Introduction to Integrated Delay and Phase-Locked Loops and Applications

16/20 May 2007, Padova, Italy

Paulo Moreira

CERN – Geneva, Switzerland

Paulo.Moreira@cern.ch

http://paulo.moreira.free.fr/microelectronics/padova/padova.htm

Outline

- Introduction
  - Early ‘history’
  - DLLs and PLLs what are they?
  - Making sure we understand each other
- Delay-Locked Loops
- DLL Applications
- Phase-Locked Loops
- PLL Applications

Early PLL History

- 1922 & 1927 First studies on oscillator synchronization:
  - E. V. Appleton. “*Automatic synchronization of triode oscillators, part iii*.” Proc. Cambridge Phil. Soc., tome 21, pp. 231–248. 1922
  - B. van der Pol. “*Forced oscillators in a circuit with non-linear resistance. (reception with reactive triode).*” Phil. Mag., tome 3, pp. 64–80, 1927
- 1932 The first publication on the PLL concept:
  - H. de Bellescise, “*La Réception Synchrone*”, Onde Electrique, vol 11, June 1932, pp. 230-240
- 1932 A team of British scientists develops the *homodyne* (*synchrodyne*) detection:
  - The signal is mixed with that of a local oscillator having the same frequency as the carrier
  - The technique was devised to eliminate the tuned stages of the superheterodyne
  - The early name was Automatic Frequency Control
- 1943 The concept is applied in TV receivers to the synchronization of the vertical and horizontal scan (Wendt & Fredendall)
- 1970 First integrated Phase-Locked Loops
  - 1968 Hans Camenzind proposed the idea to Signetics
  - 1970 two Signetics products:
    - Graham Rigby and Alan Grebene:
      - Circuit based on an emitter-coupled oscillator
    - Hans Camenzind:
      - “The 565 came out three months later and is still being manufactured”
  - This development allowed the “explosion” of the applications of phase-locked loops
  - It is very likely that you have at least one in your pocket

What are DLLs and PLLs?

- For those of you that don't know the meaning of these two groups of three letters:
  - DLL $\rightarrow$ Delay-Locked Loop
  - PLL $\rightarrow$ Phase-Locked Loop
- Let's try to be more specific:
  - DLLs are negative feedback control systems that try to adjust the propagation delay of an internal circuit so that it matches the period of a reference signal.
  - PLLs are negative feedback control systems that try to adjust the oscillation frequency and phase of an internal oscillator so that they match the frequency and phase of a reference signal.
- These look like pointless operations!
  - The 'secret' is that:
    - DLLs track the 'average' period of the reference signal
    - PLLs track the 'average' phase (and frequency) of the reference signal
- We will spend some time trying to make these statements clear.

DLL/PLL = Control System

DLLs and PLLs are control systems:

- Intrinsically:
  - DLLs are first order systems
  - PLLs are second order systems
- In practice due to the chosen architecture or the practical implementation:
  - DLLs might become second order
  - PLLs might become third (or higher) order
- It is necessary to ensure that:
  - They are <u>stable</u>
  - The <u>transient response is optimal</u>
- Since PLLs and DLLs are control systems, standard control theory applies
- The quantity being controlled is not a position, velocity, temperature…, but:
  - The propagation delay in the case of a DLL
  - The oscillator phase and frequency in the case of a PLL

Linear / Non-linear?

- DLLs and PLLs are non-linear systems
- However, under certain conditions they can be considered linear and most of the theory is usually developed under that assumption
- We will look both at cases were a linear approximation can be used and cases were the systems has to be considered as non-linear

Why to study DLLs and PLLs?

