Synthesis of Saturation Arithmetic Architectures

G. A. CONSTANTINIDES, P. Y. K. CHEUNG, and W. LUK
Imperial College of Science, Technology and Medicine, U.K.

This paper describes a synthesis technique for automating the design of linear Digital Signal Processing (DSP) systems such as digital filters. The proposed methodology makes optimized use of saturation arithmetic to produce a small design implemented directly in hardware. An analytical technique is proposed to estimate the saturation error resulting from a particular implementation, and an optimization procedure is introduced to aim for the smallest implementation satisfying user-specified bounds on saturation and roundoff error. Results are presented illustrating significant speedup and area reduction compared with standard DSP design techniques: up to 22% improvement in area and 28% improvement in speed have been obtained on Field Programmable Gate Array (FPGA) implementations.

Categories and Subject Descriptors: B.5.2 [**Register-Transfer-Level Implementation**]: Design Aids—*automatic synthesis; optimization*; B.7.1 [**Integrated Circuits**]: Types and Design Styles—*algorithms implemented in hardware*

General Terms: Algorithms, Design, Performance

Additional Key Words and Phrases: Signal processing, saturation arithmetic, synthesis

1. INTRODUCTION

When adding two numbers using two's complement representation, overflow results in a 'wrap-around' phenomenon. The result can be a catastrophic loss in signal-to-noise ratio in a DSP system. Signals in DSP designs are therefore usually either scaled appropriately to avoid overflow for all but the most extreme input vectors, or produced using saturation arithmetic components. Saturation arithmetic introduces extra hardware to avoid the wrap-around, replacing it with saturation to either the largest positive number or the largest negative number representable.

Authors' addresses: G. A. Constantinides, P. Y. K. Cheung, Department of Electrical and Electronic Engineering, Imperial College of Science, Technology and Medicine, Exhibition Road, London SW7 2BT, United Kingdom; email: {g.constantinides,p.cheung}@ic.ac.uk; W. Luk, Department of Computing, Imperial College of Science, Technology and Medicine, 180 Queen's Gate, London SW7 2AZ, United Kingdom; email: wl@doc.ic.ac.uk.

If worst-case, or 'ℓ_1' scaling of internal variables (signals) is used [Constantinides et al. 2001], overflow is not an issue, therefore standard arithmetic is to be preferred. In some cases ℓ_1 scaling can be overly pessimistic, catering for situations that are extremely rarely encountered with practical input signals. Overly pessimistic signal scaling can lead to a number of superfluous most significant bits (MSBs) in a datapath. This is particularly true in the case of digital filters with long impulse responses such as many autoregressive structures [Mitra 1998]. Under these circumstances an alternative scaling scheme could be used to reduce the datapath width and therefore save implementation area, however overflow becomes a distinct possibility.

Saturation arithmetic provides advantages in terms of allowing controlled overflows that may not significantly affect the output signal-to-noise ratio (SNR). However these advantages are provided at the cost of a somewhat larger and slower circuit compared to an implementation adopting standard two's complement arithmetic for the same binary point locations and wordlengths. Care must be taken to select the appropriate places to saturate signals and the appropriate severity of saturation to apply at those points, a process automated by the proposed procedure.

A standard design approach, when creating a saturation arithmetic implementation of a DSP system, is to determine signal scaling through simulation. Input vectors are supplied to the system, and the peak value reached by each internal signal is recorded. Signals are then scaled to ensure that the full dynamic range afforded by the signal representation would be used under excitation with the given input vectors. In contrast, this paper presents an optimization technique based on analytic saturation noise models.

The main original contributions of this paper are:

—The introduction of the saturated Gaussian distribution and the associated modelling techniques as a method for estimating saturation noise in linear time invariant systems.
—The introduction of a bound on average-case saturation error, and techniques to arrange saturation nonlinearities so as to minimize the slackness associated with this bound.
—Formulation of the combined wordlength and scaling optimization problem, and development of a heuristic algorithm to solve this problem.
—An evaluation of the proposed approach, demonstrating significant area savings and speedups over current design approaches.

After introducing the relevant background work in Section 2, the paper begins by introducing some necessary definitions in Section 3, after which a technique for analytic estimation of saturation arithmetic noise is presented in Section 4. This noise estimation procedure forms the basis of an optimization heuristic, presented in Section 5, to jointly optimize signal wordlengths and scalings. The algorithm has been implemented as part of our synthesis system *Synoptix* [Constantinides et al. 2001], and the results obtained are presented and discussed in Section 6. The paper concludes in Section 7.

2. BACKGROUND

The effects of using finite register length in fixed-point systems have been studied for some time. Oppenheim and Weinstein [1972] and Liu [1971] provide standard models for quantization errors and error propagation through linear time-invariant (LTI) systems. This early work tended to assume a fixed-wordlength computational machine, which leads to the assumption of a single uniform signal width. Custom hardware provides the freedom to design with datapath wordlengths shaped to fit the application.

There has been significant recent research into multiple precision implementations. The work on signal scaling and wordlength optimization can be broadly classified into two categories: correctness preserving transformations [Benedetti and Perona 2000; Stephenson et al. 2000], and techniques for trading off truncation/roundoff error with area [Cmar et al. 1999; Constantinides et al. 2001; Kum and Sung 2001; Leong et al. 1999; Willems et al. 1997].

Correctness preserving transformations aim to reduce the wordlengths in computations to the minimum possible while preserving the behaviour specified in the original fixed-point algorithm description. Stephenson et al. [2000] have proposed a technique based on compiler-passes involving data flow analysis, where the precision information is carried by the data flow. They propose propagating known information on data ranges both forward and backward through a data flow graph. Benedetti and Perona [2000] propose a similar approach based on interval analysis. These transformation based approaches can easily be incorporated into a high-level synthesis [DeMicheli 1994] design flow.

A more general framework for solving the wordlength determination problem revolves around the idea we describe as *lossy synthesis* [Constantinides et al. 2001]. This is the approach that correctness need not be preserved in the strict sense, but the resulting behaviour must not differ from the original by more than a user-defined amount. Most of the work on lossy synthesis has considered the issue from a software profiling perspective [Kum and Sung 2001; Leong et al. 1999]. These approaches typically use operator overloading to maintain both an 'accurate' (usually double precision floating-point) and a fixed-point representation of each signal simultaneously. After simulation with a user-specified input data set, the range of each signal can be estimated so the scaling can be decided. By comparison between the accurate and fixed-point versions of the outputs, empirical signal distortion statistics can be calculated for a given set of signal parameters.

A somewhat different approach is taken by Cmar et al. [1999], Willems et al. [1997], where simulation results are used in combination with 'format propagation' and/or user interaction. These techniques use simple analytical methods, such as determining the maximum value of an addition from the independent maxima of its two inputs, in order to propagate simulation data through the algorithm specification without losing information. These systems require less simulation overhead than those in Kum and Sung [2001], Leong et al. [1999], as simulation is typically used only for a subset of signals in the specification.

While simulation-based approaches allow the use of nonlinear and time-varying components, the quality of the resulting SNR estimates are highly dependent on the input data sets used for simulation, or the user help given. In addition, long run-times are necessary for the simulations that form the basis of the optimization routines [Kum and Sung 2001]. In contrast, we restrict the systems of interest to linear time-invariant discrete time systems [Mitra 1998], which allow us to use an analytic framework incorporating both recursive structures (i.e. those with feedback) and non-recursive structures.

While all of these approaches are candidates for wordlength optimization, none of them, with the exception of Cmar et al. [1999], consider a framework for saturation arithmetic. Cmar's method is an ad hoc one, consisting of suggesting the use of saturating components for any signal where format propagation results in a much more conservative range estimation than simulation.

