Power-Delay Characteristics of CMOS Multipliers

Thomas K. Callaway
Silicon Graphics, Inc.
Mountain View, CA 94306
tkc@mti.sgi.com

Earl E. Swartzlander, Jr.
Dept. of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712

Abstract

Minimizing the power consumption of circuits is important for a wide variety of applications, both because of increasing levels of integration and the desire for portability. Since multipliers are widely used in computers, it is also important to maximize their speed. Frequently, the compromise between these two conflicting demands is accomplished by minimizing the product of the power dissipation and the delay. This paper reports on the dynamic power dissipation and delay of CMOS implementations of four different multipliers. Simulation was used to establish a set of models for both delay and power dissipation, and those models were then used to compute the power-delay products of the multipliers.

1. Introduction

An important attribute of arithmetic circuits for most applications is maximizing the speed or throughput. For a growing number of applications, minimizing the power consumption is of equal or greater importance. The most direct way to reduce the power is to use CMOS circuits, which generally dissipate less power than their bipolar counterparts. Even for CMOS, the use of adders with minimum power consumption is attractive to increase battery life in portable computers, to avoid local areas of high power dissipation which may cause hot spots, and to reduce the need for a low impedance power and ground distribution network which may interfere with signal interconnections.

In static CMOS the dynamic power dissipation of a circuit depends primarily on the number of logic transitions per unit time [1]. As a result, the average number of logic transitions per multiplication can serve as the basis for comparing the efficiency of a variety of multiplier designs. If two multipliers require roughly the same amount of time and roughly the same number of gates, the circuit which requires fewer logic transitions is more desirable as it will require less dynamic power. However, as process geometries shrink and interconnections dominate over gates, layout considerations may become more influential. This will tend to increase the attractiveness of circuits with a regular structure and localized interconnections, represented in this paper by the modified array multiplier.

Previous attempts to characterize the delay of multiplier circuits include Winograd's analysis of the time required for multiplication [2]. He concludes that the lower bound on the time τ required to multiply an n-bit number by an m-bit number is such that:

$$\tau \geq \lceil \log_r (\lceil \log_d N \rceil + \lceil \log_d M \rceil) \rceil$$

where r is the maximum fanin, d represents the maximum number of states on a given line, $N = 2^n$, and $M = 2^m$. It is interesting to note that the lower bound on the time required is the same for both addition and multiplication.

In addition to the work performed by Winograd, Thompson [3], and Brent and Kung [4] consider the area-time complexity of binary multiplication. Their results serve to indicate that the lower bound for binary multiplication is:

$$AT^{2\alpha} = \Omega(n^{1+\alpha}) \text{ for } \alpha \in [0, 1]$$

Unfortunately, these lower bounds are strictly theoretical in nature and do not provide any information about the delay of a given multiplier circuit.

Previous attempts to estimate energy consumption (dissipation) for VLSI circuits have included attempts to estimate the worst-case energy consumption for general circuits. Kissin [5] calculated worst-case upper and lower bounds of acyclic circuits built out of inverters and 2-input AND and OR gates, where power is consumed by the wires connecting gates as well as the gates themselves. Cirit [6] attempts to measure the average power dissipation of CMOS circuits under a wide range of operating conditions by using statistical methods to calculate the probability that a gate will switch states. Jagau [7] attempts to find the worst-case power dissipation of static CMOS circuits by combining a logic simulator with results from the analog simulation of two switching reference gates. Devadas, *et al.* [8] have attempted to estimate the maximum power dissipation of CMOS circuits using boolean function manipulation combined with a simple gate model.

Baugh and Wooley [9] examined the power dissipation of array and serial-parallel multipliers, and concluded that there is little difference in the power dissipation of the two designs. Powell and Chau [10] introduce the Power Factor Approximation technique. This technique estimates the power dissipation of multiplier circuits using the following equation:

$$P_{\text{multiplier}} \approx K_{\text{multiplier}} Q^2 f_{\text{multiplier}}$$

Where

$$\begin{aligned} Q &\equiv \text{the number of bits per input} \\ K_{\text{circuit}} &\equiv \text{the circuit PFA constant} \\ f_{\text{circuit}} &\equiv \text{the frequency of operation} \end{aligned}$$

More recently, Callaway and Swartzlander have investigated the power-delay characteristics of 16-bit CMOS adders [11]. Nagendra, Owens, and Irwin [12] extend the work reported in [11], and investigate the power-delay characteristics of signed-digit, carry lookahead, and ripple carry CMOS adders. Still more recently, Callaway investigated a set of models for the delay and power dissipation of CMOS adders and multipliers [13].

