medicinska revija
medical review

UDK: 519.612
ID = 176479244

Bajčeta S. et al Med Data Rev 2010; 2 (2): 107-110 *MEDICAL DATA/Vol. 2. No2 / Jun 2010.*

*Originalni članci/*
*Original articles*

# THE PROCEDURE FOR TRANSFORMATION OF REAL-TIME NUMERICAL ALGORITHMS INTO VARIABLE PRECISION ONES

# NUMERIČKI ALGORITMI ZA TRANSFORMACIJU U REALNOM VREMENU

**Correspondence to**:

Senka Bajčeta, High School "Zarko Zrenjanin", Vrbas, Serbia

21460 Vrbas,
Save Kovačevića 91

e-mail: senca@sbb.rs
bojanvuj@uns.ac.rs, dragan.kovacevic@ieent.org, vujicicv@uns.ac.rs

Senka Bajčeta[1] Bojan Vujičić, Dragan Kovačević[3] and Vladimir Vujičić[2]

[1] High School "Zarko Zrenjanin", Vrbas, Serbia; [2] Faculty of Technical Sciences, University of Novi Sad, Chair of Electrical Measurements, Novi Sad, Serbia; [3] Institute "Nikola Tesla", Belgrade, Serbia

*Ključne reči*
Numerički algoritam, EEG signal, softverska platforma

*Key words*
numerical algorithm, EEG signal, software platform

*Abstract*

The paper addresses the issue of transformation of an arbitrary real-time numerical algorithm into variable precision one. All the details of the solution procedure are given in the paper. Its application to the selection of hardware and software platform, for realization of arbitrary scientific and technical problems, is described. The case study deals with the optimal use of CELL processor hardware and software platform in case of measurement of EEG signal which belongs to the class of non stationary signals.

## *NOMENCLATURE*

| | |
|---|---|
| $\Gamma_{max}$ | Upper limit of the relative error in representing data. |
| $\lvert\Delta x\rvert_{max}$ | Distance between two adjacent floating-point numbers. |
| $x_{min}$ | Minimum mantissa's value. |
| $t$ | Number of mantissa's digits in host computer. |
| $x$ | Real data. |
| $x_d$ | Floating or fixed-point simulated data |
| $2^L$ | Lower limit of number range (diade) to which $x_c$ belongs. |
| $L$ | Position of lower limit $2^L$ |
| $B, B_1$ | Loop counters. |
| $m$ | Total number of fixed-point digits, the number of simulated mantissa's digits. |
| $n$ | Number of digits after the decimal point in the fixed-point data. |
| $x_c$ | Real floating-point data. |
| $\varphi_d$ | Operator determining the simulated number of digits for positive real data. |
| $x_1, x_2, a, b, c, d$ | Real numbers. |
| $z_d$ | Simulated value of the arithmetic expression. |
| $2^q$ | Maximum simulated binary exponent. |
| $\lvert MaxExp\rvert$ | Maximum absolute value of the exponent in the host computer. |
| $s$ | Control logical variable. |
| $\psi_d$ | General conversion algorithm. |
| $\Gamma$ | Relative quantization error. |
| $z$ | Exact value of arithmetic expression. |

## *I INTRODUCTION*

In [1] and [2] the idea for realization of the variable accuracy of numerical algorithms in high-level programming languages, that do not have bitwise operations in its command set, is presented. At that time, most commonly used high-level programming languages were FORTRAN and PASCAL. Even then there was a lot of software written in those programming languages, especially in FORTRAN, and it was extensively used in all fields of science and technology.

On the other hand, the form of representation of numerical data was not standardized. Moreover, different number bases for the presentation of numerical data in computers - were used: a) binary, b) octal, c) decimal d) hexadecimal.

The paper [3] shows that, from the point of view of presentation error and processing numerical data error, the binary number base represents the optimal solution. Later, the development of standards for the presentation of numerical data has confirmed these results [4].

In such circumstances the method described in [1] was welcome and effective enough. It represented necessary and sufficient universal solution.

