Word-length selection for power minimization via non-linear optimization

Jonathan A. Clarke, George A. Constantinides, Peter Y. K. Cheung

This paper describes the first method for minimizing the dynamic power consumption of a Digital Signal Processing (DSP) algorithm implemented on reconfigurable hardware via word-length optimization. Fast models for estimating the power consumption of the arithmetic components and the routing power of these algorithm implementations are used within a constrained non-linear optimization formulation that solves a relaxed version of word-length optimization. Tight lower and upper bounds on the cost of the integer word-length problem can be obtained using the proposed solution, with typical upper bounds being 2.9% and 5.1% larger than the lower bounds for area and power consumption, respectively. Heuristics can then use the upper bound as a starting point from which to get even closer to the known lower bound. Results show that power consumption can be improved by up to 40% compared to that achieved when using simple word-length selection techniques, and further comparisons are made between the minimization of different cost functions that give insight into the advantages offered by multiple word-length optimization.

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, Power Consumption

Additional Key Words and Phrases: Signal processing, word-length, bitwidth, synthesis, power consumption

1. INTRODUCTION

Implementations of DSP algorithms in custom hardware allow fine grain control over the number of bits (i.e. the word-lengths) used to communicate data between the algorithm's operations. Automatic word-length optimization techniques allow for the area, power or delay of an algorithm implementation due to word-length selection to be traded off automatically for algorithm accuracy and can achieve improvements of up to 80% in hardware area consumed over simple word-length selection techniques [Constantinides 2003]. They also allow an increase in the level of abstraction away from the finite precision details of DSP algorithms, and hence have the potential of becoming an important part of the array of Computer Aided Design (CAD) tools available to combat the increasing complexity of designs and computational hardware.

However, the mainstream adoption of word-length optimization as a CAD tech-

The authors would like to acknowledge the support of Synplicity, Xilinx, Celoxica and the EPSRC under grant numbers EP/C512596/1, EP/C549481/1 and EP/E00024X/1.

nique has been hampered by the complexity of the optimization: optimal word-length selection is NP-hard, as shown in Constantinides and Woeginger [2002].

Hence the large design space that must be searched to find optimal solutions to word-length optimization problems has constrained previous work to be focussed on the minimization of implementation costs that can be estimated with low computational effort such as the sum of the signal word-lengths used or the area occupied by arithmetic components. In contrast the work presented here is able to perform word-length optimization for power consumption minimization for the first time, by using fast models to estimate the power consumed in the arithmetic components and routing wires of a DSP algorithm implemented in reconfigurable hardware. Arithmetic component power is estimated using a set of macro-models [Clarke et al. 2005], whilst a novel 'rough placement' technique [Clarke et al. 2007] is used to provide capacitance estimates for the routing wires in a system very quickly, without performing true design placement.

The location of optimal or near-optimal solutions to word-length optimization problems is facilitated in this work by the use of a novel word-length selection technique that uses constrained non-linear optimization to solve a version of word-length optimization without the constraint that word-lengths be integral. This allows tight lower and upper bounds on the optimal integral solution to word-length optimization problems to be found, with the upper bound providing a starting point for heuristics to improve on the integer word-length problem.

The main contributions of the work presented in this paper are the following:

—Fast models for estimation of the dynamic power consumption of the arithmetic operations and routing wires of DSP algorithms in reconfigurable hardware.

—A method for solving a relaxed version of word-length optimization where integer word-lengths are not enforced.

—A method for using the result of this relaxed word-length optimization problem to establish tight *a posteriori* bounds on the optimal solution of the equivalent integer problem.

—The first results that allow detailed analysis and comparison of word-length optimization techniques for uniform word-lengths, and system area or power consumption minimization.

The remainder of this paper is organized as follows. The following section summarizes existing work in the fields of both word-length optimization and the fast estimation of power consumption in reconfigurable hardware. Section 3 describes a method for quickly estimating the power consumed in the arithmetic components of DSP algorithms implemented on reconfigurable hardware, as well as a method for estimating the power consumed in the routing wires of these implementations. In Section 4 a novel method for word-length selection is presented, and finally results obtained by applying this method to both area and power cost functions over a variety of benchmark systems are detailed in Section 5.

2. BACKGROUND

2.1 Word-length optimization

Word-length optimization is the process of selecting appropriate word-lengths for the communication of data between the operations of an algorithm implementation so as to minimize some cost of that implementation whilst ensuring that its accuracy due to data quantization is within acceptable limits. For the DSP algorithms implemented on reconfigurable hardware that are targeted in this work each signal between algorithm operations is represented in two's complement, and the signal's word-length determines the number of bits used to represent the signal.

Unfortunately finding optimal solutions to word-length optimization problems is NP-hard, as shown by Constantinides and Woeginger [2002]. Hence previous work on word-length optimization has provided a number of sub-optimal techniques for finding 'good' solutions to the problem in reasonable amounts of time. These methods are discussed in Section 2.1.2, however they rely on the examination of a large number of points in the word-length optimization design space. In order to make these examinations possible in reasonable amounts of time it is essential that both the cost and accuracy of an algorithm can be calculated extremely quickly. A significant amount of prior work has thus been devoted to fast techniques for determining the accuracy of an algorithm implementation in particular, as detailed in the following subsection.

2.1.1 *Calculation of algorithm accuracy.* The fast estimation of algorithm accuracy is critical to a successful word-length optimization technique. In fixed point two's complement representation error can arise from two sources: *i)* overflow errors where the absolute value of a number is too large to be stored in the given representation, and *ii)* quantization errors where there are insufficient fraction bits to store all of the fractional part of a number.

Whilst quantization error generally results in a small loss of accuracy, overflow errors can be devastating in two's complement because of 'wrap-around'. As a result even a single overflow error can cause complete loss of accuracy in an algorithm. However an accumulation of many small quantization errors in the operations of an algorithm will also cause a substantial loss of accuracy.

In order to mitigate the effects of these sources of error it is generally best to ensure overflow almost never occurs, whilst quantization noise in each operation must be carefully balanced to achieve a desired accuracy. Significant research has been devoted to these in existing work, summarized briefly in the following sections.

2.1.1.1 *Overflow prevention: range analysis.* It is sufficient to know the range of values that a signal may take in order to select the largest valued bit required to represent that range and hence prevent overflow in that signal. Once the largest value M taken by a signal in a system is known, the position p of the largest-valued bit required to represent M in two's complement without overflow is:

$$p = \lceil \log_2(M) \rceil + 1 \tag{1}$$

where $\lceil x \rceil$ represents the rounding up of x to the nearest integer. The value p is referred to as the scaling of a signal for the remainder of this paper.

A number of methods are available for the performance of range analysis and it is assumed in this work that any one of these could be used prior to the more complex task of word-length selection. Important methods used in previous work are simulation [Sung and Kum 1995], the combination of simulation and statistical techniques in Kim et al. [1998] and Ozer et al. [2004], worst-case range propagation [Wadekar and Parker 1998], range propagation via affine arithmetic in Lee et al. [2006], and finally use of the l_1 scaling in Constantinides et al. [2001].

2.1.1.2 *Quantization noise analysis.* Whilst range analysis can be performed once and for all before word-length selection, quantization noise must be estimated for every point in the search space that is examined during word-length selection. Extremely fast methods for performing quantization noise analysis are thus essential.

The method of perturbation analysis [Constantinides 2003] is used to determine the variance of the quantization noise at the output of a system in this work. This method first measures the sensitivity of the algorithm output to noise injected individually into each signal by creating a linearized 'small signal' model of the algorithm using a first order Taylor approximation of each operation, and simulating it once for each signal with noise injected into a different signal for each simulation. The sensitivities s_i of each signal i are then used to estimate the quantization noise variance σ_y^2 at the algorithm output due to the quantization noise of variance σ_i^2 in each signal i by using:

$$\sigma_y^2 = \sum_{i \in S} s_i \sigma_i^2 \tag{2}$$

where S is the set of signals in the algorithm and the variance σ_i^2 due to truncating a signal i from word-length wl_1 to wl_2 is given by the truncation noise model in Constantinides et al. [1999]:

$$\sigma_i^2 = \frac{1}{12} 2^{2p_i} (2^{-2wl_2} - 2^{-2wl_1}) \tag{3}$$

where p_i is the scaling of signal i determined during range analysis.

A number of other methods for quantization noise analysis have been proposed in existing work as follows. Simulation has been used in [Kim et al. 1998] but is far too slow to allow optimization of any but the smallest systems. Fast analytical techniques include the multi-interval arithmetic method in Benedetti and Perona [2000] that calculates the smallest absolute values that must be represented in each signal to avoid quantization, propagation of worst-case errors via affine arithmetic [Lee et al. 2006], use of the L_2 norm to estimate the output quantization noise variance in Constantinides et al. [2001], and worst-case error propagation via Taylor-expansion in both Wadekar and Parker [1998] and Gaffar et al. [2004].

In general the fast analytical techniques for noise analysis are by far preferable for use within word-length optimization due to the number of points in the search space of the problem. Perturbation analysis is used here as it is the only fast technique that can be applied to non linear systems containing feed-back loops.