The results presented in this paper were obtained from the models developed in [13]. The delay models are based on the number of full adders in the reduction stage of the multiplier, followed by a model for the delay of the fast adder at the end. The power dissipation models are based on montecarlo simulations of the multipliers, in which the power dissipation of each individual gate is based on a parameterized model. The parameters of the gate power dissipation model are: the type of gate, the fanin, the fanout, the size of the transistors in the gate, and the direction of the output change.

2. Types of Multipliers

The following types of multipliers are investigated:

- Modified Array [14]
- Split Array [15]
- Wallace Tree [16]
- Booth Recoded Wallace Tree [17]

In general, multiplication can be viewed as repeated shifts and adds. Multiplication can be implemented very easily and simply using only an adder, a shift register, and a small amount of control logic. The advantage of this approach is that it is small. The obvious disadvantage is that it is slow. Not only is it a serial circuit, which results in delays due to latching, but the number of additions it requires is equal to the number of bits in the operands.

One fairly simple improvement to this serial shift-and-add approach is to form the matrix of partial products in parallel, and then use a 2-dimensional array of full adders to sum the rows of partial products. The only difficulty in such an approach is that the matrix of partial products is rhomboidal in shape due to the shifting of the partial products. This can be overcome by simply skewing the matrix into a square and then skewing the propagation of the sum and carry signals of the full adders accordingly. This structure is known as an array multiplier.

The advantages of the array multiplier are that it has a regular structure and a local interconnect; each cell is connected only to its neighbors. This translates into a small, dense layout in actual implementations. The disadvantage is that the worst case delay path of an n by n array multiplier goes from the upper left corner diagonally down to the lower right corner and then across the ripple carry adder. This means that the delay is linearly proportional to the operand size.

One method which can be employed to decrease the delay of an array multiplier is to replace the ripple carry adder at the bottom with a carry lookahead adder. By doing this, the delay can be reduced by roughly 10% for the 8 bit multiplier and 20% for the 32 bit multiplier while increasing the number of gates by about 5% for the 8 bit multiplier and 1% for the 32 bit multiplier. In this paper, such a multiplier is referred to as a modified array multiplier.

The area and the power dissipation of the two types of array multipliers do not differ by much. The carry lookahead adder has roughly the same power dissipation as the ripple carry adder it replaces, and only increases the gate count very slightly, while reducing the delay quite a bit. Because of this combination, only the modified array multiplier is discussed.

Wallace [16] showed that it was possible to improve the delay to logarithmic time. He noted that by using a pseudo-adder (a row of full adders with no carry chain), it is possible to sum three operands into a two operand result with only a single full adder delay. The two resulting words are added together using a carry propagate adder, and the result is the sum of the three initial operands. He proposed using pseudo-adders repeatedly in a tree structure for summing the partial products into 2 larger partial products, and then using a fast carry propagate adder (CPA) to sum them and produce the product.

The method proposed by Wallace has a delay proportional to $\log n$ for an n by n multiplier, making it faster than the array multiplier. It requires slightly more gates, but it's main disadvantage is a fairly irregular structure. This means that it requires a larger area than the array multiplier and is much more difficult to implement in VLSI.

Another method for decreasing multiplication delays is the use of modified Booth recoding [17]. Modified Booth recoding is used to reduce the height of the initial partial products matrix [18]. This matrix can then be reduced using either an array of full adders or a Wallace-type reduction tree. In the multiplier considered here, a radix-4 modified Booth recoding is performed on the original n-bit inputs. This recoding reduces the number of partial products from n to $\frac{n}{2} + 1$. Those partial products are then reduced to 2 using a Wallace tree, and the final result is produced by a carry lookahead adder.

One method for combining the increased regularity of the array multiplier with the smaller delay of the tree multipliers is to split the array into two interleaved sections [15]. The result of the split array is two numbers in carry-save notation, one for the even rows, and one for the odd rows. These two numbers are then combined using two rows of 3:2 counters in a very small Wallace tree, and the final product is generated using a carry lookahead adder.

