IEEE Instrumentation and Measurement
Technology Conference
Budapest, Hungary, May 21–23, 2001

Standard Framework for IEEE-STD-1241 in MATLAB

János Márkus and István Kollár
Department of Measurement and Information Systems
Budapest University of Technology and Economics, Budapest, Hungary, H-1521
Phone: +36 1 463-3596, Fax: +36 1 463-4112
E-mail: [markus,kollar]@mit.bme.hu, URL: http://www.mit.bme.hu/[˜markus,˜kollar]

Abstract ***– The approval of the 1241 standard means that all users and manufacturers of analog-to-digital converters should use the terminology and test methods described in the standard. However it is not an easy task to leave an already developed, tested and used environment for a new one.***

To inspire the users to use the standard, a program has been developed, which realizes most of the described algorithms. The program is also a testbed of new ideas, because it is very easy to extend the existing code with new algorithms and to compare the results to the standard using the same input and the same precision.

The program has been written in MATLAB, and for easy of usage it provides a graphical user interface. It is also flexible enough to support different input/output file formats.

The program is available through the Internet.

Keywords ***– ADC testing, IEEE-STD-1241, standardization, sine wave test method, windowing, MATLAB, data acquisition.***

I. INTRODUCTION

A. Standardization projects of ADC testing

As digital signal processing techniques become more and more efficient, the demand for better analog-to-digital converters (ADCs) is also growing. Therefore universal terminology and test methods of ADCs are also desirable. On the other hand, the currently available official standard [1] covers only quazi-static operations, while most of the fields of digital signal processing require testing under dynamic conditions.

In 1996, IEEE IMS TC-10 started a new project (IEEE-STD-1241 [2]) to address the above mentioned problem. The project, whose aim is to standardize the terminology and test methods for analog to digital converters, builds upon the already approved standard of digitizing waveform recorders [3].

Since the ADC market is world-wide, it is important to speak a universal language in the topics related to ADCs. We mention that there are two more committees in Europe (EUPAS[1] [4] and DYNAD[2] [5]), working closely together with IEEE IMS TC-10.

The research described here has been supported by the Hungarian Fund for Scientific Research, under contract No. OTKA T 033 053.

[1] EUropean Project for ADC-based devices Standardization

[2] DYNamic characterization and testing of Analogue-to-Digital converters

By the time of the conference the standard will hopefully be through the balloting phase. One TC-10 member is currently reformatting it into an IEC-acceptable format, in order to become sooner or later also an IEC standard [2].

B. Claim for a Standard Program

A true analog-to-digital converter which discretizes the input signal in time and amplitude suffers from non-idealities. There are two main categories of these errors: the nonlinear behavior of the device (the distortion) and the noise. To examine the dynamic behavior of a real ADC, several testing methods were worked out in the 80's by different manufacturers (a good summary can be found in [6]). The main test set-ups of the new standard are also based on these algorithms.

The basic steps to characterize an ADC are as following:

- setting up the device (power supply, reference, control and clock signals),
- applying an input signal,
- collecting sampled and quantized data into a computer file,
- calculating standardized parameters using off-line algorithms.

The aim of the standard is to allow anybody to check an A/D converter by means of the same test setup and get the same result. The first 3 steps of a measurement can be repeated maintaining some parameters (e.g. generator type and settings); here the last one will be examined in detail.

To calculate certain well known parameters, the standard suggests to use either one-step, or iterative algorithms. Studying them, it can be quickly concluded that even the one-step algorithms may have different implementations. Even though these algorithms have a closed-form solution, the result can be slightly different, because of the different number representations (e.g. fixed or floating point, single or double precision numbers) and calculation methods (e.g. the solution of linear equations).

The problem is even more complicated using iterative algorithms. An example of such algorithms is the 4-parameter sine wave fitting [2]. In general, the result depends on the estimated initial value, and during the iteration the round-off errors can also play an important role. The standard suggests to use matrix inversion techniques during this iteration, which also has several different numerical implementations.

In most cases this is not noticeable by the user, but in special cases (when the iterative algorithms may have local minima) users with different programs may conclude with different results, even using exactly the same data.

Furthermore, the currently available version of the standard [2] contains no exact suggestions about the selection of initial values or the stop criterion of the four-parameter fitting. This means that using different values for the non-specified part of the standard may cause different results. This is also a problem in the case when new ideas of researchers are tested and compared to the existing standard.

