Simple Algorithm for Sampling Synchronization of ADCs

Waldemar G. Kürten Ihlenfeld and Michael Seckelmann

***Abstract*—This paper describes an algorithm for synchronizing analog-to-digital converters (ADCs) to the fundamental of a continuous periodic signal. It operates like a digital phase-locked loop, is based mainly on discrete Fourier transform (DFT) on sampled data, and uses DFT-leakage detection to ensure synchronous sampling. It may be applied to distorted waveforms in general, irrespective of the type of ADC used, and no windowing functions on the sampled data need to be applied. Measurements in ac metrology in the microvolt-per-volt and microradian uncertainty levels can be done without tight hardware synchronization.**

***Index Terms*—Analog-to-digital converters (ADCs), direct digital synthesis, phase-locked loop (PLL), power measurement, power quality.**

I. Introduction

THE DISCRETE and/or fast Fourier transform (DFT/FFT) has been the mainstay of digital signal processing (DSP). From extensive literature, it is well known that any continuous periodic signal can be best described (in the frequency domain) by an integer number of samples when the observation window, in which these samples are taken, encompasses exactly an integral number of periods of its fundamental frequency [1], [2]. When this condition is not met, a phenomenon of leakage occurs, because discontinuities at the endpoints of the data window cause harmonics or sidelobes in the spectrum to appear [3]. In addition, the main spectral component (i.e., the fundamental frequency) appears smeared over many bins (multiples of the frequency resolution of the DFT/FFT). Fig. 1 shows an ordinary sampling system comprising an analog-to-digital converter (ADC), a clock generator delivering the triggering or sample rate f_s for the ADC (sampling at the high-to-low edge of the trigger), and a computer (for gathering and DSP). The clock generator must be programmable and must have a high setting resolution for f_s (which is a condition easily met by the great majority of modern commercial signal synthesizers and digital sampling voltmeters). Synchronization to the unknown signal's fundamental frequency may easily be attained when only two signal parameters are initially known: 1) its bandwidth (BW, i.e., its maximum frequency content, to satisfy Nyquist's theorem) and 2) the total number of samples n that the user wants to process (which are usually power-of-two samples in the record, when using FFT). The algorithm for "frequency locking by software" herein described circumvents the use of a phase-locked loop (PLL) or any hardware locking, even if it operates like a PLL and does not preclude absolute synchronous frequency locking.

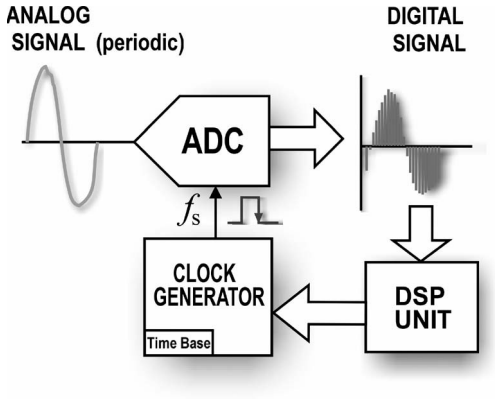

Fig. 1. Ordinary sampling system.

Section II briefly describes the algorithm. Emphasis is put on its general structure, without focusing on its intricacies, which are discussed in the subsequent sections.

Section III describes the underpinning idea of the algorithm and helps to understand the computation steps in Section II, whose background is discussed in Section IV.

Section V shows other peculiarities of the algorithm (or software) and its potentialities and shows how many problems in ac metrology can easily be handled without any hardware synchronization at all.

Section VI displays some experimental evaluations that corroborate its exceptional performance.

II. Algorithm

The principle of the algorithm can be explained as follows: Signal BW BW and the desired number of samples n are inputs to the algorithm. Consequently, a start sampling frequency (which is also a user's input) must satisfy Nyquist's criterion $f_s \geq 2 \times BW$ as the availing condition. Therefore, the ADC is triggered with frequency f_s and a record with n samples gathered by the computer.

