Low Power Adaptive Filters Based on a Combination of Genetic Optimization and Residue Number System Coding

C. Radhakrishnan and W. K. Jenkins
Department of Electrical Engineering
The Pennsylvania State University
University Park, PA
jenkins@engr.psu.edu

D. J. Krusienski
Department of Electrical Engineering
University of North Florida
Jacksonville, FL
d.krusienski.59536@unf.edu

Abstract - This paper investigates design strategies for achieving reliable performance in low power VLSI adaptive filters that are prone to transient errors due to increasingly smaller feature dimensions and supply voltages of the CMOS circuits. First it is shown that a well known stochastic search algorithm, the Genetic Algorithm, has an inherent resistance to transient (soft) errors that may occur due to feature scaling. It is then shown how modular hardware can be designed with residue number system (RNS) coding to provide improved resistance to transient (soft) errors in low power realizations of adaptive filters that optimize the filter parameters via the Genetic Algorithm.

I. INRODUCTION AND BACKGROUND

Achieving improvements in terms of speed, power and integrating complex systems on a single chip have been the chief motivating factors behind the continuous CMOS technology scaling. In addition to providing better performance and means to integrate complex systems on a single chip, scaling has resulted in the onset of many problems that affect the reliability of such circuits. The reduced feature sizes, increased pipeline depths and lower noise margins result in increased occurrence of soft or transient errors in nano scale circuits. The impact of atmospheric radiation and electrical noise effects on scaled down devices has been a widely researched subject [1, 2]. Many solutions have been proposed to solve the transient error problem at the circuit [3] and algorithm levels [4].

As scaling continues in the deep sub micron feature sizes there has been an increase in the quiescent power consumption due to an increase in leakage power and the dynamic power per transistor not decreasing at the rate expected. This coupled with high transistor density results in an alarming increase in power density and, consequently, higher operating temperatures on the chip [5]. Increased operating temperature directly affects the long term reliability of the chip. The various ageing phenomenon exhibit an exponential dependence on operating temperatures [6]. In [6] the adverse impact of scaling on ageing is explored in detail. The higher impact of ageing in scaled down devices results in a greater chance of occurrence of a permanent fault in these devices.

When adaptive echo cancellers, channel equalizers, noise cancellers, and LPC data compressors are implemented in nano-scale VLSI circuits there is a concern about how such systems will perform in the presence of both transient and permanent errors. Previous work has shown that in many circumstances transient errors have the effect of resetting the system to arbitrary initial conditions. In this case the adaptive system will re-adjust to meet the minimum MSE criterion and thereby bring the system back to a converged condition. However, there are other circumstances when transient errors can drive the system into instability, resulting in permanent system failure.

Section II reviews the principles of the Genetic Algorithm and discusses how this type of stochastic search algorithm achieves inherent fault tolerant behavior. Section III introduces fault tolerance for soft errors through hardware and arithmetic modularity via residue number system (RNS) arithmetic. It is shown how the error magnification property of redundant RNS coding enhances the ability of the GA strategy to maintain fault tolerance. Finally Section IV presents some experimental examples that demonstrate the fault tolerant behavior of GA-based adaptive filters operating with RNS arithmetic.

II. ERROR RESISTANCE OF EVOLUTIONARY ALGORITHMS

Evolutionary algorithms (EA) begin with a random set of possible solutions (the unknown parameters to be optimized), termed the population [7]. Each possible solution in the population is termed an individual. Each individual's set of parameters is termed a chromosome or genome, and each parameter is termed a gene. Depending on the nature of the problem, the chromosomes may represent real numbers or can be encoded as binary strings.

At every generation, the fitness of each individual is evaluated by a predetermined fitness function that is assumed to have an extremum at the desired optimal solution. An individual with a fitness value closer to that of the optimal solution is considered better fit than an individual with a fitness value farther from that of the optimal solution. The population is then evolved based on a set of principles rooted in evolutionary theory such as natural selection, survival of the fittest, and mutation. Natural selection is the mating of the fittest individuals (*parents*) within the population to produce a new individual (*offspring*). This equates to randomly swapping corresponding parameters (*crossover*) between the parents to produce a new, potentially better fit individual. The new offspring then replace the least fit individuals in the population, which is the survival of the fittest facet of the evolution. A portion of the population is then randomly mutated in order to add new parameters to the search. The

978-1-4244-2110-7/08/$25.00 ©2007 IEEE

expectation is that only the offspring that inherit the best parameters from the parents will survive and the population will continually converge to the best possible fitness that represents the optimal or suitable solution. Several EA paradigms exist such as the genetic algorithm, evolutionary programming, and evolutionary strategies; each emphasizing only specific evolutionary constructs, encoding, and operators.

