DATA WORDLENGTH REDUCTION FOR LOW-POWER SIGNAL PROCESSING SOFTWARE

Kyungtae Han, Brian L. Evans, and Earl E. Swartzlander, Jr.

Dept. of Electrical and Computer Engineering
The University of Texas at Austin, Austin, TX 78712-1084 USA
khan@ece.utexas.edu, bevans@ece.utexas.edu, eswartzla@aol.com

ABSTRACT

Reducing power consumption prolongs battery life and increases integration. In digital CMOS designs, switching activity is closely connected with the total power consumption. Switching activity on programmable processors implementing linear filters, fast Fourier transforms, and other signal processing operations is dominated by the hardware multiplier. In this paper, we employ wordlength reduction of multiplicands to reduce switching activity in hardware multipliers using truncation and signed right shift methods. For 32 bit $\times$ 32 bit Wallace and Radix-4 modified Booth multipliers, truncation by 16 bits achieves a 4:1 and 2:1 reduction, respectively, in switching activity, whereas signed right shift gives little or no reduction. The key contribution of this paper is the reduction of power consumption by altering multiplicands in software without any hardware modifications.

1. INTRODUCTION

Portable computing demands minimizing power dissipation due to a limited power supply. Many methods have been developed to reduce power consumption. Lowering supply voltage and minimizing hardware area are used for low-power hardware. Changing instruction order and reducing the number of operations are used for low-power software.

A major focus of low power design is to reduce the switching activity to the minimal level required to perform the computation, since power in CMOS circuits is dissipated when they are switching [1]. The required level can be varying according to application. For example of wireless communications, when the signal-to-noise ratio (SNR) of a channel is lower, a full range of computation is needed. However, when the SNR is higher, a smaller range of computation is sufficient. Therefore switching activities can be minimized by varying the computation range.

Multiply units are usually a major source of power consumption in typical DSP applications. Many digital blocks use the multiply unit for wireless communications. For example, in digital transceivers of wireless LAN (IEEE 802.11),

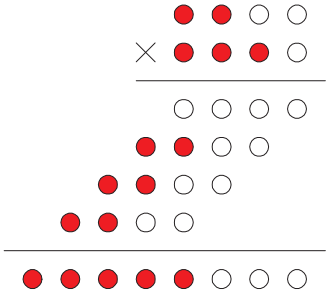

Fig. 1. Example of data wordlength reduction in a 4 bit $\times$ 4 bit multiplier using 2-bit and 3-bit multiplicands. The filled circles affect the switching power consumption during computation while the unfilled circles do not.

multiplication units are used in digital filters, equalizers, fast Fourier transforms (FFTs), inverse FFTs, etc. The proposd data wordlength reduction methods are applied to multiplication.

Data wordlength reduction methods for low-power signal processing software are proposed in this paper. A data wordlength reduction example is shown in Fig. 1. Fig. 1 shows 4 bit $\times$ 4 bit multiplication with 2-bit and 3-bit data wordlengths. The filled circles affect the switching power consumption during computation while the unfilled circles do not. This multiplication has lower power consumption compared to 4-bit and 4-bit data since there are fewer filled circles.

The data wordlength reduction methods reduce the transition counts and power consumption while not changing any hardware. The reduction methods can adaptively change data wordlength according to demand of power minimization and robust computation.

The objective of this research is to minimize power dissipation in digital signal processing blocks by reducing data wordlength at the software level while not changing the hardware architectures. After a brief summary of power analysis in CMOS circuits in Section 2 and a description of multipli-

ers in Section 3, two techniques will be presented to reduce power dissipation at the software level in Section 4.

2. BACKGROUND

2.1. Power Analysis

There are three major sources of power dissipation in digital CMOS circuits that are summarized in the following equation: [1]

$$P_{avg} = P_{switching} + P_{short-circut} + P_{leakage}. \quad (1)$$

The first term represents the switching component of power, the second term is due to the direct-path short circuit current conducting current directly from the supply to ground, and the leakage power is primarily determined by fabrication technology.

The switching component of average power is

$$P_{switching} = \alpha C_L V_{dd}^2 f_{clk} \quad (2)$$

where α is the switching activity parameter, C_L is the load capacitance, V_{dd} is the operating voltage and f_{clk} is the operating frequency. The switching power can be reduced through operation reduction, choice of number representation, exploitation of signal correlations, logic design, and physical design. The switching activity can be also reduced by optimizing the ordering of operations and by minimizing number of operations.