A: Run the DFT/FFT, pick the fundamental (i.e., the spectral line at f_1 with the highest amplitude A_1), and look for its closest sidebands (those located at $f_1 \pm$ bin from f_1, i.e., $f_1 \pm f_s/n$, whose magnitudes are designated by A_{1-} and A_{1+}, respectively). Store the values A_{1-}, A_1, A_{1+}, and f_1 in record vector $\boldsymbol{R_1}$. The first run starts with index i equal 1, and future values for the fundamental and its sidebands will

Manuscript received June 6, 2008; revised September 18, 2008. First published November 25, 2008; current version published March 10, 2009. The Associate Editor coordinating the review process for this paper was Mr. Thomas Lipe.

The authors are with the Physikalisch-Technische Bundesanstalt, 38116 Braunschweig, Germany (e-mail: guilherme.ihlenfeld@ptb.de).

Digital Object Identifier 10.1109/TIM.2008.2007053

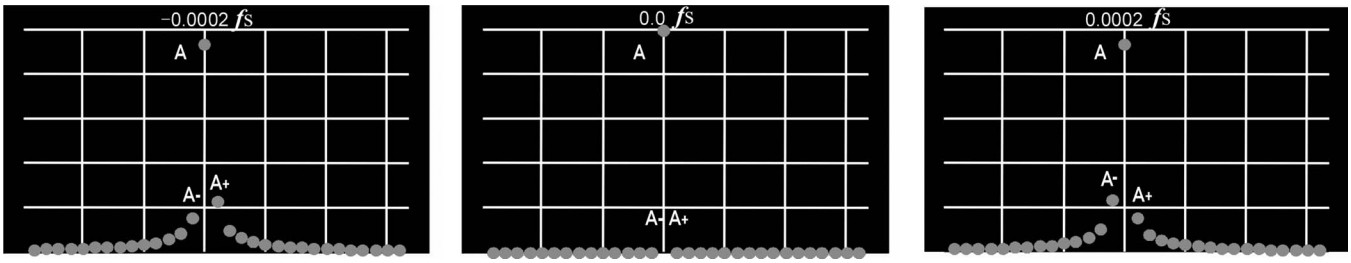

Fig. 2. DFT spectra showing lower and higher sidebands ($A-$, $A+$) near amplitude A of a sinusoidal for three sampling conditions. (Left and right) DFT leakage ($A- < A+$ and $A- > A+$) for a sampling frequency out of synchronism by ± 200 μHz/Hz. (Middle) Synchronous sampling theoretically leads to an accurate amplitude A, with $A-$ and $A+$ tending to 0.

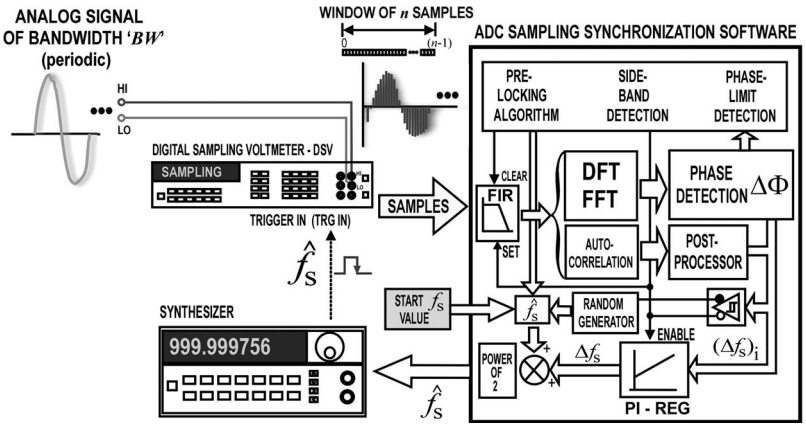

Fig. 3. Algorithm extended with auxiliary routines to improve frequency locking.

be stored in the i-indexed variables A_i, A_{i+}, and A_{i-} in vector $\boldsymbol{R_2}$.