These problems can be solved by strictly defining the algorithms, which is very cumbersome both for the groups working on the standard and for the users and manufacturers who want to use it. Another possibility is to provide standard programs, which can be used by anybody for data processing or for comparison. Additionally, such a standard program is more reliable than the programs developed individually by the users because many people will use, test and debug it.

C. Requirements

The standard program to be realized has to meet several requirements. First of all, it has to be able to calculate all the algorithms described in the standard, and the realized algorithms have to be documented in detail.

To maximize the number of the users who will use the standard, the program has to be

- available to anybody;
- reliable;
- realized in an environment which is available and known by most of the users;
- user-friendly.

Using the program for quality management, it has to support (semi)automatic processing, and has to be flexible enough to load input files with different formats.

D. Selection of the Programming Language

As a programming language for the purpose described above, the quasi-standard engineering tool, MATLAB can be chosen. This choice has the following advantages:

- it is a numerical software realized on several platforms, used by many labs and universities;
- for all calculations it uses the IEEE double precision floating point number representation (on all platforms);
- matrix and vector operations, DFT and other engineering calculations are fast, because they are realized internally in C;
- the source code of programs written in M-files can be examined by the user;
- it is rather easy to extend the existing code;
- it can be written to be easy-to-use, if based on the graphical user interface;
- it supports data acquisition to collect the test data (data acquisition toolbox);
- we have experience programming it.

The test data evaluation program is available through the Internet [7].

E. Preliminaries

In 1999, a LabView sine wave test program was announced by a workgroup in IEEE IMS TC-10 [8]. The program was originally developed to test waveform recorders, but as the two standards use the same algorithm for sine wave fitting, it can be used also to test Analog-to-Digital Converters. It implements the 4 parameter sine wave fitting procedure. Because it does not support the 3-parameter sine wave fitting, it may give worse result as this latter one when using exact coherent sampling.

After the calculation, it displays some of the calculated parameters determined from the fit (sine wave parameters) and from the residuals (noise parameters), and also displays some additional plots for visual evaluation.

We have already publicized a MATLAB program, which is graphically and algorithmically equivalent to the existing LabView program [9]. The program described here is a comprehensive extension of these, considering every relevant demand.

II. THE FRAMEWORK

The aim of the project was to develop a program which implements all the tests and calculates all the parameters which can be derived from measured data. Making this program available to the public will allow both manufacturers and users to produce the same result starting from the same data.

To make the program more useful, it has five different modes.

A. Compatible Mode

In this mode the software uses the same algorithm and the same user interface as the LabView program mentioned above.

For more information about this mode, see [8]. An example of the output window of the program in this mode can be seen in Fig. 1.

