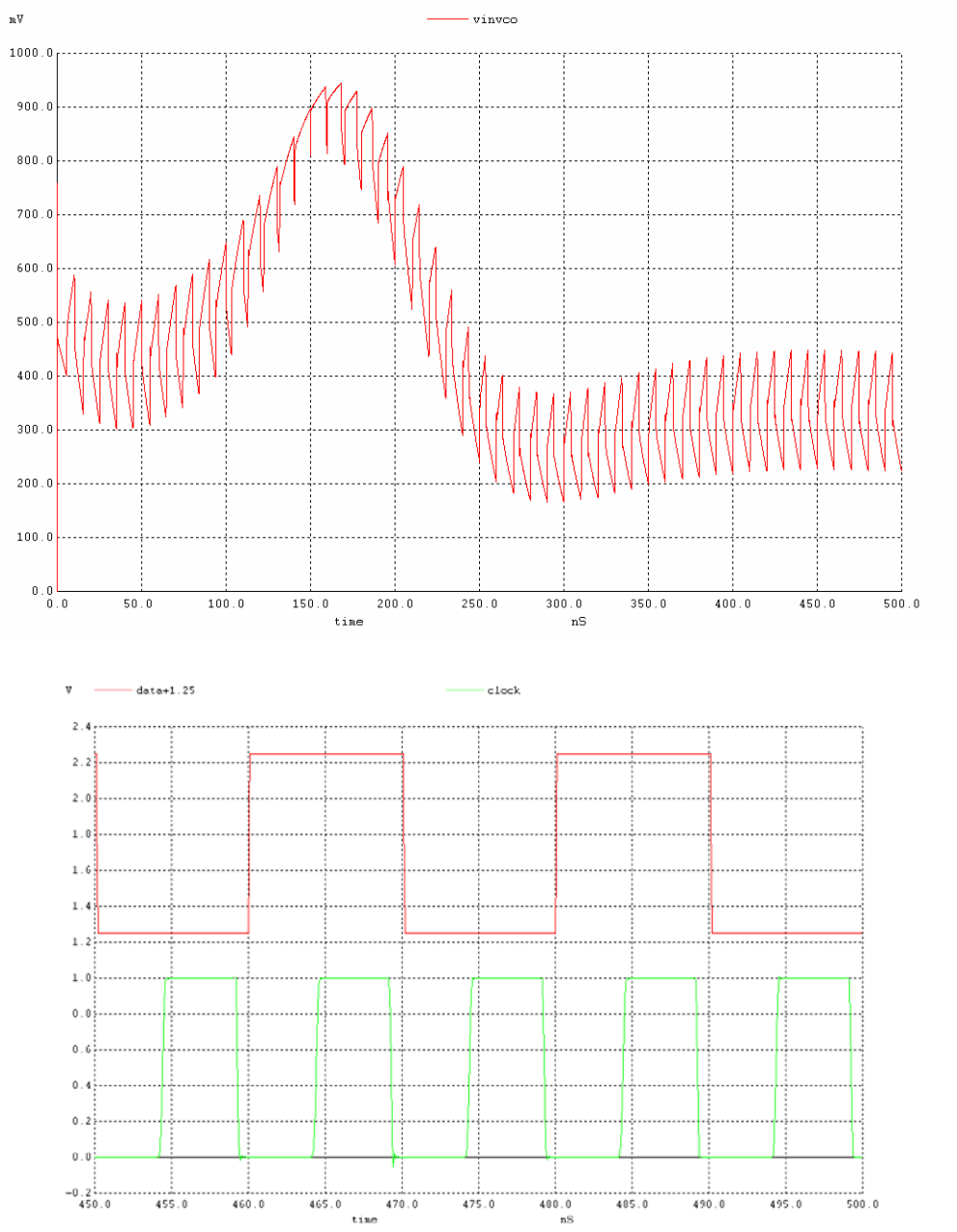


Figure 1 DPLL with a lag loop filter

Comparing these simulations to the ones with the RC loop filter (*Figure 19.27*) the jitter is about the same, the static phase error is less, but the time it takes to pull in the frequency and lock is slightly higher. Yet, looking at equation A and equation B, we see that with the zero we can make the pole small and now increase the gain of the VCO, which results in an increased frequency lock-in range and decreased lock time. (see equations 19.37 – 19.40)

19.14 Solution : Jagadeesh Gownipalli

Block diagram of Phase Frequency Detector(PFD) using Charge Pump

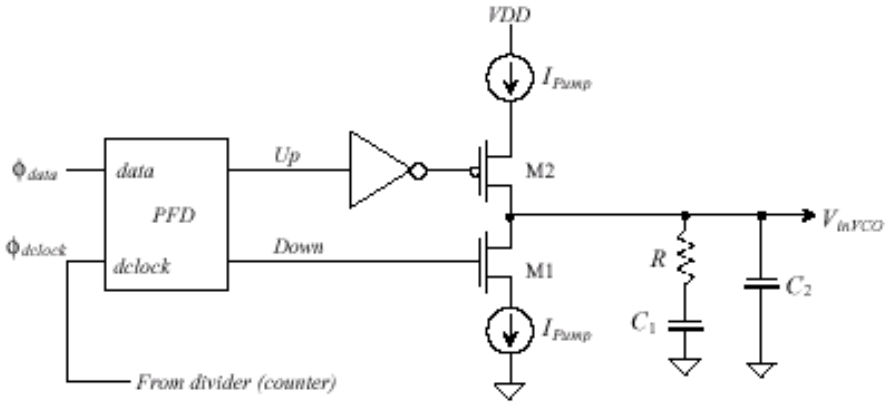


Fig.1

CMOS Implementation of PFD

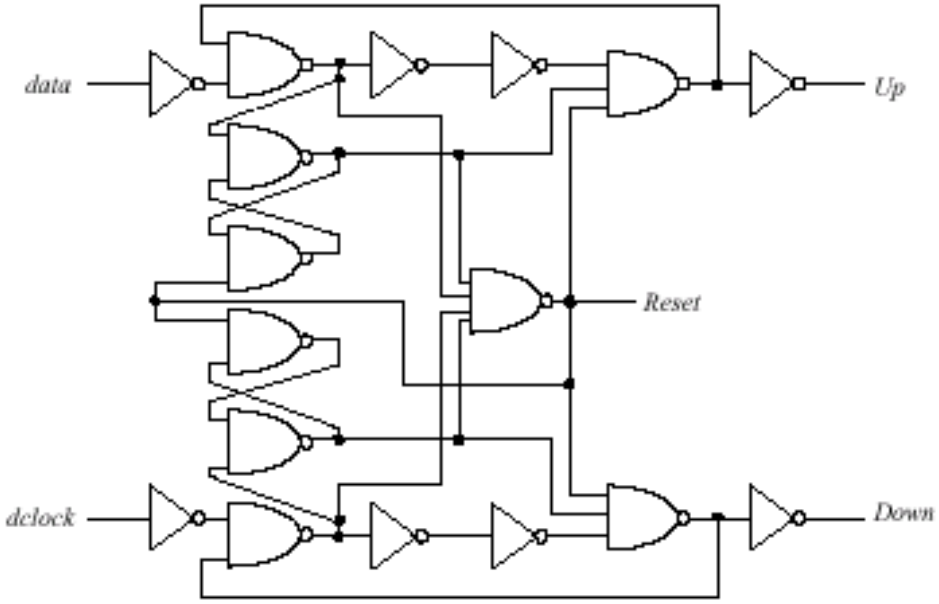


Fig.2

CMOS implementation of PFD is shown in *Fig 2*. The outputs of PFD depends on both frequency and phase of the inputs.When dclock is lagging the data the output of PDF is Up pulse indicating edge of dclock needs to speed up I.e. VCO's input voltage should increase.When data is lagging dclock then Down pulse goes high indicating dclock should slow down I.e. VCO's input voltage should decrease..

The outputs of PFD is combined into single output for driving the loop filter using charge pump. The whole circuit is shown in *Fig1*.

Using PFD, the phase difference between data and dclock is given by

$\Delta\phi = \frac{\Delta t}{Tclock}.2\pi$ (radians).

Where Tclock is period of data and Δt is time difference between Tclock and Tdclock

The output voltage of PFD using charge pump is given by

$$I_{PDI} = \frac{Ipump - (-Ipump)}{4.\pi}.\Delta\phi = K_{PDI}.\Delta\phi$$

where $K_{PDI} = \frac{Ipump}{2.\pi}$ is gain of PFD and Charge pump.

From the above equation we can say that when $\Delta\phi$ is very small but not equal to zero we can see that gain of PDF charge pump goes to zero. This is called Dead Zone.

Below are the simulations used to calculate the dead zone and values are tabulated. Table 1's Δt and I_{PDI} are from simulation results from Fig 5a – Fig12a and $\Delta\phi$, Ipump and K_{PDI} are calculated.

Δt	Phase Diff $\Delta\phi$	I_{PDI}	Ipump(uA)	KPDI(A/rad)
-5	-180	-4.95	-9.9	-3.1508593
-4	-144	-3.9	-9.75	-3.103119
-3	-108	-2.85	-9.5	-3.0235519
-2	-72	-1.8	-9	-2.8644176
-1	-36	-0.8	-8	-2.5461489
-0.5	-18	-0.35	-7	-2.2278803
-0.15	-5.4	-0.02	-1.333333333	-0.4243582
-0.1	-3.6	0	0	0
-0.09	-3.24	0	0	0
-0.05	-1.8	0	0	0
0	0	0	0	0
0.05	1.8	0	0	0
0.09	3.24	0	0	0
0.1	3.6	0	0	0
0.15	5.4	0.02	1.333333333	0.42435816
0.5	18	0.35	7	2.22788033
1	36	0.8	8	2.54614895
2	72	1.8	9	2.86441757
3	108	2.85	9.5	3.02355188
4	144	3.9	9.75	3.10311903
5	180	4.95	9.9	3.15085933

Table 1

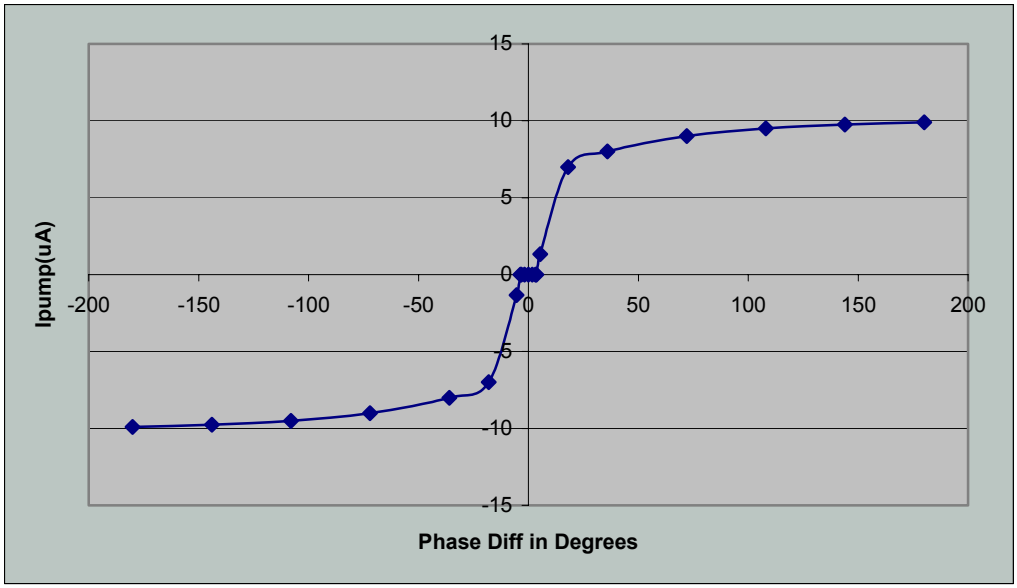


Fig 3

Fig 4 is zoomed version for Fig 3 to see dead zone clearly.

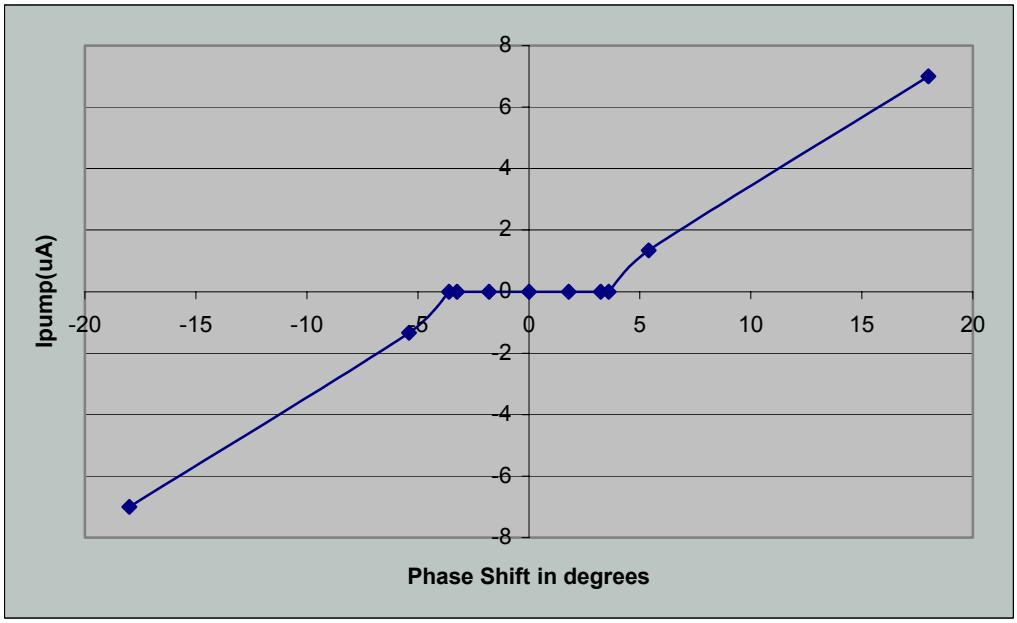


Fig 4

From the above Fig 4 we see dead zone from $-\frac{\pi}{36}$ (5 degrees) to $+\frac{\pi}{36}$ (5degrees) I.e.

$\Delta\phi$ from $+\frac{\pi}{36}$ to $-\frac{\pi}{36}$ degrees where gain is decreasing to zero.

Spice File

```
** Final 19.14 EE597 Jagadeesh Gownipalli ***
.control
destroy all
run
plot dclock data+1.25 up+2.5 down+3.75
plot vload#branch ylimit -20000n 20000n
plot vinvco
.endc

.option scale=50n
.tran 0.1n 20n 0n 10p UIC


VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 5n 10n
vdclock dclock  0       DC      0       pulse 0 1 0.0.08n 0 0 5n 10n
vload   vinvco1 vinvco  DC      0

X1      VDD     data    dclock  up      down    pfd
Xinv1   VDD     up      upi     inverter
Xinv2   VDD     down    downi   inverter

M2L     vnx     up      Vpup    VDD     PMOS L=1 W=20
M2R     vinvco1 upi     vpup    VDD     PMOS L=1 W=20
M1L     vnx     downi   vndwn   0       NMOS L=1 W=10
M1R     vinvco1 down    vndwn   0       NMOS L=1 W=10

Mpb     Vp      Vp      VDD     VDD     PMOS L=2 W=100
Ibias   Vp      Vn      DC      10u
Mnb     Vn      Vn      0       0       NMOS L=2 W=50

Mpup    Vpup    Vp      VDD     VDD     PMOS L=2 W=100
Mndwn   Vndwn   Vn      0       0       NMOS L=2 W=50


R1      vinvco  vrc     20k
C1      vrc     0       10p
C2      vinvco  0       1p


.subckt pfd     VDD     data    dclock  up      down
X1      VDD     data    n1      inverter
X2      VDD     n1      n5      n2      nand
X3      VDD     n2      n3      inverter
X4      VDD     n3      n4      inverter
X5      VDD     n4      n6      reset   n5      nand3
X6      VDD     n5      up      inverter
X7      VDD     n2      n7      n6      nand
X8      VDD     n6      reset   n7      nand
X9      VDD     reset   n8      n9      nand
X10     VDD     n9      n11     n8      nand
X11     VDD     n6      n2      n11     n8      reset   nand4
X12     VDD     dclock  n10     inverter
X13     VDD     n10     n14     n11     nand
X14     VDD     n11     n12     inverter
X15     VDD     n12     n13     inverter
X16     VDD     reset   n8      n13     n14     nand3
X17     VDD     n14     down    inverter
.ends

.subckt nand    VDD     A       B       ANANDB
M1      n1      A       0       0       NMOS    L=1     W=10
M2      ANANDB  B       n1      0       NMOS    L=1     W=10
M3      ANANDB  A       VDD     VDD     PMOS    L=1     W=20
M4      ANANDB  B       VDD     VDD     PMOS    L=1     W=20
.ends
```

```
.subckt inverter       VDD     in      out
M1      out     in      0       0       NMOS    L=1     W=10
M2      out     in      VDD     VDD     PMOS    L=1     W=20
.ends

.subckt nand3  VDD     A       B       C       OUT
M1      n1      A       0       0       NMOS    L=1     W=10
M2      n2      B       n1      0       NMOS    L=1     W=10
M3      OUT     C       n2      0       NMOS    L=1     W=10
M4      OUT     A       VDD     VDD     PMOS    L=1     W=20
M5      OUT     B       VDD     VDD     PMOS    L=1     W=20
M6      OUT     C       VDD     VDD     PMOS    L=1     W=20
.ends

.subckt nand4  VDD     A       B       C       D       OUT
M1      n1      A       0       0       NMOS    L=1     W=10
M2      n2      B       n1      0       NMOS    L=1     W=10
M3      n3      C       n2      0       NMOS    L=1     W=10
M4      OUT     D       n3      0       NMOS    L=1     W=10
M5      OUT     A       VDD     VDD     PMOS    L=1     W=20
M6      OUT     B       VDD     VDD     PMOS    L=1     W=20
M7      OUT     C       VDD     VDD     PMOS    L=1     W=20
M8      OUT     D       VDD     VDD     PMOS    L=1     W=20
.ends
```

Below are the simulation results for different values of Δt

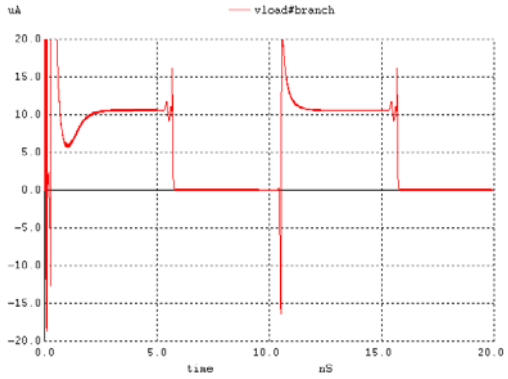


Fig 5a :Charge pump I_{PDI} for Δt =5ns

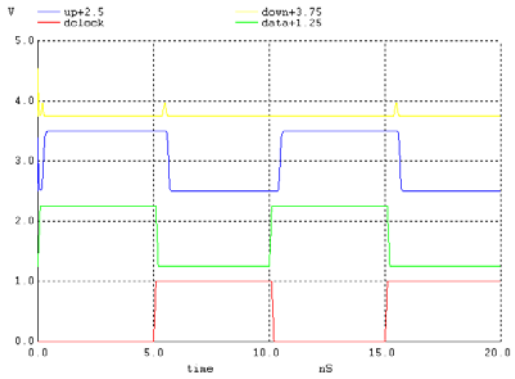


Fig 5b :PFD output for Δt =5ns

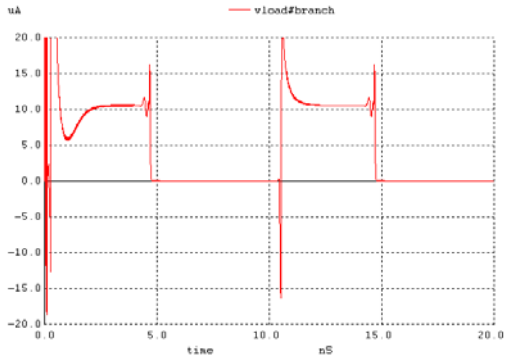


Fig 6a :Charge pump I_{PDI} for Δt =4ns

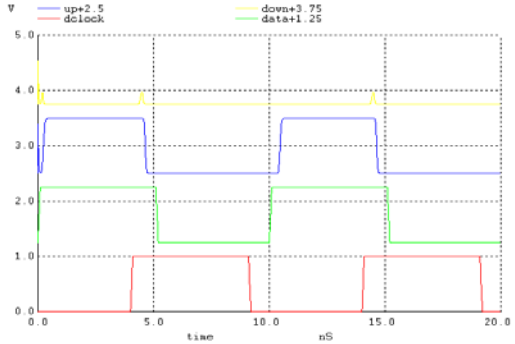


Fig 6b :PFD output for Δt =4ns

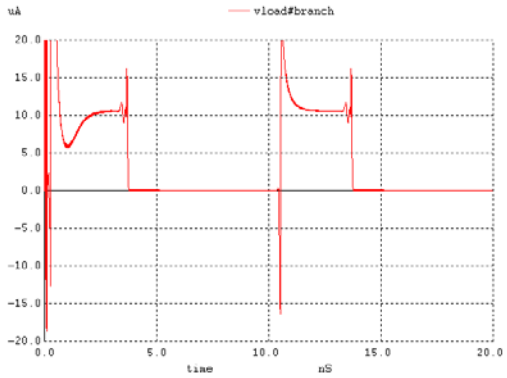


Fig 7a :Charge pump I_{PDI} for Δt =3ns

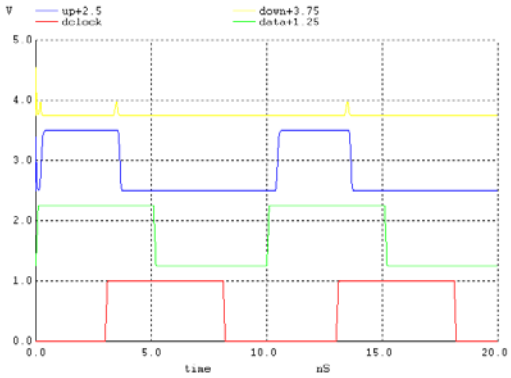


Fig 7b :PFD output for Δt =3ns

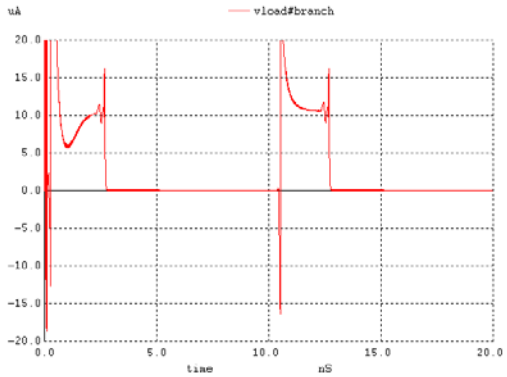


Fig 8a :Charge pump I_{PDI} for Δt =2ns

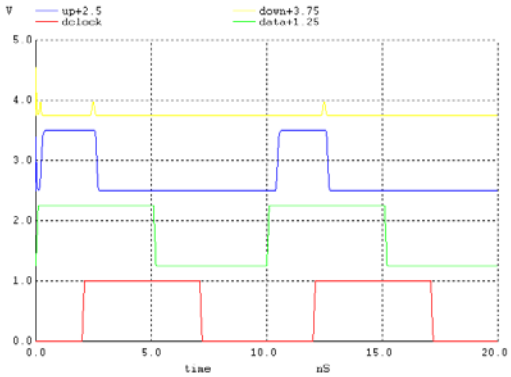


Fig 8b :PFD output for Δt =3ns

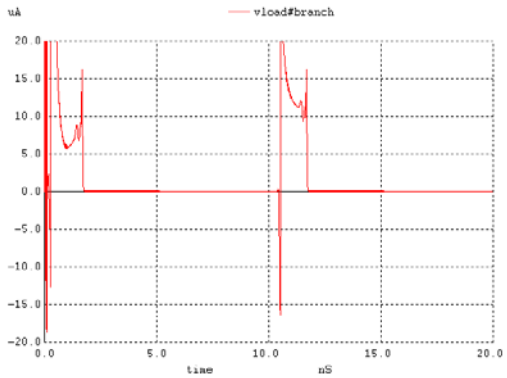


Fig 9a :Charge pump I_{PDI} for Δt =1ns

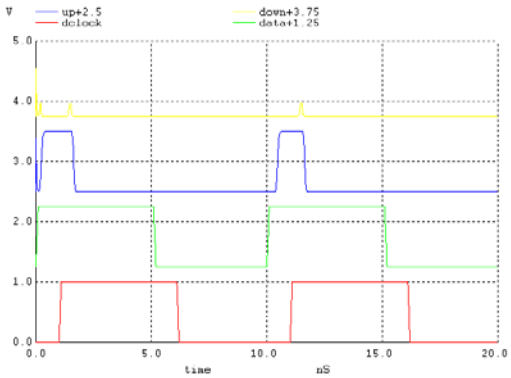


Fig 9b :PFD output for Δt =1ns

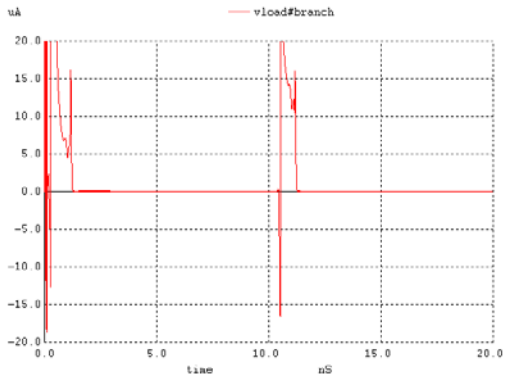


Fig 10a :Charge pump I_{PDI} for Δt =0.5ns

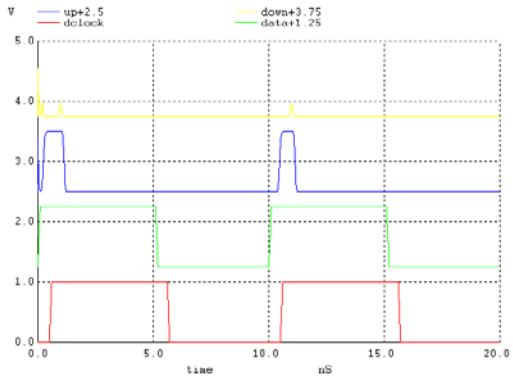


Fig 10b :PFD output for Δt =0.5ns

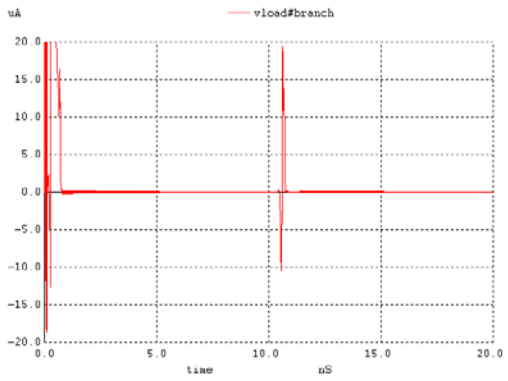


Fig 11a :Charge pump I_{PDI} for Δt =0.1ns

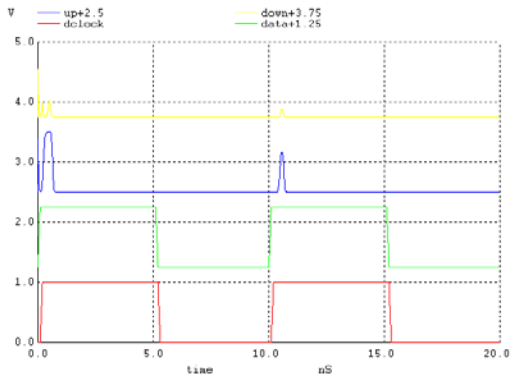


Fig 11b :PFD output for Δt =0.1ns

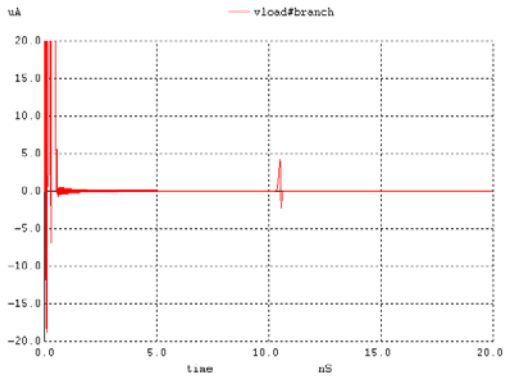


Fig 12a :Charge pump I_{PDI} for Δt =0.08ns

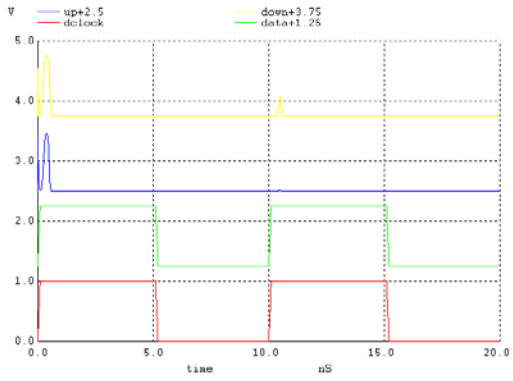
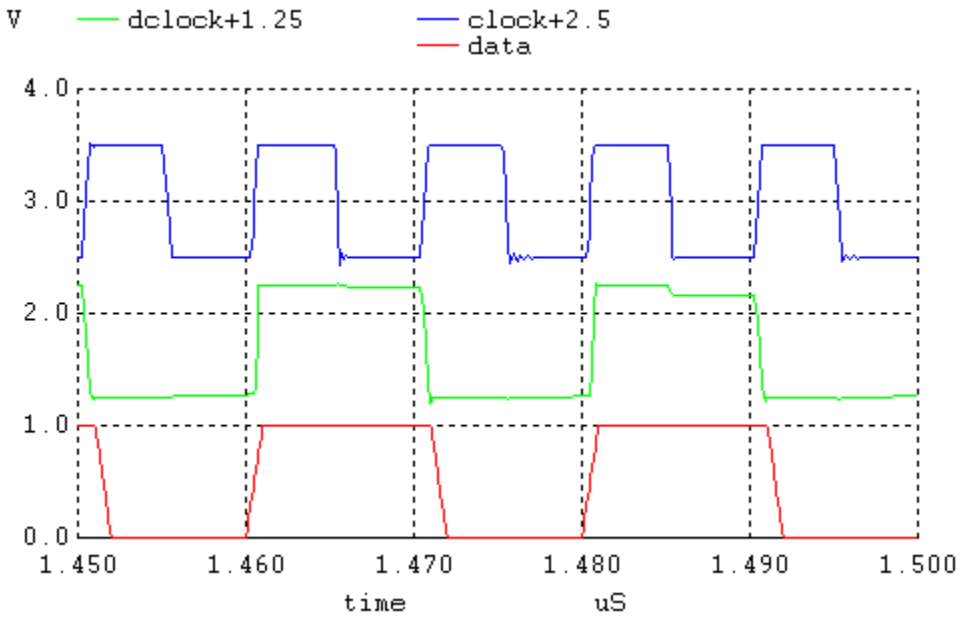


Fig 12b :PFD output for Δt =0.08ns

Solution to problem 19.15

To start with, let's simulate the fig 19.37 as it is and plot the graph:



Note: Iup = Idown for this figure is 10 uA. The loop is locked. Once the VCO is settled, the dclock and data transition on the rising edge of the clock.

