Exact Signal Measurements using FFT Analysis

Stefan Scholl
Microelectronic Systems Design Research Group
TU Kaiserslautern, Germany

1 Introduction and motivation

This tutorial describes how to accurately measure signal power using the FFT. While it is easy to calculate powers in the time domain, this is not always applicable: If a signal contains several spectral components and broadband noise, considering the frequency domain allows to measure power of these components individually or to measure SNR (by separately considering signal and noise). However, obtaining accurate power numbers after having performed an FFT is not straightforward, because several effects introduce errors during FFT processing, mostly due to windowing. This paper describes the different effects and explains how they can be avoided or compensated. Finally, it will be explained how to do accurate measurements of signal and noise power using the FFT spectrum.

2 Basics

Before we dive into the details, some basics on FFT for real valued signals (as they frequently occur in real world) are given. If you are familiar with the basics you can step to Section 3 immediately.

2.1 FFT for real valued signals

In this paper real valued time domain signals are assumed, for which a N point FFT is used to transform it into the power spectrum with bin spacing $\Delta f = f_s/N$.

To calculate the N point FFT the Matlab algorithm 1 can be used. Here, after taking the FFT, its magnitude is calculated and the bins are scaled by $1/N$. Since the spectrum is mirrored, the first half of $N/2$ bins contains all necessary information on the spectrum, the second half can be discarded. To account for this discarding, the remaining bins are scaled by a factor of 2, except the (first) DC bin. This results in a RMS valued FFT representing the *RMS spectrum*. Squaring the results leads to the *power spectrum*. Note, that for real signals a N point FFT is calculated, but the spectrum may contain only $N/2$ bins.

Algorithm 1 Calculate an FFT with Matlab

```
input:  N real time domain samples (time_signal)
twosided_fft        = (1/N) * abs( fft(time_signal,N) ); % do fft
onesided_fft(1)     = twosided_fft(1);              % copy DC bin
onesided_fft(2:N/2) = 2 * twosided_fft(2:N/2);      % double other bins
power_fft           = onesided_fft.^2;              % convert from RMS to power
output:  N/2 point power spectrum
```

If the time signal represents a voltage over a resistor R, the output of Alg. 1 can be converted to dBm on a logarithmic scale with

$$P[dBm] = 10 \cdot log_{10}\left(\frac{\text{power\_fft}}{R \cdot 1mW}\right)$$

An overview of all calculation steps is shown in Fig. 2.1.

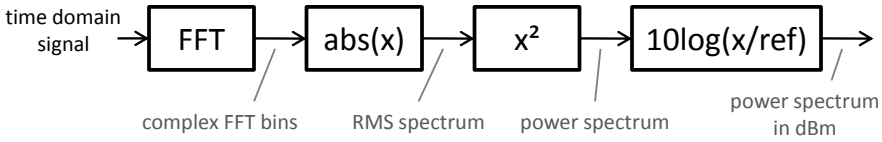

Fig. 2.1: From complex FFT output to the power spectrum in dBm

Further information on the basics of FFT can be found in [1]. Alternative means of calculating the signal power in the time domain can be found in Appendix A.

3 Effects introducing errors

In the following all effects that influence amplitude measurements using FFT are described. Some effects only apply to narrowband signals (e.g. sine), others to broadband noise and some to both.

3.1 Leakage

Spectral leakage is the effect, that the energy of the signal is distributed (smeared) among many frequency bins. A sine signal is not represented as a single sharp peak, but more like a broad bump, Fig. 3.1.

Spectral leakage is occurring because of the fact, that the FFT analyzes only a (short) slice of a signal (N samples). From this slice we usually want to extract information about the whole signal. The FFT actually outputs the spectrum of a theoretical signal, that is composed of infinite repetitions of that slice of N samples. Usually this theoretical signal has discontinuities at the borders of the slices. Therefore the spectrum output by the FFT does not exactly represent the one of the "true" signal.

For signals, whose periods (or a multiple of a period) coincidentally or intentionally fit exactly in the slice, leakage effects do not occur. This can be achieved, by coupling the signal to be measured to the analog-to-digital converter's sample clock. Since this is in practice rarely possible we assume in the following that leakage occurs.

