Threshold Logic Parallel Counters for 32-bit Multipliers

Peter Celinski[a], Sorin D. Cotofana[b] and Derek Abbott[a]

[a]Centre for High Performance Integrated Technologies & Systems (CHiPTec) and
Centre for Biomedical Engineering (CBME)
Department of Electrical & Electronic Engineering
The University of Adelaide, SA 5005,
Australia.

[b]Computer Engineering Group
Electrical Engineering Department
Delft University of Technology
Mekelweg 4, 2628 CD Delft,
The Netherlands. *

ABSTRACT

In recent years, there has been renewed interest in Threshold Logic (TL), mainly as a result of the development of a number of successful implementations of TL gates in silicon. Threshold Logic enables, in some instances, the design of digital integrated circuits with a significantly reduced transistor count and area. This paper addresses the important problem of designing technologically feasible parallel (m,n) counters using TL for binary multiplication. A number of counter design techniques are reviewed and some novel parallel counter designs are presented that allow the design of area efficient 32-bit multiplier partial product reduction circuits.

Keywords: Threshold logic, parallel counters, 32-bit multipliers, VLSI

1. INTRODUCTION

As the demand for higher performance very large scale integration processors with increased sophistication grows, continuing research is focused on improving the performance, area efficiency, and functionality of the arithmetic units contained therein. Low power dissipation and area have become major issues demanded by the high performance processor market in order to meet the high density requirements of advanced VLSI processors.

Threshold logic (TL) was introduced over four decades ago, and over the years has promised much in terms of reduced logic depth and gate count compared to traditional AND-OR-NOT logic-gate based design. However, lack of efficient physical realizations has meant that TL has, until recently, had little impact on VLSI. Efficient TL gate realizations have recently become available[1–6] and applications based on TL gates have demonstrated their potential to achieve high operating speed and significantly reduced area compared to conventional logic.[7] However, no large scale designs based on TL have been designed and implemented.

This paper presents a number of threshold logic parallel counter networks for application in 32-bit multiplier design. Section 2 gives a brief overview of threshold logic, followed by a description of parallel counters in Section 3. Section 4 contains the main results of this work and includes an overview of previous work related to parallel counter design, a description of two new proposed (15,4) counter networks, a discussion of the cost of the various designs and their application to the design of 32-bit partial product reduction circuits. A brief conclusion is given in Section 5.

2. THRESHOLD LOGIC

A threshold logic gate is functionally similar to a hard limiting neuron. The gate takes n binary inputs $X_1, X_2, \ldots, X_n$ and produces a single binary output Y, as shown in Fig. 1. A linear weighted sum of the binary inputs is computed followed by a thresholding operation.

P. Celinski, corresponding author: celinski@eleceng.adelaide.edu.au, S. D. Cotofana: sorin@ce.et.tudelft.nl, D. Abbott: dabbott@eleceng.adelaide.edu.au

Smart Structures, Devices, and Systems, Erol C. Harvey, Derek Abbott, Vijay K. Varadan,
Editors, Proceedings of SPIE Vol. 4935 (2002) © 2002 SPIE · 0277-786X/02/$15.00

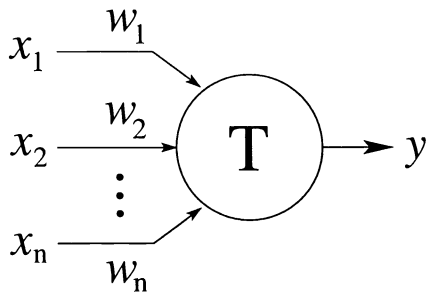


Figure 1. Threshold Gate Model

The Boolean function computed by such a gate is called a threshold function and it is specified by the gate threshold T and the weights w_1,w_2,...,w_n, where w_i is the weight corresponding to the ith input variable X_i. The output Y is given by

$$Y = \begin{cases} 1, & \text{if } \sum_{i=1}^{n} w_i X_i \geq T \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

This function can be written in a more compact form using the sgn notation as

$$Y = \text{sgn}\left(\sum_{i=1}^{n} w_i X_i - T\right), \quad (2)$$

where the sgn function is defined as $\text{sgn}(x) = 1$ if $x \geq 0$ and $\text{sgn}(x) = 0$ if $x < 0$.

