RF Systems Course: PLL Lecture II

Fractional-N Frequency Synthesis

- Frac N aims to achieve fine step size, while still using a high reference freq.
- This is accomplished by constantly changing the divide ratio N between two or more numbers.
- Higher reference freq is good for noise and settling time.
- E.g. can toggle between 8 and 9.
- divide by 8 nine times in a row and then divide by 9 once, average division ratio is:

$$\overline{N} = \frac{\text{Total Divider Input Pulses Needed}}{\text{Total Divider Output Pulses Generated}} = \frac{8\times 9 + 9\times 1}{10} = 8.1$$

- if it only divided by 8 it would take 80 input pulse to produce 10 output pulses.
- Now it takes 81 input pulses.
- Sometimes we say it swallowed an extra pulse.
- We make use of low pass nature of loop filter to “smooth out” this choppy output from divider.
- this simple example will produce tones at distinct freqs in output.
- Not desirable so we try and make this random
- Same average division ratio, but as much as possible randomly switch between division ratios.

Example:

- Determine max ref freq of int-N synth required to cover channels from 2400 to 2499MHz, spaced 3MHz apart, and channels from 5100 to 5200MHz spaced 4MHz apart.

- If int N designed to service one of these bands then could have max ref freq of 3MHz in the first case, 4MHz in the second case.
- If needs to cover both bands max ref freq is 1MHz.

Frac-N with Dual Modulus Prescaler.

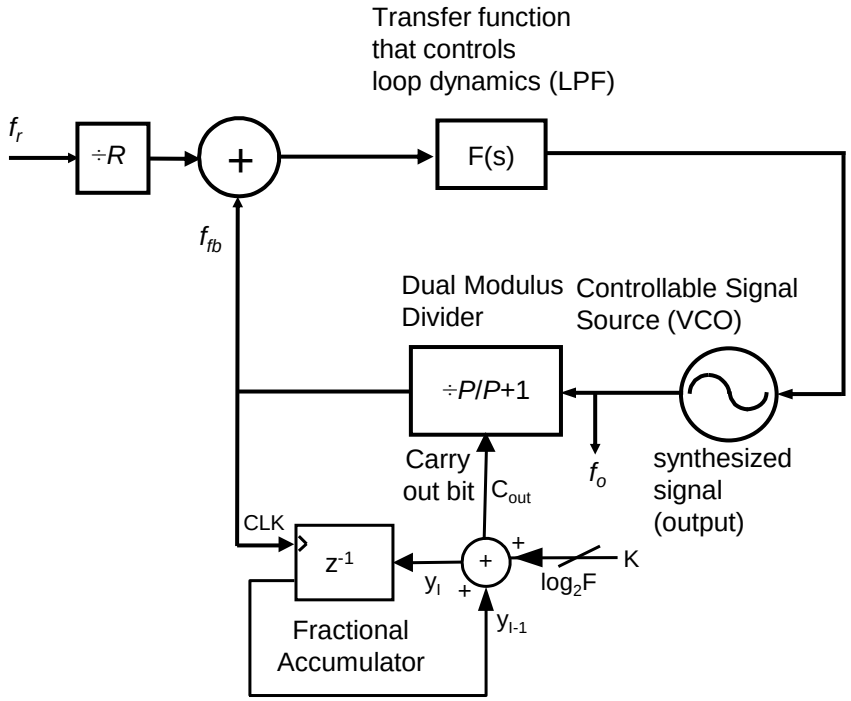
- This is a simple way to implement a frac N
- Divider is toggled between P and P+1.
- C_{out} generated using an accumulator with size of F

$$y_i = (y_{i-1} + K_i) \bmod F$$

- Simple accumulator simulation:
- Describe operation of 3-bit accumulator with K=1,2
- Size of accumulator is 2^3= 8 or F =8, y_{max} = 7 = 111
- After reaches max value will overflow and C_{out} =1
- K=1 will take 8 clock cycles for overflow.
- If K=2 will take 4 clock cycles for overflow.

Table 6.1 Accumulator Operations with $F = 8$ and $K = 1$

Clock Cycle i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
y_i	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2
y_{i-1}	NA	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1
C_{out}	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0



Frac-N with Dual Modulus Prescaler.