Today, situation is significantly different: the way of presentation of numerical data in a computer is standardized [4]. Current programming languages, above all C, have bitwise operations, so it appears that the algorithm is not needed anymore. On the other hand, in the design of modern measuring instrumentation, important place is reserved for blocks for electronic measurement and signal processing. Electronic measuring block consists of A / D converter, one or more of them, while a processing block consists of processor, one or more of them. Both blocks are selected depending on the project parameters - speed, precision and accuracy. In both cases there is a wide choice for these components. This paper deals with choice of above hardware components. Hardware that is selected has to meet necessary and sufficient conditions for the efficient operation of the designed instrument.

## *II. OPTIMAL CHOICE PROBLEM*

Today there is large amount of software written in different languages and variety of customized hardware and software platforms. So, there are wide possibilities for choice both for hardware and software, and often a unique hardware-software platform. The issue of optimal selection of software and hardware is a serious practical problem.

### *A. Solution Proposal for Optimal Choice Problem*

When designing an instrument the following steps need to be taken:

- the algorithm, or, generally, software, that realizes necessary functions of an instrument, written in a higher level programming language (such as DELPHI) has to be chosen,
- that algorithm is then transformed into an algorithm with variable precision. Then input data resolution (resolution of an A/D converter), resolution of processing device (INT – integer, FXP – fixed-point, FP – floating-point) and output resolution (resolution of a D/A converter) are varied.

- Resolutions are varied in defined limits, in order to meet the requirements of accuracy and precision for
  - D/A converter,
  - A/D converter,
  - The INT, FXP or FP arithmetic, and corresponding resolutions are chosen in order not to corrupt the requirements for accuracy and precision.
- When the optimal resolutions of D/A, A/D and INT, FXP or FP are determined, hardware that meets all requirements, including speed, is chosen.

The problem can alternatively be solved in MATLAB or C with the proper embedded tools. The problem occurs if the algorithm is too large and complex, for example, FFT algorithm for measuring harmonics, written in DELPHI, FORTRAN or other languages. This algorithm must be translated into MATLAB or C. This can be extensive work with a high probability for errors that are difficult to detect and correct. For example, if instead of a ”+” sign ”-” sign is typed, in an arithmetic expression, it is still a regular and formally correct arithmetic expression, but the error can not be detected by any debugger, any tool from the standard text editor or, in general, there is no formal way to detect and correct such errors. Testing of such algorithms by input test data is not sufficiently reliable method, but may indicate that an error exists. This alternative is therefore not recommended.

## III. IMPLEMENTATION OF THE TRANSFORMATION PROCEDURE

### *A. Problem Definition*

The method described here belongs to the class of simulation methods. The simulation is performed by the ?oating-point computer with long mantissa of real data. The upper limit of the relative error in representing data in such a computer is:

$$\Gamma_{\max} = \frac{|\Delta x_{\max}|}{x_{\min}} = 2^{1-t} \qquad (1)$$

where $t$ is the number of the digits in mantissa. Even when $t \geq 32$, it is possible to consider real data $x$ as continuous. As it is stated in [5], [6], the problem of predetermination of the quantization error is nonlinear and exact mathematical analysis is very difficult, even in the case of a simple sinusoidal input and uniform A/D converter [7]. It seems that the only applicable method in every particular case is simulation. The problem is how to simulate measurement processes in a given instrument or a measurement system with the same input and different choices of A/D converter's, D/A converter's and numeric processor's word lengths, in order to find the optimal ones. The main role in the solution of the problem is played by the algorithm shown in Fig. 1.

### *B. The procedure*

Let the input measurement quantity be represented by $x_c$, as real, floating point, data. Then, using the algorithm shown in Fig. 1, $\varphi_d$ , with the specified $m$ and $n$=1 $x_c$, is reduced to $x_d$ , floating-point data with m-1 integer digits. The digit on the $2^{-1}$ position is used for rounding-off. Thus the algorithm $\varphi_d(x_c) = x_d$ serves as a simulating algorithm of an (m-1)-digit A/D converter. Also if it is necessary to output data as a physical quantity, from the system with the floating-point numeric processor, the algorithm $\varphi_d$ can serve as an (m-1)-digit D/A converter simulating algorithm.