B: If $A_{i+} > A_{i-}$, sampling frequency f_s is lower [Fig. 2 (left)] than the correct frequency [f_s at synchronism in Fig. 2 (middle)]. Thus, increment f_s by a bin $\Delta f_s = f_s/n$; otherwise, decrement it by the same amount [because it is higher than the correct frequency, as shown in Fig. 2 (right)]. Resample the signal on run i, and recalculate the A_{is}, storing them in $\boldsymbol{R_2}$. Update $\boldsymbol{R_1}$ (by $\boldsymbol{R_1} = \boldsymbol{R_2}$). Calculate difference $\Delta f_1 = f_i - f_1$ (the difference in the estimated fundamentals). If Δf_1 is greater than or equal to 0.5 f_s/n, make the new Δf_s smaller (for instance, $\Delta f_s/2$, i.e., half the previous Δf_s), resample the signal again, and update the remaining $\boldsymbol{R_1}$ in **B**. However, if

$$\Delta f_1 < 0.5 \frac{f_s}{n} \tag{1}$$

then **C** holds.

C: Split the record of $\{n\}$ samples into two sets of $\{n_1\} + \{n_2\} = \{n\}$ samples. ($n_1 = n_2$ for n power-of-two samples, although this condition is not mandatory.) Process the DFT/FFT on each set, and compute the phase difference $\Delta\phi = (\phi_{n2} - \phi_{n1})$ of the fundamental. (Computing the ratio of the DFT/FFTs of set $\{n_2\}$ to those of set $\{n_1\}$ yields the same phase difference.) The correction Δf_s to be added to f_s at run i that leads to a better estimation of f_s toward synchronization is

$$\Delta f_s = \frac{\phi_{n2} - \phi_{n1}}{2\pi} \cdot \frac{f_s^2}{n_2 f_1} \tag{2}$$

for phase difference $\Delta\phi$ expressed in radians. The signal is resampled with a new $f_s = f_s + \Delta f_s$, once again looping in **C** until $\Delta f_s/f_s < \xi$ (with ξ being a user-defined infinitesimal threshold limit, e.g., 10^{-9} or even smaller); sidebands are detected, and record $\boldsymbol{R_1}$ is updated. The convergence is fast with a few iterations. If (1) is suddenly no longer met (because of, for example, any abrupt variation or drift in the signal's frequency), the DSP should immediately jump back to step **A**, starting the locking procedure again. Under "quasi-stationary conditions" (when (1) holds), however, the program remains in **C** most of the time, and the DSP can meanwhile "*do other things in between*," because the sampling system is "locked," and f_s can continuously be tracked to the input signal. This "*continuous tracking*" can also be attained by choosing a very small infinitesimal value for ξ so that $\Delta f_s/f_s < \xi$ is never indeed satisfied because the noise in the signal imposes limits on ξ.

In the fortunate case that the first guess of f_s is appropriate so that the phase difference in **C** is smaller than π, i.e., $\Delta\phi < \pi$, steps **A** and **B** may be suppressed (or skipped), remaining in **C**.

III. Underpinning Idea Behind the Algorithm

The algorithm is intimately related to the very familiar phenomenon of the stroboscopic effect, which is mostly noticed in older movies, when a wagon wheel seems to stop, or go forward or even backward just when the rotation speed of the spokes almost matches the film frame speed (which is a typical case that exemplifies aliasing). Unless one marks a certain spoke with a different color, a wheel turning corresponding

to π rad between frames gives the impression that the wheel has stopped. A wheel turning of less than $\pm\pi$ rad allows one to discern whether it is turning forward or backward. This condition is characteristically expressed by (1). The phase shift due to a mismatch of f_s with respect to f_1 during two frames (like the spoke slipping in movies, i.e., records $\{n_1\}$ and $\{n_2\}$) is also tied to (1). The phase slipping is exactly related to f_s, which can simply be corrected as expressed in (2).

IV. Background

Step **A** can easily be understood from the definition of the DFT, which is found in extensive literature, e.g., in [2], and is well exemplified in Fig. 2. Sideband detection alone can indeed be used for frequency synchronization. However, whereas DFT/FFT amplitude estimations suffer from noise bias (for instance, ADC quantization noise), phase differences (2) are not prone to it (provided the signal's frequency remains stable) and are therefore better estimators to attain synchronization.

