Comparison of Design and Use of Analog, Digital, and Neuronal Phase-Locked Loops

•••

Ramsey Z. Doany
Texas State University

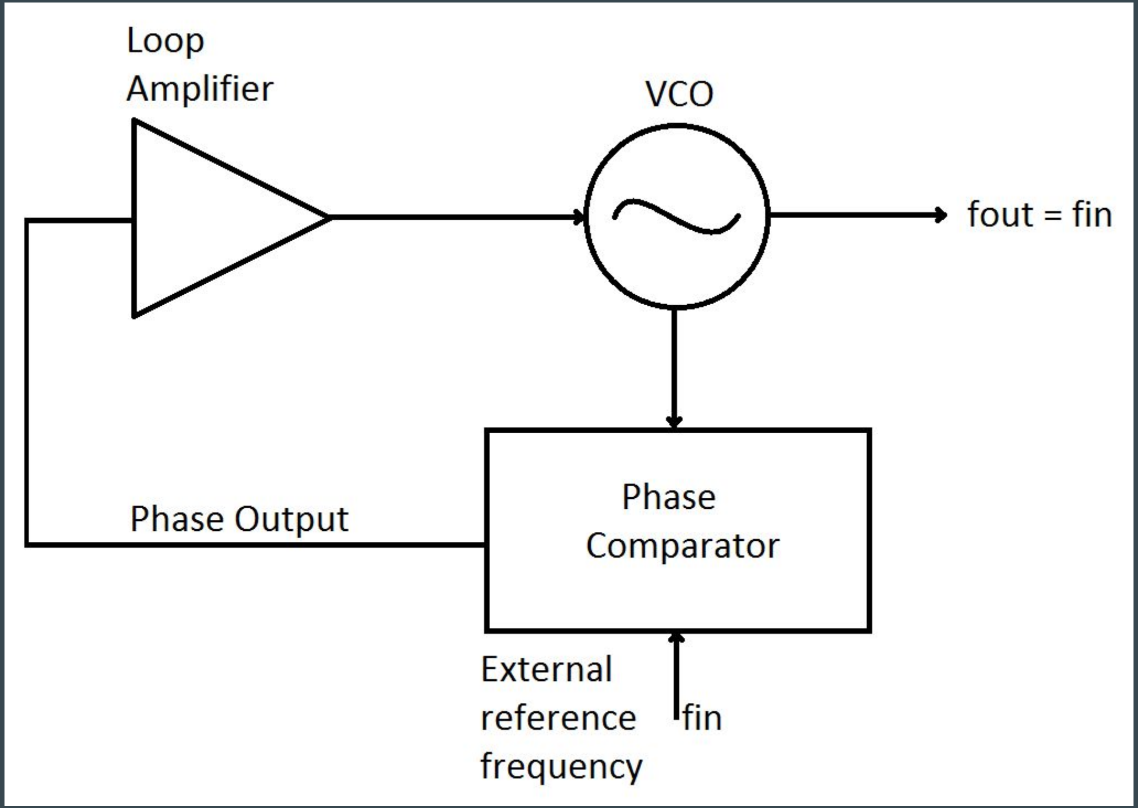
Outline

1. What is a PLL and How Are They Used?
2. History of PLLs
3. Analog PLL
4. Digital PLL
5. Software PLL
6. Neuronal PLL
7. Comparison and Conclusion

What is a Phase-Locked Loop?

- Device that produces an AC output whose phase is ‘locked’ to that of a reference frequency
- Components
  - Voltage Controlled Oscillator
    - Nonlinear
    - Produces oscillation in response to an input control voltage
  - Phase Comparator (or Phase Discriminator)
    - Nonlinear
    - Produces phase difference between two signals
  - Loop Filter

Simple PLL Block Diagram



Adapted from K. Stephan, Analog and Mixed-Signal Electronics

History of PLLs

- 1919 - Vincent and Appleton - Synchronization of Oscillators
- 1935 - Travis - “Automatic Frequency Control”
- 1940s and 1950s - LO FM Demod, Exciter for atomic particle accelerator amp
- 1964 - First modern PLL
  - Coherent Communication
- 1965 - First PLL IC
  - TV Sweeps
- 1970 - First Digital PLL → First ADPLL