- These circuits are becoming almost ubiquitous in electronic system:
  – From TV, passing through GSM, microprocessors... and down to memories
- The widespread use of these circuits is mainly due to monolithic integration:
  – Discrete implementations require a relatively high component count making them an expensive part of the system
  – Nowadays, due to the scale and integration potential of VLSI technologies these components are of modest complexity and represent only a small fraction of the system cost
- An engineer must thus be familiar with the basic concepts if he/she:
  – Wants to use it as a ‘black box’;
  – Needs to specify one;
  – Needs to design one for a specific application.

Applications

The range of applications is large. Here are some of the most important:

- Delay-Locked Loops:
  - Clock phase adjustment in clock domains of microprocessors
  - Sub-gate delay timing adjustment circuits
  - Multiple clock phase generators
  - Time-to-Digital (TDC) converters
- Phase-Locked Loops:
  - Jitter reduction
  - Skew suppression
  - Frequency synthesis
  - Clock recovery
  - Phase demodulation
  - Frequency demodulation
  - ...
- We will discuss in detail some of these applications

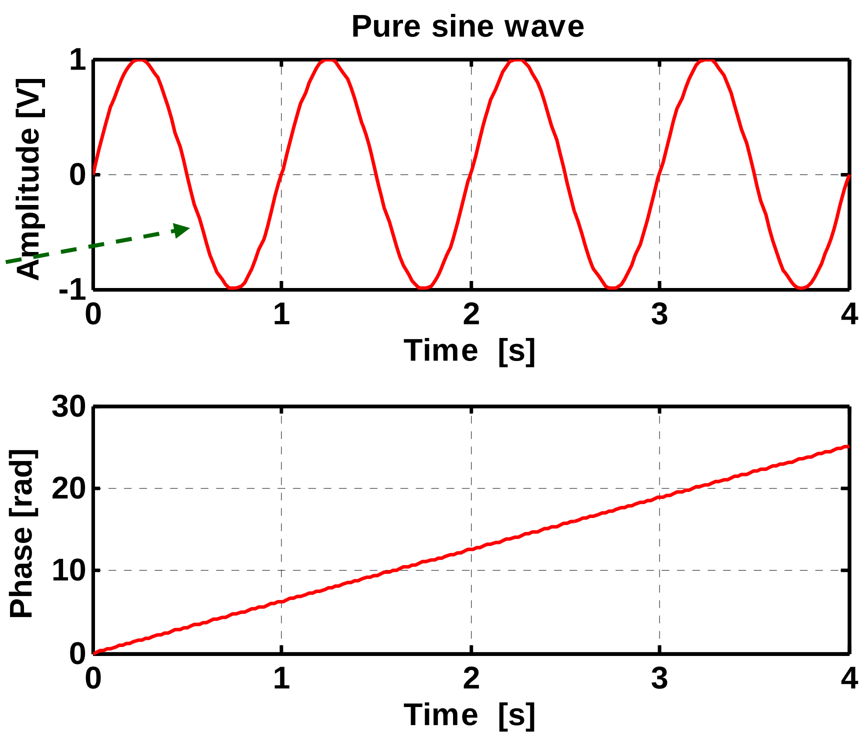
Frequency & Frequency!?

- Before proceeding it is important to clarify a point:
  - In the world of PLLs and DLLs, the word frequency is often used with respect to multiple concepts!
- Lets consider a signal that you know quite well: a <u>pure</u> sine wave:
  - The signal oscillates between two levels: ±A
  - If we assume (for simplicity) its initial phase to be zero, the zero crossings are exactly at $t = n \cdot T$, with $n = \ldots,-2, -1, 0, 1, 2,\ldots$ and $T = 1/f_0$
- Mathematically this signal is written as:

$$\boxed{A \cdot \sin\left(2 \cdot \pi \cdot f_o \cdot t + \phi_0\right)}$$

- To completely specify the signal, it is enough to specify:
  - Its amplitude: A
  - An initial phase Φ_0
  - Its oscillation frequency: f_0
- We can, alternatively, describe the signal by its Fourier transform. This will lead to a <u>*frequency domain*</u> representation of the signal:
- Mathematically, two deltas of *Dirac* at $\pm f_0$ ($\pm\omega_0$) are used to represent the sine wave:

