Ultra-Low Power PLL for Wake-up Receiver Applications

Specialization Project Progress - 7th Week

Cole Nielsen

Department of Electronic Systems, NTNU

11 October 2019 (Calendar week 41)

Autumn Timeline

Week	Dates	Tasks	Outcomes
36	2.9 - 8.9	Review PLL Design	Refreshed Knowledge
37	9.9 - 15.9	Modeling/simulation (set up)	–
38	16.9 - 22.9	Modeling/simulation	TDC/DCO Requirements
39	23.9 - 29.9	Modeling/simulation	Loop Filter/Digital Algorithms
40	30.9 - 6.10	Modeling/simulation	Loop filter, DCO, TDC, calibration
41	7.10 - 13.10	Circuit Research	DCO/Divider topologies, Ideal Virtuoso implementation
42	14.10 - 20.10	Circuit Research	TDC/other topologies
43	21.10 - 27.10	Circuit Implementation	Digital logic (schematic)
44	28.10 - 3.11	Circuit Implementation	DCO (schematic)
45	4.11 - 10.11	Circuit Implementation	Divider/other (schematic)
46	11.11 - 17.11	Circuit Implementation (TDC)	
47	18.11 - 24.11	Circuit Implementation (TDC)	TDC (schematic)
48	25.11 - 1.12	Full Circuit testing	Testbenches, find bugs, design fixes
49	2.12 - 8.12	Full Circuit testing	Design Fixes/iteration
50	9.12 - 15.12	–	–

Legend: **Done** **Current** **Revised**

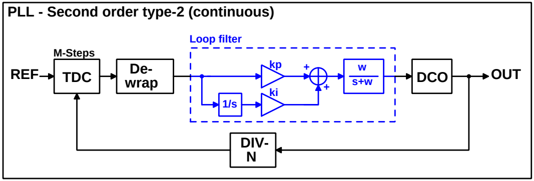
Timeline Tasks

This week

- **Simulation of PLL**
  - Finalized design of loop filter.
  - Made script to compute filter coefficients for difference equation.
  - Made new Python PLL simulation, completely in phase domain.
  - Wrote Verilog module that implements loop filter.
  - Working on schematic-based implementation of ideal PLL in Virtuoso...
- **DCO/Divider topologies**:
  - Considered LC versus ring oscillator.
    - Power limitations
    - Start up time, tuning
  - Divider topologies
    - Synchronous counter (replaces divider and TDC)
    - 2^N to 2^{N-1} multimodulus with N chained 2/3 dual-modulus dividers.
    - TSPC FF's to save power.

PLL Simulation

Final loop filter

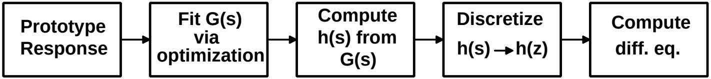


- Proportional-integral with added pole. Equal to second order type-2 (i.e. charge pump) PLL. (PI is chosen for no tracking error).
- Additional pole yields more favorable oscillator and TDC noise characteristics.
- **Not** second order, really three poles and one zero (open loop A(f) shown below).
  - Computationally optimize loop filter transfer function so closed loop response approximates desired second order transfer function.

$$A(s) = \frac{K}{s^2} \cdot \frac{(s/\omega_z + 1)}{(s/\omega_p + 1)} \tag{1}$$

PLL Simulation

Computation of discrete filter.



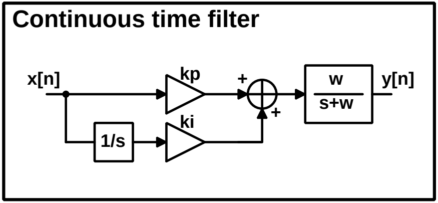
— **Goal:** Convert prototype second order response as defined by ω_n, ζ and translate to a difference equation implementable in digital logic.

- PLL closed loop response G(s) is fit to the prototype response using gradient descent optimization of the parameters of A(s):
- PLL open loop response A(s) is computed from G(s). The loop filter response h(s) is comuted from A(s)
- h(f) is discretized as h(z), h(z) is translated into a difference equation y[n].
- Difference equation is implemented as multipliers, adders and registers.