The term αC_L can also be viewed as the effective switching capacitance of the transistor nodes from charging and discharging. Therefore minimizing switching activities can effectively reduce power dissipation without impacting the circuit's operational performance [2].

Directly measuring the power consumption is difficult. The average number of transitions is usually used as an estimate of the requirement.

2.2. Software Power Minimization

Tiwari, Malik, and Wolfe [3] attempted to systematically model software power cost, because of the increasing demand for a software power analysis tool. They formulated an instruction level power model for the microprocessor after measuring the power of instruction sets. This made it possible to compare programs in terms of their energy consumption.

Lee, Tiwari, Malik, and Fujita [4] developed power analysis and minimization techniques for embedded DSP software. They found that in typical DSP applications, the multiplier in the multiply and accumulate (MAC) unit is usually a major source of power consumption. A micro-architectural power model for the multiplier was developed and analyzed for further power minimization. They observed a wide power variation of MAC instructions mainly according to the two values being multiplied in MAC unit.

They also used the operand swapping technique for a Booth multiplier [5]. The Booth multiplier does not treat the two inputs symmetrically. Their experiment showed that swapping the operations in register A and B can reduce the power for MAC instructions. They also used instruction packing, instruction scheduling, and memory bank assignment to reduce energy consumption.

2.3. Minimizing Wordlength for Low-Power

Chandrakasan *et al.* [6] showed that the wordlength affects all key parameters of a design, including speed, area, and power. Choi and Burleson [7] presented a general search-based methodology for wordlength optimization and used switching power model for the power dissipation. Considering a voltage dropping factor and the area of computing elements according to the wordlengths, they analyzed the switching power consumption assuming that the power dissipation is proportional to the area of computing element.

Erdogan and Arslan [8] showed low power multiplication schemes for finite impulse response (FIR) filters on DSP processors. They used data bus and coefficient bus separately for the filtering operation. They measured the switching activity of 8, 16, and 32 bit array multipliers for filter orders of 32, 64, and 128. They achieved up to a 63% reduction in switching activity by ordering of coefficient and using a pre-calculated value memory.

Chen, Wang, and Wu [2] presented low-power two's complement multipliers by minimizing the switching activities of partial products using the Radix-4 modified Booth algorithm [9]. They used the fact that switching activities of the unused functional blocks are minimized when input bits of unused functional blocks remain unaltered. They increased the probability that the partial products become zero by swapping input data.

Wordlength can be also changed by reconfiguring the multiplier. Kim and Papaefthymiou [10] proposed a reconfigurable pipelined multiplier architecture by adapting its structure to computational requirements over time. It can efficiently cope with variable data-rate multimedia applications such as video processing. The multiplier structures can dynamically reconfigure to lower their power consumption based on zero-valued inputs and input-rate variations.

3. MULTIPLIER

The hardware multiplier on most Programmable DSPs uses either the Wallace or the Radix-4 modified Booth algorithm [9]. For example, the TI TMS320C64 uses the Wallace algorithm and the TI TMS320C62 uses the Radix-4 modified Booth algorithm.

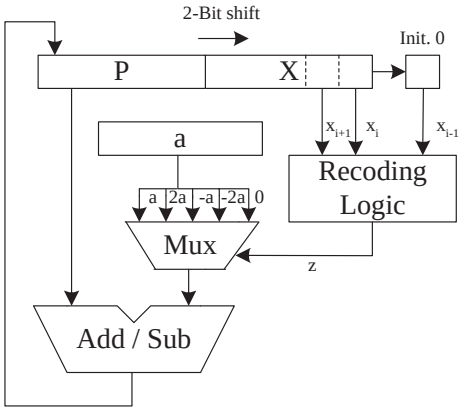

Fig. 2. A Radix-4 multiplier based on Booth's recoding. The a and x are multiplicands. P is product of multiplication. Three bits in X are recoded to z.

3.1. Wallace Multiplier

Fig. 1 shows a simple version of the partial products from a small multiplier. In a tree-based multiplier, the partial products are added using full adders. A full-adder can be considered as a 3-to-2 counter that adds three inputs and forms a two bit result. In 1964, Wallace showed that a tree structure of such counters is an efficient method to add the partial products [11].