PLL Use Today

- In every Cell Phone, TV, Radio, Computer,...
- Carrier Recovery
- Clock Recovery
- Data Recovery
- Frequency Synthesis
- Modulation/Demodulation
- LF LO in cell phone → SHF Communication
- Disk Drive Control
- Harmonic Compensation
- Motor Control
- MCUs

Usefulness/Uniqueness of PLLs

- Dependence on nonlinearity
  - Linear systems analysis (Simple PLLs)
- Almost always low order
- “Clean up”/Recover weak/noisy signals
  - Example: Deep space probes
- Reference signal can disappear
  - PLLs can ‘flywheel’

Types of PLLs

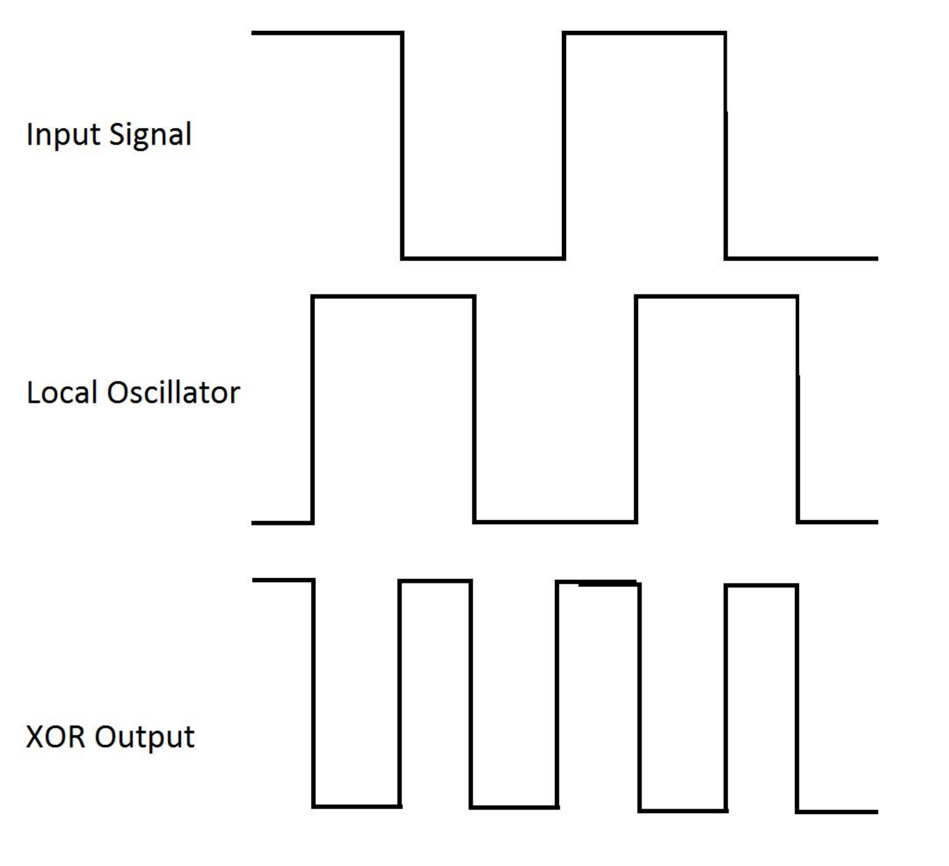
- Analog Phase-Locked Loops (PLL/APLL)
  - Usually, optimization comes with being Mixed-Signal
- (All) Digital Phase-Locked Loops (DPLL/ADPLL)
- (All) Software Phase-Locked Loops (SPLL/ASPLL)
- Neuronal Phase-Locked Loops (NPLL)

Analog PLL Design

- First-Order PLL
  - Loop Filter frequency response is equal to single pole LP filter
  - Very Inflexible
- Second-Order PLL
  - More flexible
  - Higher complexity