3. Delay Models

This section discusses methods for modeling the worst-case delay of parallel multipliers. The assumption is that these multipliers will be implemented in synchronous systems, so only the worst-case delay is of interest. Again, the goal of the delay model is to accurately predict the worst-case delay of a given multiplier circuit.

In order to verify the delay models discussed in this section, 16-bit versions of each multiplier were designed and laid out in a 2-micron, 2-level metal CMOS technology. Netlists were extracted from the layout, and a SPICE-like circuit simulator was used to simulate the worst-case delay. These delays are shown in Table 1.

Table 1. Simulated 16-Bit Multiplier Delay (nsec).

Multiplier Type	Delay (nsec)
Modified Array	93
Split Array	63
Wallace	54
Modified Booth	45

A very quick and fairly accurate method for estimating the delay of multipliers is presented in this section. The basis for this estimate is finding the worst-case delay for a full adder, and then multiplying that delay by the number of full adders in the worst-case signal path. The only other significant source of delay is the fast carry propagate adder at the bottom, whose delay can be estimated using parameterized gate delay models, as discussed in [11] and [13].

The first thing to examine in this model is the worst-case delay of a single full adder. Based on circuit simulations, the full adder has a worst-case delay of 4.2 nanoseconds. There are 2 other sources of delay in the multipliers. The first source of delay is the partial product generation, which is done either with INVERT and 2-input NOR gates, or with modified Booth recoders. The other source of delay is the final carry lookahead adder. Table 2 shows the number of full adders in the worst-case path for each of the multipliers.

Table 2. Multiplier Full Adder Delays.

Multiplier Type	Multiplier Size (bits)		
	8	16	32
Modified Array	7	15	31
Split Array	5	9	17
Wallace	4	6	8
Modified Booth	3	4	6

The delay for the partial product generation is 2, 4, or 6 nsec depending on the word size of the multiplier. A modified Booth recoder has a worst-case delay of 7.2 nsec, and the delays through the final carry lookahead adders are estimated using a parameterized gate delay model, in which the delay through each gate is a function of the gate type, fanin, fanout, transistor sizes, and direction of output change. This model can also be used for the multipliers, but the full adder model is simpler and just as accurate.

The delay for a multiplier is estimated by combining the number of full adders in the critical path times the full adder delay, plus the delay for partial product generation, plus the delay for the carry lookahead adder. Estimates for the worst-case delay of the various multipliers, along with the number of full adders in the critical path, are shown in Table 3.

Based on the accuracy of the model for 16-bit multipliers, delays are estimated for 8-bit and 32-bit multipliers as well. These estimated delays, along with the 16-bit multiplier estimated delays are presented in Table 4. There is not much difference in the delay of the 8-bit multipliers, indicating that array-type multipliers represent a viable choice for small word-size multipliers. However, for 32-bit word sizes, the differences are quite large. The delay of the modified Booth recoded Wallace tree multiplier is approximately one-third that of the modified array multiplier.

Examining Table 4 reveals that the delay for the modified and split array multipliers increases linearly with the word size. The delay for the Wallace and modified Booth multipliers increases as the log of the word size, indicating that the modified Booth multiplier will continue to be the fastest for word sizes larger than 32 bits.

Table 3. Estimated 16-Bit Multiplier Delay (nsec).

Multiplier Type	Estimated Delay (nsec)	Simulated Delay (nsec)	Adder Delays	Error (%)
Modified Array	84	93	15	-10
Split Array	64	63	9	2
Wallace	52	54	6	-4
Modified Booth	46	45	4	2

Table 4. Estimated Multiplier Delay (nsec).

Multiplier Type	Multiplier Size (bits)		
	8	16	32
Modified Array	44	84	155
Split Array	37	64	98
Wallace	36	52	61
Modified Booth	35	46	56

4. Power Models

This section is concerned with the average power dissipation of static CMOS multipliers. In the previous section, the focus was on the worst-case delay. In contrast to this focus on worst-case behavior, we are interested here in the average power dissipation. The peak instantaneous power dissipation determines things such as the size and routing of power and ground planes, but the average power dissipation determines cooling requirements and the expected battery life.