To suppress leakage, windows can be used, that are applied to the time domain samples before FFT. Commonly used windows are the *hann* window (for general purpose), the *flattop* window (for accurate amplitude measurement) or no window, i.e. a rectangular window, at all (for noise measurements), see Fig. 3.1. The amount of leakage, that is displayed using a window can be measured by *highest sidelobe* in dB and *fall off* in dB/octave (more information in [2]).

Leakage is not a problem for accurate measurements, as long as leakage does not mask spectral components, such as spurs or noise.

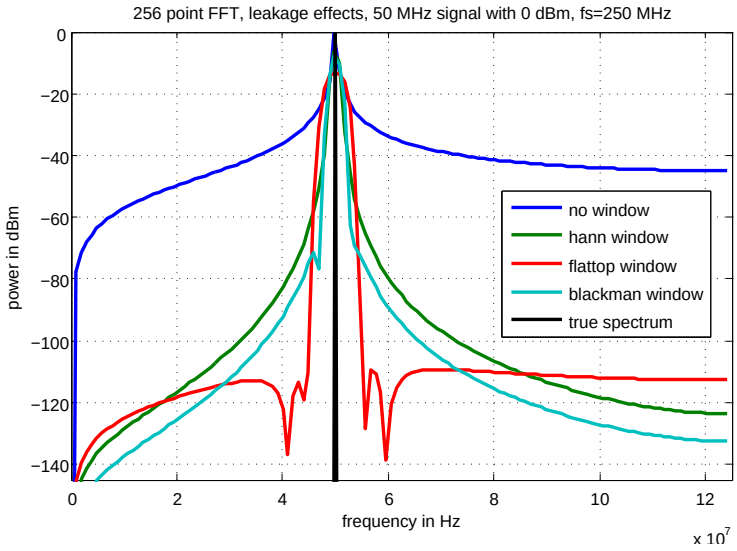

Fig. 3.1: True spectrum and the leakage effect of FFT for different windows

3.2 Coherent power gain

If a window is applied, it reduces the amplitude of the time domain signal, especially at the left and right borders of the window, as shown in Fig. 3.2.

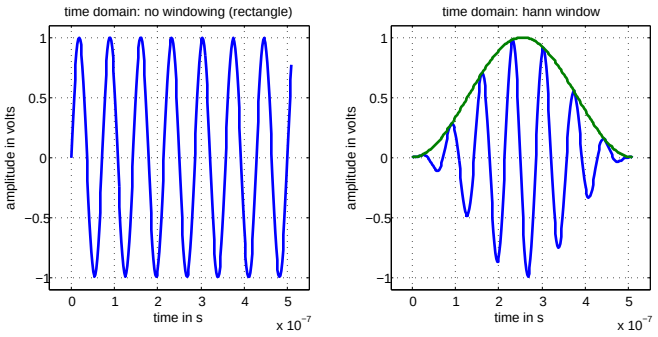

Fig. 3.2: The reason for coherent power gain: amplitude reduction due to windowing

This reduction of amplitude introduces an amplitude error, termed *coherent power gain* (CPG). The word "gain" may be misleading, since CPG actually describes a loss in signal power. Fig. 3.3 shows how the FFT amplitude is reduced in practice by the use of windows.

Every window has a fixed, characteristic CPG. Values for practically relevant windows are given in Section 3.6 or can be found in [3]. Simply add this gain to the FFT output to compensate for the power reduction. If no windowing (rectangle window) is used, there is no power loss and the coherent power gain is 1 or 0 dB.