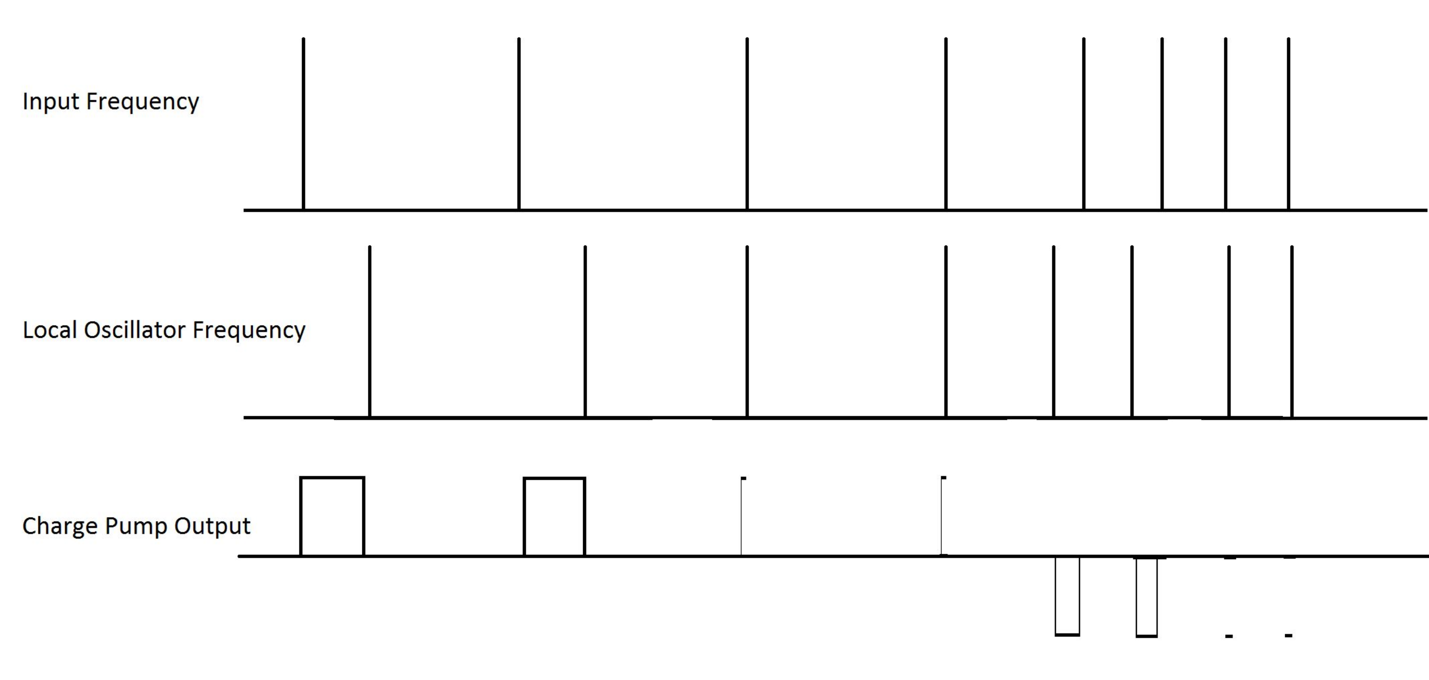
Phase Detector Output - XOR

- $\phi_{\text{LO}} \neq \phi_{\text{RF}} \rightarrow \text{IF high}$
- $\phi_{\text{LO}} = \phi_{\text{RF}} \rightarrow \text{IF low}$