2.1.2 *Word-length selection.* As explained in Section 2.1 word-length optimization is NP-hard. This section describes prior work for finding optimal or near-optimal solutions to the problem.

Uniform word-length optimization uses the same word-length for all signals in an algorithm, reducing the dimensionality of the problem to a single dimension. Optimal solutions can thus be found by using a binary search [Knuth 1997] in $O(\log_2 wl_{max})$ tests of the search space where wl_{max} is a large word-length known to satisfy accuracy constraints. Although this is the fastest word-length optimization method, results are poor: multiple word-length optimization offers significant improvements in cost over this method of up to 80% in hardware area consumed, for example [Constantinides 2003]. In Constantinides et al. [2002] optimal word-length selection is formalized as a Mixed Linear Integer Programming (MILP) problem, though these are NP-hard in general. Two iterative heuristics find near-optimal solutions by either starting from an infeasible point (due to excessive quantization noise) and gradually reducing noise [Kim et al. 1998], or by starting from a feasible point and reducing cost as in Constantinides [2006]. They may get caught in local optima but appear to achieve good results (0.7% higher area than optimal solutions for very small systems [Constantinides et al. 2002]), and are able to quickly converge to near-optimal solutions. Simulated annealing has been used in Lee et al. [2006], which is a stochastic optimization technique that is sometimes able to 'jump out' of local optima with a probability that decreases during the course of optimization.

Geometric programming has been used to find the optimal solution to a relaxed version of word-length optimization in Chan and Tsui [2006], in a similar manner to this work. However the authors only target the minimization of the sum of word-lengths of an algorithm implementation. A Geometric programming formulation cannot be used to optimize dynamic power consumed in a system as estimated using the methods summarized in this paper in Section 3.

Additionally in Chan and Tsui [2006] the authors do not use the optimal solution to the relaxed version of the word-length optimization problem as a lower-bound on the integer version of the problem to aid in the search for the optimal integer solution. In contrast the work proposed in this paper is able to give tight bounds on the optimal solutions to word-length optimization problems by using constrained non-linear optimization, as described in Section 4.

Finally, the technique in Doi et al. [2006] uses a Sequential Quadratic Programming (SQP) formulation in order to solve word-length optimization problems, however the method proposed here provides the following significant improvements over this previous work.

Power minimization via the use of logic power consumption models and routing power consumption models within a word-length optimization framework.

Tight lower and upper bounds on the minimal cost of a word-length optimization problem calculated during the course of the proposed technique.

Means for estimating the effects of the correlation in truncation noise that occurs due to the different word-lengths of fan-out branches, and for performing word-length optimization in spite of this non-convex phenomenon [Constantinides et al. 2002].

Constraints required to prevent zero padding of signals not present in Doi et al. [2006] where in contrast the optimization will be misled as naive quantisation noise estimation makes it appear as if increasing the word-length of a signal by zero-padding its least-significant bits increases the signal's accuracy.

2.2 Power consumption estimation

The area consumption of algorithm implementations has previously been the target for minimization during word-length optimization, as it is relatively simple to predict for custom implementations of arithmetic intensive circuits (see [Clarke 2008]). It is more difficult to quickly estimate power consumption accurately however. This section provides an overview of power dissipation and estimation in reconfigurable hardware, specifically Field Programmable Gate Arrays (FPGAs), whose high performance and programming complexity, both resulting from their fine-grain customizability, make them an ideal platform for the exploration of a CAD tool such as word-length optimization for power consumption minimization.

Power is dissipated in digital CMOS circuits such as FPGAs due to: *i)* small leakage currents in transistors that are in cut-off mode, *ii)* charging and discharging of parasitic capacitances when the outputs of logic gates change between logic zero and one [Roy and Prasad 2000].

In FPGAs leakage power consumption is fixed due to the construction of the device, whilst dynamic power is dependent on the rates at which logic gates within the device switch and the size of the parasitic capacitances they drive. Hence the algorithm implemented on a device has a large affect on dynamic power consumption only. The dynamic power consumed by the switching of a parasitic capacitance C at a rate of α transitions per clock cycle is given by:

$$P = \frac{\alpha}{2} \cdot f_{clk} \cdot C \cdot V_{dd}^2 \tag{4}$$

where f_{clk} and V_{dd} are the clock frequency and supply voltage of the circuit.

By knowing the capacitance and switching activity rate of every signal in a system it is thus possible to estimate dynamic power consumption. Capacitance values can be obtained from a placed and routed circuit, whilst activities can be obtained from switch-level simulation, but these are too computationally expensive to be used to evaluate points in the design space during word-length selection. Faster estimation can be obtained by abstraction.

In Poon et al. [2002] transition density is used to estimate the activity in combinational logic by using signal probabilities to propagate activity values through the circuit, however the method has high complexity for larger combinational circuits, and requires a LUT-level description of the logic in the circuit that would need to be constructed for every point in the design space at high computational cost.

Pre-placement estimation of capacitance is considered in both Anderson and Najm [2004] and Bhoj and Bhatia [2007], but although both of these exhibit high-levels of accuracy placing every circuit in the design space would be infeasible.

Macro-models have been constructed in Shang and Jha [2001], Jiang et al. [2004] and Choy and Wilton [2006]. These estimate dynamic power in an arithmetic component such as a multiplier or adder by using pre-measured power values that were obtained when similar input activities were used to those currently observed on the component in high-level simulation. Because power consumption values for components have been pre-measured, macro-models offer extremely high speed estimation of power consumption. However, the work in Shang and Jha [2001] did not include the input word-length of a component in its macro-model, thus requiring one macro-model for every input word-length of each component to be characterized and stored in memory. In Jiang et al. [2004] the authors include the input word-length of a component in its macro-model and use characterized equations of the variables N (component input word-length), A_i (average input activity in the bits of input i), and S_i (average signal probability in the bits of signal i). However 20 terms of up to cubic order combinations of the above parameters are used without justification in each component's macro-model. Our work in this area has shown that far fewer terms of lower order can be used without significant loss of accuracy [Clarke et al. 2005]. Finally in Choy and Wilton [2006] the authors use a macro-model for the power consumed in embedded multipliers that uses the average activity across the multiplier's two inputs as the parameter from which to predict power consumption via a non-linear equation.

The accuracy and low computational cost of macro-models make them ideally suited for use within word-length optimization for power minimization. However, even if all the components in a system can be mapped to a macro-model in order to predict their power consumption, there remains the problem of estimating the power consumed in the configurable routing wires that connect components together. A fast method for estimating the power consumed in the routing wires of a system from Clarke et al. [2005] is summarized in this paper in Section 3, as well as macro-models for estimating the power consumed in the arithmetic components in a DSP algorithm implemented on reconfigurable hardware [Clarke et al. 2007]. These models form the first complete dynamic power consumption estimation technique that is suitable for use within word-length optimization, and are used for that purpose in Section 4.

3. POWER CONSUMPTION MODELS

This section describes the set of power consumption models used for word-length optimization in this work. The models described are characterized for the Virtex II Pro FPGA available from Xilinx [Xilinx Inc. 2004] and are based on those described in previous work by the authors in Clarke et al. [2005] and Clarke et al. [2007]. A set of macro-models are used to estimate the power consumed in arithmetic components as described in Section 3.1, whilst a very fast 'rough placement' method is used to provide bounding box estimates to aid capacitance estimation for inter-routing wires (that connect *between* arithmetic components) in Section 3.2.

Although the models used in this paper have been characterized for FPGAs, the proposed technique could also be used to perform word-length optimization for algorithms implemented on ASICs, though the models described would need to be re-characterized using training data from ASICs.

3.1 Arithmetic component power

This section describes the macro-models used to estimate the power consumed in adders, embedded multipliers and delay registers.

Each component's macro-model is parameterized by the word-lengths of the component's inputs and the activity estimation parameters of the Dual-Bit Type (DBT) model from Landman and Rabaey [1996] for each of the component's input signals. As shown in the DBT work the standard deviation and lag-1 autocorrelation of a Gaussian signal are sufficient to estimate the bit-level activities in that signal, and work conducted by the authors in Clarke et al. [2008] has demonstrated that the bit-level activities and correlations implied by the word-level parameters of the DBT model are sufficient to estimate the activity within two's complement adders and multipliers. Hence the power consumed in these components can be estimated using only the standard deviation and lag-1 autocorrelation of the input signals to these components, assuming that the parasitic capacitances of the SLICEs within these components are uniform across the device and that their input signals are well approximated by Gaussian signals. For signals within DSP systems this approximation is known to be valid [Landman and Rabaey 1996].

The standard deviation of each signal in a system is linked to its range, as are the scalings of each signal which are established during range analysis. As signals are scaled in a consistent manner in relation to their range, it becomes unnecessary to include the standard deviation of a signal in each macro-model, unless the input signals to a component must be aligned in some way (as is required for addition where the binary point of the inputs must be aligned) [Clarke et al. 2005].