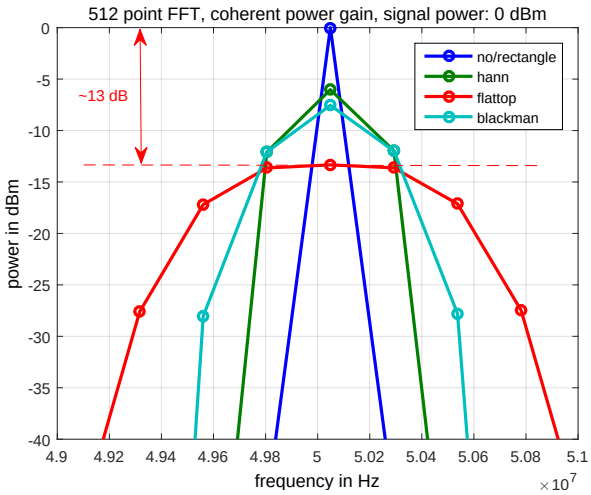

Fig. 3.3: The impact of coherent power gain (CPG): true amplitude of the sine signal is 0 dBm, errors for the different windows can be clearly observed (here frequency was adjusted to avoid leakage). For the flattop window CPG can be read as approximately 13 dB (exact: 13.33 dB).

3.3 Scalloping loss

The FFT processes digital data, which is by definition discrete both in time and frequency. Due to frequency discretization the frequency of a signal may fall in between two bins. If this is the case, the displayed power level is reduced because the signal power is spread among two bins. This loss is called *scalloping loss*. As a side effect leakage may then show up worse.

Discretization of frequencies can also be thought as sampling the corresponding continuous spectrum. Dependent on which frequencies exactly the continuous spectrum is sampled, the FFT spectrum looks different, as can be seen in Fig. 3.4. This is also called the *picket pence effect*, because the sampling is similar to view the continuous spectrum though a picket fence

Fig. 3.4 shows two different cases: on the left, the signal frequency falls nearly on a bin frequency: amplitude loss is small and leakage reduced. On the right the signal frequency falls between two bins: signal power is shared and displays reduced amplitude and leakage is stronger.

A very unpleasant property of scalloping loss is the fact, that it cannot be described by a fixed value for each window. So it cannot be used as a simple scaling factor to compensate the loss. Scalloping loss is in general even different for different spectral components. Scalloping loss largely depends on signal frequency, sample rate and the number of bins. However, one can specify a worst case loss, which occurs, if a signal frequency falls exactly half-way between two bins.

Furthermore, there are two ways to avoid scalloping loss:

- Use a flattop window, which exhibits only very little maximum scalloping loss ($<$0.02 dB)
- Couple the signal frequency properly with the sample rate, which is the same method used to avoid leakage (see Sec. 3.1).

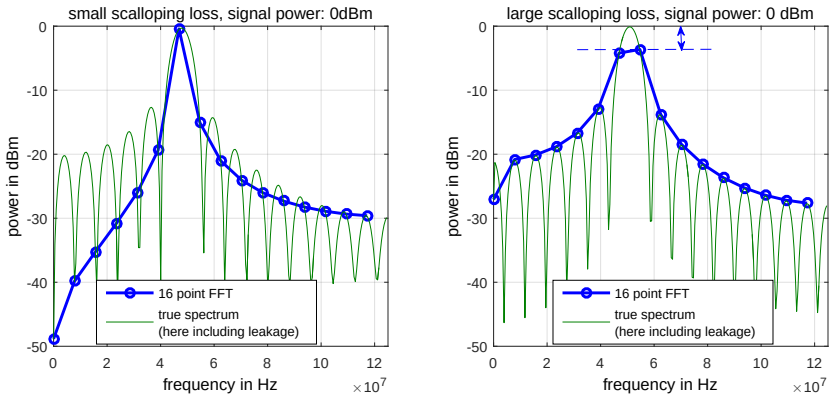

Fig. 3.4: Scalloping loss, that occurs when signal frequency falls between two bins, here no windowing was used to avoid coherent power gain

3.4 FFT processing gain

Processing gain reduces the displayed noise floor and can be explained as follows: FFT processing can be seen as sending a time signal through a bank of N filters, each with bandwidth Δf (bin spacing) and determining the power at every filter output. As the number of frequency bins or N (or filters respectively) is increased, the filters become narrower and the power at each bin (power at the "filter output") becomes smaller, see Fig. 3.5.