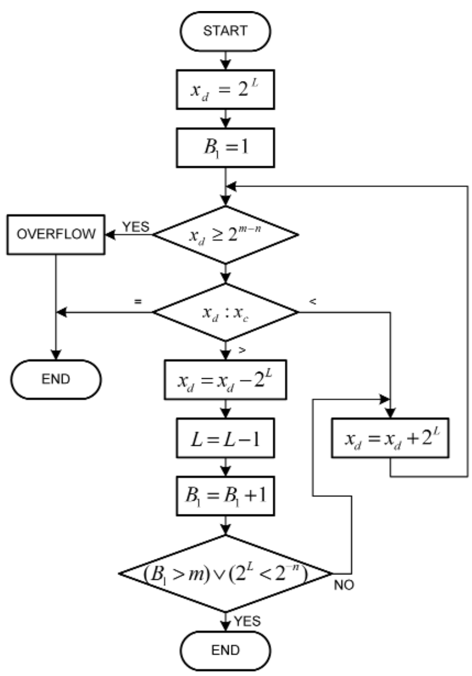


**Figure 1**. *Flow chart of the converting algorithm $\varphi_d(x_c)$.*

Let $x_1$ and $x_2$ be two real numbers with maximum $m$ digits of mantissa, in the range ($a^{m-n}$, $a^{-n}$). Let ⦸ be any of four fixed point arithmetic operators, and be any of the corresponding floating-point arithmetic operators. It is easy to see that (2) is valid:

$$x_1 \oslash x_2 = \varphi_d\,(x_1 \text{ e } x_2) \qquad (2)$$

The application of (2) is illustrated in the following example:

*Example*: If *a, b, c* and *d* are quantities like $x_1$ or $x_2$ and if $z$ is calculated using

$$z = ((a \text{x} b)/\text{c})\text{-}d \qquad (3)$$

Then $z_d$ is correspondingly given as:

$$z_d = \varphi_d\,(\varphi_d\,(\varphi_d(a\text{x}b)/c)\text{-}d) \qquad (4)$$

$z_d$ is of variable fixed-point precision with a total of $m$ digits, ($1 \le m \le t$), and with digits after the decimal point. This is the basis for development of the variable precision measurement data processing algorithms.

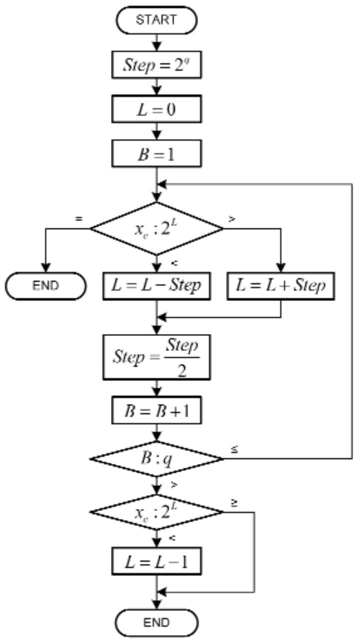


**Figure 2**. *Searching algorithm for initial value of $x_d$, i.e., $2^L$.*

Let us consider built-in functions, such as sin($x$), cos($x$) and others like them. In fact, the arguments of these functions can be simulated by arithmetic expressions such as (4), so $x$ has a fixed-point form, as mentioned above. Putting it simply

$$\sin_d(x) = \varphi_d\,(\sin(x)) \qquad (5)$$

and a simulated built-in function $\sin_d(x)$ is produced.

It is necessary to say a few words about the initial value of $x_d$ (that is $2^L$), in the $\varphi_d$ algorithm: every $x_c$ and corresponding $c_d$ belong to the same number interval $[2^L, 2^{L+1})$. It is possible to construct the efficient searching algorithm to find the number range lower limit (i.e. $2^L$) for every $x_c$ in that number range. This algorithm is shown in Fig. 2. Let the set of exponents, $\{-2^q, -2^q+1 \ldots, -1, 0, 1, \ldots 2^q-1\}$ be chosen. Then every exponent of particular number $x_c$, L, belongs to this set, i.e. $L \in \{-2^q, -2^q+1 \ldots, -1, 0, 1, \ldots 2^q-1\}$

$$\log_2 \;|xMaxExp| \ge q \qquad (6)$$

where $|xMaxExp|$ is the maximum absolute value of the exponent in the host computer.