The resulting macro-model for the power P_{add} consumed in addition is given by:

$$P_{add} = nC_0(\sigma_r, \rho_a, \rho_b) + C_1(\sigma_r, \rho_a, \rho_b) \tag{5}$$

where $\sigma_r = \frac{\sigma_S}{\sigma_L}$ i.e. the standard deviation σ_S of the input signal with smaller standard deviation divided by that of the signal with larger standard deviation σ_L, and ρ_a, ρ_b are the lag-1 autocorrelations of the adder's input signals A and B, respectively, C_0, C_1 are characterization coefficients that are functions of the above statistical parameters, and n is the number of input bits that require logic because they 'overlap' due to input signal word-lengths and scalings. Characterized values of C_0 and C_1 are stored in a table according to the corresponding values of σ_r, ρ_a, ρ_b, with linear interpolation used to estimate values of C_0 and C_1 between characterization points.

Although characterized macro-models for the power consumption in multipliers implemented in LUTs have been developed [Clarke et al. 2005], embedded multipliers have been used exclusively in the benchmarks used in this paper. However the same word-level activity parameters are applicable for the construction of power macro-models of these embedded components. No alignment of the inputs of multipliers is necessary so the standard deviation parameters from the DBT model can be omitted for each input signal, however the lag-1 autocorrelation coefficients of each input are still necessary. The relationship between the input word-lengths of embedded multipliers and their power consumption is non-linear due to multiplications larger than 18×18-bits requiring several embedded blocks, whilst smaller multiplications requiring only one. Hence a surface of power consumption values for each multiplication must be used, as shown in Figure 1.

Different surfaces are measured for several combinations of values of ρ_a, ρ_b, the lag-1 autocorrelation coefficients of the inputs A and B of the multiplier; for values of ρ_a, ρ_b that lie between characterization points, new surfaces that relate embedded multiplier input word-length to power consumption must be interpolated from existing data points. As all the DBT model statistical parameters used during power consumption estimation are measured during word-level simulation *before* word-length optimization however, the creation of new surfaces for embedded multiplier power consumption is not performed during word-length selection, and hence estimation of power consumption in embedded multipliers is quickly accomplished

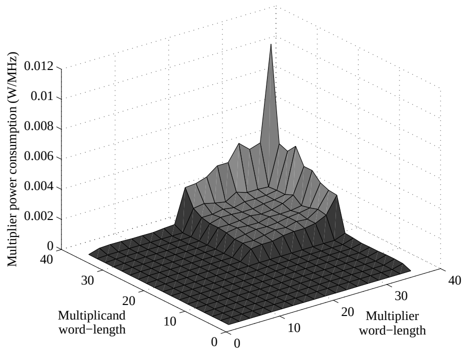

Fig. 1. The power consumed by multiplications implemented in the embedded blocks available in Virtex II Pro FPGAs for various input word-lengths when the multiplier is driven by Gaussian uncorrelated inputs.

by interpolation between sampled points of the appropriate surface.

3.2 Routing power

Whilst the power consumed in the logic and routing wires that connect logic elements within arithmetic components are estimated by the macro-models described in the preceding section, the power consumed in the routing wires that connect components together (i.e. the inter-component routing wires) is not. Unfortunately the capacitance of these wires is mostly determined by decisions made during the placement and routing steps of the FPGA tool flow, which are too computationally expensive to perform for each point in the design space evaluated during word-length selection.

In [Clarke et al. 2007] a method for estimating the capacitance of inter-routing wires was proposed that quickly performs a 'rough placement' of the arithmetic components in a DSP system to increase the accuracy of capacitance estimates. The proposed method takes advantage of the following information available during word-length optimization of a high-level description of a DSP system: the type of each block used (adder, multiplier, etc.), the topology of the circuit, the number of output nets of each component, the area of each component (which can be estimated very easily given the component's inputs as described in Clarke [2008]), and finally the fan-out of each output wire of each component.

The 'rough placement' method trades off accuracy in terms of placement fidelity to achieve faster computation time. To simplify the placement problem, the following assumptions are made: *i)* all components are square-shaped, *ii)* connections between components depart and arrive from the centre of each component, *iii)* components can overlap in the rough placement, and *iv)* the inputs of the system lie at the bottom left corner of the FPGA, and outputs lie in the top right corner.

With these assumptions the placement problem becomes symmetrical along the diagonal line between the system inputs and outputs, hence a single dimension version of the two-dimensional placement problem can be solved. This as well as the simplifying assumptions above greatly reduces the complexity of the placement problem, though some accuracy is lost.

The information and assumptions summarized above are used to formulate a linear program that approximates timing driven placement in single dimension placement problem described above. The bounding box values BB_i of each wire i are extracted from this placement and used along with each wire's fan-out value FO_i to estimate its capacitance C_i as follows:

$$C_i = \alpha BB_i + \beta FO_i \tag{6}$$

where α and β are characterization coefficients that have been predetermined for the Virtex II Pro from a selection of DSP benchmarks [Clarke et al. 2007].

The fan-out FO_i of an inter-routing wire i is easily determined as follows. Each fan-out to a register increases FO_i by one. Each fan-out to an adder increases FO_i by one, except if i is the MSB of a signal that must be sign extended but again this is easily determined according to the scalings of the adders inputs. Each fan-out to a multiplication increases FO_i by one, except if this is implemented in multiple embedded blocks in which case if the word-lengths of the multiplier and multiplicand signals are n and m respectively the increases are:

$$\text{multiplier signal fan-out increase} = \lceil m/18 \rceil \tag{7}$$

$$\text{multiplicand signal fan-out increase} = \lceil n/18 \rceil \tag{8}$$

It is assumed that all components in a system have registered outputs, hence inter-routing wire activity values are known during word-length optimization via the DBT model and the word-level statistics gathered for inter-routing signals gathered before word-level optimization.

3.3 Power consumption model accuracy

Existing work by the authors published in [Clarke et al. 2005], [Clarke et al. 2006], [Clarke et al. 2007] and [Clarke 2008] has shown that the arithmetic component power models used within the word-length optimization framework described in this paper are accurate to within a with Mean Relative Error (MRE) of 5.1% and 3.1% for adders and embedded multipliers respectively [Clarke 2008], whilst the capacitance estimation technique for inter-component routing wires shows a Root Mean Square Relative Error (RMSRE) of 42% [Clarke 2008] and the activity estimation technique based on the approximation proposed in the DBT method [Landman and Rabaey 1996] has a MRE of 2.8% [Bobba et al. 1999].

Although the RMSRE in capacitance for inter-component routing made appears high it has been shown in [Clarke 2008] that the technique is able to estimate the capacitance of the inter-routing wires with highest capacitance (i.e. power consumption "hot spots") very accurately, though the error in estimating the capacitance of short wires that connect closely packed components can be high as the placement approximation technique used is unable to account for the fine grain decisions made by the stochastic optimization algorithm used for FPGA logic placement. This "noise floor" induced by the stochastic placement has been quantified in [Clarke et al. 2007].

Thus the proposed power models are well suited for use within a word-length optimization procedure as they exhibit a high level of accuracy whilst requiring little computational effort in order to be evaluated.

4. CONSTRAINED NON-LINEAR WORD-LENGTH SELECTION

This section describes a word-length selection technique that finds optimal solutions to non integer word-length optimization problems for algorithms whose quantization noise can be estimated using the perturbation analysis method summarized in Section 2.1.1 and whose power consumption is estimated using the models described in Section 3. The proposed method provides tight upper and lower bounds to the optimal integer solution to a word-length optimization problem, with the upper bound providing a starting point for heuristics to find the better integer solutions. After the description of the proposed technique Section 5 compares the results of optimizing for different cost functions (area, power, etc.) and provides an analysis of the method's performance.

4.1 Sequential Quadratic Programming

Sequential Quadratic Programming (SQP) is an optimization method for finding a minimum of a function $f(\mathbf{x})$ as expressed in (9), where $\mathbf{x}$ is a vector and $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are constraint functions that may return a vector result, and at least one of these functions is non-linear. A good review of SQP is given in Boggs [1995], and is summarized below.

$$\begin{array}{rl} \text{minimize} & f(\mathbf{x}) \\ \text{subject to:} & \mathbf{g}(\mathbf{x}) = 0 \\ & \mathbf{h}(\mathbf{x}) \leq 0 \end{array} \tag{9}$$

SQP is an iterative approach to finding a minimum $\mathbf{x^*}$ of the function $f(\mathbf{x})$, where the constraint functions are satisfied. At each iteration the Lagrangian function of the optimization problem (9) is approximated as a quadratic programming problem at the current point $\mathbf{x}^k$. The solution of this quadratic program is used to move to a point $\mathbf{x}^{k+1}$, closer to $\mathbf{x^*}$.