Static Phase Error: When the loop is locked and if the dclock don't transition with the rising edge of the clock, than the delta between the rising edge of the clock and the rising edge of the dclock is called Static Phase Error.

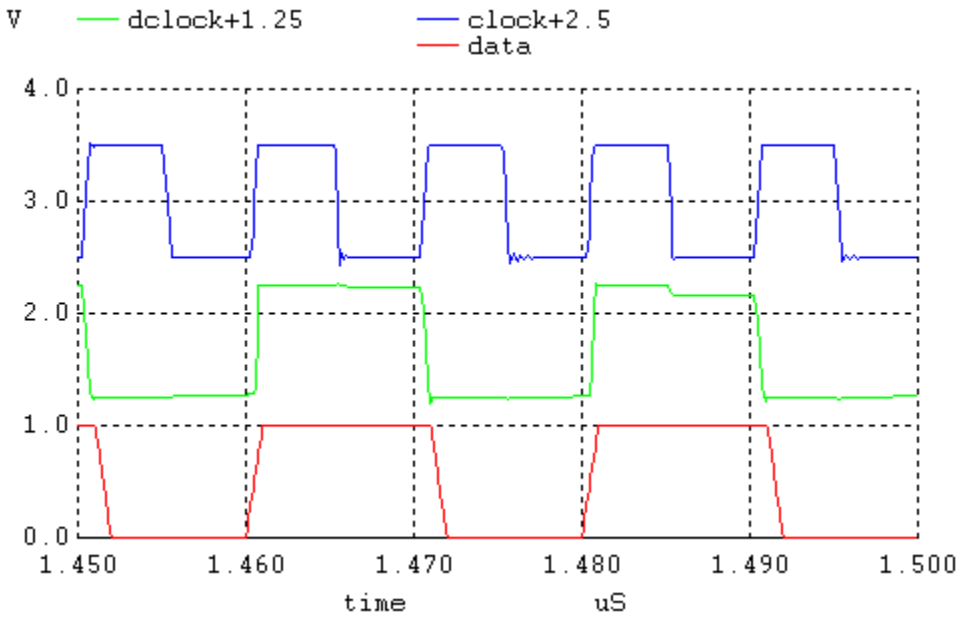
Now, when the pump currents are mismatched in fig 19.37, the change will affect the rate at which we add or remove the charge from the capacitors. But since the PFD we are using in this figure locks the loop on the basis of edge transitions, the rate at which we add or remove the charge should not affect the way we lock the signal. And hence, we will still transition on the rising edge of the clock without any error.

But, we if you use current starved phase detector, the change in the current pump will cause a static phase error. This is because of the fact that we are now NOT locking on the transition of the edges.

The goal of this example is to see if we see a static phase error when there is a mismatch in the two charge pump currents. To create a mismatch in the charge pump currents, we need to create an imbalance between the pull up current through the PMOS devices and the pull down currents through the NMOS devices in the charge pump. So, we can just change the widths of the mpup or the mndwn transistors and create a difference in the currents.

Solution to problem 19.15

To start with, let's simulate the fig 19.37 as it is and plot the graph:



Note: Iup = Idown for this figure is 10 uA. The loop is locked. Once the VCO is settled, the dclock and data transition on the rising edge of the clock.

Static Phase Error: When the loop is locked and if the dclock don't transition with the rising edge of the clock, than the delta between the rising edge of the clock and the rising edge of the dclock is called Static Phase Error.

Now, when the pump currents are mismatched in fig 19.37, the change will affect the rate at which we add or remove the charge from the capacitors. But since the PFD we are using in this figure locks the loop on the basis of edge transitions, the rate at which we add or remove the charge should not affect the way we lock the signal. And hence, we will still transition on the rising edge of the clock without any error.

But, we if you use current starved phase detector, the change in the current pump will cause a static phase error. This is because of the fact that we are now NOT locking on the transition of the edges.

The goal of this example is to see if we see a static phase error when there is a mismatch in the two charge pump currents. To create a mismatch in the charge pump currents, we need to create an imbalance between the pull up current through the PMOS devices and the pull down currents through the NMOS devices in the charge pump. So, we can just change the widths of the mpup or the mndwn transistors and create a difference in the currents.

For a 50 % imbalance between the Ipup and Idown, we can change the width of the transistors as shown in the table below:

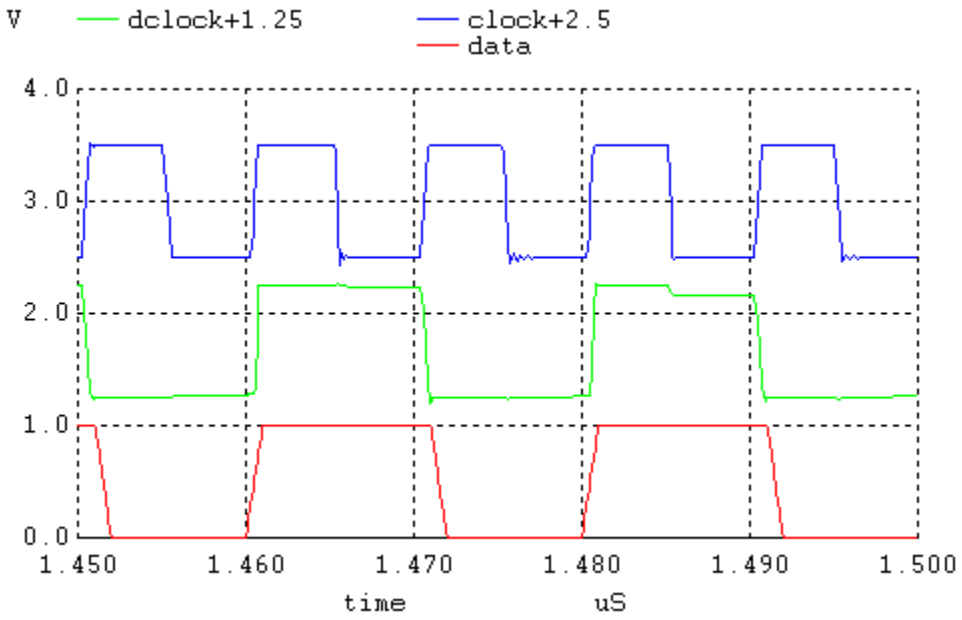
	Normal	**Different Instance of Current Mismatch**			
	1	2	3	4	5
Width of mpup	100	50	150	100	100
Width of mndwn	50	50	50	25	75
Resulting Iup	10uA	5uA	15uA	10uA	10uA
Resulting Idown	10uA	10uA	10uA	5uA	15uA

Note: Please look at the netlist code the exact location of the change (code is added at the end).

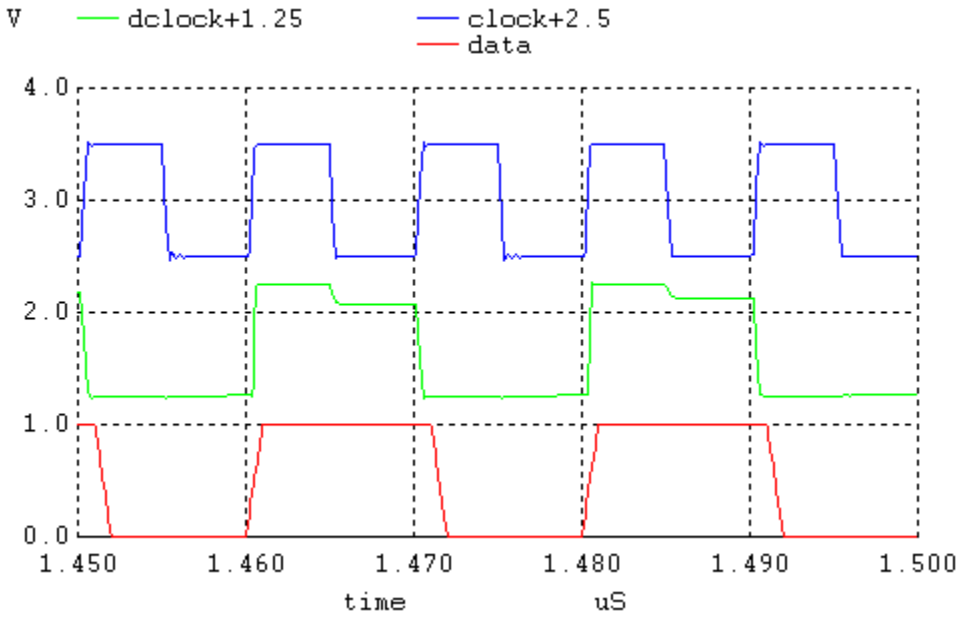
For all five instances, I have shown the resulting simulations below. After studying different simulations in details, we can conclude that there is no static phase error when the two charge pump currents are mismatched by 50 %.

Try One: Increase Pump Current

To begin, let's increase the mismatch by 50 % by increasing the widths of the transistors (mpup and mndwn) one at a time while keeping other constant and see if we find a static phase error.



Graph for Iup or Idown = 10 uA.

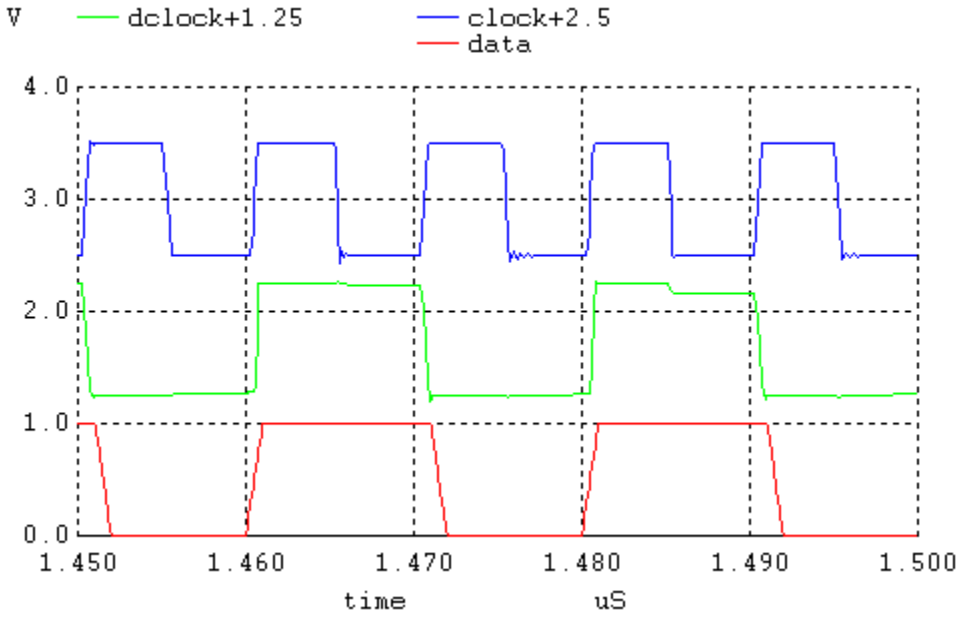


Graph for Iup or Idown = 15 uA.

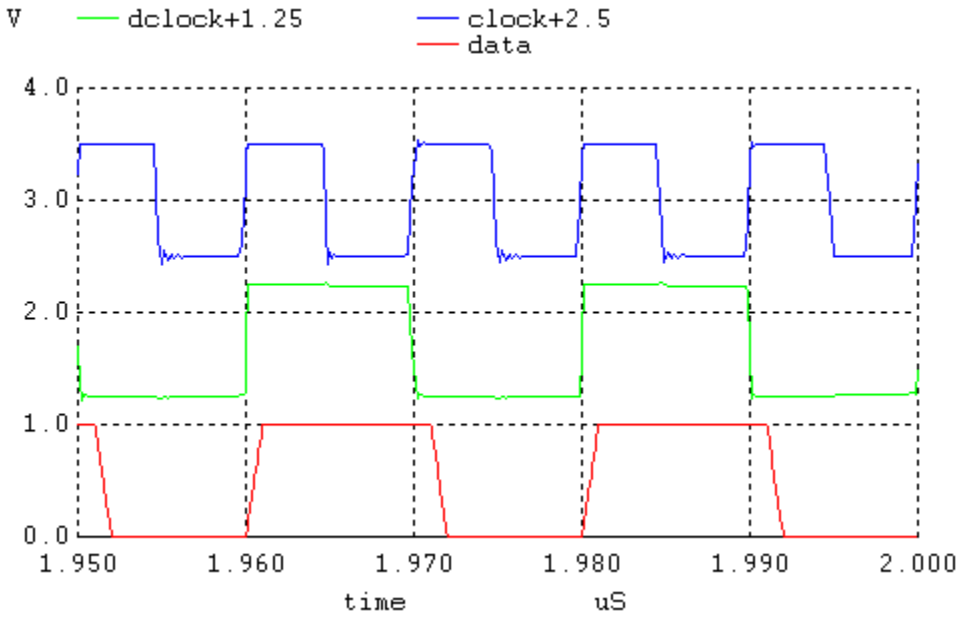
Comparing the two graphs, we can conclude that there is no static phase error when we increased the value of Iup or Idown to 15 uA.
Note: Please look at the end for the netlist details.

Try Two: Decrease pump current

On the second try, let's decrease the mismatch by 50 % by decreasing the widths of the transistors (mpup and mndwn) one at a time while keeping other constant and see if we find a static phase error.



Graph for Iup or Idown = 10 uA.



Graph for Iup or Idown = 5 uA.

Comparing the two graphs, we can conclude that there is no static phase error when we decreased the value of Iup or Idown to 5 uA.
Note: Please look at the end for the netlist details.

Conclusion:

So from the above figures, we see that by either increasing or decreasing the pump currents, the graphs are exactly the same as compared with the Iup = Idwn = 10uA (normal circuit). Hence, we can conclude that a mismatch of 50 % on the pump currents will not create a static phase error. And so the theory explained before still holds true.

Note: Please follow the next page for details on the net list.

Netlist: Code for this problem

```
.control
destroy all
run
plot vinvco
plot data dclock+1.25 clock+2.5 xlimit 1950n 2000n
.endc

.option scale=50n
.tran 1n 2000n UIC
.ic v(clock)=0

VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 20n

XPD     VDD     data    dclock  up      down    pfd
Xvco    VDD     Vinvco  clock   VCO
Xdiv    VDD     clock   dclock  dby2

Xinv1   VDD     up      upi     inverter
Xinv2   VDD     down    downi   inverter

M2L     vnx     up      Vpup    VDD     PMOS L=1 W=20
M2R     vinvco  upi     vpup    VDD     PMOS L=1 W=20
M1L     vnx     downi   vndwn   0       NMOS L=1 W=10
M1R     vinvco  down    vndwn   0       NMOS L=1 W=10

Mpb     Vp      Vp      VDD     VDD     PMOS L=2 W=100
Ibias   Vp      Vn      DC      5u
Mnb     Vn      Vn      0       0       NMOS L=2 W=50

********** Change widths of the transistors one at a time.
Mpup    Vpup    Vp      VDD     VDD     PMOS L=2 W=100
Mndwn   Vndwn   Vn      0       0       NMOS L=2 W=50

R1      vinvco  vrc     20k
C1      vrc     0       10p
C2      vinvco  0       1p

.subckt dby2    VDD     clock   dclock
M1      n1      dclock  VDD     VDD     PMOS L=1 W=20
M2      n2      clock   n1      VDD     PMOS L=1 W=20
M3      n2      dclock  0       0       NMOS L=1 W=10
M4      n3      clock   VDD     VDD     PMOS L=1 W=20
M5      n4      n2      n3      0       NMOS L=1 W=10
M6      n4      clock   0       0       NMOS L=1 W=10
M7      dclock  n3      VDD     VDD     PMOS L=1 W=20
M8      dclock  clock   n5      0       NMOS L=1 W=10
M9      n5      n3      0       0       NMOS L=1 W=10
.ends

.subckt VCO VDD Vinvco clock

M5      vn      Vn      0       0       NMOS L=1 W=10
M6      vn      vp      VDD     VDD     PMOS L=1 W=20
M5R     vp      Vinvco  Vr      0       NMOS L=1 W=100
```

```
Rset    Vr      0       10k
M6R     vp      vp      VDD     VDD     PMOS L=1 W=20

X1      VDD     Vn      Vp      out21   out1    VCOstage
X2      VDD     Vn      Vp      out1    out2    VCOstage
X3      VDD     Vn      Vp      out2    out3    VCOstage
X4      VDD     Vn      Vp      out3    out4    VCOstage
X5      VDD     Vn      Vp      out4    out5    VCOstage
X6      VDD     Vn      Vp      out5    out6    VCOstage
X7      VDD     Vn      Vp      out6    out7    VCOstage
X8      VDD     Vn      Vp      out7    out8    VCOstage
X9      VDD     Vn      Vp      out8    out9    VCOstage
X10     VDD     Vn      Vp      out9    out10   VCOstage
X11     VDD     Vn      Vp      out10   out11   VCOstage
X12     VDD     Vn      Vp      out11   out12   VCOstage
X13     VDD     Vn      Vp      out12   out13   VCOstage
X14     VDD     Vn      Vp      out13   out14   VCOstage
X15     VDD     Vn      Vp      out14   out15   VCOstage
X16     VDD     Vn      Vp      out15   out16   VCOstage
X17     VDD     Vn      Vp      out16   out17   VCOstage
X18     VDD     Vn      Vp      out17   out18   VCOstage
X19     VDD     Vn      Vp      out18   out19   VCOstage
X20     VDD     Vn      Vp      out19   out20   VCOstage
X21     VDD     Vn      Vp      out20   out21   VCOstage

X22     VDD     out21   clock   inverter
.ends


.subckt VCOstage        VDD     Vinvco  Vp      in      out
M1      vd1     Vinvco  0       0       NMOS    L=1     W=10
M2      out     in      Vd1     0       NMOS    L=1     W=10
M3      out     in      Vd4     VDD     PMOS    L=1     W=20
M4      Vd4     in      VDD     VDD     PMOS    L=1     W=20
.ends

.subckt pfd     VDD     data    dclock  up      down
X1      VDD     data    n1      inverter
X2      VDD     n1      n5      n2      nand
X3      VDD     n2      n3      inverter
X4      VDD     n3      n4      inverter
X5      VDD     n4      n6      reset   n5      nand3
X6      VDD     n5      up      inverter
X7      VDD     n2      n7      n6      nand
X8      VDD     n6      reset   n7      nand
X9      VDD     reset   n8      n9      nand
X10     VDD     n9      n11     n8      nand
X11     VDD     n6      n2      n11     n8      reset   nand4
X12     VDD     dclock  n10     inverter
X13     VDD     n10     n14     n11     nand
X14     VDD     n11     n12     inverter
X15     VDD     n12     n13     inverter
X16     VDD     reset   n8      n13     n14     nand3
X17     VDD     n14     down    inverter
.ends

.subckt nand    VDD     A       B       ANANDB
M1      n1      A       0       0       NMOS    L=1     W=10
M2      ANANDB          B       n1      0       NMOS    L=1     W=10
```

```
M3      ANANDB          A       VDD     VDD     PMOS    L=1     W=20
M4      ANANDB          B       VDD     VDD     PMOS    L=1     W=20
.ends

.subckt inverterVDD     in      out
M1      out     in      0       0       NMOS    L=1     W=10
M2      out     in      VDD     VDD     PMOS    L=1     W=20
.ends

.subckt nand3   VDD     A       B       C       OUT
M1      n1      A       0       0       NMOS    L=1     W=10
M2      n2      B       n1      0       NMOS    L=1     W=10
M3      OUT     C       n2      0       NMOS    L=1     W=10
M4      OUT     A       VDD     VDD     PMOS    L=1     W=20
M5      OUT     B       VDD     VDD     PMOS    L=1     W=20
M6      OUT     C       VDD     VDD     PMOS    L=1     W=20
.ends

.subckt nand4   VDD     A       B       C       D       OUT
M1      n1      A       0       0       NMOS    L=1     W=10
M2      n2      B       n1      0       NMOS    L=1     W=10
M3      n3      C       n2      0       NMOS    L=1     W=10
M4      OUT     D       n3      0       NMOS    L=1     W=10
M5      OUT     A       VDD     VDD     PMOS    L=1     W=20
M6      OUT     B       VDD     VDD     PMOS    L=1     W=20
M7      OUT     C       VDD     VDD     PMOS    L=1     W=20
M8      OUT     D       VDD     VDD     PMOS    L=1     W=20
.ends
```

P19_16

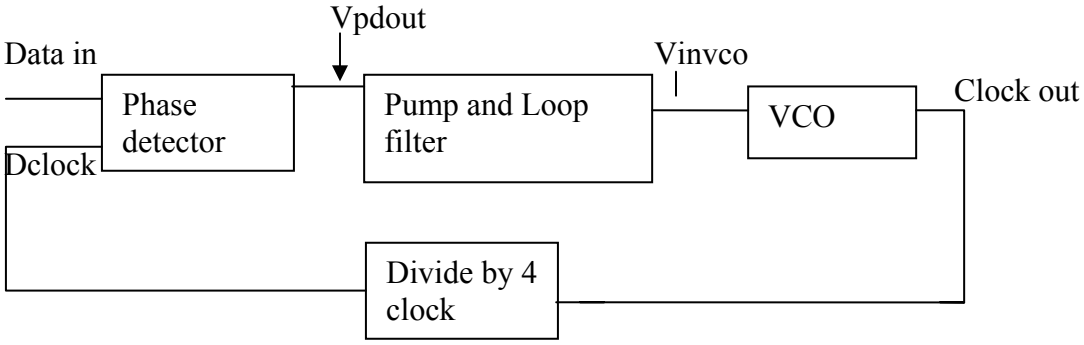
Pradip Ghimire

Problem definition

Redesign the DPLL in Ex. 19.5 so that the output remains a 100 MHz square wave signal when the input value is changed to 25MHz.

Design criteria:

The DPLL circuit consists of following block diagram to produce the 100 MHz wave signal from the input clock of 25MHz.



The gain was calculated as Kvco= 1.57x10^9 radian/V.s where as the gain of the phase detector is KPD = Ipump/4π .
The lock range for this design was set as 20MHz and the damping factor was set to be 1. With all the parameters, the final value for ΔW_l comes out to be:

$\Delta W_l = 4\pi wn$
$w_n = 4\pi/ \Delta W_l = 10 * 10^6$ radian/V.s

The R1C value was found by using $\zeta = wn/2 *R2C$
R2C= 200ns.
The capacitor value was set to be 10 pF and Resistor value was set to be 20K.
The C2 value was set to be one tenth of C1. The value for the Ipump was found by using equation, $Wn^2= Ipump* Kvco/(2\pi *2*C1)$
Ipump = 10uA

The main modification from the example was done in divider circuit because the Dclock has to be clock divide by 4 to achieve the 100 MHz output.
The simulation results show the by dividing the Dclock by N=4, we can achieve the desired results. The slight overshoot in the simulation is caused by the damping factor.

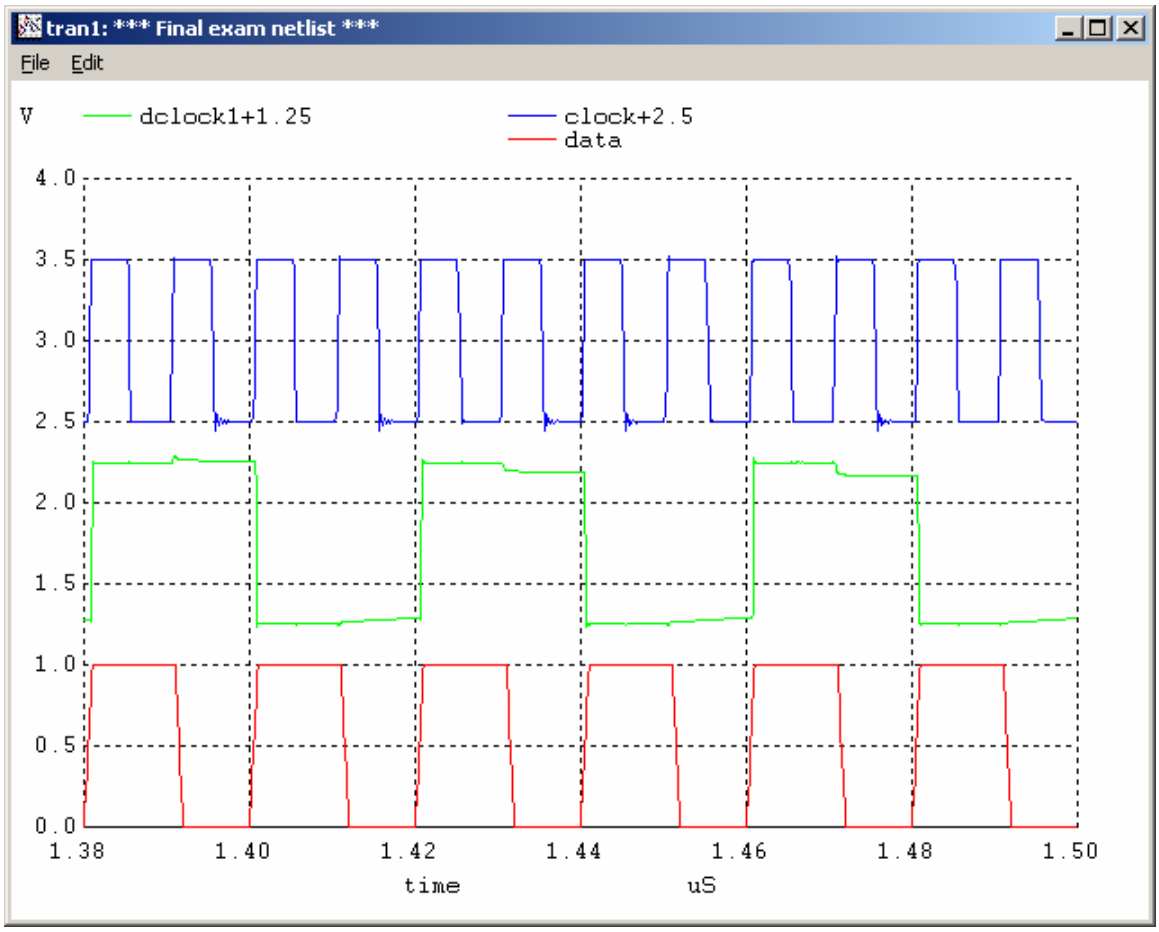


Fig1: Simulation results for output clock of 100MHz

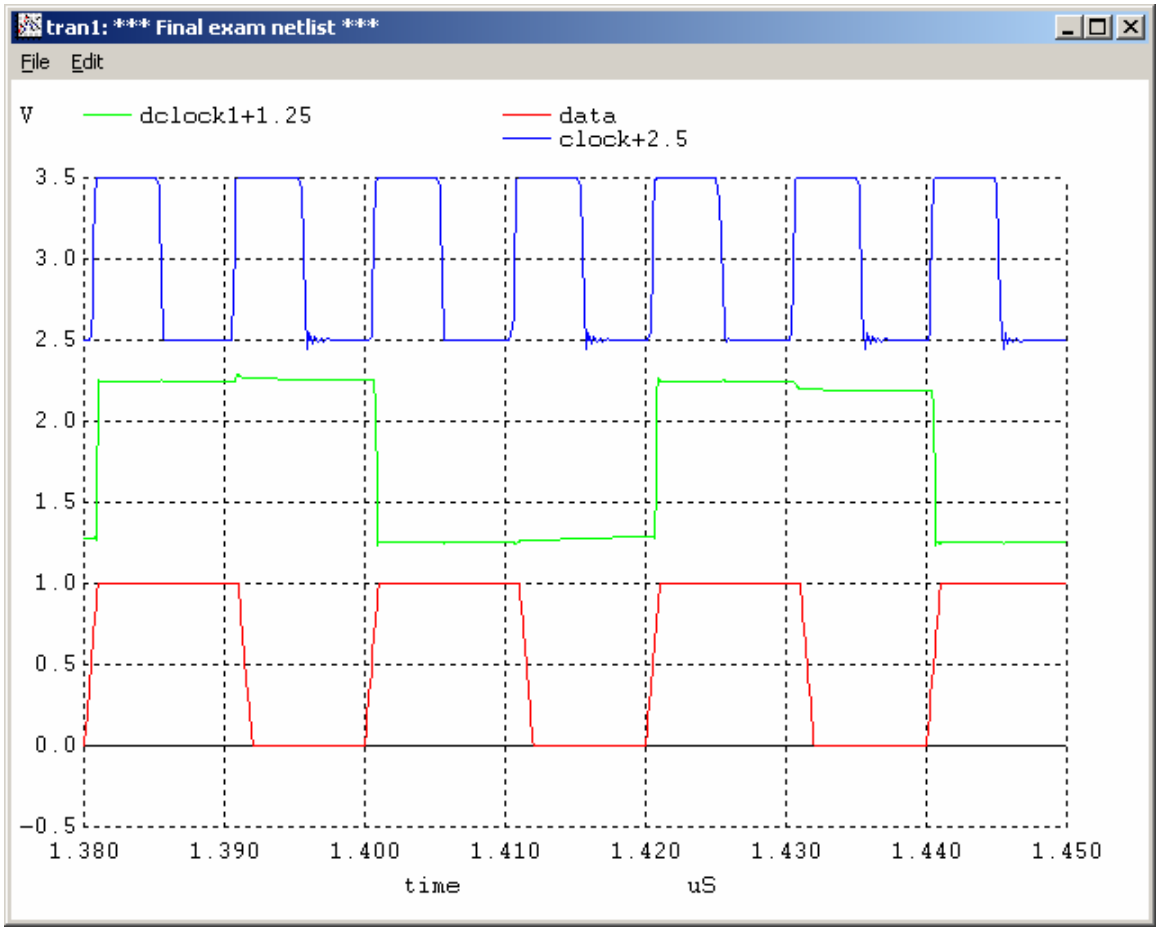


Fig2: close up view of fig1 simulation.