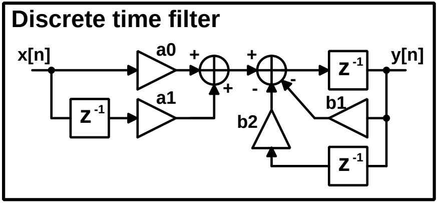
$$G(s) = \frac{A(s)}{1 + A(s)}, \quad A(s) = \frac{M}{N}\frac{K_{DCO}}{s}h(s), \quad h(s) = \frac{k_i}{s}\frac{s/\omega_z + 1}{s/\omega_p + 1}, \quad \omega_z = \frac{k_i}{k_p} \qquad (2)$$

PLL Simulation

Discrete time implementation.



Continuous time filter



Discrete time filter

$$
\begin{aligned}
y[n] = x[n]\frac{K_i\omega_p T}{\omega_z}\,\frac{1+\omega_z T}{1+\omega_p T} - x[n-1]\frac{K_i\omega_p T}{\omega_z}\,\frac{1}{1+\omega_p T} + y[n-1]\frac{2+\omega_p T}{1+\omega_p T} - y[n-2]\frac{1}{1+\omega_p T} \\
= a_0 x[n] + a_1 x[n-1] - b_1 y[n-1] - b_2 x[n-2] \qquad (3)
\end{aligned}
$$

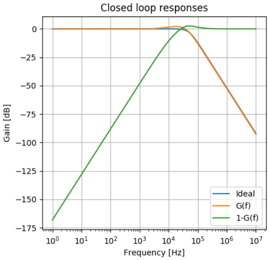
— Implementation requires:

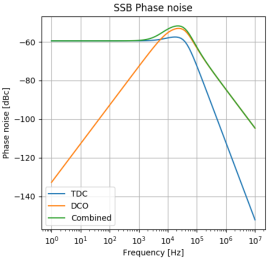
- **3 registers** (delay elements)
- **4 multipliers**
- **3 adders**

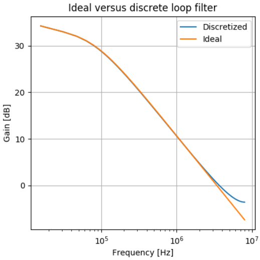
PLL Simulation

Example filter.

- Using the described method, a filter was fitted to a 50 kHz bandwidth and $\zeta \approx 0.7$, for a 2.4 GHz PLL with 16 MHz reference and 64 TDC steps.
- Fitted response is close to protoype, but with a benefit:
  - DCO noise is attenuated by 40 dB/decade below loop bandwidth (versus 20 for prototype filter)



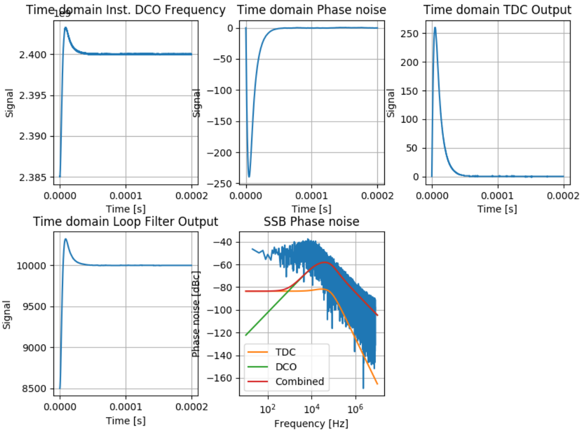




PLL Simulation

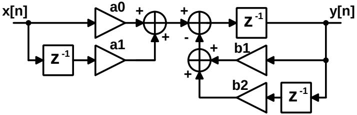
Python simulation results.

- Modified Python simulation to be 100% phase domain, for better phase noise resolution.
- Loop filter design successful - architecture achieves lock and rejects phase noise.
- There is a error in my filter design math/code, PLL bandwidth in simulation is 5x smaller than intended.



PLL Simulation

Loop filter Logic Implementation.



— Translation to hardware:

- Delay blocks are implemented as registers (3 total)
- Gain blocks are implemented as multipliers (4 total)
- Sums are adders (3 total)

— Decide bits of resolution for datapath on TDC/DCO resolution and gain coefficients.