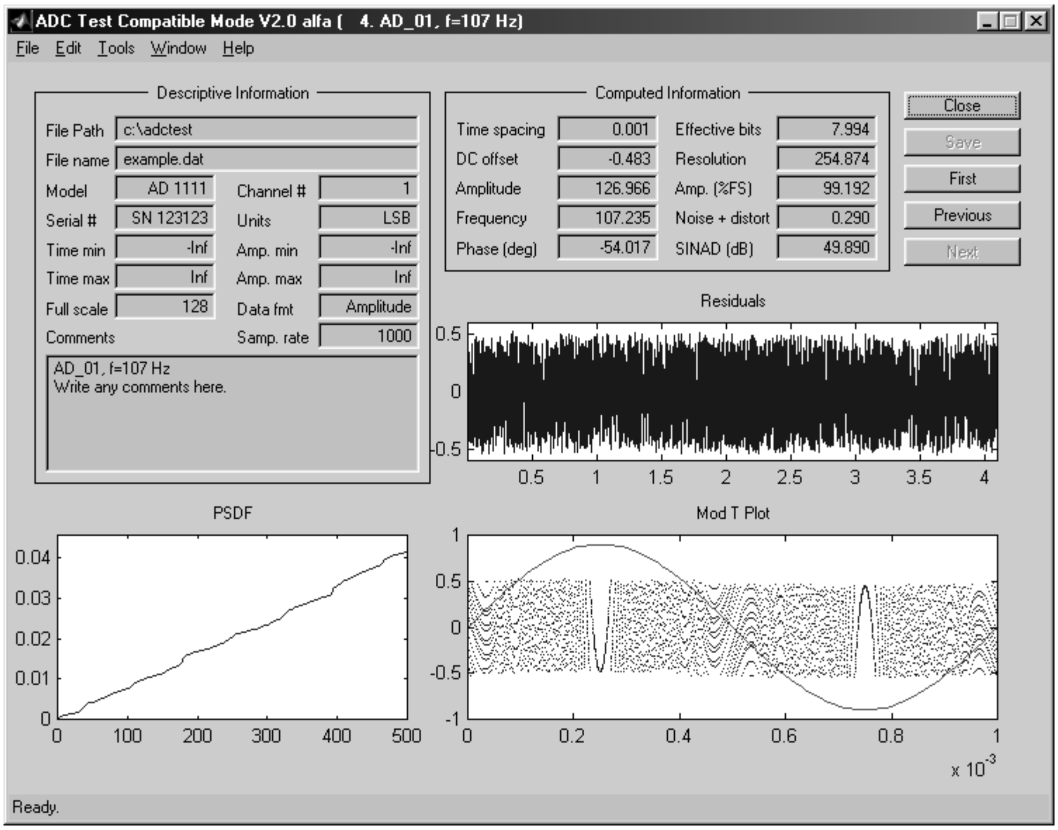

Figure 1. The output window of the program in compatible mode

B. The IEEE-STD-1241 Mode

Using this mode, the software uses only the algorithms described in the standard. Since most of the tests in the standards are based on sine wave fitting and the histogram test, these are the most important features to realize. The following list gives the developed and tested algorithms in the program (the abbreviations and section references are from the standard [2]):

- Curve fitting methods (sine wave fitting);
  3-parameter (known frequency [4.1.4.1]) and 4-parameter (general use [4.1.4.3]) Least Squares (LS) fit to sine wave data, using matrix operations.
  These methods are usually used to get the noise parameters of the device under test (SINAD and effective bits [4.5]) and in some other tests such as measuring the bandwidth of the converter [4.7.1].
- DFT (Discrete Fourier Transformation) methods [4.1.5];
  Calculating averaged DFT, one can calculate the Total Harmonic Distortion (THD [4.4.5]), using coherent sampling, when windowing is not needed. Similarly, the Total Spurious Distortion (TSD, [4.4.5]) can be calculated. Using these calculations, the Spurious-Free Dynamic Range (SFDR, [4.4.5.3]) and the Signal To Non-Harmonic Ratio [4.5.1.3] can be displayed using DFT methods.
  DFT methods are also used for testing intermodulation distortion [4.4.6] and for the noise power ratio measurements [4.5.4].
  As window types and usage is not standardized, windowed processing and window-selection is available in advanced mode, see Sec. II-D.
- Nonlinearity, Transfer Curve;
  To locate the code transitions [4.1.6] histogram testing is suggested by the standard as a dynamic test. The most prevalent method for this test is to apply a sine-wave input signal which slightly overdrives the input of the device under test [4.1.6.3].
  The program retrieves the parameters of the input sine-wave using the modified sine-wave curve fitting described in [10]. Then it checks the frequency, the amount of overdrive, and also the number of samples to validate the measurements.
  From the samples and their histogram, the INL [4.4.1] and DNL [4.4.2] can be easily calculated and displayed.
- Step response;
  The step response test [4.6] can be performed using the program. Some graphical properties can be displayed when choosing the graphical mode (see in Sec. II-C).

- Frequency response [4.7.1];
  Frequency response of the ADC can be calculated from the differential (strictly taking the differences) of the step response via DFT method (this is an alternative method against sine wave test).

The algorithms and all non-determined steps of the standard are fully and carefully documented in the documentation and in the source code of the program.

C. The Graphical Mode

While the IEEE-STD-1241 mode (see previous section) is designed for fast and automatic processing, it is sometimes worth-while to make visual evaluations and/or comparisons between two or more tests.

Hence, the graphical mode is an extended version of the standard mode. This means that additional plots can be displayed depending on the test method.

Using sine wave fitting, the following plots help discover the main error characteristic of the A/D converter:

- the residual vector, which is the difference of the samples and the fitted sine wave (using a large record size, there is not much information: see in Fig. 1).
- the Modulo Time Plot, which shows the residuals as a function of the fitted sine wave phase instead of the time (the horizontal axis is wrapped by the period length). One example can be seen in Fig. 1. Using this plot, appearance of harmonic distortion (as patterns, correlated with the base harmonic in the figure) or jitter (as noise, proportional to the first derivative of the signal) can be easily verified. For more information see [11].
- the Power Spectral Density and Distribution Function can be used for the same purpose. J. Blair [12] suggested to use the latter one. The distribution function of an ideal ADC is a straight sloping line. The harmonic and spurious distortions can be seen as discontinuities in this line. The jitter (aperture uncertainty) perturbs the line around the first harmonic. An example of an ideal ADC can be seen in Fig. 1.