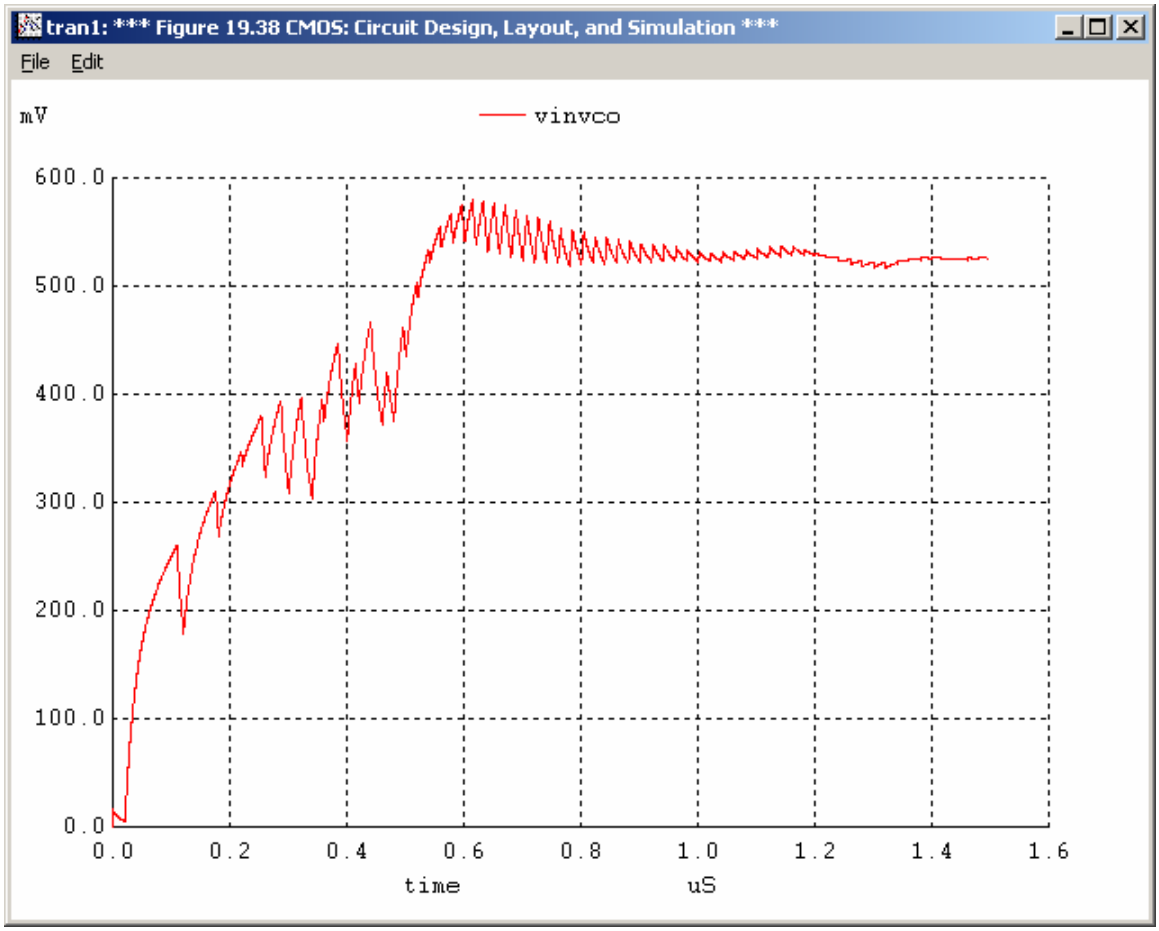


Fig 3 : VinVco

The netlist for this simulation as follows:

```
*** Final exam netlist ***

.control
destroy all
run
plot vinvco
plot data dclock1+1.25 clock+2.5 xlimit 1400n 1500n
.endc

.option scale=50n
.tran 1n 1500n UIC
.ic v(clock)=0


VDD     VDD     0       DC      1
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 20n
```

```
XPD     VDD     data    dclock  up      down    pfd
Xvco    VDD     Vinvco  clock   VCO
Xdiv    VDD     clock   dclock  dby4
Xdiv1   VDD     dclock  dclock1 dby4


Xinv1   VDD     up      upi     inverter
Xinv2   VDD     down    downi   inverter

M2L     vnx     up      Vpup    VDD     PMOS L=1 W=20
M2R     vinvco  upi     vpup    VDD     PMOS L=1 W=20
M1L     vnx     downi   vndwn   0       NMOS L=1 W=10
M1R     vinvco  down    vndwn   0       NMOS L=1 W=10

Mpb     Vp      Vp      VDD     VDD     PMOS L=2 W=100
Ibias   Vp      Vn      DC      10u
Mnb     Vn      Vn      0       0       NMOS L=2 W=50

Mpup    Vpup    Vp      VDD     VDD     PMOS L=2 W=100
Mndwn   Vndwn   Vn      0       0       NMOS L=2 W=50

R1      vinvco  vrc     20k
C1      vrc     0       10p
C2      vinvco  0       1p


.subckt dby4    VDD     clock   dclock
M1      n1      dclock  VDD     VDD     PMOS L=1 W=20
M2      n2      clock   n1      VDD     PMOS L=1 W=20
M3      n2      dclock  0       0       NMOS L=1 W=10
M4      n3      clock   VDD     VDD     PMOS L=1 W=20
M5      n4      n2      n3      0       NMOS L=1 W=10
M6      n4      clock   0       0       NMOS L=1 W=10
M7      dclock  n3      VDD     VDD     PMOS L=1 W=20
M8      dclock  clock   n5      0       NMOS L=1 W=10
M9      n5      n3      0       0       NMOS L=1 W=10
.ends

.subckt VCO VDD Vinvco clock

M5      vn      Vn      0       0       NMOS L=1 W=10
M6      vn      vp      VDD     VDD     PMOS L=1 W=20
M5R     vp      Vinvco  Vr      0       NMOS L=1 W=100
Rset    Vr      0       10k
M6R     vp      vp      VDD     VDD     PMOS L=1 W=20

X1      VDD     Vn      Vp      out21   out1    VCOstage
X2      VDD     Vn      Vp      out1    out2    VCOstage
X3      VDD     Vn      Vp      out2    out3    VCOstage
X4      VDD     Vn      Vp      out3    out4    VCOstage
X5      VDD     Vn      Vp      out4    out5    VCOstage
X6      VDD     Vn      Vp      out5    out6    VCOstage
X7      VDD     Vn      Vp      out6    out7    VCOstage
X8      VDD     Vn      Vp      out7    out8    VCOstage
X9      VDD     Vn      Vp      out8    out9    VCOstage
X10     VDD     Vn      Vp      out9    out10   VCOstage
X11     VDD     Vn      Vp      out10   out11   VCOstage
X12     VDD     Vn      Vp      out11   out12   VCOstage
X13     VDD     Vn      Vp      out12   out13   VCOstage
X14     VDD     Vn      Vp      out13   out14   VCOstage
X15     VDD     Vn      Vp      out14   out15   VCOstage
X16     VDD     Vn      Vp      out15   out16   VCOstage
X17     VDD     Vn      Vp      out16   out17   VCOstage
X18     VDD     Vn      Vp      out17   out18   VCOstage
X19     VDD     Vn      Vp      out18   out19   VCOstage
X20     VDD     Vn      Vp      out19   out20   VCOstage
X21     VDD     Vn      Vp      out20   out21   VCOstage

X22     VDD     out21   clock   inverter
```

```
.ends


.subckt     VCOstage VDD      Vn       Vp       in       out
M1          vd1      Vn       0        0        NMOS     L=1      W=10
M2          out      in       Vd1      0        NMOS     L=1      W=10
M3          out      in       Vd4      VDD      PMOS     L=1      W=20
M4          Vd4      vp       VDD      VDD      PMOS     L=1      W=20
.ends

.subckt     pfd      VDD      data     dclock   up       down
X1          VDD      data     n1       inverter
X2          VDD      n1       n5       n2       nand
X3          VDD      n2       n3       inverter
X4          VDD      n3       n4       inverter
X5          VDD      n4       n6       reset    n5       nand3
X6          VDD      n5       up       inverter
X7          VDD      n2       n7       n6       nand
X8          VDD      n6       reset    n7       nand
X9          VDD      reset    n8       n9       nand
X10         VDD      n9       n11      n8       nand
X11         VDD      n6       n2       n11      n8       reset    nand4
X12         VDD      dclock   n10      inverter
X13         VDD      n10      n14      n11      nand
X14         VDD      n11      n12      inverter
X15         VDD      n12      n13      inverter
X16         VDD      reset    n8       n13      n14      nand3
X17         VDD      n14      down     inverter
.ends

.subckt nand         VDD      A        B        ANANDB
M1          n1       A        0        0        NMOS     L=1      W=10
M2          ANANDB   B        n1       0        NMOS     L=1      W=10
M3          ANANDB   A        VDD      VDD      PMOS     L=1      W=20
M4          ANANDB   B        VDD      VDD      PMOS     L=1      W=20
.ends

.subckt inverter     VDD      in       out
M1          out      in       0        0        NMOS     L=1      W=10
M2          out      in       VDD      VDD      PMOS     L=1      W=20
.ends

.subckt nand3        VDD      A        B        C        OUT
M1          n1       A        0        0        NMOS     L=1      W=10
M2          n2       B        n1       0        NMOS     L=1      W=10
M3          OUT      C        n2       0        NMOS     L=1      W=10
M4          OUT      A        VDD      VDD      PMOS     L=1      W=20
M5          OUT      B        VDD      VDD      PMOS     L=1      W=20
M6          OUT      C        VDD      VDD      PMOS     L=1      W=20
.ends

.subckt nand4        VDD      A        B        C        D        OUT
M1          n1       A        0        0        NMOS     L=1      W=10
M2          n2       B        n1       0        NMOS     L=1      W=10
M3          n3       C        n2       0        NMOS     L=1      W=10
M4          OUT      D        n3       0        NMOS     L=1      W=10
M5          OUT      A        VDD      VDD      PMOS     L=1      W=20
M6          OUT      B        VDD      VDD      PMOS     L=1      W=20
M7          OUT      C        VDD      VDD      PMOS     L=1      W=20
M8          OUT      D        VDD      VDD      PMOS     L=1      W=20
.ends
```

PROBLEM 19.17:
(Submitted by T.VAMSHI KRISHNA)

In order to demonstrate the problems in using the small capacitors to implement loop filters consider the below circuit:

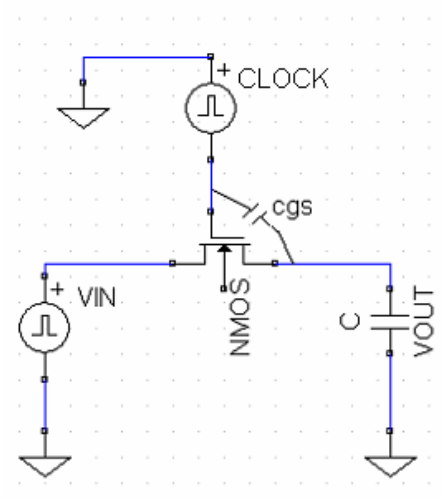


Fig 1

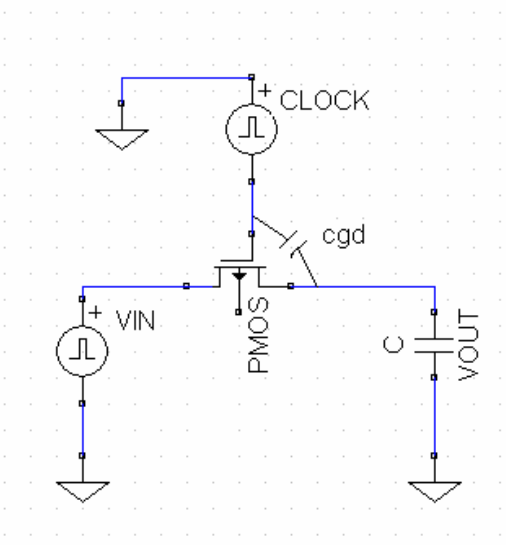


Fig 2.

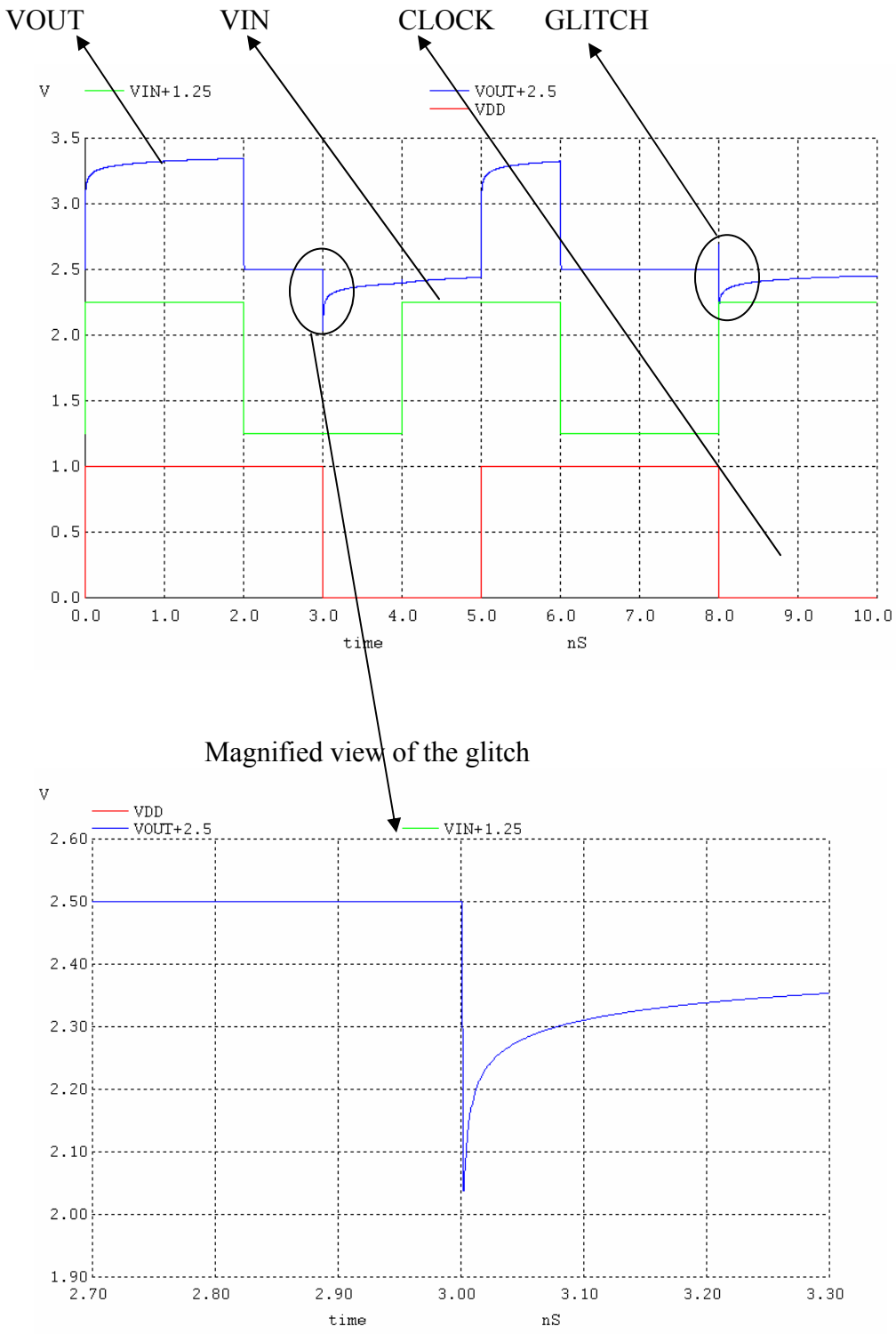
When the CLOCK is high the NMOS passes the signal VIN from input to output, but when CLOCK goes low clock feed through occurs through the gate to source capacitor and we will observe a glitch at the output if the output capacitor is small i.e in the order of femto farads.

Similarly for PMOS too when CLOCK is low the signal VIN from the input is passed to the output capacitor. But when CLOCK goes high clock feed through occurs if output capacitor is small. If the output capacitor is large then this effect of clock feed through is not seen.

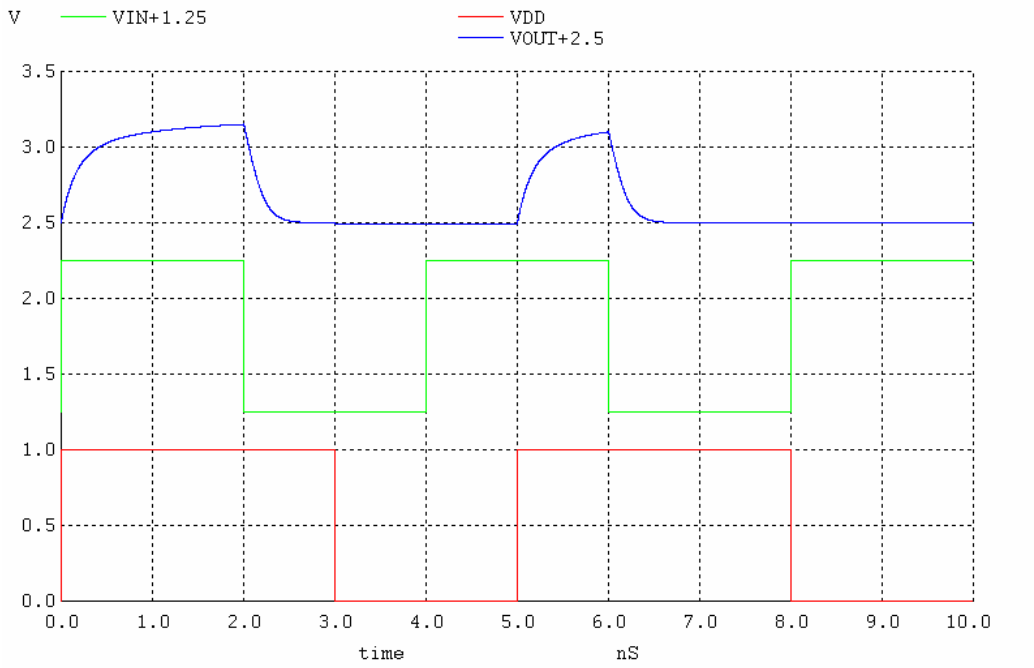
Simulating the above circuit with small and large output capacitors we can see the clock feed through if the output capacitor is small.

The simulation results are shown in the next page.

For fig 1. (Simulated with small output capacitor):

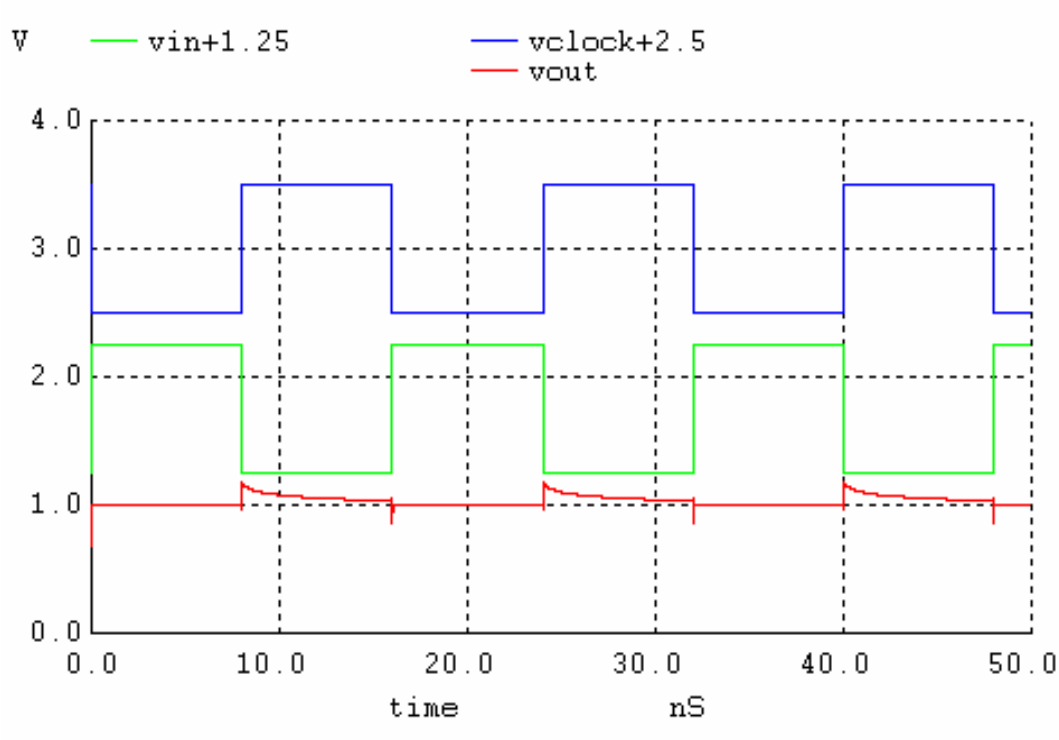


When a large capacitor is used we don't observe any glitch, the corresponding simulation is shown below.

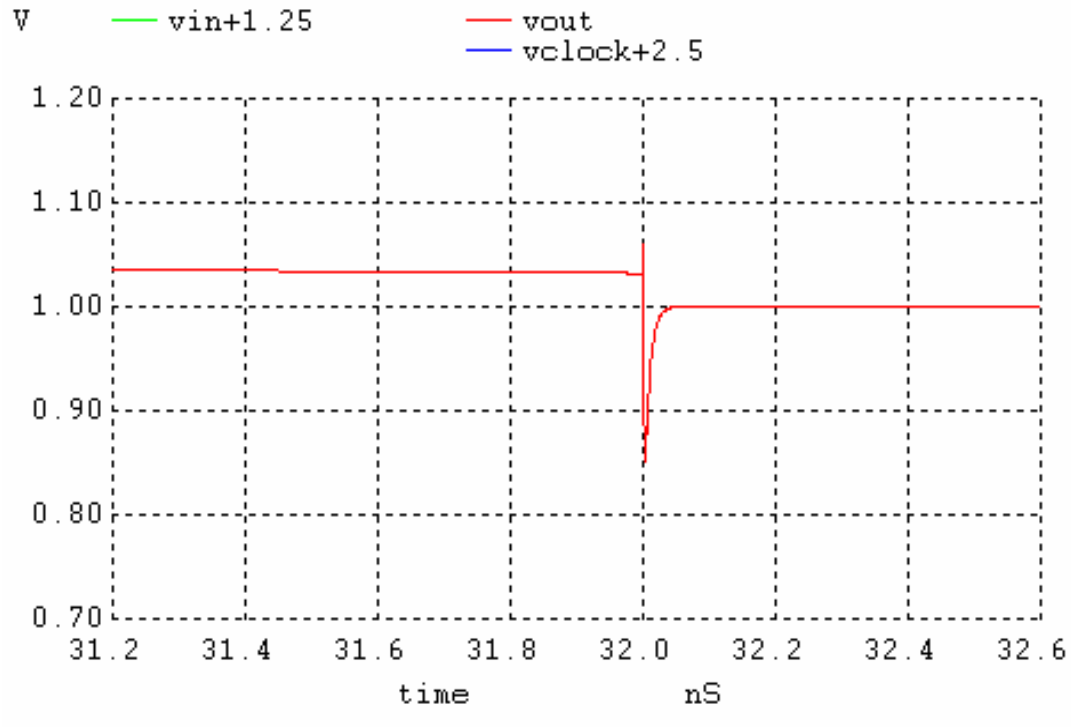


Similar argument can be followed for a PMOS.

The simulation results for a PMOS with a low value capacitor connected at the output is given below:



Magnified view of the glitch:



Now coming to the actual PLL circuit depending up on the UP and DOWN signals we steer the current in or out of the loop filter capacitor. The NMOS and PMOS capacitors in

the charge pump circuit pump or remove the charge from the capacitor. Hence if a small loop capacitor is used clock feed through occurs and we will observe glitches in VinVCO.

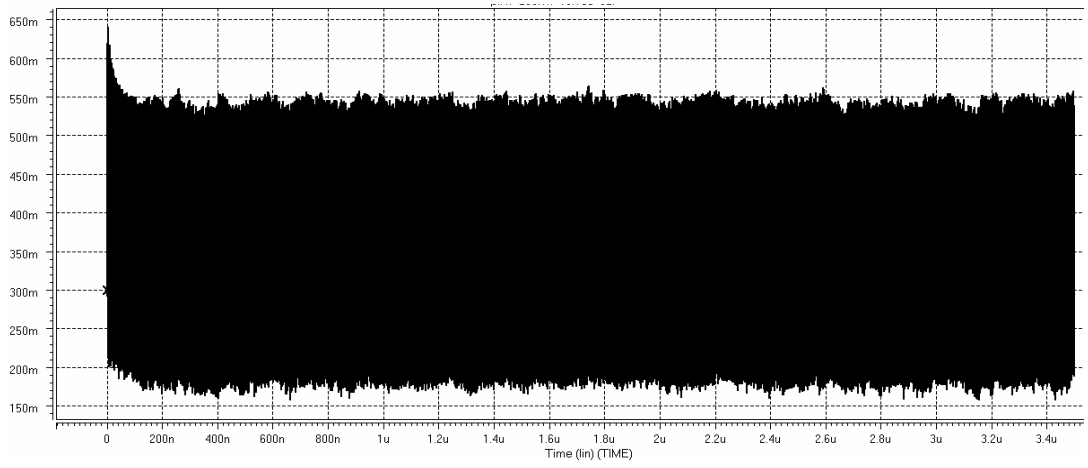
In order to demonstrate what happens if we use small loop capacitor we simulate a 500MHz clock recovery circuit and see the VinVCO. In the circuit I employed a low gain VCO and the current to be 10μA.

Furthur we know that

$$\omega_n^2 C = K_{VCO} K_{PD}$$

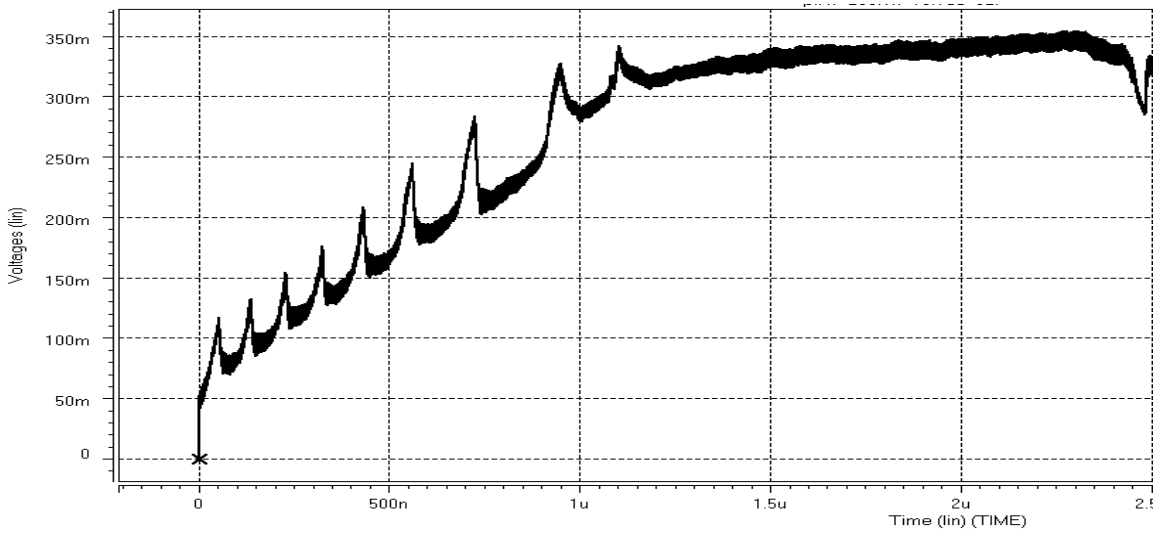
Based of the equation, for a given ω_n of 10^8 radians/sec if we determine the value of capacitor we get would be 120fF. If we simulate the DPLL with the above values we can observe the clock feed through.

The simulation results are shown below:



With the low value capacitor employed for loop filter we can see that the variation in VinVCO is varying by 400mV.

So inorder to choose a big capacitor we need to size out current source accordingly. The simulation results when current source is sized and a big capacitor is employed in the circuit is shown below



Hence if a big capacitor is chosen for loop filter the VinVCO we avoid the problem of clock feed through. Hence we need to adjust I_{pump} so as to size our capacitor accordingly. The disadvantage would be the layout area.

WINSPICE NETLIST for simulating clock feed through

```
. control
destroy all
run
PLOT vout vin+1.25 vclock+2.5
.endc

.OPTIONS SCALE=50N
.tran 1p 50n

vdd       vdd       0         dc        1
Vclock    Vclock    0         DC        0 pulse(1 0 0 0 0 8n 16n)
vin       vin       0         dc        0 pulse(0 1 0 0 0 8n 16n)
m1        vout      vclock    vin       vdd       pmos l=1 w=100
cout      vout      0         1f


SPICE MODELS

.end
```

Prob. 19.18 -
What are we sacrificing by using an equalizer? What does the minus sign indicate in the slope of the phase in fig.19.41?

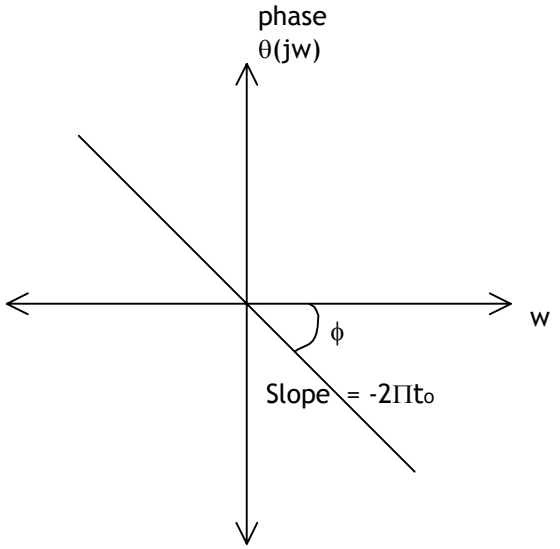
Solution –

The main drawback of using an Equalizer circuit to enable the system response to have minimal distortion, is loss of sensitivity. We sacrifice sensitivity by using the Equalizer circuit. The fact that the Equalizer circuit ends up attenuating the signal causes the desired signal level to go **down**.