In order to construct the quadratic subproblem at each iteration the first and second order partial derivatives $\mathcal{L}'(\mathbf{x})$ and $\mathcal{L}''(\mathbf{x})$ of the Lagrangian at the current point $\mathbf{x}^k$ must be known. Calculating the second order partial derivatives (i.e. the Hessian) of the Lagrangian is usually computationally expensive, so instead an estimate of the Hessian is updated after each SQP iteration using the first order partial derivatives (i.e. the gradients) of $f(\mathbf{x})$, $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ with respect to $\mathbf{x}$, by using the BFGS formula summarized in Boggs [1995]. Calculating the first-order derivatives of the Lagrangian $\mathcal{L}'(\mathbf{x})$ also requires the first-order partial derivatives of the cost and constraint functions, either from analytic equations to calculate these, or by using the finite difference method on the cost and constraint functions to measure their gradient at $\mathbf{x}^k$.

Once the quadratic subproblem has been solved a direction along which to proceed is known, and a merit function gives the distance to travel in this direction to obtain $\mathbf{x}^{k+1}$. If either $f(\mathbf{x})$ or the constraint functions are not convex the minimum found by SQP may only be a local minimum, however if they are convex the global minimum of the problem will be found.

Word-length optimization problems are not directly solvable by SQP, for a variety of reasons that are listed in the following section. It has however been possible to slightly modify the word-length optimization problem in order to allow an approximation of the problem to be solved quickly and optimally by SQP.

4.2 Word-length optimization modifications for SQP

To use SQP to find solutions for word-length optimization we must formulate our problem as shown in (9), i.e. $\mathbf{x}$ is a vector representing the signal word-lengths in the system, $f(\mathbf{x})$ is the cost function to be minimized, e.g. area or power, and $\mathbf{h}(\mathbf{x}) \leq 0$ will be used to express a constraint on the noise variance at the output of the circuit. If the circuit has multiple outputs there will be one noise constraint for each output, however, without loss of generality, we assume for the remainder of this paper that there is only one system output and hence only one noise constraint.

Although the non-linear nature of the area, power and noise variance functions indicates that SQP may be a suitable candidate for solving word-length optimization, as mentioned earlier there are several reasons why this is not possible. These reasons are dealt with in the following subsections.

4.2.1 *Relaxation of integer constraints.* This section concentrates on efforts to make the word-length optimization problem optimally solvable when the constraint on integer word-lengths is relaxed (we will call this the non integer problem from now on). Although the solution of this relaxed problem will not be integral, its cost will provide a lower-bound on the cost that can be achieved in the integer problem.

Once an optimal solution to the non integer word-length optimization problem has been found an upper bound on the optimal cost of the integer version of the problem can be obtained by rounding up the word-lengths to the nearest integer. This upper bound will give a solution with slightly increased cost but with better error performance.

The upper bound word-lengths can then be treated as a starting point for heuristics for solving the integer version of the word-length optimization problem. The iterative heuristic described in Constantinides [2006] is used for this purpose, which gradually decreases word-lengths until no more reductions can be made without introducing too much noise into the system output.

Solving the word-length optimization problem with relaxed integer constraints clearly provides significant benefits when trying to find the optimal solution of the integer version of the problem, however there remain several issues that prevent optimal solutions to the non integer word-length from being found by SQP that are approached in the following subsections.

4.2.2 *Simplification of noise model to convex form.* Any signal that fans-out to multiple components is allocated one word-length variable for each fan-out branch of the signal, allowing individual control over the cost of each component connected to by the signal. Hence each fan-out branch can be a different truncated version of the original signal, and as a result branches that share some number of truncated bits also share the same truncation noise due to the removal of these bits.

The noise model in Constantinides et al. [2002] correctly models the correlated noise injected into fan-out branches by using the noise injection scheme depicted in Figure 2(a). In this scheme, the fan-out branches of a multiple fan-out signal are ordered from longest to shortest word-length with each branch 'tapped off' from the original source in this order. Noise is injected before each tap in order to model truncation from the word-length w_n before the tap to the next (shorter)

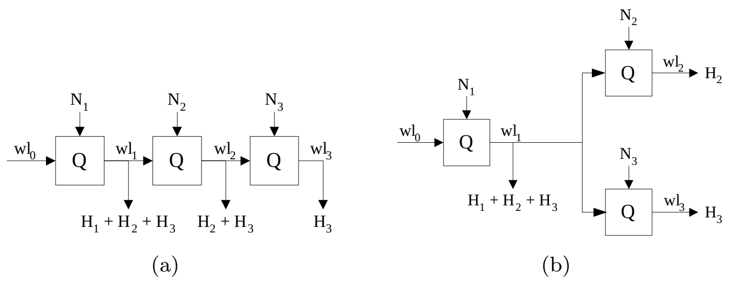

Fig. 2. Noise injection schemes for truncating a signal with three fan-out branches from word-length wl_0 to word-lengths wl_1, wl_2 and wl_3. The quantizer blocks Q inject noise signals N_1, N_2 and N_3 into the branches of the signal as a result of truncation. These noise signals pass through combinations of the transfer functions H_1, H_2 and H_3, to the system output. Scheme (a) correctly calculates the noise injected when $wl_1 > wl_2 > wl_3$. Scheme (b) only accounts for the truncation noise common to branches 1, 2 and 3.

word-length w_{n+1} after the tap.

If a change in word-length results in a re-ordering of the fan-out branches (because these must be kept in decreasing order of word-length) then the subsets of branches into which noise is injected changes. The new subsets of branches (after re-ordering) cause a different response at the system output to the truncation noise injected into each tap. Hence a decrease in word-length that causes a re-ordering of fan-out branches can cause a *decrease* in noise at the system output; this makes the system output noise non-convex for certain changes in fan-out branch word-length.

To avoid this non-convexity, in this work only the correlated noise that is common to *all* fan-out branches is modelled, as depicted in Figure 2(b). Correlated noise resulting from truncation to the longest word-length branch is injected into all branches of a multiple fan-out signal. Branches whose word-length is shorter than the longest word-length are injected with extra *un*correlated noise due to truncation from the longest word-length to the shorter word-length in that branch.

Two states can be identified for the word-length of each branch in a multiple fan-out signal under the proposed noise injection scheme: for each branch either *i)* the word-length of the branch is the longest in the fan-out, or, *ii)* the word-length of the branch is not the longest word-length in the fan-out. For the same reasons as the noise model used in Constantinides et al. [2002], non-convexity can occur when any word-length changes from state *i)* to state *ii)*, or vice versa.

However for the simplified model it is possible to determine the order of word-lengths such that the optimal solution is always found. This is done by constraining the word-length belonging to the fan-out branch that is most sensitive to truncation noise to be larger than all other word-lengths in the fan-out. Shorter word-lengths can then be assigned to the other branches that are less sensitive to truncation noise, giving a lower system cost than could be achieved using other configurations.

To form these constraints a system that has a total of b individual fan-out branches and m multiple fan-out signals requires $b-m$ linear inequality constraints to be included in the formulation of the word-length optimization problem.

Whilst the optimum ordering of branches can be determined for the simplified model, for the true noise model in Constantinides et al. [2002] a signal with n fan-outs has $n!$ possible orderings of those fan-out branches. This factorial complexity in n prevents finding the optimal ordering of word-lengths in a fan-out for all but very small values of n.

In conclusion, the proposed noise model will give less accurate estimates of truncation noise in high fan-out signals than the model in Constantinides et al. [2002] but allows the optimum configuration of word-lengths for multiple fan-out signals to be found under the proposed noise injection scheme.

4.2.3 *Preventing zero-padding of signals.* Where a signal is truncated from its original word-length n_o to the word-length n_t, the resulting noise variance σ at the system output is calculated using (10), where s is the system output's sensitivity to noise injected into the signal and p is the signal's scaling.

$$\sigma = \begin{cases} \frac{1}{12} s 2^{2p}(2^{-2n_t} - 2^{-2n_o t}) & \text{when} \quad n_o > n_t \\ 0 & \text{when} \quad n_o \leq n_t \end{cases} \tag{10}$$

Note that if n_t becomes larger than n_o then the signal is no longer being truncated but is instead having zero-valued bits appended to the LSB. However these zero-valued bits do not represent any useful information and no noise is introduced as a result of their truncation. As a result truncating a signal that is computed from one or more signals that contain zero-padded bits will cause less noise to be propagated to the system output than is estimated by (10), as n_o would not discriminate between bits that contain information and zero-padded bits, which do not. Rather than attempting to account for situations where this arises it is more prudent to constrain the word-length selections that can be made so that zero-valued bits are never appended to any signal in the system.

