Power Optimization of Parallel Multipliers in Systems with Variable Word-length

Saeeid Tahmasbi Oskuii, Per Gunnar Kjeldsberg, Lars Lundheim, Asghar Havashki

Department of Electronics and Telecommunications,
Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
email: {saeeid, pgk, lundheim, havashki}@iet.ntnu.no

Abstract. Parallel multipliers can be optimized using the intrinsic arithmetic equivalencies in their reduction-tree. In this paper, we propose a method to reduce the dynamic power consumption in parallel multipliers, operating within systems with effective word-length variation. Word-length variation induces a certain pattern of spatiotemporal correlations. The proposed method is capable to take such correlations into account resulting better solutions. The experimental results show about 16-21% reduction in the average number of transitions compared to random parallel multipliers.

1 Introduction

A digital signal processing (DSP) system can be optimized for the operation conditions, if these conditions are known in the design phase. In a DSP system operating with a nondeterministic data set, these operation conditions can involve probabilistic measures like the distribution of the input data over time and their correlations. For such systems it can in addition be advantageous to dynamically adapt the computation algorithm based on run-time knowledge about the operating condition and the current states of the inputs. Such adaptation of, e.g., the datapath word-length and other system-level parameters, can significantly improve the computation efficiency, power consumption, robustness or other merits of the system. Adaptive modulation, adaptive power control and tunable word-length are among the adaptation methods that have been proposed in the past.

Adaptive word-length variation for example performs a trade off between performance, power consumption and quality of computations. Word-length variation affects the quantization error introduced when fixed-point operators are utilized instead of ideally infinite accuracy. It is therefore the acceptable threshold of the quantization error that specifies the minimum number of bits in a DSP system. In many systems this acceptable threshold can vary over time, based on, e.g., user preferences, operating conditions such as communication channel noise, and the application that is currently being executed. It is then advantageous to be able to vary the word-length dynamically. The design-time word-length optimization methods trade off area, speed and signal quality, while dynamic word-length variation enables a trade off between power consumption and signal quality.

Dynamic word-length variation does not necessitate dedicated hardware resources for different word-lengths. The system can keep operating with its hardware resources, but use less number of bits; i.e., fix a number of bits from the least significant bit (LSB) side to zero (or one) and force no transition on these bits. As will be shown in experiments later in this paper, the average number of transitions for a multiplier drops approximately quadratically with the number of inactive bits in the input operands.

The amount of saving is less than the case where there is dedicated hardware for variable word-length multiplication. However, the dedicated hardware requires additional circuitry to switch between multipliers.

With evermore important stress on lengthening the battery lifetime in portable devices, more and more focus has been given to the power consumption in such systems. Examples of dynamic word-length control can be found in [10, 9, 18] for parts of communication systems. [3] introduces dynamic word-length variation in a 3D graphic texture mapping context. In these systems multipliers are among the main power consuming parts, and have consequently been given great attention in this respect.

In this paper, we propose a method to reduce the overall dynamic power by optimizing the reduction-tree of parallel multipliers in design-time for systems with varying word-length. In addition to systems that utilize word-length adaptation techniques for power saving, the proposed method is applicable for general-purpose processors where the word-length is defined by the program that at any given time is being executed. Through profiling of the target programs for a general-purpose processor, the power consumption of build-in multipliers can by reduced using our method.

2 Parallel Multipliers

As discussed in the previous section, the target application for multipliers with variable word-length are DSP systems and general-purpose processors. For both application areas the multiplier is often placed in the critical path and is therefore speed-limiting. Thus, we have chosen to focus on parallel multipliers, which offer the best performance compared to other multiplier types. Parallel multipliers have two basic steps of computation: partial product (PP) generation and PP accumulation. The first step generates PPs in parallel. The resulting partial product bits with different weights are accumulated using a multi-operand adder tree in the second step. The summation output from the multi-operand adder is indeed the multiplication product. The multi-operand adder reduces the number of PPs in several stages using compressor units such as full-adders and half-adders. Both computational steps, i.e. PP generation and PP accumulation, have been studied for many years resulting in various methods to improve the performance. These are surveyed in [4, 12]. Multipliers and multi-operand adders can be subject to any standard combinational logic optimization. Such circuits are, however, very difficult to optimize because of their large size and due to a prevalence of exclusive-OR operations in their logic relations. Therefore, considering the arithmetic relations in the structure of multipliers and multi-operand adders can be extremely useful and can result in larger savings. Arithmetic properties, such as commutativity, associativity and retiming, entail a large freedom in the structure of multi-operand adders. This freedom can be used to optimize area, delay, dynamic power or other parameters [7, 13, 11, 19].