However, the acquisition of n samples demands some time, and any signal frequency drift during this time can only reliably be determined after data processing. Consequently, the blind application of (2) alone will always leave a frequency bias uncorrected, appearing as systematic errors on both the amplitude and phase. The fact that one wishes to adjust the sampling frequency from data acquisition and processing corresponds to closing a digital feedback loop with components displaying time constants and delay times. To attain high accuracy, this feedback loop must be optimized, and the synchronization thus becomes a digital control feedback problem. Control theory suggests using a digital proportional–integrator (PI) regulator [4], whose parameters can be normalized to the parameters of the feedback system, so that they do not require modification to deliver a fast optimum response (with no oscillations). Its integrating characteristic removes systematic deviations of frequency, thereby improving accuracy.

To further improve accuracy, the sampled data must be filtered by a finite-impulse response (FIR) filter.

Because f_s may wildly jump soon after starting the synchronization with a starting f_s value, it is necessary to deactivate the FIR filter and PI regulator to preclude oscillations. This prelocking then occurs with less accuracy for some locking attempts. The filter and regulator are then reactivated only after the Δf_s corrections stabilize. After the filter and regulator resume, the utmost quality of frequency locking can be attained.

V. Other Important Aspects

Fig. 3 shows some other modules of the synchronization algorithm, which was developed into a complete software package. A commercial digital sampling voltmeter and a frequency synthesizer were used. Direct digital synthesizers can also be employed to synthesize f_s to a high resolution (with 48 bits or more) [5].

The upper block of routines performs a prelocking (as described before), disabling (clearing) the FIR filter and PI regulator or enabling them when necessary. Some blocks accomplish tasks already described (such as DFT/FFT or phase detection). Synchronization can also take place using autocorrelation functions (by first purging all the dc components of the signal before processing it), and its results can be used for improved noise immunity. The postprocessor block derives the phase signal for the regulator from the autocorrelation. The software allows synchronization when no interharmonics synchronous to the clock generator are close to the fundamental. Interharmonic leakage interferes with sidebands, and the locking frequency may display systematic deviations, which are more pronounced on the phase of the fundamental. This is dependent on how close an interharmonic is to the fundamental. Increasing n and, thereby, the DFT resolution (bin) seems to be a promising remedy in such cases. Nevertheless, in some cases, convergence may not be attained at all, and an oscillatory pattern may arise. In this case, it is worthwhile to adjust f_s by some random amount. The block represented by the electronic symbol of a comparator, followed by a random generator, allows just a random change in f_s to be made, aiding convergence. These random variations on f_s, followed by resampling and reprocessing (DFT/FFT), provide a random frequency scanning, where, under some conditions, the sidebands become less disturbed, and the convergence can be ensured.

In the most difficult synchronization cases, the system is permitted to dynamically adapt and modify threshold convergence limits, allowing f_s to be found with less accuracy, but this recourse was never necessary in the experiments shown in this paper.

Even if the system finds the proper locking frequency, this does not mean that the number of samples per period is automatically an integer number. The block marked with "power of 2" has the task of automatically finding, in the next run (when the system is fully locked), the next lower number of samples per period of the fundamental, which is also a radix-of-two number. Meanwhile, the target number of periods of the fundamental (which is initially set by the user) is adjusted by the system after synchronization is attained. This may be valuable when evaluating power quality parameters, such as sags, dips, swells, flicker, and fluctuating quantities, aside from other quantities, such as harmonics, ratio, phase, and effective values.

A time-varying signal frequency suggests the use of a proportional–integral–derivative regulator. Better results can be attained using self-adaptive regulators, which change their parameters according to the signal's frequency variations. For extreme frequency variations (except for $\Delta\phi < \pi$, which is still inside the phase-locking range), the regulator is programmed to become of unitary gain, keeping track of the signal's frequency variations.

Modulated signals can also be measured with the software, provided the signals are prepared in the time domain prior to locking. For example, clamping the signal to a user-defined threshold may ensure synchronization in most cases. Therefore, a 50-Hz waveform modulated with 8.8 Hz (for flicker measurements) can be sampled with 32 samples per period over 125 periods of 50 Hz (or 22 periods of 8.8 Hz).

