Multipliers and Datapaths

Hesham Al-Twaijry
Michael Flynn

Technical Report : CSL-TR-94-654

December 1994

This work was supported using facilities supported by NASA contract NAG2-842 and a fellowship from Saudi Arabia.

Multipliers and Datapaths

by

Hesham Al-Twaijry

Michael Flynn

Technical Report : CSL-TR-94-654

December 1994

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science

Stanford University

Stanford, California 94305-4055

Abstract

People traditionally have considered the number of counters in the critical path as the metric for the performance of a multiplier. This report presents the view that tree topologies which have the least number of levels do not always give the fastest possible multiplier when constrained to be part of a microprocessor. It proposes two new topologies: hybrid structure and higher order arrays which are faster than conventional tree topologies for typical datapaths.

Key Words and Phrases: Floating-Point, Multiplication, Datapaths

Contents

List of Figures

List of Tables

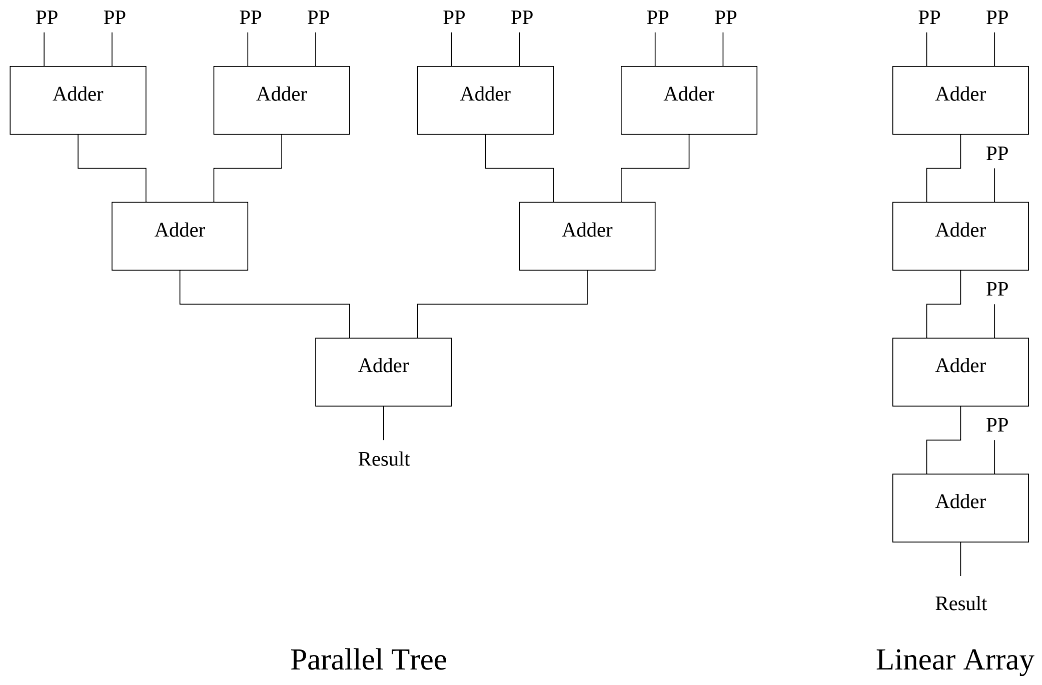


Figure 1: Multiplication Structures

1 Introduction

Multiplication is one of the most common arithmetic operations. In fact 8.72 % of all instructions in typical scientific programs are multiplies [1]. Also, multiplication is a long latency operation. In typical processors multiplication takes between 2 and 8 cycles. Consequently, having high speed multipliers are critical for the performance of processors.

When a multiplication implementation is mapped into a 2D package the resulting implementation's shape (flow of communication) is a parallelogram. This shape occurs because of the need to shift between the partial product rows to allow for the different arithmetic weights. Adding the counters that are needed to add the partial products, only exaggerates this shape even more, making it oval shaped.

Traditionally in order to achieve high performance multipliers, parallel addition of the partial products is used. These types of structures are called parallel tree structures as shown in figure 1. Another method used to reduce the partial products is to add the partial products serially, creating a linear array, as also shown in figure 1. Although both of these structures require the same amount of hardware, the tree structure requires more complex interconnections, and has fewer stages of delay.

This study investigates the relationship between the topology of the partial product interconnections and possible circuit implementations. It also studies the effect of these topologies on the latency of the multiplier, when the multiplier is part of a larger system.

The remainder of this paper is organized as follows. Section 2 lays the foundation by describing the background to the problem. Section 3 presents the layout of the multiplier.

Sign Bit (1)	Normalized Fraction (52)	Biased Exponent (11)

Figure 2: IEEE Double Precision Format

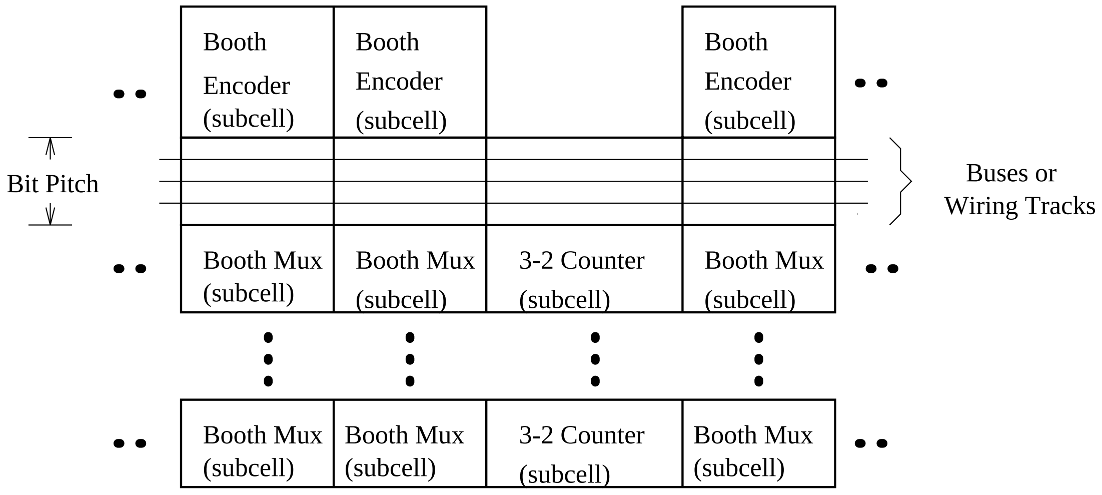


Figure 3: Datapath Structure

Section 4 gives several topologies that are used. It also introduces two new topologies. Section 5 gives some information about the circuits used. The simulation methodology is presented in section 6, followed by section 7 where the results are presented.