- So in that example low for 7 cylcles, high for 1 gives a freq of f_{clk}/F.
- If K = 3 would be high for 3 cycles low for 5 freq of $3f_{clk}/8$ so in general:

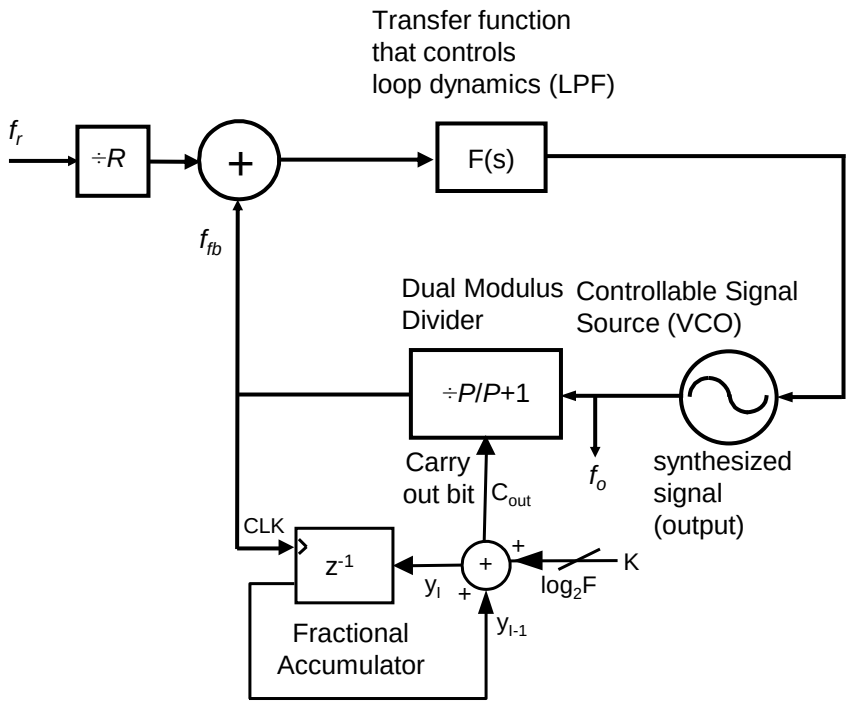
$$f_{C_{\text{out}}} = \frac{Kf_{clk}}{F}$$

- So VCO output freq is:

$$f_o = \frac{f_r}{R}\left[\frac{(P+1)K + P(F-K)}{F}\right] = \frac{f_r}{R}\left[P + \frac{K}{F}\right]$$

- So step size is

$$\text{Step Size} = \frac{f_r}{RF}$$



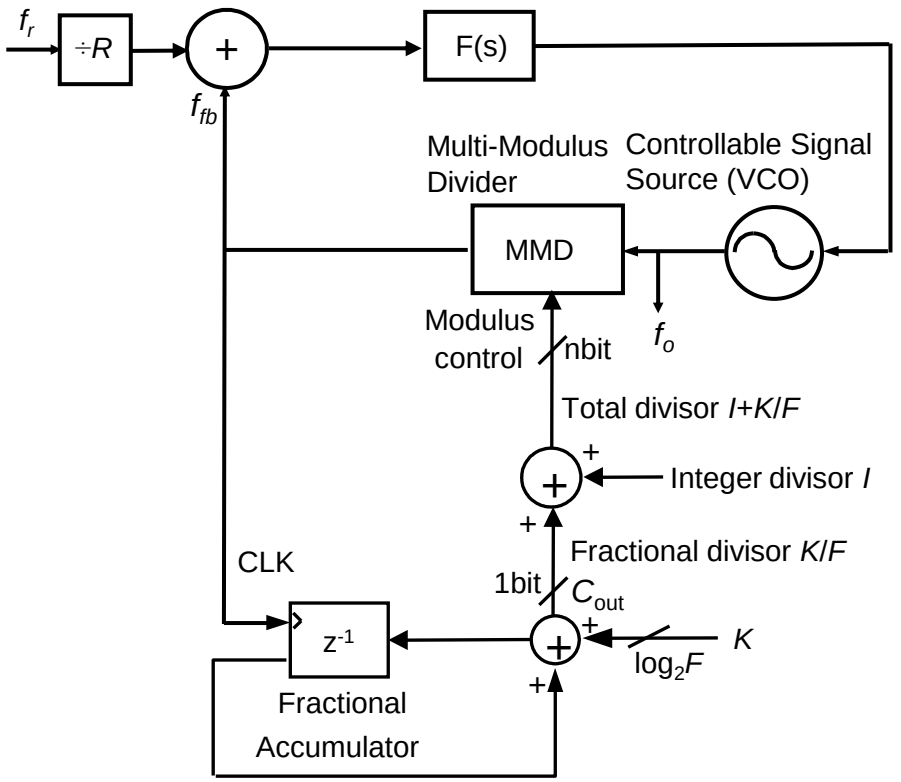
Frac N Synthesizer with Multimodulus Divider

- With a divider that can divide by many different values

$$f_o = \frac{f_r}{R}\left[I + \frac{K}{F}\right]$$

- Note commonly called a multi-modulus divider.

Now can move I which is the integer divide ratio as well as the fractional ratio.