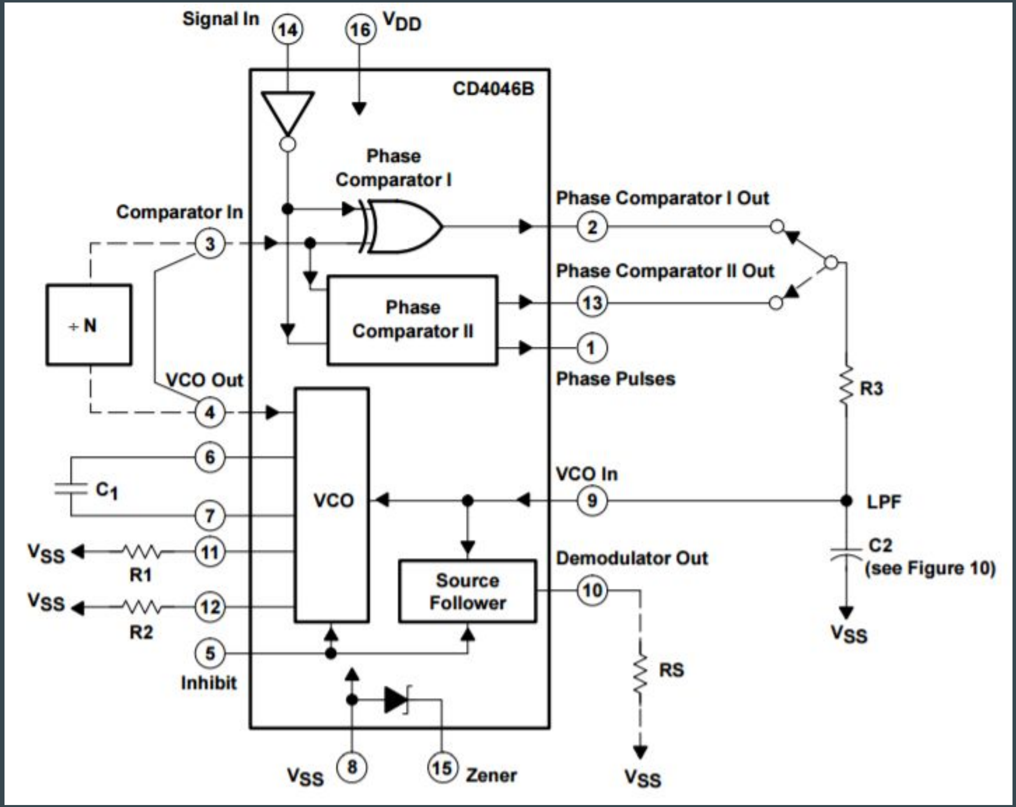


Phase Detector Output - Charge Pump

- $\phi_{RF} - \phi_{LO}$



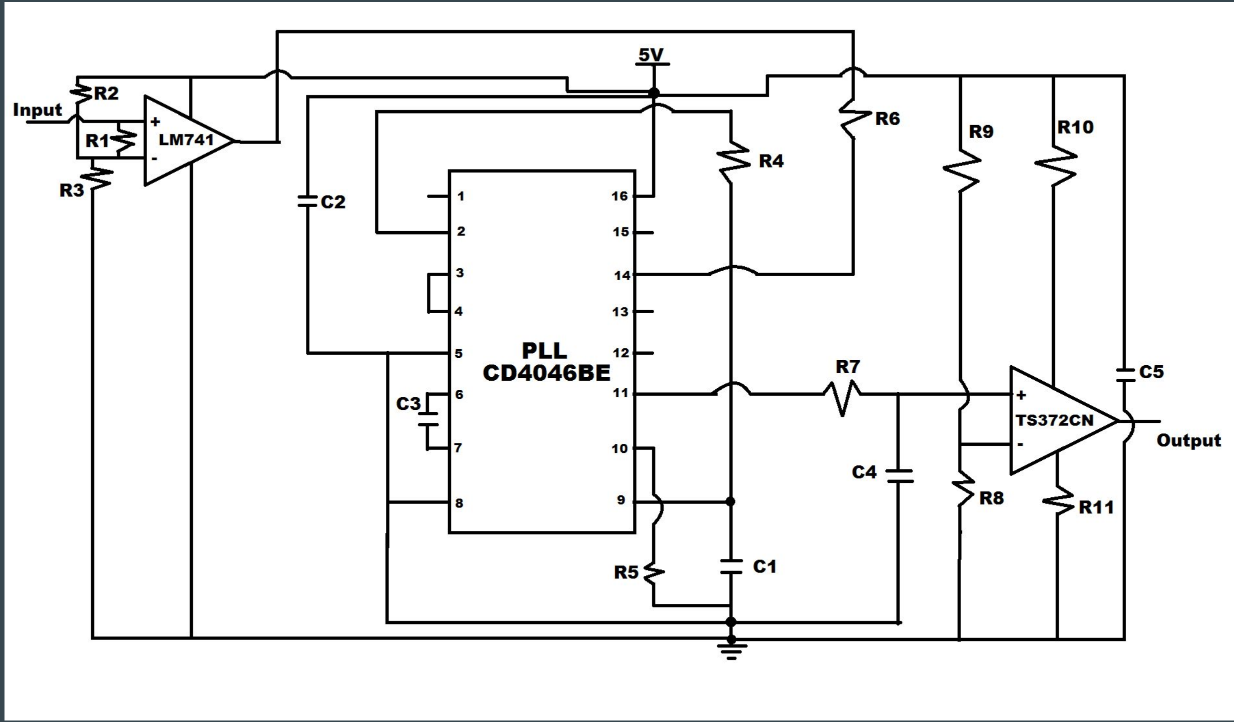
CD4046B PLL Block Diagram

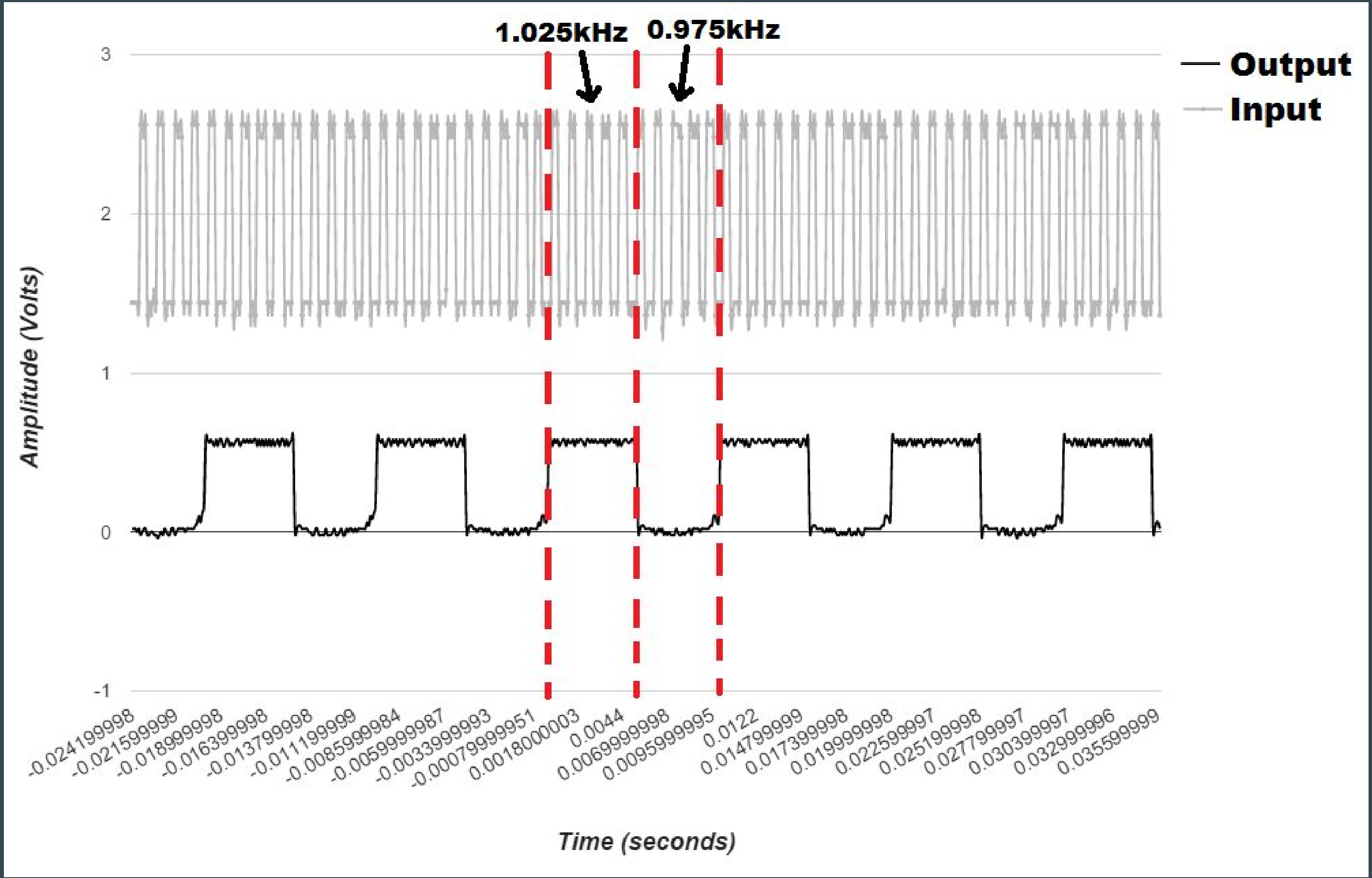


https://johnthacher.com/2017/03/03/phase-locked-loop-lab-4046/

Mixed-Signal PLL Example

- FSK Demodulator
  - f_c = 1kHz