Signal noise is further compressed by the DFT/FFT according to the square root of the number of samples. However, noise with $1/f$ characteristics is still the object of investigation.

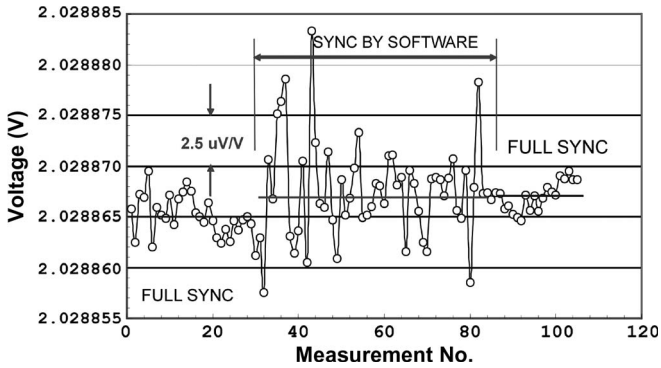

Fig. 4. Comparisons between hardware (tight synchronous sampling [5]) with and without software synchronization for a sinus signal of 62.5 Hz.

Research on uncertainty evaluations to infer the metrological capabilities of the software is being conducted and is not treated in detail here since it is out of the scope of this paper.

VI. Experimental Validation

Fig. 4 shows the results of a comparison between the proposed algorithm/software and the Physikalisch-Technische Bundesanstalt (PTB) full-synchronous sampling system, for a 62.5-Hz sinus of nearly 2 V rms. The agreement is within a few parts in 10^7 in the mean. (Any signal frequency could be used. A 62.5-Hz signal was chosen, simply because it can easily be synthesized by the PTB primary sampling standard [6].)

Fig. 5 shows the response of the software for signal frequency variations. The user started with an f_s of 1200 Hz, taking a frame (or run) of 1024 samples of a 62.5-Hz signal. In the third run, the system (in Fig. 3) locked or dynamically adjusted the observation window, with 1024 samples converging to an f_s of 1185.18 ... Hz, and remained, for some time, at this frequency but later jumped to $f_s = 1000$ Hz (for power-of-two samples per period) because of the "power-of-two" searcher algorithm. No variations on the effective value (determined from the samples) during this frequency jump occurred. On run 23, a sudden change in signal frequency (ranging from 62.5 to 72.5 Hz, i.e., 10-Hz variation) was externally intentionally caused. The PI regulator then reacted, and one run later, the system locked itself on 1003.24 ... Hz, remaining at this frequency. The voltage variation recovered to its original value within a few parts in 10^6 one run later. On run 35, it finds power-of-two samples per period (as requested), with f_s located at 1160 Hz.

Experiments conducted so far show that locking quartz-based sources within a few parts in 10^8 with this software is readily feasible. Fig. 6 shows the sampling frequency stability measured with the frequency-locking software. The slight variations thereof are indicative of how much both quartz time-based references deviate from one another over time. A low-noise sinus synthesizer (DS360 from Stanford Research), which is normally employed for characterizing ADCs, was used.

Another measurement shows the output voltage stability (Fig. 7) of the same synthesizer over almost 1 h. The measured total harmonic distortion was below -110 dBc (neglecting the components of the power-line frequency), indicating that

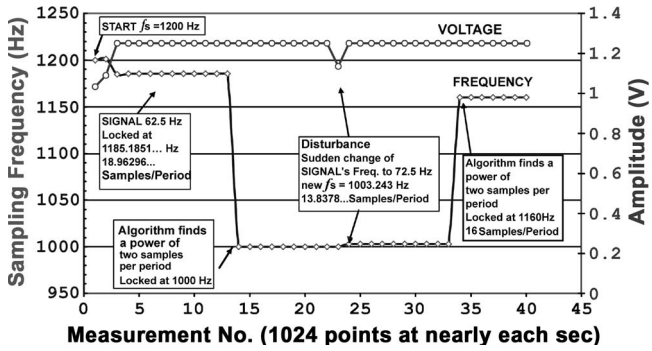

Fig. 5. Response of the software to a perturbation in the signal frequency for a sinusoidal signal of 62.5 Hz.