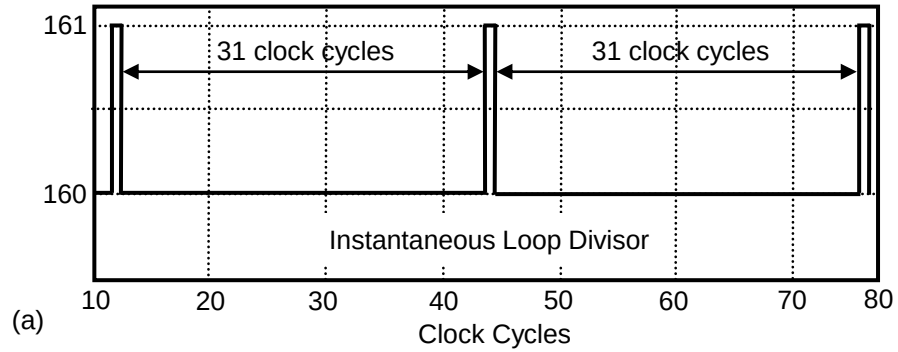


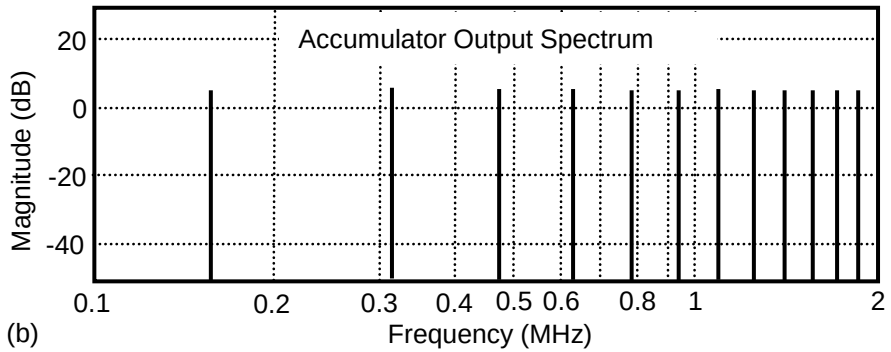
Frac N Synthesizer Example

- Design frac N synth with 11 channels from 819.2MHz to 820.96MHz.
- Step size = 160kHz, f_r/R = 5.12MHz.
- Determine F: $F = \frac{f_r}{R} \cdot \frac{1}{160kHz} = 32$
- This is a 5 bit accumulator.
- K can be set from 0-10 to conver all channels.
- Int division can be found next:

$$\frac{f_r}{R}\left(I + \frac{0}{F}\right) = \frac{f_r}{R} I = 819.2MHz$$

- Therefore I = 160.
- Total division is N = 160 + K/32.





Note this is the reason in practice we use $\Sigma\Delta$ modulators to randomize noise!

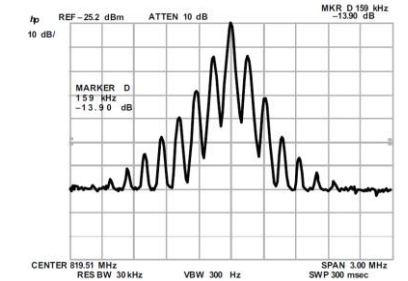
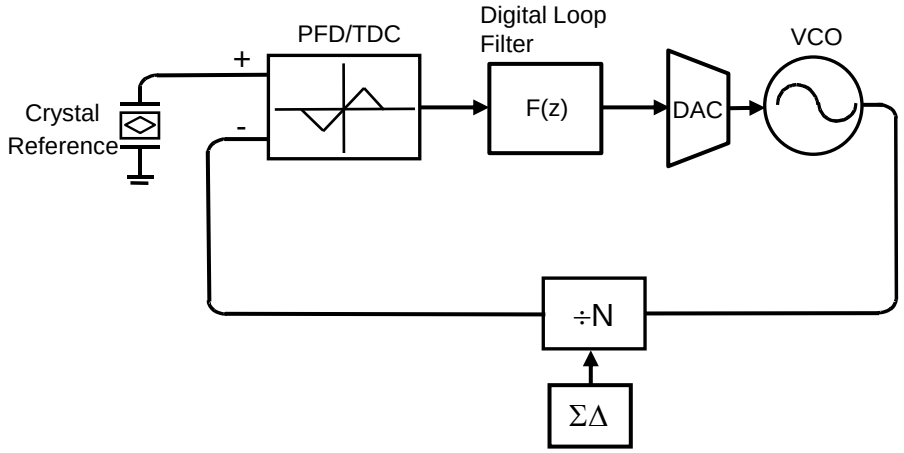


Figure 6.33 Measured output spectrum of a fractional-*N* frequency synthesizer with loop divisor $N = 160 + 1/32$ and the comparison frequency $f_r/R = 5.12$ MHz.