3. PRELIMINARIES

Definition 1. A *computation graph* $G(V, S)$ is the formal representation of an algorithm. V is a set of graph nodes, each representing an atomic computation (addition, constant coefficient multiplication, delay, branch, or input/output port), and $S \subset V \times V$ is a set of directed edges representing the data flow. An element of S is referred to as a *signal*.

The different types of node are: INPORT (primary input), OUTPORT (primary output), ADD (two-input adder), DELAY (unit-sample delay), GAIN (constant coefficient multiplier), and FORK (branching data-flow). These types are sufficient to allow any multiple-input multiple-output LTI system to be modelled.

Diagrammatically, ADD nodes are shown as a circle containing an addition symbol, GAIN nodes as a triangle, unit sample DELAYs as a box square containing z^{-1}, INPORT, OUTPORT and FORKs are implicitly represented.

For a computation to provide some useful work, its result must in some way be influenced by primary external inputs to the system. In addition, there is no reason to perform a computation whose result cannot influence external outputs. These observations lead to the definition of a well-connected computation graph.

Definition 2. A computation graph $G(V, S)$ is *well-connected* if and only if (a) there exists at least one directed path from at least one node of type INPORT to each node $v \in V$ and (b) there exists at least one directed path from each node in $v \in V$ to at least one node of type OUTPORT.

The technique described in this paper builds upon the multiple wordlength paradigm [Constantinides et al. 2001]. This approach can best be introduced by comparison to more traditional fixed-point and floating-point implementations.

Each two's complement signal $j \in S$ in a multiple wordlength implementation of computation graph $G(V, S)$, has two parameters n_j and p_j, as illustrated in Figure 1(a). The parameter n_j represents the number of bits in the representation of the signal (excluding the sign bit), and the parameter p_j represents the displacement of the binary point from the LSB side of the sign bit towards the LSB.

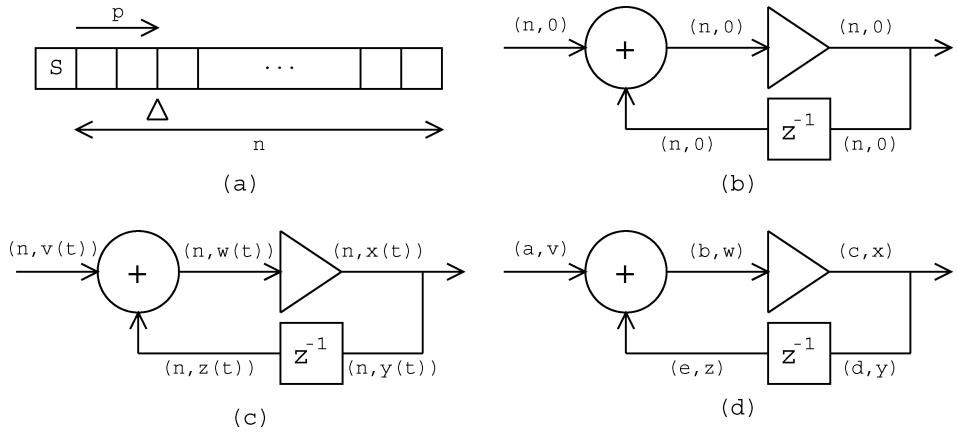


Fig. 1. The Multiple-Wordlength Paradigm: (a) signal parameters ('S' indicates sign bit), (b) fixed-point, (c) floating-point (t indicates time), (d) multiple wordlength.

A simple fixed-point implementation is illustrated in Figure 1(b). Each signal j in this block diagram representing a recursive DSP data-flow, is annotated with a tuple (n_j, p_j) showing the wordlength n_j and scaling p_j of the signal. In this implementation, all signals have the same wordlength and scaling, although shift operations are often incorporated in fixed-point designs, in order to provide an element of scaling control [Kim et al. 1998]. Figure 1(c) shows a standard floating-point implementation, where the scaling of each signal is a function of time.

A single uniform system wordlength is common to both the traditional implementation styles. This is a result of historical implementation on single, or multiple, pre-designed fixed-point arithmetic units. Custom parallel hardware implementations can allow this restriction to be overcome for two reasons, first, by allowing the parallelization of the algorithm so that different operations can be performed in physically distinct computational units, second, by allowing the customization of these computational units, shaping the precision of the datapath to the requirements of the algorithm. Together these freedoms point towards an alternative implementation style shown in Figure 1(d). This multiple wordlength implementation style inherits the speed, area, and power advantages of traditional fixed-point implementations, since the computation is fixed-point with respect to each individual computational unit. However, by potentially allowing each signal in the original specification to be encoded by binary words with different scaling and wordlength, the degrees of freedom in design are significantly increased.

Definition 3. An *annotated computation graph* $G'(V, S, A)$, is a formal representation of the multiple wordlength implementation of computation graph $G(V, S)$. A is a pair $(\mathbf{n}, \mathbf{p})$ of vectors $\mathbf{n} \in \mathbb{N}^{|S|}$, $\mathbf{p} \in \mathbb{Z}^{|S|}$, representing the wordlengths and scalings respectively, each with elements in one-to-one correspondence with the elements of S.

Definition 4. A *saturation computation graph* $G_S(V, S, C)$ is another annotated form of a computation graph $G(V, S)$. The set C takes the form

$C = \{(j_1, c_1), (j_2, c_2), \ldots, (j_p, c_p)\}$, where $j_i \in S$ and $c_i \in (0, \infty)$. C models the position j_i and cut-off c_i of each saturation nonlinearity $1 \leq i \leq p$ in the saturation system.

Definition 5. The ℓ_1-*norm* of a transfer function $H(z)$ is given by (1), where $\mathcal{Z}^{-1}\{\cdot\}$ denotes the inverse z-transform [Mitra 1998].

$$\ell_1\{H(z)\} = \sum_{t=0}^{\infty} |\mathcal{Z}^{-1}\{H(z)\}[t]| \tag{1}$$

4. NOISE MODEL

A fixed-point approximate realization of a saturation system is a nonlinear dynamical system containing two types of nonlinearities: saturations and truncations or roundoffs. Saturations are large-scale nonlinearities, whereas roundoffs are small-scale nonlinearities affecting a few of the least significant bits of a word. Thus it makes sense to consider the two effects separately, particularly since these errors are approximately uncorrelated since correlation between high-order and low-order bit patterns is unlikely [Johnson and Sandberg 1995; Constantinides 2001].

4.1 The Saturated Gaussian Distribution

In order to estimate the error incurred through the introduction of one or more saturation nonlinearities, a model is required for the Probability Density Function (pdf) of a signal undergoing saturation. A simplifying assumption is made that these pdfs may be approximated by a zero-mean Gaussian distribution. Gaussianity is a useful assumption from the modelling perspective, since the addition of two (arbitrarily correlated) zero-mean Gaussian variables forms another zero-mean Gaussian variable, and the scaling of a zero-mean Gaussian variable also forms another zero-mean Gaussian variable. It therefore follows that all internal signals in an LTI system driven by zero-mean Gaussian inputs will themselves be zero-mean Gaussian, since all signals can be expressed as a weighted sum of present and past inputs. The Gaussian assumption is also useful because the joint pdf $f_{XY}(x, y)$ of two zero-mean Gaussian variables X and Y is completely defined by their respective variances and correlation coefficient.