3.2. Radix-4 Modified Booth Multiplier

Booth recoding is a commonly used technique to recode one of the operands in binary multiplication. Fig. 2 shows a Radix-4 multiplier of $a \times x$, based on Booth's recoding. A two's complement multiplier, x, is recoded as a Radix-4 number, z, which dictates the multiples -2a, -a, 0, a, and 2a to be added to the cumulative partial product. The Radix-4 Booth's recoding table is shown in Table 1.

Table 1. Radix-4 Booth's recoding. The a and x are multiplicands. 3 bits of x are recoded into z.

x_{i+1}	x_i	x_{i-1}	z	*action*
0	0	0	0	0
0	0	1	1	a
0	1	0	1	a
0	1	1	2	2a
1	0	0	-2	-2a
1	0	1	-1	-a
1	1	0	-1	-a
1	1	1	0	0

$$\begin{array}{rcccc} & 1 & 2 & 3 & 4 \\ \times & D & C & A & 9 \\ \hline \end{array}_{(16)}$$

(a) Original Multiplication

$$\begin{array}{rcccc} & 0 & 0 & 1 & 2 \\ \times & F & F & D & C \\ \hline \end{array}_{(16)}$$

(b) Reduction via Signed Right Shift

$$\begin{array}{rcccc} & 1 & 2 & 0 & 0 \\ \times & D & C & 0 & 0 \\ \hline \end{array}_{(16)}$$

(c) Reduction via Truncation

Fig. 3. Example of 8-bit data wordlength reduction.

4. DATA WORDLENGTH REDUCTION

For low-power design, much research on wordlength minimizing methods has been focused on minimizing hardware. Our data wordlength reduction method, however, considers input data wordlength minimization without any hardware modification.

4.1. Wordlength Reduction in Multiplication

There are two kinds of data wordlength reductions. One is reduction via right-shifting and the other is reduction via left-shifting i.e., with truncation. An example of 8-bit reduction from 16-bit multiplication is shown in Fig. 3. The 16-bit multiplication is shown in Fig. 3(a). The reduction of 8-bit right-shift moves 8 bit data in most significant bit (MSB) side into least significant bit (LSB) side as shown in Fig. 3(b). One of the data, 'DCA9', is changed into 'FFDC' because the signed right shift fills the MSB side with ones when data is negative i.e., via sign extension. The reduction via truncation masks the input data with 'FF00' and the results are shown in Fig. 3(c).

4.2. Dynamic Range

Dynamic range (DR) is defined as

$$\mathrm{DR} = 20\log_{10}\frac{\text{Range of representable numbers}}{\text{Smallest positive representable number}}. \quad (3)$$

For a b-bit number, dynamic range is

$$20\log_{10}(2^b - 1). \quad (4)$$

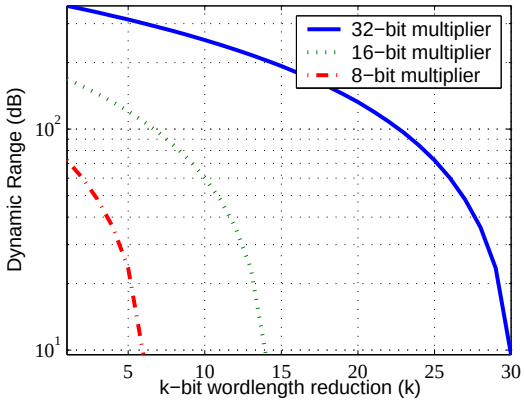

Fig. 4. Dynamic range of the multiplier output with a reduction of k bit wordlength for each multiplicand.

As the data wordlength is reduced, the dynamic range is decreased. The output wordlength of b bit $\times$ b bit multiplication for unsigned data and signed data is $2b$ and $2b-1$, respectively. In $n \times n$-bit multiplication with k-bit data reduction, the output wordlength for unsigned data and singed data is $n-k$ and $n-k-1$, respectively. Therefore dynamic range for unsigned data in the output of n bit $\times$ n bit multiplication with k-bit data reduction is

$$\mathrm{DR_{us}} = 20\log_{10}(2^{2(n-k)} - 1), \tag{5}$$

and dynamic range for signed data is

$$\mathrm{DR_{s}} = 20\log_{10}(2^{2(n-k)-1} - 1). \tag{6}$$

Table 2 and Fig. 4 show the output dynamic range of 8-bit, 16-bit, and 32-bits multipliers with k-bit wordlength reduction for signed data.