As with the delay models, the power dissipation models are verified by comparison with circuit simulations. Netlists extracted from layouts of the 16-bit multipliers are subjected to 1,000 pseudo-random inputs, and the average power dissipation per multiplication is computed on the basis of those simulations.

The basis for the multiplier power dissipation model is a model for the power dissipation of a single switching gate. The parameters for the power dissipation of a switching gate are the type of gate, fanin, fanout, transistor sizes, and direction of the output change. Of these five parameters, the direction of the output change is by far the most important. However, the other parameters must also be considered in order to generate an accurate model for the gates, as Figure 1 shows.

The parameters are calculated by running SPICE simulations on each gate for a variety of parameter values, and then performing a linear regression to develop a single model for the power dissipation of each type of gate. This model is then used in conjunction with an event-driven gate-level simulator and a gate-level description of the multiplier circuit. One thousand pseudo-random inputs are presented to each circuit, and the average power dissipation is estimated on the basis of those simulations. Figure 2 shows the average power dissipation as a function of time for each 16-bit multiplier as simulated with a circuit simulator.

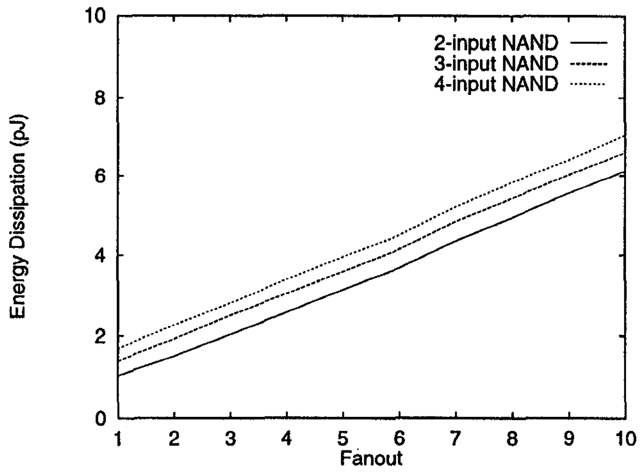

Figure 1. Energy Dissipated by a Switching NAND Gate.

Table 5 shows both the estimated and simulated average power dissipation of the four 16-bit multipliers. The simulated average power dissipation for the 16-bit multipliers is obtained through circuit simulations. The power dissipation is computed on the basis of a 100 nanosecond clock period, which is roughly the time required for the slowest multiplier to finish evaluating in the worst case.

Table 5. Estimated 16-Bit Multiplier Power Dissipation (mWatt).

Multiplier Type	Estimated (mWatt)	Simulated (mWatt)	Error (%)
Modified Array	47	44	7
Split Array	40	38	5
Wallace	31	32	-3
Modified Booth	44	41	7

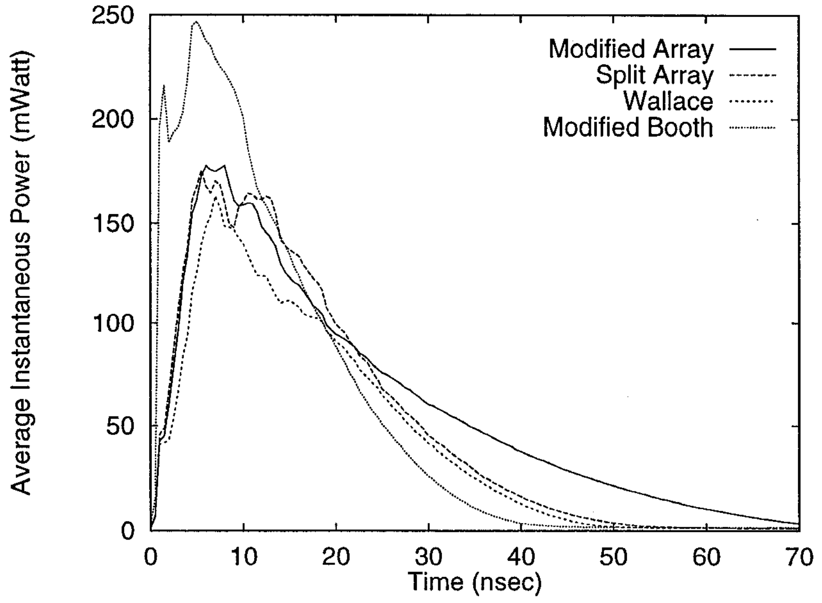

Figure 2. Simulated Power Dissipation For 16-Bit Multipliers.