In the examples presented in Section IV a real-coded version of the GA is used based on that suggested in [8]. This particular variation of the algorithm was chosen due to its enhanced convergence speed. The scheme uses a ranked elitist strategy, where the *K* fittest members of the population of *P* individuals are used to generate offspring, which replace the remaining least fit members of the population. For each offspring, two of the *K* parents are selected randomly and the crossover is performed by a random weighted average of each parent's coefficients. A portion of the offspring is randomly mutated using a Gaussian distribution with a progressively decreasing variance to aid the convergence of the population.

This process is illustrated in equation (1) below:

$$\begin{aligned}\text{offspring}_k &= \text{diag}[s_1, s_2, \ldots, s_R] * \text{parent}_i \\ &+ (I - \text{diag}[s_1, s_2, \ldots, s_R])\text{parent}_j \quad \text{for} \quad k = P, \ldots, P-K \\ &+ \text{mutation}_k \end{aligned} \quad (1)$$

where *offspring* is the generated vector of new parameters, $parent_i$ and $parent_j$ are the vectors of parent parameters $(i,j) \in [1, K]$, $i \neq j$), *mutation* is a sparse vector of random mutation values, and the s_r is a vector of random numbers $\in (0,1)$. The optimum *K* reflects the convergence speed and is problem dependent.

The structure of the GA algorithm makes it inherently fault tolerant to transient errors in the updating of the parameters during the crossover and mutation operations, and also to transient errors that might occur in the filter calculations for each member of the population. Transient errors introduce effects that are similar to crossover, as well as to the mutation of selected members of the population. If a transient error does not significantly damage the fitness of a particular individual, that individual continues to influence the population as the stochastic search progresses. However, if a transient error produces a seriously unfit individual its influence is eliminated from the search process since the erroneous particle will not be selected in subsequent search updates.

III. FAULT TOLERANCE BASED ON RNS ARITHMETIC

Additional fault tolerance can be introduced into VLSI chip designs through the use of residue number system (RNS) coding. A general class of RNS arithmetic is constructed as a direct sum of simple modular structures (either fields or rings) that have moduli that are pairwise relatively prime integers (i.e. no two have a non-unity common factor). If R(M) is a modular ring that defines the computational range of a particular signal processing task, where $M = m_1 m_2 \ldots m_L$ and $\boldsymbol{M} = \{m_1, m_2, \ldots, m_L\}$ is the moduli set, then the arithmetic within the RNS is defined by:

$$(x_1, \ldots, x_L) * (y_1, \ldots, y_L) = (z_1, \ldots, z_L) \quad (2a)$$

$$\text{and} \quad z_i = (x_i * y_i) \bmod m_i \quad \text{for } i = 1, \ldots, L, \quad (2b)$$

where $*$ denotes addition, subtraction or multiplication. Since z_i is determined entirely from x_i and y_i, RNS arithmetic is carry-free in the sense that there is no propagation of information from the i^{th} digit to the j^{th} digit for $i \neq j$. The lack of carry propagation in residue arithmetic systems means that an error occurring in one digit cannot be propagated into other digit position during subsequent operations of addition, subtraction, or multiplication. Therefore the modularity of the arithmetic provides error isolation that limits the propagation of errors between the RNS modules [9].

In order to enable RNS error detection redundancy is provided by including extra moduli that provide dynamic range beyond what is needed for the actual computation. Suppose that one redundant modulus is appended to the original moduli set, creating a total of L+1 moduli. All of the L+1 moduli must be pairwise relatively prime to ensure a unique representation for each state in the RNS code. It is well established that the addition of one redundant modulus, such that $m_{L+1} > m_i$, $i = 1, \ldots L$, is necessary and sufficient to provide error detection capability for all single residue digit errors.

Error detection is typically implemented by converting an RNS number to an associated radix representation $a_L \ldots a_1$, where the a_i's are defined by:

$$x = a_L(m_{L-1} \cdots m_1) + \ldots + a_2 m_2 + a_2 \quad (3)$$

If no errors occur x will be constrained to the legitimate range [0, M-1]. However if any one of the RNS digits is corrupted by an error the result will be mapped into the illegitimate range $[M, M_R-1]$, where $M_R = m_{L+1}M$. Therefore single RNS digit error detection is achieved through single digit redundancy (SDR) and a simple check via mixed radix conversion [10].