In reality, of course, inputs may follow a large variety of distributions that will cause the intermediate signals in the modelled system to deviate to some extent from their idealized Gaussian form. The assumption is that such a deviation will be small enough for practical cases and for the purposes to which this model will be put. Often the largest deviation from the Gaussian model is likely to be at the primary inputs to the system, rather than at internal nodes, since the internal nodes are formed by a weighted sum of present and past input values. In the most extreme example, where the LTI system under investigation approaches a normalized integrator (transfer function $H(z) = \lim_{n\to\infty} n^{-1}(1-z^{-n})/(1-z^{-1})$) and the input is made up of a stream of independent identically distributed (iid) random variables, the Gaussian approximation will clearly hold no matter what the input distribution, by the central limit theorem. In more general cases

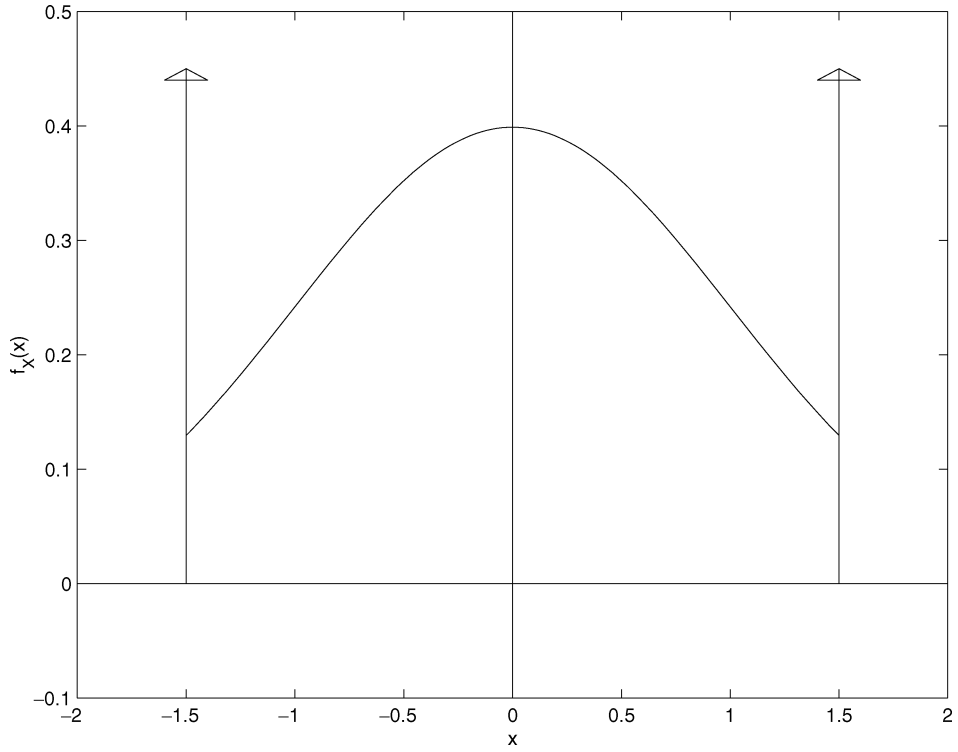


Fig. 2. The saturated Gaussian distribution with parameters ($\sigma = 1$, $c = 1.5$).

there are an abundance of extensions to the central limit theorem for specific relaxations of the constraints on independence and identical distribution [Chung 1974]. While there is no general theoretical result for all cases, it is reasonable to assume that bell-shaped distributions are common in practice. Evidence to support that assumption is available in Constantinides [2001], where modelling results are compared to simulations of 'real-world' speech input data.

By extension of the Gaussian distribution, we propose the *saturated Gaussian* distribution, as defined below and illustrated in Figure 2, to model the pdf at each internal signal within a saturation system.

Definition 6. A random variable X follows a *saturated Gaussian* distribution with parameters (σ, c) where $\sigma \geq 0$ and $c \geq 0$ if and only if its probability density function $f_X(x)$ has the form given in (2). The Gaussian distribution with mean 0 and standard deviation σ is referred to as the *underlying distribution*.

$$f_X(x) = \begin{cases} Q(c/\sigma)\left(\delta(x-c)+\delta(x+c)\right) + (1/(\sigma\sqrt{2\pi}))\exp(-x^2/(2\sigma^2)), & \text{if } |x| \leq c \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Here $\delta(\cdot)$ is the Dirac delta function and $Q(\cdot)$ represents the 'upper tail' function of the standard Gaussian distribution.

Since the pdf is an even function, all odd moments of the probability distribution vanish to zero. Expressions for the second and fourth moment of the saturated Gaussian, which will be used for modelling purposes, can be easily derived as (3) and (4) respectively.

$$\mu_2 = \sigma^2 - 2\left\{Q(c/\sigma)(\sigma^2 - c^2) + \frac{c\sigma}{\sqrt{2\pi}}\exp\left(-\frac{c^2}{2\sigma^2}\right)\right\} \tag{3}$$

$$\mu_4 = 3\sigma^4 - 2\left\{Q(c/\sigma)(3\sigma^4 - c^4) + \frac{1}{\sqrt{2\pi}}c\sigma(c^2 + 3\sigma^2)\exp\left(-\frac{c^2}{2\sigma^2}\right)\right\} \tag{4}$$

Multiplication of a saturated Gaussian random variable of parameters (σ, c) by a constant factor k results in a random variable with saturated Gaussian distribution of parameters $(k\sigma, kc)$. Clearly the multiple outputs of a branching node whose input has parameters (σ, c) will all have the same parameters (σ, c), and the output of a delay node will behave in a similar manner.

For addition the situation is more complex, since the addition of two saturated Gaussian variables does not result in a saturated Gaussian sum. Our proposed method is to model the sum, after saturation, by a saturated Gaussian distribution. This is performed by matching the first four statistical moments of the two distributions.

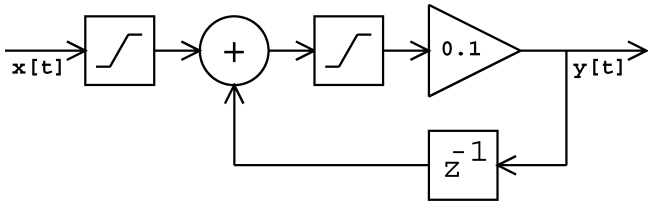
4.2 Error Propagation

Using the techniques of the previous section, it is possible to estimate the statistics of the saturation error caused by each saturation nonlinearity. In order to compare this with the user-specified bound on computation error at system outputs, it is necessary to propagate these errors to the system outputs.

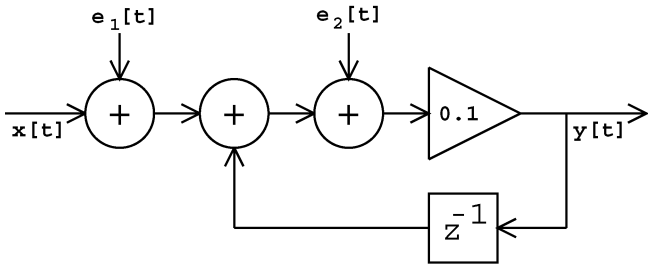
To perform the propagation, the saturation nonlinearities are linearized as shown for an example in Figure 3. This allows the use of linear system theory to predict the effect of this saturation error on the output signal-to-noise ratio. Unlike roundoff noise [Oppenheim and Weinstein 1972], it cannot be assumed that the saturation errors injected at various points within the structure are uncorrelated. In addition, a white spectral assumption on each individual error input [Oppenheim and Weinstein 1972] is not valid for saturation errors, since the spectrum of the error sequence will clearly depend heavily on the (possibly coloured) spectrum of the input sequence. These two dimensions of dependence, between pairs of error inputs and over time, require a more sophisticated error estimation model than that used for roundoff error estimation.

Estimating the cross-correlation function between saturation inputs is possible but computationally intensive. Instead, we propose a computationally efficient upper bound.

CLAIM 1. *In a saturation system let there be a total of k saturation nonlinearities, with corresponding linearized error inputs* $e_1, \ldots, e_k$ *of standard deviations* $\sigma_{e_1}, \ldots, \sigma_{e_k}$. *Let us concentrate on a single primary output with error* y, *and let* $H_i(z)$ *denote the transfer function [Mitra 1998] from error input* e_i *to this output.*



(a) A saturation system.



(b) The corresponding linearization.