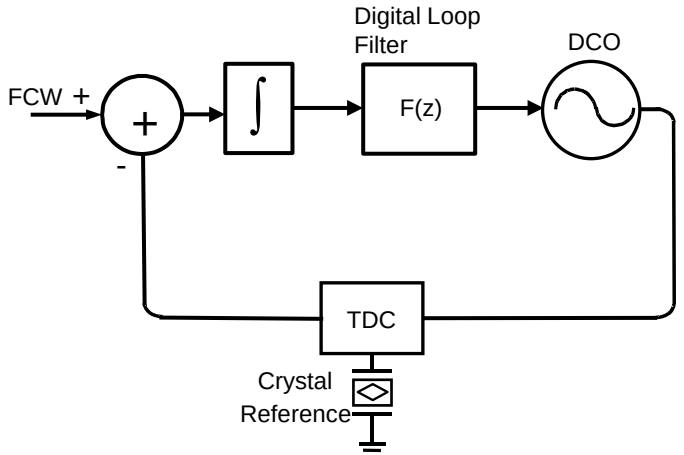
All-Digital Phase Locked Loops

- Modern CMOS technologies often make analog circuits like CPs hard to implement so there is a desire to move to more digital architectures.
- Also loop filter is often off chip.
- As a first step we can use a digital loop filter, which means we now need a DAC to convert a digital number back into a voltage for VCO.



All-Digital Phase Locked Loops

- Moving a fully digital structure.
- VCO is now a DCO which is directly controlled by bits not a voltage.
- Divider is replaced by a time-to-digital converter which compares DCO to reference and puts out a digital number.
- Input is now a freq controled digital word which is compared to feedback signal.
- Integrator is necessary to get phase rather than frequency.



Noise in All-Digital Phase Locked Loops

- TDC is a digital circuit and therefore has finite resolution and therefore digital quantization noise:

$$T_{error}{}^2 = \frac{T_{res}{}^2}{12}$$

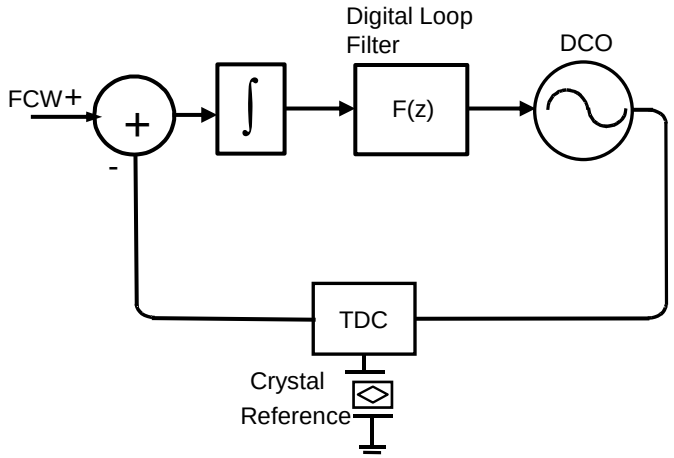
- time resolution directly causes a phase error:

$$\phi_n = 2\pi \frac{T_{error}}{T_{DCO}}$$

- Noise will be spread from DC to f_{ref} so:

$$L = \left(2\pi \frac{T_{error}}{T_{DCO}}\right)^2 \frac{1}{F_{ref}} = \frac{(2\pi)^2}{12} \cdot \frac{T_{res}{}^2 F_{DCO}{}^2}{F_{ref}}$$

$$L(s) = 10\log\left[\frac{(2\pi)^2}{12} \cdot \frac{T_{res}{}^2 F_{DCO}{}^2}{F_{ref}}\right]$$

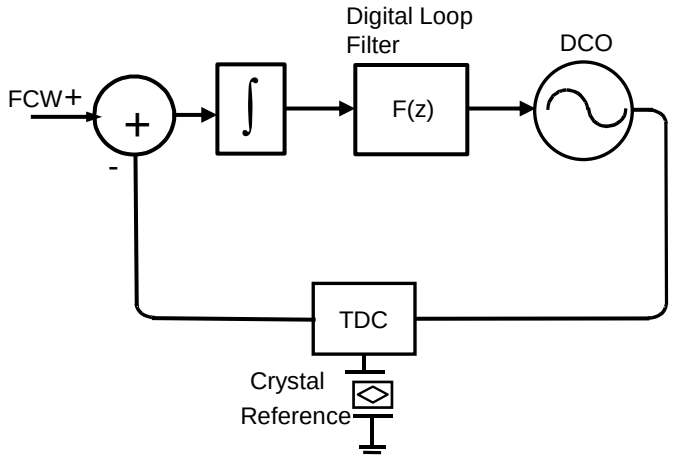


- E.g.: ADPLL designed to have DCO output freq of 5GHz, XTAL = 40MHz
- TDC has a resolution of 20ps so:
- $L(s) = -90.8dBc/Hz$
- Note does not take into account any other noise so this is a best case.