2 Background

The datapath is part of an arithmetic processor which uses the IEEE floating point arithmetic standard [2]. The format for double precision numbers is shown in figure 2. The standard defines numbers in a sign-magnitude, normalized format. The standard has a normalized significand, that is the most significant bit of the fraction is always 1, which is not stored. The significand effectively becomes 53 bits because there is no need to save the hidden "1" bit. To achieve the rounding accuracy defined by the standard, the full 106 bit result has to be calculated, even though almost half of it is used only for rounding.

To reduce the number of partial products that need to be summed, the Modified Booth Algorithm [3] is used. In this algorithm, the multiplier is partitioned into overlapping groups of 3 bits. Each group is decoded in parallel to select a partial product row. Using this algorithm the number of partial product rows is 27.

Tree topologies give structures that have a small number of delay stages at the expense of complex interconnections. However, when the multiplier is built as part of a bus-oriented CMOS datapath, tree structures do not necessarily give the best performance. A bus-oriented datapath is a highly structured design in which the data and control buses flow right through the subcells of the design as shown in figure 3. The fact that the multiplier is part of a datapath forces that the width of the subcells or *bit pitches* to be constant. Since the bit pitches are constant, one can reuse each subcell. The required structure is achieved by varying the interconnection network of the adders. Several bit pitches were selected to study the effect of the number of wiring tracks available per bit pitch. Each bit pitch is capable of supporting a different number of wiring tracks. Based on the needs of an interconnection scheme for wiring tracks, one uses either circuits that need single-ended or complementary signals.

In typical datapath designs the number of buses or tracks commonly available is around fourteen per bit pitch. Only ten of these tracks are commonly available for routing. The other four tracks are used for the routing of the two operands, result, power, and ground buses. The power and ground can be designed such that they use a single bus by mirroring[1]. These ten tracks are used to route the interconnections between the counters, in addition to the routing of the Booth muxes outputs and inputs. Therefore, since tree structures have complex interconnections they can not use complementary signals.

3 Layout

When a multipliers' parallelogram shape is laid out on a plane in silicon it is inefficient. It is wasteful because it leaves a lot of unusable silicon at the sides. Even though some hardware for the rounding and control can be hidden in the unused space. Even if there was no wasted silicon the layout would have a poor aspect ratio (ratio of height to width) since the parallelogram is 106 bits wide and the datapath is only 53 bits wide. So to improve the layout, multipliers are folded. Figure 4 gives the layout structure for the multiplier when it is folded by the maximum amount possible. It is also possible to align the partial products vertically and cause the equal weights to be shifted diagonally as shown in figure 4. Aligning is not feasible if trees are used, since aligning the partial products increases the difficulty in wiring the counters. The difficulty arises because adjacent weight's columns trees have different interconnections, so routing the different weights in each bit pitch increases the number of wiring tracks needed dramatically. Aligning the partial products is commonly used for arrays since the interconnections of adjacent weights are identical.

The general layout structure for the multiplier is given by figure 5. The multiplier operand (operand 2) is routed to the Booth encoders. The output of the Booth encoders is then routed to the Booth muxes. The counter tree/array reduces the partial products into two 106 bit operands that are stored in the latches that precedes the CPA (Carry Propagate Adder). These latches are folded, and their output drives the CPA and rounding logic that produces the final result.

[1]Mirroring is the circuit layout scheme,in which one places the power and ground buses at the edges of the subcell. The subcells are then mirrored, so that one can place the power and ground lines of adjacent cells on top of one another.