Fig. 3. Linearization of the saturation nonlinearities.

Then (5) holds, where $E\{\cdot\}$ denotes statistical expectation.

$$E\{y^2[t]\} \leq \left(\sum_{i=1}^{k} \sigma_{e_i} \ell_1\{H_i(z)\} \right)^2 \tag{5}$$

PROOF SKETCH. The value of the error y at time index t (at any specific output) is given by the well known convolution sum [Mitra 1998]. Application of the Cauchy-Schwartz inequality results in (5) for zero-mean statistically stationary input signals. □

4.3 Reducing Bound Slackness

There are certain transformations that may be performed on the graph representation of a saturation system without affecting the global system behaviour. A saturation nonlinearity may be moved through a constant coefficient multiplication, a move accompanied by a corresponding scaling in the saturation cut-off parameter (Figure 4(a)), or through a unit-sample delay (Figure 4(b)). In addition, two consecutive nonlinearities can be merged (Figure 4(c)), and multiple nonlinearities following a branching node can be reconfigured (Figure 4(d)). Although these transformations do not result in different system behaviour from an external perspective, the estimated saturation error resulting from the procedure described above can differ, due to different slackness in the upper bound (5). It is useful to minimize this slackness.

Example 1. Applying these transformations to the computation graph shown in Figure 5(a) (a second order section), results in the improved saturation system in Figure 5(b); *W1* and *W2* are wiring constructs where the

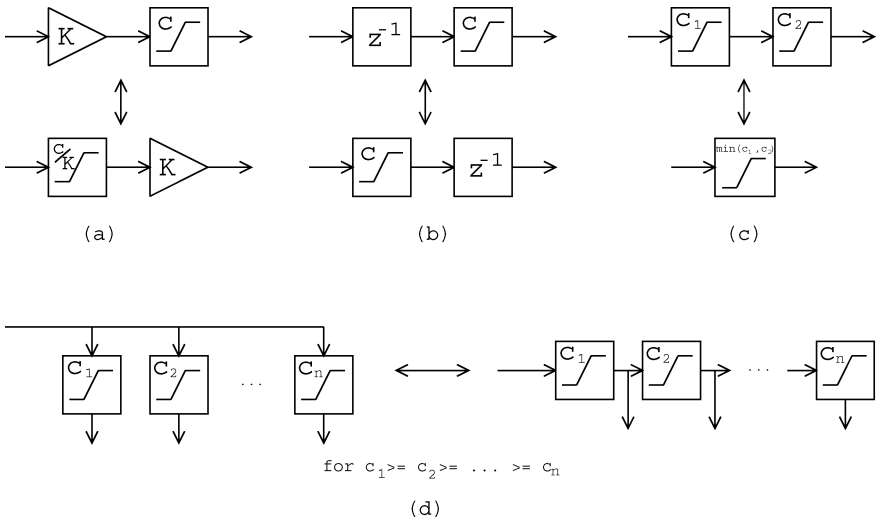


Fig. 4. Useful saturation transformations.

outputs are a permuted version of the inputs, and *S1* and *S2* are saturation constructs with possible forms illustrated in Figures 5(c) and (d) respectively. Applying the saturation noise model to this construction can result in a tighter Cauchy-Schwartz bound.

The transformations are combined in Algorithm 1 for a general saturation computation graph. Saturation nonlinearities are propagated through the computation graphs, in the reverse direction to the data flow. Back propagation allows nonlinearities to come together from different branches of a FORK, and perhaps propagate back through the FORK reducing the Cauchy-Schwartz bound. In contrast forward propagation would not allow merging of nonlinearities, as nonlinearities do not cross adder nodes. In Algorithm 1 the details of FORK nodes are omitted for brevity. A simple heuristic is used: the nonlinearities are sorted in order of cut-off and the transformation illustrated in Figure 4(d) is applied. Here COEF(v) denotes the value of the coefficient of gain node v.

Algorithm 1 (SlackReduce).
Input: A saturation computation graph $G_S(V, S, C)$
Output: An equivalent saturation computation graph
begin
do
foreach $v \in V : \exists((v, v'), c) \in C$ **do**
switch TYPE(v)
case GAIN:
$C \leftarrow C \cup \{(\text{inedge}(v), c/\text{COEF}(v))\} - \{((v, v'), c)\}$
(see Figure 4(a) for an illustration)
case DELAY:
$C \leftarrow C \cup \{(\text{inedge}(v), c)\} - \{((v, v'), c)\}$
(see Figure 4(b) for an illustration)
case FORK:
Apply transformation shown in Figure 4(d),
modify V, S and C accordingly

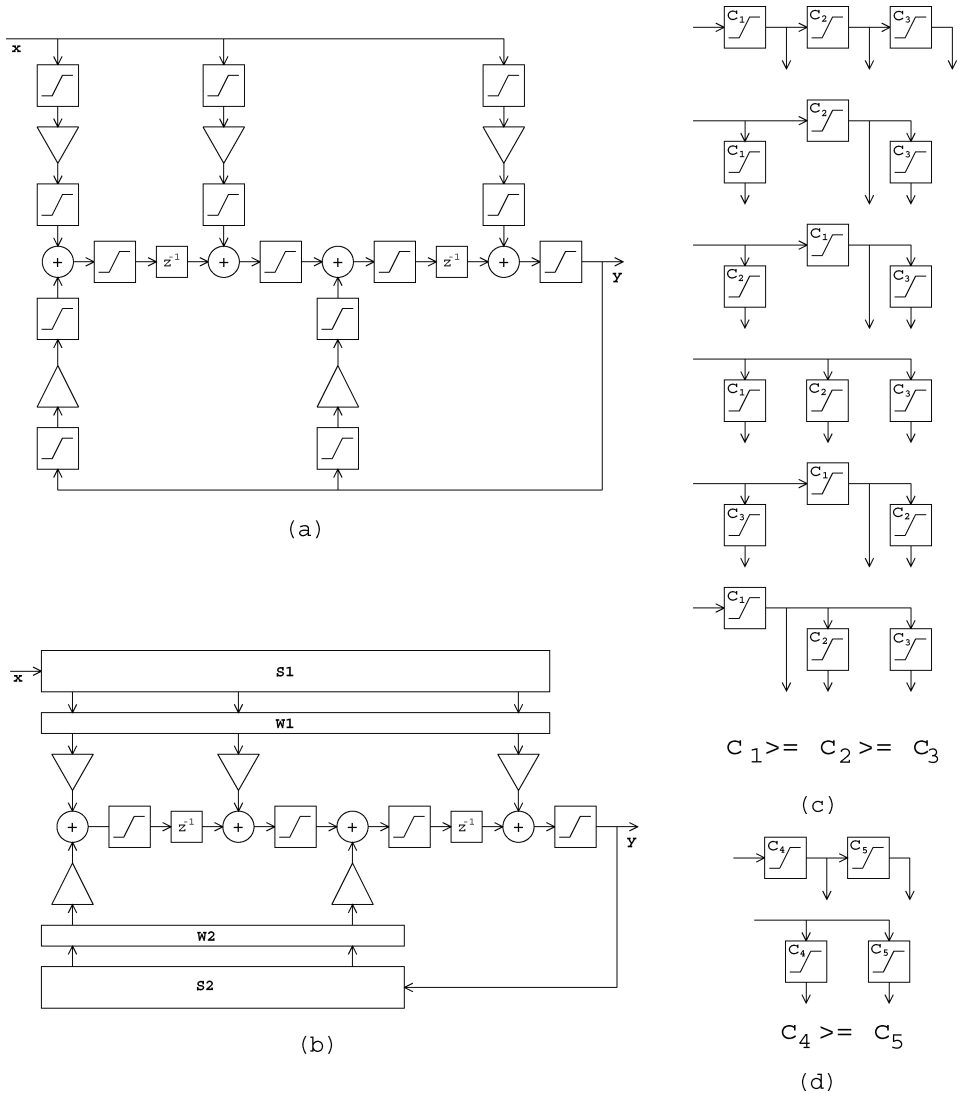


Fig. 5. Reducing estimate slackness in a second order IIR filter through saturation transformations: (a) original model (b) transformed model (c) possible forms of *S1* (d) possible forms of *S2*.