The constraints required to prevent zero-padding of signals are easily formulated for each type of component, as shown in (11-14). In (11-14) p_n and wl_n represent the scaling and word-length of the nth input signal to the component, and p_{name} represents the scaling of the output signal of the component called *name*. These constraints are summarized as follows. The output word-length wl_{add} of an adder must have fewer LSB bits than the input signal that has the most LSB bits (11). The output word-length wl_{mult} of a multiplier must have fewer bits (after scaling) than the sum of the word-lengths of its two inputs (12). The output word-length wl_{input} of an input to the system must be less than a user chosen limit I_{wl} (12). Finally the word-length wl_i of each branch i of a multiple fan-out signal with a set of B branches must be shorter than the original word-length wl_{fo} of the signal (14).

$$wl_{\text{add}} - p_{\text{add}} \leq \max(wl_1, wl_2) - \max(p_1, p_2) \tag{11}$$

$$wl_{\text{mult}} \leq wl_1 + wl_2 - p_1 - p_2 + p_{\text{mult}} \tag{12}$$

$$wl_{\text{input}} \leq I_{\text{wl}} \tag{13}$$

$$wl_i \leq wl_{\text{fo}} \quad \forall i \in B \tag{14}$$

These inequality constraints are all linear except for (11), which uses one max operator for $\max(wl_1, wl_2)$, ($\max(p_1, p_2)$ can be determined before optimization as the scalings p_1, p_2 are fixed values). Unfortunately this max operator makes the constraint in (11) non-convex. This non-convexity can be avoided by constraining the word-length of one input to an adder to be larger than the word-length of the adder's other input, allowing (11) to be simplified to:

$$wl_{\text{add}} - p_{\text{add}} \leq wl_l - \max(p_1, p_2) \tag{15}$$

where wl_l is the word-length of the adder input that is constrained to be larger than the other adder input. Of an adder's two inputs the one whose word-length should be longest is easily decided as the output of an algorithm is equally sensitive to truncation noise at either input, so the input with the greatest effect on power consumption should be truncated most to achieve minimum power.

One maximum word-length constraint is required for each word-length in a system (and each adder in a system requires an additional constraint to ensure one of its input word-lengths is larger than the other), and is included as one of the inequality constraints in $\mathbf{h}(\mathbf{x}) \leq 0$ from (9).

4.2.4 *Inter-component routing power convexity.* As described in Section 3.2, the power consumed in the inter-component routing wires in a system is estimated by:

—using simple estimates of net fan-out and a linear program to estimate the placement of components in order to calculate routing wire capacitance values, and,

—using word-level signal statistics to estimate the activity in each signal.

Wire-length values extracted from the 'rough placement' of a system are not necessarily convex in relation to word-length, as changing the word-length of a signal will affect the size of the component driven by that signal which in turn can cause the optimal component placement to change so significantly as to cause a discontinuity in wire-length values as word-lengths change.

Additionally, although 'rough placement' is many orders of magnitude faster than true placement, it is still slow (of the order of 0.5 seconds per placement of a system [Clarke et al. 2007]) compared to the rest of the models used to estimate power consumption. Also whilst it is possible to derive expressions for the derivatives of all other parts of the power model with respect to word-length, the only option available for the wire-length values is to approximate their derivatives by using finite differences, due to the LP formulation.

This means that for a system with n word-lengths, $n + 1$ evaluations of the LP formulation for placement estimation are required to calculate the derivatives of wire-length with respect to word-length at *each* iteration of SQP. This becomes computationally expensive for larger systems.

In order to circumvent these two problems, power consumption optimization is run in two phases, where each phase involves the solution of an optimization problem using SQP. During the first phase only the convex functions that affect power consumption are optimized while an initial estimate of the non-convex wire-length values is used that remains constant throughout this first phase. To allow the first SQP formulation to accurately balance the cost of the power consumed within arithmetic components against that consumed in the (constant wire-length) routing wires, a starting point that is expected to be 'close' to the power-optimal point should be chosen. Thus the area-optimal word-lengths are used as the starting point for word-length optimization for power consumption minimization.

Once the optimal solution to the first phase found it is used as the starting point for the second phase in which new wire-length estimates are used at each SQP iteration in conjunction with the other information provided by the power model to estimate power consumption. It is hoped that the second phase will terminate quickly despite each iteration of SQP taking longer than during the first phase, as the solution to the first phase should lie close to the solution of the second phase despite the effects of changing wire-length not being accounted for during the first phase.

In practice this method provides significant speedup and is able to find minima in the power consumption cost function that cannot be improved upon despite extensive experimentation, i.e. it would seem that the global minimum is found.

4.3 Summary of the proposed word-length optimization procedure

This section provides a summary of the steps performed in the proposed word-length optimization procedure for power consumption minimization for a system.

Before solving the two SQP formulations to minimize power consumption, it is necessary to perform range analysis to prevent overflow errors in the system, prepare the noise and power models, and to find a suitable set of word-lengths to use as a starting point for the first SQP formulation.

Simulation is used to perform range analysis and thus select the scaling of signals in this work, as described in Section 2.1.1. The sensitivity of the system's outputs to truncation of each signal within the system is then measured via simulation of the system in accordance with the perturbation analysis method of Constantinides [2003] as described in Section 2.1.1. During the above steps a Data Flow Graph (DFG) of the system is prepared and annotated with the scalings and sensitivities of the signals within it. The quantization noise model from Constantinides [2003], modified to only account for correlation due to shared truncation of bits between branches of a fan-out node as described in Section 4.2, uses this annotated DFG to estimate the noise injected into a system given a set of word-lengths during optimization using the SQP formulations.

The power consumption models require the lag-1 autocorrelation and standard deviation of each signal in the system to predict the activity within the system's signals and arithmetic components. These statistics are measured during the simulation performed during range analysis, and the DFG of the system is annotated with this information. During the optimization of the second SQP formulation the power consumed by the inter-routing wires in a system due to wire-length is estimated via the 'rough placement' algorithm which requires the area occupied by the components within a system to be estimated. These area estimates can be made by simple area models used for word-length optimization for area minimization, such as those described in Constantinides [2003] or Clarke [2008].

Finally, as described in the preceding section a suitable set of word-lengths that provide the starting point for the first SQP formulation must be found. The non integer word-lengths found by minimizing the area of a system using the proposed SQP word-length optimization technique are used for this purpose. A suitable starting point for the area optimization can first be found by performing uniform word-length optimization using the proposed SQP word-length optimization technique.

Once the two SQP formulations have been solved as described in Section 4.2, the non integer word-lengths that minimize power consumption are known. This solution provides a lower bound to the power consumed by the equivalent integer word-length problem. The upper bound power consumed by the integer problem is found by rounding up each of the word-lengths of the lower bound solution to the nearest integer. Finally, the upper bound word-lengths can be used as the starting point for a heuristic for integer word-length optimization, in order to find a solution closer in power consumption to the known lower bound. In this work the greedy heuristic from Constantinides [2006] is used for this purpose.

5. RESULTS

This section presents results obtained from running the word-length optimization techniques described in the preceding sections on a set of benchmark circuits in order to establish the range of advantages that can be provided by minimizing different word-length optimization cost functions. The objectives of the word-length optimization procedures compared in the results which follow are listed below.

Minimize uniform word-length, using individual scalings.

Minimize area as estimated by simple area models for arithmetic components based on their input word-length [Clarke 2008].

Minimize dynamic power as estimated by the power models in Section 3.

SQP is used to find non integer solutions to each of these word-length optimization problems. Note that all signal scalings and noise sensitivities are established once only by perturbation analysis [Constantinides 2003], before any optimizations are executed.

Table I shows the benchmark circuits used in the results presented in this section. In Figures 3 and Figures 4-9 which follow the 'short names' of test systems from Table I are used. Table I also shows the estimated area of each circuit in SLICEs and the circuit's estimated dynamic power consumption when using the minimum integer uniform word-lengths that achieve an output noise variance of at least 10^{-3}. These dynamic power consumption values were obtained at a clock frequency of 30MHz from Xilinx XPower estimates [Xilinx Inc. 2007].

The circuit types listed in Table I are explained below.

FIR. direct form transposed Finite Impulse Response filter.

PE. Polynomial Evaluator, polynomial order indicated in Table I.

LMS. Least Mean Squared adaptive filter.

IIR. direct form II transposed Infinite Impulse Response filter.

IIR SOS. direct form II transposed Infinite Impulse Response filter built in Second Order Sections, the number of which is shown in the 'order' column in Table I.

All circuits are pipelined to the extent that the output of every arithmetic component in the circuit is registered. The *IIR*, *IIR SOS* and *LMS* circuits must be implemented as multi-channel filters to allow this level of pipelining. All area and dynamic power consumption models have been characterized for the Virtex 2 Pro family of Xilinx FPGAs.