The *processing gain* (PG) exactly describes this reduction. If N is doubled, Δf is halved and the displayed noise floor is reduced by 3 dB. This affects power measurements of broadband signals, such as noise. The noise floor in an FFT plot is therefore displayed lower (by the processing gain) than it actually is. Processing gain can be calculated by

$$PG[dB] = 10 \cdot log_{10}\left(\frac{N}{2}\right)$$

and can be added to the level of noise floor to compensate this effect.

To measure noise floor accurately the results of several FFTs can be averaged to reduce amplitude fluctuations (Fig. 3.5, right).

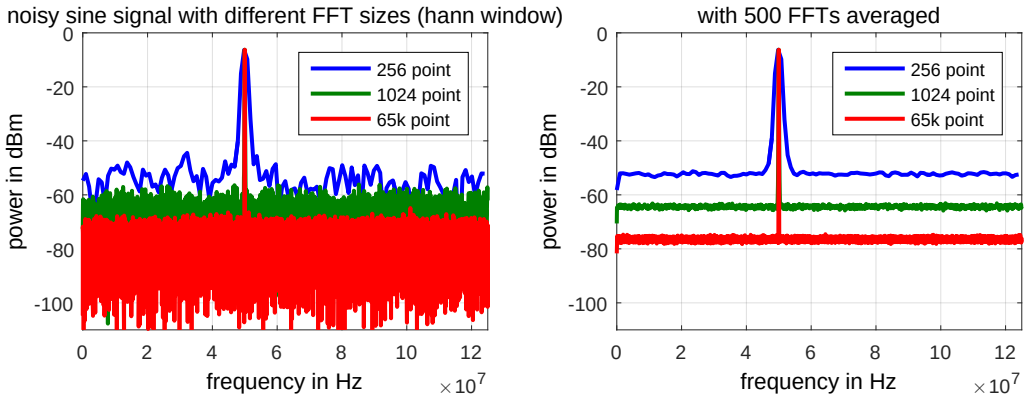

Fig. 3.5: Processing gain: The reduction of the displayed noise floor is clearly visible. To reduce fluctuations multiple FFTs have been averaged to enable a clear read of the noise floor level

3.5 Equivalent noise bandwidth

Dependent on the applied window the amount of noise that is accumulated in one frequency bin varies, based on its characteristic *equivalent noise bandwidth* (ENBW) [3, 4]. This effects leads to an increased displayed noise floor, if a non rectangular window is used. To account for this effect a correction factor needs to be subtracted from the noise floor to compensate for the larger equivalent filter bandwidth. This correction factor in dB can be calculated from the ENBW for every window, see Tab. 2. Take into account this number, if noise floor measurements are conducted.

3.6 Summary

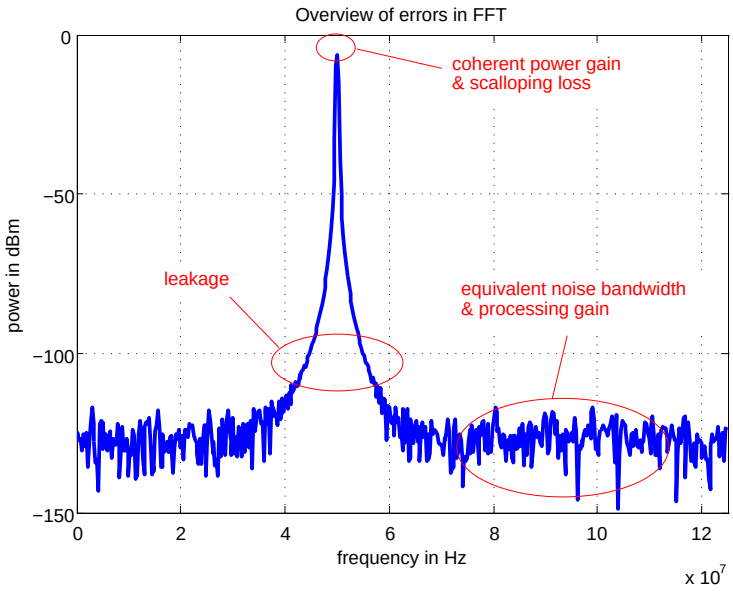

Fig. 3.6: Overview of all errors, that occur during FFT processing (here a noisy sine signal with 0 dBm, hann window)