— Will re-evaluate resolution based on power consumption after synthesis/place+route.

PLL Simulation

Resolution of datapath.

- Will use signed integers representing fixed point values.
- TDC resolution (N_{TDC}) defines required number of integer bits on input.
- DCO resolution (N_{DCO}) defines required number of integer bits on output.
- Fractional bits needed is (for gains, but applied to all):

$$\text{Fractional bits} > \lceil -\log_2(a_0 + a_1) \rceil \tag{4}$$

- Integer bits needed for gain coefficients:

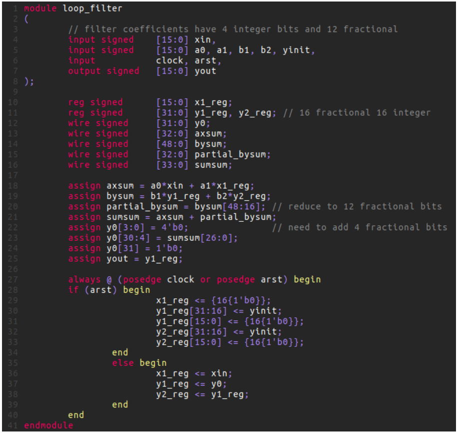
$$\text{Integer bits} > \lceil 1 - \log_2(\max(|\{a_0, a_1, b_1, b_0\}|)) \rceil \tag{5}$$

- It is expected that the minimum *required* resolutions will then be:
  - Input : 6 integer bits, 0 fractional
  - Gains : 2 integer bits, 9 fractional
  - Output : 10 integer bits, 0 fractional
- (The filter coefficients a_0, a_1, b_1, b_0 are on the order of 1)

PLL Simulation

Verilog implementation.

- Currently more bits than needed. Will make more efficient implementation later.
- Use fixed point with 4 integer bits + 12 fractional bits for filter coefficients.
- Input and output are 16 bit signed integers.
- Will multiplication/addition here be handled in synthesis, or do I need to do this?

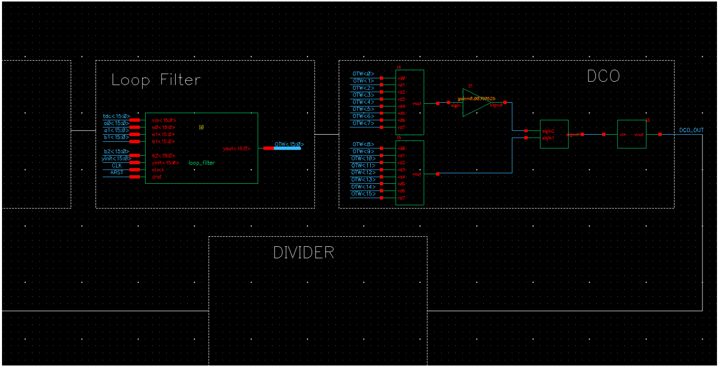


```verilog
module loop_filter
(
        // filter coefficients have 4 integer bits and 12 fractional
        input signed    [15:0] xin,
        input signed    [15:0] a0, a1, b1, b2, yinit,
        input                  clock, arst,
        output signed   [15:0] yout
);

        reg signed      [15:0] x1_reg;
        reg signed      [31:0] y1_reg, y2_reg; // 16 fractional 16 integer
        wire signed     [31:0] y0;
        wire signed     [32:0] axsum;
        wire signed     [48:0] bysum;
        wire signed     [32:0] partial_bysum;
        wire signed     [33:0] sumsum;

        assign axsum = a0*xin + a1*x1_reg;
        assign bysum = b1*y1_reg + b2*y2_reg;
        assign partial_bysum = bysum[48:16]; // reduce to 12 fractional bits
        assign sumsum = axsum + partial_bysum;
        assign y0[3:0] = 4'b0;               // need to add 4 fractional bits
        assign y0[30:4] = sumsum[26:0];
        assign y0[31] = 1'b0;
        assign yout = y1_reg;

        always @ (posedge clock or posedge arst) begin
        if (arst) begin
                        x1_reg <= {16{1'b0}};
                        y1_reg[31:16] <= yinit;
                        y1_reg[15:0] <= {16{1'b0}};
                        y2_reg[31:16] <= yinit;
                        y2_reg[15:0] <= {16{1'b0}};
                end
                else begin
                        x1_reg <= xin;
                        y1_reg <= y0;
                        y2_reg <= y1_reg;
                end
        end
endmodule
```

PLL Simulation

Ideal PLL - Virtuoso schematic.