5. SIMULATION RESULTS AND DISCUSSIONS

The Wallace multiplier in this simulation is composed of 3 to 2 compressors. The Radix-4 modified Booth multiplier in this simulation is composed of recoding logic, a multiplexer, and ripple carry adders. The ripple carry adders are built from 9-gate full adders and 5-gate half adders. For this simple model, all gates are assumed to have a unit gate delay. The gate output values are monitored and the number of transitions is counted. We implemented both Wallace and Radix-4 modified Booth multipliers at the gate-level in Verilog and simulated using Synopsys Verilog Compiler Simulator.

Table 2. Dynamic range of the output with k bit wordlength reduction (dB).

Muliplier	*k-bit Reduction*			
	4	8	12	16
8-bit	36	n/a	n/a	n/a
16-bit	132	84	36	n/a
32-bit	325	277	229	181

Table 3. Average transition counts of a 32 bit $\times$ 32 bit Wallace multiplier.

Reduction Method	*Data Wordlength*			
	4 bits	8 bits	16 bits	32 bits
Signed Right Shift	32996	31364	28634	22021
Truncation	404	1383	6138	22021

Table 4. Switching activity in a 32 bit $\times$ 32 bit Wallace multiplier.

Reduction Method	*Data Wordlength*		
	4 bits	8 bits	16 bits
Signed Right Shift	150%	142%	130%
Truncation	2%	6%	28%

5.1. Wallace Multipliers

Table 3 shows the average transition counts of the Wallace multipliers. For the data wordlength reduction, the 32×32-bit Wallace multiplier is used during signed right-shift and truncation operation. The data wordlength of 32 bits has same average counts. As the data wordlegnth is reduced by a factor of 2 from 32 bits to 16 bits, the signed right-shift reduction increases the average transition counts. The truncation method, however, decreases the counts by a factor of 4. This implies that the average transition counts with the truncation method decreases by n^2 in the Wallace multiplier. The average transition counts with 16 bit wordlength reduction by truncation are decreased by 72%, while reduction by right-shift increases by 30% of the counts as shown in Table 4.

Fig. 5 also shows the average transition counts of the data wordlength reduction in Wallace multipliers. The signed right-shift reduction increases the counts, and the truncated reduction decreases the counts.

The increase of the transition counts in the right-shift reductions results from the arithmetic right-shift, i.e. sign extension that fills the MSBs with ones causing transitions. Therefore, the signed right-shift reduction in Wallace multiplier consumes more power than no data wordlength reduction.

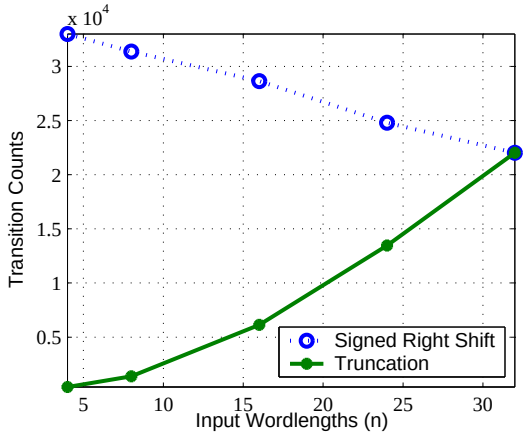

Fig. 5. Average transition counts of data wordlength reduction in Wallace multiplier.

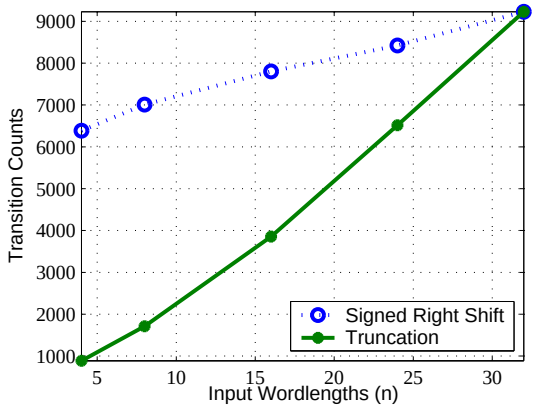

Fig. 6. Average transition counts of data wordlength reduction in Radix-4 modified Booth multiplier.

Table 5. Average transition counts of a 32-bit Radix-4 modified Booth multiplier.