Noise in All-Digital Phase Locked Loops

- The DCO is a digital circuit and therefore has a finite resolution and therefore digital quantization noise:

$$f_{error}{}^2 = \frac{f_{res}{}^2}{12}$$



- The freq resolution causes a phase error:

$$\phi_n = f_{error} T_{ref}$$

- Noise will be spread from DC to fref so:

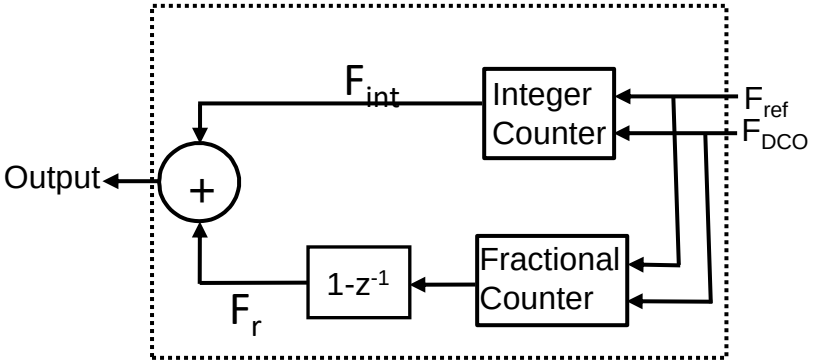
$$L = \left(f_{error} T_{ref}\right)^2 \cdot \frac{1}{F_{ref}} = \frac{1}{12} \cdot \frac{f_{res}{}^2}{F_{ref}{}^3}$$

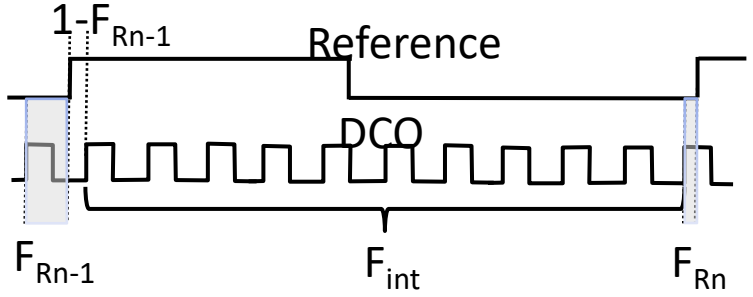
$$L(s) = 10\log\left[\frac{1}{12} \cdot \frac{f_{res}{}^2}{F_{ref}{}^3}\right]$$

Time-to-Digital Converter Architecture

- TDC needs to demine how many DCO cycles happen between reference edges.
- TDC has an integer counter and a fractional counter for increased resolution.
- need for the differentiator becomes clear looking at the figure and working out non-integer part of the result:

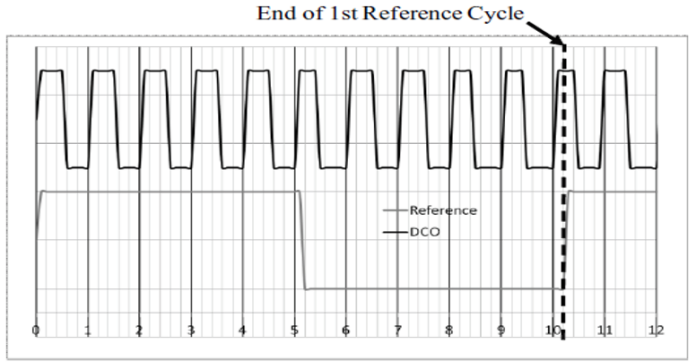
$$F_r = F_{Rn} + 1 - F_{Rn-1} = 1 + F_{Rn} - F_{Rn-1} = 1 + \left(1 - z^{-1}\right)F_{Rn} = \left(1 - z^{-1}\right)F_{Rn}$$





Why Do We Need a Fractional Count?

- ADPLL set to FCW = 10.2 with only int count.
- Assume ref and DCO are in phase at the start of the experiment.
- After 1st cycle of the ref, 10.2 cycles of the DCO have passed
- Int counter counts 10 so error is 0.2
- End of 2nd cycle 20.4 DCO cycles have passed, error is 0.4.
- After 5th cycle captures 11 DCO edges, then cycle repeats.