In [17], the authors propose a progressive reduction-tree design algorithm for reducing dynamic power in the full-adder based PP reduction-tree by searching for low power solutions among the functionally equivalent implementations of the reduction-tree. PPs with equal weight can be interchanged in the reduction tree because of the commutativity and associativity properties of addition. [17] exploits the large freedom of the interconnection order of PPs that is intrinsic in the reduction-tree. The optimization algorithm is summarized in Figure 1. The progressive reduction tree design algorithm combines the construction phase with the optimization phase. The search is localized to one stage of full-adder/half-adder at a time. In order to

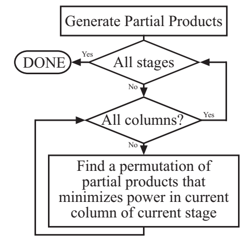
Generate Partial Products
DONE
Yes
All stages
No
All columns?
Yes
No
Find a permutation of partial products that minimizes power in current column of current stage

Fig. 1. The simplified flowchart for the optimization algorithm

find the best solution for this stage, estimation of dynamic power is required for each solution. The Simple waveform set (SWS) method introduced in [16] is a probabilistic gate-level power estimator that is utilized in the progressive reduction-tree design algorithm. Although any deterministic delay model can be used in the power estimator, for simplicity it uses a fanout-delay model for logic gates; i.e. the delay of a logic gate is assumed to be equal to the number of its fanouts. The SWS power estimator includes a mechanism for glitch filtering considerations due to inertial delay of logic gates. The interdependencies due to reconvergent-fanouts in the structure of the multiplier are taken into account using pairwise correlation coefficients between two nodes. The pairwise correlation coefficient between to nodes A and B is defined as the joint one-probabilities of A and B divided by the product of separate one-probabilities of A and B; i.e:

$$\kappa_{A,B} = \frac{p(A=1 \wedge B=1)}{p(A=1)p(B=1)} \tag{1}$$

In [5], a procedure for propagating signal probabilities from the circuit inputs toward the circuit outputs using only pairwise correlations between circuit lines and ignoring higher order correlation terms is described.

The reduction scheme for the reduction-tree and PP generation method are not restricted in the progressive reduction-tree design algorithm. However, for the reported results in Section 4, modified Baugh-Wooley [1, 6] is used as the 2's complement multiplier PP generation method. Modified Dadda/Wallace reduction scheme [2] is chosen as the reduction scheme. This reduction scheme promises minimal hardware resources and minimal output vector size.

3 Variable word-length

We will now address multiplication with variable input word-length performed on one single hardware resource that can not be changed. Let the bit-vectors X(n bits) and Y(m bits) be the input operands to an $n \times m$-bit multiplier. We denote the static probabilities of bit vectors X and Y with vectors $\mathbf{p}_X$ and $\mathbf{p}_Y$ respectively. The i:th element of the vector $\mathbf{p}_X$ represents the one-probability of the i:th bit of the input X. A multiplier that is optimized using the algorithm in Figure 1, to operate with inputs having $\mathbf{p}_X$ and $\mathbf{p}_Y$ as their static probability vectors, is denoted as $M_{\text{opt}}(\mathbf{p}_X, \mathbf{p}_Y)$. Similarly the worst-case multiplier is referred as $M_{\text{wc}}(\mathbf{p}_X, \mathbf{p}_Y)$. Interconnect orders in the worst-case multiplier are chosen so that the power consumption is maximized.

Note that even if a multiplier is optimized at design-time using two specific input static probability vectors, it may at run-time be operating under other input conditions. These conditions can also change over time.

In the context of systems with variable word-length, the X and Y inputs have l and k active bits, respectively. The $n-l$ and $m-k$ inactive bits in X and Y are assumed to be forced to zero. The active bits are assumed to have 0.5 one-probabilities. This results in uniform distributions for the random variables X and Y. It is also possible to assume non-uniform distributions for input words, e.g., a Gaussian distribution. Landman and Rabaey in [8] show that the lower bit positions in normally distributed inputs behave like completely random bits with 0.5 one-probabilities. In the scenario of the word-length variation, the random portion of the input bits are forced to zero. In this paper it is assumed that the input signals have uniform distribution. More specifically, the active input bits are uncorrelated and have 0.5 one-probabilities. For simplicity, we introduce the notation Ω_l^n which is a static probability vector of an n-bit uniform random input with l active bits. The active bits in this representation have 0.5 one-probabilities and they are temporally and spatially uncorrelated. The $n-l$ inactive bits are assigned to be zero; i.e., their one-probability is zero. For example, if the input word X has static probability vector of Ω_4^6, two least significant bits (X_1 and X_2) are zero and four most significant bits ($X_{3..6}$) have 0.5 one-probabilities. Therefore,

$$\Omega_4^6 = [\overset{\text{LSB}}{0}\ 0\ 0.5\ 0.5\ 0.5\ \overset{\text{MSB}}{0.5}] \tag{2}$$