As the power dissipation model has proven to be quite accurate for the 16-bit multipliers, estimates were made for the power dissipation of 8-bit and 32-bit multipliers as well. Table 6 gives the estimated average power dissipation for 8-, 16-, and 32-bit versions of each multiplier. The clock period used to calculate the average power dissipation is 50, 100, and 150 nanoseconds, respectively. Although the worst-case delay of the 32-bit array multiplier is 155 nanoseconds, this represents only a very small anomaly, as the pattern resulting in the worst-case delay is exceedingly unlikely to occur. The word uniform in the table heading refers to the standard clock period for each word size which is used to evaluate the power dissipation of the multipliers.

Table 6. Estimated Multiplier Uniform Average Power Dissipation (mWatt).

Multiplier Type	Multiplier Size (bits)		
	8	16	32
Modified Array	14	47	275
Split Array	14	40	248
Wallace	13	31	128
Modified Booth	25	44	163

A topic of interest concerns the way in which the power dissipation scales with increasing input word sizes. This is examined here for the 16- and 32-bit multipliers, as the overhead for the 8-bit multipliers tends to skew the analysis. The power dissipation of the modified and split array multipliers increases as about the cube of the word size, while the increase for the Wallace and modified Booth multipliers is as roughly the square of the word size. This means that the Wallace multiplier dissipates the least power for input sizes greater than 8 bits. For input sizes of 8 bits or less, the modified array, split array, and Wallace tree multipliers will all have roughly the same power dissipation.

5. Power-Delay Characteristics

Often the most important metric for a circuit is its power-delay product. A lower power-delay product indicates a faster, more efficient design. In the previous section, the power dissipation for each multiplier of a given size was computed using a clock period suitable for the slowest multiplier in the group. Based on those results, the Wallace multiplier was seen to have the lowest average power dissipation. However, because the Wallace multiplier has a much lower delay than the modified array, it is likely to be used in a system where the clock period is shorter.

In other words, using the same clock to compare the power dissipation of the four different multipliers may not reflect how the multipliers are actually used in a system. Perhaps a more accurate way to measure the power dissipation for the multipliers is to use the worst-case delay as the clock period. In that case, the average power dissipation values for the multiplier circuits are given in Table 7.

Table 7 shows that the modified array, split array, and Wallace tree multipliers all have roughly the same power dissipation when used at their maximum frequency. However, the modified Booth recoded Wallace tree multiplier has

Table 7. Estimated Multiplier Worst-Case Average Power Dissipation (mWatt).

Multiplier Type	Multiplier Size (bits)		
	8	16	32
Modified Array	16	56	275
Split Array	18	62	373
Wallace	18	60	315
Modified Booth	36	95	437

a significantly higher power dissipation than the other three. This is due in part to its shorter clock period, and in part to the fact that it is a less efficient method than the others. Now, using those values for the worst-case average power dissipation and the values for the worst-case delay, we can compute the power-delay product for each of the multipliers. This power-delay product is shown in Table 8.

Table 8. Estimated Multiplier Power-Delay Product (nsec×Watt).

Multiplier Type	Multiplier Size (bits)		
	8	16	32
Modified Array	0.70	4.70	42.6
Split Array	0.67	3.97	36.6
Wallace	0.65	3.12	19.2
Modified Booth	1.26	4.37	24.5

For 8-bit and smaller multipliers, modified Booth recoding does not represent an efficient approach. However, for 16-bit and larger multipliers, modified Booth recoding does look attractive. It represents the fastest multiplier of the four, and has a power-delay product less than the modified array. However, the most efficient multiplier in terms of the power-delay product is the Wallace tree multiplier.

6. Future Considerations

As transistor sizes continue to decrease, interconnect technology has failed to keep pace with increasingly smaller and faster gates. Because of this, wires can no longer be treated as simple zero-delay electrical connections between transistors. In the past this effect has been limited to global signals, which must traverse large portions of the chip. However, this effect is now being felt even in intra-block lines.