Any threshold function can be computed with positive integral weights and a positive real threshold, and all Boolean functions can be realized by a threshold gate network of depth at most two.[8] A TL gate can be programmed to realize many distinct Boolean functions by adjusting the threshold T. For example, an n-input TL gate with $T = n$ will realize an n-input AND gate and by setting $T = n/2$, the gate computes a majority function. This versatility means that TL offers a significantly increased computational capability over conventional AND-OR-NOT logic. Significantly reduced area and increased circuit speed can therefore be obtained, especially in applications requiring a large number of input variables, such as arithmetic.

3. THRESHOLD LOGIC PARALLEL COUNTERS

An (m,n) binary counter is a combinatorial network which generates a binary coded output vector of length n which corresponds to the number, or count, of logic 1's in the m-bit input vector. Usually $n = \log_2(m + 1)$ and such counters are referred to as *saturated.* The full adder is a particular case of a counter with 3 inputs and 2 outputs, thus it is a (3,2) counter.

The concept of parallel counters was originally proposed by Dadda[9] who also developed a scheme for interconnecting small counters to design counters with a larger number of inputs.[10] In conventional logic, higher order counters, such as (7,3), (15,4) or (31,5), have traditionally been implemented by using trees of (3,2) counters because of the disadvantages of a direct implementation.[11] However, counters consisting of such full adder trees have a relatively high delay and grow rapidly with input vector size in terms of the required number of full adders. Swartzlander[12] reported the number of full adders for an m-input population counter as $m - \log_2 m$. It would be ideal if it were possible to design area efficient higher order counters which could operate at much higher speeds than the same counters built using trees of full adders.[13] Threshold logic allows us to do exactly this.

There are a number of TL based counter designs. The known networks for implementing counters are based on networks devised for computing symmetric functions such as parity. These include designs by Muroga,[8] Kautz[14] and Minnick.[15] It was later observed that the intermediate outputs available in the Kautz network for computing parity could be used as generalized counter outputs.[16]

We will refer to these networks as the Muroga, Kautz and Minnick counters. They differ in the number of threshold gates, network depth, maximum fan-in and maximum sum of weights per gate. The maximum sum of weights is important as it sets the minimum signal difference (voltage difference in voltage mode gates, current

difference in current mode gates) which is required to be resolved by the sense amplifier in the TL gate in a given technology and under the presence of device mismatch and circuit noise.

The counter based on Muroga's network has a delay of 2 gates but is more expensive in terms of gate number. Furthermore, even though the primary inputs are only required in the input layer, the larger number of input layer gates results in an increased interconnect overhead compared to the Kautz and Minnick designs.

The Kautz counter is the most area efficient, considering interconnection, weight size and gate number requirements. However, the Kautz counter has a logarithmic worst case delay as a function of the number of inputs, ie. the (7,3) counter has a depth of 3, the (15,4) has a depth 4 and so on. This means that circuits based on this counter, while being compact, have a relatively high delay. The most significant output bit always has a delay of one threshold gate.

The Minnick counter worst case delay for all outputs is equal to two threshold gates and is independent of the order of the counter. The most significant output always has a delay of one threshold gate, and the design offers improved gate count compared to the Muroga counter. However, depending on the particular gate implementation used, unlike the Muroga network, it may exhibit timing hazards.

A different approach to counter design was claimed to have been developed in.[17] This approach is based on a sorter circuit followed by one layer of threshold gates to obtain the counter outputs. This circuit is in fact equivalent in structure to the Muroga counter reported earlier. While it was shown to have improved speed over a conventional logic full adder based design for the case of a (15,4) counter, its logic depth is the same, its interconnect requirement is significantly higher and its gate count is almost double that of the Minnick counter.

As an illustrative example, Fig. 2 shows the truth table and TL network for the (7,3) Minnick counter. The input v consists of the seven input bit lines, each having a weight of 1, and is denoted by a thick black line to differentiate it from the single bit lines.