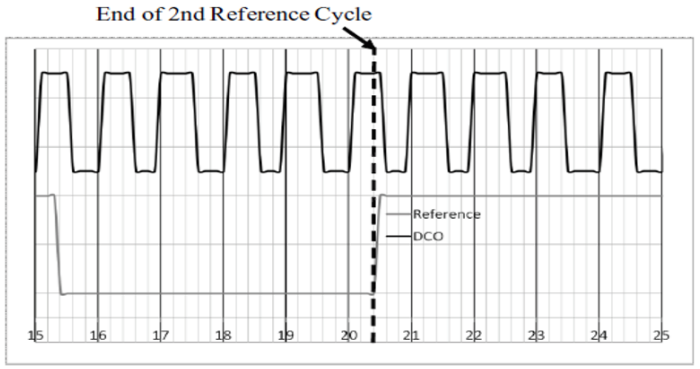


Table 6.3 Results of Using an Integer Counter for Five Reference Cycles to Produce a Ratio of 10.2

Reference Cycle	*Integer Count*	*Error*	*DCO Cycles Since Start*
1	10	0.2	10.2
2	10	0.4	20.4
3	10	0.6	30.6
4	10	0.8	40.8
5	11	0.0	51
6	10	0.2	61.2

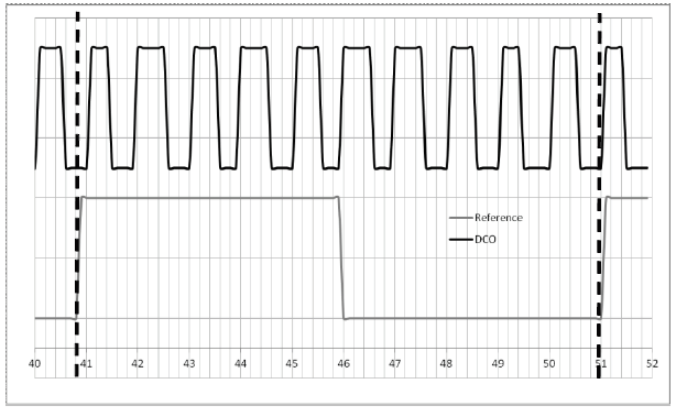


Figure 6.39 The fifth cycle of a DCO running 10.2 times faster than the reference. In this cycle 11 rather than 10 rising DCO references are caught between the two reference edges.

- Value of the error repeats every 5 cycles.
- Due to the fact that the fraction in this case is 1/5.
- With fractional count can make this error much smaller.

The Digital Loop Filter

- Previously we had analog loop filter with TF:

$$F(s) = R + \frac{1}{sC_1}$$

- This can be shown in block diagram form.
- This can be scaled to include Tref:

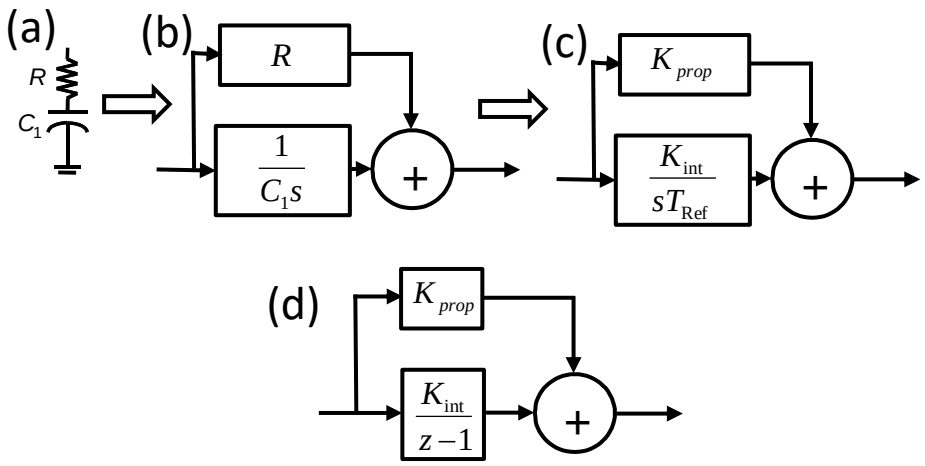
$$F(s) = K_{prop} + \frac{K_{\text{int}}}{sT_{ref}}$$

- Converting this s-domain equation to the z-domain gives:

$$F(z) = K_{prop} + K_{\text{int}}\left(\frac{1}{z-1}\right)$$

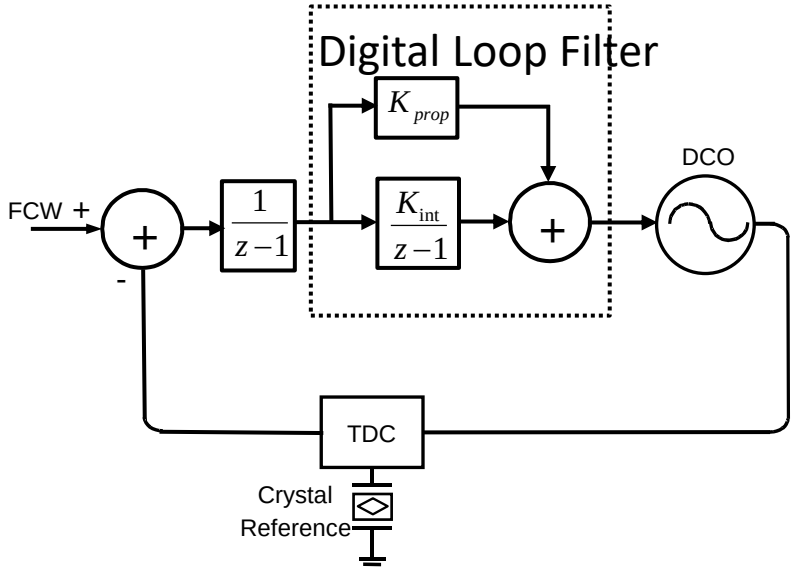
$$K_{prop} = R$$

$$K_{\text{int}} = \frac{T_{ref}}{C_1}$$



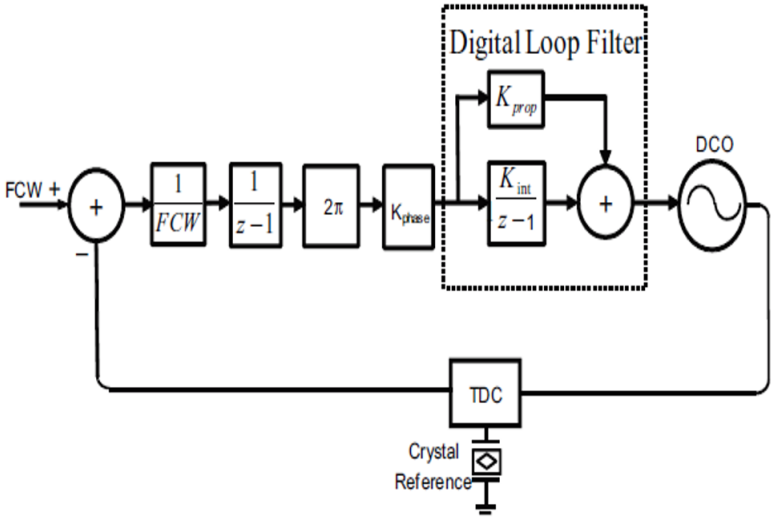
Final ADPLL with loop filter

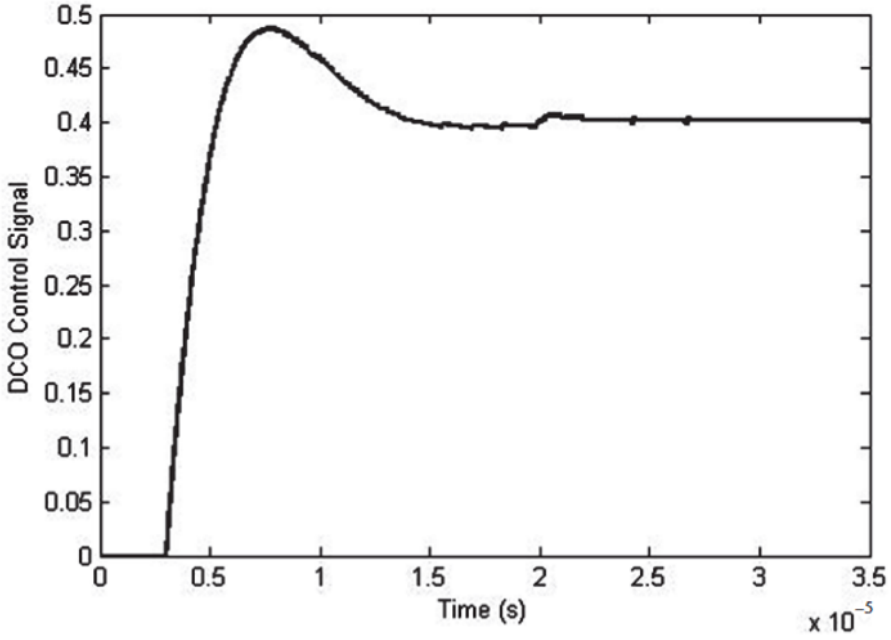
- The final ADPLL with the loop filter is shown here.



A Simple ADPLL Design

- We will take example from last lecture and make it digital.
- Output freq was 4GHz, ref was 40MHz.
- C_1 = 5.66nF, R = 530Ω, K_{VCO} = 200MHz/V, K_{phase} = 100μA/rad.
- Now make K_{prop} = R = 530, $K_{int} = T_{ref}/C_1$ = 4.42.
- To account for gain of the divider add a gain block of value 1/FCW, output of the integrator multiplied by 2π to convert result to rads. Then same value of K_{phase} can be used as previous example.
- Sensitivity of the DCO will be 200MHz/unit





- To verify loop works correctly run simulation where FCW is stepped from 100 to 102.
- Very similar to previous example.