  - Δf = 25Hz
  - Min V_{in} = 500mV
  - Data Rate = 100Hz
  - SNR = 20dB
  - TLL Output





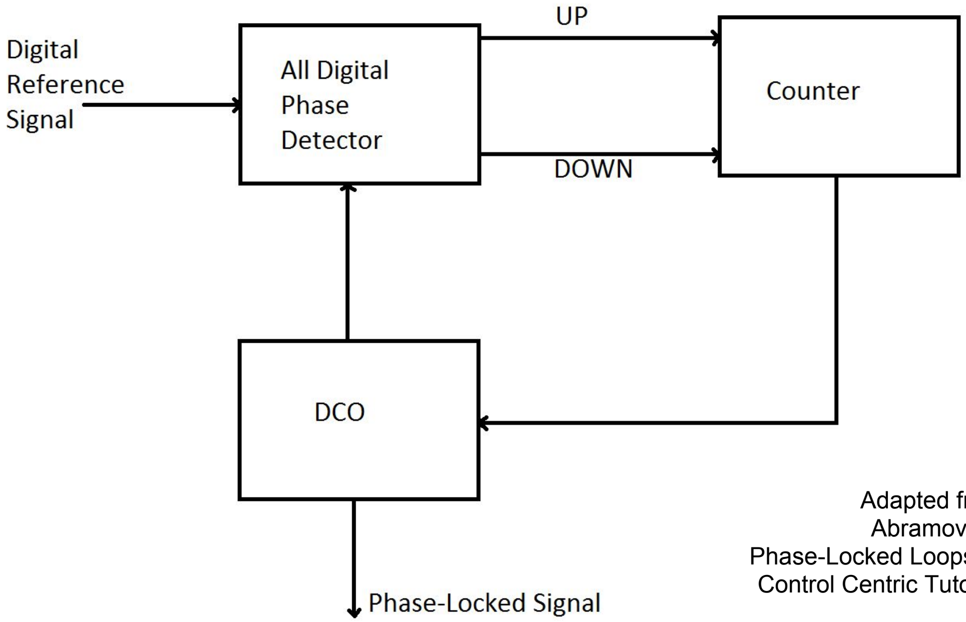
1.025kHz
0.975kHz
Output
Input
Amplitude (Volts)
3
2
1
0
-1
-0.024199998
-0.021599999
-0.018999998
-0.016399998
-0.013799998
-0.011199999
-0.0085999984
-0.0059999987
-0.0033999993
-0.00079999951
0.0018000003
0.0044
0.0069999998
0.0095999995
0.0122
0.014799999
0.017399998
0.019999998
0.022599997
0.025199998
0.027799997
0.030399997
0.032999996
0.035599999
Time (seconds)

Digital PLL Design

- Digital PLL (DPLL)
  - Phase Detector is digital
  - VCO and Loop filter still analog
- All Digital PLL (ADPLL)
  - Every PLL component replaced by digital counterpart