The output value of the algorithm is $L$. It defines the lower limit of the number interval $[2^L, 2^{L+1})$ to which $x_c$ belongs:

$$x_c \in [2^L, 2^{L+1}) \qquad (7)$$

## *C. Sign Handling*

In the conversion algorithm $\varphi_d$, it is assumed that the number $x_c$ is positive. If $x_c$ is negative, then the algorithm shown in the Fig. 3 has to be applied. Consequently, $\varphi_d$ in (4) becomes (8). $\Psi_d$ always provides

$$zd = \Psi_d\,(\Psi_d\,(\Psi_d\,(a\text{x}b)/c)-d) \quad (8)$$

a nonnegative argument for $\varphi_d$.

## *IV. GENERALIZATION*

For a given computer and given high-level programming language, $t$ is known. Then, the general conversion algorithm $\Psi_d$ may be written in the high level programming language, as a subroutine. $m$ and $n$ are parameters of the subroutine. For the three different purposes listed above, it is possible to specify $\Psi_d$ subroutine with different parameters $m$ and $n$. The entire measurement system may then be described and simulated in the chosen high-level programming language, as a single program.

The optimum values of the different parameters regarding their effect can be found by running the program with various parameters $m$ and $n$. The quantization error is simply

$$\Gamma = \frac{z_d - z}{z} \qquad (9)$$

Where, in our example, *a, b, c* and *d* in *z*, have $t$ digits in the mantissa, $z$ is given by (3), while $z_d$ is described by (8).

This method requires a computer and a chosen high-level programming language environment. The improvements in the language, such as visualization, I/O functions, object forming and programming, etc., improve the method itself.

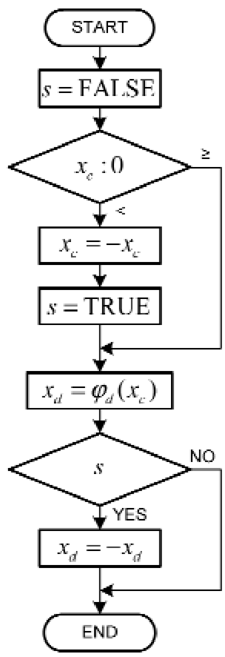


**Figure 3.** *General converting algorithm $\Psi_d(x_c)$.*

## V. CASE STUDY: MEASUREMENT OF 64 HARMONICS IN EEG SIGNAL MEASUREMENTS

In measurement and processing of biomedical signals, measurement of harmonics is very important especially in medical imaging. The papers [8, 9] explain how, by measuring 16 harmonics, is possible to suppress noise and accurately measure EEG signal. Our case study suggests how it is possible to solve even more complex problem - measuring 64 harmonics. The solution for this problem is given in the following steps.

1. step: The Radix 2 FFT algorithm, written in DELPHI, that the authors have for many years and that has been checked and tested and works reliably, is chosen.

2. step: By applying the *Solution proposal for optimal choice problem* algorithm is converted to a variable precision algorithm.

3. step: The cases of $\Psi_{AD} = \Psi_d(m,1)$ and $\Psi_{FP} = \Psi_d(m,m)$ are simulated. It is obvious that they represent special cases of operators shown in Figs 1, 2 and 3.

4. step: Then the input test trigonometric polynomial is chosen, in our case:

$$y(t) = 10 + \sum_{i=1}^{63} [(64-i)\cos i\omega t + (64-i)\sin i\omega t]$$

In Fig. 4 the graph representing the impact of the A/D converter resolution, $m_{AD} \in \{12,14,16\}$, and the FP processor resolution, $m_{FP} \in \{12,14,16,18,20,22,24,26,28\}$, on the root of the mean square absolute measurement error for single harmonic is shown.

Basically an inverse problem should be solved. We should examine whether one of the currently widely used hardware-software platform, the CELL microprocessor, meets the requirements by its optimal hardware performances.

5. step: Discussion.