A Simple ADPLL Design

- Also interesting to look where FCW isn't an int. e.g. 100.5, and 100.2
- For 100.5 get a square wave with a freq of 20MHz.
- With 100.2 get a 80% duty cycle square wave with freq of 8MHz.
- Obviously this causes a lot of spurs as shown!

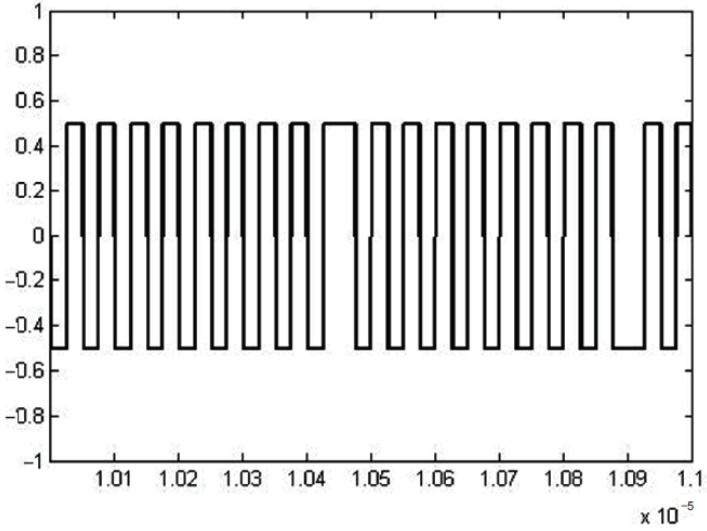


Figure 6.44 ADPLL error signal with FCW = 100.5.

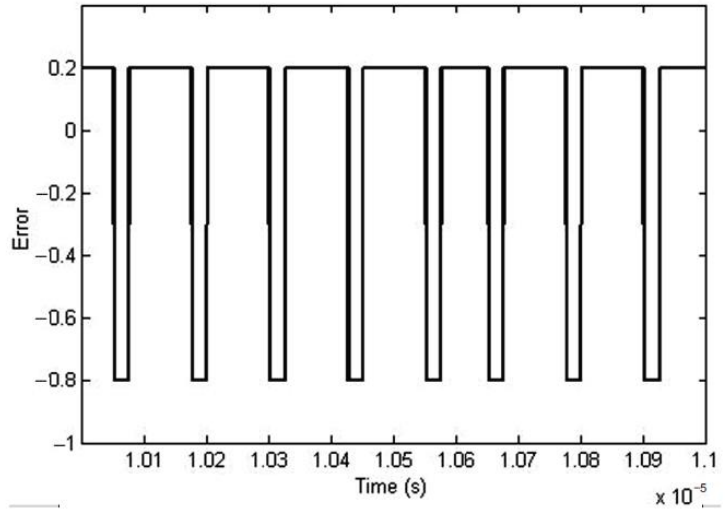


Figure 6.45 ADPLL error signal with FCW = 100.2.

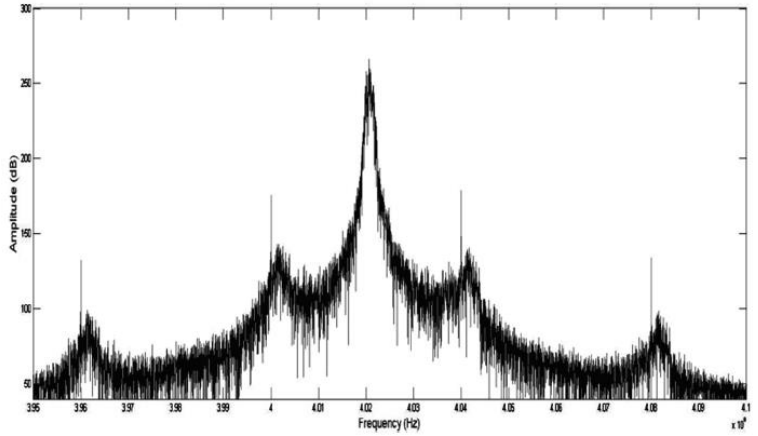
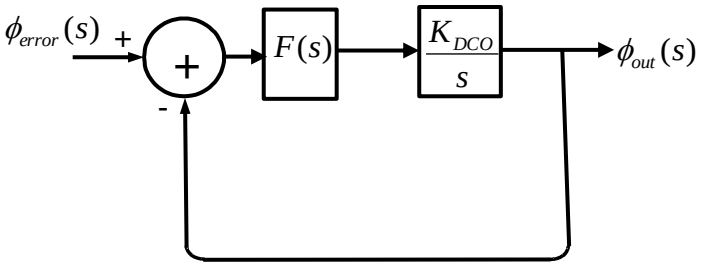


Figure 6.46 An FFT of the ADPLL DCO output with FCW = 100.5.

ADPLL Noise Calculations

- Provided that f < 1/10fref previous theory holds for doing noise TFs.



TDC Circuits