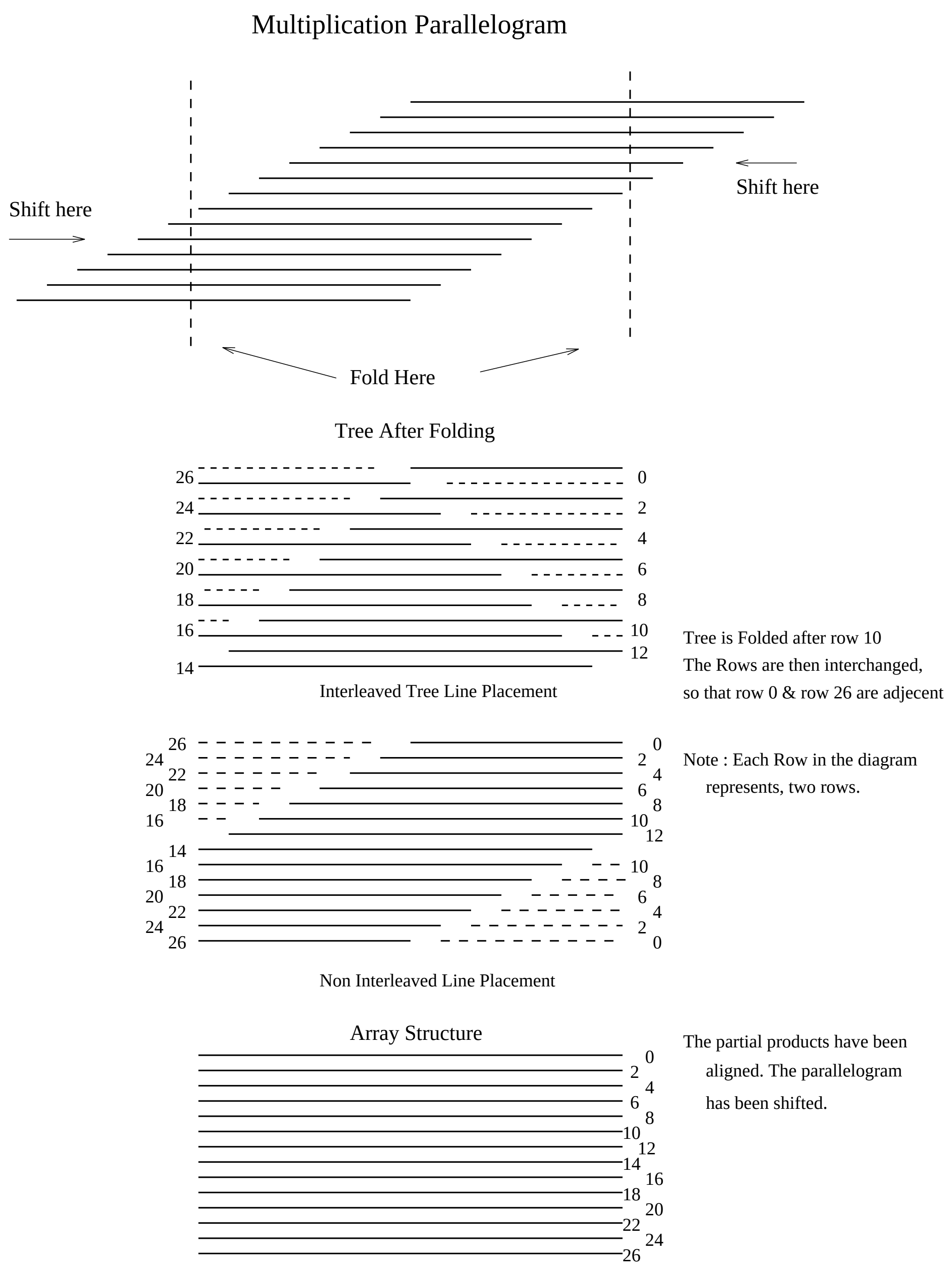

Multiplication Parallelogram
Shift here
Shift here
Fold Here
Tree After Folding
Interleaved Tree Line Placement
Tree is Folded after row 10
The Rows are then interchanged,
so that row 0 & row 26 are adjecent
Note : Each Row in the diagram
represents, two rows.
Non Interleaved Line Placement
Array Structure
The partial products have been
aligned. The parallelogram
has been shifted.

Figure 4: Multiplication Parallelogram

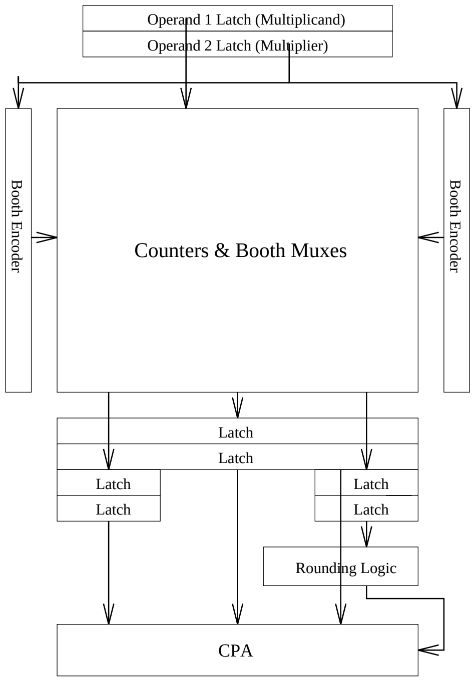


Figure 5: General Layout

4 Topology

The topologies that are used to reduce the partial products that were compared spanned the spectrum from almost directly linear arrays to several tree types as shown in figure 6, and they include:

1. *Double Linear Arrays in Parallel:* This is the simplest circuit [4]. It consists of two linear arrays in parallel. The first array sums the odd numbered partial products, while the second array sums the even partial products.

   This multiplier has a very regular structure, which avoids the folded parallelogram scheme. The partial products are aligned and the sums flow diagonally.

   The array structure allows for multipliers that are very easily pipelined. These pipelined multipliers could also be designed to produce the result iteratively. Pipelining is accomplished by adding a latch at the output of each counter, whenever a stage is required.

2. *A 4-2 counter tree:* The 4-2 counter is constructed from two 3-2 counters as shown in figure 7. The 4-2 counter is symmetric, in that it has a 2 : 1 reduction ratio, while the 3-2 counter is not symmetric.

   The 4-2 counter tree [5] has a regular and symmetric structure. 4-2 trees are commonly used in pipelined, and iterative multipliers. The symmetric nature of the 4-2 counter facilitates the addition of latches that are needed for pipelining after each 4-2 counter.