Let $\mathcal{M} = [\mu_{\rangle,|}]_{\setminus\times\Updownarrow}$ be the word-length probability density matrix ($0 \leq \mu_{i,j} \leq 1$). $\mu_{i,j}$ is the probability that the $n \times m$ multiplier will have i and j active bits in the first and the second operands respectively; i.e. $\mu_{i,j}$ is the probability that Ω_i^n and Ω_j^m are applied to X and Y respectively. $\mathcal{M}$ is visualized in Figure 3 using radii of circles as the probability. $\mathcal{M}$ satisfies the following condition:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \mu_{i,j} = 1 \tag{3}$$

Let $i_{\max}$ and $j_{\max}$ be the row and column indexes of the largest element in the word-length probability density matrix; i.e., $\mu_{i_{\max},j_{\max}}$ is the largest element of the matrix $\mathcal{M}$. Intuitively, in systems with variable word-length, the multiplier optimization can be performed for this maximal probability. That is, if $i_{\max} \times j_{\max}$-bit multiplications are the most frequent multiplications, then the multiplier $M_{\text{opt}}(\Omega_{i_{\max}}^n, \Omega_{j_{\max}}^m)$ is expected to consume the lowest power consumption. However, as will be demonstrated in the experimental results later in this paper, this is often not the case.

From the word-length probability density matrix, $\mathcal{M}$, we obtain two vectors $\overline{\mathbf{p}_X}$ and $\overline{\mathbf{p}_Y}$ which are average static probability vectors for the first and second operand over an infinitely large time interval, respectively. The η:th element in $\overline{\mathbf{p}_X}$ is equal to:

$$\overline{\mathbf{p}_X}[\eta] = \frac{1}{2} - \frac{1}{2}\sum_{i=1}^{\eta-1}\sum_{j=1}^{m} \mu_{i,j} \tag{4}$$

For example, for the word-length probability density matrix shown in Figure 3, the one-probability for 10th bit is $\overline{\mathbf{p}_X}[10] = 0.5 - 0.5 \times (0.02 + 0.05 + 0.10 + 0.15) = 0.34$. Similarly, the ζ:th element in $\overline{\mathbf{p}_Y}$ is equal to:

$$\overline{\mathbf{p}_Y}[\zeta] = \frac{1}{2} - \frac{1}{2}\sum_{j=1}^{\zeta-1}\sum_{i=1}^{n} \mu_{i,j} \tag{5}$$

For a multiplier where $n = m$ and the probability density matrix $\mathcal{M}$ is symmetrical, we simply use the notation $\overline{\mathbf{p}}$ instead of $\overline{\mathbf{p}_X}$ and $\overline{\mathbf{p}_Y}$. We use these vectors as inputs to our optimization algorithm and the optimized multiplier is denoted with $M_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$. Our experimental results shows that this multiplier in general exhibits better performance for the system with variable word-length compared to $M_{\text{opt}}(\Omega_{i_{\max}}^n, \Omega_{j_{\max}}^m)$.

In our experiments we realistically assume that the changes in word-length occur seldom and therefore, the power consumption due to the actual word-length shift is negligible; i.e., the power consumption can be estimated by estimating the power consumption for various word-lengths separately. The total power consumption P_{tot} is the weighed sum of the estimated power numbers using the word-length probability density matrix $\mathcal{M}$.

$$P_{tot} = \sum_{i=1}^{n}\sum_{i=1}^{m} \mu_{i,j} P_{i,j} \tag{6}$$

where $P_{i,j}$ is the estimated power consumption when the $n \times m$-bit multiplier is operating with $i \times j$ active bits.

A group of bits which are forced to zero due to word-length reduction are all equal and with a high probability will not experience changes in the next clock cycle. This leads to a certain pattern of spatiotemporal correlation between the primary inputs. Figure 2(a) depicts such correlated input bits, while Figure 2(b) shows a sequence of uncorrelated inputs. Figures 2(a) and 2(b) have equal one-probabilities over an infinitely large time interval. The only difference is the spatiotemporal correlations. As will be shown from our experimental results, the power consumption caused by the two input patterns can be quite different. Hence, integrating such correlations in the power estimator and optimization will lead to better solutions compared to using the independence assumption for primary inputs. As briefly discussed in Section 2, the optimization algorithm in [17] utilizes a probabilistic gate-level power estimator [16]. This power estimator captures the spatial correlations due to reconvergent fanouts in the combinational network using pairwise correlation coefficients introduced in [5].