end switch
end foreach
$C \leftarrow C - \{(j, c) \in C : \exists (j, c') \in C, c' < c\}$
(see Figure 4(c) for an illustration)
while C has changed during current iteration
end

CLAIM 2. *When executed on a well-connected saturation computation graph, Algorithm 1 terminates in a finite number of steps.*

PROOF SKETCH. The proof of this claim relies on the argument that any well-connected [Constantinides 2001] computation graph must have at least one

adder in the body of each loop. Since saturation nonlinearities cannot be migrated across adders by Algorithm 1, termination is guaranteed. □

5. COMBINED OPTIMIZATION

The method presented in Section 4 can be used to form an estimate $\mathbf{E}_G([\mathbf{n}\ \mathbf{p}], \mathbf{R})$ of the error variances incurred through the implementation of computation graph G with wordlengths $\mathbf{n}$, binary point locations $\mathbf{p}$, and input correlation matrix $\mathbf{R}[\tau] = E\{\mathbf{x}[t]\mathbf{x}[t-\tau]^T\}$, where $\mathbf{x}$ is the vector of system primary inputs. This estimate may then be compared to the user-specified bounds $\mathcal{E}$ on error variance at each output.

Since both the scaling and the wordlength of each signal can have an impact on system error and area, we treat the problem of finding a suitable annotation for the computation graph as a combined optimization (Problem 1), given a suitable area metric $A_G([\mathbf{n}\ \mathbf{p}])$.

Problem 1. *Wordlength and Scaling Optimization*. Given a computation graph $G(V, S)$ and correlation matrix $\mathbf{R}$, select $(\mathbf{n}, \mathbf{p})$ such that $A_G([\mathbf{n}\ \mathbf{p}])$ is minimized subject to (6).

$$\mathbf{n} \in \mathbb{N}^{|S|} \text{ and } \mathbf{p} \in \mathbb{Z}^{|S|} \text{ and } \mathbf{E}_G([\mathbf{n}\ \mathbf{p}], \mathbf{R}) \leq \mathcal{E} \tag{6}$$

The proposed heuristic, based on the wordlength-only optimization discussed in Constantinides et al. [2001], is shown below as Algorithm 2, which uses Algorithm 3 as an auxiliary.

After performing an ℓ_1 scaling, the algorithm determines the minimum uniform wordlength satisfying all error constraints. The design at this stage corresponds to a standard uniform wordlength design with implicit power-of-two scaling, such as may be used for an optimized uniprocessor implementation. Each wordlength is then scaled up by a factor $k > 1$, which represents a bound on the largest value that any wordlength in the final design may reach. In our implementation, $k = 2$ has been used.

This structure forms a starting point from which one wordlength or scaling value is reduced by one bit on each iteration. The signal wordlength or scaling to reduce is decided in each iteration by reducing each variable in turn until it violates an output noise constraint or reaches the defined lower-bound (Algorithm 3). At this point there is likely to have been some pay-off in reduced area, and the variable whose reduction provided the largest pay-off is chosen. Each variable's range is explored using a binary search.

Although Algorithm 2 is a greedy algorithm, both the constraints and the objective function play a role in determining the direction of movement towards the solution. As a result, this algorithm is less dependent on local information than a pure steepest-descent search.

In these algorithms, B_j denotes a lower bound on the binary point location of signal j. Typically B_j is set to be a fixed, but reasonably large, number of bits beneath the binary point location implied by ℓ_1 scaling. Reaching $p_j = B_j$ is considered equivalent to $p_j \sim -\infty$. The interpretation of this value is that it is unnecessary to calculate signal j in order to satisfy the error constraints, and so the entire cone of logic creating signal j may be optimized away.

Algorithm 2 (CombinedOptHeur).
Input: A Computation Graph $G(V, S)$
input correlation matrix $\mathbf{R}$
Output: An Optimized Annotated Computation Graph $G'(V, S, A)$, $A = (\mathbf{n}, \mathbf{p})$
begin
Calculate the variance of each signal and the correlation coefficient between inputs to adders, to be used in all calls to the error estimation subroutine
Set $\mathbf{p} \leftarrow \ell_1$ scaling vector
Determine u, the minimum uniform wordlength satisfying $\mathbf{E}_G([u \cdot \mathbf{1}\ \mathbf{p}], \mathbf{R}) \leq \mathcal{E}$
Set $\mathbf{v} \leftarrow [ku \cdot \mathbf{1}\ \mathbf{p}]$
do
Set currentcost $\leftarrow A_G(\mathbf{v})$
foreach $j \in \{1, \ldots, |S|\}$ **do**
Set bestmin $\leftarrow$ currentcost
minvaln $\leftarrow$ ExploreOptVar($\mathbf{v}$, j, 1, n_j)
minvalp $\leftarrow$ ExploreOptVar($\mathbf{v}$, $|S| + j$, B_j, p_j)
if minvaln $<$ bestmin
Set bestsig $\leftarrow j$, bestmin $\leftarrow$ minvaln
if minvalp $<$ bestmin
Set bestsig $\leftarrow |S| + j$, bestmin $\leftarrow$ minvalp
end foreach
if bestmin $<$ currentcost
$v_{\text{bestsig}} \leftarrow v_{\text{bestsig}} - 1$
while bestmin $<$ currentcost
$[\mathbf{n}\ \mathbf{p}] \leftarrow \mathbf{v}$
end

Algorithm 3 (ExploreOptVar).
Input: An optimization vector $\mathbf{v}$, variable to explore i, lower bound ℓ, and upper-bound u.
Output: A cost value minval reached by exploring possible settings for variable i.
begin
Determine $w \in \{\ell, \ldots, u\}$ such that
$\mathbf{E}_G([v_1 \ldots v_{i-1}\ w\ v_{j+1} \ldots v_{2|S|}], \mathbf{R}) \leq \mathcal{E}$ and
$\mathbf{E}_G([v_1 \ldots v_{j-1}\ (w-1)\ v_{j+1} \ldots v_{2|S|}], \mathbf{R}) \not\leq \mathcal{E}$
If such a w exists, set
minval $\leftarrow A_G([v_1 \ldots v_{j-1}\ w\ v_{j+1} \ldots v_{2|S|}])$
If no such w exists, set
minval $\leftarrow A_G([v_1 \ldots v_{j-1}\ \ell\ v_{j+1} \ldots v_{2|S|}])$
end

The error estimation subroutine used in Algorithm 2 contains some computationally expensive calculations, namely the post-adder saturation error and pdf estimation, which involves numerical integration and series approximations. However the calls to adder saturation error estimation routines in the above algorithm exhibit a high degree of temporal locality. It is highly probable that a given p_j will not change from one iteration to the next, therefore the same error estimations are often required. Rather than recalculate these each time, new error estimates for a given adder are only calculated when the c parameter

of either input saturated Gaussian distribution is changed, or the post-adder saturation nonlinearity cut-off is changed.

6. RESULTS AND DISCUSSION

To illustrate the applicability of saturation arithmetic optimization to different types of design, two fourth order IIR filers have been generated. One is a narrow bandpass elliptic filter, and one is a lowpass elliptic filter. Both filters are to be driven with a speech input [FREETEL 1993]. Clearly the narrow bandpass filter will have very high theoretical peak values at internal signals, as would be determined by ℓ_1 scaling. However an input signal that would cause internal signals to reach these peaks is unlikely to be present in a speech signal, which contains a wide range of frequency components. Thus the ratio between ℓ_1 peak value at any signal and the peak value reached during a simulation run is likely to be relatively large. In contrast the equivalent ratios in the lowpass filter are likely to be much more modest.