V	O_4	O_2	O_1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

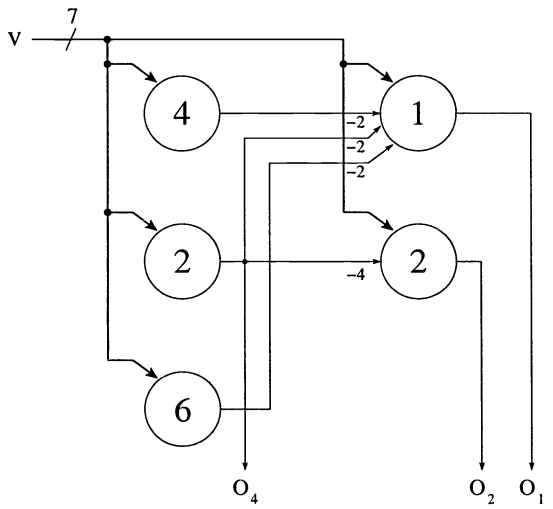


Figure 2. The (7,3) counter truth table and the Minnick TL network

4. PROPOSED THRESHOLD LOGIC COUNTER NETWORKS

The aim of this section is to develop a range of counters which may be considered to lie in-between the ends of the counter topology spectrum ranging from the Minnick design to Kautz. We focus on small counters which are feasible to implement in the currently available CMOS processes. Typically, the available precision with which the weights are implemented in VLSI is of the order of 4 to 6 bits[18] for an analog implementation, and for this reason we limit the maximum sum of weights inputs per TL gate to approximately 30.

The objective will be to develop binary counters which can be used in conjunction with the known Minnick and Kautz networks to design near area optimal 32-bit multiplier partial product matrix reduction schemes, given a

Table 1. Comparison of the various designs for saturated (3,2), (7,3) and (15,4) counters.

		TL Gate Count	Sum of Non I/P Weights	Sum of I/P Weights	Maximum Sum of Weights per Gate	Cost
(3,2)	Muroga	5	3	12	3	$5g + 15w$
	Kautz	2	2	6	5	$2g + 8w$
	Minnick	2	2	6	5	$2g + 8w$
(7,3)	Muroga	12	10	70	7	$12g + 80w$
	Kautz	3	10	21	13	$3g + 31w$
	Minnick	5	10	35	13	$5g + 45w$
(15,4)	Muroga	25	25	330	15	$25g + 355w$
	Kautz	4	34	60	29	$4g + 94w$
	Minnick	10	34	150	29	$10g + 184w$
	[1,2,2,3]	6	34	90	29	$6g + 124w$
	[1,2,3,3]	6	50	90	29	$6g + 140w$

particular CMOS TL gate implementation. Counters which trade off network depth against gate count and fan-in can be used off the critical path of the partial product reduction tree, there by reducing the overall area of the multiplier without a penalty in latency. The following sections show the known existing topologies for counters which have a maximum of 29 inputs per TL gate, including the Minnick and Kautz networks, and the proposed in-between (15,4) counters.

4.1. Depth 2 (3,2), (7,3) and (15,4) Counters

Figures 3(a)-(c) show the saturated depth 2 Minnick counters up to 15 inputs. The delay of the most significant output bit is always one TL gate, and all other outputs have a delay of 2 gates. For the (7,3) Minnick counter, the delay for each output is denoted by [1,2,2].

4.2. Logarithmic Depth (7,3) and (15,4) Counters

Figures 4(a)-(b) show the saturated logarithmic depth Kautz counters up to 15 inputs. The (3,2) Kautz counter is identical to the (3,2) Minnick counter. The (7,3) Kautz counter has the [1,2,3] delay profile, and the (15,4) Kautz counter has the [1,2,3,4] delay profile.

4.3. Proposed Depth 3 (15,4) Counters

In addition to the depth 2 and logarithmic depth counters, it is possible to devise networks by using a hybrid approach. Figure 5(a) shows the proposed (15,4) counter with a [1,2,2,3] delay profile. This means that the output O_8 is generated after 1 gate delay, outputs O_4 and O_2 are generated after 2 delays and the O_1 output after 3 delays. The O_2 and O_4 outputs are generated as in the Minnick counter in Figure 3(c) and the least significant output bit O_1 is generated in Kautz fashion as in Figure 4(b) using O_2, O_4 and O_8. Figure 5(b) shows the (15,4) counter with [1,2,3,3] delay profile. This counter computes the outputs O_1, O_2 and O_4 in Minnick fashion based on the value of O_8. Both counters have a maximum sum of weights equal to 29, but the [1,2,2,3] counter has a total of 34 non-I/O weights, compared to 50 for the [1,2,3,3] design. Other counters such as [1,3,3,3] and [1,3,2,3] are possible but would not reduce the gate count, thus increasing delay with no reduction in cost and will therefore be ignored.