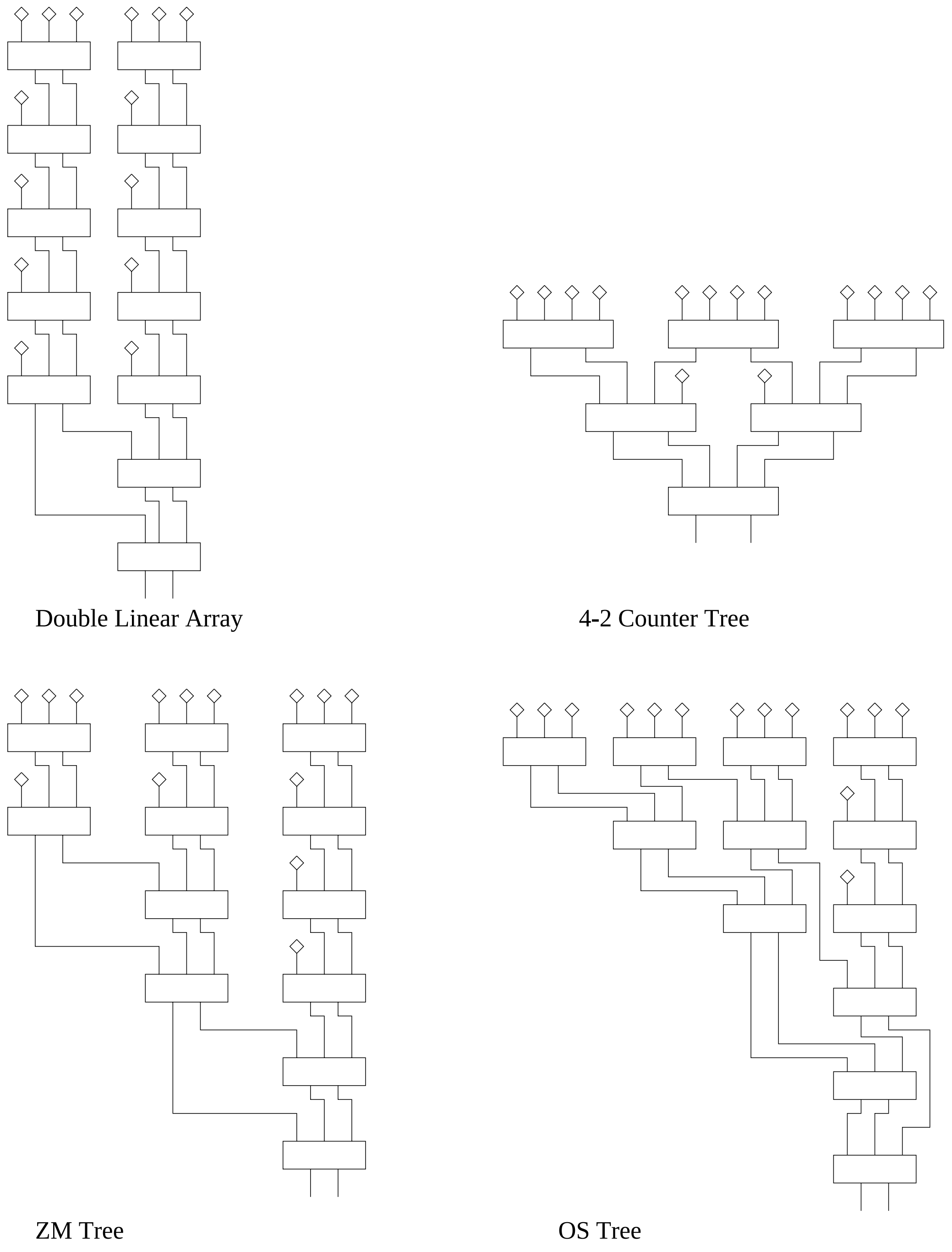


Figure 6: Structures for Comparison

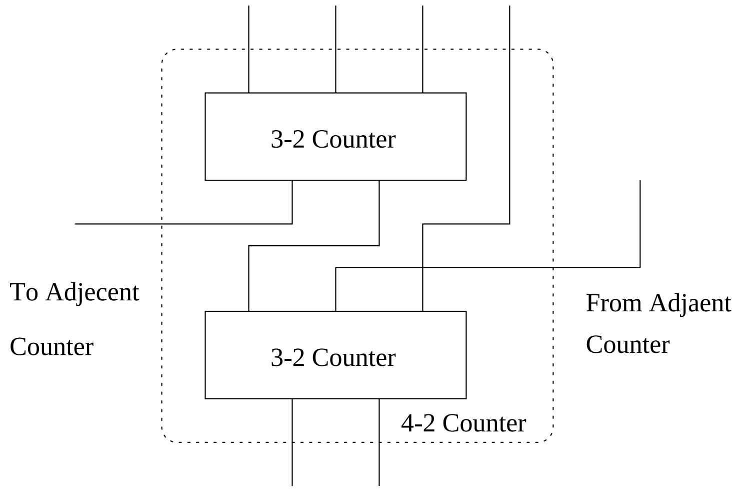


Figure 7: 4-2 Counter Structure

Iterative and pipelined 4-2 counter trees use the same structure for each bit pitch. Also, the multiplier is not folded, since the iterative tree only operates on small horizontal portions of the parallelogram consecutively, so the width increase of the multiplier is small. Both of these factors lead to an extremely regular structure.

When the 4-2 counter tree is used in a non-iterative multiplier folding is required. In addition different weights in the multiplier require slightly different trees, even though zeros are inserted in the empty slots in the counters. Folding and the different tree structures both contribute to produce a structure that is less regular than the iterative trees.

The 4-2 counters do not have a constant requirement for wiring tracks. The number of wiring tracks increases by two when the number of partial products is doubled. Their wiring requirement is similar to that of Wallace trees [7]. However the growth rate of wiring tracks in 4-2 trees is smaller. Also, their wiring requirement is more regular, since Wallace trees which use 3-2 counters are extremely irregular making them notoriously difficult to layout.

The advantage of the binary tree reduction of the 4-2 trees is not all that significant for IEEE double precision numbers since the significand size is not a power of two.

3. *A ZM tree:* This is the Balanced Delay Tree proposed by Zuras and McAllister [6]. The ZM tree is based upon the idea of balanced delay chains of counters. Trees are constructed by combining progressively longer serial chains into serial chains below them. The connection between the two chains is made when the total delay of the upper chain is equal to the delay of the lower chain. That is the connection is made when the number of counters in the critical path of upper chain of counters is as long as the delay of the critical path of the chain of lower counters. This method builds ZM trees of type one, which require only two tracks to feed the output of one counter to the input of a non-adjacent counter.

This tree structure has a very regular layout and it requires only a few primitive cells. This type of tree generally uses more levels of counter delay than the Wallace tree [7] gives, for most values of partial products that must be summed. To reduce the number of levels, ZM trees are connected in parallel to produce higher order ZM trees of type two, etc. These higher order trees, which require a larger number of tracks, are constructed by connecting type one ZM trees in parallel using 4-2 compactors. These higher order trees are less regular and require more wiring tracks.

ZM trees are not easily pipelined. The pipelining of a ZM tree requires that the outputs of the Booth muxes that are not at the first level, ie. those Booth muxes whose output is after the first latch, must be latched in addition to the outputs of the 3-2 counters. So the number of latches required is greater than the number of latches in a 4-2 counter tree. ZM trees can be built to produce the result iteratively using structure that is similar to 4-2 tree.

4. *An OS tree:* This is the Overturned Stairs Tree that was proposed by Mou and Jutand [8]. This method divides a tree into a body and a root. The root is the last 3-2 counter in the tree. The body is constructed recursively. In that a body of height k, where k is the number of 3-2 counters in the critical path, is constructed from a body of height k-1 and a linear array of height k-2. The linear array and the body are joined using a 5-3 counter. The 5-3 counter is constructed from two 3-2 counters in series. This method build OS trees of type one. This tree structure requires a few primitive cells. It requires 3 tracks to route signals between non-adjacent counters. The OS tree uses more wiring tracks than the ZM tree. The OS tree needs more primitive cells, and it has a less regular structure, compared to the ZM tree.