The phase response of a distortion-free system has a slope of $-2\Pi t_o$. The phase response of the system is indicative of how good the output signal tracks the input. The value signifies that the output data tracks the input, over all frequencies of interest. (The phase angle response is the tangent of the [|X|/R], where |X| is the imaginary part of impedance, and R represents the real part.) But. the slope of the phase response being $-2\Pi t_o$, it implies that

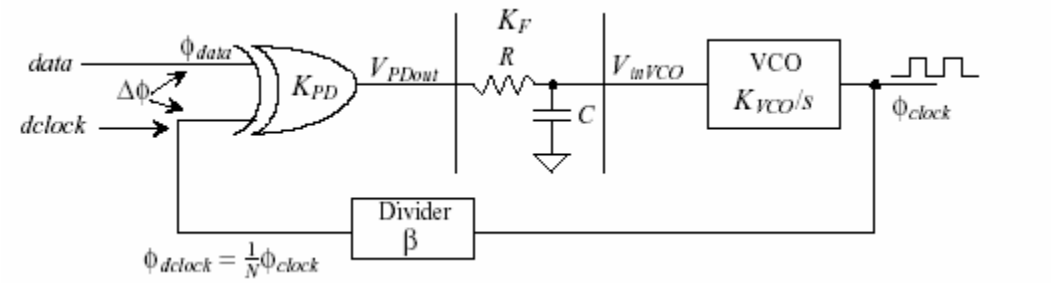
Slope = $\phi/w = -2\Pi t_o$,

Hence, $\phi = -2\Pi$ = -360 degrees, i.e., the output is lagging (negative 360 deg.) the input signal by one cycle. (The minus sign simply indicates the output of the system occurs after its input).

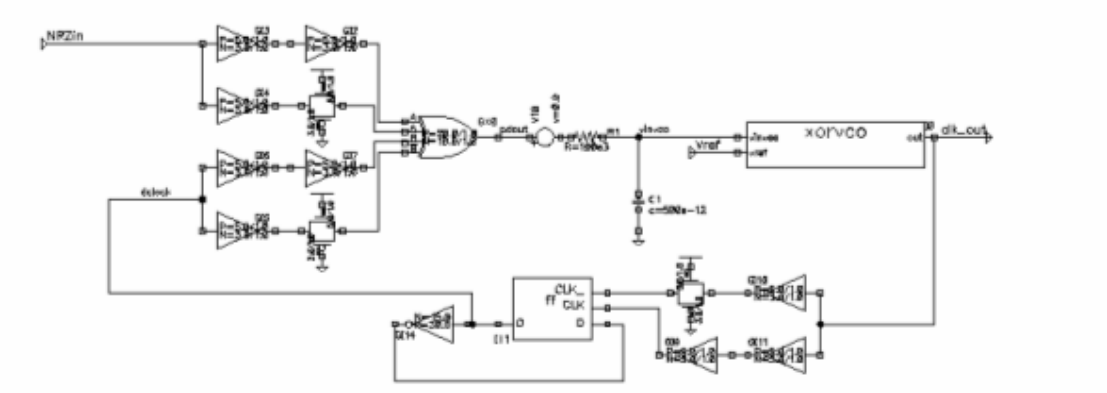


19.19. Using the DPLL from Ex.19.2 demonstrate the false locking as seen in Fig.19.46

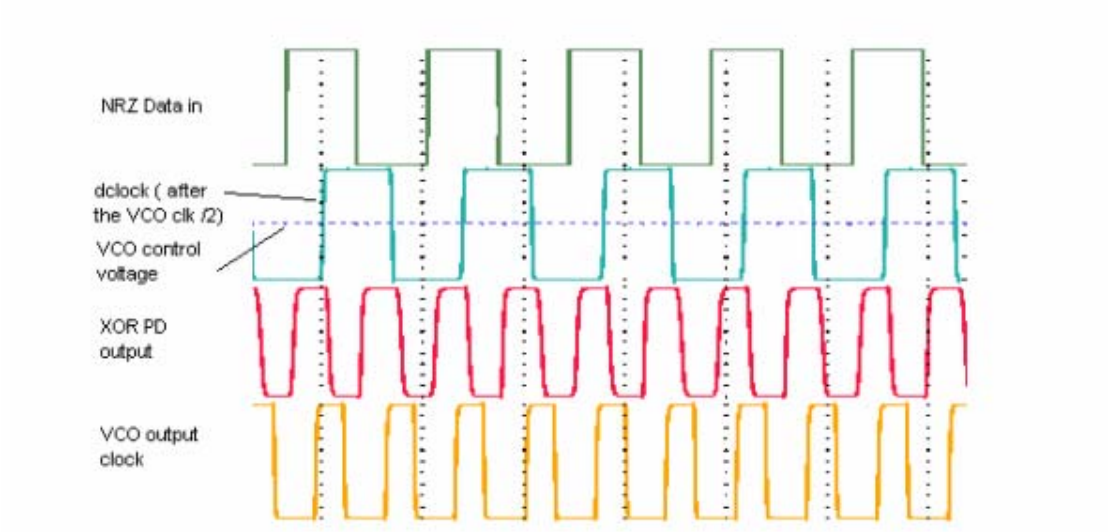
The diagram of the DPLL is shown in below. We pick N=2 for our circuit.



Real circuit:



Normal locked situation:



As shown in the above picture, the DLL is locked. Dclock is divided by 2 from the VCO output clock (CLKOUT). We got same pulse width of the "1" and "0" from the output of XOR phase detector. Thus, the average output voltage of the phase detector remains VDD/2 and the PLL is locked. We can use the rising edge of the CLKOUT signal to latch the correct data.

False locked situation:

The following simulation result shows the false locked situation. If the data input comes like 00110011 strings, we may still get the correct output frequency for the CLKOUT and its divided-by-2 signal dclock. However, as shown below, even the data input and dclock signals are not lined up correctly as the previous waveform, the average voltage of the PD output signal is still VDD/2. Thus, the VCO control voltage will keeps the same and we get a stable phase error. Because of this phase error, we may end up to latch a wrong data by CLKOUT.

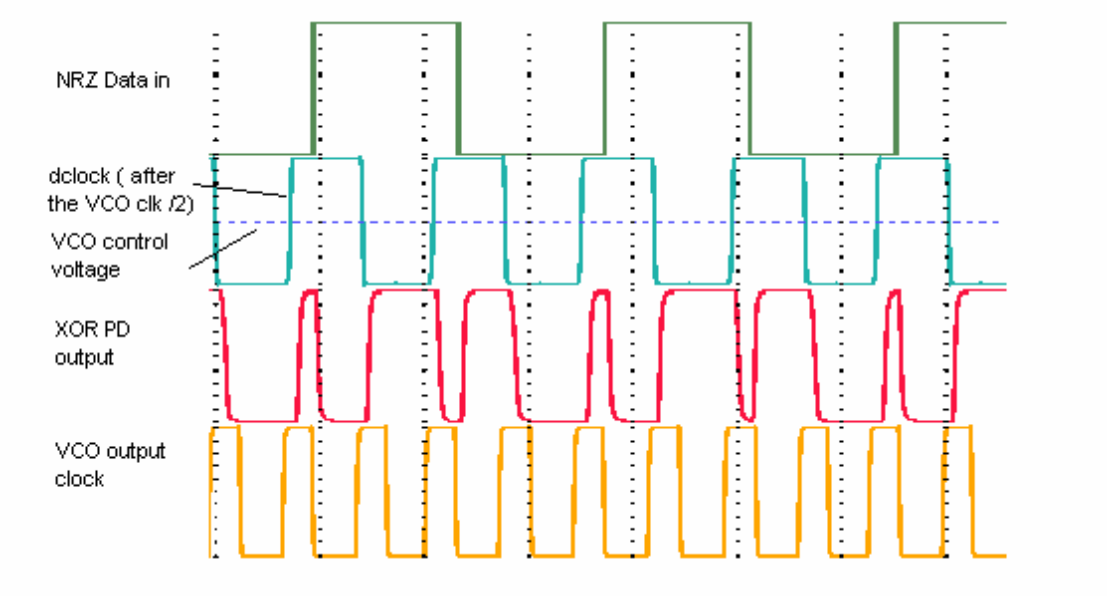


Fig.19.46 shows the similar problem. We may fix this problem if we add 010 in the input data strings.

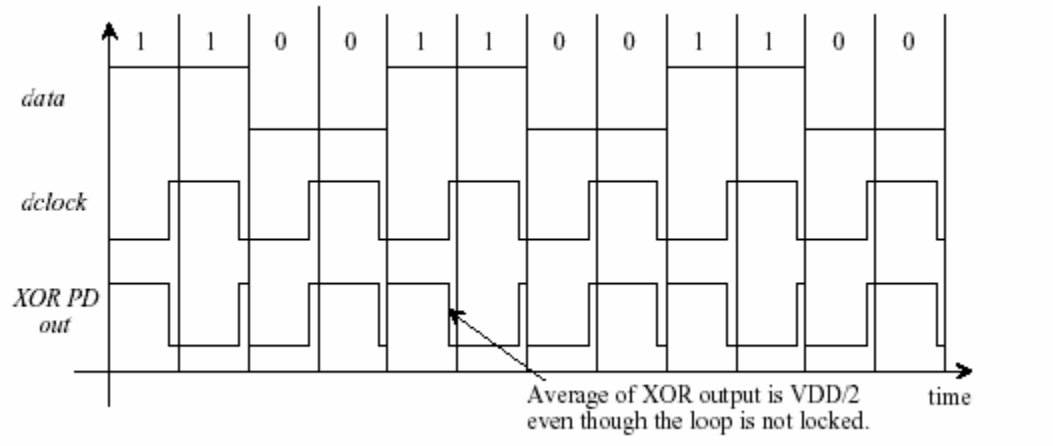


Fig.19.46.

YANTAO MA

yantaoma@rocketmail.com

Prob 19-20: design an edge detector, like one in Fig19.47 for use in the DPLL in Ex19.2. Regenerate the simulation data seen in Figs. 19.27 and 19.28.

[Ans]:

Edge detector design can be similar like the one in figure 19.47. However, based on input data rate=100MHz, pure inverter delay gate design would require large number of gates to generate appreciable width of edge detector pulse for XOR PD (in Ex19.2).

Edge Detector is designed as the following circuit Fig 19-20-1 and Fig 19-20-2. The characteristics are shown in the Fig 19-20-3.

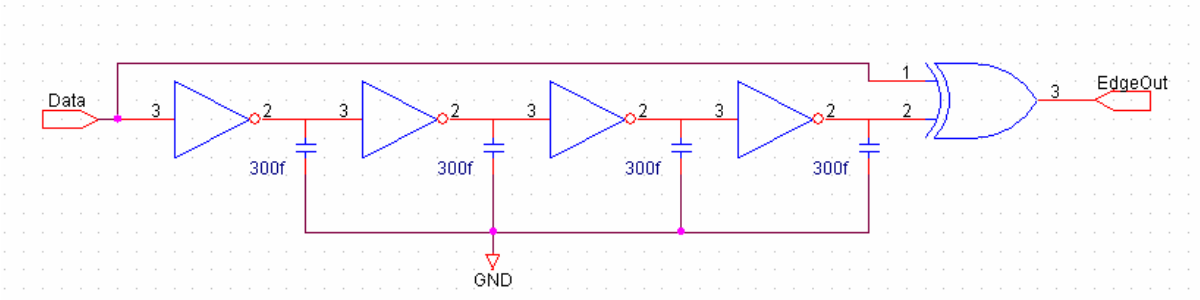


Figure 19-20-1: Edge Detector circuit design

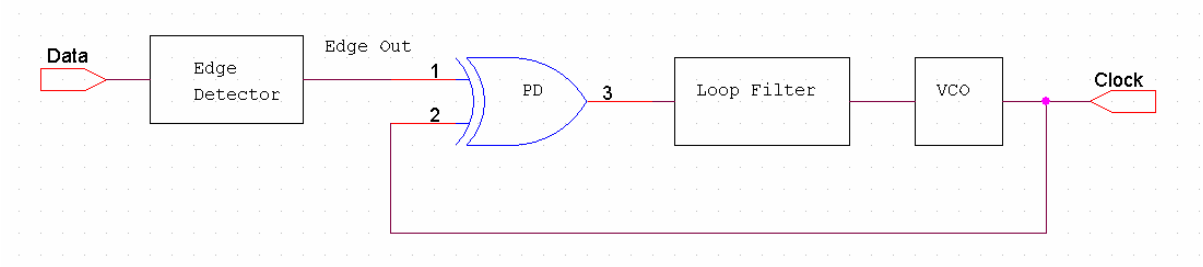


Figure 19-20-2: Clock Recovery circuit for NRZ using Edge Detector circuit.

Design Notes:

(1). Ideally, Vinvco should be a DC level under lock-in status. Practically, it will swing/oscillate around a DC level causing clock jitter.

(2). Clock recovered shall ideally have 50% duty cycle. Fine-tune on loop filter R, C values and VCO bias design is required. XOR PD and above circuit is not sensitive to 50% duty cycle input NRZ data stream apparently.

(3). Edge detector design parameters and delay stage numbers are heavily dependent on desired locking frequency and sensitivity of Phase Detector.

(4). Using Ex19.2 RC loop filter design parameters Rf=1k, Cf=2.5pf, simulations w/ edge detector for alternative data stream. Circuit will NOT lock in the middle of data stream due to the large oscillations on Vinvco. Refers to Fig 19-20-4. Intuitively, Cf is too small. Pick 7pF.

(5). Simulation to regenerate Figure 19.27 and Fig 19.28 are shown as Fig 19-20-5 and Fig 19-20-6: Rf =1k Cf=7pF Cdelay=300fF

(6). It is possible for designed DPLL locking on the wrong frequency(ies) when adjacent frequency space/difference less than system jitter. In the case of locking in wrong frequency, Loop filter design parameter should be reviewed before adjusting VCO gain. However, due to relatively low gain of VCO designed, it is very likely to stay unlocking (oscillating around desired 100MHz) after edge detector introduced. By simulation, VCO control voltage varies by 50mV, which leads to less than 1.25MHz variation. Jitter will be about 125ps

(LEFT) Pure inverter gate delay element
Each inverter Tdelay = 36.2ps/stage delay=0.89ns/stage

(RIGHT) Edge detector designed delay

Each stage (Cdelay=300f)

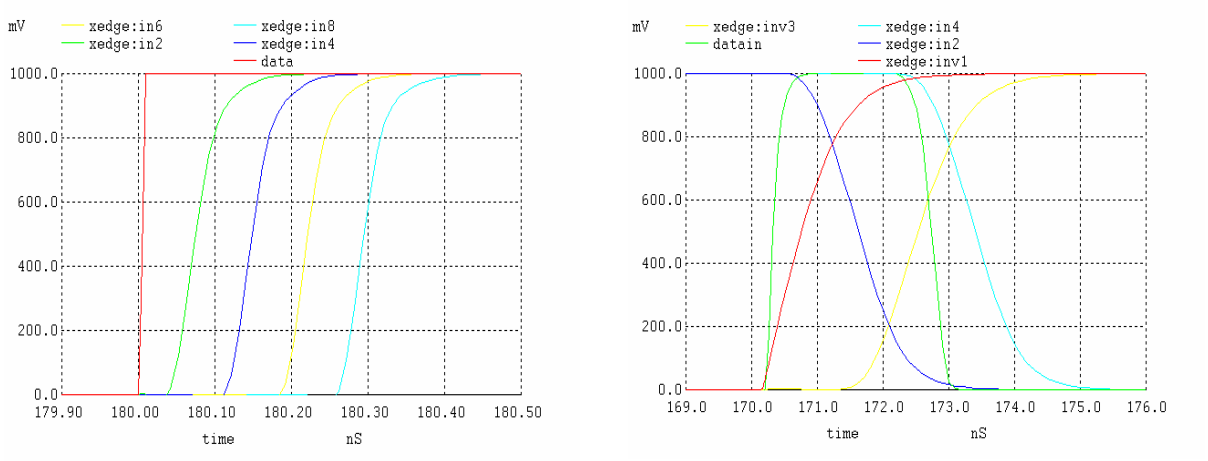


Fig 19-20-3 Delay element characteristics

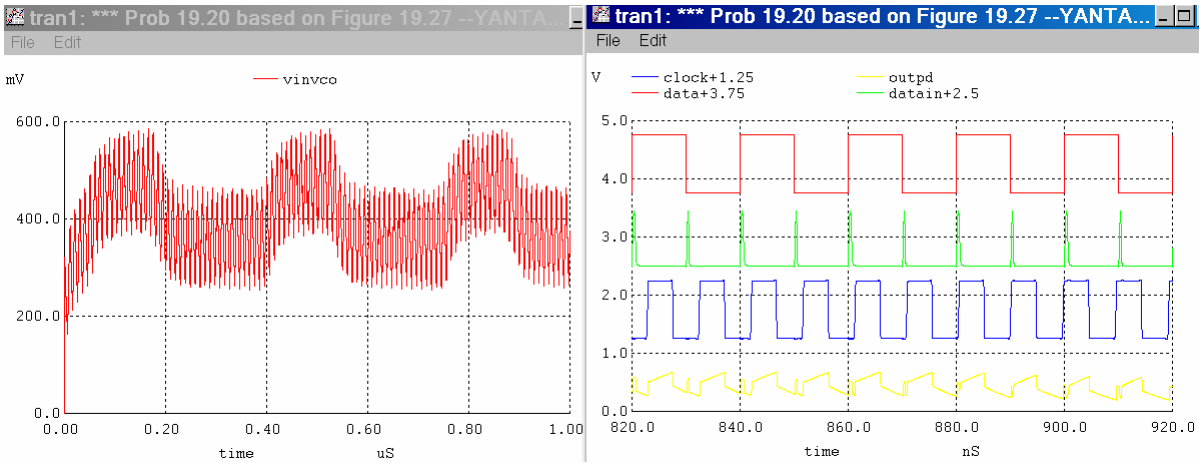


Fig 19-20-4 Simulation using Ex19.2 parameters (Rf=1k, Cf=2.5pf)

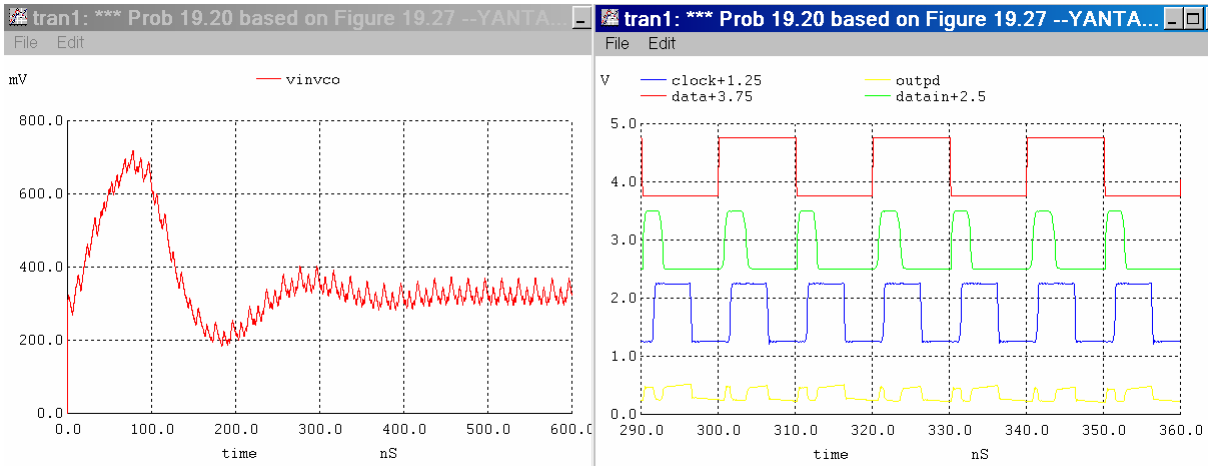


Figure 19-20-5: Regenerate DPLL Simulation for Figure 19.27 and figure 19.48
(Simulation condition: Rf =1k Cf=7pF Cdelay=300fF)

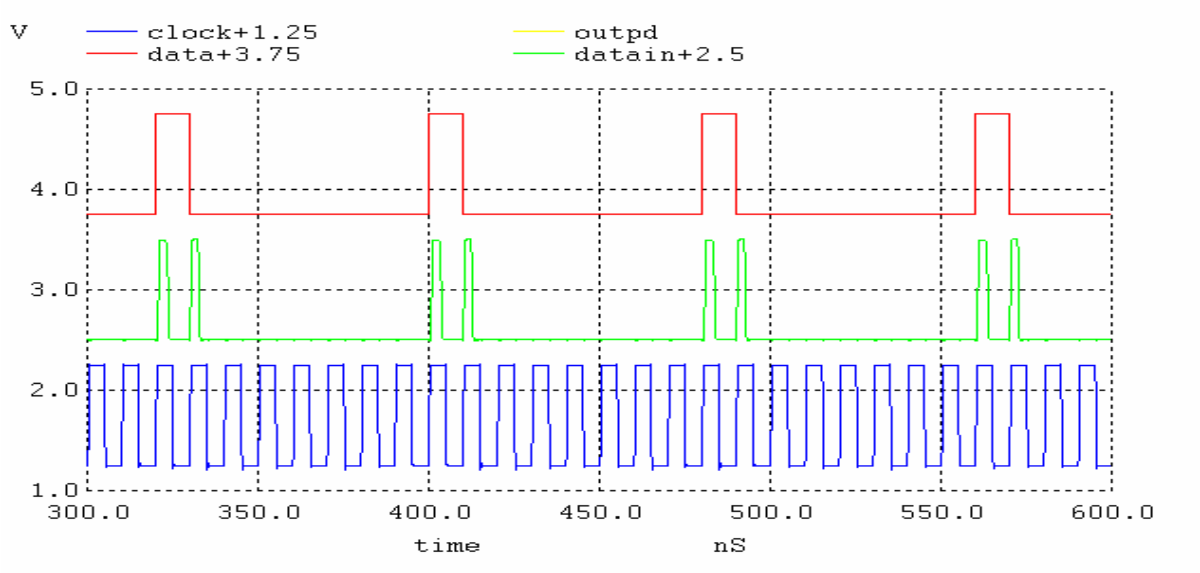


Figure 19-20-6: Regenerate DPLL Simulation for Figure 19.28

*** Prob: 19-20 CMOS: Circuit Design, Layout, and Simulation ****** Prob 19.20 based on Figure 19.27 --YANTAO MA
*** CMOS: Circuit Design, Layout, and Simulation ***

```
.control
destroy all
run
plot data+3.75 datain+2.5 clock+1.25 outpd
plot vinvco
*xlimit 450n 500n
.endc

.option scale=50n
.tran 10p 600n UIC
.ic v(clock)=0.3


VDD     VDD     0       DC      1
*vdata  data    0       DC      0       pulse 0 1 0 0 0 10n 20n
vdata   data    0       DC      0       pulse 0 1 0 0 0 10n 80n

XEDGE   VDD     data    datain  EDGED
XPD     VDD     datain  clock   outpd   XORPD

Xvco    VDD     Vinvco  clock   VCO

Rf      outpd   Vinvco  1k
Cf      Vinvco  0       7p      ic=0.3

.subckt EDGED   VDD     in      outpd
X1      VDD     in      inv1    inverter
C1      inv1    0       300f
X2      VDD     inv1    in2     inverter
C2      in2     0       300f
X3      VDD     in2     inv3    inverter
C3      inv3    0       300f
X4      VDD     inv3    in4     inverter
C4      in4     0       300f
;X5     VDD     in4     inv5    inverter
;X6     VDD     inv5    in6     inverter
;X7     VDD     in6     inv7    inverter
;X8     VDD     inv7    in8     inverter
;X9     VDD     in8     inv9    inverter
;X10    VDD     inv9    in10    inverter
;X11    VDD     in10    inv11   inverter
;X12    VDD     inv11   in12    inverter
```

```
XORIN   VDD     in      in4     inv1    inv3    outpd   xor
.ends


.subckt XORPD   VDD     A       B       outpd
M1      Ai      A       VDD     VDD     PMOS L=1 W=20
M2      Ai      A       0       0       NMOS L=1 W=10
M3      Bi      B       VDD     VDD     PMOS L=1 W=20
M4      Bi      B       0       0       NMOS L=1 W=10

M5      n1      A       VDD     VDD     PMOS L=1 W=20
M6      outpd   Ai      n1      VDD     PMOS L=1 W=20
M7      n1      B       VDD     VDD     PMOS L=1 W=20
M8      outpd   Bi      n1      VDD     PMOS L=1 W=20

M9      outpd   A       n2      0       NMOS L=1 W=10
M10     n2      B       0       0       NMOS L=1 W=10
M11     outpd   Ai      n3      0       NMOS L=1 W=10
M12     n3      Bi      0       0       NMOS L=1 W=10
.ends


.subckt VCO VDD Vinvco clock

M5      vn      Vn      0       0       NMOS L=1 W=10
M6      vn      vp      VDD     VDD     PMOS L=1 W=20
Rlow    vp      0       30k
M5R     vp      Vinvco  Vr      0       NMOS L=1 W=100
Rrange  Vr      0       100k
M6R     vp      vp      VDD     VDD     PMOS L=1 W=20

X1      VDD     Vn      Vp      out21   out1    VCOstage
X2      VDD     Vn      Vp      out1    out2    VCOstage
X3      VDD     Vn      Vp      out2    out3    VCOstage
X4      VDD     Vn      Vp      out3    out4    VCOstage
X5      VDD     Vn      Vp      out4    out5    VCOstage
X6      VDD     Vn      Vp      out5    out6    VCOstage
X7      VDD     Vn      Vp      out6    out7    VCOstage
X8      VDD     Vn      Vp      out7    out8    VCOstage
X9      VDD     Vn      Vp      out8    out9    VCOstage
X10     VDD     Vn      Vp      out9    out10   VCOstage
X11     VDD     Vn      Vp      out10   out11   VCOstage
X12     VDD     Vn      Vp      out11   out12   VCOstage
X13     VDD     Vn      Vp      out12   out13   VCOstage
X14     VDD     Vn      Vp      out13   out14   VCOstage
X15     VDD     Vn      Vp      out14   out15   VCOstage
X16     VDD     Vn      Vp      out15   out16   VCOstage
X17     VDD     Vn      Vp      out16   out17   VCOstage
X18     VDD     Vn      Vp      out17   out18   VCOstage
X19     VDD     Vn      Vp      out18   out19   VCOstage
X20     VDD     Vn      Vp      out19   out20   VCOstage
X21     VDD     Vn      Vp      out20   out21   VCOstage

X22     VDD     out21   clock   inverter
.ends

.subckt xorVDD  A       B       ANOT    BNOT    axorb
M1xb    n6b     B       VDD     VDD     PMOS L=1 W=20
M2xb    axorb   BNOT    n6b     VDD     PMOS L=1 W=20
M3xb    axorb   B       n7b     0       NMOS L=1 W=10
M4xb    n7b     A       0       0       NMOS L=1 W=10
M5xb    n6b     A       VDD     VDD     PMOS L=1 W=20
M6xb    axorb   ANOT    n6b     VDD     PMOS L=1 W=20
M7xb    axorb   BNOT    n8b     0       NMOS L=1 W=10
M8xb    n8b     ANOT    0       0       NMOS L=1 W=10
.ends

.subckt VCOstage VDD    Vinvco  Vp      in      out
M1      vd1     Vinvco  0       0       NMOS    L=1     W=10
M2      out     in      Vd1     0       NMOS    L=1     W=10
```

```
M3      out     in      Vd4     VDD     PMOS    L=1     W=20
M4      Vd4     in      VDD     VDD     PMOS    L=1     W=20
.ends

.subckt inverter        VDD     in      out
M1      out     in      0       0       NMOS    L=1     W=10
M2      out     in      VDD     VDD     PMOS    L=1     W=20
.ends

* BSIM4 models
*
```

Kevin Berkenmeier
EE597
Dec. 16, 2003

Derive Equation 19.21:

Tr = 2.2 ● (C1 Tclock/Kv ● 2Ipump) = № of clock cycles × Tclk

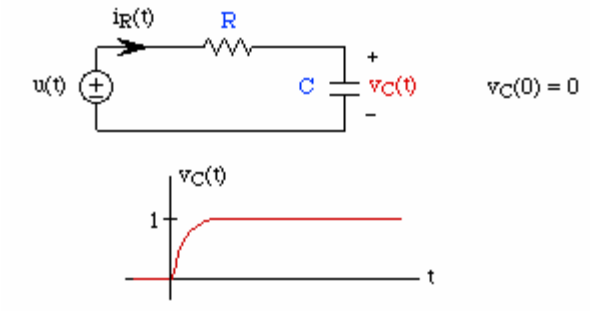
Tr equals the time it takes the DLL to respond to an input step in phase.

A benefit of the DLL is that the loop filter can be modeled as a first order feedback loop…Kf = 1/sC1.

The derivation of Eq. 19.71 will begin using the step response of the first order RC circuit and then replacing with the DLL response equations.

Using the first order increasing exponential formula:

v = Vf(1-e^¯t/RC)
v/Vf = 1 - e^¯t/RC
1 – v/Vf = e^¯t/RC
ln(1 – v/Vf) = ¯t/RC
t = -RC ln(1 – v/Vf)



Using the 90% point for t2 and the 10% point for t1 and Vf =1 (amplitude = 1), then

$t2 = -RC \ln(1-.9)$
$t2 = 2.3RC$

$t1 = -RC \ln(1-.1)$
$t1 = .1RC$

$Tr = t2 – t1 = 2.3RC - .1Rc = 2.2RC$

We know that the frequency of the reference clock must be exactly related to the frequency of the input data, but there will be instantaneous changes in the phase of the input data in which the output of the DLL must follow.

Also modeling the instantaneous changes in Φin by ΔΦin/s (a step function with the amplitude of ΔΦin), we get a change in the output phase given by Eq. 19.70

$\Delta\Phi out = \Delta\Phi in/(s + Kv \bullet 2Ipump/C1Tclock)$

In using the RC first order response equation and equating Tr = t2 – t1, the time it takes the DLL to respond to an input step in phase is simply

$Tr = 2.2 \bullet (C1\ Tclock/Kv \bullet 2Ipump) =$ № of clock cycles × Tclk

Prob19_22:- Design a Nominal delay line of 1ns (vindel=500mV) using the current starved delay element.