For uncorrelated circuit nodes, the pairwise correlation coefficients are equal to 1. The optimization algorithm in [17] assumes that the primary input bits are spatially uncorrelated. This means that the value of one bit is independent of the value of any other bit, both of the same input operand and of the other input operand. Therefore the correlation coefficients for the primary inputs are set to 1. We want to include the spatial correlations of primary inputs, which are present in our target application, in the power estimator. Consequently, we replace the default primary input correlation coefficients with pre-computed pairwise correlation coefficients in a matrix form, denoted as $\mathcal{C}$. The correlation coefficient matrix $\mathcal{C}$ is an $(n+m)\times(n+m)$ matrix. The element in row i and column j of the correlation coefficient matrix is the pairwise correlation coefficient between input nodes i and j; i.e.

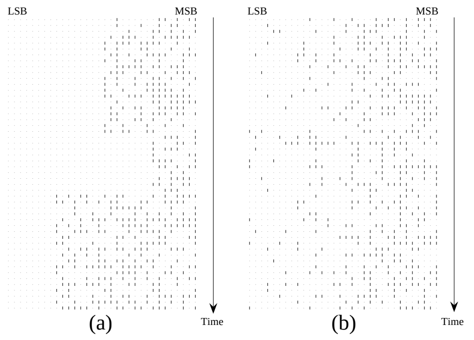

Fig. 2. (a) Correlated input bits pattern expected in the systems with word-length variation (a) Spatiotemporally uncorrelated input bits pattern

$$\mathcal{C}_{i,j} = \kappa_{I_i,I_j} \tag{7}$$

where I_i and I_j are the i:th and j:th primary input nodes respectively $(1 \leq i, j \leq n+m)$.

$$I_i = \begin{cases} X_i & \text{if } i \leq n \\ Y_{i-n} & \text{if } n < i \leq n+m \end{cases} \tag{8}$$

The matrix $\mathcal{C}$ can be approximated using a large number of random inputs with the desired pattern of word-length variation and computing the correlation coefficients using Eq. 1.

In addition to these spatial correlations, we need to include temporal correlations in the power estimation procedure as well. In fact, temporal correlations are very important and estimation of power without considering temporal correlations can be very inaccurate. The temporal correlation is modeled using lag-one temporal correlation ratio, ρ_i:

$$\rho_i = \frac{E[I_i[\eta]I_i[\eta-1]] - E[I_i[\eta]]^2}{E[I_i[\eta]^2] - E[I_i[\eta]]^2} \qquad 1 \leq i \leq n+m \tag{9}$$

where $E[\cdot]$ is the ensemble average of the random variable $\cdot$ and $I_i[\eta]$ is the i:th primary input at time instance η. The maximum magnitude of ρ_i is 1. ρ_i is 0 if I_i is temporally independent. Two methods (exact and approximative) for computing bit-level temporal correlations from word-level signal statistics are presented in [15]. Similar to the method for approximating spatial pairwise correlation coefficients, we use a large number of random inputs with the desired pattern of word-length variation for estimating the values of lag-one temporal correlation ratios.

In the power estimation procedure that is embedded in the optimization algorithm, the primary input nodes are initialized with four waveforms of holding-one, holding-zero, zero-to-one transition, and one-to-zero transition denoted as W_{11}, W_{00}, W_{01}, and W_{10}, respectively. The occurrence probabilities of such waveforms under the spatiotemporal independence assumption are set to p_i^2, $(1-p_i)^2$, $(1-p_i)p_i$ and $p_i(1-p_i)$ respectively where p_i is the static probability of the corresponding input node i. With presence of temporal correlation ρ_i, the occurrence probabilities are altered to:

$$\begin{cases} p(W_{11}) = \rho_i(p_i - p_i^2) + p_i^2 \\ p(W_{00}) = \rho_i(p_i - p_i^2) + (1-p_i)^2 \\ p(W_{01}) = (1-\rho_i)(p_i - p_i^2) \\ p(W_{10}) = (1-\rho_i)(p_i - p_i^2) \end{cases} \tag{10}$$

We denote the multiplier that is optimized for $\overline{\mathbf{p}_X}$ and $\overline{\mathbf{p}_Y}$ including the spatiotemporal correlations with $M^*_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$. Note that $M_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ and $M^*_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ refer to different multipliers, as spatiotemporal correlations are considered for the latter, while it is not considered for the former.

4 Experiments

The optimization algorithm, the power estimator and a VHDL generator for the designed multipliers are implemented in C++. This CAD tool generates optimized (and worst-case) multipliers for the given static probabilities, spatial correlation coefficients and temporal correlation ratios. The VHDL code for the generated multiplier structures are simulated using the ModelSim logic simulator from Mentor Graphics and the sum of average number of transitions per clock cycle for all nodes in the multiplier are reported. The average number of transitions per clock cycle is obtained from simulating the circuit for 10000 input samples with the desired pattern and static-probabilities. As the capacitances within the reduction-tree do not have large variations, the average number of transitions gives a good estimate of the dynamic power consumption.