6.1 Area Results

Using simulation to determine signal scaling avoids overflow for the specific input sequences provided although, importantly, not necessarily for all input sequences. The scaling of each signal, hence the area of a simulation-scaled system, depends on the length of input sequence used for simulation. The longer the input sequence, the more likely it is to encounter the 'tails' of the pdf of an internal signal. Simulating the system on a short input sequence may result in a smaller area at the cost of a failure to meet the error constraints on unencountered input sequences, when compared to lengthy simulation runs.

The proposed approach tolerates overflow errors if these errors help to achieve a small implementation cost, and is able to estimate the severity of such errors for the average case rather than for a specific input vector. The difference between the circuits resulting from the two design methodologies is that the proposed approach provides a guarantee of average-case error performance, whereas the error performance for simulation is only guaranteed for a particular input sequence. The shorter the input sequence, the more likely the system is to violate the error constraints under realistic circumstances; an analysis that should be borne in mind when considering the presented results.

6.1.1 *Bandpass Filter.* Figure 6 shows a comparison of different design approaches for area-error tradeoffs in the bandpass filter example. Simulation-based results are illustrated as regions (a) to (c): the upper curve in each region corresponds to simulation with a relatively long input sequence (10^5 samples at 8 kHz, a spoken announcement [FREETEL 1993]), whereas the lower curves correspond to a short input sequence (10^3 samples at 8 kHz, a spoken word). Region (a) illustrates a system that has been simulation-scaled, and implemented using the optimum uniform wordlength. Region (c) illustrates a system that has been simulation-scaled and then wordlength optimized using the technique presented in [Constantinides et al. 2001]. Plot (d), shown in bold, corresponds to the optimization procedure proposed in this paper. Plot (e) shows an ℓ_1-scaled

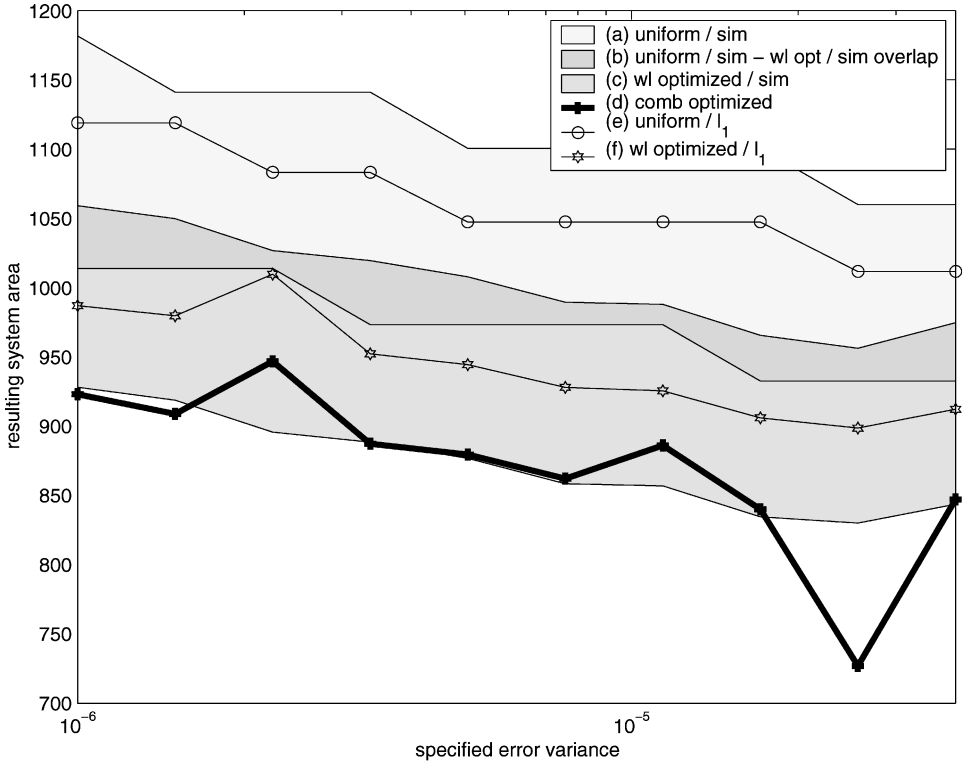


Fig. 6. Comparison of different design approaches for trading-off system area and error, for a fourth order narrow bandpass elliptic IIR filter.

Table I. Average Percentage Improvement of Algorithm 2 over Alternative Approaches for a Fourth Order Narrow Bandpass Elliptic IIR Filter

	Simulation Scaling		
	Short Input Sequence	Long Input Sequence	ℓ_1 Scaling
uniform wordlength	10.6%	21.8%	18.0%
multiple wordlength	0.3%	13.3%	7.9%

system with optimum uniform wordlength, and finally plot (f) illustrates an ℓ_1-scaled system with optimized wordlengths.

The regions (a) to (c) illustrate that a range of areas are achievable using simulation scaling, depending on the length of the input vector presented. This range encompasses areas below and above the ℓ_1-scaled case; above because of saturation arithmetic overheads, and below because of overly pessimistic ℓ_1 scaling. Plot (d) illustrates that superior area results can be achieved through the technique described, approximately matching the lower limit of region (c). However unlike plot (d), the simulation-scaled designs plotted in region (c), are not likely to meet the error specification for alternative input sequences. The results on this benchmark are summarized in Table I, which illustrates the improvement due to our technique, compared with six known approaches.

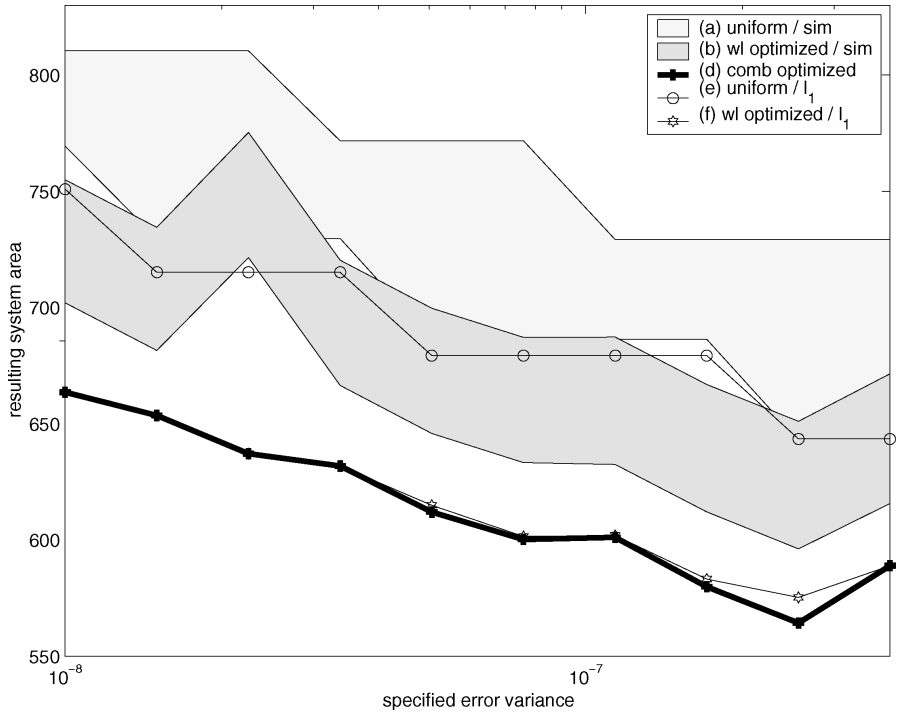


Fig. 7. Comparison of different design approaches for trading-off system area and error, for a 4th order lowpass elliptic IIR filter.