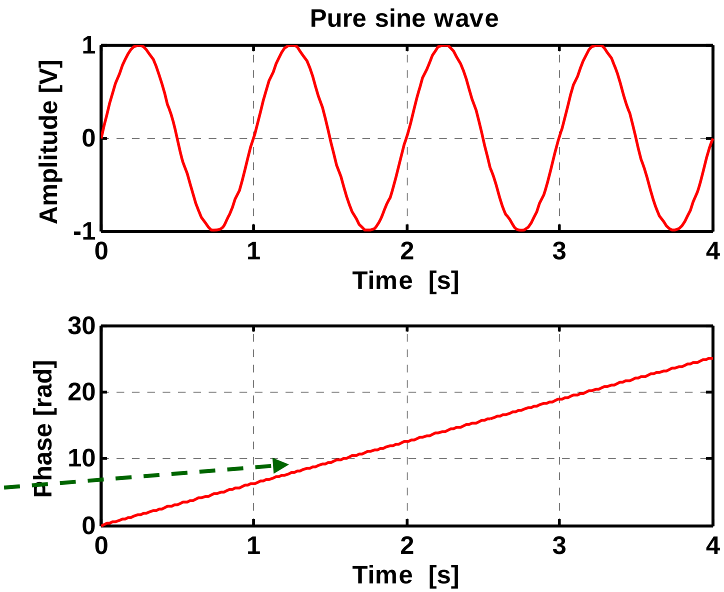
$$\boxed{A \cdot \left(j \cdot \pi \cdot \delta(\omega_0 + \omega) - j \cdot \pi \cdot \delta(\omega_0 - \omega)\right)}$$



Phase is most important

- DLLs and PLLs are devices sensitive to the phase:
  - They are “insensitive” to the amplitude of the signal (almost always)
- Thus, to analyze a DLL or a PLL it is enough to know how the signal phase behaves as function of time (or, in more ‘fancy words’, in the *time domain)*:

$$2 \cdot \pi \cdot f_o \cdot t + \phi_0$$



- In the case of the ‘pure’ sine wave (starting at time t=0), the phase is a ramp of slope: $2 \cdot \pi \cdot f_0$

Time domain

$$\phi(t) = 2\pi \cdot f_0 \cdot \int u(t)\, dt$$

- Alternatively, we can take a Fourier transform of the phase signal and obtain its frequency domain representation:

Frequency domain

$$\phi(f) = \frac{f_0}{j \cdot f} \cdot \left[\frac{1}{j \cdot 2\pi \cdot f} + \tfrac{1}{2}\delta(f) \right]$$

A more realistic phase signal

Example:

Noisy square wave oscillator.

- Mathematically:

$$s(t) = sign[\sin(\phi_s(t))]$$

where:

$$\phi_s(t) = 2 \cdot \pi \cdot \langle f_o \rangle \cdot t + \phi_0 + \int_{-\infty}^{t} f_n(t)\, dt$$

The signal is strictly not periodic but it is still possible to define an average frequency