Because the hybrid designs have a reduced cost compared to the (15,4) Minnick networks, they are useful in reducing the gate count and area wherever 3 gate delay counters do not increase the critical path delay of the overall network, and where the 4 gate delays of the Kautz design would do so. The cost of each of the counters designs will be evaluated and compared in the next section.

4.4. Cost Evaluation and Comparison of Counter Schemes

Table 1 compares the various counter designs for up to 15 inputs in terms of gate count, sum of non-input weights and the maximum sum of weights per gate.

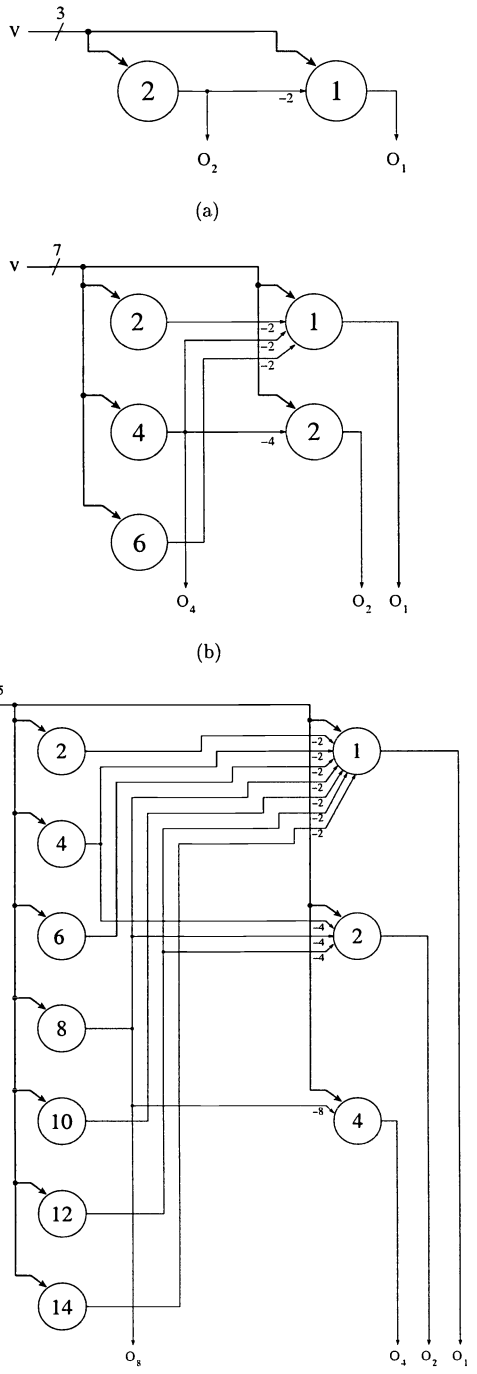
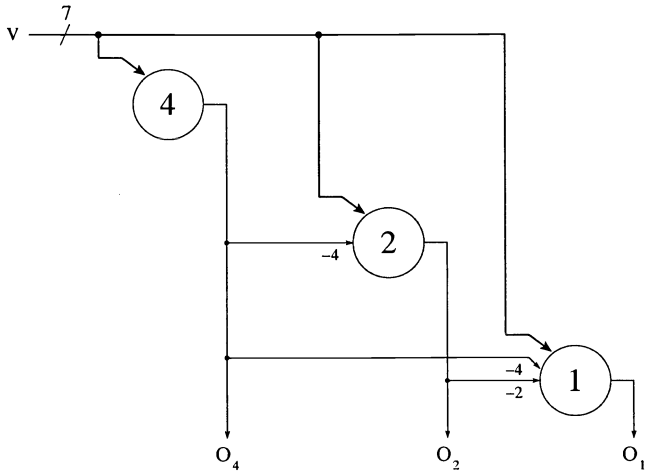
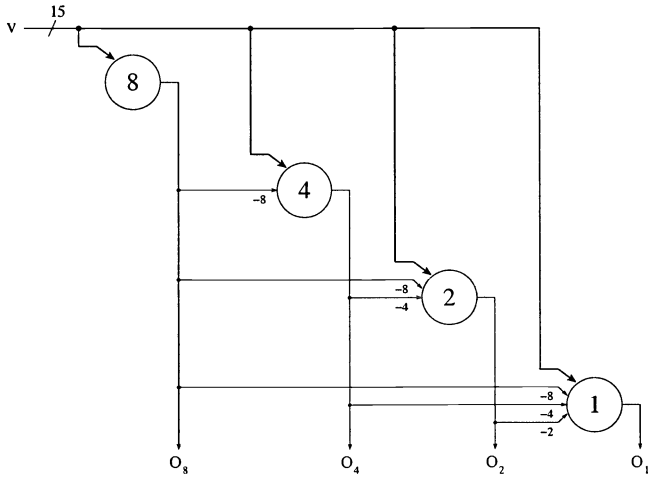