To summarize, area improvements over ℓ_1-scaling approaches of 8% to 18% have been achieved by using the optimization approaches described in this paper.

6.1.2 *Lowpass Filter.* Figure 7 shows the equivalent comparison for the lowpass filter example. In this case plot (e) lies consistently beneath region (a) (as with plot (f) and region (b)). This is due to the saturation arithmetic overhead being very significant in this example. Indeed, high quality solutions may be achieved through the use of wordlength optimization alone [Constantinides et al. 2001] without the need for saturation arithmetic. Plot (d), representing the technique proposed in this paper performs consistently well, matching [Constantinides et al. 2001] in most cases and improving upon it in others. Thus by *judicious* placement of saturators, it is even possible to improve the area consumption of this example. The results on this benchmark are summarized in Table II.

In summary, area improvements over ℓ_1-scaling of 0.3% to 11% have been achieved using the optimization approaches described in this paper.

6.1.3 *General Autoregressive Filters.* Area results have thus far been illustrated for fixed system function, while varying the specification on maximum error variance. Figure 8 illustrates the variation of area with system function, for constant error specification and again implemented on an Altera

Table II. Average Percentage Improvement of Algorithm 2 over Alternative Approaches for a 4th Order Lowpass Elliptic IIR Filter

	Simulation Scaling		
	Short Input Sequence	Long Input Sequence	ℓ_1 Scaling
uniform wordlength	12.2%	20.0%	11.0%
multiple wordlength	5.7%	13.0%	0.3%

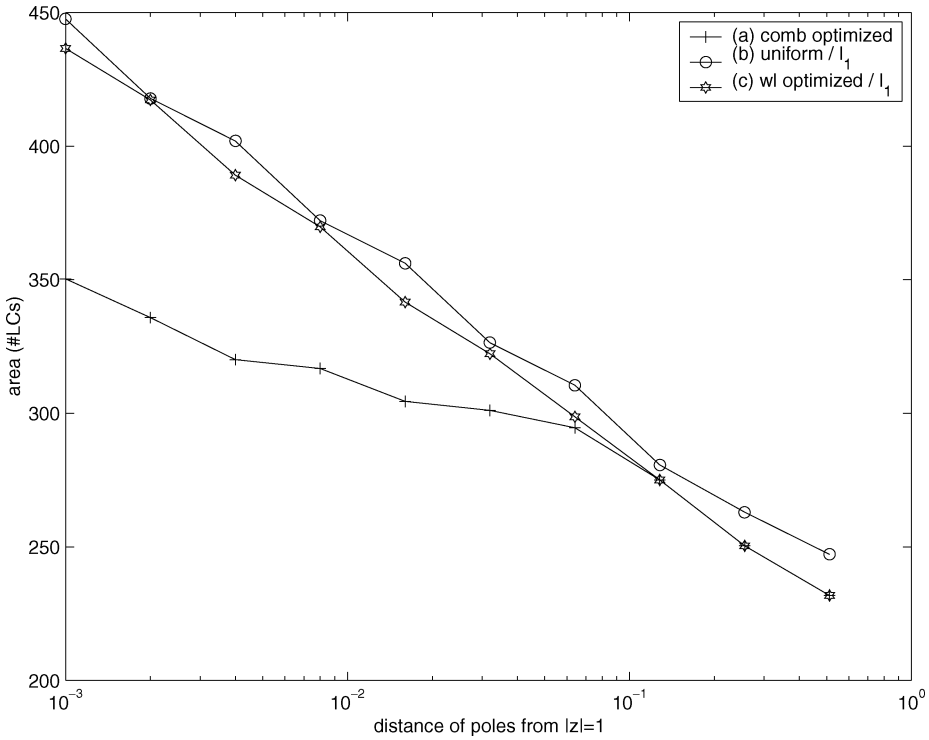


Fig. 8. Variation of system area with pole location.

Flex10k device [Altera Corporation 1998]. The area of ten second order autoregressive filters is plotted against their complex conjugate pole location on the z-plane [Mitra 1998]. Compared to worst-case ℓ_1 scaling, area savings of up to 20% have been achieved for all systems having poles of magnitude greater than approximately 0.9. This threshold value is dependent on the overhead associated with saturating components and will therefore be dependent on implementation technology.

6.2 Clock Frequency Results

There are significant timing overheads associated with the use of saturation arithmetic. While Algorithm 2 does not explicitly consider circuit speed, it is instructive to place the points on Figure 6 on a speed/area design-space diagram. This is shown in Figure 9, where the short simulation run results are

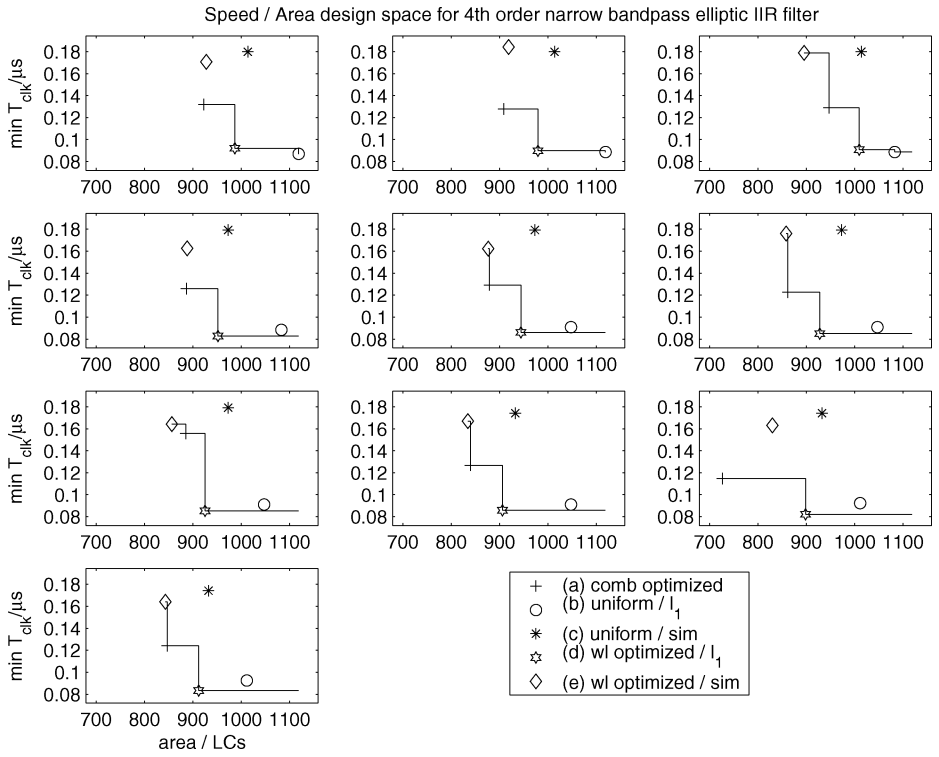


Fig. 9. Alternative design approaches, and their speed/area design-space locations.

used for representation of simulation-scaled systems thus providing a worst-case scenario for comparison with the proposed approach. There are ten graphs, corresponding to the ten error variance specifications in Figure 6. Speed results are obtained from Altera MaxPlus II on the fully placed and routed design in an Altera Flex10kRC240-3 device, however similar results are expected for any lookup-table based FPGA. The Pareto optimal points[1] in Figure 9 are joined by solid lines.

The results of Figure 9 demonstrate that although the difference in area is small between short-run simulation-based scaling wordlength-optimized systems and those resulting from Algorithm 2, there is a significant speed difference. The source of this consistent speedup, averaging 27.6% over uniform wordlength and 23.7% over optimized wordlength structures, is illustrated in Figure 10 where the saturator locations and degree (number of most significant bits saturated) are illustrated for a single optimization example.