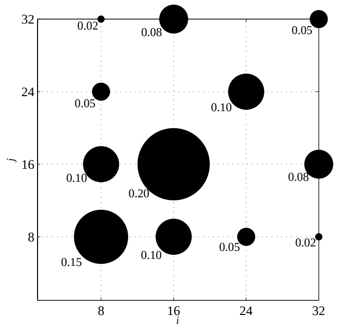

Fig. 3. Visualization of an example word-length probability density matrix $\mathcal{M}$ using circles' radii

In order to have a fair comparison of our optimization method, the optimized and worst-case multipliers are compared with random multipliers as well. For each example, we have generated ten random multipliers for which the permutations of equal-weight PPs are chosen randomly regardless of their transition activities. The average number of transitions reported for random multipliers are mean values obtained from these ten random multipliers.

4.1 General-Purpose Multiplier

In the first part of the experiments, we have considered a parallel multiplier that is operating in a general purpose processor. The programs executed on this processor are controlled by a variety of applications, exploiting the multiplier with different resolutions. By profiling a number of applications, we can have a rough estimate of the word-length probability density matrix. As an example, we have assumed that a 32×32-bit multiplier is embedded in a general-purpose processor. The word-length probability density matrix, $\mathcal{M}$, is visualized in Figure 3. The radius of a circle that is centered at location (i, j) is proportional to $\mu_{i,j}$; i.e., the probability of utilizing the multiplier for a $i \times j$-bit multiplication. For instance, 20% of the multiplications performed on the multiplier circuit will be 16×16-bit multiplication, while only 5% of the multiplications will be 32×32-bit.

Table 1. Average number of transitions for different multipliers

Multiplier	Input pattern applied to both operands					
Structure	Ω^{32}_{32}	Ω^{32}_{24}	Ω^{32}_{16}	Ω^{32}_{8}	$\overline{\mathbf{p}}$	$\overline{\mathbf{p}^*}$
$M_{\text{opt}}(\Omega^{32}_{32}, \Omega^{32}_{32})$	$\underline{6324}$	3574	1339	228	1728	1737
$M_{\text{opt}}(\Omega^{32}_{24}, \Omega^{32}_{24})$	6828	$\underline{3322}$	1312	226	1712	1734
$M_{\text{opt}}(\Omega^{32}_{16}, \Omega^{32}_{16})$	7109	4053	$\underline{1250}$	226	1833	1867
$M_{\text{opt}}(\Omega^{32}_{8}, \Omega^{32}_{8})$	7182	4084	1512	$\underline{207}$	1947	1967
$M_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$	6907	3805	1271	217	$\underline{1693}$	1781
$M_{\text{wc}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$	7167	4258	1715	307	2249	2132
$M^*_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$	6755	3579	1265	213	1719	$\underline{1709}$
$M^*_{\text{wc}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$	7369	4371	1749	305	2234	2193
$\overline{M_{\text{Random}}}$	7258	4180	1588	271	2063	2045

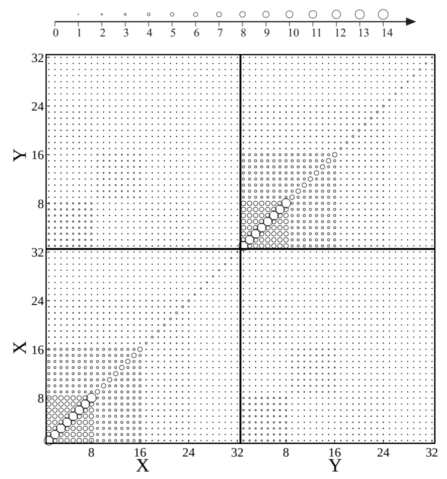

Fig. 4. Visualization of correlation coefficient matrix, $\mathcal{C}$, between primary input bits in example 1

Using Eq. 4 and Eq. 5, $\overline{\mathbf{p}_Y}$ and $\overline{\mathbf{p}_X}$ can be computed as illustrated in Figure 5. The least significant bits have lower average one-probabilities because they are often forced to zero. For the most significant bits, one-probabilities equal to 0.5.