Figure 3. Minnick counter networks. (a) Depth 2 (3,2) counter with [1,2] delay profile, (b) depth 2 (7,3) counter with [1,2,2] delay profile and (c) depth 2 (15,4) counter with [1,2,2,2] delay profile.

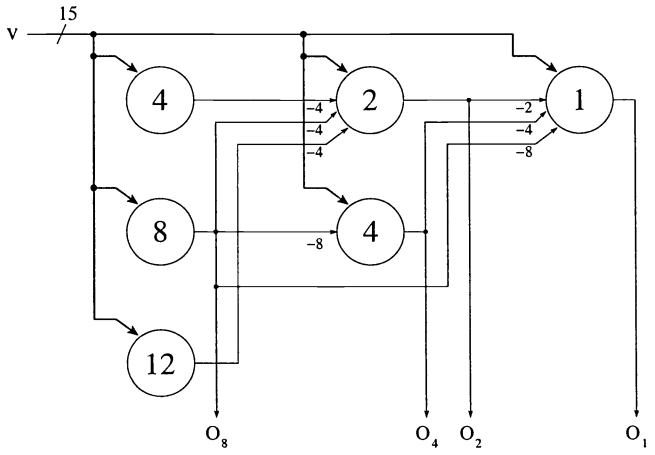


(a)

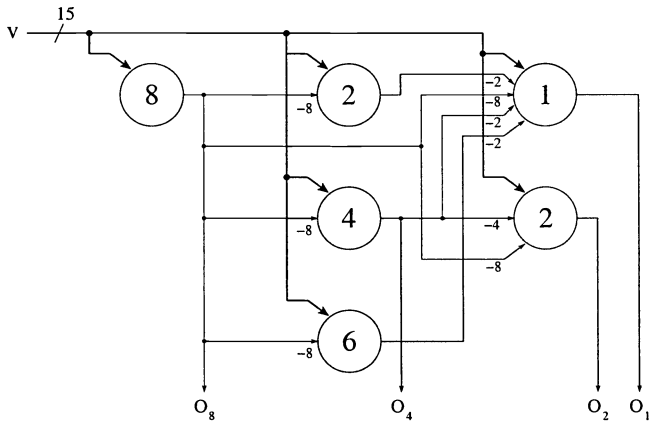


(b)

Figure 4. Kautz counter networks. (a) Depth 3 (7,3) counter with [1,2,3] delay profile and (b) depth 4 (15,4) counter with [1,2,3,4] delay profile.



(a)



(b)

Figure 5. Hybrid counter networks. (a) Depth 3 (15,4) counter with [1,2,2,3] delay profile and (b) depth 3 (15,4) counter with [1,2,3,3] delay profile.