The frequency noise is a random variable with zero mean

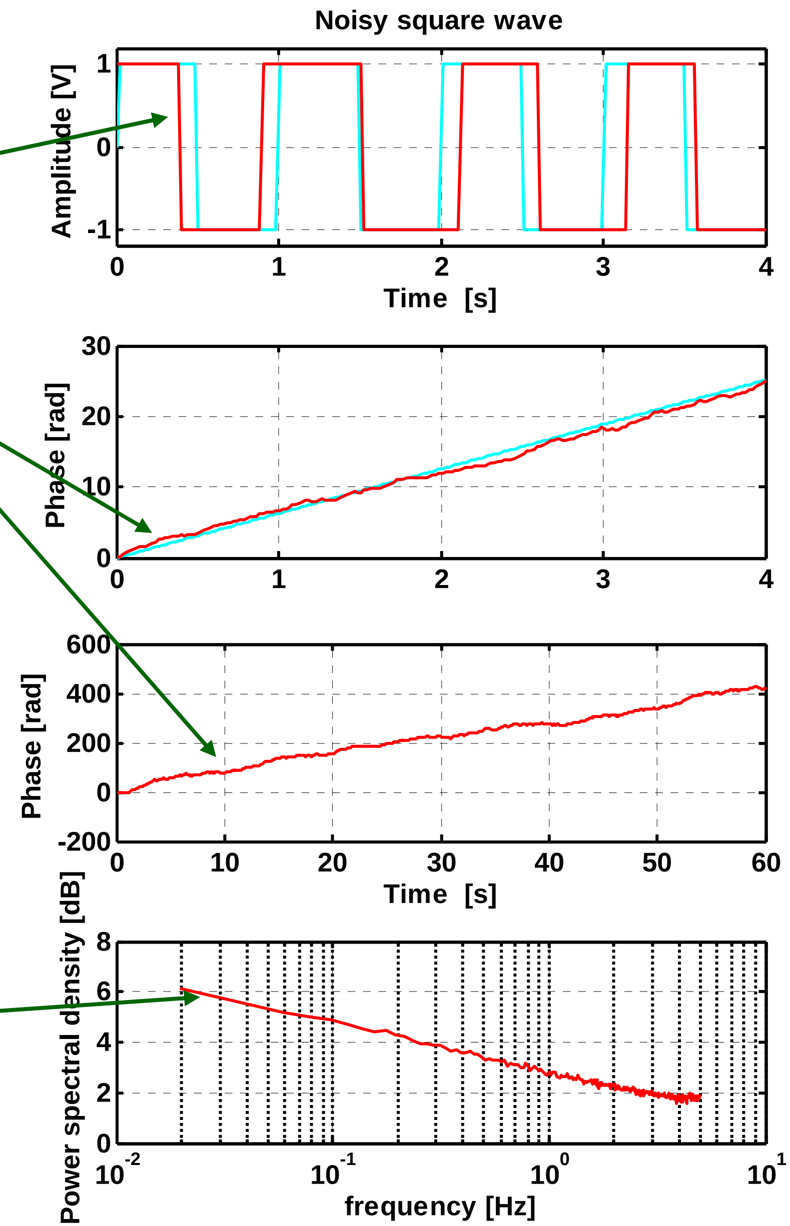
The phase noise is cumulative, that is, the 'end' of a cycle is the 'starting point' of the next.

The phase noise is also called jitter

- For each outcome we can define the Fourier transform $\phi_s(f)$, but in this case the meaningful quantity is the power spectral density:

$$\phi_{ss}(f) = \lim_{T \to \infty} \frac{\langle \phi_s(f) \cdot \phi_s^*(f) \rangle}{2 \cdot T}$$

This is the quantity you can measure using a spectrum analyzer



Summarizing

We are "only" interested in the Phase

- PLLs and DLLs lock to "periodic" signals;
- In general these signals have both time dependent amplitude and phase;
- PLLs and DLLs are sensitive to the signal's phase (and not amplitude);
- Since the phase is **the** signal we can look at it both in:
  - The time domain;
  - The frequency domain.
- The phase signal has both:
  - A time domain representation $\phi(t)$
  - A frequency domain representation $\phi(f)$, $\phi(\omega)$ or Laplace transform $\phi(s)$
- Example: consider a PLL locked to a 1 GHz jittery clock signal:
  - The average frequency of the reference signal is 1 GHz;
  - Its VCO is working at an average frequency of 1 GHz;
  - This PLL might have, e. g., a 10 MHz signal bandwidth only;
  - This means that any spectral content of the phase noise that is above 10 MHz will be low-pass filtered by the PLL phase transfer function.