Table I. Benchmark Circuits

Short Name	Type	Order	Area (SLICEs)	Power (mW)
F7l	FIR, low-pass	7	100	1.4
F7h	FIR, high-pass	7	100	1.6
F15l	FIR, low-pass	15	194	2.8
F15h	FIR, high-pass	15	190	3.0
F31l	FIR, low-pass	31	383	5.5
F31h	FIR, high-pass	31	373	5.9
PE	Polynomial evaluator	7	162	2.9
L2	Adaptive LMS	2	181	4.1
L4	Adaptive LMS	4	538	12.0
L8	Adaptive LMS	8	1152	25.0
I2l	IIR ladder, low-pass	2	158	3.6
I2h	IIR ladder, high-pass	2	139	3.0
I3l	IIR ladder, low-pass	3	239	5.8
I3h	IIR ladder, high-pass	3	196	4.3
S2l	IIR, SOS, low-pass	2	243	5.8
S2h	IIR, SOS, high-pass	2	183	3.7
S3l	IIR, SOS, low-pass	3	364	8.7
S3h	IIR, SOS, high-pass	3	295	6.2

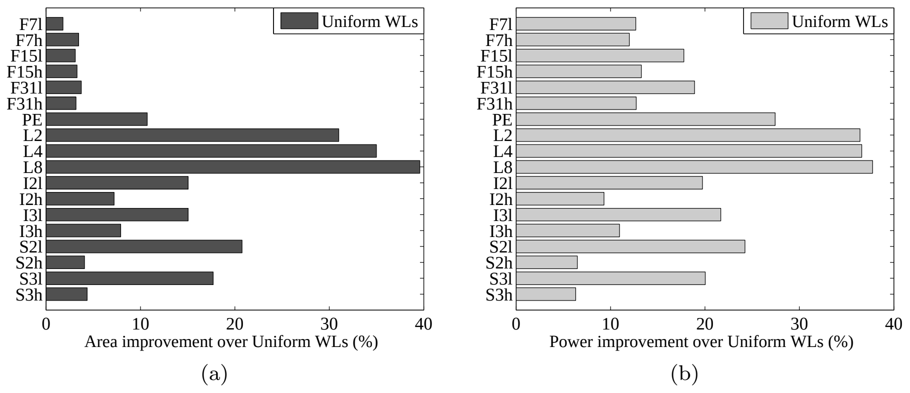

Fig. 3. (a) The increase in area when using the optimum uniform word-lengths instead of the minimum area word-lengths. (b) The increase in dynamic power consumption when using the optimum uniform word-lengths instead of the minimum power word-lengths.

5.1 Optimal non integer improvements in area and power consumption

In the following subsections the improvements in area and power consumption offered by non integer word-length optimization for area and power minimization respectively will be presented. All circuits were optimized to achieve an output noise variance (due to truncation within the circuit) of 10^{-3}.

5.1.1 *Area and power improvements over uniform word-length optimization.* Figure 3 shows the improvements compared to uniform word-length optimization offered by area optimization (a), and power consumption (b), in terms of area and power consumption in (a) and (b) respectively.

In Figure 3(a) a range of differences of up to 40% improvement in area over uniform word-length optimization can be seen, with the largest differences in area between area optimization and uniform word-length optimization obtained for the LMS adaptive filter systems and for the low-pass IIR and IIR SOS systems.

The large area improvement achieved for these systems is mainly due to large differences in the sensitivities of the outputs of these circuits to truncation noise introduced to different signals in the circuits. In the LMS adaptive filters for example, truncation at the inputs of the multipliers and accumulators used for the adjustment of coefficient values causes noise at the output of the system that is several orders magnitude larger than truncation elsewhere in the circuit.

By its nature uniform word-length optimization cannot choose to perform less truncation for these signals and thus can only make a small number of word-length reductions before noise constraints are broken. Individual word-length optimization however can choose to only slightly truncate very noise-sensitive signals whilst achieving improvements by using more truncation for less noise-sensitive signals.

In Figure 3(b) we can see that power consumption optimization also shows significant benefits over uniform word-length optimization, and in general larger improvements are possible for power over uniform word-lengths than for area over uniform word-lengths, particularly for the FIR and PE systems.

The differences between uniform word-length optimization and power consumption minimization are due to the same reasons cited for area optimization. Larger power improvements seen for the FIR and PE systems in particular are due to the less noise-sensitive signals in these circuits having a greater effect on power consumption than on area. These signals are the inputs to the embedded multipliers in these circuits, where during area optimization only one embedded multiplier per multiplication is required, whilst during power optimization the power consumed by the input signals to these multipliers is still affected by word-length.

5.1.2 *Comparison of minimum area and minimum power optimizations.* The results in Figure 4 show: the difference between the area of power optimal circuits and area optimal circuits in (a), and the difference between the power of area optimal circuits and power optimal circuits in (b). Area optimization achieves an improvement of only 0.52% on average over the area of power optimization. Power consumption optimization achieves a range of improvements between 1% and 15% over the power consumption of area-optimal circuits.

The larger improvements in power consumption achieved for the FIR, PE and LMS circuits are mainly due to area optimization not truncating the inputs to embedded multipliers in these systems, as only one embedded block is required per multiplication to achieve noise constraints. As a result the power consumption of the signals driving these multipliers is high at the area optimal point. Power consumption optimization is also able to make reductions in power compared to that achieved during area optimization by more heavily truncating the word-lengths of inter-routing wires with a high capacitance (due to high fan-out or circuit topology).

However the gap between the area of power optimal systems and that of area optimal systems is small as shown in Figure 4(a). To achieve minimum area during area optimization the inputs to the adders in each circuit are heavily truncated, but the inputs to multipliers are not as doing so has no effect on their area. Be-

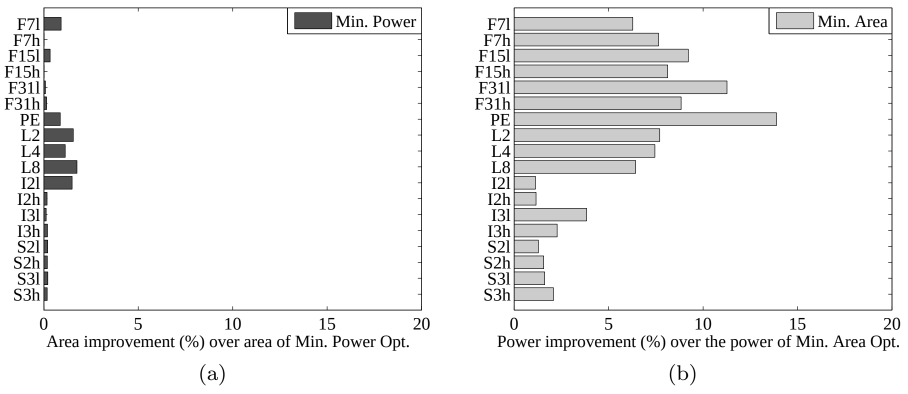

Fig. 4. (a) The increase in area when using the minimum power word-lengths instead of the minimum area word-lengths. (b) The increase in dynamic power consumption when using the minimum area word-lengths instead of the minimum power word-lengths.

cause there is an exponential relationship between quantization noise variance and signal truncation (as shown in (3)) power optimization needs to only slightly reduce the high level of truncation used for adders in order to be able to use significant truncation for signals with high power consumption in order to achieve the same output noise variance as in area optimization. The slight reduction in the level of truncation used for adder inputs means that the area of power optimal systems is only slightly worse than that of area optimal systems.

Finally, the IIR and IIR SOS systems do not show significant power consumption improvements over the power achieved by area optimization, because whilst power consumption optimization once again improves over area optimization by more heavily truncating the inputs to multipliers in these circuits, these gains are balanced out by the cost of increasing the word-lengths of the adders in the circuit as these have pipeline registers in between them to allow three channels of data to be processed which consume extra area.

5.2 Integer problem lower and upper bounds

Figures 5 and 6 show the overheads in area (Figure 5) and power consumption (Figure 6) when using the upper-bounds on the costs of the integer versions of the word-length optimization problems for area and power consumption minimization, compared to the lower bound area and power consumption for these word-length optimization problems. Lower bounds are obtained from the optimal non integer word-lengths found for each circuit, whilst upper bounds are found by rounding these optimal non integer word-lengths up to the nearest integers, as described in Section 4.2.1. Also shown are the overheads in terms of area and power consumption when using the integer word-length result found by the the proposed combination of integer upper-bound and heuristic summarized in Section 4.2.1, compared to the integer word-length lower-bound.

The average gap in area between the lower and upper-bounds is 2.9%, whilst the average gap in area between the lower-bound and the heuristic result is 1.4%. For power optimization the average gap between the lower and upper-bounds is 5.1%,

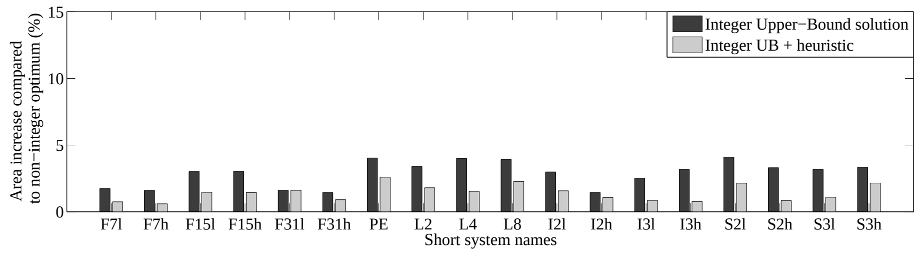

Fig. 5. The percentage increase in area given by the integer word-length upper-bound and the proposed combination of integer upper-bound and heuristic, relative to the lower-bound given by the minimum area non integer word-lengths.