Modularity of an RNS arithmetic code introduces three important properties that aid in managing the effects of transient errors:

i) *Hardware modularity* - RNS arithmetic is executed in short word length modules that tend to break long delay paths and result in fewer interconnects. Also, RNS designs have the capability to reduce power consumption in VLSI implementations [11].

ii) *Arithmetic modularity* – Modularity results in the lack of error propagation among modules and facilitates error detection strategies that are not so easily implemented in conventional long (2's-complement) word length processors.

iii) *Error magnification* – RNS error magnification occurs because all single digit errors map the erroneous result to large values. When used in the design of GA-based adaptive filters this property tends to make the erroneous result become an unfit member of the population, and hence it has little influence on the stochastic search algorithm.

The error magnification property is formally established by the following theorem:

Theorem: Let an RNS set of moduli $\boldsymbol{M} = \{m_1, m_2, \ldots, m_L\}$ be augmented with one redundant modulus m_{L+1} such that $m_i < m_{j+1}$ for $i = 1, \ldots, L$. Let $x = x_1 x_2 \ldots x_{L+1}$ be the RNS representation of a number $x \; \varepsilon \; [0, M\text{-}1]$. If a single digit error occurs in one of the residue digits x_i, $i = 1, \ldots, L+1$, RNS error magnification will cause the erroneous x_e to be mapped into the illegitimate range $[M, M_R\text{-}1]$.

Proof: The Chinese Remainder Theorem (CRT) defines how a residue number $x_1 x_2 \ldots x_{L+1}$ is mapped into the corresponding computational ring $R(M_R)$, as shown below:

$$\langle x \rangle \text{mod } M_R = \left\langle \sum_{i=1}^{3} \hat{m}_i \left\langle \hat{m}_i^{-1} x_i \right\rangle_{\text{mod } m_i} \right\rangle \text{mod } M_R \qquad (4)$$

where $M_R = \prod_{i=1}^{L+1} m_i$, $\hat{m}_i = \dfrac{M_{L+1}}{m_i}$, and $x \; \varepsilon \; [0, M_R - 1]$.

Note that the size ordering of the m_i's implies that:

$$\hat{m}_L < \ldots < \hat{m}_2 < \hat{m}_1$$

Let $\quad e_i' = \left\langle \hat{m}_i^{-1} e_i \right\rangle \text{mod } m_i, \quad \text{where } x_i^e = x_i + e_i.$

Then $\quad E_i = \hat{m}_i e_i', \quad \text{where } 0 – e_i' – m_i - 1 \qquad (5)$

is the error that occurs in x_e due to the single digit error e_i that occurred in x_i. From equation (5) the maximum and minimum errors in x_e can be determined:

(minimum error from the i^{th} digit):	$E_{i\,min} = \hat{m}_i$
(maximum error from the i^{th} digit):	$E_{i\,max} = M_R - \hat{m}_i$
(overall minimum single digit error):	$E_{min} = M$
(overall maximum single digit error):	$E_{max} = M_R - M$

Therefore $M \le e_i' \le M_R - \boldsymbol{M}$, which implies that x_e is mapped to the interval $[M, M_R\text{-}1]$. Q.E.D.

Note that the RNS error magnification property can be seen in equation (5) where the error e_i' is multiplied by $\hat{m}_i$, which is a relatively large factor for every value of i. The RNS signed computational range, the RNS legitimate and illegitimate ranges, and the single digit error ranges are illustrated in figure 2.

IV. EXPERIMENTAL EXAMPLES

To illustrate error magnification in a redundant RNS an example of a 233-tap FIR notch filter is presented in figure 3 and 4. This filter is used to eliminate noise from the frequency band 1.20 – 1.30 MHz. in a random noise radar system. The filter was implemented in a redundant RNS with moduli {71, 79, 83, 89, 97}, where 97 was defined to be the redundant modulus. In figure 3a the output of the RNS-SDR filter is shown operating on a noisy input signal without the occurrence of transient errors. In figure 3b the output of the filter is shown with RNS single digit transient errors occurring at iteration n = 1000 and held in the same error condition for 100 iterations. In figure 4 the simulation was repeated with randomly occurring single digit errors. In both cases during the occurrence of the errors the output of the filter is magnified (i.e. it is mapped into the illegitimate range), thereby revealing that a single digit error has occurred. It is also shown in figures 3c and 4b that during the occurrence of the errors the highest order mixed radix digit a_{L+1} becomes nonzero, thereby providing a direct error detection mechanism.