If the selected test method uses a windowed DFT, a useful tool is to view the selected window and its (interpolated) DFT (to see the leakage reduction, mainlobe behaviour and other properties).

By measuring linearity, the ideal and measured transfer curve, and the INL and the DNL can be visualized.

Testing the step response of the ADC, the (properly delayed) ideal and measured step responses can be seen, which immediately shows the order of overshoot and precursor. Using this measurement setup, the impulse response and its DFT can be selected.

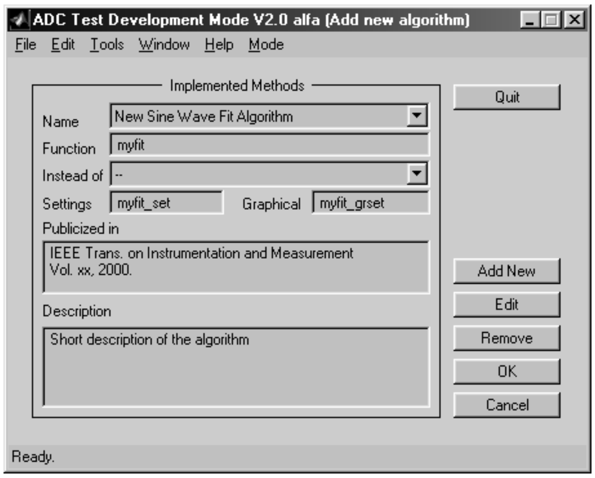

Figure 2. Setting new algorithm in the development mode

D. The Advanced and Development Mode

The draft standard describes several test methods for characterizing an A/D converter. Since its first version, several new test methods have been researched. Some of them use more accurate models and therefore give more accurate results than the standard. To compare these new methods to the existing standard, we have developed a mode (Advanced mode) in which custom tests can be executed.

To initialize a custom test, the researcher must first switch the program into the development mode. In this mode he is able to add his own function which calculates and displays the adequate parameters. The program communicates with these functions through a parameter structure which contains all necessary parameters (could be easily extended by the researcher). A development screen can be seen in Fig. 2.

Studying the literature, there are several concepts which would be useful to be integrated into this environment:

- frequency domain calculations: better frequency selection [13] and usage of minimal side-lobe energy windows [14];
- ADC modeling and sine wave fitting: System Identification for Data Acquisition Characterization (SIDAC) project by Schoukens et al. [15] and Zhang's method [16];
- selected algorithms from the DYNAD [5] project;
- linearity check as suggested by Arpaia [17] and/or by Adamo [18];

To integrate the researched methods in this framework has another advantage. It is that newly researched algorithms and test methods can be realized and compared to the existing standard within the same common program. It causes that sooner or later the best new algorithms could become a part of the existing standard. Using this kind of development, this program can be an example of international co-operation in our field.

III. INTERFACES

To achieve the goal of making a standard program which is used by everybody, the software need to be more than precise. It needs to support also the different preferences and the different input/output files. In the following two sections some general and applied considerations will be taken.

A. *Interface to the users*

Few people enjoy reading long help documnets in order to use a program. Many people simply start the program, and proceed without instruction. If it is not working as they think, then they conclude that the program is not good enough or the programmer has an unfollowable thought.

To make the program more usable, all parameters can be set on the graphical user interface. A sample window in compatible mode can be seen in Fig. 3.

After a few uses of the software it is rather exhausting to repeatedly set all parameters for every measurement. There is also the possibility that someone will make a mistake while trying to repeat a test and will end with a different result.

To eliminate these problems, the program supports descriptor files which can be generated manually or automatically. These standard ASCII files can be edited easily. An example of such a file can be seen in Fig. 4. Each file begins with the `Mode` parameter, and the beginning of a new test setup (which defines a new descriptor) is signed by either a `Mode` or a `Test` keyword. In the test file it is enough to mark only the differences of the tests, moreover, wildcards could be also used in the file names. Using this method, the automatic processing of many measurements (such as at quality control) becomes very easy.

Another advantage of using these descriptor files is the self-documentation of the measurements. Using these files, the measurements are repeatable in every environment, and users can get exactly the same results.