Tab. 1: Overview of errors introduced during FFT processing

Error effect	**Source**	**Sine signal errors**	**Noise floor errors**
Leakage	finite length of FFT	spectral purity	negligible
CPG	windowing	amplitude	yes
Scalloping Loss	discretization	amplitude (variable)	no
Processing Gain	FFT filter bank property	no	yes
ENBW	windowing	no	yes

Tab. 2: Overview over different windows and their parameters, for more windows see [3]

Window	**Highest sidelobe**	**Fall off (per octave)**	**CPG**	**Scalloping loss (max)**	**ENBW correction**
No (rectangle)	-13 dB	-6 dB	0 dB	3.92 dB	0 dB
Hann	-32 dB	-18 dB	6.02 dB	1.42 dB	1.76 dB
Blackman	-58 dB	-18 dB	7.54 dB	1.10 dB	2.38 dB
Flattop	n/a	n/a	13.3 dB	0.02 dB	5.76 dB

4 Exact measurements

Exact measurements of signals can be done using the FFT, if the errors described above are properly taken into account. In the following sine signals and noise will be described separately since they are subject to different error effects.

4.1 Sine signals and narrowband signals

Sine signals and other narrowband signals (bandwidth smaller than the bin spacing Δf) show up as peaks in the spectrum.

The true power of narrowband signals P_{true} can be calculated from the displayed power P_{displ} (= power of spectrum peak) by

$$P_{true}[dBm] = P_{displ}[dBm] + CPG[dB] + scalloping\ loss[dB] \tag{4.1}$$

Coherent power gain can be read from Tab. 2. Note, that scalloping loss is not constant. If the signal energy is captured by a single bin, 0 dB can be assumed, if the signal energy is equally distributed among two bins, choose the worst case value from Tab. 2. If exact values are required and the signal cannot be centered on a single bin, the easiest way is to use a flattop window, which exhibits almost no scalloping loss (always $<$0.02 dB). For further information on the flattop window and its computationally efficient usage (if required) see [5].

Measurement of frequency is straightforward if the desired accuracy is coarser than the bin spacing. Otherwise interpolation methods can be applied, that improve frequency resolution without using more FFT points [1].

4.2 Noise and other wideband signals

Here two signal types have to be distinguished: white noise, i.e. noise that has a constant power density over the whole frequency band, and arbitrary noise or broadband signals.

4.2.1 Reading the average noise floor (white noise)

This method is applicable to white noise, where the true noise power P_{true} can be calculated from the displayed noise floor P_{floor} by using:

$$P_{true}[dBm] = P_{floor}[dBm] + CPG[dB] + PG[dB] - ENBW_{corr}[dB] \tag{4.2}$$

For an accurate measurement the displayed noise floor has to be determined precisely. However, due to random properties of noise this may be difficult. To obtain a clear power level of the noise floor, there are two solutions:

- FFT averaging: average multiple FFTs (linear scaled power spectrum) to reduce the randomness of the noise floor (see Fig. 3.5)
- Bin averaging: average the bins of the linear scaled power spectrum of a single FFT (exclude bins representing DC and other unwanted components), as described in the next section

4.2.2 Summing of FFT bins (arbitrary noise and wideband signals)

Summing of FFT bins is an elegant way to measure the power of all kind of noise or broadband signals. Instead of determining the noise floor by means described above, frequency bins ($FFT(i)$, power spectrum) of interest that contain signal components to be measured, are simply summed. Note, that the summation has to be done using linear values (not dBs) and that no correction of processing gain is required, since it is inherently considered by the summation process.

$$P_{true}[dBm] \quad = \quad 10 \cdot log_{10} \left(\sum_i FFT(i)[linear] \right) + CPG[dB] - ENBW_{corr}[dB]$$

The advantage of this method is, that the bins that contribute to the measurement, and therefore the spectral components, can be chosen freely, which allows for very flexible measurements.