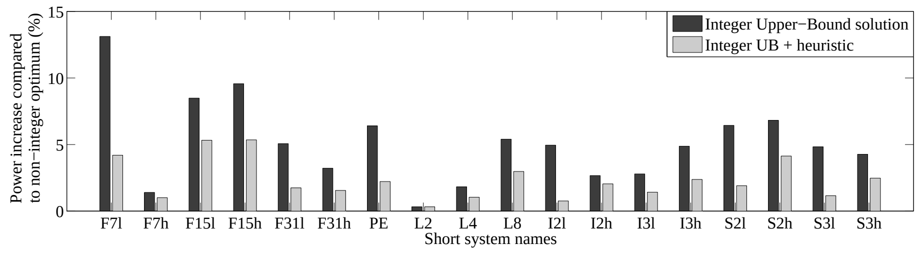

Fig. 6. The percentage increase in power consumption given by the integer word-length upper-bound and the proposed combination of integer upper-bound and heuristic, relative to the lower-bound given by the minimum power non integer word-lengths.

whilst the average gap between the lower-bound and the proposed combination of upper bound and heuristic is 2.3%. Clearly tight lower and upper-bounds on the integer word-length optimization problems for area and power optimization can be achieved using our method, and the simple integer word-length selection heuristic can be used to improve on the integer upper-bound found and obtain cost values close to the lower-bounds found.

5.3 Comparison of integer word-length optimization techniques

Figure 7 shows the overheads in power consumption when using either the minimum uniform integer word-lengths or the heuristic from Constantinides [2006] alone, compared to the power consumption given by the lower-bound on the cost of the integer problem. Also included in Figure 7 are the overheads when using upper-bounds on the costs of the integer versions of the word-length optimization problems for power consumption minimization and the proposed combination of integer upper-bound and heuristic that were shown in Figure 6.

The minimum uniform integer word-lengths are found by rounding up the minimum non-integer word-lengths to the nearest integer, whilst the heuristic from Constantinides [2006] performs bit-by-bit word-length reduction for each signal in a system as summarised in Section 4.2.1, starting from the system with uniform word-length $2 \times wl_U$, where wl_U is the minimum uniform integer word-length.

In Figure 7 it can be seen that the power consumption offered by using the minimum uniform integer word-length, obtained by rounding up the minimum non-

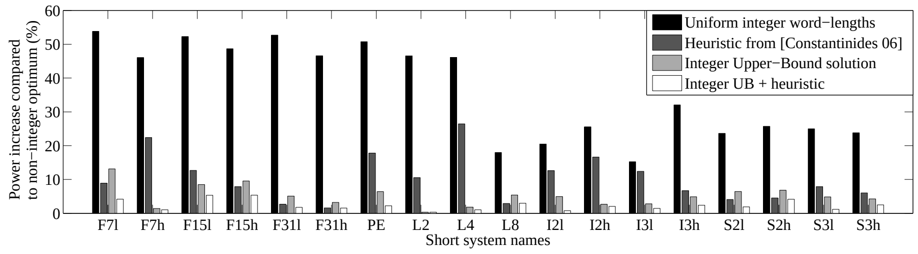

Fig. 7. The percentage increase in power consumption given by the uniform integer word-length solution, the solution found by the heuristic in [Constantinides 2006], the integer upper-bound, and the proposed combination of integer upper-bound and heuristic, relative to the lower-bound given by the minimum power non integer word-lengths.

integer word-length, is higher by 20 to 50% compared to the proposed combination of integer upper-bound and heuristic.

More interestingly, although the heuristic used in Constantinides [2006] is able to approach the power consumption found by the proposed technique in some cases (such as for the test circuits F15h, F31l, F31h and L8), it often gets caught in a local optimum before it is able to approach the power consumption solution found by the proposed method, resulting in power consumption levels that can be higher by over 20% (as is seen for the test circuits F7h and L4). In fact for several circuits the heuristic is unable to find a solution that improves upon the power consumption of the word-length optimization upper-bound found by the proposed solution.

5.4 Run times

Figure 8 shows the amount of time in minutes to run the non integer word-length optimization procedures on a 3GHz Pentium 4 computer. All optimization procedures and supporting code are implemented entirely in Matlab, hence significant speed-up of these results could be achieved with a C implementation, for example.

Uniform word-length optimization is very fast and completes in 0.7 seconds on average. Area optimization takes 39 seconds on average, with the longest time taken to complete at around 5 minutes, for the larger FIR test system.

Power consumption optimization shows a range of computation times ranging from around 30 seconds to up to almost 40 minutes for the largest systems. Power consumption optimization takes significantly longer to execute because partial derivatives of power are not currently calculated, though they could be for logic power, inter-routing wire activity and fan-outs. Instead partial derivatives are estimated at each SQP iteration using finite differences of the power cost function.

Figure 9 shows the amount of time in minutes required to run the heuristic from Constantinides [2006] in order to optimize the power consumption of each system, starting from the uniform word-length $2 \times wl_U$, where wl_U is the minimum uniform integer word-length. The power consumption results achieved by running this heuristic alone in this way were shown in Figure 7 and showed that the proposed combination of integer upper-bound and heuristic could achieve significantly lower power consumption for several systems.

Figure 9 shows that using the heuristic from Constantinides [2006] can be very

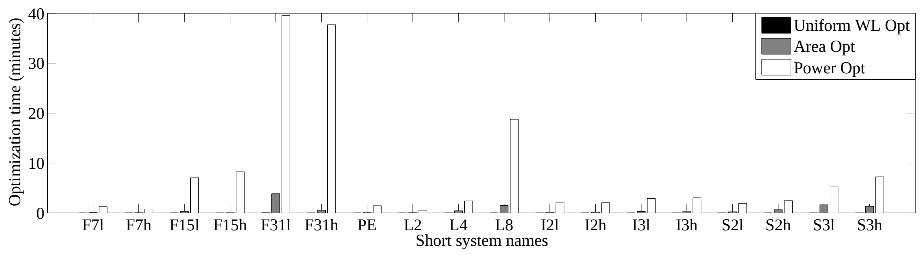

Fig. 8. The time required to run the non integer word-length optimization procedures.

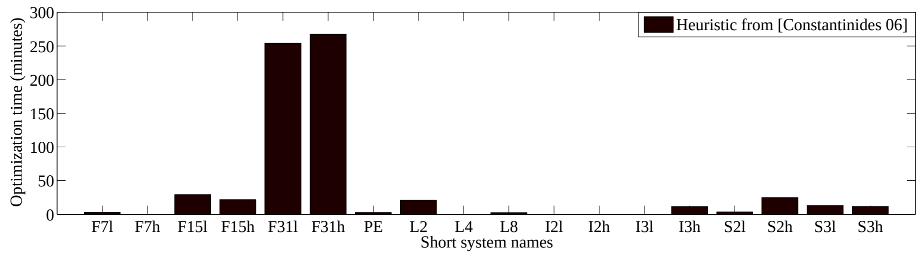

Fig. 9. The time required to run the integer word-length optimization heuristic from Constantinides [2006].

time consuming, taking over 250 minutes for the largest circuits. The long runtime of the heuristic for these circuits can be explained by the fact that it performs bit-by-bit reduction in word-lengths, but evaluates the reductions offered by every signal in the system before reducing the word-length of one signal by one bit.

It can be seen that for some circuits the heuristic's runtime is very short, but it should be noted that these are circuits where the heuristic became caught in local optima long before reaching the levels of power consumption achieved by the proposed method.

Hence the results in Figures 7, 8 and 9 indicate that the proposed method is able to find solutions to word-length optimization problems whose cost is closer to known lower bounds than the heuristic proposed in Constantinides [2006], and is able to do so faster than the heuristic.

5.5 Power optimization improvements in un-pipelined circuits

In this section we present power consumption minimization results for single channel, un-pipelined versions of the IIR, IIR SOS and LMS filters, and the PE circuit. Fast power consumption models are not available for adders and multipliers whose inputs are not registered (and hence contain glitches), so the power P_u in an unregistered component is approximated by multiplying the component's power P_r when inputs *are* registered (estimated by our existing models) by a factor dependent on the number of un-registered components preceding it:

$$P_u = P_r 2^D \tag{16}$$

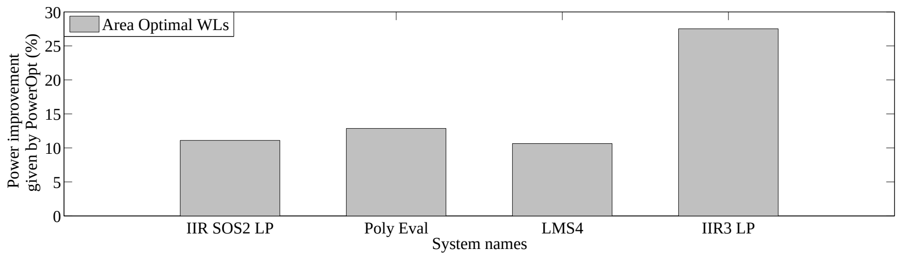

Fig. 10. The non integer word-length optimization for power minimization improvements achieved compared to the optimizations shown when un-registered components are modelled.

where D is the number of un-registered components preceding the component. Extra power due to glitches in the inter-routing wires after the component is estimated by multiplication by the same factor 2^D.