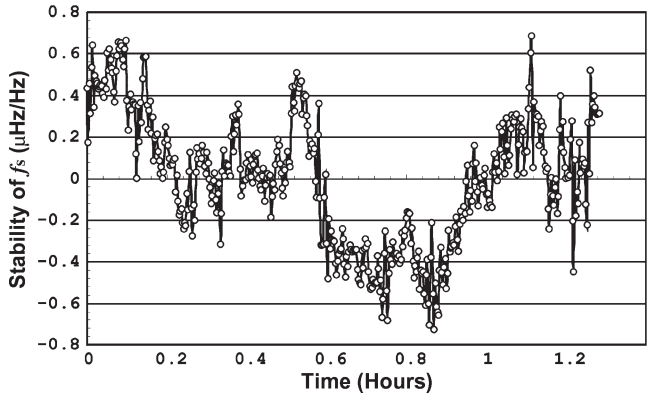

Fig. 6. Locking frequency f_s stability for a low-noise sinus synthesizer using the proposed algorithm.

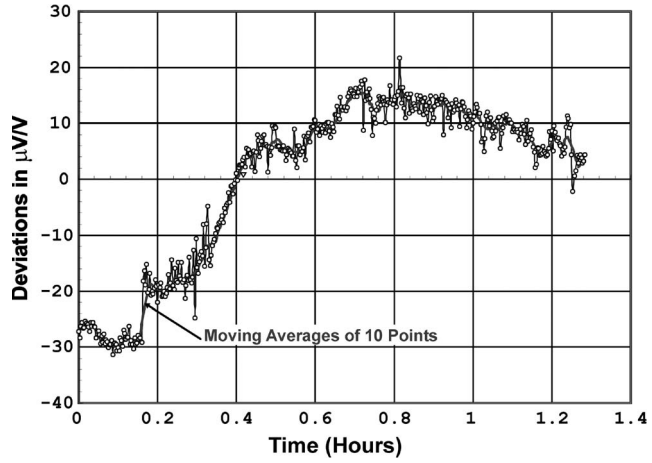

Fig. 7. Amplitude stability over time for a low-noise sinus synthesizer.

the software may be used to characterize ADCs as well (as shown in Fig. 8), without applying windowing functions for eliminating sidelobes. However, characterizing high-resolution ADCs may require the use of additional filtering to improve measurement quality.

Measurements of the ratio and amplitude at 1 kHz were also done with this software and the PTB primary sampling standard, indicating deviations of smaller than 2 μV/V for amplitude and 10 μrad for phase. Fig. 9 shows the results for 62.5 Hz, demonstrating that the ac ratio measurements from the

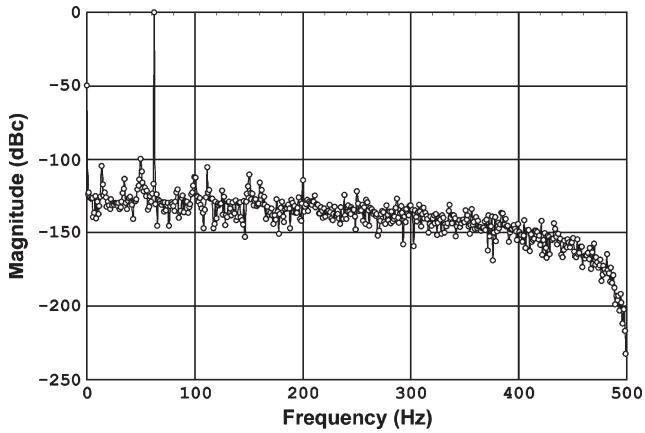

Fig. 8. Harmonic distortions of a particular ADC characterized with a high-purity sinus synthesizer of below −100 dBc.