B. *Interface to the measurements and other programs*

In addition to the graphical user interface, the usability of the program considerably depends also on file handling. The larger the number of the measuring devices/softwares, the larger the number of different data file formats. The test program needs to be flexible enough from this point of view.

Since the program is developed for use on several platforms, it supports standard ASCII data files (which can be completed with descriptor files mentioned above). It support fixed point, floating point, and also hexadecimal representation of the input data. This latter format is suitable for processing uploaded memory fields without editing the source file manually.

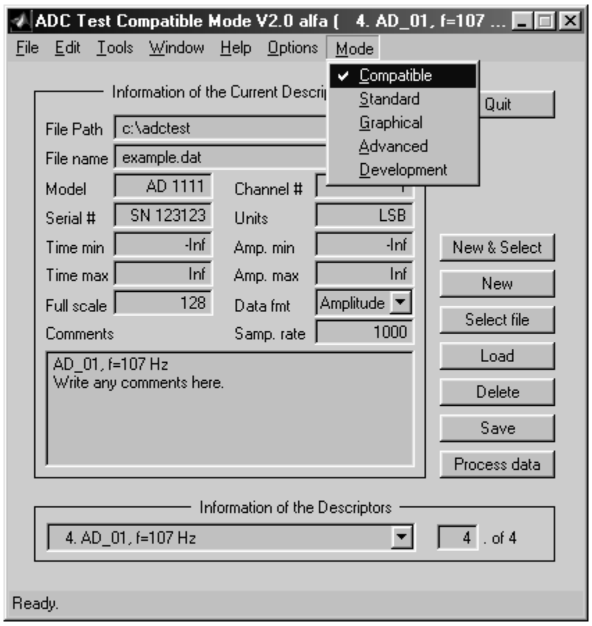

Figure 3. The descriptor's window in compatible mode

In the non-compatible modes the option of using a user-defined input processing function is also available. In this case arbitrary files can be processed, such as standard wav-files on a PC.

The calculated result files also have several formats. The calculated vectors (such as the residuals, power spectral distribution function, INL, DNL) can be saved either into ASCII files or into standard MATLAB mat-files. The calculated single values are put into a result file which has the same format as the descriptor files. All calculated parameters can be processed using a user-defined output function. For more details on the specification of the input and output, see the program user manual [7].

IV. NUMERICAL ASPECTS

The program uses the IEEE double precision numbers for calculations, as is standard in MATLAB. This is currently the most precise widely available number representation for scientific computations.

In the standard, many calculations are suggested to be executed in matrix format. For efficiency in speed and precision of computation, the matrix and vector commands supported by the core of MATLAB are used.

For more precise solution, Singular Value Decomposition (SDV) is used in the steps of algorithms such as to get the solution of an overdetermined linear equation set.

```
% Mode          compatible
% Test          sinewave
% Name          AD_01, f=107 Hz

% Comments
% Write any comments here.

% FilePath      C:\adctest
% FileName      example.dat

% Model         AD 1111
% Serial        SN 123123

[...]

% FullScale     256

% Channel       1
% Units         Volt
% DataFormat    amplitude
% SampleRate    1000
```

Figure 4. A descriptor file

Currently in the 4-parameter sine wave fitting (which is an iterative algorithm), the software uses Newton's method as it is described in the standard. There are faster and/or more precise methods for the same purpose [19]. The advanced mode can be used to test these other methods.

V. RESULTS AND FUTURE WORK

This paper has described a standard program [7] related to ADC testing. To make such a program has several advantages. First, everybody can use the same algorithm for testing, no differences are caused by dissimilar realizations. Second, a standard program specifies the non-definite details of the standard. Third, the manufacturer can converge to the standard methods using the program. Furthermore, if this program gets widely used, it will become more and more reliable because many users will test and comment it.

Moreover, as this program has the advanced and development mode, new algorithms and test methods can also be realized and compared to the standard in the same environment, on the same data. If the program becomes widely used, eventually the best methods will migrate into the standard. This program can be an international test-bed of new ideas in the field of measurement.

In the near future the members of IEEE IMS TC-10, EUPAS and also DYNAD will be asked to test and comment this program. In addition it will also be worthwhile to extend the standard program with new algorithms related to ADC-testing developed by either these committees or other researchers.