- Have loop filter (imported from Verilog module) and DCO implemented.
- Need to implement divider/TDC in Verilog (couldn't be easily done with ahdlLib).
  - Wasted quite a bit of time re-learning Verilog, so ran out of time for this.



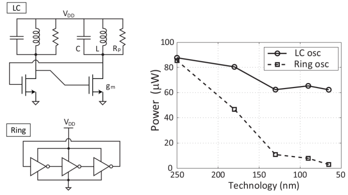
DCO topologies

LC Oscillator versus Ring Oscillator.

— **LC Oscillator**.

- Inductor Q and g_m/I_D doesn't scale well with process.
- Thus minimum I_D for $A_V = g_m R_p > 1$ needed for startup doesn't scale well.
- At 2 GHz, need >60μW minimum to satisfy start-up conditions. Power increased by extra phases and margin needed to account for variation.
- Start-up time $>> 1\mu s$.
- Requires level control, buffering to drive mixers.
- **Good phase noise.**

2 GHz oscillator minimum power versus process node scaling analysis from pg. 89 of [1].



(a) LC oscillator and ring oscillator schematics

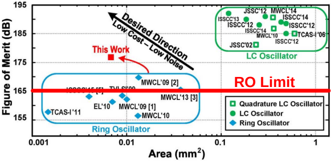
(b) Simulated power consumption

DCO topologies

LC Oscillator versus Ring Oscillator.

- **Ring oscillator**
  - Ring oscillator power scales well with process node. In 20 nm can achieve 2 GHz oscillator with ca. $1\mu W$
  - Extra phases with little to no power penalty.
  - Instant start-up.
  - Can reset to known phase, which can be exploited in PLL for faster lock.
  - Small area.
  - **Bad phase noise (+20 dB vs LC).**
- **Conclusion**: LC power prohibitive, have to use ring oscillator.

Oscillator figure of merit comparison.

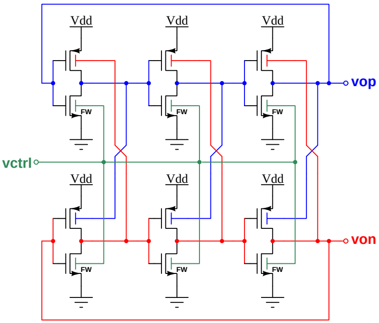


DCO topologies

Ring oscillator architecture.

- Proposed ring oscillator: pseudo-differential with back-gate control of frequency.
- Frequency tuning is highly linear with V_{ctrl}, allows for 0-V_{DD} control range.
- Use DAC to set control voltage (10 bit?) for medium frequency tuning.
- Design for 10% fractional tuning range, so 10 bits is 240 kHz/LSB at 2.4 GHz.
- Utilize 5-bit unit-capacitor based digital capacitor for fine tuning, approximately 7.5 kHz/LSB
- Need frequency step to be $<<$ PLL bandwidth (50 kHz).

Pseudo-differential ring oscillator utilizing backgate-based coupling and frequency control.



DCO frequency tuning and K_{DCO}

Backgate coupled pseudo-differential ring oscillator

— The center frequency of the oscillator, with $V_{ctrl} = V_{DD}/2$:

$$f_c = \frac{\mu_n C_{ox}}{4 \ln 2 NC} \left(\frac{W}{L}\right)_n \left[V_{DD}\left(\frac{7}{8 \ln 2} - 1 + \frac{\gamma}{2 \ln 2} - \frac{\gamma}{2}\right) - V_{t0}\left(\frac{1}{\ln 2} - 1\right)\right] \tag{6}$$

— The fractional tuning range of the oscillator is:

$$\frac{\Delta f}{f_c} = \frac{1}{2} \cdot \frac{\gamma V_{DD}\,(1 - \ln 2)}{V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\,(1 - \ln 2)} \tag{7}$$

