Ultra-Low Power PLL for Wake-up Receiver Applications

Specialization Project Progress - 3rd Week

Cole Nielsen
Department of Electronic Systems, NTNU

13 September 2019 (Week 37)

This Week

Timeline Tasks

— **Primary:** Start-up phase to modeling and simulation of PLL.
- Implement each PLL component as class.
  — Ref. Clock, TDC, loop filter, DCO, divider.
- Simulate system in discrete time (uniform step size):
  — Each iteration, call update method of objects representing PLL components.
  — Store node values within PLL to arrays.
  — For now have to used fixed time step for filtering and approach to generating phase noise (random phase walk in DCO phase signal).
- Set up Git repository for code, also for report/presentations.

Implementation in Python

Progress

- Ran into issue with my Python installations (3.6/3.7)
  - Virtual environments was not set up correct when I installed 3.7, so now I have issues with packages not working between 3.6/3.7
  - I.E. Can't use Control package in 3.6, can't use plotting package in 3.7
  - Wasted too much time trying to fix, currently just using 2.7
- Have implemented all blocks for PLL; preliminary phase noise plotting/analysis methods.
  - Can plot SSB phase noise.
  - Can fit phase noise model with 1/f dependency on freq. (corresponds to random phase walk component that is dominant in ring oscillators).
  - Need to include level quantization in loop filter.

Implementation in Python

Still To Do

- Implement level-quantization in loop filter.
- Automatic analysis of performance (lock detection, residual phase modulation, lock-in/pull-in range).
- Automatic optimization (using gradient descent) of PLL parameters?
- Z-domain modeling of loop? Develop (by hand) some ideal transfer funtions for loop.

Specification (unchanged)

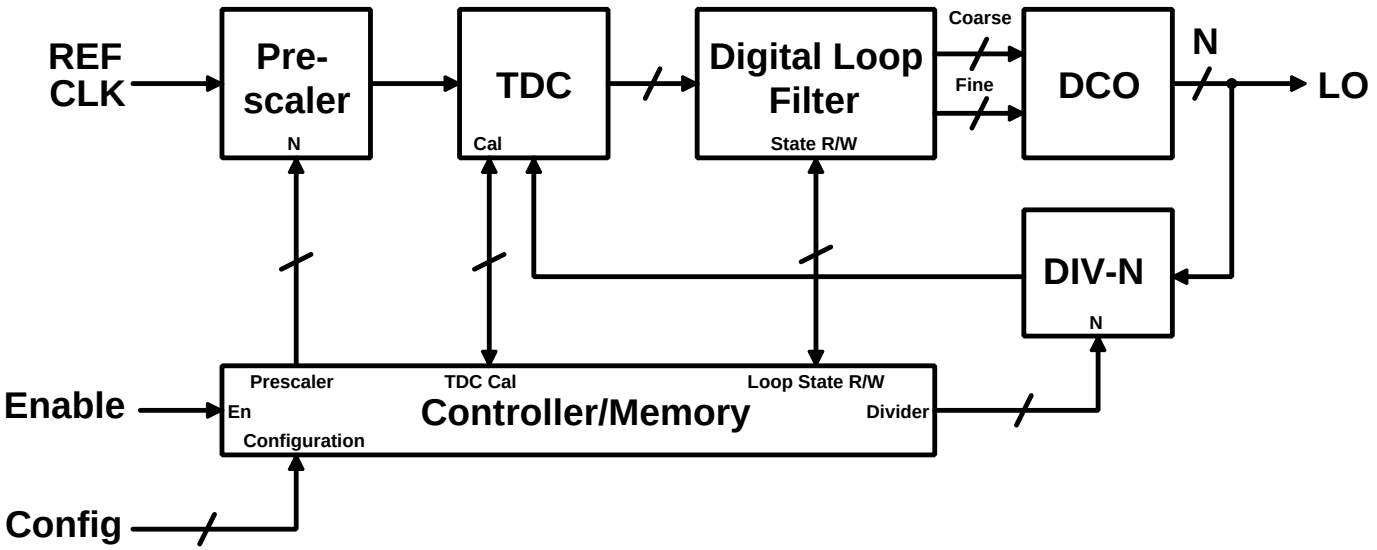
Preliminary Performance Targets

Parameter	Value	Unit	Notes
Frequency	2.4-2.4835	GHz	2.4G ISM Band
Ref. frequency	16	MHz	Yields 6 channels
Power	≤ 100	μW	
Residual FM	≤ 107	kHz_{RMS}	BER $\leq$ 1e-2, f_{dev}=$\pm$250 KHz
Initial Lock Time	≤ 50	μs	Upon cold start
Re-lock Time	≤ 5	μs	Coming out of standby
Bandwidth	100	kHz	(nominally), tunable

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

Architecture (unchanged)

Block Diagram



Power Targets

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

Autumn Timeline

Week Number	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	Ideal (ahdlLib?) implementation in Cadence of PLL
41	7.10 - 13.10	Circuit Research	DCO/Divider topologies
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	–	–

*I will write the report simultaneously with the work.

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.
- Establish 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.