In order to determine the area cost of each of the counter networks, it is necessary to define a cost function which accounts for interconnect, weight area and the area of the sense amplifier component of each TL gate area. The area of the sense amplifier component will be referred to as the *gate area*, and does not include the area required to implement the weights. Unlike conventional AND-OR-NOT logic, total TL circuit size is not easily related to the number of gates as, in addition to interconnect, the area of TL circuits depends on the number of inputs as well as the weight size. The contribution of each component to the total cost depends on the CMOS technology and the gate design. Given these considerations, a reasonable definition of a suitable cost function, C, is a follows:

$$C = \sum gates \times g + \sum w_{I/P} \times w_1 + \sum w_{nonI/P} \times w_2, \quad (3)$$

where g, w_1 and w_2 are technology and circuit dependent parameters which account for the relative contribution of the gates, input weights, hidden layer weights and interconnect, respectively. The factor $w_{I/P}$ refers to the number of input layer unit weights and $w_{nonI/P}$ refers to number of hidden layer unit weights. Typically the input and hidden layer weights will contribute equally to area and thus w_1=w_2(=w), which means that a unit input weight occupies the same area as a hidden layer unit weight. However, certain gate designs, especially capacitive differential type gates,[19,20] are particularly well suited to the area efficient implementation of counter networks and allow the input weights to be shared, thus reducing the total cost.

Given the definition in Equation (3), the cost for each counter is evaluated in terms of the g and w parameters in the final column of Table 1. As discussed previously, the parameters g and w and therefore the cost depend on the particular circuit design and technology used in the implementation of the TL gate. For example, in a 0.35μm, 4M-2P STTL[20] implementation without input weight sharing, the area occupied by 10 weight forming capacitors is equal to the area of one gate, so that $g = 10w$. In this case the (15,4) Minnick counter occupies 2.1 times more area than (15,4) Kautz but has 2 gates less worst case delay. Similar calculations can be performed to compare cost for other technologies and implementations.

4.5. Application of Counters to 32-bit Partial Product Reduction

One of the important applications of counters is the reduction of the partial product matrix (PPM) in parallel multipliers. The bits forming the columns of the matrix are the inputs to a succession of counters which reduce the matrix to two rows. These two words are then added using a fast carry-propagate adder to produce the final result of the multiplication. The majority of the delay and area of multipliers is associated with the reduction of the PPM.

One possible reduction of the partial product matrix for a 32×32 direct multiplication of unsigned numbers is shown in Fig. 6. The partial product matrix in the top of the figure consists of 32 rows, each containing 32 bits, and is generated using an array of AND gates.

In the reduction scheme shown, Stage I employs the (15,4), (7,3) counters and full adders to reduce the matrix to that shown in Stage II. Stage II has a maximum column height of 10 bits and is further reduced by using (15,4), (7,3) and (3,2) counters to obtain the matrix of height 4 shown in Stage III. Finally, two rows are obtained by using conventional CMOS (4:2) compressors in Stage III. These two rows are usually the inputs to a fast carry-propagate adder (not shown) which calculates the final product. The reason for using the conventional CMOS compressors is that they will typically achieve the reduction of Stage III in less area than would be possible when using either TL counters or TL compressors.

During the reduction process, not all inputs in each counter are used, and the unused inputs are simply connected to logic 0. It is therefore possible to reduce the number of TL gates in some of the counters in the proposed scheme. For example, when using a (15,4) counter to add a column of 10 bits, the outputs of the input layer gates with thresholds of 12 and 14 will be zero and need not be computed. Furthermore, Kautz counters and the [1,2,2,3] counter can be used off the critical path (there are a number of critical paths) to reduce the total area. The possible area optimizations obtained by removing unnecessary TL gates in such counters, as well as the cost evaluation of such 32-bit multiplication schemes is the subject of ongoing work.

5. CONCLUSIONS

Various threshold logic parallel counter networks for application in 32-bit multiplier design were reviewed and two new (15,4) counters which trade delay for gate count, were proposed. A brief discussion of the cost of the various designs and their application to the design of 32-bit partial product reduction circuits was also presented.

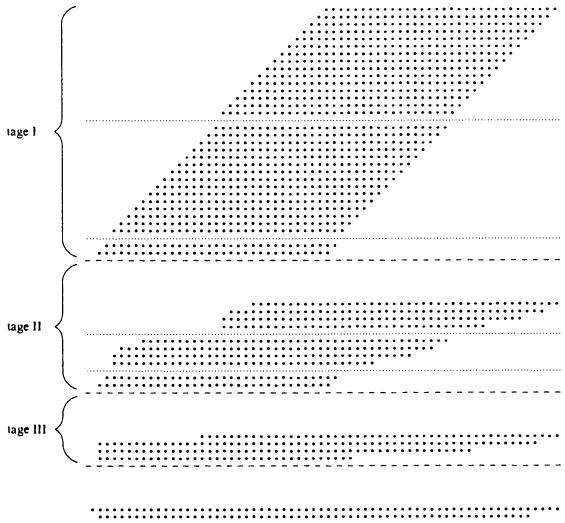


Figure 6. One possible reduction scheme for 32×32 partial product matrix

ACKNOWLEDGMENTS

The support of the Australian Research Council and the Sir Ross and Sir Keith Smith Fund is gratefully acknowledged.