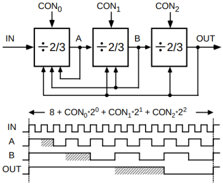
— If a N-bit DAC is used to control the oscillator, the resulting DCO gain is therefore:

$$K_{DCO} = \frac{\Delta f}{2^{N_{DAC}}} = \frac{f_c}{2^{N_{DAC}+1}} \cdot \frac{\gamma V_{DD}\,(1 - \ln 2)}{V_{DD}\left(\frac{7}{8} - \ln 2 + \frac{\gamma}{2} - \frac{\gamma}{2}\ln 2\right) - V_{t0}\,(1 - \ln 2)} \tag{8}$$

Divider topologies

Divider options.

- Divider design is more trivial than the other components.
- **Synchronous counter.**
  - Use ripple counter + TSPC FF's for low power.
  - Lower jitter due to being synchronous.
- **Multi-modulus divider (asynchronous).**
  - N chained 2/3 dual modulus dividers. 7 stage?
  - Tunable from 2^N to $2^{N+1} - 1$. N = 7 yields 128-255 range for divider modulus.
- Jitter compounds with divider chains, effect on phase noise???



Specification (unchanged)

System Performance Targets

Parameter	Value	Unit	Notes
Frequency	2.4-2.4835	GHz	2.4G ISM Band
Ref. frequency	16	MHz	Yields 6 channels
Power	$\leq$ 100	μW	
FSK BER	$\leq$ 1e-2		2FSK with f_{dev}=$\pm$250 KHz
Initial Lock Time	$\leq$ 50	μs	Upon cold start
Re-lock Time	$\leq$ 5	μs	Coming out of standby
Bandwidth	50	kHz	(nominally), tunable

Additionally: PLL output should support IQ sampling at LO frequency.

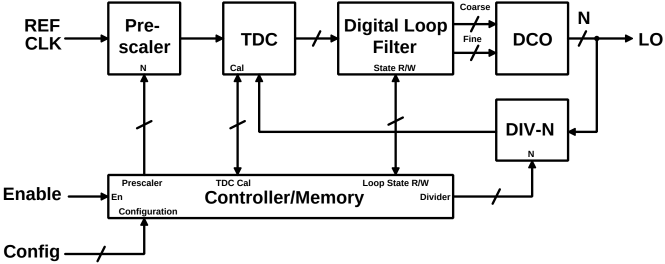
Specification (unchanged)

PLL Component Performance Targets

Parameter	Value	Unit	Notes
DCO LSB Resolution	$\leq$ 50	kHz	Determined from quantization noise.
DCO DNL	< 1	LSB	Ensures monotonicity
TDC Resolution	0.95	ns	
TDC Resolution (bits)	6	bits	

Architecture (unchanged)

Block Diagram



Power Targets

DCO	TDC	Divider	Other	SUM
70 μW	20 μW	10 μW	$<<$ 1 μW	100 μW

Project Phases

Autumn 2019

— System modeling and simulation.

- Learn PLL theory in detail
- Evaluate feasability of PLL architectures (counter, TDC-based)
- Determine requirements for TDC/DCO/Divider/logic (bits of resolution, accuracy etc) to meet PLL performance specifications.
- Determine digital logic for loop filter, validate stability and lock time performance.

— Research ultra-low power circuit topologies to implement system components that will meet determined requirements.

— Translate component-level specifications into schematic-level circuit designs.

- Try, fail, try again until functional at schematic level.
  - — I expect the TDC to be difficult.

Project Phases (continued)

Spring 2020

- — Finalize schematic-level design.
- — Estabilish thorough tests for PLL performance (automated?) to help in layout.
- — Layout of PLL.
  - Design iteration until design specs met.
  - Probably very time consuming.
- — Full characterization/validation of design performance.
  - Comprehensive Corners/Monte-Carlo testing (time consuming??)
  - More design iteration if new issues crop up...
- — Thesis paper writing.

References

[1] "Ultra-Low Power Wake-Up Receivers for Wireless Sensor Networks", N. Pletcher, J.M Rabaey, 2008.
http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-59.html