ADPLL Implementation I

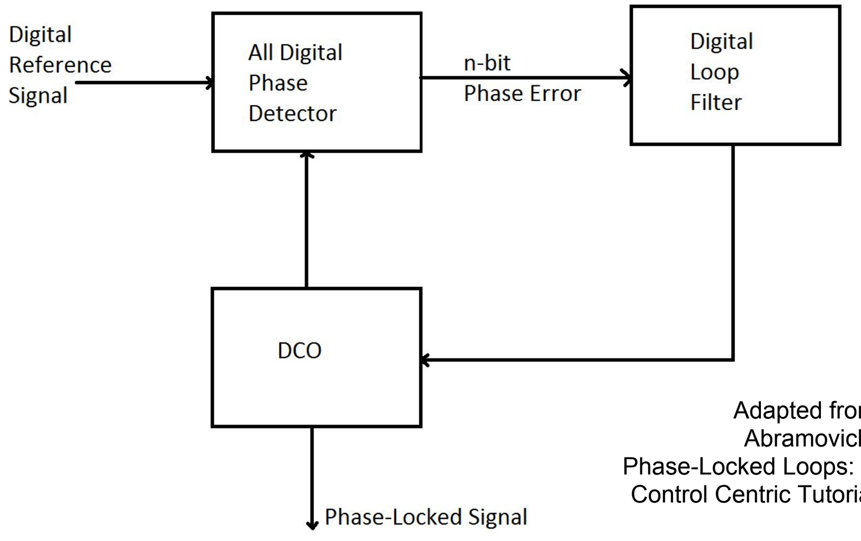
- Digital phase detector produces Up/Down count → Counter
- Counter acts as loop filter
- Counter controls DCO



Adapted from Abramovich, Phase-Locked Loops: A Control Centric Tutorial

ADPLL Implementation II

- Digital Phase Detector produces error value
  - n-bit
- Digital Loop Filter adjusts DCO accordingly



Adapted from Abramovich, Phase-Locked Loops: A Control Centric Tutorial

Digital PLL Implementation/Uses

- Partially or totally digital
- Not very susceptible to circuit drift (as compared to APLL)
- Can be implemented in FPGA
- Integratable into small IC packages
- Often more practical in non communication applications than APLL
- Often less practical than SPLL for many applications

Software PLL Design

- Mostly used for clock/data recovery
- Very similar to simulation of HW PLL
- Heuristic
  - CS Problem solving method
  - Used when standard mathematics too slow/complex
  - Approximate solution

Software PLL Implementation/Uses

- If Nyquist Theorem is met, loop filter can be SW
- Very flexible
- With a high enough sample rate and sufficient processor speed, any PLL can be software implemented
- Due to high processor speeds and implementation speed, SPLLs are often more practical than APLLs or DPLLs
- SPLLs can be used in real-time or in post processing
- SPLLs still do not perform as well at high frequency
    - Problem across SW as a whole

Neuronal PLL Design

- PLL Problem: RF >> LO or RF << LO or RF has large and fast swing
  - Phase detector gain → 0
  - Feedback Loop Fails
  - Circuit cannot lock
- Neuronal PLL
  - Artificial Neural Network (ANN) placed in feedback loop
    - Large number of “neurons”
      - Processing elements
    - Working together to solve problem
    - Elements given different “weights” in calculation

Neuronal PLL Uses

- PLL wide range input
- Automatic speech recognition
  - Speech rate changes disrupt other PLL locks
  - PRESENCE
    - PREdictive SENsor Control and Emulation (Moore 2007)
- Temporal Coding → Rate Coding Conversion
  - Neural “spike” timing → Average time/frequency
  - Biomedical simulations of brain
- Trade-offs
  - Complexity
  - Cost

Comparison

- Analog PLL
  - Rarely pure analog
  - Usually mixed-signal
  - Communication applications
- Digital PLL
  - DPLL/ADPLL
  - FPGA
  - Non Communication Applications
- Software PLL
  - SPLL/ASPLL
  - Cheap, quick implementation
- Neuronal PLL
  - Wide input freq. Range
  - Brain modeling

Questions?