Soln:

I=kn'/2{(Vgs-Vt)^2}

For I=15u , Wn=5 Ln=1 Wp=10 Lp=10 at vindel =0.5V

I=15u

I=Cdv/dt

If switching point is Vsp

T1=VspC/I

Next Invertor switches when its input goes from VDD to Vsp

T2=(VDD-Vsp).C/I

T1+T2=C.VDD/I

Device sizes are Wn=5 Ln=1 Wp=10 Lp=1

N=I/fosc.C.VDD

C=5/2(WpLp+WnLn)*Cox'*(0.05um)*(0.05um)

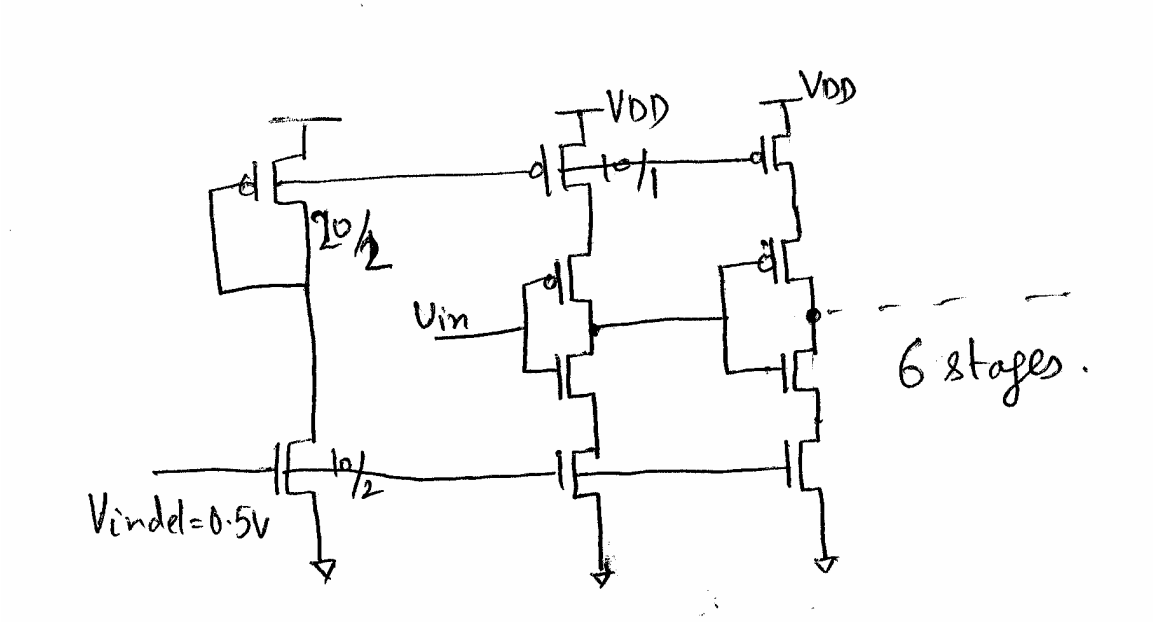
=2.34ff

N=15uA/500Mhz.2.33ff.1

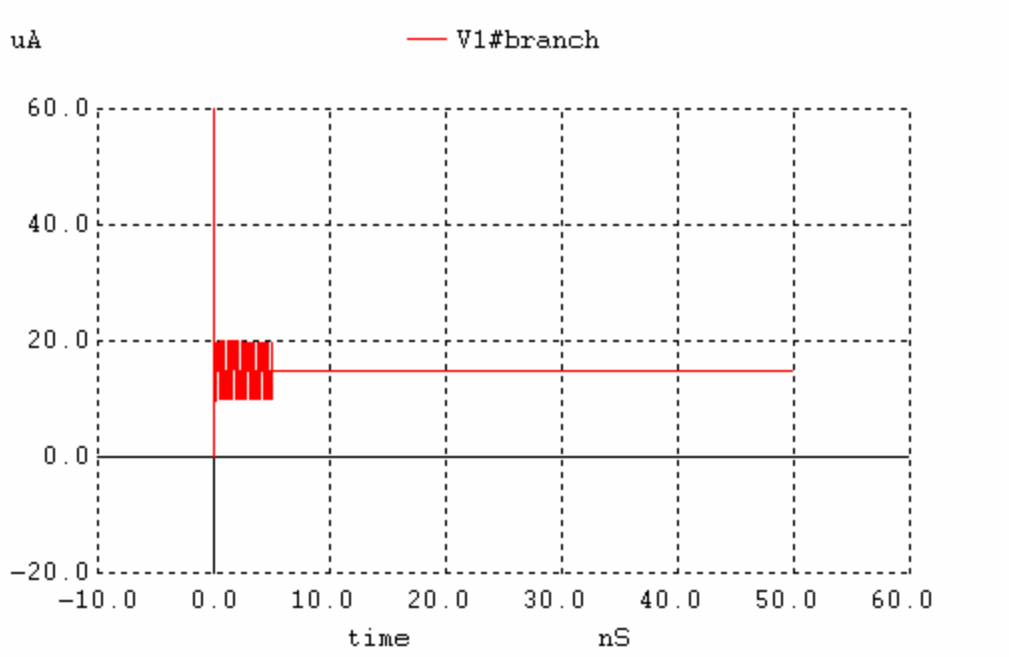
=6 stages

I=15u , Cox'=25ff/um^2

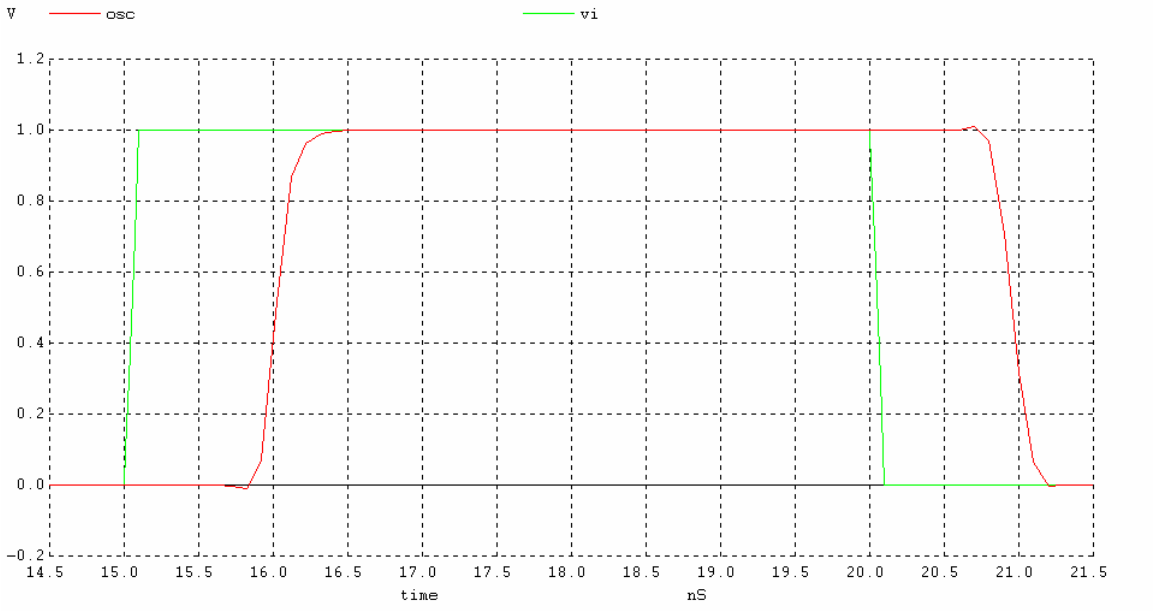
N=6.

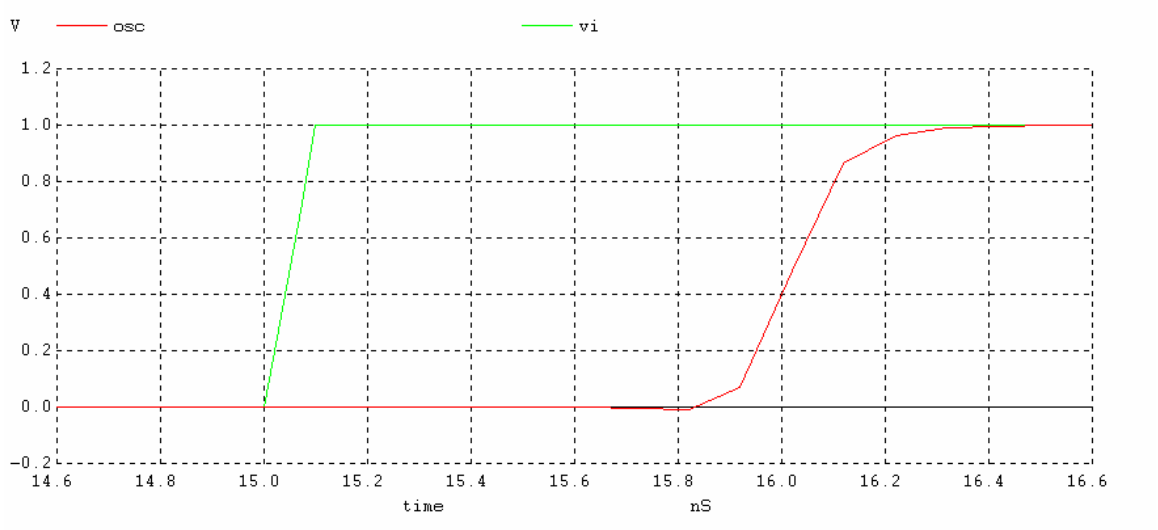


Here I have used 6 stages , Input is vi and output is osc , did simulation for 0.8V VDD , 1V VDD and 1.2V VDD

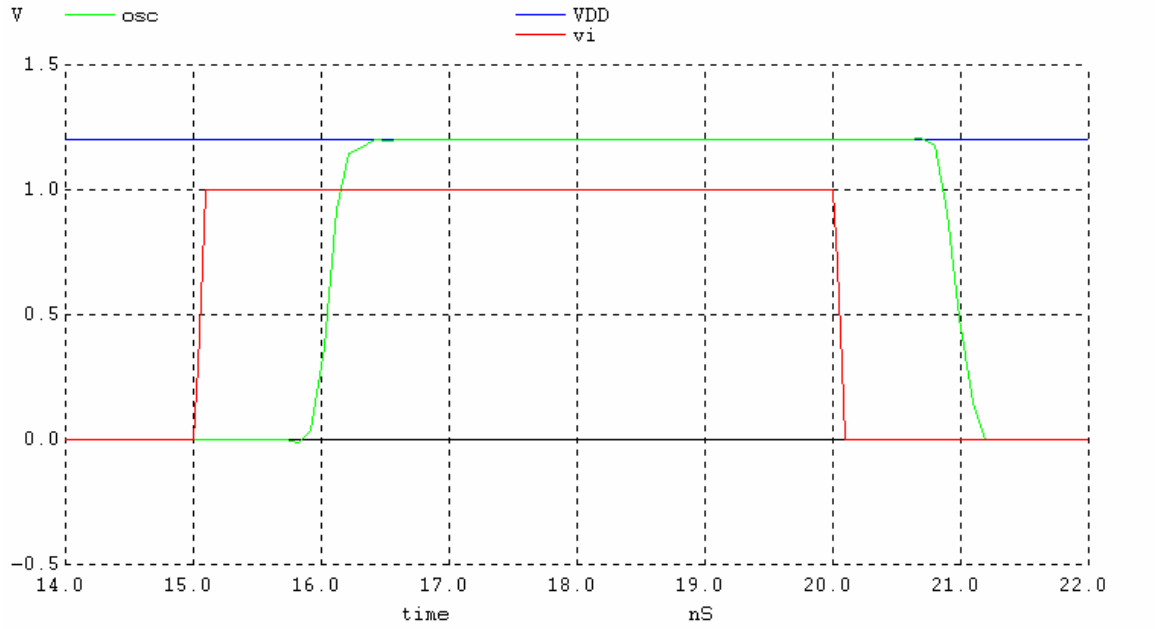


Ibias=15u , as per simulation

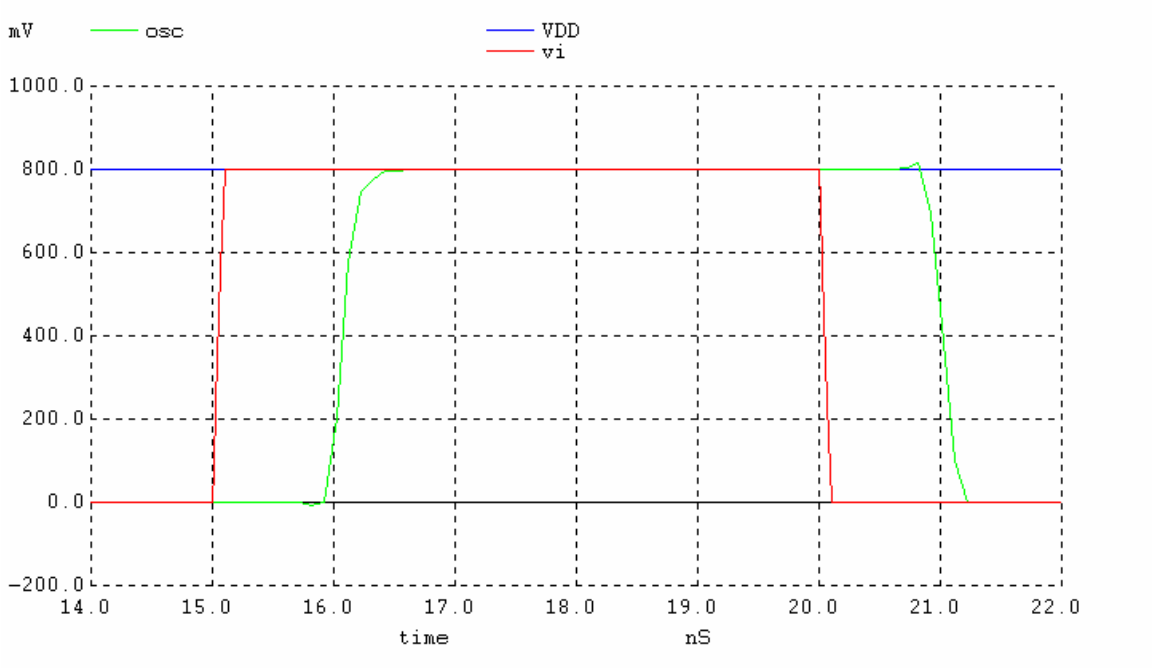




Simulation done at VDD=1V , td=1ns
Delay(Vi rising to osc rising) =1ns



VDD=1.2V Delay (vi rising to osc rising) =1.03ns



VDD=0.8V
(Vi rising to osc rising) =1.02 ns

This Ciruit delay does not change significantly with VDD because Vgs =0.5V for the NMOS for all the cases,

Final Examination problem : 19.23

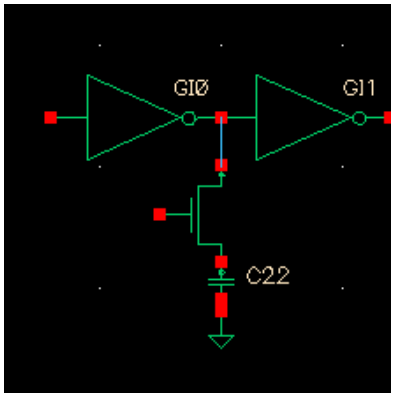
By Pandurang K. Irkar

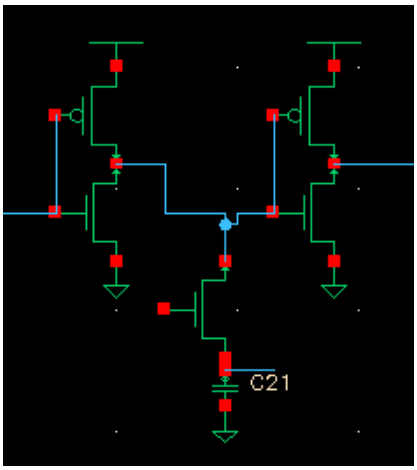
- Repeat problem 19.22 (design a nominal delay of 1ns when vindel=500mV. Determine the delay's sensitivity to variation in VDD) for the inverter delay cell in fig 19.55

Solution:

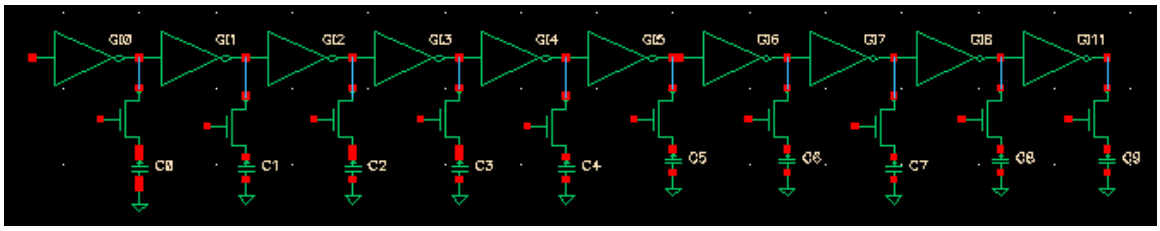
I) The simplified schematic of one stage of the nominal delay shown in fig. These individual stages are connected in series to achieve 1 ns nominal delay.

.... =>





fig(a)- individual delay stage (b) Schematic of individual stage



c) 10 individual stages connected to achieve 1 ns delay

From the above fig(b), the total capacitance on the drains of M1 and M2 is given by

Ctotal = Cout (previous stage) +Cin (next stage) + C
=Cox'(WpLp+WnLn) + (3/2) Cox'(WpLp+WnLn) +C
= (5/2) Cox'(WpLp+WnLn)+C (1)

Based on the charging and discharging time, the total time is equal to the addition of the charging and discharging time ie full cycle (t1+t2)

T=t1+t2
=Ctotal * (VDD/Id) (2)

To achieve specific dealy, we can use N stages then the total time equal to N*T

In this problem, we have to generate nominal 1ns delay ie to have complete cycle, the time =2ns
The frequency = 500Mhz
The no of stages required can be determined using

Freq= 1/(N*T)= Id/(N*Ctotal*VDD) (3)

Let us use a center drain current =10uA.

Ctotal =5/2 * 25*fF/um^2* (4*1.25 +8*1.25) (0.05)^2= 2.3fF
C << Ctotal, Wn=4um, Wp=8um, Lp=Ln=1.25um

The total no of stages are required to achieve the nominal 1ns delay is as follows:
From equation (3)

N= Id/(Freq*ctotal*VDD)= 10uA/(500 Mhz*2.3f*1) = 8.69

We use N=10; to achieve 1ns delay

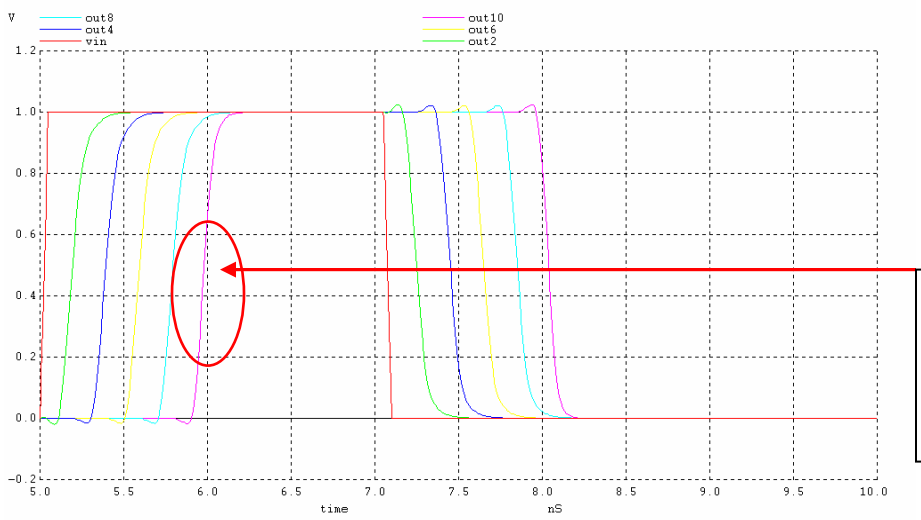
The fig(c) shows that all the 10 stages are connected in series fashion. **This schematic is simulated and found the dealy is equal to 1 ns at VDD =1V and Vindel =0.5V**

The Netlist is as follows:

```
********** Generating 1 ns delay using inverter delay cell (transistor and cap )
.control
destroy all
run
plot vin out2 out4 out6 out8 out10
.endc
.option scale=50n
.tran 10p 10n 5n 10p UIC
VDD     VDD     0       DC      1V
* For sensitivity check of delay with VDD variation
*VDD    VDD     0       DC      0.75V
*VDD    VDD     0       DC      1.25V
Vindel  vindel  0       DC      0.5
Vin     Vin     0       DC      0       pulse 0 1 0 50p 50p 2n 5n
Xdelay1         VDD     vin     out1    vindel  s1 delay
Xdelay2         VDD     out1    out2    vindel  s2 delay
Xdelay3         VDD     out2    out3    vindel  s3 delay
```

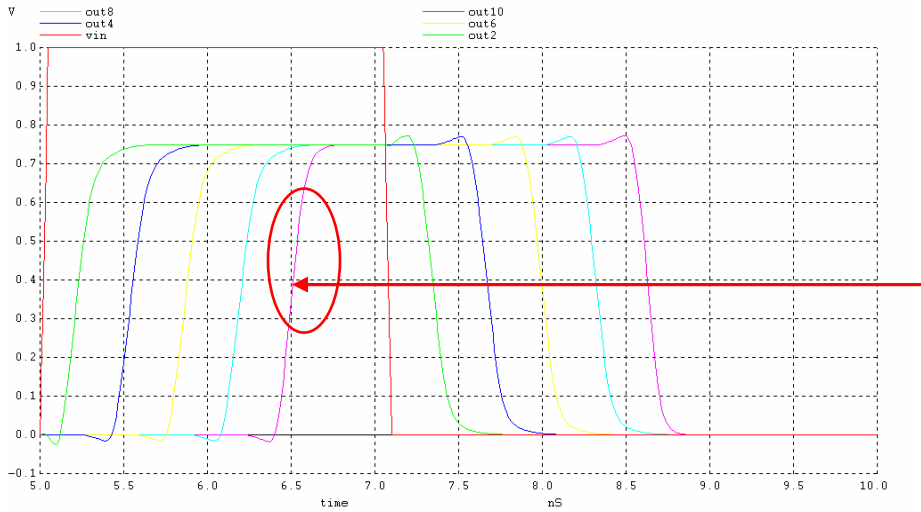
```
Xdelay4          VDD    out3    out4    vindel   s4 delay
Xdelay5          VDD    out4    out5    vindel   s5 delay
Xdelay6          VDD    out5    out6    vindel   s6 delay
Xdelay7          VDD    out6    out7    vindel   s7 delay
Xdelay8          VDD    out7    out8    vindel   s8 delay
Xdelay9          VDD    out8    out9    vindel   s9 delay
Xdelay10         VDD    out9    out10   vindel   s10 delay
.subckt delay    VDD    in      out     vindel   S
M1      out      in     0       0       NMOS  L=1.25  W=4
M2      out      in     VDD     VDD     PMOS  L=1.25  W=8
M3      out      vindel s       0       NMOS  L=1.25  W=4
C       s        0      1f
.ends
* BSIM4 models
.end
```

VDD=1V



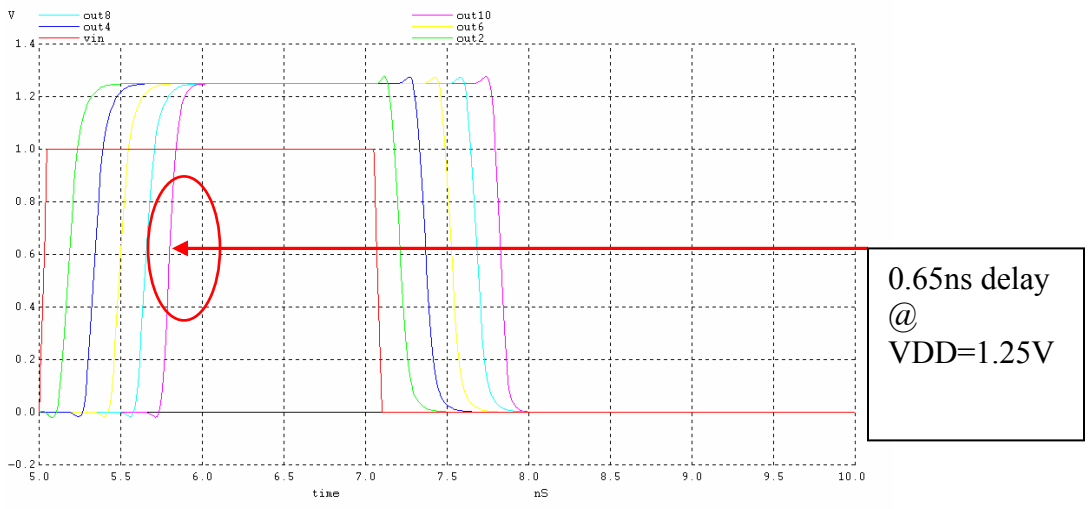
1ns delay @ VDD=1V

VDD=0.75V



1.5ns delay @ VDD=0.75V

VDD=1.25V



II) Sensitivity of delay with the variation of VDD

Simulation shows that as VDD increases the current increases resulting in shorter delay and vice-versa.

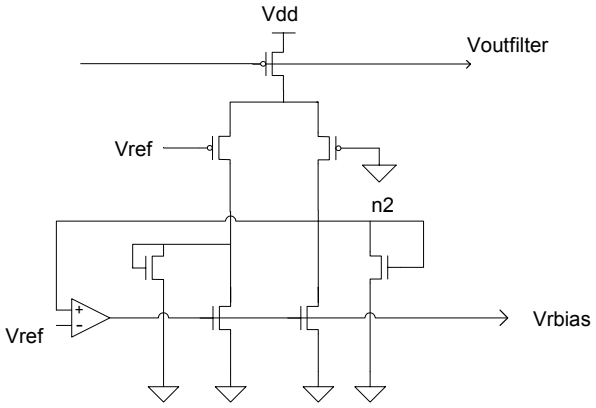
Sensitivity of delay = change in delay/change in VDD
= (1.5ns - 0.65ns)/(1.25-0.75)V
= 0.85ns/0.5V
= 1.7ns/V

The 1ns delay has been calculated and simulated. Also we checked the sensitivity of the delay with VDD and found that as VDD increases the delay decreases and vice-versa.

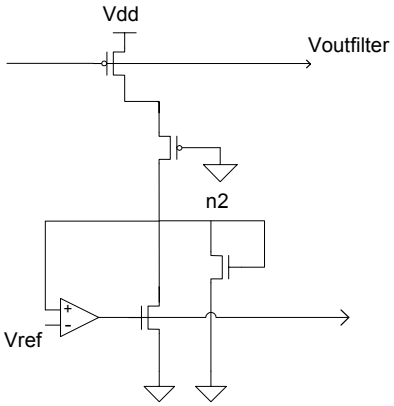
John Chen

Problem 19.24

Suppose, as seen in Fig 19.78, that instead of using a half-replica of the delay cell in Fig. 19.58, the full delay cell is used to generate V_{rbias}. Electrically is there any difference in V_{rbias} when comparing the full- and half-replica circuit? What may be the benefit of a full-replica of the delay cell?



full-replica of delay cell

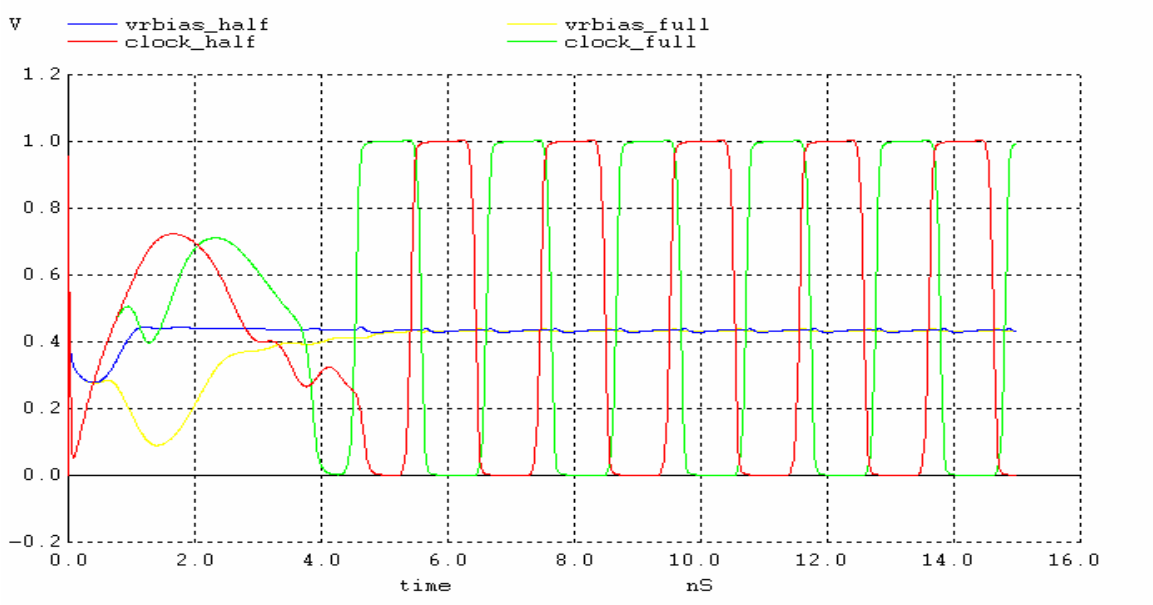


half-replica of delay cell

The schematic of two V_{rbias} bias circuits are shown as above. The first circuit used full-replica of the delay cell and the second one used half-replica of the delay cell. For the bias circuit with full-replica of the delay cell, the input voltage (=Vref) exceeds the maximum allowable input voltage for this differential amplifier. This turns one side of diff-pair (with input voltage of Vref) off and all the current flows in other side (with input voltage of 0V). The operation just likes the half-replica delay cell. So, electrically, there is no difference between the full- and the half-replica circuits. The main benefit of using full-replica of the delay cell is better matched parasitic between the delay cell and the bias circuit. So the output of each delay cell could be closer to Vref. In the practical implementation, the V_{rbias} bias circuit could use the same layout as delay cell to always get the same parasitic.