As the interconnection of transistors becomes as much or more of a performance limiter than the transistors themselves, the focus will shift from trying to reduce the number of transistors or gates in the critical path to limiting the amount of wiring in the critical path. In such a scenario, highly repetitive and locally connected structures such as the modified array and split array multipliers will tend to become more promising than they are at present.

Another trend which encourages the return to small, simple structures is the continuing decrease of transistor threshold voltages. As power supply voltages have decreased, threshold voltages have been lowered as well. Because a MOS transistor's leakage current increases exponentially as the threshold voltage decreases, this trend increases the quiescent or DC power dissipation of static CMOS structures. One method for countering this increase in DC power is to use an absolute minimum of transistors, again favoring small, simple structures.

7. Conclusions

In this paper, we have examined the power dissipation and delay of four different multipliers and 3 different word sizes. The delay and power dissipation were modeled using two relatively simple models which are quite accurate.

We have shown that for small multipliers, there is little difference in either the delay or power dissipation for the multipliers, except for the modified Booth multiplier, which has a significantly larger power dissipation than the other three. Also, we have shown that for 16-bit and larger multipliers, there do exist significant differences in both the worst-case delay and the average power dissipation.

Examining the power-delay products of the multipliers indicates that the Wallace multiplier is the most efficient multiplier for word sizes of 8 through 32 bits.

References

[1] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, MA: Addison-Wesley Publishing Company, 1988.

[2] S. Winograd, "On the time required to perform multiplication," *Journal of the ACM*, vol. 14, pp. 793–802, 1967.

[3] C. D. Thompson, "Area-time complexity for VLSI," *Proceedings of the 11th Annual ACM Symposium on the Theory of Computation*, pp. 81–88, 1979.

[4] R. P. Brent and H. T. Kung, "The chip complexity of binary arithmetic," *Proceedings of the 12th Annual ACM Symposium on the Theory of Computation*, 1980.

[5] G. Kissin, "Measuring energy consumption in VLSI circuits: A foundation," *Proceedings of the 14th ACM Symposium on the Theory of Computing*, pp. 99–104, 1982.

[6] M. Cirit, "Estimating dynamic power consumption of CMOS circuits," *ICCAD*, pp. 534–537, 1987.

[7] U. Jagau, "SIMCURRENT - an efficient program for the estimation of the current flow of complex CMOS circuits," *DAC*, pp. 396–399, 1990.

[8] S. Devadas, K. Keutzer, and J. White, "Estimation of power dissipation in CMOS combinational circuits using boolean function manipulation," *IEEE Transactions on CAD*, vol. 11, pp. 373–380, 1992.

[9] C. R. Baugh and B. A. Wooley, "Digital multiplier power estimates," *IEEE Transactions on Communications*, pp. 1287–1288, 1975.

[10] P. M. Chau and S. R. Powell, "Estimating power dissipation of VLSI signal processing chips: The PFA technique," *VLSI Signal Processing, IV*, pp. 250–259, 1990.

[11] T. K. Callaway and E. E. Swartzlander, Jr., "Estimating the power consumption of CMOS adders," *Proceedings of the 11th Symposium on Computer Arithmetic*, pp. 210–216, 1993.

[12] C. Nagendra, R. M. Owens, and M. J. Irwin, "Power-delay characteristics of CMOS adders," *IEEE Transactions on VLSI Systems*, vol. 2, pp. 377–381, 1994.

[13] T. K. Callaway, *Area, Delay, and Power Modeling of CMOS Adders and Multipliers*. Ph. D. dissertation, The University of Texas at Austin, 1996.

[14] A. D. Pezaris, "A 40ns 17-bit by 17-bit array multiplier," *IEEE Transactions on Computers*, vol. C-20, pp. 442–447, 1971.

[15] J. Iwamura, K. Suganuma, M. Kimura, and S. Taguchi, "A CMOS/SOS multiplier," *Proceedings of the ISSCC*, pp. 92–93, 1984.

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

[17] O. L. MacSorley, "High-speed arithmetic in binary computers," *IRE Proceedings*, vol. 49, pp. 67–91, 1961.

[18] I. Koren, *Computer Arithmetic Algorithms*. Englewood Cliffs, NJ: Prentice Hall, 1993.