REFERENCES

1. T. Shibata and T. Ohmi, "An intelligent MOS transistor featuring gate-level weighted sum and threshold operations," in *IEDM, Technical Digest*, IEEE, (New York, NY, USA), December 1991.
2. M. Avedillo, J. Quintana, A. Rueda, and E. Jiménez, "Low-power CMOS threshold-logic gate," *IEE Electronics Letters* **31**, pp. 2157–2159, December 1995.
3. H. Özdemir, A. Kepkep, B. Pamir, Y. Leblebici, and U. Çiliniroğlu, "A capacitive threshold-logic gate," *IEEE JSSC* **31**, pp. 1141–1149, August 1996.
4. P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Low power, high speed, charge recycling CMOS threshold logic gate," *IEE Electronics Letters* **37**, pp. 1067–1069, August 2001.
5. J. F. Ramos, J. A. H. Lopez, M. J. Martin, J. C. Tejero, and A. Gago, "A threshold logic gate based on clocked coupled inverters," *International Journal of Electronics* **84**(4), pp. 371–382, 2001.
6. M. Padure, S. Cotofana, and S. Vassiliadis, "A low-power threshold logic family," in *Proc. IEEE International Conference on Electronics, Circuits and Systems*,
7. K. Kotani, T. Shibata, M. Imai, and T. Ohmi, "Clocked-neuron-MOS logic circuits employing auto-threshold-adjustment," in *ISSCC Digest of Technical Papers*, pp. 320–321, 1995.
8. S. Muroga, *Threshold Logic and Its Applications*, Wiley, New York, 1971.
9. L. Dadda, "Some schemes for parallel multipliers," *Alta Freq.* **34**, pp. 349–355, May 1965.
10. L. Dadda, "Composite parallel counters," *IEEE Transactions on Computers* **C-29**, pp. 942–946, October 1980.
11. P. J. Song and G. D. Micheli, "Circuit and architecture tradeoffs for high-speed multiplication," *IEEE Journal of Solid State Circuits* **26**, pp. 1184–1198, September 1991.
12. E. E. Swartzlander, "Parallel counters," *IEEE Transactions on Computers* **C-22**, pp. 1021–1024, 1973.
13. V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," *IEEE Transactions on Computers* **C-45**, pp. 294–305, March 1996.
14. W. H. Kautz, "The realization of symmetric switching functions with linear input logical elements," *IRE Transactions on Electronic Computers* **EC-10**, pp. 371–378, March 1961.
15. R. C. Minnick, "Linear-Input Logic," *IRE Transactions on Electronic Computers* **EC-10**, pp. 6–16, March 1961.

16. S. D. Cotofana and S. Vassiliadis, "Periodic symmetric functions, serial addition and multiplication with neural networks," *IEEE Transactions on Neural Networks* **9**, pp. 1118–1128, November 1998.
17. E. Rodriguez-Villegas, M. Avedillo, J. Quintana, G. Huertas, and A. Rueda, "A νMOS based sorter for arithmetic applications," *VLSI Design* **11**(2), pp. 129–136, 2000.
18. R. Lauwereins and J. Bruck, "Efficient implementation of a neural multiplier," in *Proc. 2nd Intern. Conference on Microelectronics for Neural Networks*, pp. 217–230, October 1991.
19. J. L. Garcia, J. F. Ramos, and A. G. Bohórquez, "A balanced capacitive threshold logic gate," in *Proc. DCICS'2000*, (Montpellier, France), 2000.
20. P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Compact parallel (m,n) counters based on self timed threshold logic," *IEE Electronics Letters* **38**, pp. 633–635, June 2002. To appear in July 2002.