The simulation below compared V_{rbias} and frequency of VCOs with those two bias circuits. The spice input file includes two VCOs with full/half-replica of delay cell V_{rbias} bias circuits. As we can see, V_{rbias} and frequencies are the same for both VCO circuits. However, the simulation also shows it takes longer time for V_{rbias} generated by full-replica bias circuit to reach Vref (in about 5ns) since it takes time to turn off the other side of diff-pair.

The simulation results compared V_{rbias} and frequency of those two bias circuits.



The next page is simulation netlist.

*** Problem 19.24 CMOS: Circuit Design, Layout, and Simulation ***

```
.control
destroy all
run
let clock_full =inp2
let clock_half =inp
plot  clock_half clock_full vrbias_half vrbias_full
.endc

.option scale=50n
.tran 10p 15n UIC

VDD   VDD   0       DC     1
Vref   Vref   0       DC     0.5
Vinvco  vindel  0      DC     0.4


Mpb   Vp     Vp     VDD   VDD   PMOS L=2 W=100
Ibias   Vp     Vn     DC    10u
Mnb   Vn     Vn     0       0       NMOS L=2 W=50

.ic v(inp)=0
.ic V(inm)=0
**********************VCO_full****************
X1     VDD   vpbias vrbias_full     inp     inm     o1p     o1m    delay
X2     VDD   vpbias vrbias_full     o1p     o1m     o2p     o2m    delay
X3     VDD   vpbias vrbias_full     o2p     o2m     o3p     o3m    delay
X4     VDD   vpbias vrbias_full     o3p     o3m     o4p     o4m    delay
X5     VDD   vpbias vrbias_full     o4p     o4m     o5p     o5m    delay
X6     VDD   vpbias vrbias_full     o5p     o5m     o6p     o6m    delay
X7     VDD   vpbias vrbias_full     o6p     o6m     o7p     o7m    delay
X8t    VDD   vpbias          o7p     o7m     outp          delay_last
X8i    VDD   vpbias          o7m     o7p     outm          delay_last

Xpb3   VDD   outp    inp     inverter
Xpb2   VDD   outm    inm     inverter
**********************bias ckt1
Mb1    vpbias vindel  vr      0       NMOS L=1 W=100
Mb2    vpbias vpbias  VDD   VDD   PMOS L=1 W=20
Mb3    n1     vpbias  VDD   VDD   PMOS L=1 W=70
Mb4    n2     0       n1      VDD   PMOS L=1 W=20
Mb5    n2     vrbias_full     0       0       NMOS L=1 W=10
Mb6    n2     n2      0       0       NMOS L=2 W=10
Rr     vr     0       10k
X1     VDD   n2      vref    vrbias_full     pdiff
```

```
mf1    f2  vref      n1     VDD        PMOS L=1 W=20
mf2    f2  f2        0  0       NMOS L=1 W=10
Mf3    f2  vrbias      0        0        NMOS L=1 W=10
**********VCO_full end**********


**********VCO_Half**********
Xw1   VDD   vpbias vrbias_half    inp2    inm2   o1p2   o1m2   delay
Xw2   VDD   vpbias vrbias_half    o1p2    o1m2   o2p2   o2m2   delay
Xw3   VDD   vpbias vrbias_half    o2p2    o2m2   o3p2   o3m2   delay
Xw4   VDD   vpbias vrbias_half    o3p2    o3m2   o4p2   o4m2   delay
Xw5   VDD   vpbias vrbias_half    o4p2    o4m2   o5p2   o5m2   delay
Xw6   VDD   vpbias vrbias_half    o5p2    o5m2   o6p2   o6m2   delay
Xw7   VDD   vpbias vrbias_half    o6p2    o6m2   o7p2   o7m2   delay
Xw8t  VDD   vpbias          o7p2    o7m2   outp2          delay_last
Xw8i  VDD   vpbias          o7m2    o7p2   outm2          delay_last

Xpb3wVDD   outp2  inp2    inverter
Xpb2wVDD   outm2  inm2    inverter
**********bias ckt2
Mbw1  vpbias2          vindel  vr2     0       NMOS L=1 W=100
Mbw2  vpbias2          vpbias2         VDD     VDD    PMOS L=1 W=20
Mbw3  nn1      vpbias2         VDD     VDD    PMOS L=1 W=70
Mbw4  nn2      0       nn1     VDD     PMOS L=1 W=20
Mbw5  nn2      vrbias_half     0       0       NMOS L=1 W=10
Mbw6  nn2      nn2     0       0       NMOS L=2 W=10
Rrw     vr2     0       10k
Xw1   VDD   nn2     vref    vrbias_half     pdiff
**********VCO2 end**********


********subckts**********
.subckt delay   VDD   vpbias vrbias  vp      vm      vop     vom
Md1    n1      vpbias VDD   VDD    PMOS L=1 W=70
Md2    vom     vp      n1      VDD    PMOS L=1 W=20
Md3    vop     vm      n1      VDD    PMOS L=1 W=20
Md4    vop     vrbias  0       0       NMOS L=1 W=10
Md5    vop     vop     0       0       NMOS L=2 W=10
Md6    vom     vrbias  0       0       NMOS L=1 W=10
Md7    vom     vom     0       0       NMOS L=2 W=10
.ends

.subckt delay_last      VDD   vpbias vp      vm      vop
Ml1    n1      vpbias VDD   VDD    PMOS L=1 W=70
Ml2    vom     vp      n1      VDD    PMOS L=3 W=20
```

```
Ml3     vop     vm      n1      VDD   PMOS L=3 W=20
Ml4     vop     vom     0       0     NMOS L=3 W=10
Ml6     vom     vom     0       0     NMOS L=3 W=10
.ends

.subckt pdiff   VDD     Vp      Vm      vout
M1      n1      vb      VDD     VDD   PMOS L=1 W=20
M2      vb      vp      n1      VDD   PMOS L=1 W=20
M3      vout    vm      n1      VDD   PMOS L=1 W=20
M4      vb      vb      0       0     NMOS L=1 W=10
M5      vout    vb      0       0     NMOS L=1 W=10
.ends

.subckt inverter        VDD     in      out
M1      out     in      0       0     NMOS L=1    W=10
M2      out     in      VDD     VDD   PMOS L=1    W=20
.ends
*
.model  nmos  nmos  level = 14 …
.end
```

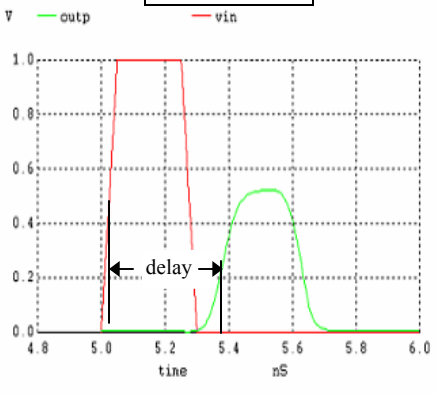
19.25 [Ravindra P]
For the delay line that generated the simulation results in Fig. 19.62 determine, using simulations, the delay's sensitivity to changes in VDD. Plot the VCDL's delay as a function of VDD with V_{indel} held at 500mV.

Soln.

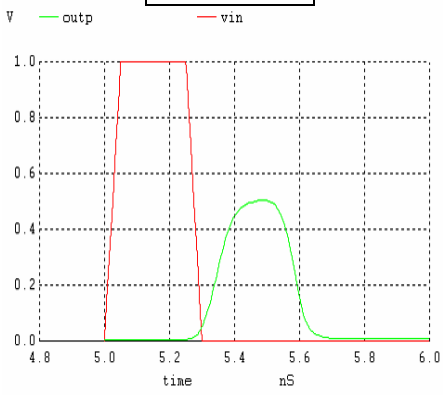
Both the bias circuit and the delay element are sensitive to changes in VDD. Looking at the bias circuit in fig. 19.58, change in VDD changes Iref in the drain-gate connected MOSFET. This changes V_{pbias} ($V_{outfilter}$) [the bias voltage] and affects the operation of the delay element.

The following simulations show the delay of the VCDL [outp] for three cases, when VDD=0.9, VDD=1 and VDD=1.1 As VDD increases, there is more current to charge the devices; so the devices will be fast and hence the delay of VCDL would decrease. The simulations confirm this.

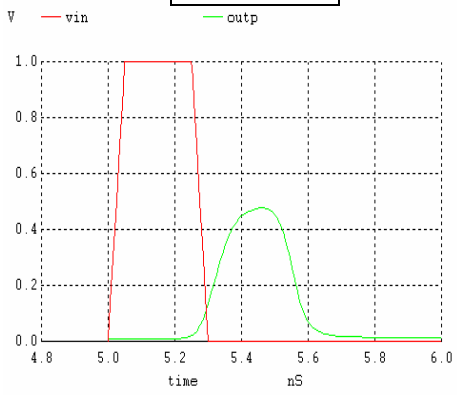
VDD=0.9V



VDD=1V

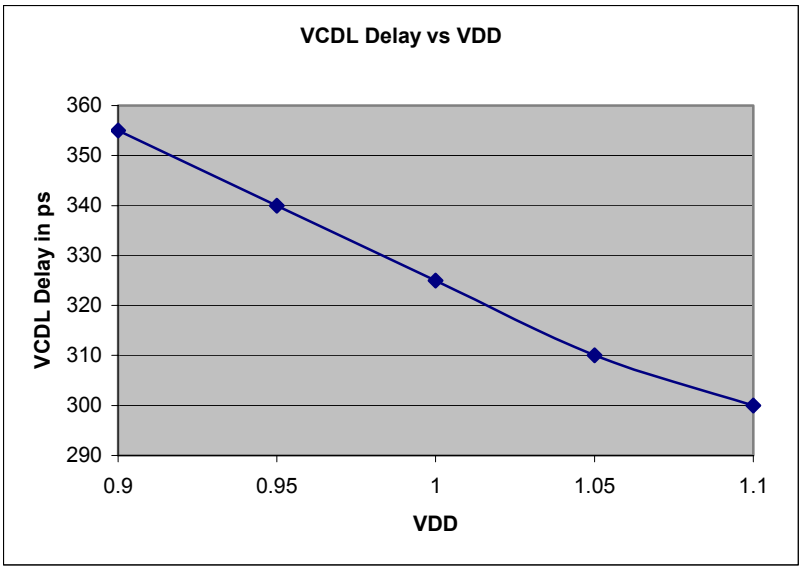


VDD=1.1V



V_{indel} is held at 500mv. To find VCDL's delay as a function of VDD, transient analysis is done with different values of VDD. The delay is the difference between the 50% points of outp and vin. To estimate the delay, it would be easy if we plot vin and outp+. 25

VDD	VCDL delay
0.9 V	355ps
0.95 V	340ps
1.0 V	325ps
1.05 V	310ps
1.1 V	300ps



NETLIST

```
*** prob 19.25 ***
.control
destroy all
run
PLOT     vin outp
.endc
.option scale=50n
.tran 10p 6n 5n 10p UIC

VDD      VDD      0        DC       1
Vref     Vref     0        DC       0.5
Vindel   vindel   0        DC       0.5
Vin      Vin      0        DC       0        pulse 0 1 5n 50p 50p 0.2n 2n

Xdline   VDD      vref     Vindel   vind     vini     outp     outm     dline
Xtg      VDD      0        VDD      vin      vind     tg
Xin      VDD      vin      vini     inverter

.subckt dline     VDD      vref     vindel   inp      inm      outp     outm
Xbias    VDD      vref     vpbias   vrbias   vindel                     dbias
X1       VDD      vpbias   vrbias   inp      inm      o1p      o1m      delay
X2       VDD      vpbias   vrbias   o1p      o1m      o2p      o2m      delay
X3       VDD      vpbias   vrbias   o2p      o2m      o3p      o3m      delay
X4       VDD      vpbias   vrbias   o3p      o3m      o4p      o4m      delay
X5       VDD      vpbias   vrbias   o4p      o4m      o5p      o5m      delay
X6       VDD      vpbias   vrbias   o5p      o5m      o6p      o6m      delay
X7       VDD      vpbias   vrbias   o6p      o6m      o7p      o7m      delay
X8       VDD      vpbias   vrbias   o7p      o7m      outp     outm     delay
.ends

.subckt  delay    VDD      vpbias   vrbias   vp       vm       vop      vom
M1       n1       vpbias   VDD      VDD      PMOS L=1 W=200
M2       vom      vp       n1       VDD      PMOS L=1 W=20
M3       vop      vm       n1       VDD      PMOS L=1 W=20
M4       vop      vrbias   0        0        NMOS L=1 W=10
M5       vop      vop      0        0        NMOS L=2 W=10
M6       vom      vrbias   0        0        NMOS L=1 W=10
M7       vom      vom      0        0        NMOS L=2 W=10
.ends

.subckt dbias     VDD      vref     vpbias   vrbias   vindel
M1       vpbias   vindel   vr       0        NMOS L=1 W=100
M2       vpbias   vpbias   VDD      VDD      PMOS L=1 W=20
M3       n1       vpbias   VDD      VDD      PMOS L=1 W=200
M4       n2       0        n1       VDD      PMOS L=1 W=20
M5       n2       vrbias   0        0        NMOS L=1 W=10
M6       n2       n2       0        0        NMOS L=2 W=10
Rr       vr       0        10k
X1       VDD      n2       vref     vrbias   pdiff
.ends

.subckt  pdiff    VDD      Vp       Vm       vout
M1       n1       vb       VDD      VDD      PMOS L=1 W=20
M2       vb       vp       n1       VDD      PMOS L=1 W=20
M3       vout     vm       n1       VDD      PMOS L=1 W=20
M4       vb       vb       0        0        NMOS L=1 W=10
M5       vout     vb       0        0        NMOS L=1 W=10
.ends

.subckt inverter  VDD      in       out
M1       out      in       0        0        NMOS     L=1     W=10
M2       out      in       VDD      VDD      PMOS     L=1     W=20
.ends

.subckt  tg       VDD      pg       ng       in       out
M1       out      pg       in       VDD      PMOS L=1 W=20
M2       out      ng       in       0        NMOS L=1 W=10
.ends
```

Work presented by: Sandeep Pemmaraju

PROBLEM 19.26:

PART1: DC ANALYSIS:

The delay element shown in the figure below does not use the reference voltage.
The plus and minus terminals are connected to Mp1, Mp2 and Mp3, Mp4 respectively as shown below.
Lets discuss what happens when the Inp is swept from 0 to 1 volt with Inm kept constant at 0.5V.Intially when Inp=0, then all the PMOS devices are ON but Mp3 and Mp4 will conduct more current than Mp1 and Mp2.This makes the drain potential of Mn4 greater than Mn1 (with the same gate potential on each MOSFETs). This keeps OUTp around a V_{dssat} and OUTm at $V_{DD} - 2.V_{dssat.}$

In the same lines it can be said that as Inp increase, Mp3 and Mp4 start to turn OFF and the drain voltage of Mn1 increases (i.e. OUTp starts to increase). The drain potential of i.e. OUTm starts to decrease to zero.

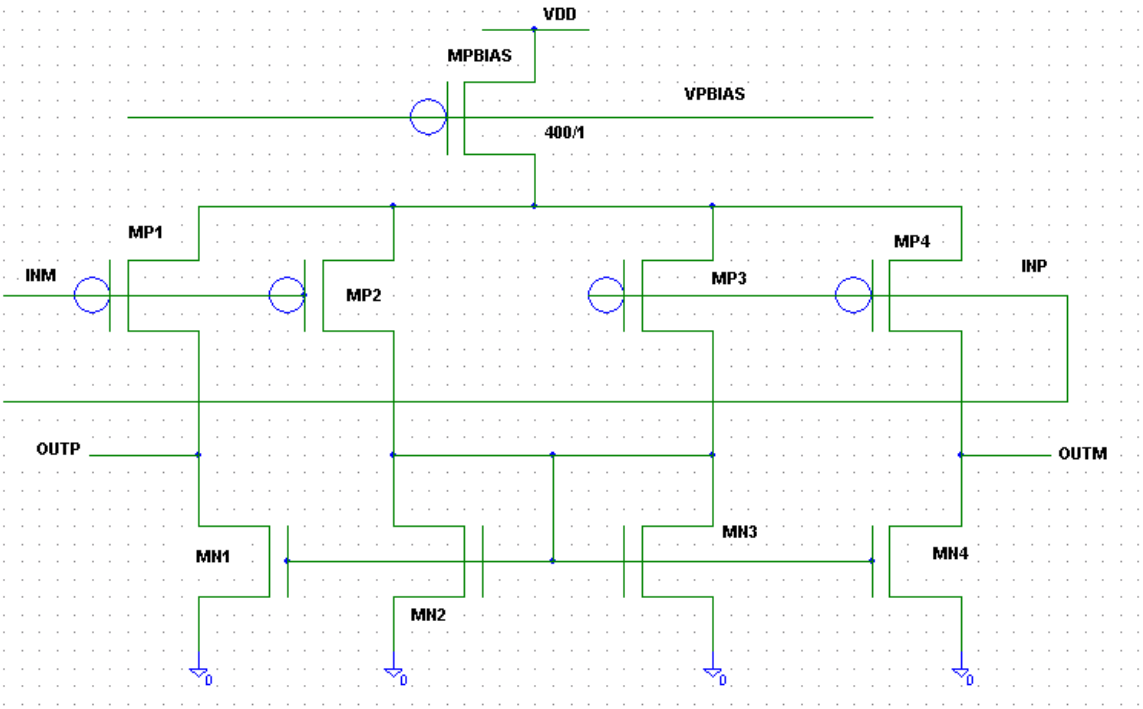


Figure1: showing the delay element without the reference voltage

The above discussion becomes more clear observing the plots shown below:

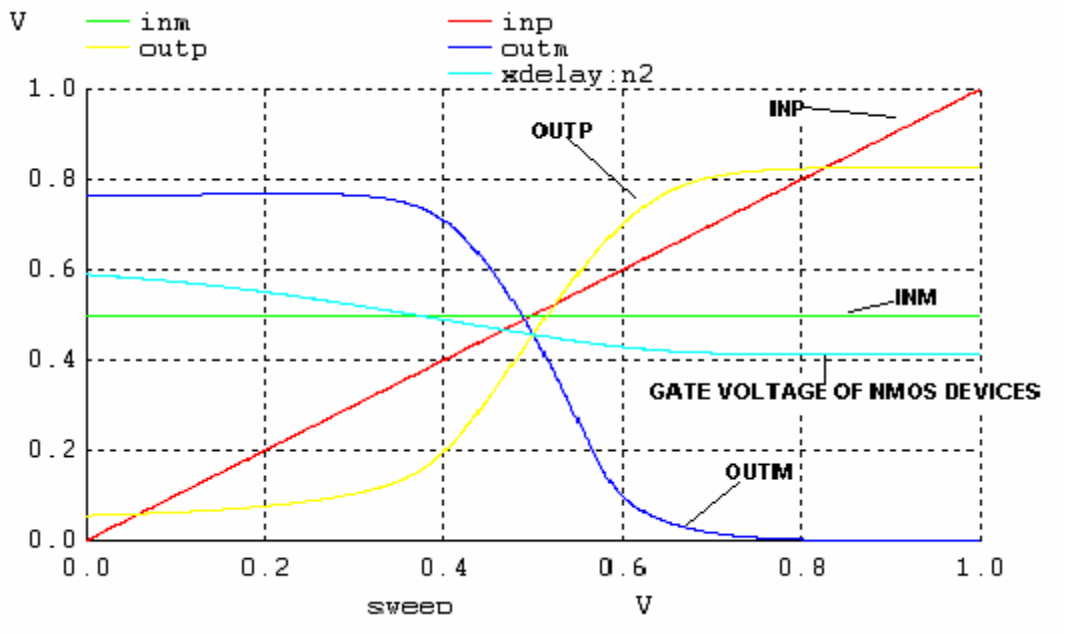


Figure2: Variations in outm, outp, gate voltage of NMOS devices.

One can see some other interesting points in the graph:
The Inm node increases even though it is connected to a voltage source. This is because as Inp increase, Mp3 and Mp4 shut off and hence only Mp1 and Mp2 will be conducting. The current flowing through Mn2 will set its gate potential but at the same time since Mn3 is OFF, its drain will try to come down to ground potential, so there is a fight between the two which results in a reduced gate potential of Mn1 and Mn2.But to keep the total current constant through each branch, the source voltages of Mp1 and Mp2 will increase.
So the bottom line here is the gate voltage of NMOS devices is almost a constant except a small decrease with increase in Inp.

PART2: Voltage Controlled Delay Line (VCDL):
The circuit in figure1 can be used as a delay element. Here the Nmos devices are self biased and hence the outputs of the delay line can go higher unlike the delay element in the textbook. But there are many issues that are to be considered when using this circuit as a delay element.
First lets discuss how the above circuit can be used as a voltage controlled delay line. Here the inputs are connected to two PMOS devices. This makes the input capacitance of the delay stage twice of that the delay element in Figure 19.61.So the delay is more when compared to the circuit in fig19.61.Lets simulate the circuit shown in figure19.61 but with only 4 stages and compare the results to figure19.62 in text book. The schematic of the circuit that is simulated is shown below and following the schematic are the results.

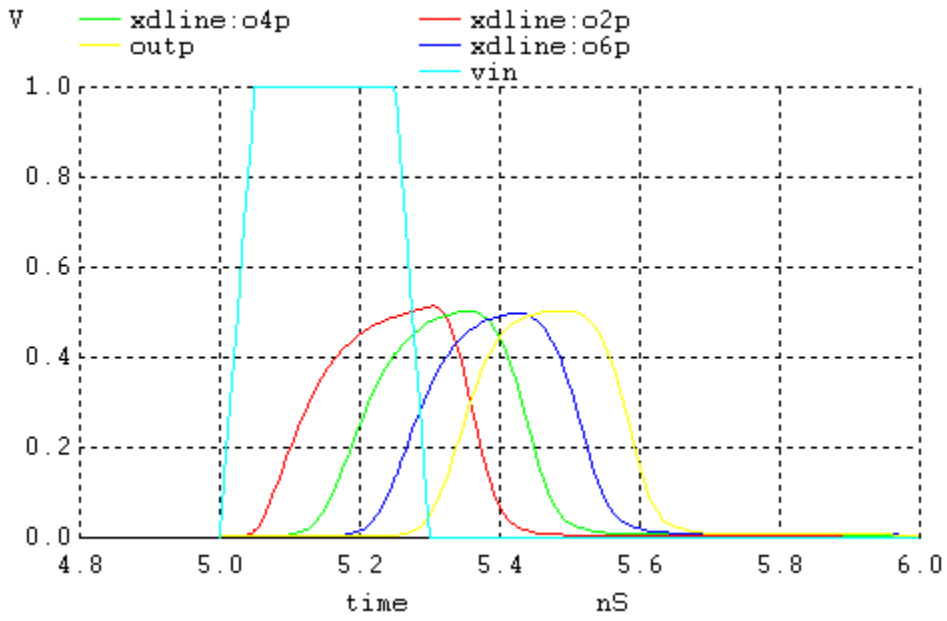


Figure4: This is the same figure as in fig19.62 of text book (using 8 stages).

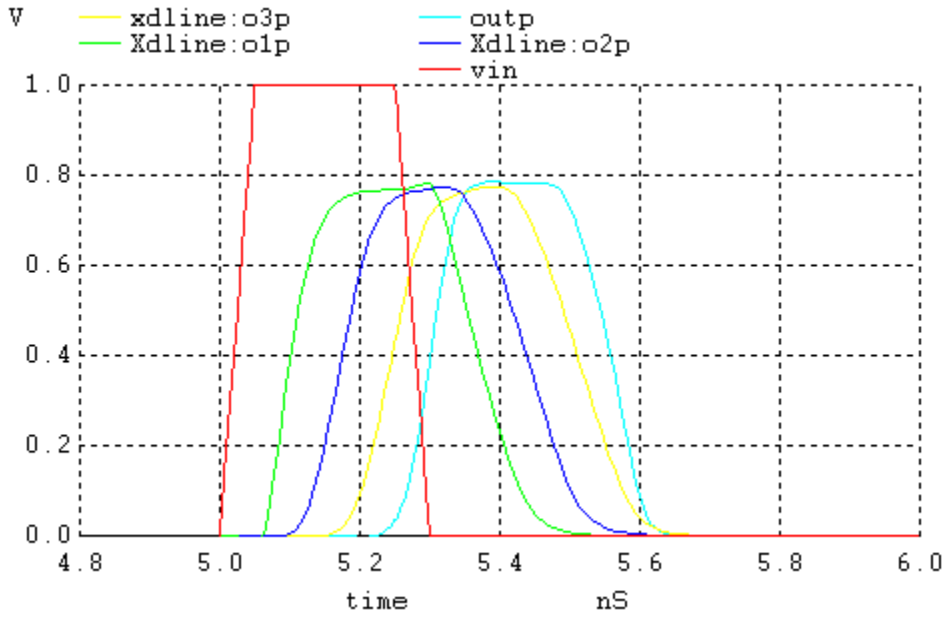


Figure3: Showing the plots generated using figure1 with 4 stages in series.

Comparing figure3 and figure4 it is clear that the present circuit, which was simulated with only 4 stages, has greater delay. So the disadvantage of this topology is that the minimum frequency that can be attained would be less.

PART3: Dependence on VDD:

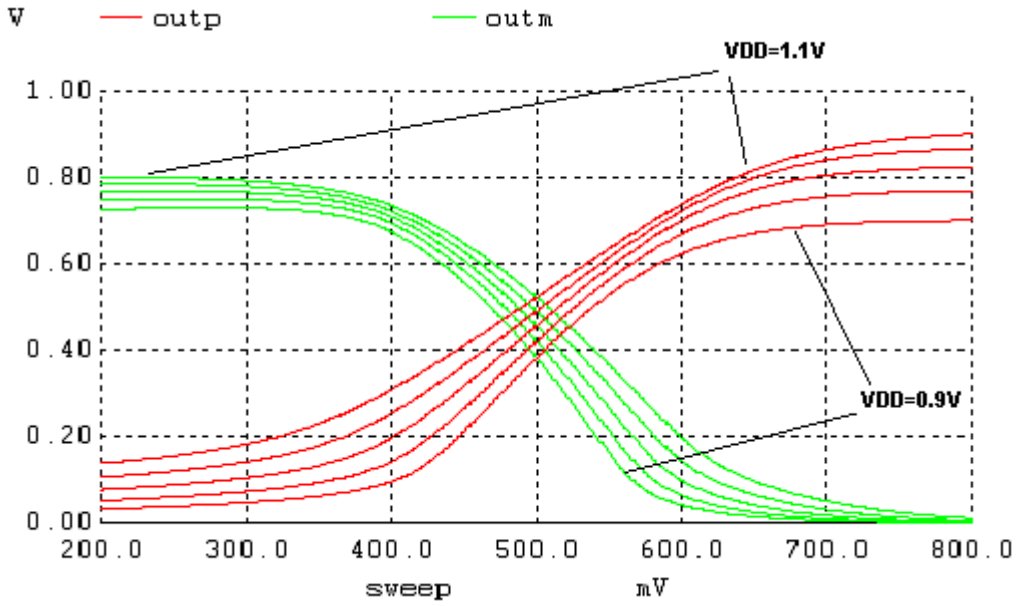


Figure5: Showing how the output changes with Vdd

Description of figure5: Inp was swept from 0 to 1V with Inm=0.5V and V_{DD} swept simultaneously from 0.9V to 1.1V in steps of 50mV.

Variation in OUTm:

Initially when Inp=0, Mp3 and Mp4 are ON and source more current than Mp1 and Mp2 (V_{SG} for Mp3, 4 is higher than Mp1, 2.). Hence outm is at a greater potential determined by V_{DD}-2.V_{dssat}.AS Inp increases Mp3 and Mp4 tend to shut off and hence outm is pulled to ground by Mn4 (remember gate of Mn4 is at higher potential determined by the current flowing through the gate drain connected Mn2).

Variation in OUTp:

(1) Initially when Inp=0,almost all the lines of Inp are at a potential depending on the over-head voltage available which is V_{DD}.It is to be remembered that a V_{dssat} is sufficient at the drain of Mn1 to make the desired current flow.

(2) For lower values of V_{DD}, the drain of Mpbias or source of Mp3 and Mp4 will be at a lower potential and hence mp3 and Mp4 turn off for a lower value of Inp (they turn off when V_{SG}=0) Since they turn off for a lower value of Inp, Mp1 and Mp2 start to take decision at a lower value of Inp and hence OUTp starts to increase at lower value of Inp when compared to a Higher V_{DD} plot (Observe the steepness in the slope of OUTp). This can be clearly seen in the plot above. The red lines with higher value of V_{DD} tend to increase at a higher value of Inp.

(3) As said in the previous lines (point (2) above), the drain of Mpbias will be at a higher voltage for a higher V_{DD} and hence the voltage on the drain of Mp1 will be higher to source the same current. So, the final value of OUTp will be at a higher potential for higher values of V_{DD}.