If the optimal resolution of a CELL processor, $m_{FP}=24$, the Fig. 4 clearly shows that for the FFT in 256 points (2 times oversampling, 64 harmonics), the optimum resolution of a processor does not jeopardize the accuracy of measurements of any of the three simulated A/D converters. Moreover, it is shown that it is enough that the resolution of the applied FP processor be 4 bits higher than the resolution of an applied A/D converter in order not to jeopardize the accuracy of the measurement of harmonics. This is very important finding because the CELL processor operates in its optimum mode in terms of speed in that case [10].

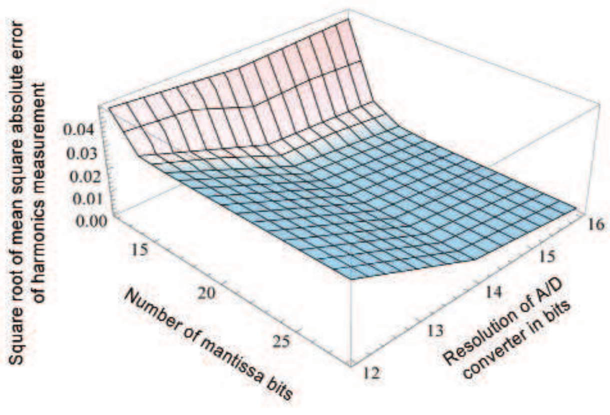


**Figure 4.** *Plot of simulated measuring process of 64 harmonics*

By simply counting arithmetic operations in FFT algorithm and the speed of measurement of harmonics we can determine that numerical processor usage for this is only 0.0015%. This proves that CELL processor can perform many other tasks simultaneously.

## *VI. CONCLUSION*

The procedure for transformation of real-time numerical algorithms into variable precision ones is presented in the paper. The procedure is the key tool for solving hardware-software platform optimal choice problem. Its application is demonstrated in the case study of design of contemporary instrument for measurement of harmonics of EEG signal. This case study is, in fact, inverse problem solution – the procedure confirmed that the CELL microprocessor platform can be used in instrument design in its optimal numeric processing regime.


## *Apstrakt*

Ovaj rad rešava problem numeričkog algoritma za transformaciju iz realnog vremena u preciznu varijablu. Za rešenje naučnih i tehničkih problema neophodno je napraviti izbor hardvera i softverske platforme. Analiza slučaja koji je obradjen daje optimalnu primenu CELL procesora i softverske platforme pri merenju EEG signala koji pripada klasi nestacionarnih signala.



## *REFERENCES*

[1] V. Vujicic and Z. Ranic, "A Simulation Method for Error Analysis in Real Time Numeric Algorithms," Numer. Appl. Math., vol. 1, pp. 657-660, 1989.

[2] V. Vujicic, I. Zupunski, S. Milovancev, "Predetermination of the Quantization Error in Digital Measurement Systems," IEEE Trans. Instr. Meas., vol. 46, No. 2, April 1997

[3] V. Vujicic, "The Choice of Optimal Arithmetics for Application in Measurement Systems (in Serbian)", Proceedings of the XXXIII annual meeting JUREMA, Plitvice - Zagreb, April 1988.

[4] IEEE Standard 754-2008, Standard for Floating Point Arithmetic

[5] L. R. Rabiner and B. Gold, "Theory and Application of Digital Signal Processing," Englewood Cliffs, NJ: Prentice-Hall, 1975.

[6] A. V. Oppenheim and R. W. Schafer, "Digital Signal Processing," Englewood Cliffs, NJ: Prentice-Hall, 1975.

[7] M. F. Wagdy and V. M. Ng, "Validity of Uniform Quantization Error Model for Sinusoidal Signals Without and With Dither," IEEE Trans. Instrum. Meas., vol. 38, No. 3, June 1989.

[8] B. Santrac, M. Sokola, Z. Mitrovic, I. Zupunski and V. Vujicic, "A Novel Method for Stochastic Measurement of Harmonics at Low Signal-to-Noise Ratio," IEEE Trans. Instr., Meas., Vol. 58, No. 10, pp. 3434-3441, October 2009.

[9] P. Sovilj, S. Milovančev, V. Vujičić, "Digital stohastic measurement of non-stationary signal with an example of EEG signal measurement", IEEE Trans. Instrum. Meas., accepted for printing, 2010.

[10] http://en.wikipedia.org/wiki/Cell_microprocessor


*The paper was accepted* 19.05.2010