Experimental results showing the fault tolerant behavior of an adaptive filter using the GA adaptive algorithm implemented with redundant RNS arithmetic are presented in figures 5 – 8. In these experiments each of the individual filters was implemented in RNS arithmetic, although the updates were implemented in 2's-complement binary arithmetic. These experiments all used a population of 10 and a *K* factor of 3. All were implemented in the same RNS system used in the previous FIR filter example. Figure 5 shows the baseline (error free) learning curve for a length N = 4 FIR adaptive filter used in the system identification configuration to identify an FIR system with a unit pulse response of $\mathbf{h} = [2.0, -0.5, 0.5, 1.0]^T$.

In figure 6 the previous experiment was repeated but with a single RNS digit error introduced randomly during the calculations performed in 3-out-of-of-10 individual filters. Note that the adaptation rapidly recovers as the erroneous (unfit) individuals are removed from the population due to their poor fitness values. Figures 7 and 8 show extreme cases where transient errors were created at iteration 10 in 8-out-of-10 and 9-out-of-10 individual filters. In each case there are transient disturbances in the learning curve, after which the adaptation rapidly recovers.

Finally, figures 9-11 show repeats of the previous three experiments where 3-out-of-10, 8-out-of-10, and 9-out-of-10 of the individual filters were corrupted at iteration 10 by single transient errors. In these final three experiments RNS arithmetic was replaced by normal real arithmetic so the RNS error magnification effect was removed from the adaptive process. By pairwise comparing the results of figures 6 & 9, 7 & 10, and 8 & 11 it is seen that in each case the RNS arithmetic resulted in faster post-error convergence than the results obtained with real arithmetic. In all experiments that were conducted rapid error recovery was seen in the adaptation process and the convergence rate appeared to be enhanced by the error magnification property provided by RNS single digit redundancy.

REFERENCES

[1] Peter Hazucha and Christer Svensson, "Impact of CMOS technology scaling on the Atmoshperic Neutron Soft Error Rate.," IEEE Trans. On Nuclear Science, Vol. 47, No. 6, December 2000 .

[2] Premkishore ShivaKumar, Stephen W.Keckler, Doug Burger, Michael Kistler, Lorenzo Alvisi, "Modelling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic," Proceedings of the International Conference on Dependable Systems and Networks, pp. 389 – 398, June 2002.

[3] P. Landman and J. Rabaey, "Architectural power analysis: The dual bit type method," *IEEE Trans. VLSI Syst.*, pp. 173-187, June 1995.

[4] R. Hegde and N. R. Shanbhag, "Soft digital signal processing," IEEE Trans. On Very Large Scale Integration (VLSI) Systems, vol. 8, no. 4, pp. 813-823, Dec. 2001.

[5] M. Pedram and S. Nazarian, "Thermal modeling, analysis, and management in VLSI circuits: Principles and methods," *Proceedings of the IEEE*, Special Issue on On-chip Thermal Engineering, vol. 94, no. 8, pp 1487-1501, August 2006

[6] Jayanth Srinivasan, Sarita V. Adve, Pradip Bose, Jude A. Rivers, "The Impact of Technology Scaling on Lifetime Reliability," Proceedings of International Conference on Dependable Systems and Networks, pp 177, June2004.

[7] D. J. Krusienski, "Enhanced Structured Stochastic Global Optimization Algorithms for IIR and Nonlinear Adaptive Filtering", Ph.D. Thesis, The Pennsylvania State University, 2004.

[8] Mars, P., Chen, J.R., Nambiar, R., Learning Algorithms: Theory and Applications in Signal Processing, Control, and Communications, CRC Press, Inc., 1996.

[9] W. K. Jenkins, "Finite Arithmetic Concepts," Chapter 9 in *Handbook on Digital Signal Processing*, J. F. Kaiser and S. K. Mitra, eds., John Wiley and Sons, Inc., 1993.

[10] W. K. Jenkins, C. Radhakrishnan and S. Pal, "Fault tolerant signal processing for masking errors in VLSI signal processors," *Proceedings of the International Symposium on Circuits and Systems*, New Orleans, LA, pp 2570-2573, May 2007.

[11] G. L. Bernocchi, G. C. Cardarilli, A. Del Re, A. Nannarelli and M. RE, "Low power adaptive filter based on RNS components," *Proceedings of the International Symposium on Circuits and Systems*, New Orleans, LA, pp 3211-3214, May 2007.