Table 1 summarizes the results for this example. Different multiplier structures are placed in different rows of this table. The actual input pattern that is applied to different multiplier structures are shown in different columns. The last column denoted as $\overline{\mathbf{p}^*}$ is the input pattern with a word-length variation similar to that of Figure 2(a). The input pattern satisfies the word-length probability distribution shown in Figure 3 as well as the spatiotemporal correlations due to minimal word-length variation assumption discussed in Section 3. The pairwise spatial correlation coefficients and lag-one temporal correlation ratios for primary input bits are obtained from Eq. 1 and Eq. 9 by generating a large number of random inputs (100000 samples) with the desired pattern, i.e., similar to Figure 2.(a), for word-length variation. The spatial correlation coefficient matrix for primary input bits is shown in Figure 4 using radii of circles to visualized corelation coefficient values. The lag-one temporal correlation ratio is illustrated in Figure 5. In Table 1, the column with input pattern $\overline{\mathbf{p}}$ is an uncorrelated random input that satisfies the one-probabilities in Eq. 4 and Eq. 5. Note that the input patterns $\overline{\mathbf{p}}$ are $\overline{\mathbf{p}^*}$ are different because the former has uncorrelated random bits (Figure 2.(b)) while the latter is correlated (Figure 2.(a)).

The multiplier $M^*_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$ refers to the multiplier that is optimized for $\overline{\mathbf{p}}$ including the pairwise correlation coefficient matrix for primary inputs in Figure 4 and temporal correlations in Figure 5. $M_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$ is the optimized multiplier for $\overline{\mathbf{p}}$ without considering spatiotemporal correlations of the inputs. The numbers reported in the last row ($\overline{M_{\text{Random}}}$) are average number of transitions for ten random multipliers.

From Table 1 it can be seen that $M^*_{\text{wc}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$ has about 28% more transitions compared to $M^*_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$. Compared to randomly interconnected multipliers, the reduction in the average number of transitions is about 16%. Comparing the two multipliers

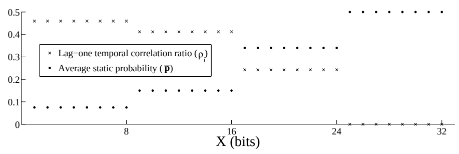

Fig. 5. Average static probabilities and lag-one temporal correlation ratio for primary input bits in example 1

$M^*_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$ and $M_{\text{opt}}(\overline{\mathbf{p}}, \overline{\mathbf{p}})$, it can be concluded that considering spatiotemporal correlations for primary inputs reduces the average number of transitions even further. An important conclusion from Table 1 is that if we do not have information about the word-length probability density matrix in Figure 3, even optimizing the multiplier for its full word-length can reduce the power consumption significantly. In this example, $M_{\text{opt}}(\Omega^{32}_{32}, \Omega^{32}_{32})$ experiences 15% less transitions compared to random multipliers. However, in order to achieve further reduction in power consumption, information about input patterns is necessary including spatiotemporal correlations.

4.2 FFT processor with variable word-length

A relevant case for using adaptive multiplier word-length could be the Fast Fourier Transform (FFT) computation in an OFDM receiver. An $n \times m$-bit multiplier is utilized in the FFT processor. This multiplier computes the multiplication product of the data input (input X with n bits) and the twiddle factor (input Y with m bits). With a fading channel, the channel noise power experiences large variation in time. Thus, the requirements to quantizing errors in the receiver DSP will also vary with time. For simplicity we assume a constant signal power $\sigma^2_X = 1$ for each subcarrier at the output of the FFT. Furthermore, we assume a Rayleigh fading channel resulting in a subcarrier noise at the FFT output. This noise will have a time-varying power σ^2_C resulting in a subcarrier channel signal to noise ratio (CSNR) γ with the exponential distribution:

$$f_\gamma(\gamma) = \frac{1}{\overline{\gamma}} \exp(-\frac{\gamma}{\overline{\gamma}}) \tag{11}$$

where $\overline{\gamma}$ is the mean value of the random variable γ. Furthermore, the quantization noise due to finite multiplier word-lengths should be negligible in comparison to the channel noise. This is ensured by keeping the quantization noise power, σ^2_Q, 20 dB below the channel noise, i.e.

$$\sigma^2_Q \leq \frac{1}{100} \sigma^2_X \tag{12}$$

Assuming the model in [14, Section 6.4.2] the quantization power with an n bit data word-length will then be given as

$$\sigma^2_Q = 2^{1-\nu-2n} \tag{13}$$

for an FFT of length 2^ν. The error due to finite twiddle factor word-length is assumed to be negligible compared to σ^2_X if $m = n + 2$ bits are used for these values.