   OS tree structure can give the optimal (minimum) number of counter levels for most numbers of partial products. However, to achieve this , one has to use higher order OS trees. Higher order OS trees can be built by replacing the linear arrays with OS trees of type one. The higher order trees require more wiring tracks. Increasing the order of the OS by one increases the number of tracks by three.

   OS trees are not easily pipelined. The pipelining of a OS tree requires that the outputs of the Booth muxes that are not at the first level, ie. those Booth muxes whose output is added to the outputs of the first level counters, must be latched in addition to the outputs of the 3-2 counters. So the number of latches required is greater than the number of latches in a 4-2 counter tree. OS trees can be built to produce the result iteratively using structure that is similar to ZM tree. However, OS trees are not typically used for iterative multipliers, since 4-2 trees give a more regular topology, that uses the same number of counter levels.

5. *A Hybrid Solution.* The Hybrid solution is a non-interleaved solution that tries to make the best use of the tracks available. The unfolded regions of the parallelogram uses a ZM/OS tree of type one. While the folded regions upper part is a ZM/OS tree of type one and the lower part is a linear array. This type of organization allows the circuit to use complementary circuits for pitches that it would not normally be possible. In this solution the critical path is no longer the column with the largest

number of partial products, but is now at the edge of the layout. The critical path becomes the largest linear array. This array occurs at the hinge point where the folding occurred. This circuit has the ability to trade off between the width of the total design and the number of counter levels. So the number of 3-2 counter levels can be decreased by one for the increase of 4 bit pitches. The decision to choose a ZM or OS tree affects the extent of the trade off. In that the OS tree needs more wires but it uses less levels, so one can continue the trade off width for number of counter levels for a larger number of levels.

The hybrid solution can also be pipelined, similarly to the ZM tree. However, this design is primarily proposed to reduce the latency of the multiplier using the smallest number of wiring tracks available. It is not designed for iterative solutions, since iterative solutions sum only a small number of partial products at a time and wiring tracks are not a concern.

6. *Higher Order Arrays:* This is a class of arrays in which the 3-2 counters are designed as several linear array chains. The chains are combined in parallel when the delay of the upper chain is equal to the delay of the lower chain. This class of arrays can in fact be thought of as a collection of ZM trees of type one. The ZM trees have been designed for the column with the largest number of inputs. This design is replicated for all other columns. In this design the non-critical columns are not optimized. This design trades of the performance of the non-critical columns for regularity. The design is very regular which allows the use of the aligned partial product method. The regularity of the higher order tree is proportional to the number of linear arrays that are combined. The smaller the number of arrays the more regular the design.

   Higher order trees can be classified according to the lengths of the chains of partial products before the combining occurs. For example the 6-6-8-8 array has a linear array that combines 6 partial products which is combined with an array the combines 6 partial products. The resulting structure is then combined with an array that combines 8 partial products. Finally the resulting structure is combined with an array that sums 8 partial products.

   Higher order arrays are just as easily pipelined as arrays. However since their design is proposed to reduce the latency of the multiplier using the smallest number of wiring tracks available, pipelined iterative higher order trees are not very attractive.

Because our design is part of a datapath, we would like to keep the number of wiring tracks for each column constant. We can accomplish this for ZM/OS trees by constraining the folded section of the parallelogram to use a ZM or OS tree of type one, for each group of inputs that have to be summed together, while the unfolded part uses a ZM or OS tree of type two. This constraint has no effect when one is using a Booth encoded multiplier for multiplying two IEEE double precision numbers. 4-2 trees can not achieve a constant number of tracks, although the difference is only two tracks between the maximum and minimum number of tracks required.

5 Circuits

Based upon the number of wiring tracks one has available and the interconnection requirements of the structure chosen, one uses either single-ended or complementary signal circuits. Single-ended signals include both the static or pass transistor logic families [9]. While complementary signal circuits include both the domino [10], NORA[11], and CVSL[12] logic families.

The static logic uses NMOS and PMOS transistor trees. These trees are never simultaneously active in steady state, and have no steady state power dissipation. The pass transistor logic family uses NMOS or NMOS/PMOS transistors for steering the input to the output, and an inverter for amplification. Gates in this logic family are similar to inverting muxes. The domino logic family gate is a pre-charge logic gate, where the internal node is precharged high and conditionally discharged. The internal node is buffered from the outside using an inverter. This is a monotonic[2] gate, so one needs to generate and propagate both the signal and its complement. NORA is similar to domino logic, except there is no inverter and one alternates between precharge and predischarge logic. It is a dense logic family that is faster than domino logic, because it has no inverter. It is not commonly used because of its poor noise margins. CVSL or cascode voltage switch logic uses two cross-coupled PMOS transistors, and it builds two NMOS trees that are connected to the PMOS transistors. The NMOS trees are similar to ECL differential steering trees. This logic family is extremely good for implementing complex gates such as parity tree. An expanded discussion about the merits and disadvantages of each logic family when implementing counters can be found in Song[13]

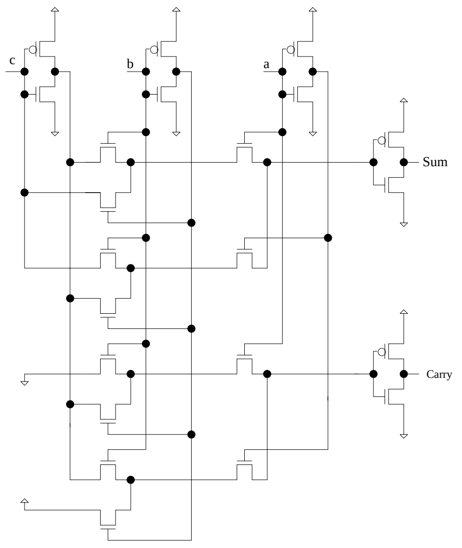
The circuits that were modeled are shown in figure 8. For the single-ended circuits, pass transistor logic family was chosen, and for the complementary signal circuits, domino logic was used.