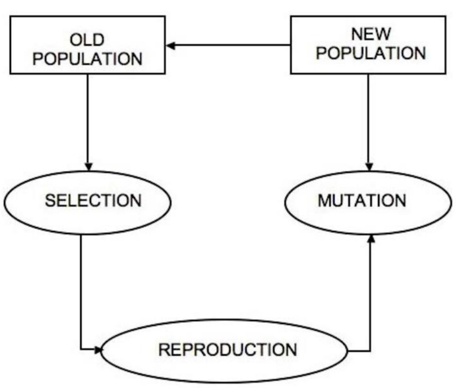

Figure 1. Genetic algorithm cycle.

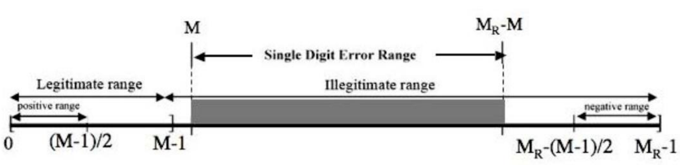

Figure 2. Distribution of single digit errors in a redundant RNS.

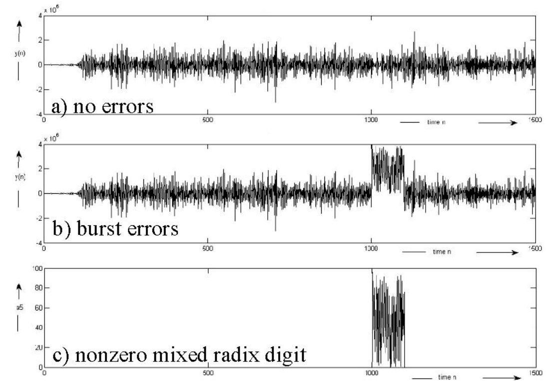

Figure 3. Error magnification of burst errors in a 233-tap FIR notch filter used in random noise radar (notch from 1.2-1.3 mHz).

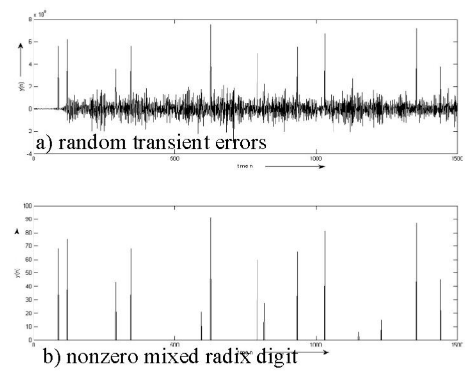

Figure 4. Error magnification of transient errors in a 233-tap FIR notch filter used in random noise radar (notch from 1.2-1.3 mHz). Note that $a_5 \neq 0$ on iterations when errors occur.

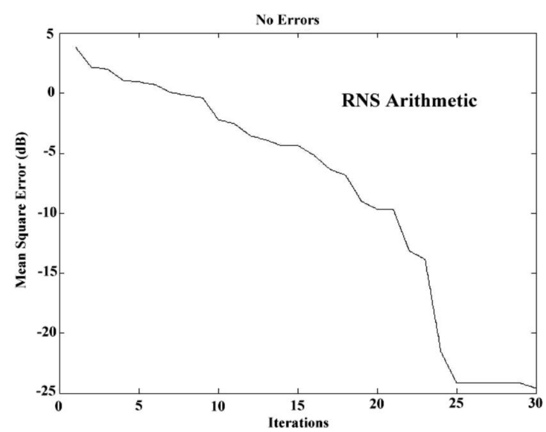

Figure 5. Baseline learning curve for the genetic algorithm.

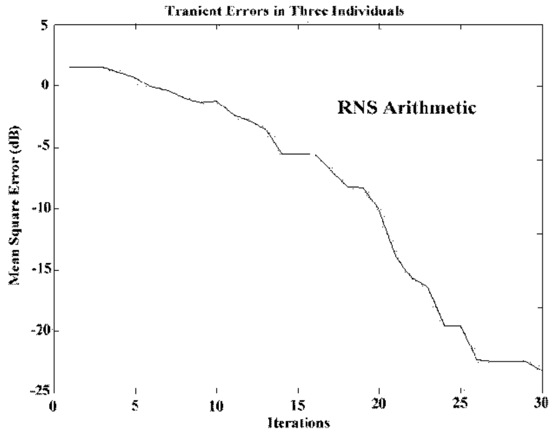

Figure 6. Learning curve for the GA with transient errors in three members of the population (RNS arithmetic).