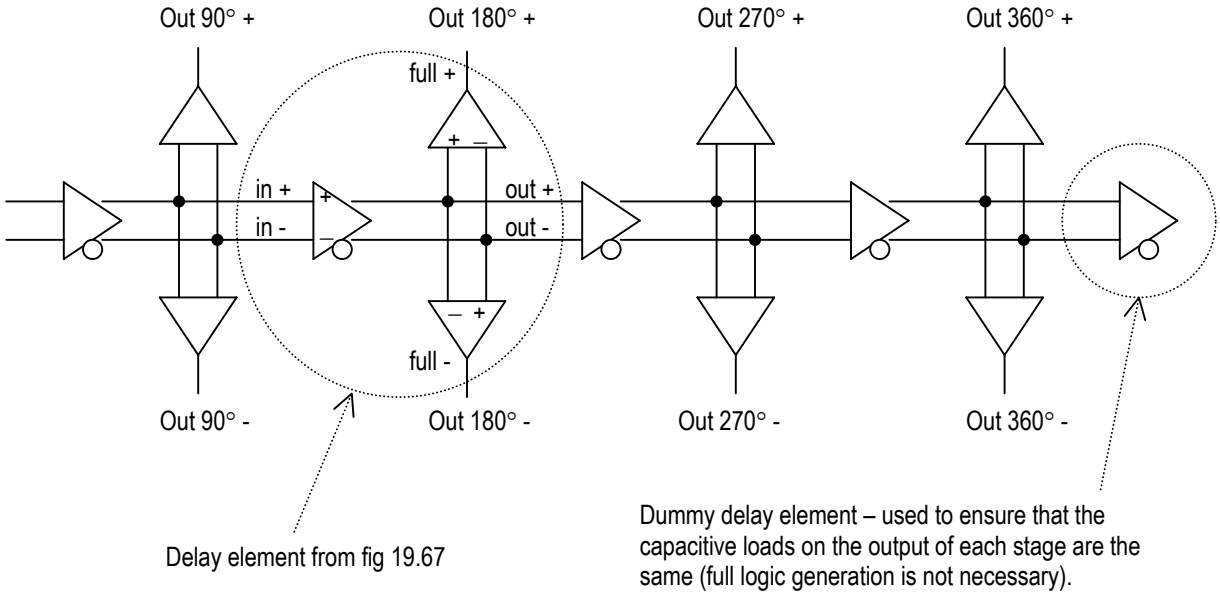
NETLIST:

```
.control
destroy all
run
plot      vin Xdline:o1p Xdline:o2p xdline:o3p outp  ylimit 0 1
.endc
.options scale=50n          rshunt=1e10
.tran 10p 6n 5n 10p UIC

vdd       vdd       0         DC        1
Vindel    vindel    0         DC        0.5
Vin       vin       0         DC        0         pulse 0 1 5n 50p 50p 0.2n 5n

XTG       vdd       0         vdd       vin       inp       TG
Xinv      vdd       vin       inm       inverter
Xdline    vdd       vindel    inp       inm       outp      outm      dline

.Subckt dline       vdd       vindel    inp       inm       outp      outm
Xbias     vdd       vpbias    vindel    bias
X1        vdd       vpbias    inp       inm       o1p       o1m       delay
X2        vdd       vpbias    o1p       o1m       o2p       o2m       delay
X3        vdd       vpbias    o2p       o2m       o3p       o3m       delay
X4        vdd       vpbias    o3p       o3m       outp      outm      delay
.ends

.subckt   bias      vdd       vpbias    vindel
Mindel    vpbias    vindel    vr        0         NMOS L=1 W=100
rr        Vr        0         10k
Mpbias    vpbias    vpbias    vdd       vdd       PMOS l=1 W=10
.ends

.subckt   delay     vdd       vpbias    inp       inm       outp      outm
Mbias     n1        vpbias    vdd       vdd       PMOS L=1 W=400
Mp1       outp      inm       n1        vdd       PMOS L=1 W=20
Mp2       n2        inm       n1        vdd       PMOS L=1 W=20
Mp3       n2        inp       n1        vdd       PMOS l=1 W=20
Mp4       outm      inp       n1        vdd       PMOS L=1 W=20
Mn1       outp      n2        0         0         NMOS L=1 W=10
Mn2       n2        n2        0         0         NMOS l=1 W=10
Mn3       n2        n2        0         0         NMOS l=1 w=10
Mn4       outm      n2        0         0         NMOS L=1 W=10
.ends

.subckt   inverter  vdd       in        out
Mp        out       in        vdd       vdd       PMOS L=1 W=20
Mn        out       in        0         0         NMOS L=1 W=10
.ends

.subckt   tg        VDD       pg        ng        in        out
M1        out       pg        in        VDD       PMOS L=1 W=20
M2        out       ng        in        0         NMOS L=1 W=10
.ends
```

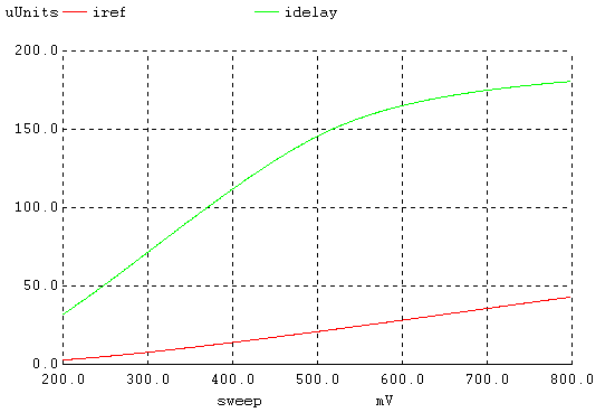
Solution to problem 19.27 (by Eric Booth)

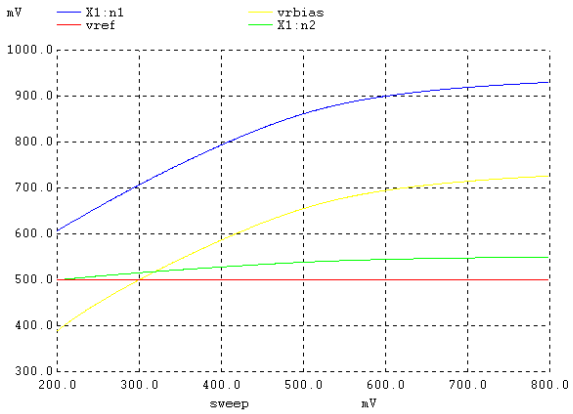
Use the delay element in Fig. 19.67 to implement a VCDL. Use the VCDL in Fig. 19.65 to generate the waveforms in Fig. 19.66. Show the 90, 180, 270, and 360 degree outputs of the VCDL swinging to full-logic levels.

Below is a block diagram of the VCDL using the delay element from fig 19.67. Only 4 stages are needed for the 500ps delay because the input capacitance of the level-restoring diff-amps contributes to the total capacitive load of each stage, increasing the delay. When using this VCDL configuration, a dummy delay-cell can be used at the end of the line to ensure that the capacitive load of each stage is equal. The positive 360° output is fed back to the PFD and will be in phase with the input when the DLL is locked.

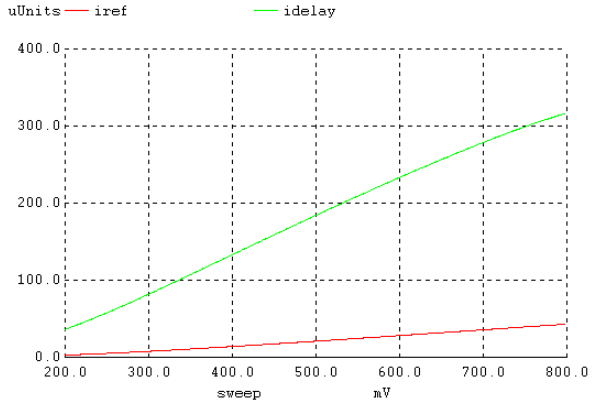


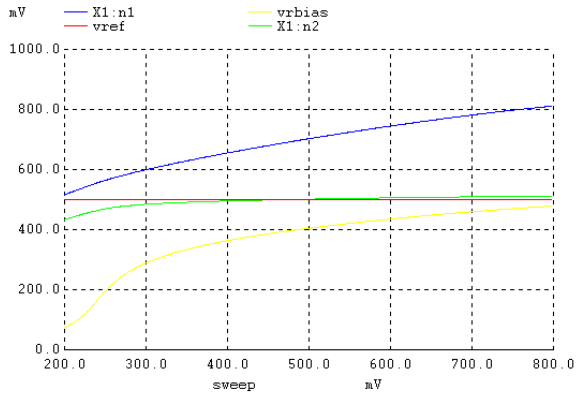
In order for the new VCDL to operate properly, I had to address a couple of issues with the bias circuit. Below, I regenerated the plots of figure 19.58 with the biased PMOS sized at 200/1 (without stepping Vdd). The plot on the left shows that although the reference current is linear over the full range, the current mirrored to the other branch of the circuit (and the delay elements) is only linear up to Vinvco=.5V. Above that, the gain drops to almost 0. Although it is not obvious at first glance, the plot on the right shows why this happens. First, the voltage-controlled resistor reference voltage (vrbias) begins to approach Vref+Vthn causing the NMOS to triode. Second, the Vsg of the 20/1 PMOS (with it's gate connected to ground) is not large enough to source the required current when Vinvco>.5V. This can be seen by n1 approaching Vdd and having no-where else to go without causing the other PMOS to triode.



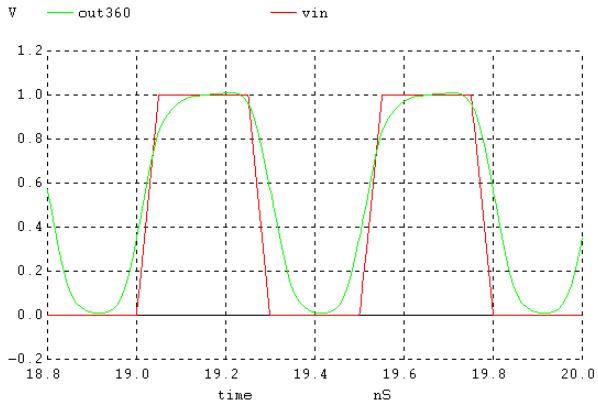


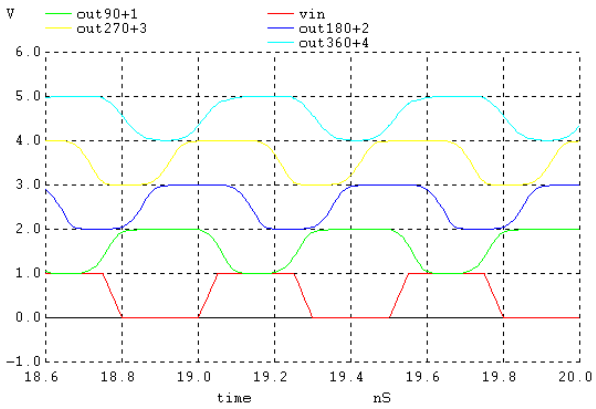
To fix these problems, I increased the width of both NMOS' and the 20/1 PMOS to 50. Below, I again regenerated the plots of figure 19.58 with the new device sizes. We can see that not only are both currents linear over the full range, but node n2 is held much more steady at Vref.





An analysis of the VCDL showed that the nominal delay (Vinvco=.5V) was 450ps, and the gain Kv was 850ps/V. Since the Kv of the new VCDL was not significantly different than the Kv from the example, I dropped my new VCDL into the DLL without modifying Ipump or C1. Below are plots of the DLL's output with a 2GHz input signal. The figure on the left is very similar to fig 19.66. The loop locks up to the rising edge of the input, and there is a visible duty cycle error caused by regenerating full logic levels. The figure on the right is the output of each of the 4 phases with respect to the input.





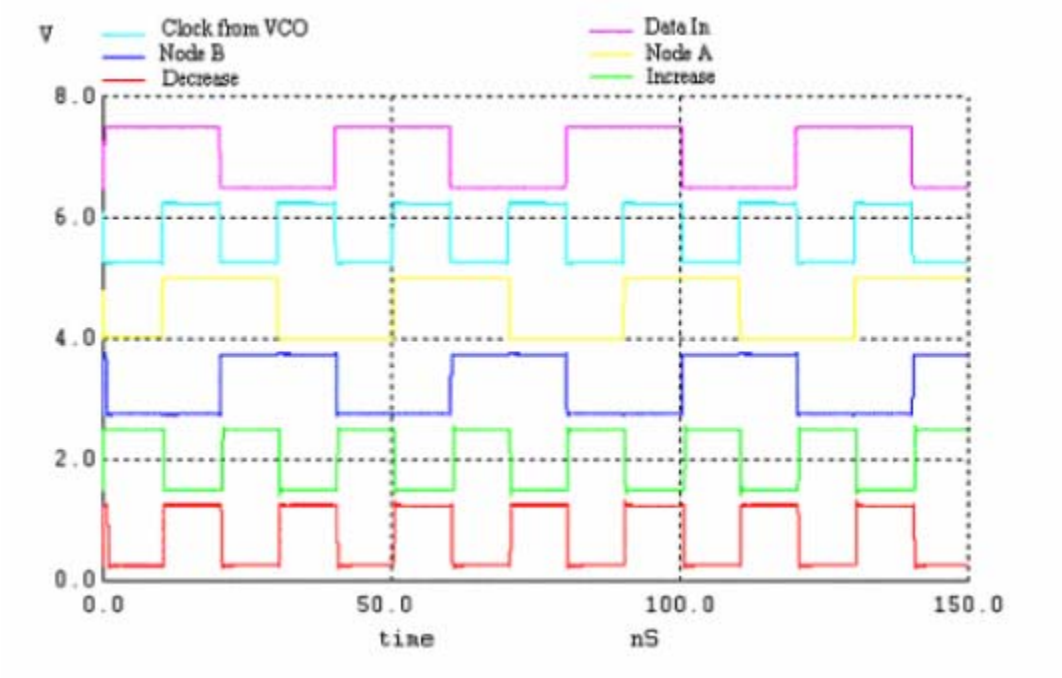
The simulation shows all outputs swinging to full logic levels as expected, however the only output signal with the correct phase shift is 360°. The 90°, 180° and 270° outputs are actually occurring at 180°, 240° and 300° respectively. The reason for this is that the delay through the tg/inverter (td1) at the beginning of the line, and the delay through the logic restoring elements (td2) are significant compared to the total delay of the line. I characterized these two delays and found that td1=50ps and td2=100ps. This is 30% of the period (or about 120°) when the loop is locked at 2GHz. Since 120° of the delay line are taken up by td1 and td2, the feedback forces the 4 delay cells to make up the remaining 240°, or 60° per cell. Because the 360° output will always be in phase with the input when the DLL is locked, we can refer to it as ϕ_0. We can then recalculate the phase shift of the multi-phase output signals with the extra delay added in. The 90° output will occur at ϕ_0 + 120° (td1+td2) + 60°, or 180°. The remaining outputs will each occur 60° later (the cell to cell delay), giving the 180°, 240°, 300° and 360° output signals seen in the simulation. This shows that this topology should only be used for generating multi-phase outputs when the delay through the logic-restoring elements is small compared to the delay through the delay elements.

Problem 19.28 **Dennis Montierth**

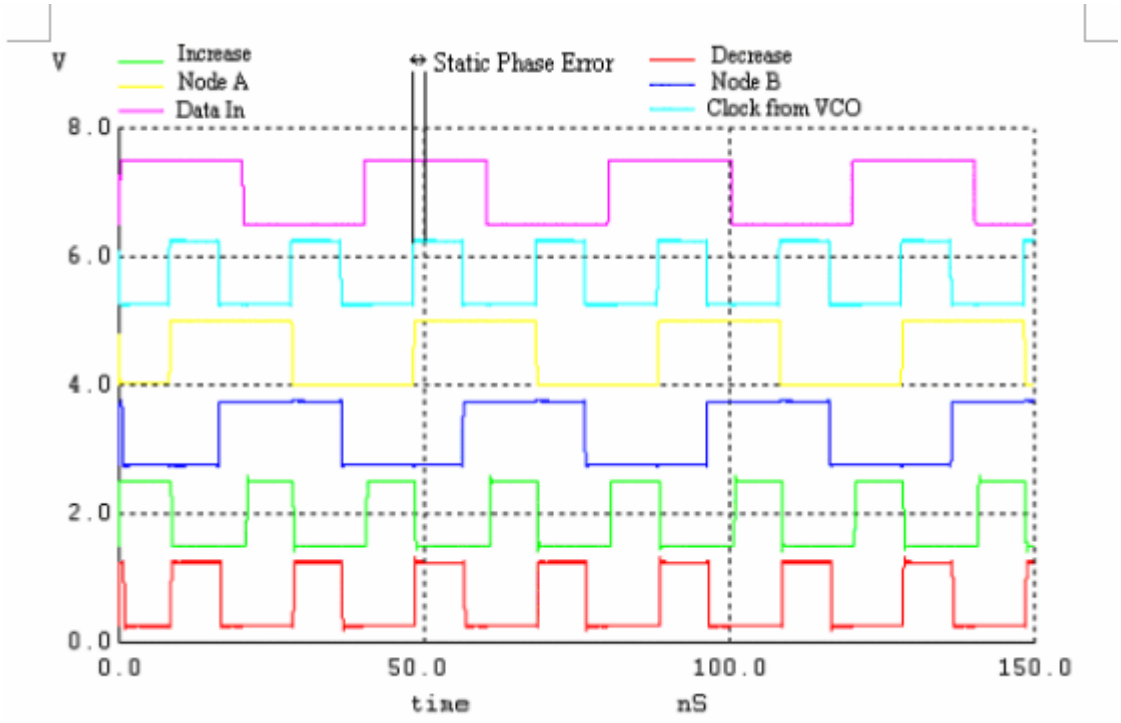
Will a VCO, which does not produce an exact 50% duty cycle, have an adverse affect on a clock recovery circuits operation?

I am going to simulate the Hogge Phase detector to show that it is very dependant on the VCO's duty cycle and that not having a 50% duty cycle does affect the operation of the clock recovery circuit.

Here is the operation of the Clock Recovery Circuit where the PLL is locked up and the VCO is producing a 50% duty cycle. There is no static error because the VCO waveform is rising exactly in the middle of the data. The thing to note here is that if you integrate the increase pulse and the decrease pulse they will be exactly the same. This is the case when the PLL is in lock.



Now notice what happens when I change the duty cycle of the clock produced by the VCO being high only 40% of the time. The next graph shows that when the PLL is locked in (Increase is on the same amount of time as decrease) there is a static phase error. The PLL will still be able to lock onto the correct frequency but the rising edge of the output clock will not rise in the center of data.



You can look at Node A, Node B, Data-in, and the VCO clock to explain why this is the case. Node A gets XOR'D with Data-in to produce the increase signal. Node A loads in Data-in on the rising edge of the VCO clock cycle. Therefore, the increase signal does not turn on until data transitions while the data loaded into Node A remains the same. In the case where the PLL is locked the rising edge of the VCO clock turns off the increase signal. This makes the width of the increase signal dependant on the amount of time the VCO cycle stays low after the data transition. Node B is XOR'D with Node A to produce the decrease signal. Node B loads in Node A's value on the falling edge of the VCO clock cycle. Node A transitioning causes the decrease signal to turn on. The decrease signal stays on until the VCO transitions low again. This makes the decrease signal the width of the VCO high time. If you look carefully at the graph, the only case where you can have increase and decrease on for the same amount of time and also have the VCO rising in the center of data is to have a 50% duty cycle. Again, this is because decrease is on for the width of the VCO cycle high time and decrease is on for the width of the VCO low time after the data transitions. I have shown this using a data stream of 1, 0, 1, 0 … but it would work the same no matter what the data looks like (but only having pulses where you get data transitions) since the increase and decrease pulses occur based on the data transition.

Kloy Debban
P19.29
To equalize the sizes of increase and decrease in figure 19.70 is the same as equalizing the delay seen by the XOR gate, between the Q output of the DFF and the NRZ data. There are two ways to think about this: The delay through the DFF either has to be decreased or the NRZ data has to be delayed, (see figure 19.71.)

The only way to decrease the delay through the DFF would be to omit the inverters used as buffers on the outputs of the DFF, (see figure 19.69.) Since the XOR gate needs (as inputs,) both the output and the inverted output of the DFF, only one of these inverters can be omitted. This will not decrease the delay through the DFF enough to compensate for the added delay of the 2 inverters and the capacitor used in figure 19.71.

So, trying to add delay between the NRZ data and the XOR gate seems to be the only way to equalize the sizes of increase and decrease.

It seems logical that one could add another DFF on the input of NRZ and clock it on the falling edge of clock (inverted clock) and the Q output of this DFF would just be NRZ delayed the same amount as the Q output of the other DFF, (the theoretical curves used to verify this can be seen below in figure 1.)

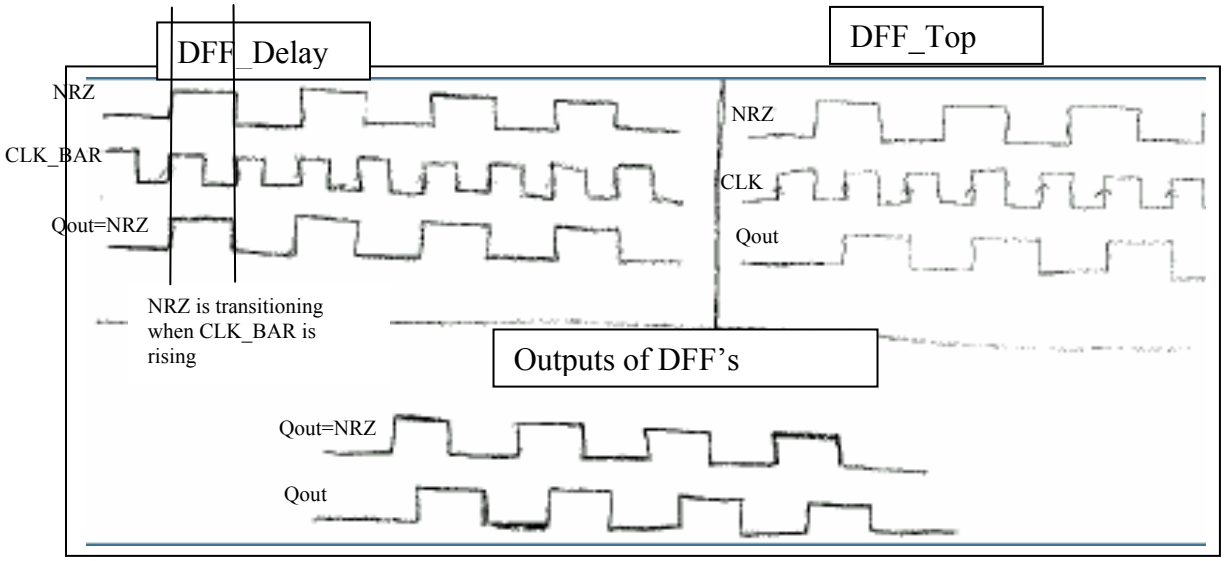


Figure 1.

The WinSPICE netlist used to attempt this is at the end of this solution. As can be seen below in figure 2, the width of increase actually became wider and decrease stayed the same, (compare to figure 19.70.) The reason for this is because the falling edge of clock occurs while the NRZ data is not completely set, (or is transitioning from high to low or low to high, see figure 1.) This causes the DFF to output something unpredictable and makes NRZ appear to happen earlier in time. As a result the delay between NRZ and node A in figure 19.71 is even greater, making increase wider. The WinSPICE plot below in figure 2, verifies this.

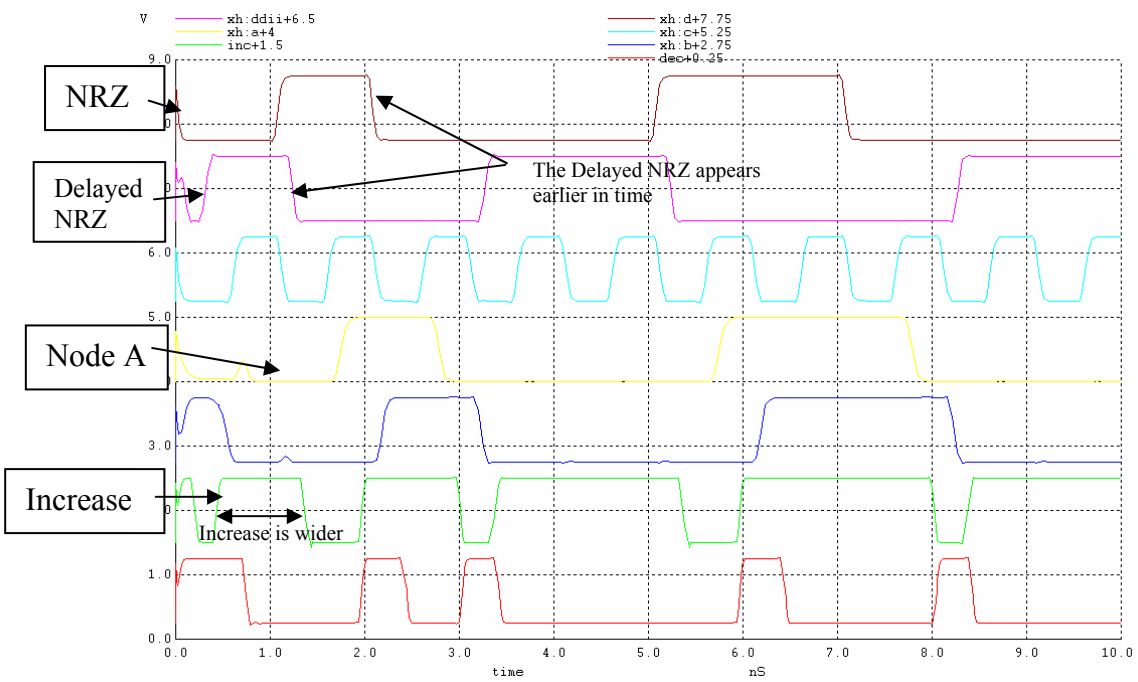


Figure 2.

Next, I tried putting an AND gate, with inputs of NRZ and the Q output of the top DFF in figure 19.71. The output of this AND gate was the top input to the Increase XOR gate. This was done thinking that output of the AND would not transition to a 1 until a 1 was output from the DFF. The netlist is displayed below in figure 3.

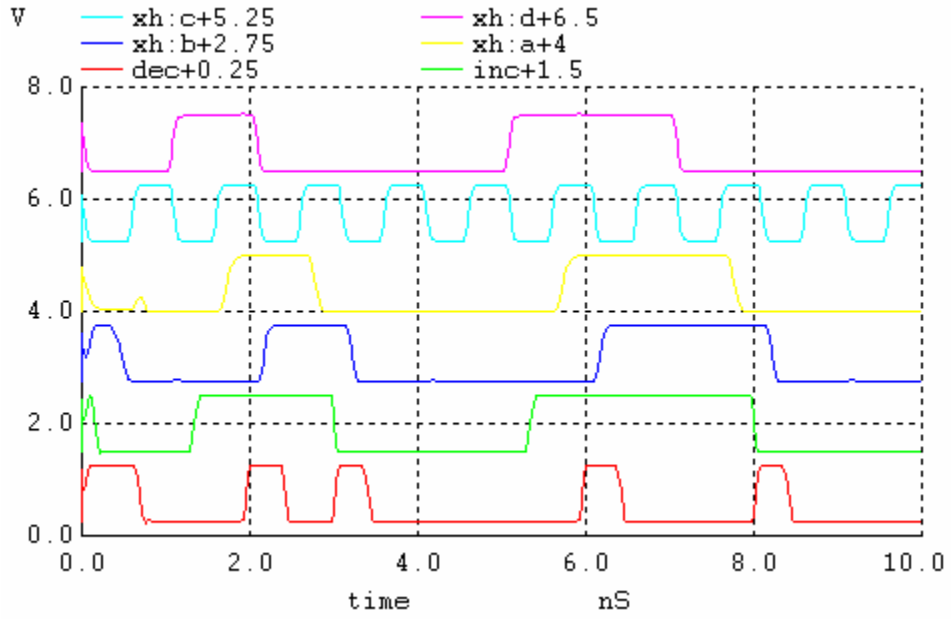


Figure 3

As can be seen in figure 3, this did not work either. Increase is still wider than decrease.Therefore, I was not able to find another method that worked to equalize the widths of increase and decrease.

```
*** Figure 19.70 with added DFF CMOS: Circuit Design, Layout, and Simulation ***

.control
destroy all
run
plot dec+0.25 inc+1.5 xh:b+2.75 xh:a+4 xh:c+5.25 xh:ddii+6.5 xh:d+7.75
.endc

.option scale=50n
.tran 100p 10n UIC

VDD     VDD     0       DC      1
Vnrz    NRZ     0       DC      0       PWL 0n 0 0.95n 0 1.05n 1 1.95n 1 2.05n 0 4.95n 0 5.05n 1 6.95n 1 7.05n 0
Vclock  clock   0       DC      0       PULSE 0 1 0.45n 0 0 0.4n 1n

Xh      VDD     NRZ     clock   inc     dec     hoggepd

.subckt hoggepd VDD     NRZ     clock   incb    decb

X1      VDD     NRZ     di      inverter
X2      VDD     di      d       inverter
X4      VDD     clock   ci      inverter
X5      VDD     ci      c       inverter

**DFF1_delay
M1d     n1d     d       VDD     VDD     PMOS L=1 W=20
M2d     n2d     ci      n1d     VDD     PMOS L=1 W=20
M3d     n2d     d       0       0       NMOS L=1 W=10
M4d     n3d     ci      VDD     VDD     PMOS L=1 W=20
M5d     n3d     n2d     n4d     0       NMOS L=1 W=10
M6d     n4d     ci      0       0       NMOS L=1 W=10
M7d     ddi     n3d     VDD     VDD     PMOS L=1 W=20
M8d     ddi     ci      n5d     0       NMOS L=1 W=10
M9d     n5d     n3d     0       0       NMOS L=1 W=10

X13     VDD     ddi     dd      inverter
X14     VDD     dd      ddii    inverter

**DFF_top
M1t     n1t     d       VDD     VDD     PMOS L=1 W=20
M2t     n2t     c       n1t     VDD     PMOS L=1 W=20
M3t     n2t     d       0       0       NMOS L=1 W=10
M4t     n3t     c       VDD     VDD     PMOS L=1 W=20
M5t     n3t     n2t     n4t     0       NMOS L=1 W=10
M6t     n4t     c       0       0       NMOS L=1 W=10
M7t     Ai      n3t     VDD     VDD     PMOS L=1 W=20
M8t     Ai      c       n5t     0       NMOS L=1 W=10
M9t     n5t     n3t     0       0       NMOS L=1 W=10

X3      VDD     Ai      A       inverter
X7      VDD     A       Aii     inverter


**DFF_bottom
M1b     n1b     A       VDD     VDD     PMOS L=1 W=20
M2b     n2b     ci      n1b     VDD     PMOS L=1 W=20
M3b     n2b     A       0       0       NMOS L=1 W=10
M4b     n3b     ci      VDD     VDD     PMOS L=1 W=20
M5b     n3b     n2b     n4b     0       NMOS L=1 W=10
M6b     n4b     ci      0       0       NMOS L=1 W=10
M7b     Bi      n3b     VDD     VDD     PMOS L=1 W=20
M8b     Bi      ci      n5b     0       NMOS L=1 W=10
M9b     n5b     n3b     0       0       NMOS L=1 W=10
```

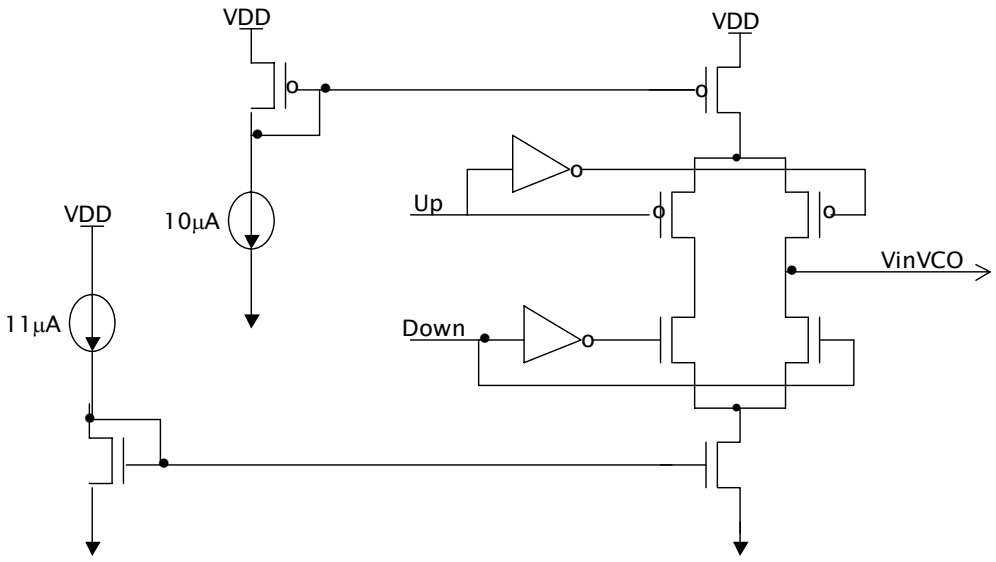
```
X6          VDD         Bi          B           inverter
X8          VDD         B           Bii         inverter

**xor_top
M1xt        n6t         ddii        VDD         VDD         PMOS L=1 W=20
M2xt        inc         dd          n6t         VDD         PMOS L=1 W=20
M3xt        inc         ddii        n7t         0           NMOS L=1 W=10
M4xt        n7t         A           0           0           NMOS L=1 W=10
M5xt        n6t         A           VDD         VDD         PMOS L=1 W=20
M6xt        inc         Aii         n6t         VDD         PMOS L=1 W=20
M7xt        inc         dd          n8t         0           NMOS L=1 W=10
M8xt        n8t         Aii         0           0           NMOS L=1 W=10

**xor_bottom
M1xb        n6b         B           VDD         VDD         PMOS L=1 W=20
M2xb        dec         Bii         n6b         VDD         PMOS L=1 W=20
M3xb        dec         B           n7b         0           NMOS L=1 W=10
M4xb        n7b         A           0           0           NMOS L=1 W=10
M5xb        n6b         A           VDD         VDD         PMOS L=1 W=20
M6xb        dec         Aii         n6b         VDD         PMOS L=1 W=20
M7xb        dec         Bii         n8b         0           NMOS L=1 W=10
M8xb        n8b         Aii         0           0           NMOS L=1 W=10

X9          VDD         inc         inci        inverter
X10         VDD         inci        incb        inverter
X11         VDD         dec         deci        inverter
X12         VDD         deci        decb        inverter

**X13       VDD         d           diii        inverter
**Cd        diii        0           100f
**X14       VDD         diii        dii         inverter

.ends

.subckt inverter        VDD         in          out
M1          out         in          0           0           NMOS    L=1     W=10
M2          out         in          VDD         VDD         PMOS    L=1     W=20
.ends

* BSIM4 models
```

Problem 19.30:

Do the currents in the charge pump in Fig. 19.73 have to be equal for proper DPLL clock-recovery operation? Why or why not? What happens if the currents aren't equal? What happens if the MOS switches turn on at different speeds than the PMOS switches? Use SPICE to support your answers.