6 Simulation

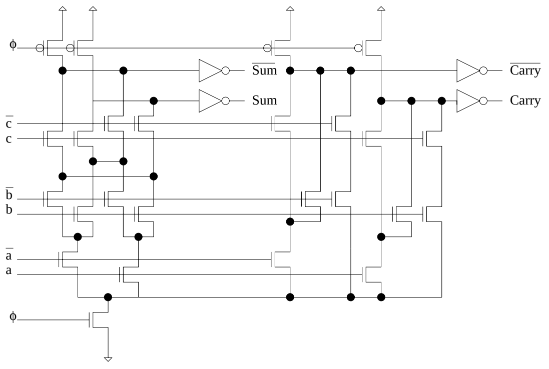
The circuits were simulated using HSPICE. They were simulated for an HP $0.8\mu m$ processes. The simulations are run for typical processing conditions at 25°C. The simulation includes the wire delays that are modeled using the Ersatzco [14] wire model. This model calculates the wire RC delay by placing half the wires capacitance on each side of the wires resistance. The capacitance is calculated using the parallel plate model, with fringing capacitance. This model has the advantage of being computationally simple, while still providing accurate results. The transistor models include an approximation of the gate and source capacitances that is calculated automatically by HSPICE.

The tree circuits were simulated, using two cases. The first case has the inputs for the two trees of the folded part of the parallelogram interleaved. The second case has them separated (non-interleaved). The first case has the advantage that it leads to a smaller circuit because it uses fewer Booth encoders. It also has the advantage that the Booth encoding is faster because there is less wire capacitance between the latch outputs and the

[2]All signals change in only one direction when evaluating.



Pass Transistor 3-2 Counter



Domino 3-2 Counter

Figure 8: Circuits Used

Booth encoder inputs. The second method has the advantage that adjacent counters in which the input of one is the output of the previous counter do not have to use a wiring track.

The delays are measured from the time the input is latched into the circuit by the system clock in the latches to the time the result becomes available at the output of the trees before the CPA.

7 Results

For simplicity two bit pitches where chosen to compare the various interconnection methods. The sizes of the various circuits used for each bit pitch is given in table 1.

Table 2 gives the area for each possible design. In this table for the type column, A refers to Array, interleaved is represented by the letter I, the first N refers to non-interleaved, the second N represent narrow, and W refers to wide. For the circuit used column S-E is single-ended and comp is complementary. From this table it is apparent that the areas needed by the designs are the same. This is because of the assumption that the design of the multiplier is part of a larger design. The only difference between the different designs is in the use of the available wiring tracks.

There is some lost silicon area that is not used in the Booth encoder columns. This lost area corresponds to the area in the Booth encoder column that is adjacent to the counters. This is shown in figure 3 where the lost space corresponds to the area above the counters. The reason that the wide bit pitches have a smaller area is because their 3-2 counters have a smaller length Thus, generating less wasted area. The table also shows that the Array and hybrid designs are the largest narrow designs. This is because the complementary circuits

Circuit	Bit pitch	Type	Length (μ)	Width (μ)
Latch	Narrow	Pass Trans	250	40
	Wide	Pass Trans	130	80
3-2 Counter	Narrow	Pass Trans	72	40
	Narrow	Domino	105	40
	Wide	Domino	55	80
4-2 Counter	Narrow	Pass Trans	145	40
	Wide	Domino	210	80
Booth Encoder	Narrow	Pass Trans	32	248
	Wide	Pass Trans	20	450
Booth Mux	Narrow	Pass Trans	32	40
	Wide	Pass Trans	20	80

Table 1: Subcell Circuit Sizes

Structure	Type	Circuit used	Length (mm)	Width (mm)	Area (mm^2)
Double Linear Arrays	A-N	Comp	4.26	2.970	12.6522
ZM Trees	I-N	S-E	3.40	2.725	9.2650
	N-N	S-E	3.40	2.970	10.0980
	I-W	Comp	1.18	5.220	6.1596
	N-W	Comp	1.18	5.670	6.6906
4-2 Trees	I-N	S-E	3.40	2.725	9.2650
	N-N	S-E	3.40	3.050	10.3700
	I-W	Comp	1.18	5.220	6.1596
	N-W	Comp	1.18	5.750	6.7850
OS Tree	I-N	S-E	3.40	2.725	9.2650
	N-N	S-E	3.40	2.970	10.0980
	I-W	Comp	1.18	5.220	6.1596
	N-W	Comp	1.18	5.670	6.6906
Hybrid	N-N	Comp	4.26	2.970	12.6522
High Order Trees	A-N	Comp	4.26	2.970	12.6522

Table 2: Area of Each Circuit

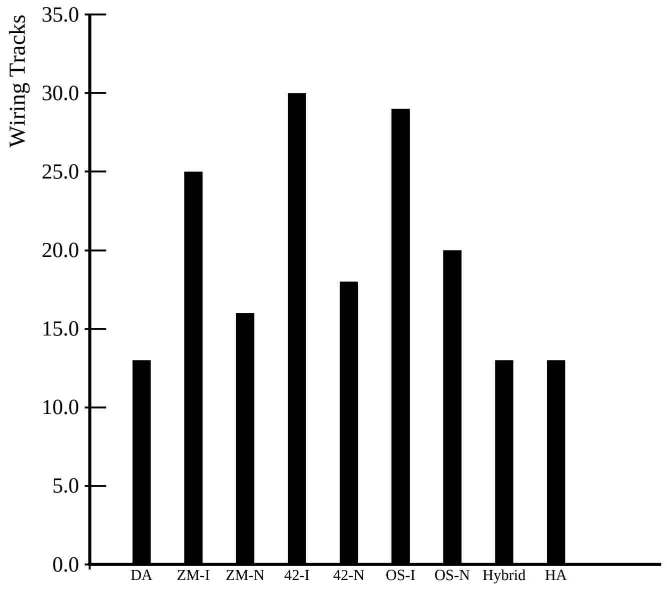


Figure 9: Minimum Number of wiring tracks needed to use Complementary Signal Circuits

used for the counters are larger than the single-ended circuits used for the counters.

The critical number of wiring tracks needed for each kind of the tree to use the complementary signal circuits is given by figure 9. The critical column for wire congestion for most topologies is not the critical path, but rather the hinge point where the folding occurred for the tree organizations. This is because there are two comparably sized partial products sets that must be reduced, and they share the wiring tracks.

From this figure non-interleaved placement of partial product rows allows the use of complementary signals before the interleaved method for all circuit topologies, since they require a smaller number of wiring tracks. The hybrid solution and the higher order trees require the smallest number of wiring tracks, because they use linear arrays that do not need any wiring tracks.