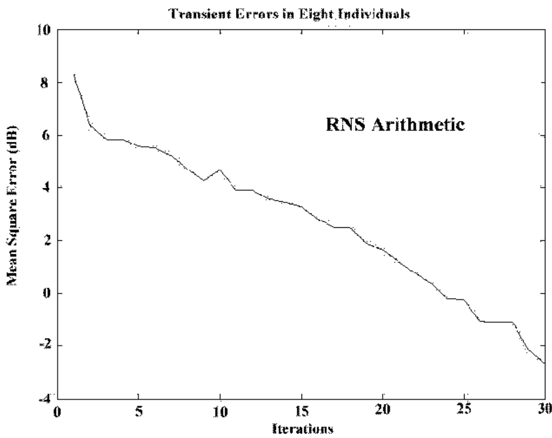

Figure 7. Learning curve for the GA with transient errors in eight members of the population (RNS arithmetic).

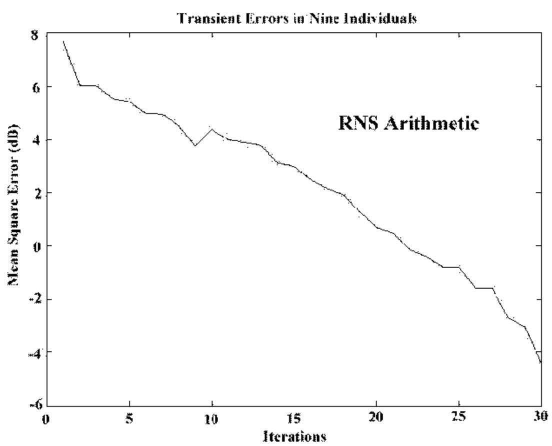

Figure 8. Learning curve for the GA with transient errors in nine members of the population (RNS arithmetic).

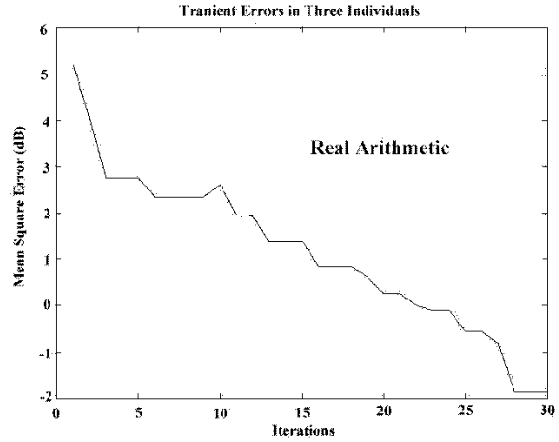

Figure 9. Learning curve for the GA with transient errors in three members of the population (real arithmetic)

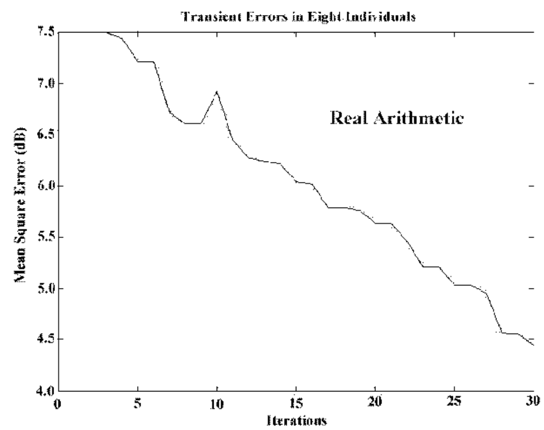

Figure 10. Learning curve for the GA with transient errors in eight members of the population (real arithmetic)

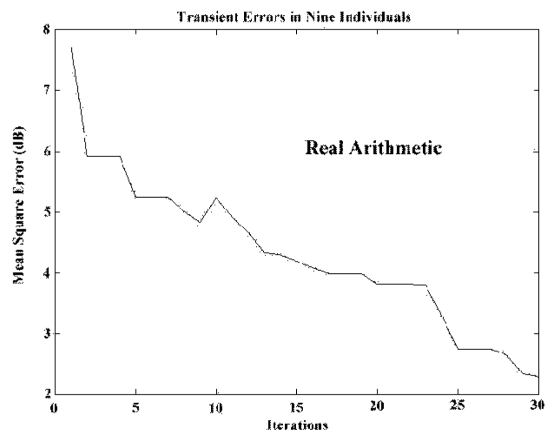

Figure 11. Learning curve for the GA with transient errors in nine members of the population (real arithmetic).