Comparing Figures. 10(a) and (b), simulation-based scaling has resulted in a large number of low degree saturators. In contrast the optimized saturators are fewer in number, but are generally of higher degree. Although aiming to reduce system implementation area, Algorithm 2 has also resulted in significant

[1]A Pareto optimal point is one that is not dominated in all design objectives by another design point.

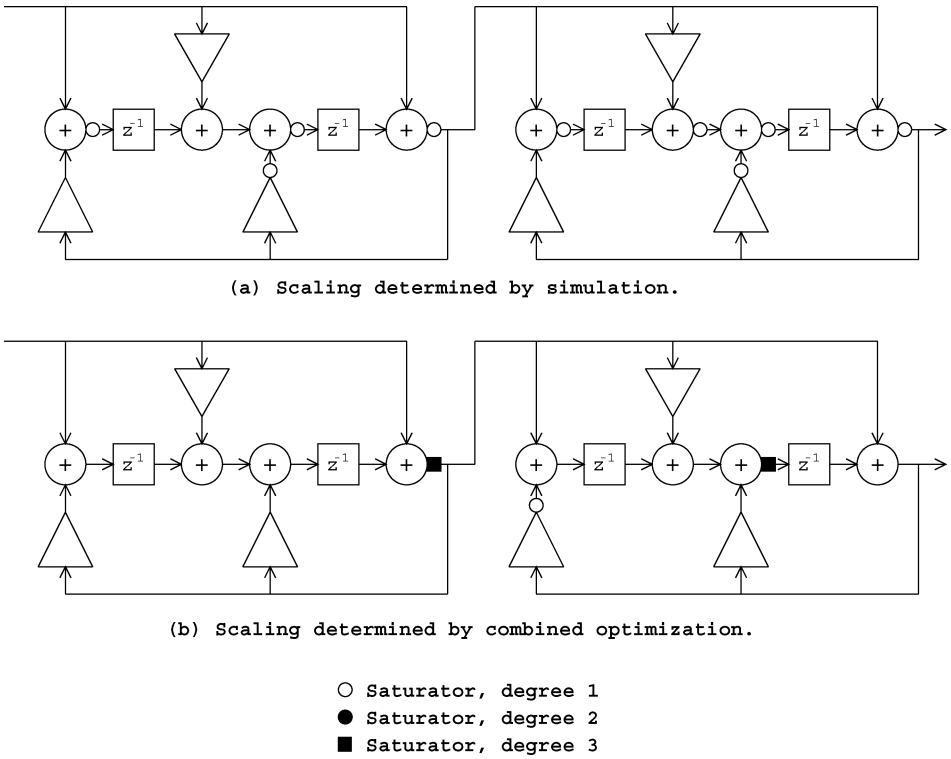


Fig. 10. Saturator locations and degrees for the fourth order narrow bandpass elliptic IIR filter.

speedup over simulation-based approaches by using only a small number of saturators. The Cauchy-Schwartz bound tends to drive the solution towards using fewer saturators in order to minimize the potential error cross-correlation effects.

7. CONCLUSIONS

A novel technique for design automation of saturation arithmetic systems has been presented. The technique is based on an analytic saturation noise estimation method, which estimates the average-case noise power, hence signal-to-noise ratio. In contrast to truncation and rounding, auto- and cross-correlations between linearized saturation nonlinearities have been accounted for using a bound derived through the Cauchy-Schwartz inequality. Techniques have been presented to reduce the slackness associated with such a bound.

The heuristic presented in Constantinides et al. [2001] has been extended to incorporate combined scaling and wordlength optimization. The results of such an optimization have been discussed for real examples of DSP systems and contrasted with more traditional approaches to scaling optimization. It has been shown that allowing rare saturation errors can result in fast and small implementations of IIR filters when the poles of the filter are close to the unit circle. Such IIR filters are particularly suited to saturation arithmetic design,

due to their long impulse responses. The algorithm described in this paper can be used to determine an appropriate location and severity for each saturation nonlinearity.

For future work, it is possible to consider power consumption as a factor in the optimization problems discussed in this paper. It is highly likely that the multiple wordlength design paradigm can be effectively used to reduce power consumption [Constantinides 2003]; indeed Ercegovac et al. [1999] have proposed a multiple wordlength approach for addition targeting low power in ASIC implementations. Another promising direction of future research concerns methods to perform wordlength and scaling optimization for nonlinear or time-varying systems [Constantinides 2003].

ACKNOWLEDGMENTS

The authors wish to acknowledge the support of the Engineering and Physical Research Council (U.K.) and Hewlett-Packard Laboratories, and the helpful comments from the anonymous reviewers.

REFERENCES

ALTERA CORPORATION. 1998. *Altera Databook*. San Jose: Altera Corporation.

BENEDETTI, A. AND PERONA, P. 2000. Bit-width optimization for configurable DSPs by multi-interval analysis. In *Proceedings of the 34th Asilomar Conference on Signals, Systems and Computers*.

CHUNG, K.-L. 1974. *A Course in Probability Theory*. Academic Press, New York.

CMAR, R., RIJNDERS, L., SCHAUMONT, P., VERNALDE, S., AND BOLSENS, I. 1999. A methodology and design environment for DSP ASIC fixed point refinement. In *Proceedings on Design Automation and Test in Europe*, München.

CONSTANTINIDES, G. A. 2001. *High Level Synthesis and Word Length Optimization of Digital Signal Processing Systems*. Ph.D. thesis, University of London.

CONSTANTINIDES, G. A. 2003. Perturbation analysis for word-length optimization. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*.

CONSTANTINIDES, G. A., CHEUNG, P. Y. K., AND LUK, W. 2001. The multiple wordlength paradigm. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines* Rohnert Park, CA, April–May.

DEMICHELI, G. 1994. *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, New York.

ERCEGOVAC, M., KIROVSKI, D., AND POTKONJAK, M. 1999. Low-power behavioural synthesis optimization using multiple precision arithmetic. In *Proceedings of the 37th Design Automation Conference*.

FREETEL. 1993. Esprit project 6166: FREETEL database.

JOHNSON, K. K. AND SANDBERG, I. W. 1995. A separation theorem for finite precision digital filters. *IEEE Trans. Circuits and Syst. I 42*, 9 (September), 541–545.

KIM, S., KUM, K., AND SUNG, W. 1998. Fixed-point optimization utility for C and C++ based digital signal processing programs. *IEEE Trans. Circuits Syst. II 45*, 11 (November), 1455–1464.

KUM, K.-I. AND SUNG, W. 2001. Combined word-length optimization and high-level synthesis of digital signal processing systems. *IEEE Trans. Comput. Aided Design 20*, 8 (August), 921–930.

LEONG, M. P., YEUNG, M. Y., FU, C. W., HENG, P. A., AND LEONG, P. H. W. 1999. Automatic floating to fixed point translation and its application to post-rendering 3D warping. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, 240–248.

LIU, B. 1971. Effect of finite word length on the accuracy of digital filters—a review. *IEEE Trans. Circuit Theory CT-18*, 6, 670–677.

MITRA, S. K. 1998. *Digital Signal Processing*. McGraw-Hill, New York.

OPPENHEIM, A. V. AND WEINSTEIN, C. J. 1972. Effects of finite register length in digital filtering and the fast fourier transform. *IEEE Proceedings 60*, 8, 957–976.

STEPHENSON, M., BABB, J., AND AMARASINGHE, S. 2000. Bitwidth analysis with application to silicon compilation. In *Proceedings of the SIGPLAN Programming Language Design and Implementation*, Vancouver, British Columbia, (June).

WILLEMS, M., BÜRSGENS, V., KEDING, H., GRÖTKER, T., AND MEYER, M. 1997. System-level fixed-point design based on an interpolative approach. In *Proceedings of the 34th Design Automation Conference* (June). 293–298.

Received May 2002; revised March 2003; accepted April 2003