The results of the HSPICE simulations for the narrow bit pitch are summarized in figure 10 and in figure 11 which gives the results using the technology independent "fanout of 4" (FO4) metric. In these figures there is a difference between the overall delay and sum of the Booth and tree delays in the complementary circuits. This difference is due to the delay in the clock used to initiate the complementary circuit. This difference does occur in reality although it can be minimized by careful design.

For the narrow bit pitch the hybrid solution gives the fastest multiplier, since it is able to use the faster complementary signal circuits. The higher order arrays are also able to use complementary signals and need fewer levels of 3-2 counters than the hybrid scheme. However, they are slower than the hybrid solution because they have much longer wires.

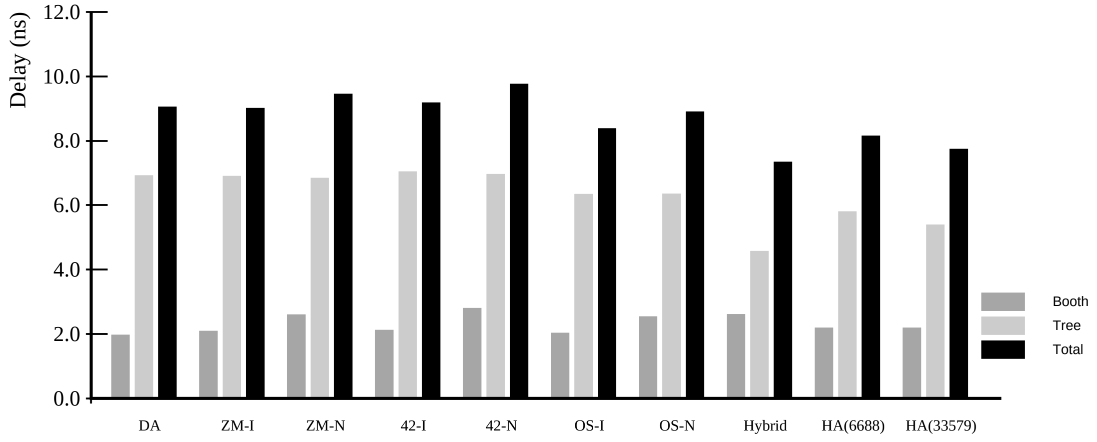


Figure 10: Narrow Bit Pitch Delay

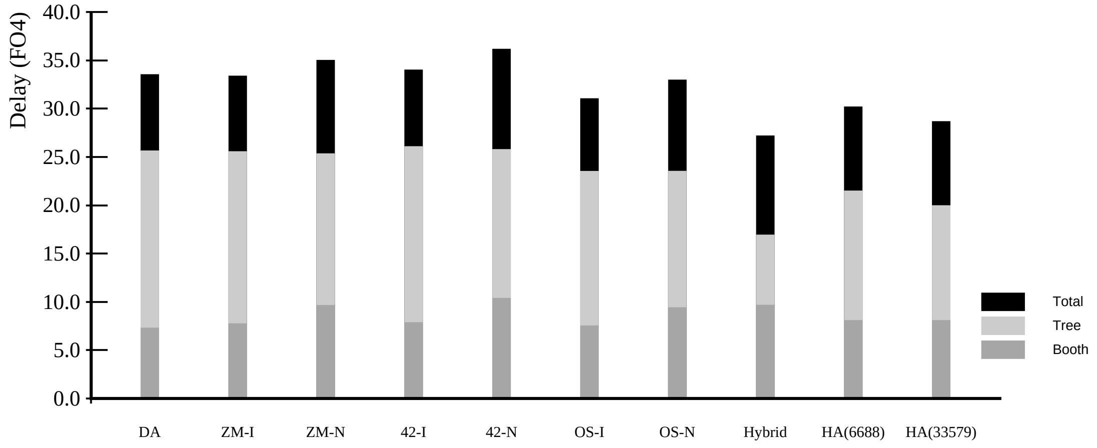


Figure 11: Narrow Bit Pitch Delay in “Fan Out of 4”