We are now interested in the probability that Eq. 12 is fulfilled for the different multiplier sizes. This is indeed the values of the word-length probability density matrix $\mathcal{M}$. Assuming an FFT of length 2^6, we find that this is equivalent to requiring the CSNR (in dB) to be in a given interval:

$$\mu_{b,(b+2)} = P\left(41 - 6.02(b+1) < \gamma_{\text{dB}} \leq 41 - 6.02b\right) \tag{14}$$

Assuming a scenario with $\overline{\gamma}_{\text{dB}} = 30\text{dB}$ and using the distribution in Eq. 11, we can compute the probabilities as shown in Table 2. The elements of matrix $\mathcal{M}$ which are

Table 2. Word-length probabilities for the multiplier operating within an FFT with fading channel

$\mu_{5,7}$	0.0001	$\mu_{9,11}$	0.0154	$\mu_{13,15}$	0.2594
$\mu_{6,8}$	0.0002	$\mu_{10,12}$	0.0592	$\mu_{14,16}$	0.0049
$\mu_{7,9}$	0.0010	$\mu_{11,13}$	0.2032	$\mu_{15,17}$	0.0001
$\mu_{8,10}$	0.0039	$\mu_{12,14}$	0.4526		

Table 3. Average number of transitions for different multipliers

Multiplier Structure	Applied input pattern									
	$(\Omega_{15}^{15}, \Omega_{17}^{17})$	$(\Omega_{14}^{15}, \Omega_{16}^{17})$	$(\Omega_{13}^{15}, \Omega_{15}^{17})$	$(\Omega_{12}^{15}, \Omega_{14}^{17})$	$(\Omega_{11}^{15}, \Omega_{13}^{17})$	$(\Omega_{10}^{15}, \Omega_{12}^{17})$	$(\Omega_{9}^{15}, \Omega_{11}^{17})$	$(\Omega_{8}^{15}, \Omega_{10}^{17})$	$(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$	$(\overline{\mathbf{p}_X^*}, \overline{\mathbf{p}_Y^*})$
$M_{\text{opt}}(\Omega_{15}^{15}, \Omega_{17}^{17})$	$\underline{1158}$	1052	885	742	610	467	391	304	743	732
$M_{\text{opt}}(\Omega_{14}^{15}, \Omega_{16}^{17})$	1201	$\underline{1021}$	852	740	600	489	391	292	780	719
$M_{\text{opt}}(\Omega_{13}^{15}, \Omega_{15}^{17})$	1238	1072	$\underline{844}$	727	599	489	386	297	804	712
$M_{\text{opt}}(\Omega_{12}^{15}, \Omega_{14}^{17})$	1262	1112	904	$\underline{699}$	601	493	397	308	816	715
$M_{\text{opt}}(\Omega_{11}^{15}, \Omega_{13}^{17})$	1251	1110	937	758	$\underline{560}$	474	377	289	800	741
$M_{\text{opt}}(\Omega_{10}^{15}, \Omega_{12}^{17})$	1249	1113	959	789	617	$\underline{443}$	363	288	811	770
$M_{\text{opt}}(\Omega_{9}^{15}, \Omega_{11}^{17})$	1271	1138	985	834	688	520	$\underline{345}$	294	834	816
$M_{\text{opt}}(\Omega_{8}^{15}, \Omega_{10}^{17})$	1277	1145	975	841	698	568	428	$\underline{278}$	832	823
$M_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$	1237	1084	859	725	584	480	385	296	$\underline{729}$	711
$M_{\text{wc}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$	1422	1272	1120	985	828	678	521	406	932	960
$M_{\text{opt}}^*(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$	1236	1081	848	702	578	478	389	297	780	$\underline{696}$
$M_{\text{wc}}^*(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$	1417	1279	1128	988	831	679	542	418	929	966
$\overline{M_{\text{Random}}}$	1359	1218	1046	892	745	610	482	365	887	878

not given in Table 2 are zero. We choose the largest multiplier size to be 15 × 17-bit. In order to save power during the operation of the FFT processor, the multiplication size will vary, based on the quality of the channel. For instance, when a 12 × 14-bit multiplier is needed three bits from LSB side will be forced to zero on both inputs.

Table 3 summarizes the average number of transitions for different multiplier structures. Each row shows a different multiplier structure and each column shows the input pattern that is applied to the multiplier. The rightmost column, $(\overline{\mathbf{p}_X^*}, \overline{\mathbf{p}_Y^*})$, is the variable word-length input pattern with the probabilities shown in Table 2. It is again assumed that word-length transitions do not happen often. The input pattern $(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ shows the uncorrelated random input bits where the input bits have the average one-probabilities obtained from Eq. 4 and Eq. 5. $M_{\text{opt}}^*(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ is the multiplier that is optimized for $(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$, considering the spatiotemporal correlations for primary inputs. $M_{\text{opt}}(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ is the multiplier that is optimized for $(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ but without considering the spatiotemporal correlations for primary inputs. The lowest power consumption when $(\overline{\mathbf{p}_X^*}, \overline{\mathbf{p}_Y^*})$ is applied to the multiplier, is found in row $M_{\text{opt}}^*(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$ which has 21% and 28% less transitions compared to random multipliers and the worst-case multiplier respectively. The most probable multiplication in this example is 12 × 14-bit. Table 3 shows that $M_{\text{opt}}(\Omega_{12}^{15}, \Omega_{14}^{17})$ reduces the number of transitions significantly when $(\overline{\mathbf{p}_X^*}, \overline{\mathbf{p}_Y^*})$ is applied. However, the reduction is less than $M_{\text{opt}}^*(\overline{\mathbf{p}_X}, \overline{\mathbf{p}_Y})$.