Reduction Method	*Data Wordlength*			
	4 bits	8 bits	16 bits	32 bits
Signed Right Shift	6386	7008	7801	9230
Truncation	888	1712	3855	9230

5.2. Radix-4 Modified Booth Multipliers

Table 5 shows the average transition counts of the Radix-4 modified Booth multiplier. For the data wordlength reduction, the 32×32-bit Booth multiplier is used during signed right-shift and truncation operation. The data wordlength of 32 bits has same average counts. As the data wordlegnth is reduced by a factor of 2 from 32 bits to 16 bits, the signed right-shift reduction and the truncation method decrease the average transition counts.

The average transition counts in 16 bit wordlength reduction by the signed right shift and the truncation are decreased by 15% and 58% respectively as shown in Table 6.

Fig. 6 shows the average transition counts of the data wordlength reduction in Radix-4 modified Booth multiplier.

Table 6. Switching activity in a 32-bit Radix-4 modified Booth multiplier.

Reduction Method	*Data Wordlength*		
	4 bits	8 bits	16 bits
Signed Right Shift	69%	76%	85%
Masking	10%	19%	42%

The signed right shift reduction induces consecutive ones as shown in Fig. 3(b). Signed right shift reduction in Booth multipliers, however, decreases the transition counts since the recoding logic recodes consecutive ones into zeros as shown in Table 1.

The decrease of the transition counts by signed right-shifting and truncation in Booth multipliers results in less power consumption than no data wordlength reduction.

6. CONCLUSION

Two kinds of data wordlength reduction methods in Wallace multipliers and Booth multipliers are proposed and simulated. Data wordlength reduction by the truncation method decreases the average transition counts and power consumption. In the Wallace multiplier, the average transition counts with reduction by truncation decreases at an n^2 rate, although reduction by signed right-shift increases the counts. For 32×32-bit Wallace multiplier, average transition counts with 16 bit wordlength reduction by masking is decreased by 72%, while reduction by signed right-shift increases by 30%. In the Radix-4 modified Booth multiplier, the average counts are decreased for both the truncation and the signed right shift methods. For 32×32-bit Radix-4 modified Booth multiplier, average transition counts by 16-bit truncation and by 16-bit signed right shift is decreased by 15% and 58% respectively. The proposed wordlength reduction methods reduce power consumption by altering multiplicands in software without any hardware modifications. The drawbacks of this process are decreasing the dynamic range and required additional truncation or signed right shift operations. Future extensions of this work include finding optimum wordlength reduction and minimizing the additional operations.

7. REFERENCES

[1] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits," *Proceedings of the IEEE*, vol. 83, pp. 498–523, 1995.

[2] Oscal T.-C. Chen, Sandy Wang, and Yi-Wen Wu, "Minimization of switching activities of partial products for designing low-power multipliers," *IEEE Transactions on VLSI Systems*, vol. 11, pp. 418–433, 2003.

[3] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: A first step towards software power minimization," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, San Jose, CA, Nov. 1994, pp. 429–435.

[4] M. T. Lee, V. Tiwari, S. Malik, and M. Fujita, "Power analysis and minimization techniques for embedded DSP software," *IEEE Transactions on VLSI Systems*, vol. 5, pp. 123–135, 1997.

[5] A.D. Booth, "A signed binary multiplication technique," *Quart. J. Mech. Appl. Math.*, vol. 4, pp. 236–240, 1951.

[6] A. P. Chandrakasan, M. Potkonjak, J. Rabaey, and R. W. Brodersen, "Optimizing power using transformations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, pp. 12–31, 1995.

[7] H. Choi and W. P. Burleson, "Search-based wordlength optimization for VLSI/DSP synthesis," in *Proc. IEEE Workshop on VLSI Signal Processing*, Oct. 1994, vol. 7, pp. 198–207.

[8] A. T. Erdogan and T. Arslan, "Low power multiplication scheme for FIR filter implementation on single multiplier CMOS DSP processors," *IEE Electronics Letters*, vol. 32, pp. 1959–1960, 1996.

[9] Behrooz Parhami, *Computer Arithmetic algorithm and hardware designs*, Oxford University Press, 2000.

[10] S. Kim and M. Papaefthymiou, "Reconfigurable low-energy multiplier for multimedia system design," in *Proc. IEEE Workshop on VLSI*, Apr. 2000, pp. 129–134.

[11] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Transactions on Computers*, vol. 13, pp. 14–17, 1964.