Yes, the currents in the charge pump need to be equal for proper DPLL clock-recovery operation. Otherwise the amount of charge transferred to V_{inVCO} during each UP pulse is not equivalent to the amount of charge removed from V_{inVCO} during each DOWN pulse. The gain of the PFD with the charge pump will not be constant, since the gain = $I_{pump}/2\pi$ will depend on whether charge is being added or removed. As a result, the lock time will be longer, and the static phase error will be larger (unless the UP and DOWN pulse widths are adjusted to compensate for the difference in UP and DOWN current, so that the total amount of charge added and removed is constant).

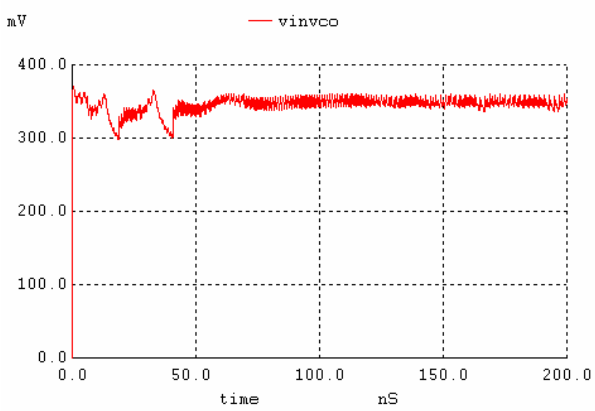
To demonstrate the longer lock time and static phase error in SPICE, the charge pump circuit was modified as shown below. The UP current was set at 10μA and the DOWN current set at 11μA.



The first set of plots shows the lock time and static phase error when both UP and DOWN currents are 10μA. For the sake of a fast simulation, the initial conditions of C_1 and C_2 were set at 300mV.

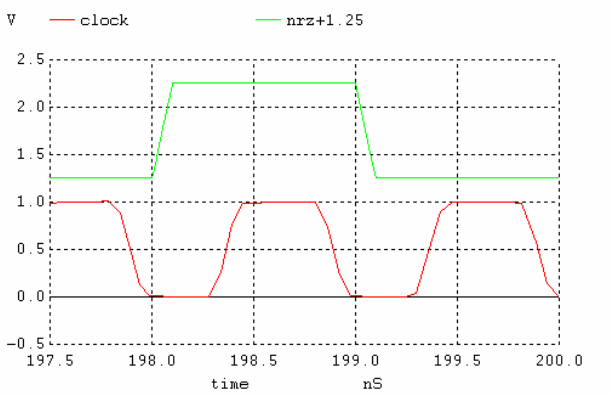
Plot of VCO control voltage

75ns time to lock



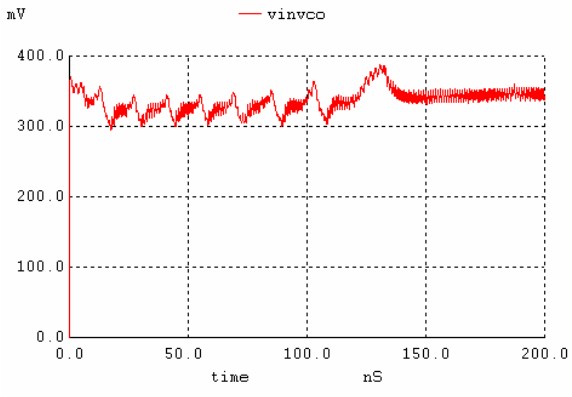
Plot of clock vs NRZ data

Static Phase Error is 180ps (early)

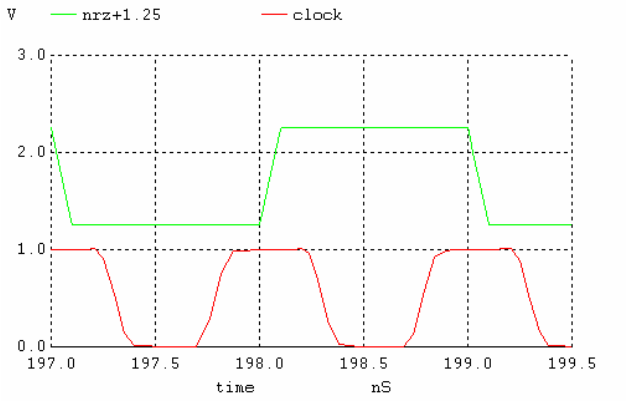


The next plots were taken with UP current = 10μA and DOWN current = 11μA. As can be seen the lock time and static phase error are worse.

~140ns lock time (almost 2 times worse)

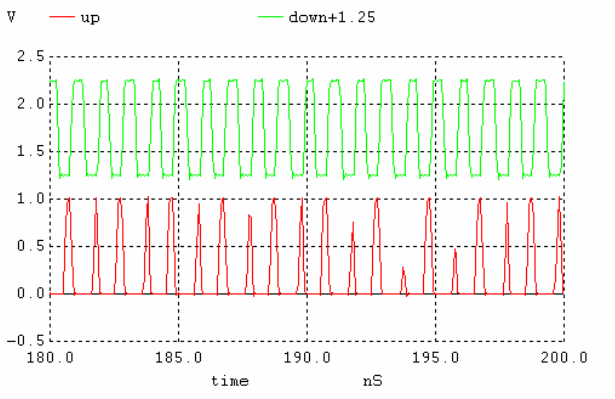


Static Phase Error is 230ps (late)

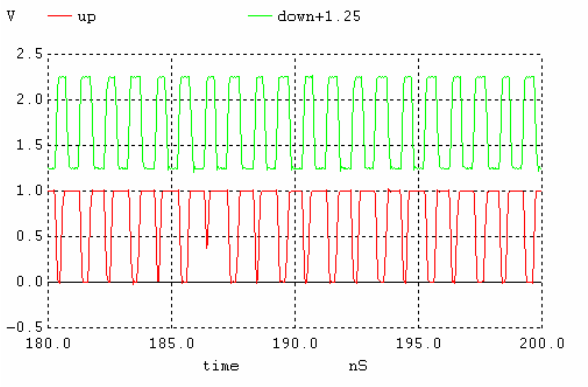


The next plots show the UP and DOWN signals for the two cases (equal and different UP/DOWN currents). For equal currents, UP and DOWN are enabled at opposite times. For different currents, UP is enabled for a longer period of time and overlaps the time when DOWN is enabled (pushing and pulling current at the same time, which is not very efficient).

UP = 10µA, DOWN=10µA:



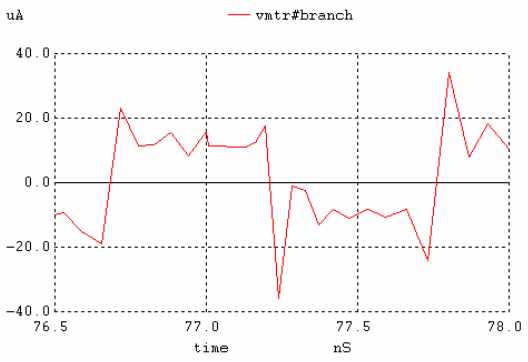
UP=10µA, DOWN=11µA:



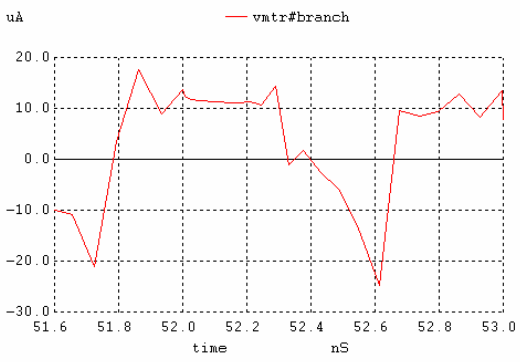
If the NMOS switches turn on at a different speed than the PMOS switches, there will also be a static phase error introduced. If the switches turn on at different speeds there will be a different amount of charge injected or removed from the V_{inVCO} node than intended. This is essentially the same problem as having different UP and DOWN currents.

The SPICE simulations below show the current added or removed from the V_{inVCO} node with 1) equal NMOS/PMOS switching speeds, and 2) slower NMOS switching speed (by adding inverters to the signal path). The total charge can be found by approximating the area under the curve.

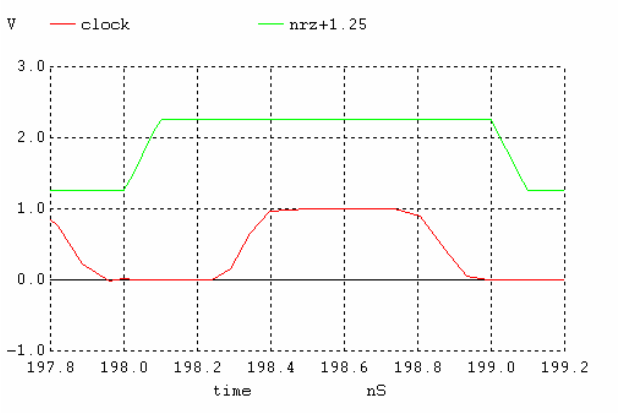
1. Equal switching speeds; approx. equal charge added and removed with each UP/DOWN pulse.



2. Slower NMOS switching speed. ~5fC added during UP pulse and ~2fC removed during DOWN pulse



The different amounts of charge added in each UP/DOWN cycle results in a static phase error of approximately 240ps, taken from the plot below, in comparison to the previous result of 180ps with equal switching times.



Problem 19.31

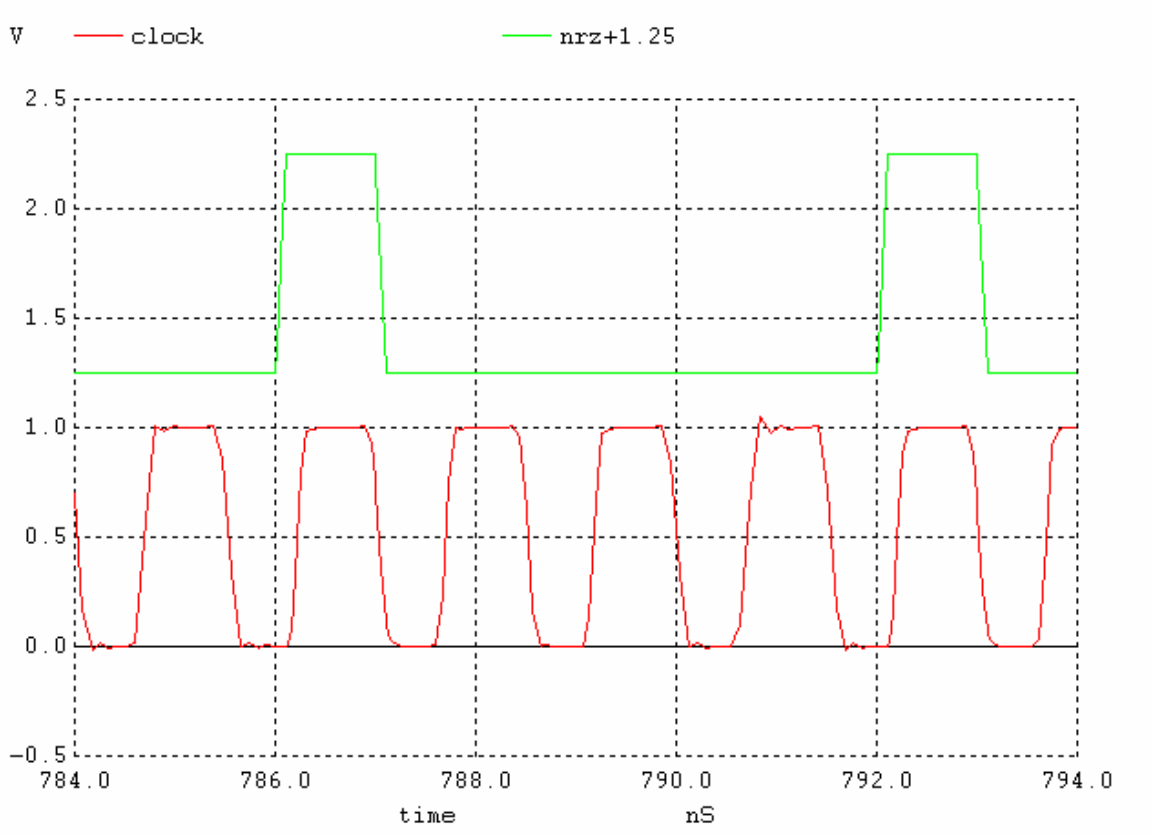
Tris Tanadi (ttanadi@hotmail.com)

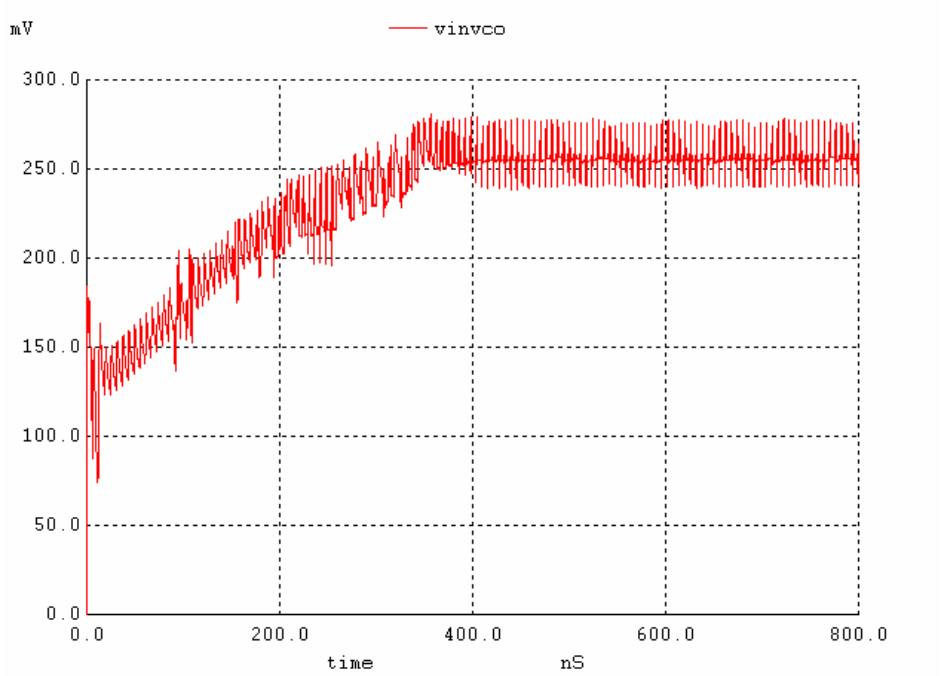
Modify the input signal of 1GHz clock recovery circuit to show false locking and comment on what can be done in a practical clock-recovery circuit to eliminate false locking.

Answer.

False locking is a phenomenon that the loop filter locks to the multiple or submultiples of the input frequency.

The input signal from Fig.19.64 is changed from NRZ to 4 zeros follow by a single one repetition signal. If the PLL is locking to the right frequency, the output clock signal will be 5 clock cycles between two consecutive inputs of one. From simulation, it shows that PLL locks at 680MHz instead of 1GHz.





Different method on how to eliminate/detecting false locking:

1. Reducing the gain of the VCO.
   This is basically limiting the VCO operating frequency to be around the input signal frequency. This kind of approach will require a priori knowledge of the input frequency.

2. Initialization of the PLL during start up.
   This approach is basically trying to lock or bring the VCO frequency to the input signal frequency by sending NRZ data during the start up of the circuit. Once this initialization is done the real data can be transmitted and if designed correctly the loop filter will still lock to the right frequency even with a random incoming data.

3. Detecting false lock with Bandpass
   A bandpass can be added to the output of the clock. Once the PLL is locked, the bandpass can be used to detect the clock frequency. By setting the bandpass frequency, it is possible to detect whether the clock frequency is within a certain range from the desired input frequency. If no tone is detected, then a false lock flag can be raised.

4. Using DLL to detect false lock.
   A DLL can be used to detect false lock. The delay through the DLL is designed to be around the input clock frequency. Once the PLL is locked, the clock signal will be compared with the clock signal that sees the DLL delay. If their edges are matched, it means no false lock is occurred.

Problem 19.32

Replace the charge pump used in the DPLL in Fig. 19.73 with the active-PI loop filter seen in Fig. 19.50. Calculate the loop filter component values. Using the new loop filter regenerate Figs. 19.74 and 19.75.

From Figures 19.74 and 19.75 we know the following information about the system response:

$$\omega_n = 100 \times 10^6 \ \frac{rad}{s}$$

$$K_{VCO} = 11 \times 10^9 \ \frac{rad}{V \cdot s}$$

$$K_{PD} = \frac{V_{DD}}{\pi}$$

$$\zeta = 1$$

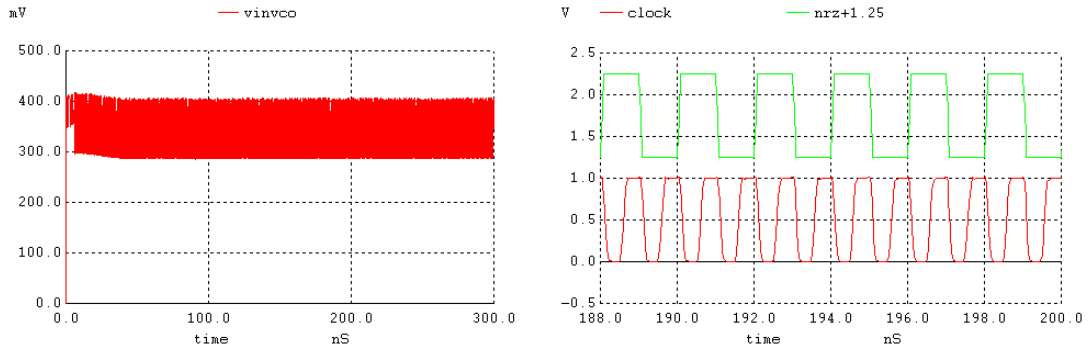
Now, using equations 19.42 through 19.45 and an aribtrary C of $1pF$ we can solve for R_1 and R_2:

$$\zeta = \frac{\omega_n R_2 C}{2} \Rightarrow R_2 = \frac{2\zeta}{\omega_n C} = 20k\Omega$$

$$\omega_n = \sqrt{\frac{K_{PD} K_{VCO}}{R_1 C}} \Rightarrow R_1 = \frac{K_{PD} K_{VCO}}{\omega_n^2 C} = 350k\Omega$$

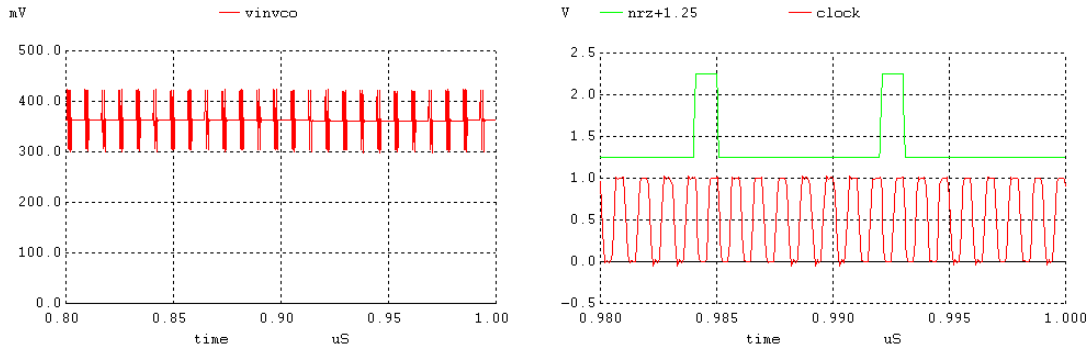
Hopefully, this careful selection of active-PI loop filter components will yield a result similar to that of Figures 19.74 and 19.75.

Figure 1: Reproduction of Figure 19.74



There is one important note concerning Figure 1. The initial condition on C was tuned to allow the DPLL to lock on the correct harmonic. Because the gain of the VCO was so large, this is a very important step.

Figure 2: Reproduction of Figure 19.75



The netlist for Figure 1 is seen below. The netlist for Figure 2 is the same as Figure 1, except the simulation is done for 1μs and the NRZ data is 00000001 not 01010101.

```
*** Problem 19.32a CMOS: Circuit Design, Layout, and Simulation ***

.control
destroy all
run
plot vinvco
plot up down+1.25 xlimit 190n 200n
plot clock nrz+1.25 xlimit 190n 200n
.endc

.option scale=50n
.tran 100p 300n UIC

VDD   VDD   0     DC    1
Vref  Vref  0     DC    0.5
Vnrz  NRZ   0     DC    0     PULSE 0 1 0 0 0 0.9n 2n

Xdline      VDD   vref  Vinvco      clocki      clock outp  outm  dline
Xpb3  VDD   outp  clocki      inverter
Xpb2  VDD   outm  clock inverter
Xh    VDD   NRZ   clock inc   dec   hoggepd
xactpi      inc   dec   vinvco      vdd   actpi

.subckt     actpi inc   dec   vinvco      VDD
Xinv  VDD   inc   inci  inverter
R1t   inci  vm    350k
R1b   dec   vm    350k
R2    vm    vrc   20k
Cf    Vinvco      vrc   1p    IC=-150m
Eopamp      Vinvco      0     Vp    vm    1e6
Vp    Vp    0     DC    0.5
.ends

.subckt     hoggepd     VDD   NRZ   clock incb  decb

X1    VDD   NRZ   di    inverter
X2    VDD   di    d     inverter
```

```
X4    VDD   clock ci    inverter
X5    VDD   ci    c     inverter

M1t   n1t   d     VDD   VDD   PMOS L=1 W=20
M2t   n2t   c     n1t   VDD   PMOS L=1 W=20
M3t   n2t   d     0     0     NMOS L=1 W=10
M4t   n3t   c     VDD   VDD   PMOS L=1 W=20
M5t   n3t   n2t   n4t   0     NMOS L=1 W=10
M6t   n4t   c     0     0     NMOS L=1 W=10
M7t   Ai    n3t   VDD   VDD   PMOS L=1 W=20
M8t   Ai    c     n5t   0     NMOS L=1 W=10
M9t   n5t   n3t   0     0     NMOS L=1 W=10

X3    VDD   Ai    A     inverter
X7    VDD   A     Aii   inverter

M1b   n1b   A     VDD   VDD   PMOS L=1 W=20
M2b   n2b   ci    n1b   VDD   PMOS L=1 W=20
M3b   n2b   A     0     0     NMOS L=1 W=10
M4b   n3b   ci    VDD   VDD   PMOS L=1 W=20
M5b   n3b   n2b   n4b   0     NMOS L=1 W=10
M6b   n4b   ci    0     0     NMOS L=1 W=10
M7b   Bi    n3b   VDD   VDD   PMOS L=1 W=20
M8b   Bi    ci    n5b   0     NMOS L=1 W=10
M9b   n5b   n3b   0     0     NMOS L=1 W=10

X6    VDD   Bi    B     inverter
X8    VDD   B     Bii   inverter

M1xt  n6t   dii   VDD   VDD   PMOS L=1 W=20
M2xt  inc   diii  n6t   VDD   PMOS L=1 W=20
M3xt  inc   dii   n7t   0     NMOS L=1 W=10
M4xt  n7t   A     0     0     NMOS L=1 W=10
M5xt  n6t   A     VDD   VDD   PMOS L=1 W=20
M6xt  inc   Aii   n6t   VDD   PMOS L=1 W=20
M7xt  inc   diii  n8t   0     NMOS L=1 W=10
M8xt  n8t   Aii   0     0     NMOS L=1 W=10

M1xb  n6b   B     VDD   VDD   PMOS L=1 W=20
M2xb  dec   Bii   n6b   VDD   PMOS L=1 W=20
M3xb  dec   B     n7b   0     NMOS L=1 W=10
M4xb  n7b   A     0     0     NMOS L=1 W=10
M5xb  n6b   A     VDD   VDD   PMOS L=1 W=20
M6xb  dec   Aii   n6b   VDD   PMOS L=1 W=20
M7xb  dec   Bii   n8b   0     NMOS L=1 W=10
M8xb  n8b   Aii   0     0     NMOS L=1 W=10

X9    VDD   inc   inci  inverter
X10   VDD   inci  incb  inverter
X11   VDD   dec   deci  inverter
X12   VDD   deci  decb  inverter

X13   VDD   d     diii  inverter
cd    diii  0     100f
X14   VDD   diii  dii   inverter

.ends
```

```
.subckt dline     VDD   vref  vindel      inp   inm   outp  outm

Xbias VDD   vref  vpbias      vrbias      vindel      dbias
X1    VDD   vpbias      vrbias      inp   inm   o1p   o1m   delay
X2    VDD   vpbias      vrbias      o1p   o1m   o2p   o2m   delay
X3    VDD   vpbias      vrbias      o2p   o2m   o3p   o3m   delay
X4    VDD   vpbias      vrbias      o3p   o3m   o4p   o4m   delay
X5    VDD   vpbias      vrbias      o4p   o4m   o5p   o5m   delay
X6    VDD   vpbias      vrbias      o5p   o5m   o6p   o6m   delay
X7    VDD   vpbias      vrbias      o6p   o6m   o7p   o7m   delay
X8t   VDD   vpbias            o7p   o7m   outp        delay_last
X8i   VDD   vpbias            o7m   o7p   outm        delay_last
.ends

.subckt     delay VDD   vpbias      vrbias      vp    vm    vop   vom
M1    n1    vpbias      VDD   VDD   PMOS L=1 W=200
M2    vom   vp    n1    VDD   PMOS L=1 W=20
M3    vop   vm    n1    VDD   PMOS L=1 W=20
M4    vop   vrbias      0     0     NMOS L=1 W=10
M5    vop   vop   0     0     NMOS L=2 W=10
M6    vom   vrbias      0     0     NMOS L=1 W=10
M7    vom   vom   0     0     NMOS L=2 W=10
.ends

.subckt     delay_last  VDD   vpbias      vp    vm    vop
M1    n1    vpbias      VDD   VDD   PMOS L=1 W=200
M2    vom   vp    n1    VDD   PMOS L=1 W=20
M3    vop   vm    n1    VDD   PMOS L=1 W=20
M4    vop   vom   0     0     NMOS L=1 W=10
M6    vom   vom   0     0     NMOS L=1 W=10
.ends

.subckt dbias     VDD   vref  vpbias      vrbias      vindel
M1    vpbias      vindel      vr    0     NMOS L=1 W=100
M2    vpbias      vpbias      VDD   VDD   PMOS L=1 W=20
M3    n1    vpbias      VDD   VDD   PMOS L=1 W=200
M4    n2    0     n1    VDD   PMOS L=1 W=20
M5    n2    vrbias      0     0     NMOS L=1 W=10
M6    n2    n2    0     0     NMOS L=2 W=10
Rr    vr    0     10k
X1    VDD   n2    vref  vrbias      pdiff
.ends

.subckt     pdiff VDD   Vp    Vm    vout
M1    n1    vb    VDD   VDD   PMOS L=1 W=20
M2    vb    vp    n1    VDD   PMOS L=1 W=20
M3    vout  vm    n1    VDD   PMOS L=1 W=20
M4    vb    vb    0     0     NMOS L=1 W=10
M5    vout  vb    0     0     NMOS L=1 W=10
.ends


.subckt inverter  VDD   in    out
M1    out   in    0     0     NMOS  L=1   W=10
M2    out   in    VDD   VDD   PMOS  L=1   W=20
.ends
```