5 Conclusions

We have proposed a method to reduce the power consumption in parallel multipliers, when the word-length is varying. The optimizer utilizes the arithmetic equivalencies within the reduction-tree of the multiplier and selects low-power solutions among the numerous functionally equivalent solutions. The optimization inputs are the average one-probabilities and spatiotemporal correlations for primary input bits. Word-length variation introduces spatiotemporal correlations between primary inputs. Our method is capable of capturing such correlations. Compared to random multipliers, the average number of transitions is reduced 16-21% in the optimized multipliers.

References

1. C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm," *IEEE Trans. Computers*, vol. C-22, pp. 1045–1047, 1973.
2. K. C. Bickerstaff, M. J. Schulte, and E. E. Swartzlander, Jr., "Reduced area multipliers," in *Proc. Intr. Conf. on App.-Specific Array Processors*, pp. 478–489, 1993.
3. J. Chittamuru, W. Burleson, and J. Euh, "Dynamic wordlength variation for low-power 3D graphics texture mapping," in *IEEE Workshop on Signal Processing Systems*, pp. 251–256, 2003.
4. M. D. Ercegovac and T. Lang, *Digital Arithmetic.* Morgan Kaufmann Publ., 2004.
5. S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Riccó, "Estimate of signal probability in combinational logic networks," in *Proc. 1st European Test Conf.*, pp. 132–138, 1989.
6. M. Hatamian and G. L. Cash, "A 70-MHz 8-bit x 8 bit parallel pipelined multiplier in 2.5-μm CMOS," *IEEE J. Solid-State Circ.*, vol. SC-21, no. 4, pp. 505–513, 1986.
7. K.-Y. Khoo, Z. Yu, and A. N. Willson, "Bit-level arithmetic optimization for carry-save additions," in *Proc. IEEE/ACM Intr. Conf. on Computer-aided design*, pp. 14–19, 1999.
8. P. E. Landman and J. M. Rabaey, "Architectural power analysis: the dual bit type method," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 3:2, pp. 173–187, 1995.
9. W. Ling and Y. Savaria, "Variable-precision multiplier for equalizer with adaptive modulation," in *Proc. 47th Midwest Symp. Circuits and Syst.*, pp. I–553–556, 2004.
10. K. T. M. Muroyama, S. Yamaguchi, and H. Yasuura, "A design method for a low power equalization circuit by adaptive bitwidth control," in *IEEE Intr. Symp. Communications and Information Technology*, pp. 704–709, 2004.
11. V. Oklobdzija, D. Villeger, and S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," *IEEE Trans. Computers*, vol. 45, no. 3, pp. 294–306, 1996.
12. B. Parhami, *Computer Arithmetic - Algorithms and Hardware Design.* New York, US: Oxford University Press, 2000.
13. M. Potkonjak and J. M. Rabaey, "Optimizing resource utilization using transformations," in *Proc. IEEE/ACM Intr. Conf. on Comp.-aided design*, pp. 88–91, 1991.
14. J. Proakis and D. G. Manolakis, *Digital Signal Processing, Principles, Algorithms, and Aplications.* New Jersey, USA: Prentice Hall, 3rd ed., 1996.
15. S. Ramprasad, N. R. Shanbhag, and I. N. Hajj, "Analytical estimation of transition activity from word-level signal statistics," in *Proc. Design Automation Conf.*, pp. 582–587, 1997.
16. S. Tahmasbi Oskuii, P. G. Kjeldsberg, and E. J. Aas, "Probabilistic gate-level power estimation using a novel waveform set method," in *Proc. 17th Great Lakes Symp. on VLSI*, pp. 37–42, March 2007.
17. S. Tahmasbi Oskuii, P. G. Kjeldsberg, and O. Gustafsson, "Power optimized partial product reduction interconnect ordering in parallel multipliers," in *Proc. 25th IEEE Norchip Conf.*, (Aalborg, Denmark), November 2007.
18. S. Yoshizawa and Y. Miyanaga, "Tunable wordlength architecture for a low power wireless OFDM demodulator," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E89-A, no. 10, pp. 2866–2873, 2006.
19. Z. Yu, L. Wasserman, and A. Willson, "A painless way to reduce power dissipation by over 18% in Booth-encoded carry-save array multipliers for DSP," in *Proc. IEEE Workshop Signal Processing Syst.*, pp. 571–580, October 2000.