Figure 10 shows the improvements in power obtained by non integer word-length optimization for power minimization compared to area minimization. Compared to the results that contrasted the minimum power and minimum area word-length optimization methods in Section 5.1.2 we see that the much larger range of component power consumptions has given rise to a much larger gap between the power of the two optimizations, particularly for the *IIR 3 LP* and *SOS 2 LP* circuits whose pipelined versions showed a gap of less than 5% between the power of the two optimizations in Section 5.1.2. Here we see a difference of greater than 10% for the four circuits, with improvements of almost 30% for the *IIR 3 LP* circuit.

These results indicate that in cases where not every output of an arithmetic component in a circuit can be pipelined, word-length optimization for power consumption minimization may be able to offer large improvements in power consumption over other available techniques.

6. CONCLUSION

This paper has described a novel technique for finding optimal or near-optimal solutions to the problem of integer word-length selection for power consumption optimization. The first complete set of dynamic power consumption models suitable for quickly evaluating points in the design space have been described and used to provide the first set of results for word-length optimization for power consumption minimization. Results show that for a specific set of noise constraints area and power consumption can be improved by up to 40% over uniform word-length optimization when integer constraints are relaxed.

Tight upper and lower bounds on the costs obtainable are obtained by solving a relaxed version of the word-length optimization problem where integer constraints are removed. The upper bounds for area and power consumption are 2.9% and 5.1% larger than the lower bounds on average, respectively. Heuristics can then improve on the upper bounds giving integer word-length solutions whose area and power is 1.4% and 2.3% larger than the lower bounds on average, respectively.

Results have been presented that show that the proposed word-length optimization technique for power consumption minimization is able to find solutions whose power consumption is closer to the known lower bound than the heuristic proposed in Constantinides [2006], and have also indicated that the proposed method is able to find such solutions in less time than the heuristic.

For algorithm implementations where component costs in terms of area and power are of a similar order it has been seen that the word-length optimization of either of these two costs obtains similar system area and power. However when the cost of components in a system differ in orders of magnitude between cost functions differences in power consumption between the optimal points of the different cost functions of up to 30% are observed. The phenomenon that cause such differences between the area and power optimal points of the circuits studied are: *i)* the higher power consumption of certain inter-routing nets due to high fan-out or circuit topology, *ii)* the power consumed in routing to embedded multipliers whose cost is fixed given a particular noise constraint during area optimization, and *iii)* the higher power consumption of components whose inputs contain glitches as they are not registered.

The results of this paper suggest that future work in the field of word-length optimization should be particularly focussed on the following:

—The identification of computations and algorithms whose component costs differ by orders of magnitude and hence require new cost models. Power models for components whose inputs are not registered are an ideal candidate.

—The development of heuristics that take advantage of the known lower bound to the cost of a word-length optimization problem.

—The continued development of fast methods for estimating algorithm accuracy in systems for which this is not currently possible.

REFERENCES

Anderson, J. H. and Najm, F. N. 2004. Power estimation techniques for FPGAs. *IEEE Trans. on VLSI Systems 12,* 10, 1015–1027.

Benedetti, A. and Perona, P. 2000. Bit-width optimization for configurable DSPs by multi-interval analysis. In *Proc. Asilomar Conf. Sig. Sys. and Comp.* Vol. 1. 355–359.

Bhoj, S. and Bhatia, D. 2007. Pre-route interconnect capacitance and power estimation in FPGAs. *Proc. Int. Conf. Field Programmable Logic and Applications*, 159–164.

Bobba, S., Hajj, I. N., and Shanbhag, N. R. 1999. Analytical expressions for power dissipation of macro-blocks in dsp architectures. *Proc. Int. Conf. VLSI Design*, 358–363.

Boggs, P. T. 1995. Sequential quadratic programming. *Acta Numerica 4*, 1–52.

Chan, S. and Tsui, K. 2006. The wordlength determination problem of linear time invariant systems with multiple outputs - a geometric programming approach. *Proc. IEEE Int. Symp. Circuits and Systems*, 5211–5214.

Choy, N. C. K. and Wilton, S. J. E. 2006. Activity-based power estimation and characterization of DSP and multiplier blocks in FPGAs. In *Proc. IEEE Int. Conf. Field Programmable Technology.* 253–256.

Clarke, J. A. 2008. High-level power optimization for digital signal processing in reconfigurable logic. Ph.D. thesis, Department of Electrical and Electronic Engineering, Imperial College London.

Clarke, J. A., Constantinides, G. A., and Cheung, P. Y. K. 2007. On the feasibility of early routing capacitance estimation for FPGAs. In *Proc. Int. Conf. Field-Programmable Logic and Applications.* 234–239.

Clarke, J. A., Gaffar, A. A., and Constantinides, G. A. 2005. Parameterized logic power consumption models for FPGA-based arithmetic. In *Proc. Int. Conf. Field-Programmable Logic and Applications.* 626–629.

Clarke, J. A., Gaffar, A. A., Constantinides, G. A., and Cheung, P. Y. K. 2006. Fast word-level power models for synthesis of FPGA-based arithmetic. In *Proc. IEEE Int. Symp. Circuits and Systems.* 1299–1302.

Clarke, J. A., Gaffar, A. A., Constantinides, G. A., Cheung, P. Y. K., and Smith, A. M. 2008. Glitch-aware output switching activity from word-level statistics. In *Proc. IEEE Int. Symp. Circuits and Systems.* 1792–1795.

Constantinides, G. A. 2003. Perturbation analysis for word-length optimization. In *Proc. IEEE Symp. Field-Programmable Custom Computing Machines.* 81–90.

Constantinides, G. A. 2006. Word-length optimization for differentiable nonlinear systems. *ACM Trans. Des. Autom. Electron. Syst. 11,* 1, 26–43.

Constantinides, G. A., Cheung, P. Y. K., and Luk, W. 1999. Truncation noise in fixed-point sfgs [digital filters]. *IEE Electronics Letters 35,* 23, 2012–2014.

Constantinides, G. A., Cheung, P. Y. K., and Luk, W. 2001. The multiple wordlength paradigm. In *Proc. IEEE Symp. Field-Programmable Custom Computing Machines.* 51–60.

Constantinides, G. A., Cheung, P. Y. K., and Luk, W. 2002. Optimum wordlength allocation. In *Proc. IEEE Symp. Field-Programmable Custom Computing Machines.* 219–228.

Constantinides, G. A. and Woeginger, G. J. 2002. The complexity of multiple wordlength assignment. *Applied Mathematics Letters 15,* 2, 137–140.

Doi, N., Horiyama, T., Nakanishi, M., and Kimura, S. 2006. Bit-length optimization method for high-level synthesis based on non-linear programming technique. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci. E89-A,* 12, 3427–3434.

Gaffar, A. A., Mencer, O., Luk, W., and Cheung, P. Y. K. 2004. Unifying bit-width optimisation for fixed-point and floating-point designs. *Proc. IEEE Symp. Field-Programmable Custom Computing Machines*, 79–88.

Jiang, T., Tang, X., and Banerjee, P. 2004. Macro-models for high level area and power estimation on FPGAs. In *Proc. ACM Great Lakes Symp. VLSI.* 162–165.

Kim, S., Kum, K.-I., and Sung, W. 1998. Fixed-point optimization utility for C and C++ based digital signal processing programs. *IEEE Trans. Circuits and Sys. II: Analog and DSP 45,* 11, 1455–1464.

Knuth, D. E. 1997. *The art of computer programming.* Vol. 3: Sorting and Searching. Addison-Welsey.

Landman, P. E. and Rabaey, J. M. 1996. Activity-sensitive architectural power analysis: the dual bit type method. *IEEE Trans. CAD of Integrated Circuits and Systems 15,* 6, 571–587.

Lee, D.-U., Gaffar, A. A., Cheung, R. C. C., Mencer, O., Luk, W., and Constantinides, G. A. 2006. Accuracy-guaranteed bit-width optimization. *IEEE Trans. CAD of ICs and Sys. 25,* 10, 1990–2000.

Ozer, E., Nisbet, A. P., and Gregg, D. 2004. Stochastic bit-width approximation using extreme value theory for customizable processors. In *Proc. Int. Conf. Compiler Construction.* 250–264.

Poon, K. K. W., Yan, A., and Wilton, S. J. E. 2002. A flexible power model for FPGAs. In *Proc. Int. Conf. Field Programmable Logic and Applications.* 312–321.

Roy, K. and Prasad, S. 2000. *Low-power CMOS VLSI circuit design.* Wiley-Interscience.

Shang, L. and Jha, N. K. 2001. High-level power modeling of CPLDs and FPGAs. In *Proc. of the Int. Conf. Comp. Design.* IEEE Computer Society, 46–53.

Sung, W. and Kum, K.-I. 1995. Simulation-based word-length optimization method for fixed-point digital signal processing systems. *IEEE Trans. Sig. Proc. 43,* 12, 3087–3090.

Wadekar, S. A. and Parker, A. C. 1998. Accuracy sensitive word-length selection for algorithm optimization. In *Proc. Int. Conf. Comp. Design.* 54–61.

Xilinx Inc. 2004. *Virtex-2 Pro and Virtex-2 Pro X Platform FPGAs: Complete Data Sheet.* San Jose.

Xilinx Inc. 2007. *Xilinx power estimator user guide.* San Jose.