REFERENCES

[1] IEC 60748-4, *Semiconductor Devices – Integrated Circuits – Part 4: Interface integrated circuits – Sec. 2: Blank detail specification for linear analogue-to-digital converters*, 2 edition, 1997.

[2] IEEE TC-10, *IEEE Std 1241 Draft – Standard for Terminology and Test Methods for Analog-to-Digital Converters*, Version VS022500, February 2000, URL: http://grouper.ieee.org/groups/1241/.

[3] IEEE TC-10, *IEEE Std 1057-94 – IEEE Standard for Digitizing Waveform Recorders*, December 1994, URL: http://grouper.ieee.org/groups/1057/.

[4] *EUPAS – European Project for ADC-based Devices Standardization*, URL: http://elve.le.ttu.ee/mesel_www_home/R&D/ADC/EUPAS.HTM.

[5] C. Morandi et al., "DYNAD: a framework IV SMT project addressed to the development of dynamic test techniques for analog-to-digital converters," *Computer Standards & Interfaces*, vol. 22, no. 2, pp. 113–119, June 2000, URL: http://www.fe.up.pt/˜hsm/dynad/.

[6] Bruce E. Peetz, "Dynamic testing of waveform recorders," *IEEE Trans. on Instrumentation and Measurement*, vol. 32, no. 1, pp. 12–17, March 1983.

[7] J. Márkus, *'ADC Test Data Evaluation Program for Matlab' Home Page*, URL: http://www.mit.bme.hu/services/ieee/ADC-test.

[8] J. J. Blair, "Sine-fitting software for IEEE standards 1057 and 1241," in *Proc. of the 16th IEEE Instr. and Meas. Technology Conference*, IMTC/99, Venice, Italy, May 1999, vol. 3, pp. 1504–1506.

[9] J. Márkus and I. Kollár, "Standard environment for the sine wave test of ADC's," Submitted to the Measurement Journal of IMEKO, February 2001.

[10] N. Giaquinto and A. Trotta, "Fast and accurate ADC testing via an enhanced sine wave fitting algorithm," *IEEE Trans. on Instrumentation and Measurement*, vol. 46, no. 4, pp. 1020–1024, August 1997.

[11] F. H. Irons and D. M. Hummels, "The modulo time plot – a useful data acquisition diagnostic tool," *IEEE Trans. on Instrumentation and Measurement*, vol. 45, no. 3, pp. 734–738, June 1996.

[12] J. J. Blair, "A method for characterizing waveform recorder errors using the power spectral distribution," *IEEE Trans. on Instrumentation and Measurement*, vol. 41, no. 5, pp. 604–610, October 1992.

[13] I. Kollár, "Evaluation of sine wave tests of ADCs from windowed data," *Computer Standards & Interfaces*, vol. 22, no. 4, pp. 261–268, October 2000.

[14] M. Bertocco et al., "Windows for ADC dynamic testing via frequency-domain analysis," in *Proc. of the 17th IEEE Instr. and Meas. Technology Conference*, IMTC/2000, Baltimore, Maryland USA, May 2000, vol. 1, pp. 114–118.

[15] G. Vandersteen, Y. Rolain, and J. Schoukens, "System identification for data acquisition characterization," in *Proc. of the 15th IEEE Instr. and Meas. Technology Conference*, IMTC/98, St. Paul, Minnesota USA, May 1998, vol. 2, pp. 1198–1202.

[16] J. Q. Zhang and S. J. Ovaska, "ADC characterization by an eigenvalue method," in *Proc. of the 15th IEEE Instr. and Meas. Technology Conference*, IMTC/98, St. Paul, Minnesota USA, May 1998, vol. 2, pp. 1198–1202.

[17] P. Arpaia et al., "ADC testing based on IEEE 1057-94 standard – some critical notes," in *Proc. of the 17th IEEE Instr. and Meas. Technology Conference*, IMTC/2000, Baltimore, Maryland USA, May 2000, vol. 1, pp. 119–124.

[18] F. Adamo, F. Attivissimo, and N. Giaquinto, "Measurement of ADC integral nonlinearity via DFT," in *Proc. of the XVIth IMEKO World Congress*, IMEKO 2000, EWADC, Vienna, Austria, September 2000, vol. X, pp. 3–8.

[19] T. R. McComb, J. Kuffel, and B. C. le Roux, "A comparative evaluation of some practical algorithms used in the effective bits test of waveform recorders," *IEEE Trans. on Instrumentation and Measurement*, vol. 38, no. 1, pp. 37–42, February 1989.