4.3 Example

Fig. 4.1 shows a typical FFT measurement of a sine signal plus white noise. A hann window has been used, so the displayed powers are not the true ones. The displayed power of the sine signal can be read out: $P_{displ} = 5.9\,dBm$. Using Eq. 4.1 the true power of the sine signal is $P_{true} = 5.9\,dBm + 6.0\,dB + 0\,dB = 11.9\,dB$ (CPG is 6 dB, for the scalloping loss zero has been assumed since the peak falls nearly exactly on a single bin).

The noise floor can be evaluated using Eq. 4.2: $P_{true} = -18.3dBm + 6.0dB + 27.1dB - 1.8dBm = 13.0\,dBm$. (CPG is again 6 dB, processing gain for a 1024 point FFT is 27.1 dB, correction factor for ENBW is approx. $-1.8\,dB$)

To compare: the test signal of Fig. 4.1 has been synthesized from a 11.94 dBm sine signal plus a 13.02 dBm noise signal.

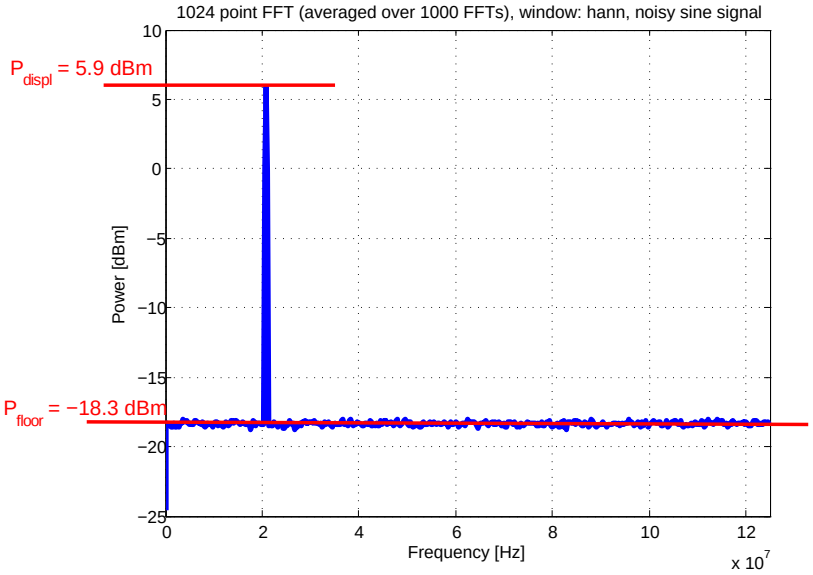

Fig. 4.1: FFT of a typical measurement: for the sine 5.9 dBm is displayed, the true value is 11.9 dBm; the displayed noise floor of -18.3 dBm corresponds to a total noise power of 13 dBm.

References

[1] R. G. Lyons, *Understanding Digital Signal Processing*. Pearson, third ed., 2011.

[2] A. H. Michael Cerna, "The Fundamentals of FFT-Based Signal Analysis and Measurement," *National Instruments, Application Note 041*, 2000.

[3] F. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," in *Proceedings of the IEEE*, vol. 66, 1978/2005.

[4] "FFT Window Function, Limits on FFT Analysis," *Bores Signal Processing*.

[5] R. Lyons, "Reducing FFT Scalloping Loss Errors Without Multiplication [DSP Tips and Tricks]," *Signal Processing Magazine, IEEE*, vol. 28, pp. 112–116, March 2011.

Appendix

A RMS and power in the time domain

Power or the RMS of an arbitrary digital signal can be easily calculated in the time domain. However, in contrast to measurements based on FFT analysis separation between different signal components in the frequency domain is difficult. For a time domain voltage signal s RMS equals the standard deviation, power equals the variance.

$$V_{rms} = \sigma(s)$$

$$P = var(s) = \sigma^2(s) = V_{rms}^2 \tag{A.1}$$

If the signal represents a voltage over a specific resistor R (e.g. 50 Ohms) the absolute power in dBm can be calculated:

$$P[dBm] = 10 \cdot log_{10}\left(\frac{var(s)}{R \cdot 1mW}\right) = 10 \cdot log_{10}\left(\frac{\sigma^2(s)}{R \cdot 1mW}\right)$$