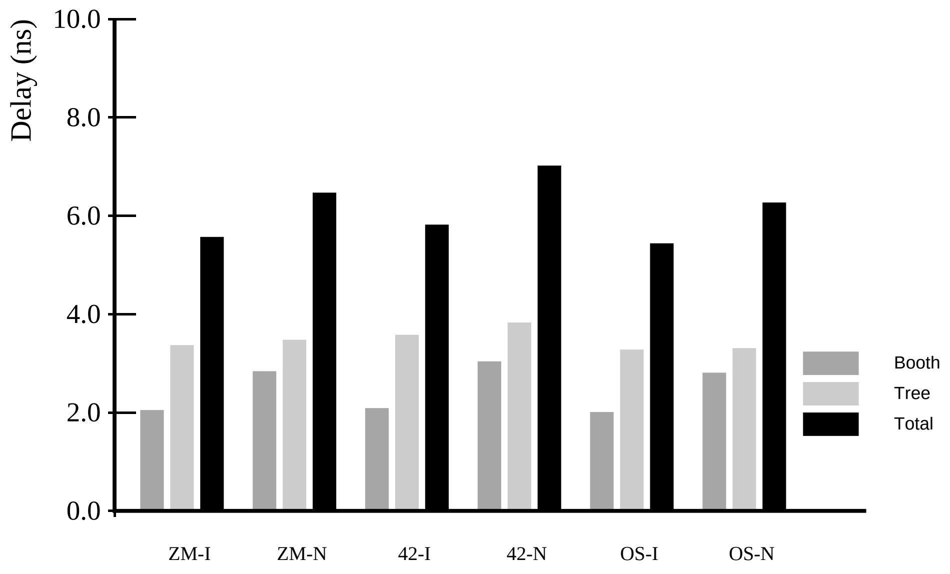


Figure 12: Wide Bit Pitch

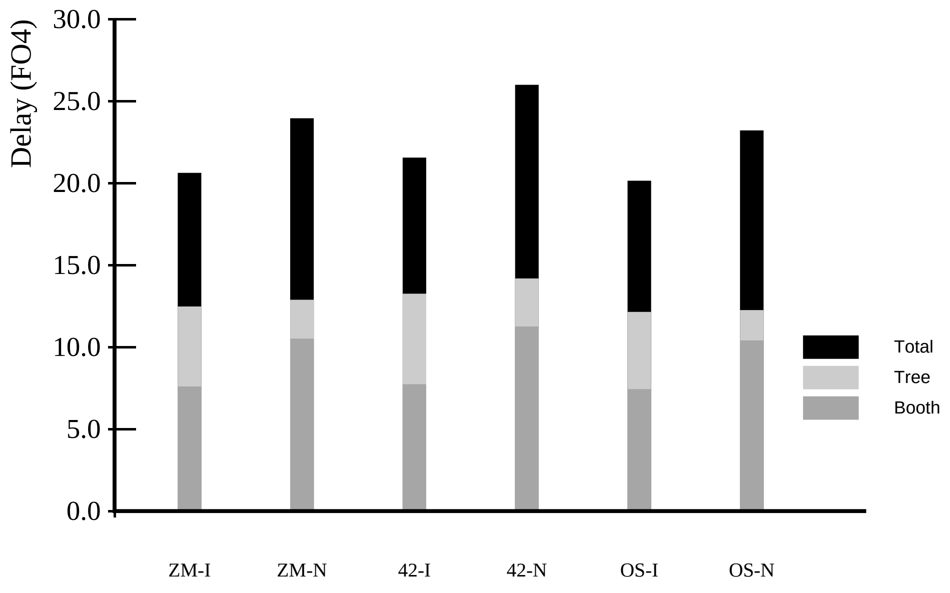


Figure 13: Wide Bit Pitch Delay in “Fan Out of 4”

They should give better performance for the ECL circuits where wire capacitance is not such a dominating factor. Higher order arrays also have the advantage of being much more regular than the hybrid solution. The advantage of the hybrid topology and higher order arrays remains until the bit pitch is sufficient to allow the use of a tree with complementary circuits.

Surprisingly, for the narrow bit pitch the double-linear array, which is extremely regular, gives comparable performance to the tree topologies that can only use single-ended signal circuits.

The results for the wide bit pitch are summarized in figure 12 and in figure 13. The wide bit pitch has better performance than the narrow bit pitch, since the wide bit pitch allows the use of complementary circuits with any topology. However, this wide bit pitch is not commonly used because of its large value (80μ). The wide bit pitch circuits also have a large difference in their aspect ratio, in that they are much wider than they are long.

Between the tree topologies, the OS tree gives the best performance. This is due to two factors. The first factor is that it is able to achieve the same number of levels as the wallace trees. The second factor is that it has an even distribution of wire lengths. The ZM tree provides better performance than the 4-2 tree, even though they have the same number of counter levels, because it has a more even distribution of the wire lengths between the counters.

As a general consideration the interleaved topology multipliers are faster than the non-interleaved multipliers as they speed up Booth encoding by using less capacitance between the Booth encoders and the Latch outputs. However, interleaved topologies require more wiring tracks, so it is possible, if the bit pitch had only 20 tracks, the non-interleaved topologies would be able to use complementary signal circuits, while the interleaved topology can only use single-ended circuits.

8 Conclusion

The number of 3-2 counters in the critical path of the multiplier is not the deciding factor in determining the latency of a multiplier. The topology chosen is at least as important in determining the latency. Therefore, to build the fastest possible circuits one has to trade-off the number of 3-2 counter levels with the number of available wiring tracks per bit pitch.

The paper introduced the hybrid structure and higher order arrays which offer alternatives, such that complementary signals can be used for narrow bit pitches. The hybrid structure and higher order arrays performance is comparable and which one is better should depend on the implementation and available bit pitches. When wires are not a limitation OS trees give the best performance. ZM trees give the best performance only in those situations when they are able to use complementary logic and OS trees can not.

The non-interleaved organization of partial product rows allows the use of complementary signals before the interleaved organization of the partial product rows. However the non-interleaved organization requires a larger area.

These results are not only unique to CMOS circuits, but they can be generalized to ECL circuits, in the Differential vs. Single-Ended issue.

9 Acknowledgement

The authors wish to thank N. Quach for his assistance throughout this work.

References

[1] Stuart Oberman and Michael Flynn, "Design issues in Floating Point Division", *Technical Report: CSL-TR-94-647* Stanford University.

[2] An American National Standard, "IEEE Standard for Floating Point Arithmetic", *ANSI/IEEE standard 754-1985*

[3] O.L. McSorley, "High Speed Arithmetic in Binary Computers", *Proceedings of the IRE*, 49(1), pp. 67-91, Jan 1961.

[4] J. L. Hennessy and D. A. Patterson, "Computer Architecture, A Quantitative Approach", *Morgan-Kaufmann*, pp. A44-A48, 1990.

[5] M. Santoro, "Design and Clocking of VLSI Multipliers", *Ph.D. Thesis, Stanford University*, Oct. 1989

[6] D. Zuras and W. McAllister, "Balanced Delay Trees and Combinatorial Division in VLSI,"*IEEE J. Solid-State Circuits*, vol SC-21, No.5, pp. 814-819, Oct. 1986

[7] C. S. Wallace, "A Suggestion for a Fast Multiplier", *IEEE Trans. Electronic Computers*, pp. 14-17, Feb. 1964.

[8] Z. Mou and F. Jutand, "A Class of Close to Optimum Adder Trees allowing Regular and Compact Layout", *IEEE Trans. Computers*, pp. 251-254, 1990.

[9] C. A.Mead and L. A. Conway, "Introduction to VLSI systems", Reading, MA, Addison Wesley, 1980.

[10] R. Krambeck, C. Lee and H-F. Lew, "High Speed Compact Circuits with CMOS", *IEEE Journal of Solid State*, pp. 614-618, June 1982.

[11] N. Goncalves and H. DeMan, "NORA: A Race Free Dynamic CMOS technique for Pipelined Logic Structures", *IEEE Journal of Solid State Circuits*, Vol SC-18, No.3, pp. 261-266, June 1983.

[12] L. Heller, W. Griffin, J. Davis and N. Thomas, "Cascode Voltage Switch Logic: A differential CMOS Logic Family", *IEEE International Solid State Conference*, pp. 16-19, Feb. 1984.

[13] P. Song, "New circuit and structures for combinatorial multipliers", *Ph.D. Thesis Stanford University*, 1993.

[14] M. Horowitz, EE371 Class notes.