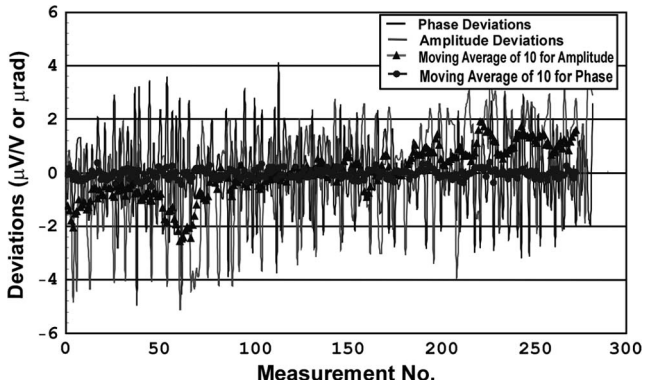

Fig. 9. Locking stability for (circle) phase and (triangle) amplitude using an ac source of the PTB primary ac sampling power standard.

sampled data of two signals can be done with synchronization by software with one of the signals.

VII. Conclusion and Outlook

This simple and elegant algorithm operates on DFT-leakage detection and leads to a fast ADC synchronization, circumventing the use of complex synchronization hardware (except for the need for a common-frequency synthesizer). High accuracy for amplitude and phase (of some parts in 10^6) can be attained, and virtually any number of ac quantities can easily be determined (for instance, RMS, phase, ratio, and ac power).

Furthermore, it is of utmost practical importance, because it can be employed in the field (where sampling synchronization is a real problem), allowing ac sampling with very high accuracy and measurement uncertainties of as low as those of the PTB primary ac sampling system [6].

Other transformations are being investigated to allow for improvements in frequency locking and noise mitigation, including power quality routines.

Acknowledgment

Readers interested in the software are encouraged to contact the author.

References

[1] W. G. Kürten Ihlenfeld and M. Seckelmann, "Simple algorithm for sampling synchronization of ADCs," in *Proc. CPEM*, Broomfield, CO, 2008, pp. 676–677.

[2] A. V. Oppenheim and R. W. Shafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice–Hall, 1989, ch. 2–3, pp. 8–131.

[3] W. Kester, *Mixed-Signal and DSP Design Techniques*. Norwood, MA: Analog Devices, 2000, ch. 5, pp. 1–25.

[4] W. Leonhard, *Digitale Signalverarbeitung in der Mess- und Regelungstechnik*. Stuttgart, Germany: B. G. Teubner, 1989, ch. 8.

[5] W. G. Kürten Ihlenfeld, K. Dauke, A. Suchy, and P. Räther, "Three-phase primary ac power sampling standard with improved frequency resolution," in *Proc. CPEM*, Broomfield, CO, 2008, pp. 486–487.

[6] G. Ramm, H. Moser, and A. Braun, "A new scheme for generating and measuring active, reactive, and apparent power at power frequencies with uncertainties of 2.5×10^{-6}," *IEEE Trans. Instrum. Meas.*, vol. 48, no. 2, pp. 422–426, Apr. 1999.

Waldemar G. Kürten Ihlenfeld was born in União da Vitória, Brazil, in 1960. He received the B.Eng. degree from the Federal University of Paraná, Curitiba, Brazil, in 1983 and the M.Eng. and Ph.D. (*magna cum laude*) degrees in electrical engineering from Braunschweig Technical University, Braunschweig, Germany, in 1994 and 1997, respectively.

From 1983 to 1999, he was with the Central Laboratory–LAC, which is a research institute of the Federal University of Paraná and the power utility company of Paraná (south of Brazil). In 1999, he joined the Physikalisch-Technische Bundesanstalt (PTB), Braunschweig, where he has been developing precision electronic circuits for ac power measurements and mathematical modeling of sampling systems.

Michael Seckelmann was born in Braunschweig, Germany, in 1976. He received vocational training as a "Physiklaborant" from the Physikalisch-Technische Bundesanstalt (PTB), Braunschweig, in 1996. He is currently working toward the Dipl.-Ing. FH degree in electrical engineering from the Anhalt University of Applied Sciences, Koethen, Germany.

After his military service, he continued his education and became a software developer and information technology communication consultant. In 2001, he joined the Department of Mechanics and Acoustics, PTB. Since 2004, he has been with the Department of Electrical Energy Measurements, PTB.